Code

Trusting No One, Getting Nowhere: Soft Policy and the Janus-Faced Nature of Social Capital in Evacuation Networks

Replication Code

Timothy Fraser, Northeastern University

February 19, 2021

This study tests what factors drive evacuation. We estimate evacuation using data from Facebook Data for Good. Their project tallies up the number of Facebook users who were identified to be in the disaster hit area consistently two weeks prior to the disaster. Then, they tally up how many people went from any neighborhood in that region to any other neighborhood. Unfortunately, this data is measured at the neighborhood level, using special polgyons. This is not a level joinable with census data, and by data sharing agreement, the researchers cannot share the original raw data without permission from Facebook. So, to remedy both these issues, we aggregated everything up from the neighborhood level to the census tract level. This is described in detail in fb_network_maker_code.Rmd. While we can't share the original base data, located in the fb_data folder, we can share all other data for replication, which we describe below!

0. Packages

First, let's load some packages!

Hide

```
library(tidyverse) # for tidy data manipulation
# Using the most recent version of dplyr, we can bind_rows() for sf objects
# Download the most recent version here:
# library(remotes)
# remotes::install_github("tidyverse/dplyr")
library(dplyr)
library(dtplyr) # for dplyr functions with data.table, the speedy data manipulation pack
age
library(ggeffects)
# GIS packages
library(sf) # for tidy spatial data
library(rgdal) # for spatial operations
library(tigris) # for accessing census boundaries
library(tidycensus) # for downloading census data
library(censusapi) # for downloading census data
# Networks Packages
library(tidygraph)
library(ggraph)
#remotes::install_github("luukvdmeer/sfnetworks")
library(sfnetworks)
library(gstat)
```

1. Building the Network

1.1 Gather Social Vulnerability Index

Hide

```
# We downloaded the CDC's Social Vulnerability Index at the census tract level
# for the year 2018 from the following location on Sun, November 22, 2020:
# https://www.atsdr.cdc.gov/placeandhealth/svi/data_documentation_download.html

# First, resave into .rds format, for quicker access
read_csv("raw_data/covariates/SVI2018_US.csv") %>%
    select(fips = FIPS, state = ST_ABBR,
         # Grab the SVI
         svi = RPL_THEMES,
         # Grab the four subindices
         svi_socioeconomic = RPL_THEME1, # Socioeconomic – RPL_THEME1
         svi_household_disability = RPL_THEME2, #Household Composition & Disability – RP
L_THEME2
         svi_minority = RPL_THEME3, #Minority Status & Language – RPL_THEME3
         svi_housing_tranport = RPL_THEME4 # Housing Type & Transportation – RPL_THEME4
         )  %>%
  # -999 means no data, so let's remove those
    mutate_at(vars(svi:svi_housing_tranport),
            funs(if_else(. == -999, NA_real_, .))) %>%
  # and save in a more comfortable file format
  saveRDS("raw_data/covariates/svi_tract.rds")
```

County subdivision

Hide

```
# Load packages
library(tidyverse)
library(tigris)
library(sf)
library(rgdal)

# Aggregate census tract measures to zipcode level
aggregate_csub = function(MYSTATE){

  print(MYSTATE)

  # Import 2018 SVI for given state
  tracts <- tigris::tracts(state = MYSTATE, year = 2010) %>%
    st_as_sf() %>%
    select(tracts = GEOID10, geometry) %>%
    left_join(by = c("tracts" = "fips"),
              y = read_rds("raw_data/covariates/svi_tract.rds"))


  # For each state, import zipcodes from TIGRIS
  csub <- tigris::county_subdivisions(state = MYSTATE, year = 2010) %>%
    st_as_sf() %>%
    select(csub = GEOID10, geometry)

  # Join together the census tract measures into the zipcode
  csub %>%
    st_join(tracts) %>%
    as.data.frame() %>%
    group_by(csub) %>%
    summarize(svi = mean(svi, na.rm = TRUE),
              svi_socioeconomic = mean(svi_socioeconomic, na.rm = TRUE),
              svi_household_disability = mean(svi_household_disability, na.rm = TRUE),
              svi_minority = mean(svi_minority, na.rm = TRUE),
              svi_housing_tranport = mean(svi_housing_tranport, na.rm = TRUE)) %>%
    ungroup() %>%
    return()
}

data.frame(state = state.abb) %>%
  split(.$state) %>%
  map_dfr(~aggregate_csub(.$state), .id = "state") %>%
  saveRDS("raw_data/covariates/svi_csub.rds")
```

1.2 Census

Hide

```
library(tidyverse)
library(tidycensus)
library(censusapi)
# Load API key
tidycensus::census_api_key("97b29053a89e5396ef806bf7f4fae05b7387fabd")

#vars <- load_variables(year = 2018, dataset = "acs5", cache = TRUE) %>%
#  mutate_at(vars(label, concept), funs(tolower(.)))


get_census = function(state){
  get_acs(
    year = 2019,state = state,
    geography = "county subdivision",
    variables = c("B01003_001E", # Total Population
                  # Age
                  "B01001_020",
                  "B01001_021",
                  "B01001_022",
                  "B01001_023",
                  "B01001_024",
                  "B01001_025",
                  "B01001_044",
                  "B01001_045",
                  "B01001_046",
                  "B01001_047",
                  "B01001_048",
                  "B01001_049",
                  "B01001_026", # Gender

                  "B02001_002E", #Race similarity: Estimate!!Total!!White alone
                  "B02001_003E", #Race similarity: Estimate!!Total!!Black or African Ame
rican alone
                  "B02001_004E", #Race similarity: Estimate!!Total!!American Indian and
 Alaska Native alone
                  "B02001_005E", #Race similarity: Estimate!!Total!!Asian alone
                  "B02001_006E", #Race similarity: Estimate!!Total!!Native Hawaiian and
 Other Pacific Islander alone
                  "B03001_003E", # Hispanic or Latino
                  "B19083_001E", #Income inequality: Estimate!!Gini Index)
                  "B06009_004", # Population with some college or higher
                  "B23025_003E", #Employment equality: Estimate!!Total!!In labor force!!
Civilian labor force
                  "B23025_005E", #Employment equality: Estimate!!Total!!In labor force!!
Civilian labor force!!Unemployed

                  "B19013_001", # Median Household Income
                  "B25105_001", # Median Monthly Housing Cost
                  "B08128_006E", #Local government linkage: Estimate!!Total!!Local gover
nment workers
                  "B08128_007E", #State government linkage: Estimate!!Total!!State gover
nment workers
                  "B08128_008E"), #Federal government linkage: Estimate!!Total!!Federal
```

```
 government workers
    survey = "acs5") %>%
    select(geoid = GEOID, variable, estimate) %>%
    mutate(variable = variable %>% dplyr::recode(
      "B01003_001" = "pop", # Total Population
      # Age
      "B01001_020" = "pop_age_65_66_male",
      "B01001_021" = "pop_age_67_69_male",
      "B01001_022" = "pop_age_70_74_male",
      "B01001_023" = "pop_age_75_79_male",
      "B01001_024" = "pop_age_80_84_male",
      "B01001_025" = "pop_age_85_over_male",

      "B01001_044" = "pop_age_65_66_female",
      "B01001_045" = "pop_age_67_69_female",
      "B01001_046" = "pop_age_70_74_female",
      "B01001_047" = "pop_age_75_79_female",
      "B01001_048" = "pop_age_80_84_female",
      "B01001_049" = "pop_age_85_over_female",

      "B01001_026" = "pop_women", # Gender
      "B02001_002" = "pop_white",
      "B02001_003" = "pop_black", #Race similarity: Estimate!!Total!!Black or African Am
erican alone
      "B02001_004" = "pop_natam", #Race similarity: Estimate!!Total!!American Indian and
Alaska Native alone
      "B02001_005" = "pop_asian", #Race similarity: Estimate!!Total!!Asian alone
      "B02001_006" = "pop_pacific", #Race similarity: Estimate!!Total!!Native Hawaiian a
nd Other Pacific Islander alone
      "B03001_003" = "pop_hisplat", # Hispanic or Latino
      "B06009_004" = "pop_some_college",
      "B19083_001" = "income_inequality", #Income inequality: Estimate!!Gini Index)

      "B23025_003" = "pop_labor_force", #Employment equality: Estimate!!Total!!In labor
 force!!Civilian labor force
      "B23025_005" = "pop_unemployed", #Employment equality: Estimate!!Total!!In labor f
orce!!Civilian labor force!!Unemployed

      "B19013_001" = "median_income",   #Estimate!!Median Household income (dollars)!
      "B25105_001" = "median_monthly_housing_cost",
      "B08128_006" = "employees_muni", #Local government linkage: Estimate!!Total!!Local
government workers
      "B08128_007" = "employees_state", #State government linkage: Estimate!!Total!!Stat
e government workers
      "B08128_008" = "employees_fed", #Federal government linkage: Estimate!!Total!!Fede
ral government workers
    )) %>%
    return()
}

c("FL", "GA", "AL") %>%
  map_dfr(~get_census(.)) %>%
  pivot_wider(id_cols = geoid,
              names_from = variable,
```

```
              values_from = estimate) %>%
  # Tally up population in this age category
  mutate(pop_age_65_plus = pop_age_65_66_male + pop_age_67_69_male +
         pop_age_70_74_male + pop_age_75_79_male +
         pop_age_80_84_male + pop_age_85_over_male +
         pop_age_65_66_female + pop_age_67_69_female +
         pop_age_70_74_female + pop_age_75_79_female +
         pop_age_80_84_female + pop_age_85_over_female) %>%
  # % percentage of population
  mutate_at(vars(
    pop_age_65_plus, pop_women,
    pop_white, pop_black, pop_natam,
    pop_asian, pop_pacific, pop_hisplat,
    pop_some_college,
    employees_muni, employees_state, employees_fed),
            funs(. / pop)) %>%
  mutate(pop_unemployed = pop_unemployed / pop_labor_force) %>%
  select(-c(pop_age_65_66_male, pop_age_67_69_male,
         pop_age_70_74_male, pop_age_75_79_male,
         pop_age_80_84_male, pop_age_85_over_male,
         pop_age_65_66_female, pop_age_67_69_female,
         pop_age_70_74_female, pop_age_75_79_female,
         pop_age_80_84_female, pop_age_85_over_female)) %>%
  saveRDS("raw_data/covariates/census.rds")


# Repeat for tracts

get_census = function(state){
  get_acs(
    year = 2019,state = state,
    geography = "tract",
    variables = c("B01003_001E", # Total Population
                  # Age
                  "B01001_020",
                  "B01001_021",
                  "B01001_022",
                  "B01001_023",
                  "B01001_024",
                  "B01001_025",
                  "B01001_044",
                  "B01001_045",
                  "B01001_046",
                  "B01001_047",
                  "B01001_048",
                  "B01001_049",
                  "B01001_026", # Gender

                  "B02001_002E", #Race similarity: Estimate!!Total!!White alone
                  "B02001_003E", #Race similarity: Estimate!!Total!!Black or African Ame
rican alone
                  "B02001_004E", #Race similarity: Estimate!!Total!!American Indian and
 Alaska Native alone
                  "B02001_005E", #Race similarity: Estimate!!Total!!Asian alone
```

```
                  "B02001_006E", #Race similarity: Estimate!!Total!!Native Hawaiian and
 Other Pacific Islander alone
                  "B03001_003E", # Hispanic or Latino
                  "B19083_001E", #Income inequality: Estimate!!Gini Index)
                  "B06009_004", # Population with some college or higher
                  "B23025_003E", #Employment equality: Estimate!!Total!!In labor force!!
Civilian labor force
                  "B23025_005E", #Employment equality: Estimate!!Total!!In labor force!!
Civilian labor force!!Unemployed

                  "B19013_001", # Median Household Income
                  "B25105_001", # Median Monthly Housing Cost
                  "B08128_006E", #Local government linkage: Estimate!!Total!!Local gover
nment workers
                  "B08128_007E", #State government linkage: Estimate!!Total!!State gover
nment workers
                  "B08128_008E"), #Federal government linkage: Estimate!!Total!!Federal
 government workers
    survey = "acs5") %>%
    select(geoid = GEOID, variable, estimate) %>%
    mutate(variable = variable %>% dplyr::recode(
      "B01003_001" = "pop", # Total Population
      # Age
      "B01001_020" = "pop_age_65_66_male",
      "B01001_021" = "pop_age_67_69_male",
      "B01001_022" = "pop_age_70_74_male",
      "B01001_023" = "pop_age_75_79_male",
      "B01001_024" = "pop_age_80_84_male",
      "B01001_025" = "pop_age_85_over_male",

      "B01001_044" = "pop_age_65_66_female",
      "B01001_045" = "pop_age_67_69_female",
      "B01001_046" = "pop_age_70_74_female",
      "B01001_047" = "pop_age_75_79_female",
      "B01001_048" = "pop_age_80_84_female",
      "B01001_049" = "pop_age_85_over_female",

      "B01001_026" = "pop_women", # Gender
      "B02001_002" = "pop_white",
      "B02001_003" = "pop_black", #Race similarity: Estimate!!Total!!Black or African Am
erican alone
      "B02001_004" = "pop_natam", #Race similarity: Estimate!!Total!!American Indian and
Alaska Native alone
      "B02001_005" = "pop_asian", #Race similarity: Estimate!!Total!!Asian alone
      "B02001_006" = "pop_pacific", #Race similarity: Estimate!!Total!!Native Hawaiian a
nd Other Pacific Islander alone
      "B03001_003" = "pop_hisplat", # Hispanic or Latino
      "B06009_004" = "pop_some_college",
      "B19083_001" = "income_inequality", #Income inequality: Estimate!!Gini Index)

      "B23025_003" = "pop_labor_force", #Employment equality: Estimate!!Total!!In labor
 force!!Civilian labor force
      "B23025_005" = "pop_unemployed", #Employment equality: Estimate!!Total!!In labor f
orce!!Civilian labor force!!Unemployed
```

```
      "B19013_001" = "median_income",   #Estimate!!Median Household income (dollars)!
      "B25105_001" = "median_monthly_housing_cost",
      "B08128_006" = "employees_muni", #Local government linkage: Estimate!!Total!!Local
government workers
      "B08128_007" = "employees_state", #State government linkage: Estimate!!Total!!Stat
e government workers
      "B08128_008" = "employees_fed", #Federal government linkage: Estimate!!Total!!Fede
ral government workers
    )) %>%
    return()
}

c("FL", "GA", "AL") %>%
  map_dfr(~get_census(.)) %>%
  pivot_wider(id_cols = geoid,
              names_from = variable,
              values_from = estimate) %>%
  # Tally up population in this age category
  mutate(pop_age_65_plus = pop_age_65_66_male + pop_age_67_69_male +
         pop_age_70_74_male + pop_age_75_79_male +
         pop_age_80_84_male + pop_age_85_over_male +
         pop_age_65_66_female + pop_age_67_69_female +
         pop_age_70_74_female + pop_age_75_79_female +
         pop_age_80_84_female + pop_age_85_over_female) %>%
  # % percentage of population
  mutate_at(vars(
    pop_age_65_plus, pop_women,
    pop_white, pop_black, pop_natam,
    pop_asian, pop_pacific, pop_hisplat,
    pop_some_college,
    employees_muni, employees_state, employees_fed),
            funs(. / pop)) %>%
  mutate(pop_unemployed = pop_unemployed / pop_labor_force) %>%
  select(-c(pop_age_65_66_male, pop_age_67_69_male,
         pop_age_70_74_male, pop_age_75_79_male,
         pop_age_80_84_male, pop_age_85_over_male,
         pop_age_65_66_female, pop_age_67_69_female,
         pop_age_70_74_female, pop_age_75_79_female,
         pop_age_80_84_female, pop_age_85_over_female)) %>%
  saveRDS("raw_data/covariates/census_tract.rds")
```

1.3 Partisanship

County

Second, let's download data on shares of the vote that went to specific parties between 2000 and 2016. (This study's COVID outcome data occurs prior to the 2020 presidential election.)

Hide

```
# ImportCounty Presidential Election Results 2000-2016 Data
# from MIT Elections Data & Science Lab
# https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/VOQCHQ

read_csv("raw_data/covariates/MIT_countypres_2000_2016.csv") %>%
  filter(office == "President") %>%
  filter(party %in% c("democrat", "republican")) %>%
  rename(fips = FIPS) %>%
  # Convert unique county id FIPS code from numeric to a five digit character code
  mutate(fips = str_pad(fips, width = 5, side = "left", pad = "0")) %>%
  # Aggregate voteshare by county, party, and year
  group_by(fips, party, year) %>%
    # Create the share of votes that went to each party per county-year election
  summarize(voteshare = sum(candidatevotes, na.rm = TRUE) / sum(totalvotes, na.rm = TRUE
)) %>%
    # Pivot wider
  pivot_wider(
    id_cols = c(fips),
    names_from = c(party, year),
    names_sep = "_",
    values_from = voteshare) %>%
  write_csv("raw_data/covariates/county_elections.csv")
```

Precinct

Hide

```
# Download the MIT Election Lab's precint level returns
# https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/LYWX3D

#load("raw_data/covariates/presidential_precincts_2016.rda")
#df1 %>%
#  saveRDS("raw_data/covariates/presidential_precincts_2016.rds")
#remove(df1)

#read_rds("raw_data/covariates/presidential_precincts_2016.rds") %>% head()
read_rds("raw_data/covariates/presidential_precincts_2016.rds") %>%
  filter(state_postal %in% c("AL", "FL", "GA")) %>%
  filter(office == "US President") %>%
  select(state = state_postal, fips = county_fips,
         precinct, candidate, party, votes, mode) %>%
    mutate(fips = str_pad(fips, width = 5, side = "left", pad = "0")) %>%
  # Tally total votes per candidate across all modes of voting (eg. early vote, absente
e, in person)
  group_by(state, fips, precinct, candidate) %>%
  summarize(votes = sum(votes, na.rm = TRUE)) %>%
  ungroup() %>%
  # Tally total votes
  group_by(state, fips, precinct) %>%
  mutate(total = sum(votes)) %>%
  ungroup() %>%
  mutate(candidate = candidate %>% na_if("") %>% dplyr::recode(
    "[Write-in]" = "Write-In",
    .missing = "Other")) %>%
# Pivot wider
  pivot_wider(
    id_cols = c(state, fips, precinct, total),
    names_from = candidate,values_from = votes, values_fill = list(votes = 0)) %>%
  mutate(dem_percent = `Hillary Clinton` / total * 100,
         rep_percent = `Donald Trump` / total * 100) %>%
  mutate_at(vars(dem_percent, rep_percent), funs(if_else(is.na(total), 0, as.numeric
(.))) ) %>%
  select(state, fips, precinct, total, dem_percent, rep_percent) %>%
  saveRDS("raw_data/covariates/dorian_precinct_results_2016.rds")
```

Hide

```
# Download 2012 precinct maps (virtually identical to 2016)
get_shape = function(STATE){
  tigris::voting_districts(state = STATE) %>%
    st_as_sf() %>%
    st_transform(CRS(paste0("+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs"))) %>%
    select(geoid = GEOID10, name = NAME10, full_name = NAMELSAD10, geometry) %>%
    return()
}
c("FL", "GA", "AL") %>%
  map(~get_shape(.)) %>%
  dplyr::bind_rows() %>%
  mutate(geoid = str_pad(geoid, width = 8, side = "left", pad = "0")) %>%
  saveRDS("shapes/precincts.rds")

# First, let's grab a data.frame full of the precincts we want to join in
shape <- read_rds("shapes/precincts.rds") %>%
  # creating a single ID with all identifiers jammed in, spaceless, lowercase
  mutate(id = paste(geoid, name, sep = "") %>% str_remove_all(" |/") %>% tolower()) %>%
  as.data.frame() %>%
  select(geoid, name, id)

# Let's grab a single row from the outcome data,
# and try to identify a match in the shape data
get_match = function(sample){

  print(sample$row_id)

  shape %>%
    filter(str_detect(id, sample$precinct) & str_detect(id, sample$fips)) %>%
    mutate(num = sample$row_id) %>%
    return()
}

# Second, let's grab the dataset of data with identifiers we want to join into the shape
s
dat <- read_rds("raw_data/covariates/dorian_precinct_results_2016.rds") %>%
  mutate(precinct = str_remove_all(precinct, " | /") %>% tolower()) %>%
  mutate(row_id = 1:n()) %>%
  split(.$row_id) %>%
  map_dfr(~get_match(.))


read_rds("raw_data/covariates/dorian_precinct_results_2016.rds") %>%
  mutate(row_id = 1:n()) %>%
  left_join(by = c("row_id" = "num"), y = dat) %>%
  select(geoid, row_id, name, state, fips, precinct, total, dem_percent, rep_percent) %
>%
  saveRDS("raw_data/covariates/precincts_joined.rds")
```

Hide

```
aea <- "+proj=aea +lat_1=20 +lat_2=60 +lat_0=40 +lon_0=-96 +x_0=0 +y_0=0 +ellps=GRS80 +d
atum=NAD83 +units=m +no_defs"


states <- read_rds("shapes/counties.rds") %>%
  #  st_transform(crs = aea) %>%
  group_by(state_fips = str_sub(geoid, 1,2)) %>%
  summarize(geometry = st_union(geometry))

shapes <- read_rds("shapes/precincts.rds") %>%
  #st_transform(crs = aea) %>%
  left_join(by = "geoid",
            y = read_rds("raw_data/covariates/precincts_joined.rds") %>%
              select(geoid, total, dem_percent, rep_percent)) %>%
  st_join(states) %>%
  filter(!is.na(state_fips))

ggplot() +
  geom_sf(data = states, fill = NA, color = "darkgrey", size = 5) +
  geom_sf(data = shapes,
          mapping = aes(fill = dem_percent), color = "white", size = 0.1) +
  scale_fill_gradient2(low = "#DC267F", high = "#648FFF",
                       midpoint = 50, mid = "white",na.value = "grey") +
  geom_sf(data = states, fill = NA, color = "black", size = 1) +
  theme_void(base_size = 14) +
  theme(legend.position = c(0.3, 0.4)) +
  labs(fill = "(%) Voted Democrat\nin 2016 Election\nfor President") +
  ggsave("viz/fig_B5_precincts.png", dpi = 500, width = 5, height = 5)
```

Interpolate

Hide

```
# Now, let's interpolate these
aea <- "+proj=aea +lat_1=20 +lat_2=60 +lat_0=40 +lon_0=-96 +x_0=0 +y_0=0 +ellps=GRS80 +d
atum=NAD83 +units=m +no_defs"

# Get the area within our study region
states <- read_rds("shapes/counties.rds") %>%
  st_transform(crs = aea) %>%
  group_by(state_fips = str_sub(geoid, 1,2)) %>%
  summarize(geometry = st_union(geometry))

# Format precinct data into a shapefile
read_rds("shapes/precincts.rds") %>%
  st_transform(crs = aea) %>%
  left_join(by = "geoid",
            y = read_rds("raw_data/covariates/precincts_joined.rds") %>%
              select(geoid, total, dem_percent, rep_percent)) %>%
  st_join(states) %>%
  filter(!is.na(state_fips)) %>%
  saveRDS("raw_data/covariates/precincts_data.rds")


# Import census tracts
tracts <- read_rds("shapes/tracts.rds") %>%
  st_transform(crs = aea)

# The shapefile is measured in meters,
# so when we make the fishnet cellsize 5000,
# that means 10,000 m (10 km).

tracts %>%
  st_make_grid(cellsize = 10000, what = "polygons") %>%
  st_as_sf() %>%
  mutate(id = 1:n()) %>%
  select(id, geometry = x) %>%
  # Now join in the states
  st_join(states) %>%
  # Zoom into just grid cells within our study region
  filter(!is.na(state_fips)) %>%

  saveRDS("raw_data/covariates/precinct_fishnet.rds")

# Import fishnet grid
nodes <- read_rds("raw_data/covariates/precinct_fishnet.rds")

# Import precinct data
dat <- read_rds("raw_data/covariates/precincts_data.rds")

# Make a grid of shapes from the polygon
m <- gstat::gstat(id = "id", formula = dem_percent ~ 1,
                  data = dat %>%
                    na.omit() %>%
                    as("Spatial"),
                  set = list(idp = 2))
```

```
# Generate predictions
predict(m, nodes) %>%
  dplyr::select(predicted = id.pred, geometry) %>%
  # Add in a unique id from the original grid cells
  mutate(id = nodes$id) %>%
  saveRDS("raw_data/covariates/precincts_predicted.rds")


grid <- read_rds("raw_data/covariates/precincts_predicted.rds") %>%
  st_as_sf() %>%
  ungroup() %>%
  as.data.frame() %>%
  select(-geometry)

# Import fishnet grid
nodes <- read_rds("raw_data/covariates/precinct_fishnet.rds") %>%
  left_join(by = "id", y = grid)

remove(grid)

# Average predictions among tracts
tracts %>%
  st_join(nodes) %>%
  as.data.frame() %>%
  select(-geometry) %>%
  group_by(geoid) %>%
  summarize(dem_percent = mean(predicted, na.rm = TRUE)) %>%
  ungroup() %>%
  saveRDS("raw_data/covariates/precincts_tracts.rds")

# Average predictions among divisions
read_rds("shapes/csub.rds") %>%
  st_transform(crs = aea) %>%
  st_join(nodes) %>%
  as.data.frame() %>%
  select(-geometry) %>%
  group_by(geoid) %>%
  summarize(dem_percent = mean(predicted, na.rm = TRUE)) %>%
  ungroup() %>%
  saveRDS("raw_data/covariates/precincts_csub.rds")


# Join votes into tracts
votes <- tracts %>%
  left_join(by = "geoid",
            y = read_rds("raw_data/covariates/precincts_tracts.rds"))

# Finally, let's visualze this pattern of Democratic votes
ggplot() +
  geom_sf(data = states, fill = NA, color = "darkgrey", size = 5) +
  geom_sf(data = votes,
          mapping = aes(fill = dem_percent), color = NA) +
  geom_sf(data = counties, fill = NA, color = "white", size = 0.1) +
```

```
  geom_sf(data = states, fill = NA, color = "black", size = 1) +
  theme_void(base_size = 14) +
   scale_fill_gradient2(low = "#DC267F", high = "#648FFF",
                        midpoint = 50, mid = "white",na.value = "grey") +
 theme(plot.caption = element_text(hjust = 0.5),
       legend.position = c(0.3, 0.4)) +
  labs(fill = "(%) Votes for Democrat\n2016 Election\nfor President",
       caption = "Spatially interpolated across 10 km grid cells\nand averaged by census
tracts (n = 4416).") +
  ggsave("viz/votes_tracts.png", dpi = 500, width = 5, height = 5)



# Join votes into division
votes <- read_rds("shapes/csub.rds") %>%
  st_transform(crs = aea) %>%
  left_join(by = "geoid",
            y = read_rds("raw_data/covariates/precincts_csub.rds"))

counties <- read_rds("shapes/counties.rds")

# Finally, let's visualze this pattern of Democratic votes
ggplot() +
  geom_sf(data = states, fill = NA, color = "darkgrey", size = 5) +
  geom_sf(data = votes,
          mapping = aes(fill = dem_percent), color = "grey", size = 0.1) +
  geom_sf(data = counties, fill = NA, color = "white", size = 0.2) +
  geom_sf(data = states, fill = NA, color = "black", size = 1) +
  theme_void(base_size = 14) +
   scale_fill_gradient2(low = "#DC267F", high = "#648FFF",
                        midpoint = 50, mid = "white",na.value = "grey") +
 theme(plot.caption = element_text(hjust = 0.5),
       legend.position = c(0.3, 0.4)) +
  labs(fill = "(%) Votes for Democrat\n2016 Election\nfor President",
       caption = "Spatially interpolated across 10 km grid cells\nand averaged by county
subdivision (n = 427).") +
  ggsave("viz/fig_B7_votes_csub.png", dpi = 500, width = 5, height = 5)


# Finally, let's visualze this pattern of Democratic votes as tiles.
ggplot() +
  geom_sf(data = states, fill = NA, color = "darkgrey", size = 5) +
  geom_sf(data = nodes,
          mapping = aes(fill = predicted), color = "white", size = 0.1) +
  #geom_sf(data = counties, fill = NA, color = "grey", size = 0.1) +
  geom_sf(data = states, fill = NA, color = "black", size = 1) +
  theme_void(base_size = 14) +
   scale_fill_gradient2(low = "#DC267F", high = "#648FFF",
                        midpoint = 50, mid = "white",na.value = "grey") +
 theme(plot.caption = element_text(hjust = 0.5),
       legend.position = c(0.3, 0.4)) +
  labs(fill = "(%) Votes for Democrat\n2016 Election\nfor President",
       caption = "Spatially interpolated across 10 km grid cells (n = 2341).") +
  ggsave("viz/fig_B6_votes_fishnet.png", dpi = 500, width = 5, height = 5)
```

Hide

```
rm(list = ls())
```

1.4 FEMA

Additionally, the levels of soft and hard infrastructure investment in each community could play a big role in evacuation. For example, if FEMA has spent lots of money developing a civilian community response team, that could really help with evacuation. Or if they have spent considerable funds in hazard mitigation, that could help as well. Let’s mine FEMA’s excellent geocoded data for these kinds of information.

CERT (Community Emergency Response Teams)

Let’s import FEMA’s Community Emergency Response Team dataset Found here: https://www.fema.gov/sites/default/files/2020-08/fema_community-emergency-response-team_09-18-2015_0.xlsx (https://www.fema.gov/sites/default/files/2020-08/fema_community-emergency-response-team_09-18-2015_0.xlsx) And at this site: https://www.fema.gov/about/openfema/data-sets/community-emergency-response-team-cert-dataset (https://www.fema.gov/about/openfema/data-sets/community-emergency-response-team-cert-dataset)

Hide

```
dat <- readxl::read_excel("raw_data/covariates/fema/fema_community-emergency-response-te
am_09-18-2015_0.xlsx", sheet = 3) %>%
  # Simplify column names
    magrittr::set_colnames(value = names(.) %>% tolower() %>%
                             str_replace_all(" |[(]|[)]|[/]|[,]|[.]", "_") %>%
                               str_replace_all("__", "_")) %>%
  mutate(id = 1:n()) %>%
  select(id, name = name_of_cert_program, state = cert_program_state,
         city = cert_program_city, zip = cert_program_zip,
         housing_org = sponsoring_housing_organization,
         jurisdiction = jurisdiction_type_county_local_or_state_,
         areas_served = zip_codes_or_counties_within_jurisdiction_only_for_local_and_cou
nty_cert_programs_,
         date_updated = date_cert_program_last_updated_mm_dd_yyyy_format_,
         trainings_per_year = `4_1__as_of_today's_date_on_average_how_many_times_per_yea
r_does_your_program_conduct_the_cert_basic_training_course?_type:_number`,
         classes_total = `5_1__number_of_cert_classes_that_have_graduated_since_your_pro
gram_started?_type:_number`,
         trainees_total = `6_1__number_of_individuals_who_have_completed_cert_basic_trai
ning_course_since_your_program_started?_type:_number`)

# First, we're going to tally Community Emergency Response Team Training Information.
# We'll tally four important measures.
# 1. How many/Is there a community response taining program in your community?
# 2. How many groups has it trained?
# 3. How many individuals per capita has it trained?
# 4. How many times a year does your program conduct trainings?

zip <- dat %>%
  # For each id,
  split(.$id) %>%
  # Identify any 5 digit zipcodes,
  # and keep just the unique ones per ids
  map_dfr(. %>% group_by(state) %>%
            summarize(zipcode = areas_served %>%
                        str_extract_all("[0-9]{5}") %>%
                        unlist() %>%
                        # Let's add in the original zipcode just in case
                        c(zip) %>% unique()), .id = "id") %>%
  mutate(zipcode = str_pad(zipcode, width = 5, side = "left", pad = "0")) %>%
  distinct()
# Now, extract counties
counties <- dat %>%
  split(.$id) %>%
  map_dfr(. %>% group_by(state) %>%
            summarize(county = areas_served %>%
                        # Remove zipcodes
                        str_remove_all(pattern = "[0-9]{5}") %>%
                        # What's left over should be the county; let's remove the label
                        str_remove_all(pattern = " [(]County[)]") %>%
                        str_remove_all(pattern = "[(]|[)]") %>%
                        str_remove_all(pattern = " Local") %>%
                        # Split into difference cells, and keep only the unique ones
```

```
                        str_split(pattern = ",") %>%
                        unlist() %>% str_trim(side = "both") %>%
                        na_if("") %>% unique()), .id = "id")

# Let's tally the number of individuals who have

# Get equal area conic projection
aea <- "+proj=aea +lat_1=20 +lat_2=60 +lat_0=40 +lon_0=-96 +x_0=0 +y_0=0 +ellps=GRS80 +d
atum=NAD83 +units=m +no_defs"

# Let's identify all census tracts that our CERT teams are involved in
# Grab zipcodes
zip_key <- read_rds("shapes/zipcodes.rds") %>%
  select(-area_land) %>%
  st_transform(crs = aea) %>%
  st_join(read_rds("shapes/csub.rds") %>%
            select(-name) %>%
            st_transform(crs = aea)) %>%
  as.data.frame() %>%
  select(zipcode, geoid) %>%
  distinct()

# Grab counties
county_key <- read_rds("shapes/counties.rds") %>%
  rename(fips = geoid) %>%
  st_transform(crs = aea) %>%
  st_join(read_rds("shapes/csub.rds") %>%
            select(-name) %>%
            st_transform(crs = aea)) %>%
  as.data.frame() %>%
  select(fips, name, geoid) %>%
  distinct()


# Last, build a key that identifies the
# relevant census tracts affected by each CERT program
bind_rows(
  # Get the list of zipcodes affected
  zip %>%
    left_join(by = "zipcode", y = zip_key) %>%
    filter(!is.na(geoid)),
  # Get the list of counties affected
  counties %>%
    mutate(county = tolower(county)) %>%
    filter(state %in% c("FL", "GA", "AL")) %>%
    left_join(by = c("county" = "name"),
              y = county_key %>%
                mutate(name = tolower(name))) %>%
    filter(!is.na(geoid))) %>%
  select(id, state, geoid) %>%
  distinct() %>%
  saveRDS("raw_data/covariates/fema/cert.rds")

remove(counties, county_key, zip_key, zip)
```

```
# I think the logical thing to do would be to divide
# the number of classes and trainees by the number of census tracts they serve.

read_rds("raw_data/covariates/fema/cert.rds") %>%
  left_join(by = c("id", "state"), y = dat %>% mutate(id = as.character(id))) %>%
  # Zoom into just the distinct pairs
  select(id, geoid, trainings_per_year, classes_total, trainees_total) %>%
  distinct() %>%
  # Calculate the total number of divisions per program
  group_by(id) %>%
  mutate(total_tracts = n()) %>%
  ungroup() %>%
 # filter(str_detect(geoid, "05017080400"))
  # Now for each tract, let's record...
  group_by(geoid) %>%
  summarize(
    # How many programs are operated in this census tract?
    programs = n(),
    # Is there at least one operated?
    cert = if_else(programs > 0, 1, 0),
    # Given the number of tracts the program operates in,
    # how many trainings per year does it run, per tract?
    trainings_per_year = sum(trainings_per_year, na.rm = TRUE) / mean(total_tracts, na.r
m = TRUE),
    # Given the number of tracts the program operates in,
    # how many classes has it run, per tract?
    classes = sum(classes_total) / mean(total_tracts, na.rm = TRUE),
    # Given the number of tracts the program operates in,
    # how many trainees has it trained, per tract?
    trainees = sum(trainees_total) / mean(total_tracts, na.rm = TRUE)) %>%
  ungroup() %>%
  saveRDS("raw_data/covariates/fema/cert_csub.rds")



# Download zipcode population from the American Community Survey
tidycensus::census_api_key("97b29053a89e5396ef806bf7f4fae05b7387fabd")
pop <- get_acs(
  year = 2019, state = c("FL", "AL", "GA"),
  geography = "county subdivision",
  variables = c("pop" = "B01003_001E")) %>%
  select(geoid = GEOID, pop = estimate)

# Finally, let's map it!
sub <- read_rds("shapes/csub.rds") %>%
  left_join(by = "geoid",
            y = read_rds("raw_data/covariates/fema/cert_csub.rds"))  %>%
  left_join(by = "geoid",
            y = pop) %>%
  mutate(trainees_per_thous = trainees / pop * 1000)

#dat %>%
#  filter(id %in% read_rds("raw_data/covariates/fema/cert.rds")$id) %>%
```

```
#  summarize(trainees_total = sum(trainees_total, na.rm = TRUE),
#         programs = n())

ggplot() +
  geom_sf(data = sub, fill = NA, color = "darkgrey", size = 5) +
  # Add a log-transformed trainees scale.
  # In a few cases, there were 0 people trained
  # We're going to a dd a tiny-but-reasonable constant (lowest observed divided by 2)
  # So that this can still be mapped
  geom_sf(data = sub, mapping = aes(fill = if_else(
    condition = trainees_per_thous == 0,
    true = log(sort(unique(trainees_per_thous))[2] / 2),
    false = log(trainees_per_thous))), color = "black", size = 0.1) +
  scale_fill_viridis(option = "viridis") +
  theme_void(base_size = 14) +
  theme(plot.subtitle = element_text(hjust = 0.5),
        legend.position = c(0.3, 0.40)) +
  labs(subtitle = "Trainees (n = 1,024,369) by Community\nEmergency Response Teams (n =
 148)\namong County Subdivisions (n = 427)",
       fill = "Trainees per\n1000 residents\n(log-scaled)") +
  ggsave("viz/fig_B2_cert.png", dpi = 500, width = 4, height = 5)
```

EMPG (Emergency Management Performance Grants)

Description from website: "This dataset contains EMPG recipients as reported by their State and a summary of the funded program support as reported by the Recipient in the Grants Reporting Tool (GRT).

The EMPG Program provides resources to assist state, local, tribal and territorial governments in preparing for all hazards, as authorized by Section 662 of the Post Katrina Emergency Management Reform Act (6 U.S.C ’ 762) and the Robert T. Stafford Disaster Relief and Emergency Assistance Act, as amended (42 U.S.C. ’’ 5121 et seq.). Title VI of the Stafford Act authorizes FEMA to make grants for the purpose of providing a system of emergency preparedness for the protection of life and property in the United States from hazards and to vest responsibility for emergency preparedness jointly in the federal government and the states and their political subdivisions. The FY 2016 EMPG will provide federal funds to assist state, local, tribal and territorial emergency management agencies to obtain the resources required to support the National Preparedness Goal’s (NPG’s) (the Goal’s) associated mission areas and core capabilities. The federal government, through the EMPG Program, provides necessary direction, coordination, and guidance, and provides necessary assistance, as authorized in this title to support a comprehensive all hazards emergency preparedness system.

The Emergency Management Performance Grant Program is to support a comprehensive, all-hazard emergency preparedness system by building and sustaining the core capabilities contained in the National Preparedness Goal.

Examples include:

-Completing the Threat and Hazard Identification and Risk Assessment (THIRA) process;

-Strengthening a state or community’s emergency management governance structures;

-Updating and approving specific emergency plans;

-Designing and conducting exercises that enable whole community stakeholders to examine and validate core capabilities and the plans needed to deliver them to the targets identified through the THIRA;

-Targeting training and verifying identified capabilities;

-Initiating or achieving a whole community approach to security and emergency management."

Hide

```
dat <- read_csv("raw_data/covariates/fema/EmergencyManagementPerformanceGrants.csv")

dat %>% head()
# Uh-oh, just kidding, they didn't geocode this dataset.
remove(dat)
```

NHS (National Household Survey)

The National Household Survey (NHS) tracks progress in personal disaster preparedness through investigation of the American public's preparedness actions, attitudes, and motivations. FEMA administers the survey in English and Spanish via landline and mobile telephone to a random sampling of approximately 5,000 adult respondents. The survey includes a nationally representative sample as well as hazard-specific oversamples which may include earthquake, flood, wildfire, hurricane, winter storm, extreme heat, tornado, and urban event. FEMA delays publishing the data until approximately the release of the summary results for the subsequent NHS iteration. For example, FEMA published the 2017 data package at approximately the same time as the publication of the 2018 NHS Summary.

Each zip file may include an analysis summary, the survey instrument, raw weighted and unweighted data, aggregated data analysis, and a codebook with weighting overviews.

Raw Data: Datasets may include unedited raw data. As such, users should plan to clean the data as needed prior to analysis.

Citation: Users should cite the date the data was accessed or retrieved from fema.gov. In addition, users must clearly state that "FEMA and the Federal Government cannot vouch for the data or analyses derived from these data after the data have been retrieved from the Agency's website."

Hazard Mitigation Grants

https://www.fema.gov/openfema-data-page/hazard-mitigation-assistance-mitigated-properties (https://www.fema.gov/openfema-data-page/hazard-mitigation-assistance-mitigated-properties)

September 9, 2020 version 1

This dataset contains the properties that were mitigated by projects funded under the Hazard Mitigation Assistance (HMA) grant programs. FEMA administers three programs that provide funding for eligible mitigation planning and projects to reduce disaster losses and protect life and property from future disaster damages. The three programs are the Hazard Mitigation Grant Program (HMGP), Flood Mitigation Assistance (FMA) grant program, and Pre-Disaster Mitigation (PDM) grant program. This dataset also contains data from the HMA grant programs that were eliminated by the Biggert Water Flood Insurance Reform Act of 2012 (BW-12): Repetitive Flood Claims (RFC) grant program and Severe Repetitive Loss (SRL) grant program. For more information on the Hazard Mitigation Assistance grant programs, please visit: https://www.fema.gov/hazard-mitigation-assistance (https://www.fema.gov/hazard-mitigation-assistance).

The dataset contains properties by project identifier, city, zip code, state and region and does not contain any Personally Identifiable Information (PII). The mitigated property dataset can be joined to the OpenFEMA Hazard Mitigation Assistance Funded Project dataset by the Project Identifier field. Note, not all projects in the Hazard

Mitigation Assistance Funded Project dataset will have mitigated properties (e.g., Planning and Management Cost projects). In some cases data was not provided by the subgrantee (sub-recipient), grantee (recipient) and/or entered into the FEMA mitigation grant systems. The information is likely available as part of the paper file which is considered the file of record.

This is raw, unedited data from FEMA's mitigation grant systems (NEMIS-MT and e-Grants) and as such is subject to a small percentage of human error. The financial information is derived from FEMA's mitigation grant systems and not FEMA's official financial systems. Due to differences in reporting periods, status of obligations and how business rules are applied, this financial information may differ slightly from official publication on public websites such as http://www.usaspending.gov (http://www.usaspending.gov); this dataset is not intended to be used for any official federal financial reporting.

Missing values - In some cases data was not provided by the subgrantee (subrecipient), grantee (recipient) and/or entered into the FEMA mitigation grant systems. The information is likely available as part of the paper file which is considered the file of record.

A newer version of this OpenFEMA data set has been released. This older dataset version will no longer be updated and will be archived by the end of April 2020. The following page details the latest version of this data set: https://www.fema.gov/openfema-dataset-hazard-mitigation-assistance-mitigated-properties-v2 (https://www.fema.gov/openfema-dataset-hazard-mitigation-assistance-mitigated-properties-v2). CSV and JSON Files can be downloaded from the 'Full Data' section.

To access the dataset through an API endpoint, visit the 'API Endpoint' section of the above page. Accessing data in this fashion permits data filtering, sorting, and field selection. The OpenFEMA API Documentation page provides information on API usage.

If you have media inquiries about this dataset, please email the FEMA News Desk FEMA-News-Desk@dhs.gov (mailto:FEMA-News-Desk@dhs.gov) or call (202) 646-3272. For inquiries about FEMA's Hazard Mitigation Assistance grant program data and Open government program, please contact the OpenFEMA team via email OpenFEMA@fema.dhs.gov (mailto:OpenFEMA@fema.dhs.gov) and FEMA-HMAAnalytics@fema.dhs.gov (mailto:FEMA-HMAAnalytics@fema.dhs.gov).

Hide

```
# Let's import these ~60,000 records of hazard mitigation projects
dat <- read_csv("raw_data/covariates/fema/HazardMitigationAssistanceMitigatedProperties.
csv") %>%
  # Let's retrieve the county fips code
  mutate(fips = paste(
    str_pad(stateNumberCode, width = 2, side = "left", pad = "0"),
    str_pad(countyCode, width = 3, side = "left", pad = "0"),
    sep = "")) %>%
  select(project = projectIdentifier, #Single-value identifier that uniquely identifies
 a project. Disaster-based projects use the convention of DR-disaster number-project num
ber-suffix (e.g., DR-1761-0001-M) while non-disaster projects use the unique project num
ber (e.g., FMA-PJ-10-WA-2017-006)
         id,
         state,
         fips,
         zipcode = zipCode,
         properties = numberOfProperties, # Number of properties meeting this criteria
         amount_paid = actualAmountPaid, # Amount paid to the property owner.
         status, # Current status of the project such as Approved, Awarded Obligated, Co
mpleted, Closed
         date_approved = dateApproved, # Date the project was initially approved by FEM
A.
         date_closed = dateClosed, # Date the project was closed by FEMA
         program = programArea # Hazard Mitigation Assistance grant program areas such a
s FMA - Flood Mitigation Assistance grant program, HMGP - Hazard Mitigation Grant Progra
m, LPDM - Legislative Pre-disaster Mitigation grant program, PDM - Pre-disaster Mitigati
on grant program, RFC - Repetitive Flood Claims grant program, SRL - Severe Repetitive L
oss grant program
        )  %>%
    filter(state %in% c("Florida", "Georgia", "Alabama")) %>%
  # Only count projects finished before August 1, the month of the hurricane
  filter(date_closed < "2019-08-01")


dat %>%
  # For each zipcode, let's tally...
  group_by(zipcode) %>%
  # The actual dollar amount spent is often unclear.
  # Instead, it seems more useful to quantify the number of projects (rows)
  # or the number of properties involved in each project (properties)
  # Then, we can divide this number by the population
  summarize(projects = n(),
            properties = sum(properties, na.rm = TRUE),
            amount_paid = sum(amount_paid, na.rm = TRUE)) %>%
  ungroup() %>%
  saveRDS("raw_data/covariates/fema/mitigation_zipcode.rds")


# Get Albers Equal Area Conic Projection
#https://spatialreference.org/ref/esri/north-america-albers-equal-area-conic/
aea <- "+proj=aea +lat_1=20 +lat_2=60 +lat_0=40 +lon_0=-96 +x_0=0 +y_0=0 +ellps=GRS80 +d
atum=NAD83 +units=m +no_defs"
```

```
# Download zipcode population from the American Community Survey
tidycensus::census_api_key("97b29053a89e5396ef806bf7f4fae05b7387fabd")
pop <- get_acs(
  year = 2019, state = c("FL", "AL", "GA"),
  geography = "zcta",
  variables = c("pop" = "B01003_001E")) %>%
  select(zipcode = GEOID, pop = estimate)

# Let's identify all census tracts that our CERT teams are involved in
# Grab zipcodes
read_rds("shapes/zipcodes.rds") %>%
  select(-area_land) %>%
  # Join in mitigation project tallies
  left_join(by = "zipcode",
            y =  read_rds("raw_data/covariates/fema/mitigation_zipcode.rds")) %>%
  # Join in population
  left_join(by = "zipcode", y = pop) %>%
  # Calculate mitigation tallies per 1000 residents
  mutate(projects = projects / pop * 1000,
         properties = properties / pop * 1000,
         amount_paid = amount_paid / pop * 1000) %>%
  # If the value is infinite, that means we recorded a population of 0
  # in that census division; maybe it's the everglades. In that case,
  # we should record this as a 0, because that money wasn't applied to communities
  mutate_at(vars(projects, properties, amount_paid),
            funs(if_else(is.infinite(.), 0, as.numeric(.) ))) %>%
  # Apply an Equal Area Projection
  st_transform(crs = aea) %>%
  # Join in division
  st_join(read_rds("shapes/csub.rds") %>%
            st_transform(crs = aea)) %>%
  as.data.frame() %>%
  # Calculate mean rate of hazard mitigation per census tract
  group_by(geoid) %>%
  summarize(projects = mean(projects, na.rm = TRUE),
            properties = mean(properties, na.rm = TRUE),
            amount_paid = mean(amount_paid, na.rm = TRUE)) %>%
  ungroup() %>%
  # If it's still missing, this means they didn't have any projects,
  # so we should set this as 0
  mutate_at(vars(projects, properties, amount_paid),
            funs(if_else(is.na(.), 0, as.numeric(.)))) %>%
  saveRDS("raw_data/covariates/fema/mitigation_csub.rds")

remove(dat, pop)

# Finally, let's map it!
sub <- read_rds("shapes/csub.rds") %>%
  left_join(by = "geoid",
            y = read_rds("raw_data/covariates/fema/mitigation_csub.rds"))

ggplot() +
  geom_sf(data = sub, fill = NA, color = "darkgrey", size = 5) +
  # Add a log-transformed properties scale.
```

```
  # In a few cases, there were 0 properties affected.
  # We're going to a dd a tiny-but-reasonable constant (lowest observed divided by 2)
  # So that this can still be mapped
  geom_sf(data = sub, mapping = aes(fill = if_else(
    condition = properties == 0, true = log(sort(unique(properties))[2] / 2),
    false = log(properties))), color = "black", size = 0.1) +
  scale_fill_viridis(option = "viridis") +
  theme_void(base_size = 14) +
  theme(plot.subtitle = element_text(hjust = 0.5),
        legend.position = c(0.3, 0.40)) +
  labs(subtitle = "Total Properties Improved (n = 13083)\nby FEMA Hazard Mitigation\nAss
istance Grant Projects (n = 9103)\namong County Subdivisions (n = 427)",
       fill = "Properties per\n1000 residents\n(log-scaled)") +
  ggsave("viz/fig_B1_hazard_mitigation.png", dpi = 500, width = 4, height = 5)
```

1.5 Rainfall

Next, we're going to approximate how much rain fell in each census tract every week during our study period.

NOAA has an excellent dataset hosted by the Global Historical Climatology Network (https://www.ncdc.noaa.gov/data-access/land-based-station-data/land-based-datasets/global-historical-climatology-network-ghcn (https://www.ncdc.noaa.gov/data-access/land-based-station-data/land-based-datasets/global-historical-climatology-network-ghcn)). GHCN "is an integrated database of climate summaries from land surface stations across the globe that have been subjected to a common suite of quality assurance reviews. The data are obtained from more than 20 sources. Some data are more than 175 years old while others are less than an hour old." They host data on an FTP server that contains daily precipitation, broken into yearly datasets (ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/by_year/ (ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/by_year/)). A README file is available here (https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/by_year/readme.txt (https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/by_year/readme.txt)). The locations of recording stations is available here (https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/ghcnd-stations.txt (https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/ghcnd-stations.txt)), and we can use these to locate and spatially interpolate these rainfall levels.

The journal article describing GHCN-Daily is: Menne, M.J., I. Durre, R.S. Vose, B.E. Gleason, and T.G. Houston, 2012: An overview of the Global Historical Climatology Network-Daily Database. Journal of Atmospheric and Oceanic Technology, 29, 897-910, doi:10.1175/JTECH-D-11-00103.1 (doi:10.1175/JTECH-D-11-00103.1).

To acknowledge the specific version of the dataset used, please cite: Menne, M.J., I. Durre, B. Korzeniewski, S. McNeal, K. Thomas, X. Yin, S. Anthony, R. Ray, R.S. Vose, B.E.Gleason, and T.G. Houston, 2012: Global Historical Climatology Network - Daily (GHCN-Daily), Version 3. [indicate subset used following decimal, e.g. Version 3.12]. NOAA National Climatic Data Center. http://doi.org/10.7289/V5D21VHZ (http://doi.org/10.7289/V5D21VHZ) [access date].

Get data

Hide

```
# First, let's make a rainfall folder
dir.create("raw_data/covariates/rainfall")

# Get Albers Equal Area Conic Projection
#https://spatialreference.org/ref/esri/north-america-albers-equal-area-conic/
aea <- "+proj=aea +lat_1=20 +lat_2=60 +lat_0=40 +lon_0=-96 +x_0=0 +y_0=0 +ellps=GRS80 +d
atum=NAD83 +units=m +no_defs"



# File readme available here:
# https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/by_year/readme.txt

# Station information
# https://catalog.data.gov/dataset/global-historical-climatology-network-daily-ghcn-dail
y-version-3/resource/bbd84e54-5a2e-4b2b-bedf-e6611b5113d7
# https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/ghcnd-stations.txt

read_delim(
  file = "https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/ghcnd-stations.txt",
  delim = " ",
  col_names = FALSE) %>%
  dplyr::select(station_id = 1, latitude = 2, longitude = 3) %>%
  mutate_at(vars(latitude, longitude), funs(. %>% as.numeric)) %>%
  # Convert to an SF object with the following coordinates
  st_as_sf(coords = c("longitude", "latitude"), crs = CRS(paste0("+proj=longlat +ellps=W
GS84 +datum=WGS84 +no_defs"))) %>%
  saveRDS("raw_data/covariates/rainfall/us_stations.rds")


# Now download 2020 rainfall data
# We're going to have to download it into a different directory, since it is HUGE.
#download.file(
#  url = "ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/by_year/2020.csv.gz",
#    destfile = "raw_data/covariates/rainfall/us_rainfall_2020.csv.gz")

# Download 2019 rainfall data
download.file(
  url = "ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/by_year/2019.csv.gz",
    destfile = "raw_data/covariates/rainfall/us_rainfall_2019.csv.gz")


# Write decompressed data
# (Note: This requires a LOT of computational power.
# I had to scale up my cloud server from 2 GB of RAM
# to 8 GB of RAM to pull this off.)
read_csv("raw_data/covariates/rainfall/us_rainfall_2019.csv.gz",
         col_names = FALSE) %>%
  # Grab just these vectors
  dplyr::select(station_id = 1, date = 2, type = 3, value = 4) %>%
  # Grab entries from August and September
  filter(str_sub(as.character(date), 5,6) %in% c("08", "09", "10")) %>%
  # Filter to just precipitation
```

```
  filter(type == "PRCP") %>%
  # Convert dates to appropriate format
    mutate(date = lubridate::make_date(
    year = str_sub(date, 1,4),
    month = str_sub(date, 5,6),
    day = str_sub(date, 7,8))) %>%
  # save to file
  saveRDS("raw_data/covariates/rainfall/us_rainfall_2019.rds")

# Now remove raw US rainfall
unlink("raw_data/covariates/rainfall/us_rainfall_2019.csv.gz")
```

Get Subset of Stations

Hide

```
## For the US, let's use the Contiguous US Albers Equal Area Conic Projection
#https://spatialreference.org/ref/esri/north-america-albers-equal-area-conic/
aea <- "+proj=aea +lat_1=20 +lat_2=60 +lat_0=40 +lon_0=-96 +x_0=0 +y_0=0 +ellps=GRS80 +d
atum=NAD83 +units=m +no_defs"



# Import census tracts
tracts <- read_rds("shapes/tracts.rds") %>%
  st_transform(crs = aea)


ggplot() +
  geom_sf(data = tracts) +
  geom_sf(data = read_rds("raw_data/covariates/rainfall/us_stations.rds") %>%
  sample_n(size = 2000), color = "red") +
  theme_void()

read_rds("raw_data/covariates/rainfall/us_stations.rds") %>%
  st_transform(crs = aea) %>%
  # Now join the codes of county subdivisions in the study area
  st_join(tracts) %>%
  # Filter to stations in the study area
  filter(!is.na(geoid)) %>%
  # save to file
  saveRDS("raw_data/covariates/rainfall/dorian_stations.rds")

# These are the locations of weather stations
ggplot() +
  geom_sf(data = tracts, mapping = aes(geometry=geometry),
          color = "darkgrey", fill = "black", size = 5) +
  geom_sf(data = tracts, mapping = aes(geometry=geometry),
          color = "grey", fill = "black", size = 0.1) +
  geom_sf(data = read_rds("raw_data/covariates/rainfall/dorian_stations.rds"),
          mapping = aes(geometry=geometry),
          color = "lightblue", size = 2, alpha = 0.75) +
  theme_void(base_size = 14) +
  theme(plot.subtitle = element_text(hjust = 0.5)) +
  labs(subtitle = "Land Surface Rainfall Data Stations near Hurricane Dorian\n(n = 2065
 stations)") +
  ggsave("viz/fig_B3_dorian_rainfall_stations.png", dpi = 300)
```

Interpolate

Hide

```
## For the US, let's use the Contiguous US Albers Equal Area Conic Projection
#https://spatialreference.org/ref/esri/north-america-albers-equal-area-conic/
aea <- "+proj=aea +lat_1=20 +lat_2=60 +lat_0=40 +lon_0=-96 +x_0=0 +y_0=0 +ellps=GRS80 +d
atum=NAD83 +units=m +no_defs"


# Import census tracts
tracts <- read_rds("shapes/tracts.rds") %>%
  st_transform(crs = aea)
# Import counties
counties <- read_rds("shapes/counties.rds")$geoid %>% unique()

# The shapefile is measured in meters,
# so when we make the fishnet cellsize 7500,
# that means 35,000 m (35 km).

tracts %>%
  st_make_grid(cellsize = 35000, what = "polygons") %>%
  st_as_sf() %>%
  mutate(id = 1:n()) %>%
  select(id, geometry = x) %>%
  saveRDS("raw_data/covariates/rainfall/dorian_fishnet.rds")

sub <- read_rds("shapes/csub.rds") %>%
  st_transform(crs = aea)

# Visualize fishnet grid here
ggplot() +
  # Plot background
  geom_sf(data = sub, fill = "black", color = NA) +
  # Plot fishnet
  geom_sf(data = read_rds("raw_data/covariates/rainfall/dorian_fishnet.rds"),
          color = "white", size = 0.1, fill = NA) +
  # Plot station locations
  geom_sf(data = read_rds("raw_data/covariates/rainfall/dorian_stations.rds") %>%
            filter(str_sub(geoid, 1,5) %in% counties),
          mapping = aes(geometry=geometry),
          color = "cyan", size = 0.5) +
  theme_void()
```

Hide

```
# Import rainfall data
rain <- read_rds("raw_data/covariates/rainfall/dorian_stations.rds") %>%
  left_join(by = "station_id",
            y = read_rds("raw_data/covariates/rainfall/us_rainfall_2019.rds") %>%
              # Now filter to just rain measurements from these sites
              filter(station_id %in% read_rds("raw_data/covariates/rainfall/dorian_stati
ons.rds")$station_id) %>%
              filter(type == "PRCP") %>%
              # Aggregate by day
              mutate(day = lubridate::ceiling_date(date, unit = c('day'))) %>%
              group_by(station_id, day) %>%
              summarize(rain = sum(value, na.rm = TRUE))) %>%
  filter(!is.na(rain)) %>%
  filter(day > "2019-08-20")


# Import fishnet grid
nodes <- read_rds("raw_data/covariates/rainfall/dorian_fishnet.rds")

interpolate = function(DATE){
  print(DATE)
  # Make a grid of shapes from the polygon
  m <- gstat::gstat(id = "id", formula = rain ~ 1,
                    data = rain %>%
                      filter(day == DATE) %>%
                      na.omit() %>%
                      as("Spatial"),
                    set = list(idp = 2))
  # Generate predictions
  predict(m, nodes) %>%
    dplyr::select(predicted = id.pred, geometry) %>%
    # Add in a unique id from the original grid cells
    mutate(id = nodes$id) %>%
    mutate(day = DATE) %>%
    return()
}


data.frame(day = rain$day %>% unique()) %>%
  split(.$day) %>%
  map(~interpolate(.$day), .id = "day") %>%
  dplyr::bind_rows() %>%
  saveRDS("raw_data/covariates/rainfall/dorian_rainfall_predicted.rds")
```

Hide

```
grid <- read_rds("raw_data/covariates/rainfall/dorian_rainfall_predicted.rds") %>%
  st_as_sf() %>%
  ungroup() %>%
  as.data.frame() %>%
  select(-geometry)

# Import fishnet grid
nodes <- read_rds("raw_data/covariates/rainfall/dorian_fishnet.rds") %>%
  left_join(by = "id", y = grid)

remove(grid)

# Average predictions among tracts
#tracts %>%
#  st_join(nodes) %>%
#  as.data.frame() %>%
#  select(-geometry) %>%
#  group_by(geoid, day) %>%
#  summarize(avg_rainfall = mean(predicted, na.rm = TRUE)) %>%
#  ungroup() %>%
#  saveRDS("raw_data/covariates/rainfall/dorian_rainfall_tracts.rds")

# Join rainfall into tracts
#rain <- tracts %>%
#  left_join(by = "geoid",
#            y = read_rds("raw_data/covariates/rainfall/dorian_rainfall_tracts.rds"))

# Finally, let's visualze this pattern of weekly rainfall, a useful way of adjusting for
exposure and damage to multiple storms during this hurricane season.
#ggplot() +
#  geom_sf(data = rain,
#          mapping = aes(fill = avg_rainfall), color = NA) +
#  theme_void() +
#  scale_fill_viridis(option = "plasma") +
#  facet_wrap(~day, ncol = 7) +
#  theme(plot.caption = element_text(hjust = 0.5),
#        legend.position = "bottom") +
#  guides(fill = guide_colorbar(barwidth = 10, barheight = 0.5)) +
#  labs(fill = "Daily\nRainfall",
#       caption = "Spatially interpolated across 35 km grid cells and averaged by census
tract.") +
#  ggsave("viz/rainfall_tracts.png", dpi = 500, width = 8, height = 10)


sub <- read_rds("shapes/csub.rds") %>%
  st_transform(crs = aea)

# Average predictions among divisions
sub %>%
  st_join(nodes) %>%
  as.data.frame() %>%
  select(-geometry) %>%
  group_by(geoid, day) %>%
```

```
  summarize(avg_rainfall = mean(predicted, na.rm = TRUE)) %>%
  ungroup() %>%
  saveRDS("raw_data/covariates/rainfall/dorian_rainfall_csub.rds")

# Join rainfall into divisions
rain <- sub %>%
  left_join(by = "geoid",
            y = read_rds("raw_data/covariates/rainfall/dorian_rainfall_csub.rds"))

# Finally, let's visualze this pattern of weekly rainfall, a useful way of adjusting for
exposure and damage to multiple storms during this hurricane season.
ggplot() +
  geom_sf(data = rain,
          mapping = aes(fill = avg_rainfall), color = NA) +
  theme_void() +
  scale_fill_viridis(option = "plasma") +
  facet_wrap(~day, ncol = 14) +
  theme(plot.caption = element_text(hjust = 0.5),
        legend.position = "bottom") +
  guides(fill = guide_colorbar(barwidth = 10, barheight = 0.5)) +
  labs(fill = "Daily\nRainfall",
       caption = "Spatially interpolated across 35 km grid cells and averaged by county
 subdivision.") +
  ggsave("viz/other_rainfall_csub.png", dpi = 500, width = 12, height = 6)
```

Hide

```
remove(nodes, pred, shapes, rain, sub)
```

1.6 Build Network

Hide

```
# Finally, let's convert this into a network object.
# This network object is in tidygraph, via the sfnetworks package;
# It is geocoded, and can be mapped.
# It is a directed network, weighted by the number of evacuees

# Get key subsets
soc <- read_csv("raw_data/covariates/sci_county_subdivisions_2021_01_28.csv") %>%
  filter(year == 2018) %>%
  select(csub, social_capital, bonding, bridging, linking) %>%
  filter(str_sub(csub, 1,5) %in% read_rds("shapes/counties.rds")$geoid)

svi <- read_rds("raw_data/covariates/svi_csub.rds") %>%
  filter(state %in% c("FL","GA","AL")) %>%
  select(csub, contains("svi")) %>%
  filter(str_sub(csub, 1,5) %in% read_rds("shapes/counties.rds")$geoid)

# Import your network object
sfnetworks::sfnetwork(
  nodes = read_rds("shapes/csub_centroid.rds"),
  edges = read_rds("raw_data/dorian_edges_csub.rds"),
  force = TRUE,
  directed = TRUE,
  node_key = "geoid",
  edges_as_lines = TRUE,
  length_as_weight = FALSE) %>%
  # Now briefly transform the coordinate projection system
  # to North American Equidistant Conic
  st_transform(crs = CRS(paste0("+proj=eqdc +lat_0=0 +lon_0=0 +lat_1=20 +lat_2=60 +x_0=0
+y_0=0 +ellps=GRS80 +datum=NAD83 +units=m +no_defs"))) %>%
  activate("edges") %>%
  # And calculate the distance between tracts in km
  mutate(km = as.numeric(st_length(geometry, which = "Euclidean") / 1000)) %>%
  # Now transform back to WSG84
  st_transform(CRS(paste0("+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs"))) %>%
  # Now, let's add in some node traits
  activate("nodes") %>%
  # Join in Social Capital Indices
  left_join(by = c("geoid" = "csub"),
            y = soc) %>%
  # Join in SVI
  left_join(by = c("geoid" = "csub"),
            y = svi) %>%
  # Join in Census Traits
  left_join(by = "geoid",
            y = read_rds("raw_data/covariates/census.rds")) %>%
  # Join in county political partisanship
  mutate(county = str_sub(geoid, 1,5)) %>%
  left_join(by = c("county" = "fips"),
            y = read_csv("raw_data/covariates/county_elections.csv")) %>%
  # Join in precinct political partisanship, averaged to division level
  left_join(by = c("geoid"),
            y = read_rds("raw_data/covariates/precincts_csub.rds") %>%
              rename(dem_percent_precinct = dem_percent)) %>%
```

```
  # Join in FEMA Hazard Mitigation data
  left_join(by = "geoid",
            y = read_rds("raw_data/covariates/fema/mitigation_csub.rds")) %>%
  # Join in FEMA Community Emergency Response Team data
  left_join(by = "geoid",
            y = read_rds("raw_data/covariates/fema/cert_csub.rds")) %>%
  # Fill in missing with 0, because they were not tabulated
  mutate_at(vars(projects, properties, amount_paid, programs,
                 cert, trainings_per_year, classes, trainees),
            funs(if_else(is.na(.), 0, as.numeric(.)))) %>%
  # Also, population control our CERT data
  # (but not Hazard data; that's already a rate)
  mutate_at(vars(programs, trainings_per_year, classes, trainees),
            funs(. / pop * 1000)) %>%
  # Also, some tracts have 0 persons; please set these to zero.
  mutate_at(vars(programs, trainings_per_year, classes, trainees),
            funs(if_else(is.infinite(.), 0, as.numeric(.) ))) %>%
  # Join in total rainfall
  # Let's evaluate the total estimated rainfall per census tract
  # between 2020-02-02 and 2020-02-16 (the last fourteen days)
  # (# Our evacuation data begins on 2-16)
  left_join(by = "geoid",
    y = read_rds("raw_data/covariates/rainfall/dorian_rainfall_csub.rds") %>%
      filter(day >= "2019-08-14",
             day < "2019-08-28") %>%
      group_by(geoid) %>%
      summarize(rainfall_14days = sum(avg_rainfall, na.rm = TRUE)) %>%
      ungroup()) %>%
  # save
  saveRDS("raw_data/dorian.rds")


# Get North American Equidistant Conic Projection
#https://spatialreference.org/ref/esri/102010/
aed <- "+proj=eqdc +lat_0=0 +lon_0=0 +lat_1=20 +lat_2=60 +x_0=0 +y_0=0 +ellps=GRS80 +dat
um=NAD83 +units=m +no_defs"

remove(centroids, edgelist, neighborhoods, ids, states,counties, soc, svi)
```

1.7 Facebook Mobility

Hide

```
# 3. Data

intra_more <- read_rds("raw_data/dorian.rds") %>%
  activate("edges") %>%
  # Zoom into movement between neighborhoods within the same census csub
  filter(from == to) %>%
  filter(evacuation > 0) %>%
  mutate(from_csub = .N()$geoid[from],
         to_csub = .N()$geoid[to]) %>%
  as.data.frame() %>%
  # Calculate total evacuation within a census csub
  group_by(csub = from_csub, date_time) %>%
  summarize(evacuation = sum(evacuation, na.rm = TRUE)) %>%
  ungroup()

# Calculate decrease in movement within census csubs
intra_less <- read_rds("raw_data/dorian.rds") %>%
  activate("edges") %>%
  # Zoom into movement between neighborhoods within the same census csub
  filter(from == to) %>%
  # Evaluate DECREASE in movement
  filter(evacuation < 0) %>%
  mutate(from_csub = .N()$geoid[from],
         to_csub = .N()$geoid[to]) %>%
  as.data.frame() %>%
  # Calculate total evacuation within a census csub
  group_by(csub = from_csub, date_time) %>%
  summarize(evacuation = sum(abs(evacuation), na.rm = TRUE)) %>%
  ungroup()

# Calculate increase in movement between census csubs
inter_more <- read_rds("raw_data/dorian.rds") %>%
  activate("edges") %>%
  # Zoom into movement between different census csubs
  filter(from != to) %>%
  # Evaluate INCREASE in movement
  filter(evacuation > 0) %>%
  mutate(from_csub = .N()$geoid[from],
         to_csub = .N()$geoid[to]) %>%
  as.data.frame()
# Now calculate it for both sender and receivers
inter_more <- bind_rows(
  inter_more %>%
    group_by(csub = from_csub, date_time) %>%
    summarize(evacuation = sum(evacuation, na.rm = TRUE)) %>%
    ungroup(),
  inter_more %>%
    group_by(csub = to_csub, date_time) %>%
    summarize(evacuation = sum(evacuation, na.rm = TRUE)) %>%
    ungroup()) %>%
  group_by(csub, date_time) %>%
  summarize(evacuation = sum(evacuation, na.rm = TRUE)) %>%
  ungroup()
```

```
# Calculate DECREASE in movement between census csubs
inter_less <- read_rds("raw_data/dorian.rds") %>%
  activate("edges") %>%
  # Zoom into edges between different census csubs
  filter(from != to) %>%
  # Evaluate DECREASE in movement
  filter(evacuation < 0) %>%
  mutate(from_csub = .N()$geoid[from],
         to_csub = .N()$geoid[to]) %>%
  as.data.frame()
# Now calculate it for both sender and receivers
inter_less <- bind_rows(
  inter_less %>%
    group_by(csub = from_csub, date_time) %>%
    summarize(evacuation = sum(abs(evacuation), na.rm = TRUE)) %>%
    ungroup(),
  inter_less %>%
    group_by(csub = to_csub, date_time) %>%
    summarize(evacuation = sum(abs(evacuation), na.rm = TRUE)) %>%
    ungroup()) %>%
  group_by(csub, date_time) %>%
  summarize(evacuation = sum(evacuation, na.rm = TRUE)) %>%
  ungroup()

# Bind these lists together into a single data.frame
bind_rows(
  intra_more, intra_less,
  inter_more, inter_less, .id = "type") %>%
  mutate(type = recode_factor(
    type,
    "1" = "evacuation_intra_more", "2" = "evacuation_intra_less",
    "3" = "evacuation_inter_more", "4" = "evacuation_inter_less")) %>%
  pivot_wider(
    id_cols = c(csub, date_time),
    names_from = type,
    values_from = evacuation) %>%
  mutate_at(vars(matches("evacuation")),
            funs(if_else(is.na(.), as.numeric(0), as.numeric(.)))) %>%
  mutate(evacuation_more = evacuation_intra_more + evacuation_inter_more,
         evacuation_less = evacuation_intra_less + evacuation_inter_less) %>%
  saveRDS("raw_data/dorian_csub_movement.rds")

# Now let's add that into the dorian network object
read_rds("raw_data/dorian.rds") %>%
  activate("nodes") %>%
# Finally, let's summarize the total population movement over time
  left_join(by = c("geoid" = "csub"),
            y = read_rds("raw_data/dorian_csub_movement.rds") %>%
    group_by(csub) %>%
    summarize_at(vars(contains("evacuation")),
                 funs(sum(., na.rm = TRUE))) %>%
  ungroup()) %>%
```

```
  saveRDS("raw_data/dorian.rds")

rm(list= ls())
```

1.8 Time-series dataset

Hide

```
# Get a grid of all census county subdivision-date-time pairs in our study
expand_grid(
  # Get the full range of census divisions in our study
  geoid = read_rds("shapes/csub.rds")$geoid %>% unique(),
  # Get the full range of dates in our study
  # Our evac data starts on August 28, but we're going to grab August 22
  # so we can lag rainfall by a few days
  data.frame(date_time = seq(from = lubridate::ymd_hms("2019-08-22 00:00:00 UTC"),
                             to = lubridate::ymd_hms("2019-10-27 16:00:00 UTC"),
                             by = "hours")) %>%
    filter(str_sub(date_time, 12,13) %in% c("00", "08", "16"))) %>%
  # Join in the outcomes
  left_join(by = c("geoid" = "csub", "date_time"),
            y = read_rds("raw_data/dorian_csub_movement.rds")) %>%
  # If missing, fill in with zero
  mutate_at(vars(contains("evacuation")),
            funs(if_else(is.na(.), 0, as.numeric(.)))) %>%
  # Join in daily rainfall
  mutate(day = str_sub(date_time, 1,10) %>% lubridate::ymd()) %>%
  left_join(by = c("geoid", "day"),
            y = read_rds("raw_data/covariates/rainfall/dorian_rainfall_csub.rds")) %>%
  # Next, we're going to make a few lags
  arrange(geoid, day) %>%
  group_by(geoid) %>%
  mutate(avg_rainfall1 = lag(avg_rainfall, 1),
         avg_rainfall2 = lag(avg_rainfall, 2),
         avg_rainfall3 = lag(avg_rainfall, 3),
         avg_rainfall4 = lag(avg_rainfall, 4),
         avg_rainfall5 = lag(avg_rainfall, 5)) %>%
  ungroup() %>%
  # Join in all other covariates from before, which we already saved into the network
  left_join(by = c("geoid"),
            y = read_rds("raw_data/dorian.rds") %>%
              activate("nodes") %>%
              as.data.frame() %>%
              select(-geometry, -contains("evacuation"), -contains("rainfall"))) %>%
   # Join in area, and make population density
  left_join(by = c("geoid"),
            y = read_rds("shapes/csub.rds") %>%
              as.data.frame() %>%
              select(geoid, area_land)) %>%
  mutate(pop_density = pop / area_land) %>%
  # Now filter to just the study period
  filter(day >= "2019-08-28") %>%
  # Save our timeseries dataset
  saveRDS("raw_data/dorian_ts.rds")
```

2. Visualize

Hide

```
# Import network
net <- read_rds("raw_data/dorian.rds") %>%
  activate("edges")

# Import a background, based on counties
back <- read_rds("shapes/counties.rds") %>%
  summarize(geometry = st_union(geometry))

# Get state boundary lines
states <- read_rds("shapes/counties.rds") %>%
  group_by(state = str_sub(geoid, 1,2) %>%
              recode_factor("12" = "Florida",
                            "13" = "Georgia",
                            "01" = "Alabama")) %>%
  summarize(geometry = st_union(geometry))

# Import counties
counties <- read_rds("shapes/counties.rds")

# Import divisions
sub <- read_rds("shapes/csub.rds") %>%
  left_join(by = c("geoid" = "geoid"),
            y = net %>%
              activate("nodes") %>%
              as.data.frame() %>%
              select(geoid, rainfall_14days))

# Visualize!
ggplot() +
  # Give a nice grey background
  geom_sf(data = back, fill = NA, color = "darkgrey", size = 5) +
  # Highlight rainfall by division (in mm, since the original unit is 10ths of a mm)
  geom_sf(data = sub, mapping = aes(fill = rainfall_14days / 10),
          color = NA, size = 0.25)  +
  scale_fill_viridis(option = "plasma", direction = 1) +
  # Add some county boundaries over top
  geom_sf(data = counties, fill = NA, color = "grey", size = 0.1, alpha = 0.5) +
  # Add some state boundaries overtop
  geom_sf(data = states, fill = NA, color = "black", size = 1) +
  # Add edges over top
  geom_sf(data = st_as_sf(net, "edges") %>%
            filter(evacuation > 0), color = "white", alpha = 0.1, size = 0.2) +
  # Add cities size by evacuation over top
  geom_sf(data = st_as_sf(net, "nodes") %>%
            filter(evacuation_more > 0),
          mapping = aes(size = evacuation_more),
          color = "#648FFF", alpha = 0.5) +

  theme_void(base_size = 14)  +
  #facet_wrap(~date_time, ncol = 10) +
  theme(legend.position = c(0.3, 0.4), legend.box = "horizontal",
        plot.subtitle = element_text(hjust = 0.5)) +
  labs(size = "# of Evacuees\n(August 28\nto October 27)",
```

```
       fill = "Total Daily\nRainfall (mm)\nAugust 14-28",
       subtitle = "Evacuation Pathways (n = 130,067) After Hurricane Dorian\nBetween Cou
nty Subdivisions (n = 427) over 2 Months") +
  ggsave("viz/fig_1_evacuation_network_over_time.png", dpi = 500, width = 5.2, height =
5)
```

3. Modeling dataset

Format Dataset

Hide

```
# Now get counties, based on their evacuation order
mycounties <- read_rds("shapes/counties.rds") %>%
  mutate(type = case_when(
    # Florida
    str_sub(geoid, 1,2) == 12 & name %in% c("Martin", "Palm Beach", "Brevard",
                                            "St. Lucie", "St. Johns") ~ "Mandatory",
    str_sub(geoid, 1,2) == 12 &name %in% c("Flagler", "Nassau",
                                           "Osceola", "Glades", "Hendry",
                                           "Okeechobee", "Highlands") ~ "Voluntary",
    str_sub(geoid, 1,2) == 12 & name %in% c("Duval",  "Volusia",
                                            "Indian River", "Martin",
                                            "Broward", "Miami-Dade") ~ "None",
    # Georgia
    str_sub(geoid, 1,2) == 13 & name %in% c("Bryan", "Camden", "Chatham",
                                            "Glynn", "Liberty", "McIntosh") ~ "Mandator
y",
    str_sub(geoid, 1,2) == 13 & name %in% c("Charlton", "Effingham") ~ "None",
    TRUE ~ NA_character_) %>% factor(levels = c("Mandatory", "Voluntary", "None"))) %>%
  # Now zoom into just the coastal counties I have classified
  filter(!is.na(type)) %>%
  as.data.frame() %>%
  select(order = type, geoid)


# Import time-series dataset
read_rds("raw_data/dorian_ts.rds") %>%
  # Let's create indices of each of these measures
  # Each of these are count variables; we need to log-transform them if they are to show
variation well
  # We're going to add a small constant to each,
  # equal to half the value of the smallest non-zero value, and then log-transform them
  mutate_at(vars(projects, properties, amount_paid,
                 programs, trainings_per_year, classes, trainees),
            funs(if_else(. == 0,  true = sort(unique(.))[2] / 2, false = .))) %>%
  mutate(hard = (scale(log(projects)) + scale(log(properties) ) + scale(log(amount_paid)
)) / 3,
         soft = (scale(log(programs) ) + scale(log(trainings_per_year) ) +
                       scale(log(classes) ) + scale(log(trainees) )) / 4) %>%
  # Finally, we can't evaluate evacuation without a log-function.
  # More than likely, the excess zeros in this dataset are not real;
  # people did move, but just too small a number to detect.
  # This analysis will try out adding a very very small-but-realistic number,
  # to allow a log-transformation to work with this data.
  # A common rule of thumb is to replace 0s with
  # half the next-smallest value in your dataset.
  # This means that we don't give these zeros crazy statistical power by making them tin
y
  # Also, we'll population control it into a rate per 1000 residents
  mutate_at(vars(contains("evac")),
            funs(if_else(condition = . == 0,
                         true = (sort(unique(.))[2] / 2) / pop * 1000,
                         false = . / pop * 1000))) %>%
```

```
  # Two census tracts have no census information, indicating no one lives there.
  # (Also, no one evacuated there.)
  # And 3 county subdivisions are consistently missing all census trait data,
  # so we'll remove those (they're the ones missing % race data)
  # Let's cut them for the time being
  filter(!is.na(pop_black)) %>%
  # A handful of census tracts are missing data.
  # We're going to impute the county mean for them
  # svi - 1 / 427 missing
  # median_income - 6 / 427
  # income inequality = 4 / 427
  # svi_minority = 1 / 427
  # svi_housing_transport = 1 / 427
  # svi_household_disability = 1 / 427
  # svi_socioeconomic = 1 / 427
  # pop_black = 3 / 427,
  # pop_unemployed = 4 / 427
  # bonding, bridigng, linking = 1 / 427
  # Impute county mean
  #(since the data is time invariant,
  # we want to do this in temporal groups too)
  group_by(date_time, county) %>%
  mutate_at(vars(contains("svi"), median_income,
                 income_inequality, pop_unemployed,
                 social_capital, bonding, bridging, linking),
            funs(if_else(is.na(.), mean(., na.rm = TRUE), .))) %>%
  ungroup() %>%
  # Rescale predictors
  mutate_at(vars(contains("rainfall"), social_capital, bonding, bridging, linking,
                 contains("svi"), hard, soft, contains("pop"),
                 employees_muni, median_income, income_inequality,
                 median_monthly_housing_cost,
                 contains("democrat"), contains("republican"), dem_percent_precinct),
            funs(scale(.))) %>%
  # Join in county evacuation orders
  left_join(by = c("county" = "geoid"), y = mycounties) %>%
  mutate(order = case_when(
    is.na(order) ~ "None",
    # Fix so that any county observation BEFORE September 2 has NO evacuation order,
  # since they didn't get announced until then
    day < "2019-09-02" ~ "None",
    TRUE ~ as.character(order)) %>%
           factor() %>% relevel(ref = "None")) %>%
  saveRDS("raw_data/dorian_ts_dataset.rds")

remove(mycounties)
```

4. Overall Model

Coastal Map

Hide

```
## Basic Model

library(lme4)
library(lmerTest)
library(texreg)

# We're going to zoom into counties with mandatory evacuation orders.

counties <- read_rds("shapes/counties.rds")
mycounties <- read_rds("shapes/counties.rds") %>%
  mutate(type = case_when(
    # Florida
    str_sub(geoid, 1,2) == 12 & name %in% c("Martin", "Palm Beach", "Brevard",
                                            "St. Lucie", "St. Johns") ~ "Mandatory",
    str_sub(geoid, 1,2) == 12 &name %in% c("Flagler", "Nassau",
                                           "Osceola", "Glades", "Hendry",
                                           "Okeechobee", "Highlands") ~ "Voluntary",
    str_sub(geoid, 1,2) == 12 & name %in% c("Duval",  "Volusia",
                                            "Indian River", "Martin",
                                            "Broward", "Miami-Dade",
                                            "Clay", "Putnam",  "Lake",
                                            "Seminole", "Orange") ~ "None",
    # Georgia
    str_sub(geoid, 1,2) == 13 & name %in% c("Bryan", "Camden", "Chatham",
                                            "Glynn", "Liberty", "McIntosh") ~ "Mandator
y",
    str_sub(geoid, 1,2) == 13 & name %in% c("Charlton", "Effingham", "Brantley") ~ "Non
e",
    TRUE ~ NA_character_) %>% factor(levels = c("Mandatory", "Voluntary", "None"))) %>%
  # Now zoom into just the coastal counties I have classified
  filter(!is.na(type))


# https://time.com/5666941/hurrican-dorian-evacuation-orders-counties/

mycolors <- c("#9FC58F", "#648FFF", "#785EF0","#DC267F","#FE6100","#FFB000")

ggplot() +
  geom_sf(data = counties, fill = NA, color = "darkgrey", size = 5) +
  geom_sf(data = counties, fill = "black", color = "darkgrey", size = 0.1) +
  geom_sf(data = mycounties, mapping = aes(fill = type), color = "white", size = 1) +
  #geom_sf_label(data = mycounties %>%
  #                filter(name != "Okeechobee") %>%
  #                mutate(name = if_else(name == "Palm Beach", "Palm\nBeach", name)),
  #              mapping = aes(label = name), nudge_x = 1,
  #             color = "black", size = 2) +
  scale_fill_manual(values = mycolors[c(2,4,6)]) +
  theme_void(base_size = 14) +
  theme(plot.caption = element_text(hjust = 0.5),
        legend.position = c(0.3, 0.45)) +
  labs(caption = "Coastal Counties by Evacuation Orders as of September 2, 2019",
       fill = "Evacuation Order") +
  ggsave("viz/fig_2_evac_counties.png", dpi = 500, width = 5, height = 5)
```

```
# Get observed edges in hotspot cities
read_rds("raw_data/dorian.rds") %>%
  # Filter to hotspot cities that evacuated at least once
  activate("nodes") %>%
  filter(evacuation_more > 0) %>%
  # Get the county of origin
  mutate(county = str_sub(geoid,1,5)) %>%
  # Now zoom into nodes in one of these coastal counties of origin
  filter(county %in% mycounties$geoid) %>%
  as.data.frame() %>%
  select(geoid) %>%
  saveRDS("raw_data/coastal.rds")
```

Basic Model

Hide

```
library(lme4)
library(lmerTest)
library(texreg)

match <- read_rds("raw_data/coastal.rds")

# Let's get our simple set of county subdivisions, time invariant
dat <- read_rds("raw_data/dorian_ts_dataset.rds") %>%
  # Zoom into just nodes with at least one evacuee
  # (aka those we used to make our matching experiment)
  filter(geoid %in% match$geoid) %>%
  # Now join in the matching weights
  left_join(by = "geoid", y = match) %>%
  # Zoom into period after start of evacuation orders
  filter(day <= as.Date("2019-09-10"),
          day >= as.Date("2019-08-28")) %>%
  mutate(svi_minority = ntile(svi_minority, 10) %>% scale())

m1 <- dat %>%
  lmer(formula = log(evacuation_more) ~ avg_rainfall2 + order + hard + soft +
         bonding + bridging + linking +
         svi_socioeconomic + svi_minority +
         svi_housing_tranport + svi_household_disability +
         employees_muni + pop_density + dem_percent_precinct +
         (1 | geoid))

m2 <- dat %>%
  lmer(formula = log(evacuation_intra_more) ~ avg_rainfall2 +  order + hard + soft +
         bonding + bridging + linking +
         svi_socioeconomic + svi_minority +
         svi_housing_tranport + svi_household_disability +
         employees_muni + pop_density + dem_percent_precinct +
         (1 | geoid))
m3 <- dat %>%
  lmer(formula = log(evacuation_inter_more) ~ avg_rainfall2 +  order + hard + soft +
         bonding + bridging + linking +
         svi_socioeconomic + svi_minority +
         svi_housing_tranport + svi_household_disability +
        employees_muni + pop_density + dem_percent_precinct +
         (1 | geoid))



m4 <- dat %>%
  lmer(formula = log(evacuation_less) ~ avg_rainfall2 +  order + hard + soft +
         bonding + bridging + linking +
         svi_socioeconomic + svi_minority +
         svi_housing_tranport + svi_household_disability +
         employees_muni + pop_density + dem_percent_precinct +
         (1 | geoid))
m5 <- dat %>%
  lmer(formula = log(evacuation_intra_less) ~ avg_rainfall2 +  order + hard + soft +
         bonding + bridging + linking +
```

```
         svi_socioeconomic + svi_minority +
         svi_housing_tranport + svi_household_disability +
         employees_muni + pop_density + dem_percent_precinct +
         (1 | geoid))
m6 <- dat %>%
  lmer(formula = log(evacuation_inter_less) ~ avg_rainfall2 +  order + hard + soft +
         bonding + bridging + linking +
         svi_socioeconomic + svi_minority +
         svi_housing_tranport + svi_household_disability +
        employees_muni + pop_density + dem_percent_precinct +
        (1 | geoid))

texreg::htmlreg(
  list(m1,m2,m3, m4,m5,m6),
  file = "viz/table_A2.html",
  bold = 0.10,
  stars = c(0.001, 0.01, 0.05, 0.10),
  caption.above = TRUE,
  caption = "<b>Linear Mixed Models of Total Facebook User Movement (log-transformed)<br
>during Hurricane Dorian over 14 days (42 time-steps) (n = 4,956)</b><br><i>with random
 effects by coastal county subdivisions (n = 118)</i>",
  custom.header = list("Evacuation<br>(Movement Greater than pre-Crisis)" = 1:3,
                       "Shelter in Place<br>(Movement Less than pre-Crisis)" = 4:6),
  custom.model.names = c("Increased Total Movement", "Increased Movement Within Subdivis
ions",
                         "Increased Movement Between Subdivisions",
                         "Decreased Total Movement", "Decreased Movement Within Subdivis
ions", "Decreased Movement Between Subdivisions"),
   custom.coef.map = list(

    "soft" = "Community Preparedness Index",
    "hard" = "Hazard Mitigation Index",
    "bonding" = "Bonding Social Capital Index",
    "bridging" = "Bridging Social Capital Index",
    "linking" = "Linking Social Capital Index",
    "employees_muni" = "Government Capacity",
    "dem_percent_precinct" = "(%) Voted Democrat",
    "svi_socioeconomic" = "Socioeconomic Vulnerability",
    "svi_minority" = "Minority Status & Language",
    "svi_housing_tranport" = "Housing Type & Transportation",
    "svi_household_disability" = "Household Composition & Disability",
    "pop_density" = "Population Density",
     "avg_rainfall2" = "Rainfall (2-day lag)",
        "orderMandatory" = "Mandatory Evacuation",
    "orderVoluntary" = "Voluntary Evacuation",
    "(Intercept)" = "Constant"),

  groups = list(
    "<b>Policy Toolkits</b>" = 1:2,
    "<b>Community Resources</b>" = 3:5,
    "<b>Governance</b>" = 6:7,
    "<b>Social Vulnerability</b>" = 8:12,
    "<b>Disaster Conditions</b>" = 13:15),
  custom.gof.rows = list(
```

```
    "R2 (Standard / Conditional)" = list(m1,m2,m3,m4,m5,m6) %>%
      map(~performance::r2_nakagawa(.)[1]) %>%
      unlist(),
    "R2 (Adjusted / Marginal)" = list(m1,m2,m3,m4,m5,m6) %>%
      map(~performance::r2_nakagawa(.)[2]) %>%
      unlist(),
    "Mean VIF" = list(m1,m2,m3,m4,m5,m6) %>%
      map(~car::vif(.) %>% mean()) %>%
      unlist()),
  include.fstat = TRUE, include.rsquared = FALSE,
  include.adjrs = FALSE, include.bic = FALSE,
  include.groups = FALSE, include.aic = FALSE,
   custom.gof.names = c(
    rep(NA, 2),
    # "Var: geoid:county (Intercept)" = "Variance (Subdivisions & Counties)",
    #"Var: county (Intercept)" = "Variance (Counties)",
    "Var: geoid (Intercept)" = "Variance (Subdivisions)",
    "Var: Residual" = "Residual Variance"),
      custom.note = "*** p < 0.001, ** p < 0.01, * p < 0.05, . p < 0.10. All predictors
 rescaled to allow for comparison between beta coefficients, interpreted as the projecte
d increase in the log of movement rates per 1000 residents, given an increase in the pre
dictor by one standard deviation.")

remove(m1,m2,m3,m4,m5,m6)
```

Hide

```
mycolors <- c("#9FC58F", "#648FFF", "#785EF0","#DC267F","#FE6100","#FFB000")


dat %>%
  pivot_longer(
    cols = c(hard,soft, bonding, bridging, linking),
    names_to = "measure", values_to = "value") %>%
  mutate(measure = measure %>% recode_factor(
    "bonding" = "Bonding\nSocial Capital",
    "bridging" = "Bridging\nSocial Capital",
    "linking" = "Linking\nSocial Capital",
    "hard" = "Hazard\nMitigation",
    "soft" = "Community\nPreparedness")) %>%
  ggplot(mapping = aes(x = value, y = evacuation_inter_more, color = measure)) +
  geom_jitter(alpha = 0.5) +
  scale_y_log10() +
  geom_smooth(method = "lm", color = "black") +
  scale_color_manual(values = mycolors[-1]) +
  facet_grid(~ measure, scales = "free") +
  guides(color = FALSE) +
  theme_classic(base_size = 14) +
  theme(panel.border = element_rect(fill = NA, color = "black")) +
  labs(x = "Standard Deviations from Sample Mean",
       y = "Evacuation Between Subdivisions\nper 1000 residents (log-scaled)") +
  ggsave("viz/fig_6_scatterplot.png", dpi = 500, width = 7, height = 4)
```

Hide

```
mycolors <- c("#9FC58F", "#648FFF", "#785EF0","#DC267F","#FE6100","#FFB000")

# Increased movement between subdivisions
out <- bind_rows(
  m3 %>%
    ggpredict(terms = c("linking [-3:3]"), type = "random", ci.lvl = 0.95, back.transfor
m = TRUE) %>%
    as.data.frame() %>%
    mutate(label = "Linking Social Capital"),
  m3 %>%
    ggpredict(terms = c("soft [-3:3]"), type = "random", ci.lvl = 0.95, back.transform =
TRUE) %>%
    as.data.frame() %>%
    mutate(label = "Community Preparedness"))

out %>%
  ggplot(mapping = aes(x = x, y = predicted, ymin = conf.low, ymax = conf.high,
                       fill = label, color = label)) +
  geom_line() +
  geom_ribbon(alpha = 0.5) +
  #scale_y_log10() +
  facet_wrap(~label, scales = "free_y") +
  scale_fill_manual(values = mycolors[c(3,6)]) +
  scale_color_manual(values = mycolors[c(3,6)]) +
  guides(fill = FALSE, color = FALSE) +
  theme_classic(base_size = 14) +
  theme(panel.border = element_rect(fill = NA, color = "black")) +
  labs(x = "Standard Deviations from the Mean",
       y = "Evacuation Between Subdivisions\nper 1,000 residents") +
  ggsave("viz/marginal.png", dpi = 500, width = 5, height = 3.5)
```

Hide

```
# Increased movement between subdivisions
out <- bind_rows(
  m1 %>%
    ggpredict(terms = c("bonding [-3:3]"), type = "random", ci.lvl = 0.95, back.transfor
m = TRUE) %>%
    as.data.frame() %>%
    mutate(label = "Bonding",
           type = "Evacuation\n(Total Increased Movement)"),
  m1 %>%
    ggpredict(terms = c("linking [-3:3]"), type = "random", ci.lvl = 0.95, back.transfor
m = TRUE) %>%
    as.data.frame() %>%
    mutate(label = "Linking",
           type = "Evacuation\n(Total Increased Movement)"),

  m6 %>%
    ggpredict(terms = c("bonding [-3:3]"), type = "random", ci.lvl = 0.95, back.transfor
m = TRUE) %>%
    as.data.frame() %>%
    mutate(label = "Bonding",
           type = "Shelter-in-Place\n(Decreased Movement\nBetween Subdivisions)"),
  m6 %>%
    ggpredict(terms = c("linking [-3:3]"), type = "random", ci.lvl = 0.95, back.transfor
m = TRUE) %>%
    as.data.frame() %>%
    mutate(label = "Linking",
           type = "Shelter-in-Place\n(Decreased Movement\nBetween Subdivisions)"))

out %>%
  ggplot(mapping = aes(x = x, y = predicted, ymin = conf.low, ymax = conf.high,
                       fill = label, color = label)) +
  geom_line() +
  geom_ribbon(alpha = 0.5) +
  #scale_y_log10() +
  facet_wrap(~type, scales = "free_y") +
  scale_fill_manual(values = mycolors[c(3,6)]) +
  scale_color_manual(values = mycolors[c(3,6)]) +
  guides(color = FALSE) +
  theme_classic(base_size = 14) +
  theme(panel.border = element_rect(fill = NA, color = "black"),
        legend.position = "bottom") +
  labs(x = "Standard Deviations from the Mean",
       y = "User Movement per 1000 residents\n(Increases or Decreases)",
       fill = "Type of\nSocial Capital") +
  ggsave("viz/fig_5_marginal_sc.png", dpi = 500, width = 5, height = 4)
```

5. Network Model

Next, we're going to repeat this experiment on the edges themselves.

Build Dataset

Hide

```
# Get observed edges in hotspot cities
nodes <- read_rds("raw_data/dorian.rds") %>%
  # Filter to hotspot cities that evacuated at least once
  activate("nodes") %>%
  filter(evacuation_more > 0) %>%
  # Zoom into just between-subdivision, positive evacuation
  activate("edges") %>%
  filter(evacuation > 0, from != to) %>%
  # Get the county of origin
  mutate(from_geoid = .N()$geoid[from]) %>%
  # Now zoom into edges that start in one of these coastal-ish counties of origin
  filter(from_geoid %in% read_rds("raw_data/coastal.rds")$geoid) %>%
  # Zoom into just the period after evacuation orders
  filter(date_time >= "2019-08-28 00:00:00 UTC",
         date_time <= "2019-09-10 16:00:00 UTC")

# Extract the full range of dates
dates <- data.frame(date_time = seq(from = lubridate::ymd_hms("2019-08-28 00:00:00"),
    to = lubridate::ymd_hms("2019-09-10 16:00:00"),
    by = "hours")) %>%
    filter(str_sub(date_time, 12,13) %in% c("00", "08", "16"))

# Now, we're going to extract the concise edgelist
el <- nodes %>%
  mutate(from_geoid = .N()$geoid[from],
         to_geoid = .N()$geoid[to]) %>%
  as.data.frame() %>%
  select(from_geoid, to_geoid, evacuation, km, date_time)


# Get rain
rain <- read_rds("raw_data/covariates/rainfall/dorian_rainfall_csub.rds") %>%
  # Next, we're going to make a few lags
  arrange(geoid, day) %>%
  group_by(geoid) %>%
  mutate(avg_rainfall1 = lag(avg_rainfall, 1),
         avg_rainfall2 = lag(avg_rainfall, 2),
         avg_rainfall3 = lag(avg_rainfall, 3),
         avg_rainfall4 = lag(avg_rainfall, 4),
         avg_rainfall5 = lag(avg_rainfall, 5)) %>%
  ungroup() %>%
  select(geoid, day, avg_rainfall2)

# Recreate our nodes dataset of hotspot cities of interest
nodes <- read_rds("raw_data/dorian.rds") %>%
  # Filter to hotspot cities that evacuated at least once
  activate("nodes") %>%
  filter(evacuation_more > 0) %>%
  # Now zoom into edges that start in one of these coastal-ish counties of origin
  activate("edges") %>%
  mutate(from_geoid = .N()$geoid[from]) %>%
  filter(from_geoid %in% read_rds("raw_data/coastal.rds")$geoid) %>%
  # Add these node traits
```

```
  activate("nodes") %>%
  as.data.frame() %>%
  # Let's create indices of each of these measures
  mutate(hard = (scale(projects) + scale(properties) + scale(amount_paid)) / 3,
         soft = (scale(programs) + scale(trainings_per_year) +
                   scale(classes) + scale(trainees)) / 4) %>%
  # Join in area
  left_join(by = "geoid",
            y = read_rds("shapes/csub.rds") %>%
              as.data.frame() %>%
              select(geoid, area_land)) %>%
  mutate(pop_density = pop / area_land) %>%
  mutate_at(vars(contains("svi"), median_income,
                 income_inequality, pop_unemployed,
                 social_capital, bonding, bridging, linking),
            funs(if_else(is.na(.), mean(., na.rm = TRUE), .))) %>%
  ungroup() %>%
  select(geoid, hard, soft, social_capital, bonding, bridging, linking,
         svi_socioeconomic, svi_minority, svi_housing_tranport, svi_household_disabilit
y,
         employees_muni, pop_density, pop, dem_percent_precinct, pop_black, pop_hisplat)

# We're going to add back in zeros
expand_grid(
  el %>% select(from_geoid, to_geoid),
  dates) %>%
  # Join in distance
  left_join(by = c("from_geoid", "to_geoid"),
            y = el %>%
              select(from_geoid, to_geoid, km) %>%
              distinct()) %>%
  # Join in evacuation
  left_join(by = c("from_geoid", "to_geoid", "date_time"),
            y = el %>%
              select(from_geoid, to_geoid, evacuation, date_time)) %>%
  # Fill in the new rows with zeros, since they represent times when no evacuees flowed
 here.
    mutate(evacuation = if_else(is.na(evacuation),
                                true = 0, false = as.numeric(evacuation))) %>%
  # Join in daily rainfall
  mutate(day = str_sub(date_time, 1,10) %>% lubridate::ymd()) %>%
  left_join(by = c("from_geoid" = "geoid", "day"),
            y = rain %>% rename(from_avg_rainfall2 = avg_rainfall2)) %>%
  left_join(by = c("to_geoid" = "geoid", "day"),
            y = rain %>% rename(to_avg_rainfall2 = avg_rainfall2)) %>%
  # Join in the following source traits
  left_join(by = c("from_geoid" = "geoid"), y = nodes) %>%
  # Join in just a few destination traits
  left_join(by = c("to_geoid" = "geoid"),
            y = nodes %>% select(geoid, to_hard = hard, to_soft = soft,
                                 to_pop_density = pop_density,
                                 to_svi_socioeconomic = svi_socioeconomic,
                                 to_bonding = bonding, to_bridging = bridging,
                                 to_linking = linking,
```

```
         to_svi_minority = svi_minority,
         to_svi_housing_transport = svi_housing_tranport,
         to_svi_household_disability = svi_household_disability,
         to_employees_muni = employees_muni, to_pop = pop,
         to_dem_percent_precinct = dem_percent_precinct,
         to_pop_black = pop_black, to_pop_hisplat = pop_hisplat)) %>%
  # Cut any rows where pop_black is missing - these lack census data for many traits
  # and are not comparalbe.
  filter(!is.na(pop_black)) %>%
  distinct() %>%
  saveRDS("raw_data/county/grid.rds")


remove(dates, el, nodes, mycounties, counties, rain)


mycounties <- read_rds("shapes/counties.rds") %>%
  mutate(order = case_when(
    # Florida
    str_sub(geoid, 1,2) == 12 & name %in% c("Martin", "Palm Beach", "Brevard",
                                            "St. Lucie", "St. Johns") ~ "Mandatory",
    str_sub(geoid, 1,2) == 12 &name %in% c("Flagler", "Nassau",
                                           "Osceola", "Glades", "Hendry",
                                           "Okeechobee", "Highlands") ~ "Voluntary",
    # Georgia
    str_sub(geoid, 1,2) == 13 & name %in% c("Bryan", "Camden", "Chatham",
                                            "Glynn", "Liberty", "McIntosh") ~ "Mandator
y",
    TRUE ~ NA_character_) %>% factor(levels = c("Mandatory", "Voluntary", "None"))) %>%
  # Now zoom into just the coastal counties I have classified
  filter(!is.na(order)) %>%
  as.data.frame() %>%
  select(geoid, name, order)

# Import the most-comparable county subdivisions
read_rds("raw_data/county/grid.rds") %>%
  # Population control our outcome
  mutate(evacuation = evacuation / pop * 100000) %>%
# Get the second smallest value and divide by two
# Make the adjustment to allow for log-transformation
  mutate(evacuation = if_else(evacuation == 0,
                              true = sort(unique(.$evacuation))[2] / 2,
                              false = evacuation)) %>%
  # Impute the mean for rainfall
  mutate_at(vars(contains("rainfall")),
            funs(if_else(is.na(.), mean(., na.rm = TRUE), .))) %>%
    # Rescale predictors
  mutate_at(vars(contains("rainfall"),km, social_capital, bonding, bridging, linking,
                 contains("svi"), hard, soft, contains("pop"),
                 employees_muni,
                 contains("democrat"), contains("republican"), dem_percent_precinct),
            funs(scale(.))) %>%
  # Identify from_county
  mutate(from_county = str_sub(from_geoid, 1,5),
```

```
       to_county = str_sub(to_geoid, 1,5)) %>%
# Join in the evacuation order of that county
left_join(by = c("from_county" = "geoid"), y = mycounties) %>%
left_join(by = c("to_county" = "geoid"), y = mycounties %>%
              rename(to_order = order)) %>%
mutate_at(vars(to_order, order), funs(if_else(is.na(.), "None", as.character(.)) %>%
         factor() %>% relevel(ref = "None"))) %>%
saveRDS("raw_data/county/matched.rds")
```

Model

Hide

```
# Import our observed evacuation pathways between counties
dat <- read_rds("raw_data/county/matched.rds")

# Overall (August 28 - Sept 10)
m1 <- dat %>%
  mutate_at(vars(bonding, svi_minority), funs(ntile(., 4) %>% scale())) %>%
  lmer(formula = log(evacuation) ~ km + order +
         to_avg_rainfall2 + to_order + #to_hard + to_soft +
         to_svi_socioeconomic + to_svi_minority + to_pop_density +
         from_avg_rainfall2 + hard + soft +
         svi_socioeconomic + svi_minority +
         svi_household_disability + svi_housing_tranport +
         bonding + bridging + linking + employees_muni + pop_density +
         dem_percent_precinct +
         (1 | from_geoid) +
         (1 | from_geoid:to_geoid))

m2 <- dat %>%
  filter(day > "2019-09-02") %>%
  mutate_at(vars(bonding, svi_minority), funs(ntile(., 4) %>% scale())) %>%
  lmer(formula = log(evacuation) ~ km + order +
         to_avg_rainfall2 + to_order + #to_hard + to_soft +
         to_svi_socioeconomic + to_svi_minority + to_pop_density +
         from_avg_rainfall2 + hard + soft +
         svi_socioeconomic + svi_minority +
         svi_household_disability + svi_housing_tranport +
         bonding + bridging + linking + employees_muni + pop_density +
         dem_percent_precinct +
         (1 | from_geoid) +
         (1 | from_geoid:to_geoid))


m3 <- dat %>%
  filter(day > "2019-09-06") %>%
  mutate_at(vars(bonding, svi_minority), funs(ntile(., 4) %>% scale())) %>%
  lmer(formula = log(evacuation) ~ km + order +
         to_avg_rainfall2 + to_order + #to_hard + to_soft +
         to_svi_socioeconomic + to_svi_minority + to_pop_density +
         from_avg_rainfall2 + hard + soft +
         svi_socioeconomic + svi_minority +
         svi_household_disability + svi_housing_tranport +
         bonding + bridging + linking + employees_muni + pop_density +
         dem_percent_precinct +
         (1 | from_geoid) +
         (1 | from_geoid:to_geoid))


texreg::htmlreg(
  list(m1,m2,m3),
  file = "viz/table_A3.html",
  bold = 0.10,
  stars = c(0.001, 0.01, 0.05, 0.10),
  caption.above = TRUE,
```

```
  caption = "<b>Dyadic Linear Mixed Models of Total Facebook User Movement (log-transformed)<br>leading up to Hurricane Dorian (August 28 - 10) over 14 days (42 time-steps) (n = 68,460)</b><br><br><i>with random effects by destination subdivision (n = 163), nested by origin subdivision (n = 118)</i><br><b>Among Hotspots with Any Evacuation (Movement Greater than pre-Crisis Between Subdivisions)</b>",
  custom.header = list("Leading up to Landfall<br>(n = 68,460)" = 1,
                       "Post Evacuation Order<br>(n = 39,120)" = 2,
                       "Landfall<br>(n = 19,560)" = 3),
  custom.model.names = c("8/28 - 9/10",
                         "9/2 - 9/10",
                         "9/6 - 9/10"),

  custom.coef.map = list(
    # Source Traits
    "soft" = "Community Preparedness",
    "hard" = "Hazard Mitigation",
    "bonding" = "Bonding Social Capital",
    "bridging" = "Bridging Social Capital",
    "linking" = "Linking Social Capital",
    "employees_muni" = "Government Capacity",
    "dem_percent_precinct" = "(%) Voted Democrat",
    "svi_socioeconomic" = "Socioeconomic Vulnerability",
    "svi_minority" = "Minority Status & Language",
    "svi_housing_tranport" = "Housing Type & Transportation",
    "svi_household_disability" = "Household Composition & Disability",
    "pop_density" = "Population Density",
    # Disaster Conditions
    "from_avg_rainfall2" = "Rainfall (2-day lag)",
    "orderMandatory" = "Mandatory Evacuation",
    "orderVoluntary" = "Voluntary Evacuation",
    # Destination Traits
    "to_soft" = "Community Preparedness (Dest.)",
    "to_hard" = "Hazard Mitigation (Dest.)",
    "to_svi_socioeconomic" = "Socioeconomic Vulnerability (Dest.)",
    "to_svi_minority" = "Minority Status & Language (Dest.)",
    "to_pop_density" = "Population Density (Dest.)",
    "to_avg_rainfall2" = "Rainfall (2-day lag) (Dest.)",
    "to_orderMandatory" = "Mandatory Evacuation (Dest.)",
    "to_orderVoluntary" = "Voluntary Evacuation (Dest.)",
    "km" = "Distance (km)",
    "(Intercept)" = "Constant"),
  groups = list(
    # Source
    "<b>Policy Toolkits (Source)</b>" = 1:2,
    "<b>Community Resources (Source)</b>" = 3:5,
    "<b>Governance (Source)</b>" = 6:7,
    "<b>Social Vulnerability (Source)</b>" = 8:12,
    "<b>Disaster Conditions (Source)</b>" = 13:15,
    # Destination
    "<b>Social Vulnerability (Destination)</b>" = 16:18,
    "<b>Disaster Conditions (Destination)</b>" = 19:22),
  custom.gof.rows = list(
    "R2 (Standard / Conditional)" = list(m1,m2,m3) %>%
      map(~performance::r2_nakagawa(.)[1]) %>%
```

```
      unlist(),
    "R2 (Adjusted / Marginal)" = list(m1,m2,m3) %>%
      map(~performance::r2_nakagawa(.)[2]) %>%
      unlist(),
    "Mean VIF" = list(m1,m2,m3) %>%
      map(~car::vif(.) %>% mean()) %>%
      unlist()),
  include.fstat = TRUE, include.rsquared = FALSE,
  include.adjrs = FALSE, include.bic = FALSE,
  include.groups = FALSE, include.aic = FALSE,
  custom.gof.names = c(
    rep(NA, 2),
    "Var: from_geoid:to_geoid (Intercept)" = "Variance (Destination City, Source City)",
    "Var: from_geoid (Intercept)" = "Variance (Source City)",
    "Var: Residual" = "Residual Variance"),
  custom.note = "*** p < 0.001, ** p < 0.01, * p < 0.05, . p < 0.10. All predictors resc
aled to allow for comparison between beta coefficients, interpreted as the projected inc
rease in the log of movement rates per 1000 residents, given an increase in the predicto
r by one standard deviation.")
```

Map

Hide

```
# Get pairs of cities, classified categorically by date range
dat <- read_rds("raw_data/county/matched.rds") %>%
  mutate(class = case_when(
    date_time >= "2019-08-28" & date_time < "2019-09-02" ~ "8/28 - 9/1",
    date_time >= "2019-09-02" & date_time < "2019-09-06" ~ "9/2 - 9/5",
    date_time >= "2019-09-06" & date_time < "2019-09-11" ~ "9/6 - 9/10",
    TRUE ~ "Other")) %>%
  select(from_geoid, to_geoid, date_time, class) %>%
  distinct()

# Key points
# Identify the source cities
key <- bind_rows(
  data.frame(geoid = read_rds("raw_data/county/matched.rds")$from_geoid %>% unique(),
                  designation = "Source"),
  # Identify just the destination cities which are not source cities
  data.frame(geoid = unique(read_rds("raw_data/county/matched.rds")$to_geoid)[! unique(r
ead_rds("raw_data/county/matched.rds")$to_geoid)%in% unique(read_rds("raw_data/county/ma
tched.rds")$from_geoid)],
             designation = "Destination"))

# Now get counties, based on their evacuation order
mycounties <- read_rds("shapes/counties.rds") %>%
  mutate(type = case_when(
    # Florida
    str_sub(geoid, 1,2) == 12 & name %in% c("Martin", "Palm Beach", "Brevard",
                                            "St. Lucie", "St. Johns") ~ "Mandatory",
    str_sub(geoid, 1,2) == 12 &name %in% c("Flagler", "Nassau",
                                           "Osceola", "Glades", "Hendry",
                                           "Okeechobee", "Highlands") ~ "Voluntary",
    # Georgia
    str_sub(geoid, 1,2) == 13 & name %in% c("Bryan", "Camden", "Chatham",
                                            "Glynn", "Liberty", "McIntosh") ~ "Mandator
y",
    TRUE ~ NA_character_) %>% factor(levels = c("Mandatory", "Voluntary", "None"))) %>%
  # Now zoom into just the coastal counties I have classified
  filter(!is.na(type)) %>%
  as.data.frame() %>%
  select(order = type, geoid)


mysoft <- read_rds("raw_data/county/matched.rds") %>%
  select(geoid = from_geoid, soft) %>%
  distinct()

# Import network
net <- read_rds("raw_data/dorian.rds") %>%
  # Identify nodes in the model
  activate("nodes") %>%
  # Join in the designation for key source & destination nodes
  left_join(by = "geoid", y = key) %>%
  mutate(designation = if_else(is.na(designation), "Other", designation)) %>%
  filter(designation != "Other") %>%
```

```
  # Zoom into just cities that are either sources or destinations in our coastal analysi
s
  # Identify edges by time
  activate("edges") %>%
  mutate(from_geoid = .N()$geoid[from],
         to_geoid = .N()$geoid[to]) %>%
  # Join in the date classification
  left_join(by = c("from_geoid", "to_geoid", "date_time"),
            y = dat) %>%
  mutate(class = if_else(is.na(class), "Other", class)) %>%
  filter(class != "Other") %>%
  # Population Control number of evacuees
  mutate(from_pop = .N()$pop[from]) %>%
  mutate(evacuation = evacuation / from_pop * 1000) %>%
  # Threshold network
  filter(from != to) %>%
  filter(evacuation > 0) %>%
    # Join in evacuation order
  activate("nodes") %>%
  mutate(from_county = str_sub(geoid, 1,5)) %>%
  left_join(by = c("from_county" = "geoid"), y = mycounties) %>%
  mutate(order = if_else(is.na(order), "None", as.character(order)) %>%
           factor(levels = c("Mandatory", "Voluntary", "None"))) %>%
  # Reclassify nodes by key traits
  mutate(linking = ntile(linking, 2) %>% dplyr::recode_factor("1" = "Low", "2" = "High"
)) %>%
  left_join(by = "geoid", y = mysoft) %>%
  mutate(soft = ntile(soft, 2) %>% dplyr::recode_factor("1" = "Low", "2" = "High")) %>%
  # Now identify some key edge traits
  activate("edges") %>%
  mutate(from_order = .N()$order[from]) %>%
  mutate(linking_edge = case_when(
    .N()$linking[from] == "High" & .N()$linking[to] == "High" ~ "High",
    .N()$linking[from] == "High" & .N()$linking[to] == "Low" ~ "High",
    .N()$linking[from] == "Low" & .N()$linking[to] == "High" ~ "Low",
    .N()$linking[from] == "Low" & .N()$linking[to] == "Low" ~ "Low")) %>%

  mutate(soft_edge = case_when(
    .N()$soft[from] == "High"  ~ "High",
    .N()$soft[from] == "High" ~ "High",
    .N()$soft[from] == "Low"  ~ "Low",
    .N()$soft[from] == "Low"  ~ "Low"))


# Import a background, based on counties
back <- read_rds("shapes/counties.rds") %>%
  summarize(geometry = st_union(geometry))

# Get state boundary lines
states <- read_rds("shapes/counties.rds") %>%
  group_by(state = str_sub(geoid, 1,2) %>%
             recode_factor("12" = "Florida",
                           "13" = "Georgia",
                           "01" = "Alabama")) %>%
```

```
  summarize(geometry = st_union(geometry))

# Import counties
counties <- read_rds("shapes/counties.rds")

# Import divisions
sub <- read_rds("shapes/csub.rds") %>%
  left_join(by = c("geoid" = "geoid"),
            y = net %>%
              activate("nodes") %>%
              as.data.frame() %>%
              select(geoid, rainfall_14days))

mycolors <- c("#9FC58F", "#648FFF", "#785EF0","#DC267F","#FE6100","#FFB000")


# Visualize!
ggplot() +
  # Give a nice grey background
  geom_sf(data = back, fill = NA, color = "#cfcfcf", size = 5) +
  geom_sf(data = sub, fill = "black",
          color = NA, size = 0.25)  +
  # Add some county boundaries over top
  geom_sf(data = sub, fill = NA, color = "#404040", size = 0.1, alpha = 0.5) +
  # Add some state boundaries overtop
  #geom_sf(data = mycounties, mapping = aes(fill = type)) +
  #scale_fill_manual(values = mycolors[c(2,4,6)]) +
  geom_sf(data = states, fill = NA, color = "darkgrey", size = 1) +
  # Add edges over top
  geom_sf(data = st_as_sf(net, "edges"),
          mapping = aes(size = evacuation,
                        color = factor(from_order,
                                       levels = c("None", "Voluntary", "Mandatory"))), a
lpha = 0.5) +
  scale_color_manual(values = mycolors[c(4,2,6)]) +
  # Add cities
  geom_sf(data = st_as_sf(net, "nodes") %>%
            filter(!is.na(soft)), color = "white", size = 1.5) +
   geom_sf(data = st_as_sf(net, "nodes") %>%
            filter(is.na(soft)), color = "darkgrey", size = 1.5) +
  theme_void(base_size = 14)  +
  #facet_wrap(~date_time, ncol = 10) +
  theme(legend.position = c(0.35, 0.35),
        legend.box = "vertical",
        plot.subtitle = element_text(hjust = 0.5),
        plot.margin = unit(x = c(0,0,0,0), "cm")) +
  # Break up by date-ranges
  #facet_wrap(~class, ncol = 4) +
  #guides(color = FALSE) +
  labs(size = "Evacuees per\n1,000 residents", color = "Source County\nEvacuation Order"
,
       subtitle = "Coastal Evacuation Pathways Every 8 Hours\nLeading up to Hurricane Do
rian (n = 68,460)\nfrom Hotspot Coastal County Subdivisions (n = 118)\nto Destination Su
```

```
bdivisions (n = 163) over 14 days") +
  ggsave("viz/fig_4_coastal_network.png", dpi = 500, width = 4.5, height = 6)
```

6. Case Study

Map

Hide

```
wgs <- "+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs"

# Identify just places with at least 1 evacuee total
nodes <- st_as_sf(read_rds("raw_data/dorian.rds"), "nodes") %>%
  filter(evacuation_more > 0) %>%
  as.data.frame() %>%
  select(geoid) %>%
  unlist()

# Now get counties, based on their evacuation order
mycounties <- read_rds("shapes/counties.rds") %>%
  filter(name %in% "Palm Beach")

# Get county subdivisions, focusing on the coast
sub <- read_rds("shapes/csub.rds") %>%
  filter(str_sub(geoid, 1,5) %in% mycounties$geoid) %>%
  filter(!name %in% c("Belle Glade-Pahokee", "Glades", "Western Community")) %>%
  left_join(by = "geoid", y = read_rds("raw_data/dorian_ts_dataset.rds") %>%
  select(geoid, soft) %>%
  distinct()) %>%
  mutate(soft = scale(soft))

# Zoom into tracts overlapping these subdivisions
mytracts <- read_rds("shapes/tracts.rds") %>%
  filter(str_sub(geoid, 1,5) %in% mycounties$geoid) %>%
  st_join(sub %>% select(name, geometry)) %>%
  filter(!is.na(name)) %>%
  as.data.frame()

# Get traits for these tracts, paired with polygons
tracts <- read_rds("shapes/tracts.rds") %>%
  # Zoom into just the coastal tracts
  filter(geoid %in% mytracts$geoid) %>%
  # Exclude the two enormous rural tracts
  mutate(area_land = st_area(geometry)) %>%
  filter(!geoid %in% c("12099980400", "12099007912"))  %>%
  # Join in key social capital
  left_join(by = "geoid",
            # Let's rescale versus the state mean
            y = read_rds("raw_data/covariates/sci_census_tracts_2021_01_28.rds") %>%
              filter(year == 2018) %>%
              filter(str_sub(geoid, 1,5) %in% mycounties$geoid) %>%
              mutate_at(vars(bonding, bridging, linking),
                        funs(scale(.))) %>%
              select(geoid, bonding, bridging, linking)) %>%
  # Join in demographics
  left_join(by = "geoid",
            y = read_rds("raw_data/covariates/census_tract.rds")) %>%
  # Join in partisanship
  left_join(by = "geoid",
            y = read_rds("raw_data/covariates/precincts_tracts.rds")) %>%
  mutate_at(vars(pop_hisplat, pop_black), funs(pop_hisplat*100)) %>%
  # Join in SVI
```

```
  left_join(by = c("geoid" = "fips"),
            y = read_rds("raw_data/covariates/svi_tract.rds") %>%
              filter(state %in% c("FL","GA","AL")) %>%
              select(fips, contains("svi"))) %>%
  mutate_at(vars(contains("svi")), funs(scale(.)))



# Let's take our original model
dat <- read_rds("raw_data/dorian_ts_dataset.rds") %>%
  filter(geoid %in% nodes) %>%
  # Zoom into just Palm Beach subdivisions
  filter(county %in% mycounties$geoid) %>%
  # Zoom into before/during the crisis
  filter(day >= "2019-09-02",
         day <= "2019-09-10") %>%
  # And let's tally up evacuees
  group_by(geoid) %>%
  summarize(evacuation_intra_more = sum(evacuation_intra_more, na.rm = TRUE),
            evacuation_inter_more = sum(evacuation_inter_more, na.rm = TRUE),
            evacuation_more = sum(evacuation_more, na.rm = TRUE))


net <- read_rds("raw_data/dorian.rds") %>%
  activate("nodes") %>%
  select(geoid, bonding, bridging, linking, pop, median_income,
         pop_hisplat, pop_black, geometry) %>%
  filter(str_sub(geoid, 1,5) %in% mycounties$geoid) %>%
  filter(geoid %in% sub$geoid) %>%
  # Join in evacuation tally
  left_join(by = "geoid", y = dat)


# Download hydrography
wa <- tigris::area_water(state = "FL", county = "Palm Beach", year = 2019) %>%
  st_as_sf() %>%
  st_transform(crs = wgs) %>%
  st_join(sub) %>%
  filter(!is.na(name))
wl <- tigris::linear_water(state = "FL", county = "Palm Beach", year= 2019) %>%
  st_as_sf() %>%
  st_transform(crs = wgs) %>%
  st_join(sub) %>%
  filter(!is.na(name)) %>%
  st_crop(sub)
```

Hide

```
# Let's create a series of maps, and bind them together
soft <- ggplot() +
  geom_sf(data = tracts, color = "darkgrey", fill = NA, size = 5) +
  geom_sf(data = sub, mapping = aes(fill = soft)) +
  geom_sf(data = tracts, fill = NA,
          color = "darkgrey", color = 0.5) +
  scale_fill_viridis(option = "plasma") +
  # Get water
  geom_sf(data = wa, fill = "steelblue", color = "steelblue", alpha = 0.20) +
  geom_sf(data = wl, color = "steelblue", alpha = 0.2) +

  geom_sf(data = sub, color = "black", fill = NA, color = 2) +
  #geom_sf_label(data = sub, mapping = aes(label = name)) +
  geom_sf(data = st_as_sf(net, "nodes"),
          mapping = aes(size = evacuation_inter_more + 5),
          fill = NA, color = "white", show.legend = TRUE) +
    geom_sf(data = st_as_sf(net, "nodes"),
          mapping = aes(size = evacuation_inter_more),
          fill = NA, color = "black") +
  #geom_sf_label(data = sub, mapping = aes(label = str_replace_all(name, " ", "\n")),
  #              nudge_x = -0.05, nudge_y = -0.01, size = 2) +
  theme_void(base_size = 14) +
  theme(plot.subtitle = element_text(hjust = 0.5),
        legend.position = "bottom",
        plot.margin = unit(c(0,0,0,0), "cm")) +
    guides(fill = guide_colorbar(barwidth = 10, barheight = 0.5)) +
labs(size = "Evacuation Between\nSubdivisions\nper 1000 residents",
       fill = "Std. Dev. from\nPalm Beach Mean",
       subtitle = "Community Preparedness\nIndex") +
    scale_size_continuous(range = c(0, 10))

linking <- ggplot() +
  geom_sf(data = tracts, color = "darkgrey", fill = NA, size = 5) +
  geom_sf(data = tracts, mapping = aes(fill = linking),
          color = "darkgrey", color = 0.5) +
    scale_fill_viridis(option = "plasma") +

  #  scale_fill_gradient2(low = "#DC267F", high = "#648FFF",
  #                       midpoint = 50, mid = "white",na.value = "grey") +
  # Get water
  geom_sf(data = wa, fill = "steelblue", color = "steelblue", alpha = 0.20) +
  geom_sf(data = wl, color = "steelblue", alpha = 0.2) +

  geom_sf(data = sub, color = "black", fill = NA, color = 2) +
  #geom_sf_label(data = sub, mapping = aes(label = name)) +
  geom_sf(data = st_as_sf(net, "nodes"),
          mapping = aes(size = evacuation_inter_more + 5),
          fill = NA, color = "white") +
    geom_sf(data = st_as_sf(net, "nodes"),
          mapping = aes(size = evacuation_inter_more),
          fill = NA, color = "black") +
  theme_void(base_size = 14) +
```

```
  theme(plot.subtitle = element_text(hjust = 0.5),
        legend.position = "bottom",
        plot.margin = unit(c(0,0,0,0), "cm")) +
    guides(fill = guide_colorbar(barwidth = 10, barheight = 0.5)) +
labs(size = "Evacuation Between\nSubdivisions\nper 1000 residents",
       subtitle = "Linking\nSocial Capital",
       fill = "Std. Dev. from\nPalm Beach Mean") +
    scale_size_continuous(range = c(0, 10))


svi <- ggplot() +
  geom_sf(data = tracts, color = "darkgrey", fill = NA, size = 5) +
  geom_sf(data = tracts, mapping = aes(fill = svi_socioeconomic),
          color = "darkgrey", color = 0.5) +
    scale_fill_viridis(option = "plasma") +

  #  scale_fill_gradient2(low = "#DC267F", high = "#648FFF",
  #                       midpoint = 50, mid = "white",na.value = "grey") +
  # Get water
  geom_sf(data = wa, fill = "steelblue", color = "steelblue", alpha = 0.20) +
  geom_sf(data = wl, color = "steelblue", alpha = 0.2) +

  geom_sf(data = sub, color = "black", fill = NA, color = 2) +
  #geom_sf_label(data = sub, mapping = aes(label = name)) +
  geom_sf(data = st_as_sf(net, "nodes"),
          mapping = aes(size = evacuation_inter_more + 5),
          fill = NA, color = "white") +
    geom_sf(data = st_as_sf(net, "nodes"),
          mapping = aes(size = evacuation_inter_more),
          fill = NA, color = "black") +
  theme_void(base_size = 14) +
  theme(plot.subtitle = element_text(hjust = 0.5),
        legend.position = "bottom",
        plot.margin = unit(c(0,0,0,0), "cm")) +
    guides(fill = guide_colorbar(barwidth = 10, barheight = 0.5)) +
labs(size = "Evacuation Between\nSubdivisions\nper 1000 residents",
       subtitle = "Socioeconomic Vulnerability\n(Social Vulnerability Index)",
       fill = "Std. Dev. from\nPalm Beach Mean)") +
    scale_size_continuous(range = c(0, 10))



minority <- ggplot() +
  geom_sf(data = tracts, color = "darkgrey", fill = NA, size = 5) +
  geom_sf(data = tracts, mapping = aes(fill = svi_minority),
          color = "darkgrey", color = 0.5) +
    scale_fill_viridis(option = "plasma") +
 # Get water
  geom_sf(data = wa, fill = "steelblue", color = "steelblue", alpha = 0.20) +
  geom_sf(data = wl, color = "steelblue", alpha = 0.2) +

  geom_sf(data = sub, color = "black", fill = NA, color = 2) +
  #geom_sf_label(data = sub, mapping = aes(label = name)) +
  geom_sf(data = st_as_sf(net, "nodes"),
```

```
          mapping = aes(size = evacuation_inter_more + 5),
          fill = NA, color = "white") +
    geom_sf(data = st_as_sf(net, "nodes"),
          mapping = aes(size = evacuation_inter_more),
          fill = NA, color = "black") +
  theme_void(base_size = 14) +
  theme(plot.subtitle = element_text(hjust = 0.5),
        legend.position = "bottom",
        plot.margin = unit(c(0,0,0,0), "cm")) +
    guides(fill = guide_colorbar(barwidth = 10, barheight = 0.5)) +
  labs(size = "Evacuation Between\nSubdivisions\nper 1000 residents",
       subtitle = "Minority Status & Language\n(Social Vulnerability Index)",
       fill = "Std. Dev. from Palm Beach Mean") +
    scale_size_continuous(range = c(0, 10))

library(ggpubr)
ggpubr::ggarrange(soft, linking, svi,minority, ncol = 4,
                  common.legend = TRUE, legend = "bottom") +
  ggsave("viz/fig_8_case_studies.png", dpi = 500, width = 10, height = 6)
```

Map with Names

Hide

```
wgs <- "+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs"

# Identify just places with at least 1 evacuee total
nodes <- st_as_sf(read_rds("raw_data/dorian.rds"), "nodes") %>%
  filter(evacuation_more > 0) %>%
  as.data.frame() %>%
  select(geoid) %>%
  unlist()

# Now get counties, based on their evacuation order
mycounties <- read_rds("shapes/counties.rds") %>%
  filter(name %in% "Palm Beach")

# Get county subdivisions, focusing on the coast
sub <- read_rds("shapes/csub.rds") %>%
  filter(str_sub(geoid, 1,5) %in% mycounties$geoid) %>%
  filter(!name %in% c("Belle Glade-Pahokee", "Glades", "Western Community")) %>%
  left_join(by = "geoid", y = read_rds("raw_data/dorian_ts_dataset.rds") %>%
  select(geoid, soft) %>%
  distinct()) %>%
  mutate(soft = scale(soft))

# Zoom into tracts overlapping these subdivisions
mytracts <- read_rds("shapes/tracts.rds") %>%
  filter(str_sub(geoid, 1,5) %in% mycounties$geoid) %>%
  st_join(sub %>% select(name, geometry)) %>%
  filter(!is.na(name)) %>%
  as.data.frame()

# Get traits for these tracts, paired with polygons
tracts <- read_rds("shapes/tracts.rds") %>%
  # Zoom into just the coastal tracts
  filter(geoid %in% mytracts$geoid) %>%
  # Exclude the two enormous rural tracts
  mutate(area_land = st_area(geometry)) %>%
  filter(!geoid %in% c("12099980400", "12099007912"))  %>%
  # Join in key social capital
  left_join(by = "geoid",
            # Let's rescale versus the state mean
            y = read_rds("raw_data/covariates/sci_census_tracts_2021_01_28.rds") %>%
              filter(year == 2018) %>%
              filter(str_sub(geoid, 1,5) %in% mycounties$geoid) %>%
              mutate_at(vars(bonding, bridging, linking),
                        funs(scale(.))) %>%
              select(geoid, bonding, bridging, linking)) %>%
  # Join in demographics
  left_join(by = "geoid",
            y = read_rds("raw_data/covariates/census_tract.rds")) %>%
  # Join in partisanship
  left_join(by = "geoid",
            y = read_rds("raw_data/covariates/precincts_tracts.rds")) %>%
  mutate_at(vars(pop_hisplat, pop_black), funs(pop_hisplat*100)) %>%
  # Join in SVI
```

```
  left_join(by = c("geoid" = "fips"),
            y = read_rds("raw_data/covariates/svi_tract.rds") %>%
              filter(state %in% c("FL","GA","AL")) %>%
              select(fips, contains("svi"))) %>%
  mutate_at(vars(contains("svi")), funs(scale(.)))

# Download hydrography
wa <- tigris::area_water(state = "FL", county = "Palm Beach", year = 2019) %>%
  st_as_sf() %>%
  st_transform(crs = wgs) %>%
  st_join(sub) %>%
  filter(!is.na(name))
wl <- tigris::linear_water(state = "FL", county = "Palm Beach", year= 2019) %>%
  st_as_sf() %>%
  st_transform(crs = wgs) %>%
  st_join(sub) %>%
  filter(!is.na(name)) %>%
  st_crop(sub)
```

Hide

```

# Let's create a series of maps, and bind them together
ggplot() +
  geom_sf(data = tracts, color = "darkgrey", fill = NA, size = 5) +
  geom_sf(data = sub, fill = "white") +
  geom_sf(data = tracts, mapping = aes(fill = pop / (as.numeric(area_land) / 1e+6)),
          color = "darkgrey", color = 0.5) +
  scale_fill_viridis(option = "plasma") +
  # Get water
  geom_sf(data = wa, fill = "steelblue", color = "steelblue", alpha = 0.20) +
  geom_sf(data = wl, color = "steelblue", alpha = 0.20) +

  geom_sf(data = sub, color = "black", fill = NA, color = 2) +
  geom_sf_label(data = sub, mapping = aes(label = str_replace_all(name, "-", "\n")),
                nudge_x = c(0.05, -0.05), nudge_y = -0.01, size = 3) +
  theme_void(base_size = 14) +
  theme(plot.subtitle = element_text(hjust = 0.5),
        legend.position = "right",
        plot.margin = unit(c(0,0,0,0), "cm")) +
    guides(fill = guide_colorbar(barwidth = 0.5, barheight = 10)) +
labs(size = "Evacuation Between\nSubdivisions\nper 1000 residents",
       fill = "Population\nDensity\n(thousands)\nper Square\nKilometer",
       subtitle = "Palm Beach County\nCoastal Subdivisions") +
    scale_size_continuous(range = c(0, 10)) +
  ggsave("viz/fig_7_case_study_names.png", dpi = 500, width = 4, height = 5.5)
```

Table

Hide

```
# Now get counties, based on their evacuation order
mycounties <- read_rds("shapes/counties.rds") %>%
  filter(name %in% "Palm Beach")


# Let's take our original model
evac <- read_rds("raw_data/dorian_ts_dataset.rds") %>%
  # Zoom into just Palm Beach subdivisions
  filter(county %in% mycounties$geoid) %>%
  # Zoom into before/during the crisis
  filter(day >= "2019-09-02",
         day <= "2019-09-10") %>%
  # And let's tally up evacuees
  group_by(geoid) %>%
  summarize(evacuation_intra_more = sum(evacuation_intra_more, na.rm = TRUE),
            evacuation_inter_more = sum(evacuation_inter_more, na.rm = TRUE),
            evacuation_more = sum(evacuation_more, na.rm = TRUE))

# Get county subdivisions, focusing on the coast
sub <- read_rds("shapes/csub.rds") %>%
  filter(str_sub(geoid, 1,5) %in% mycounties$geoid) %>%
  filter(!name %in% c("Belle Glade-Pahokee", "Glades", "Western Community"))  %>%
  left_join(by = "geoid", y = read_rds("raw_data/dorian.rds") %>%
              activate("nodes") %>%
              as.data.frame()) %>%
  as.data.frame() %>%
  # Get needed detailed variables
  select(geoid, name, pop,  median_income, #svi_socioeconomic,
         pop_black, pop_hisplat, #svi_minority,
         bonding, bridging, linking,
         dem_percent_precinct, employees_muni,
         trainees, amount_paid) %>%
  # Rescale social capital as z-scores, because otherwise no one will know what it means
  mutate_at(vars(bonding, bridging, linking),
            funs(scale(.))) %>%
  # Now fix decimal places;
  # Convert race/ethnicity measures to full percentages, no decimals
  mutate_at(vars(pop_black, pop_hisplat), funs(round(.*100, 0))) %>%
  # Measure population in thousands
  mutate_at(vars(pop, median_income), funs(round(. / 1000, 0))) %>%
  # Round our new z-scores to 1 significant digit
  mutate_at(vars(bonding, bridging, linking),
            funs(round(., 1))) %>%
  # Join in evacuation
  left_join(by = "geoid", y = evac) %>%
  # Round to 0 digits
  mutate_at(vars(dem_percent_precinct, trainees, amount_paid, contains("evacuation")),
            funs(round(.,0))) %>%
  # Convert municipal employees to per 100,000 residents, with 1 decimal place
  mutate(employees_muni = round(employees_muni*100, 1)) %>%
  # Pivot into a tidy data.frame
  pivot_longer(
    cols = -c(geoid, name),
```

```
    names_to = "variable",
    values_to = "value", values_transform = list(value = as.character)) %>%
  # And classify each variable value as above or below the median
  group_by(variable) %>%
  mutate(level = ntile(as.numeric(value), 2) %>%
           dplyr::recode("1" = "Low", "2" = "High")) %>%
  ungroup() %>%
  mutate(name = name %>% str_replace_all(pattern = "-", "/\n")) %>%
  mutate(type = case_when(
    variable %in% c("evacuation_more",
                    "evacuation_intra_more",
                    "evacuation_inter_more") ~ "Evacuees 9/2 - 9/10\n(per 1000 resident
s)",
    variable %in% c("pop", "svi_socioeconomic", "median_income",
                    "svi_minority", "pop_hisplat", "pop_black") ~ "Social\nVulnerabilit
y",
    variable %in% c("bonding", "bridging", "linking") ~ "Social Capital\n(Z-score)",
    variable %in% c("amount_paid", "trainees") ~ "Policy Tools Used\n(per 1000 resident
s)",
    variable %in% c("employees_muni", "dem_percent_precinct") ~ "Governance",
    TRUE ~ NA_character_) %>%
      factor(levels = c("Evacuees 9/2 - 9/10\n(per 1000 residents)",
                        "Social Capital\n(Z-score)",  "Policy Tools Used\n(per 1000 resi
dents)",
                        "Social\nVulnerability", "Governance"))) %>%
  mutate(variable = variable %>% recode_factor(
    "evacuation_more" = "Total",
    "evacuation_intra_more" = "Within Subdivisions",
    "evacuation_inter_more" = "Between Subdivisions",
    "pop" = "Population (thousands)",
    "svi_socioeconomic" = "Socioeconomic\nVulnerability Index",
    "median_income" = "Median Income (thousands)",
    "svi_minority" = "Minority Status\n& Language Index",
    "pop_hisplat" = "% Hispanic/Latino",
    "pop_black" = "% Black",
    "bonding" = "Bonding",
    "bridging" = "Bridging",
    "linking" = "Linking",
    "trainees" = "Community Emergency\nResponse Trainees",
    "amount_paid" = "Hazard Mitigation\nSpending",
    "employees_muni" = "Municipal Employees\nper 100,000 residents",
    "dem_percent_precinct" = "% Voted Democrat") ) %>%
  mutate(indicator = if_else(variable == "Between Subdivisions" & level == "High", 1, 0
)) %>%
  # Median income is currently classified on a wealth scale,
  # but I need to flip it to vulnerability, where more wealth means "Low" vulnerabiity
  # To do so, we're going to adjust the coding like so:
  mutate(level = case_when(
    variable == "Median Income (thousands)" & level == "Low" ~ "High",
    variable == "Median Income (thousands)" & level == "High" ~ "Low",
    TRUE ~ level)) %>%
  mutate(mylevel = case_when(
    level == "Low" ~ "Low",
    TRUE ~ as.character(type)) %>%
```

```
      factor(levels = c("Evacuees 9/2 - 9/10\n(per 1000 residents)",
                        "Social Capital\n(Z-score)",  "Policy Tools Used\n(per 1000 resi
dents)",
                        "Social\nVulnerability", "Governance", "Low")))


mycolors <- c("#9FC58F", "#648FFF", "#785EF0","#DC267F","#FE6100","#FFB000")

sub %>%
  ggplot(mapping = aes(x = variable, y = reorder(name, indicator),
                       fill = mylevel, label = value)) +
  geom_tile(color = "black", mapping = aes(size = level)) +
  geom_text(mapping = aes(color = level)) +
  scale_size_manual(values = c(0.6, 0.1)) +
  scale_color_manual(values = c("black", "black")) +
  scale_fill_manual(values = c(mycolors[6:2], "white")) +
  facet_wrap(~type, scales = "free_x", strip.position = "top", ncol = 5) +
  theme_classic(base_size = 14) +
  theme(panel.border = element_rect(fill = NA, color = "black"),
        axis.text.x = element_text(angle = 30, hjust = 1),
        legend.position = "top") +
  scale_x_discrete(position = "bottom")  +
  guides(color = FALSE, fill = FALSE, size = FALSE) +
  labs(x = NULL, y = NULL, fill = "Level of Covariate\n(Above/Below Median)") +
  ggsave("viz/fig_9_case_study_table.png", dpi = 500, width = 12, height = 5)
```