Health Impact Assessment of Affordable Housing Development in Connecticut

library(tidyverse)  
library(sf)  
library(here)  
library(tidycensus)  
library(daymetr)

# read in PM2.5 daily average by census tract from EPA FAQSD  
pm\_tract <- read.delim(gzfile("2019\_pm25\_daily\_average.txt.gz"), sep = ",") %>%  
## include only Connecticut tracts - FIPS starting with 9  
 filter(str\_detect(`FIPS`, "^9")) %>%   
## include only index date - July 4, 2019  
 filter(`Date` == "2019/07/04") %>%   
## format FIPS to match GEOID format used by tidycensus get\_acs  
 mutate(GEOID = str\_pad(FIPS, 11, side = "left", pad = "0"))  
  
# read in ozone daily 8-hour maximum by census tract from EPA FAQSD  
o3\_tract <- read.delim(gzfile("2019\_ozone\_daily\_8hour\_maximum.txt.gz"), sep = ",") %>%  
## include only Connecticut tracts - FIPS starting with 9  
 filter(str\_detect(`FIPS`, "^9")) %>%   
## include only index date - July 4, 2019  
 filter(`Date` == "2019/07/04") %>%   
## format FIPS to match GEOID format used by tidycensus get\_acs  
 mutate(GEOID = str\_pad(FIPS, 11, side = "left", pad = "0"))

# read housing data from CT Open Data  
housing\_town <- read.csv("Affordable\_Housing\_by\_Town\_2011-2022.csv") %>%   
 filter(`Year` == 2019) %>%   
 mutate(`At least 10%?` = cut(`Percent.Affordable`, breaks = c(0,9.99999,100),  
 labels = c("No", "Yes"))) %>%   
 rename("TOWN\_NAME" = "Town") %>%   
 mutate(`Units Needed` = (1-(`Percent.Affordable`/10))\*`X2010.Census.Units`\*(1/9))  
  
# set units needed to 0 for towns already at or above 10% threshold  
housing\_town$`Units Needed`[housing\_town$`At least 10%` == "Yes"] = 0

# read county-level ischemic heart disease mortality rates from CDC WONDER  
ischemic <- read.delim("Underlying Cause of Death, 2018-2021, Single Race.txt", sep = "\t")[1:8,-1] %>%   
 mutate(`CNTY\_NAME` = substr(`County`, 1, nchar(`County`)-11), .before = 2)

# list of ACS variable names for reference  
vars <- load\_variables(year = 2019, dataset = "acs5", cache = TRUE)

options(tigris\_use\_cache = TRUE)  
  
# get CT census tract polygon  
ct\_tracts <- get\_acs(state = "CT", geography = "tract", variables = "B17013\_002", geometry = TRUE, year = 2019) %>%   
## join PM2.5 data  
 left\_join(., pm\_tract, by = "GEOID") %>%   
## join ozone data  
 left\_join(., o3\_tract, by = "GEOID")

## Getting data from the 2015-2019 5-year ACS

# read CT town lines  
town\_lines <- st\_read("CT Town Lines/Town\_Lines.shp")

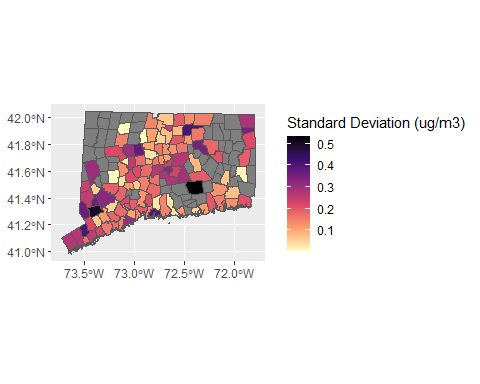
## Reading layer `Town\_Lines' from data source   
## `C:\Users\saira\OneDrive\Documents\YSPH\YCCCH\CT HIA\CT Town Lines\Town\_Lines.shp'   
## using driver `ESRI Shapefile'  
## Simple feature collection with 1431 features and 19 fields  
## Geometry type: LINESTRING  
## Dimension: XY  
## Bounding box: xmin: -73.72779 ymin: 40.95089 xmax: -71.78723 ymax: 42.05061  
## Geodetic CRS: WGS 84

# read CT town polygon  
town\_polygon <- st\_read("CT Vicinity Town Polygon/CT\_Vicinity\_Town\_Polygon.shp") %>%  
 filter(STATE\_COD == "CT") %>%   
# join housing data by town - using inner over left join to exclude area of Long Island Sound  
 inner\_join(., housing\_town, by = "TOWN\_NAME")

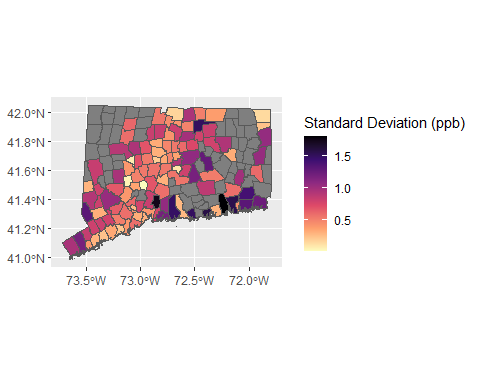
## Reading layer `CT\_Vicinity\_Town\_Polygon' from data source   
## `C:\Users\saira\OneDrive\Documents\YSPH\YCCCH\CT HIA\CT Vicinity Town Polygon\CT\_Vicinity\_Town\_Polygon.shp'   
## using driver `ESRI Shapefile'  
## Simple feature collection with 1780 features and 16 fields  
## Geometry type: MULTIPOLYGON  
## Dimension: XY  
## Bounding box: xmin: -74.12239 ymin: 40.5114 xmax: -71.12559 ymax: 42.61442  
## Geodetic CRS: WGS 84

# using spreadsheet from 2010 census tract to town conversion tool from CT Data Collaborative: https://tract2town.ctdata.org/  
tract\_town <- read.csv("Census\_Tracts\_with\_Town\_Names\_\_2010\_.csv") %>%   
## join families by tract to towns by tract  
 mutate(GEOID = str\_pad(GEOID, 11, side = "left", pad = "0")) %>%   
 right\_join(ct\_tracts, by = "GEOID") %>%  
## clean rows - delete undefined township, for now keeping rows with no exposure data (East Lyme, Wolcott)  
 filter(`Town` != "County subdivisions not defined")   
  
## clean rows - assign tracts with multiple towns and 0 families to one town  
tract\_town$Town[tract\_town$Town == "East Granby, Windsor Locks, Suffield"] <- "East Granby"  
tract\_town$Town[tract\_town$Town == "Groton, Ledyard"] <- "Groton"  
tract\_town$Town[tract\_town$Town == "Norwalk, Darien, Greenwich, Stamford, Westport"] <- "Norwalk"  
  
## assigning all 20 families in tract to Union, this is the only tract containing Union  
tract\_town$Town[tract\_town$Town == "Stafford, Union"] <- "Union"   
  
## splitting "Canaan, Norfolk" estimate among both since both are only found in this tract  
tract\_town <- tract\_town %>% bind\_rows(slice(., which(tract\_town$Town == "Canaan, Norfolk")))   
tract\_town$Town[tract\_town$Town == "Canaan, Norfolk"] <- "Canaan"  
tract\_town$Town[nrow(tract\_town)] <- "Norfolk"  
tract\_town$estimate[tract\_town$Town == "Canaan"] <- floor(tract\_town$estimate[642]/2)  
tract\_town$estimate[tract\_town$Town == "Norfolk"] <- ceiling(tract\_town$estimate[833]/2)  
  
# summarize mean PM2.5 and standard deviation of tract-level estimates by town  
pm\_town <- tract\_town %>%  
 rename(`TOWN\_NAME` = `Town`) %>%   
 group\_by(`TOWN\_NAME`) %>%   
 summarize(`Average PM2.5 (ug/m3)` = mean(`pm25\_daily\_average.ug.m3.`, na.rm = TRUE), `Standard Deviation (ug/m3)` = sd(`pm25\_daily\_average.ug.m3.`, na.rm = TRUE))   
  
# summarize mean PM2.5 and standard deviation of tract-level estimates by town  
o3\_town <- tract\_town %>%  
 rename(`TOWN\_NAME` = `Town`) %>%   
 group\_by(`TOWN\_NAME`) %>%   
 summarize(`Average O3 (ppb)` = mean(`ozone\_daily\_8hour\_maximum.ppb.`, na.rm = TRUE), `Standard Deviation (ppb)` = sd(`ozone\_daily\_8hour\_maximum.ppb.`, na.rm = TRUE))   
  
# join to town polygon  
pm\_town\_sf <- town\_polygon %>%   
 left\_join(pm\_town, by = "TOWN\_NAME")  
  
o3\_town\_sf <- town\_polygon %>%   
 left\_join(o3\_town, by = "TOWN\_NAME")  
  
# summarize families by town and join housing by town  
pop\_town <- tract\_town %>%   
 group\_by(`Town`) %>%   
 summarize(`Families Pre` = sum(`estimate`)) %>%  
 rename(`TOWN\_NAME` = `Town`) %>%   
 full\_join(housing\_town, by = "TOWN\_NAME")

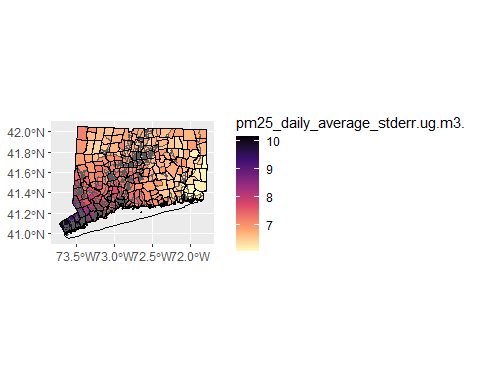
# map PM2.5 standard deviation by town  
ggplot() + geom\_sf(data = pm\_town\_sf, aes(fill = `Standard Deviation (ug/m3)`)) +   
 scale\_fill\_viridis\_c(option = "magma", direction = -1)



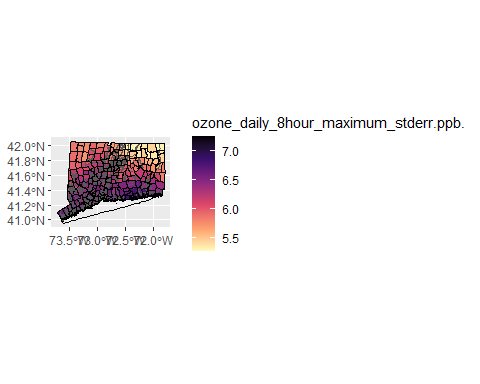
# map ozone standard deviation by town  
ggplot() + geom\_sf(data = o3\_town\_sf, aes(fill = `Standard Deviation (ppb)`)) +   
 scale\_fill\_viridis\_c(option = "magma", direction = -1)



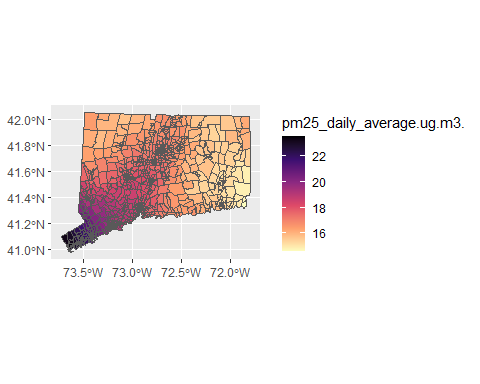
# map PM2.5 standard error by census tract with town lines  
ggplot() + geom\_sf(data = ct\_tracts, aes(fill = pm25\_daily\_average\_stderr.ug.m3.)) +   
 scale\_fill\_viridis\_c(option = "magma", direction = -1) + geom\_sf(data = town\_lines)



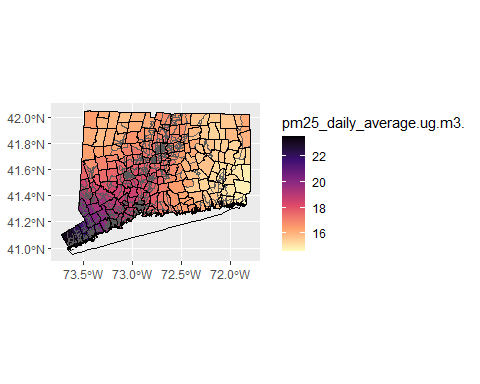
# map ozone standard error by census tract with town lines  
ggplot() + geom\_sf(data = ct\_tracts, aes(fill = ozone\_daily\_8hour\_maximum\_stderr.ppb.)) +   
 scale\_fill\_viridis\_c(option = "magma", direction = -1) + geom\_sf(data = town\_lines)



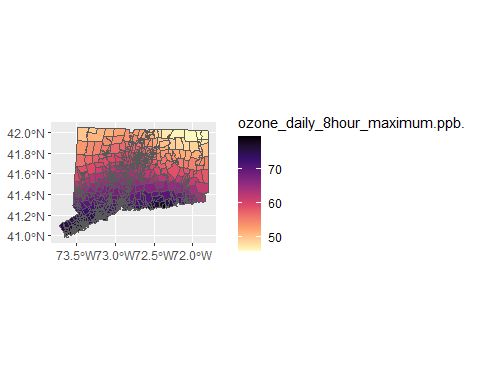
# map PM2.5 by census tract without town lines  
ggplot() + geom\_sf(data = ct\_tracts, aes(fill = pm25\_daily\_average.ug.m3.)) +   
 scale\_fill\_viridis\_c(option = "magma", direction = -1)



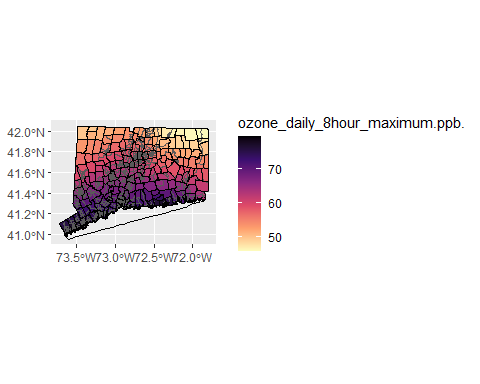
# map PM2.5 by census tract with town lines  
ggplot() + geom\_sf(data = ct\_tracts, aes(fill = pm25\_daily\_average.ug.m3.)) +   
 scale\_fill\_viridis\_c(option = "magma", direction = -1) + geom\_sf(data = town\_lines)



# map O3 by census tract without town lines  
ggplot() + geom\_sf(data = ct\_tracts, aes(fill = ozone\_daily\_8hour\_maximum.ppb.)) +   
 scale\_fill\_viridis\_c(option = "magma", direction = -1)

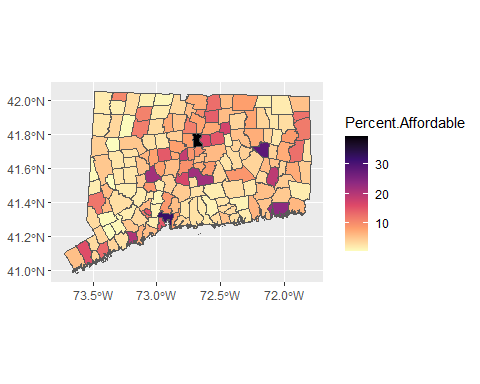


# map O3 by census tract with town lines  
ggplot() + geom\_sf(data = ct\_tracts, aes(fill = ozone\_daily\_8hour\_maximum.ppb.)) +   
 scale\_fill\_viridis\_c(option = "magma", direction = -1) + geom\_sf(data = town\_lines)

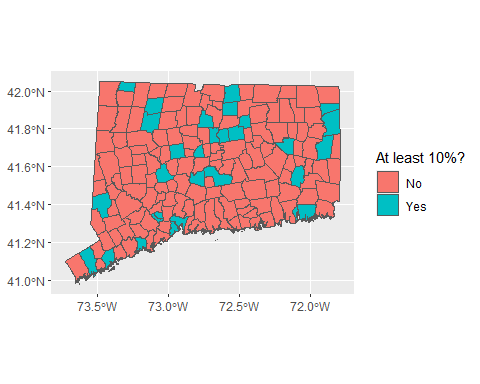


# map temperature by census tract

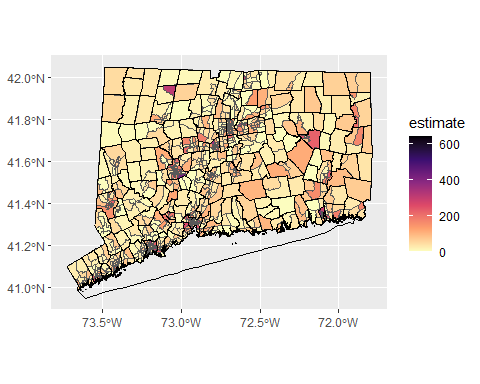
# map percent of affordable housing units by town  
ggplot() + geom\_sf(data = town\_polygon, aes(fill = `Percent.Affordable`)) +   
 scale\_fill\_viridis\_c(option = "magma", direction = -1)



# map 8-30g exemptions by town  
ggplot() + geom\_sf(data = town\_polygon, aes(fill = `At least 10%?`))



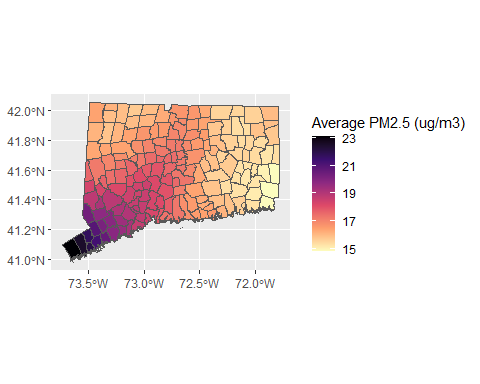
# map families below poverty level by census tract  
ggplot() + geom\_sf(data = ct\_tracts, aes(fill = estimate)) +   
 scale\_fill\_viridis\_c(option = "magma", direction = -1) + geom\_sf(data = town\_lines)



# map low-income population by census tract  
  
# map rent burdened population by census tract

# map baseline incidence of heart disease mortality by census tract

# map PM2.5 by town  
ggplot() + geom\_sf(data = pm\_town\_sf, aes(fill = `Average PM2.5 (ug/m3)`)) +   
 scale\_fill\_viridis\_c(option = "magma", direction = -1)



# simulate random movement into new units  
pop\_town <- pop\_town %>%   
 mutate(`Moved In` = rpois(nrow(.), round(`Units Needed`, 0)))  
  
# list towns pre-move by family (one row per family) and draw random sample (moving out) equal to number of families moved in  
moved\_out\_from <- rep(pop\_town$TOWN\_NAME, times = pop\_town$`Families Pre`) %>%   
 sample(sum(pop\_town$`Moved In`), replace = FALSE)   
  
# summarize families moved out by town and join pop\_town  
pop\_town\_moved <- moved\_out\_from %>%  
 data.frame() %>%   
 rename(`TOWN\_NAME` = 1) %>%   
 group\_by(`TOWN\_NAME`) %>%   
 summarize(`Moved Out` = n()) %>%   
 full\_join(pop\_town, by = "TOWN\_NAME")  
  
# replace NA's for number moved out with 0's (towns not drawn in sample of families)  
pop\_town\_moved$`Moved Out`[is.na(pop\_town\_moved$`Moved Out`)] <- 0  
  
# subtract families moved out and add families moved in to initial estimate for final estimate  
pop\_town\_moved <- pop\_town\_moved %>%   
 mutate(`Families Post` = `Families Pre` + `Moved In` - `Moved Out`)  
  
#join to town polygon  
towns\_moved <- town\_polygon %>%   
 left\_join(pop\_town\_moved, by = "TOWN\_NAME")  
  
# check that total families moved in and moved out match - value will vary with each simulation  
sum(pop\_town\_moved$`Moved Out`)

## [1] 46963

sum(pop\_town\_moved$`Moved In`)

## [1] 46963

# check that total families pre and post match - value should match Census estimate  
sum(pop\_town\_moved$`Families Pre`)

## [1] 61049

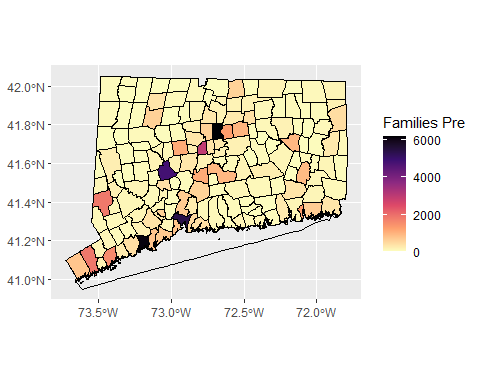
sum(pop\_town\_moved$`Families Post`)

## [1] 61049

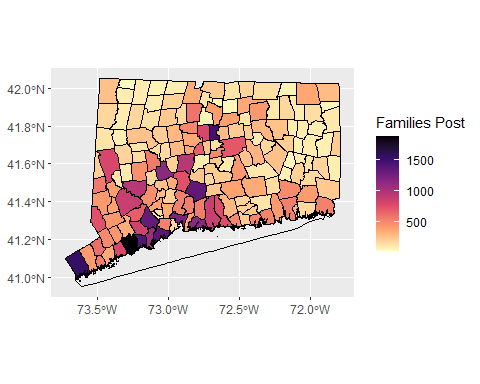
sum(ct\_tracts$estimate)

## [1] 61049

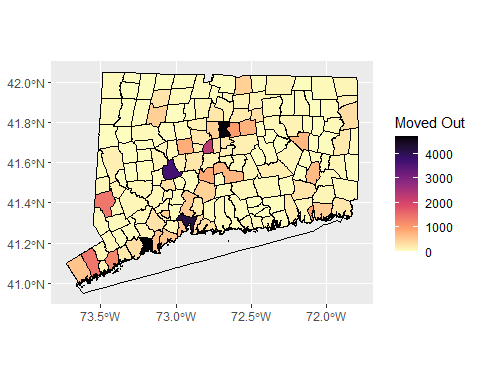
# map families by town pre-move  
ggplot() + geom\_sf(data = towns\_moved, aes(fill = `Families Pre`)) +   
 scale\_fill\_viridis\_c(option = "magma", direction = -1) + geom\_sf(data = town\_lines)



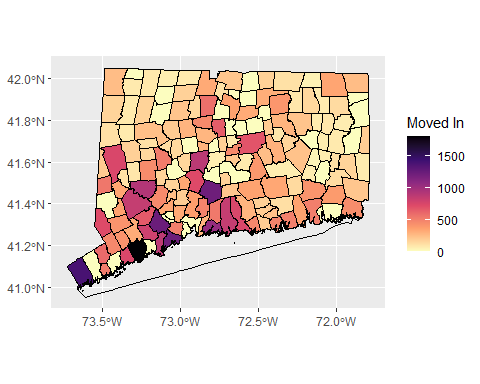
# map families by town post-move  
ggplot() + geom\_sf(data = towns\_moved, aes(fill = `Families Post`)) +   
 scale\_fill\_viridis\_c(option = "magma", direction = -1) + geom\_sf(data = town\_lines)



# map families moving out or staying  
ggplot() + geom\_sf(data = towns\_moved, aes(fill = `Moved Out`)) +   
 scale\_fill\_viridis\_c(option = "magma", direction = -1) + geom\_sf(data = town\_lines)



# map families moving in or staying  
ggplot() + geom\_sf(data = towns\_moved, aes(fill = `Moved In`)) +   
 scale\_fill\_viridis\_c(option = "magma", direction = -1) + geom\_sf(data = town\_lines)



# assign baseline incidence by town  
ischemic\_town <- town\_polygon %>%  
 select(`CNTY\_NAME`, `TOWN\_NAME`) %>%   
 left\_join(ischemic, by = "CNTY\_NAME") %>%  
 group\_by(`TOWN\_NAME`) %>%   
 summarize(`IHD` = mean(`Crude.Rate`))  
  
# assign dose-response relationships / health impact functions  
## ischemic heart disease and PM2.5 - PLACEHOLDER, MADE UP  
dr\_pm\_ischemic <- 0.08 # 8% decrease in IHD mortality per 10 ug/m3 decrease in PM2.5  
  
## ischemic heart disease and O3  
## others  
  
  
# list towns post-move by family moved (one row per family moved)  
moved\_in\_to <- rep(pop\_town$TOWN\_NAME, times = pop\_town$`Moved In`)  
  
# combine towns pre-move by family and towns post-move by family to assign pre- and post- location to each family  
town\_moved\_pre\_post <- data.frame(moved\_out\_from, moved\_in\_to) %>%  
## join pre-move exposure  
 rename(`TOWN\_NAME` = `moved\_out\_from`) %>%   
 left\_join(pm\_town, by = "TOWN\_NAME") %>%  
 ## join pre-move incidence rates  
 left\_join(ischemic\_town, by = "TOWN\_NAME") %>%   
## join post-move exposure  
 select(`moved\_out\_from` = `TOWN\_NAME`, `TOWN\_NAME` = `moved\_in\_to`, `PM2.5 Pre` = `Average PM2.5 (ug/m3)`, `IHD Pre` = `IHD`) %>%  
 left\_join(pm\_town, by = "TOWN\_NAME") %>%  
 select(`moved\_out\_from`, `moved\_in\_to` = `TOWN\_NAME`, `PM2.5 Pre`, `PM2.5 Post` = `Average PM2.5 (ug/m3)`, `IHD Pre` = `IHD Pre`) %>%   
## calculate change in exposure  
 mutate(`Change in PM2.5 (ug/m3)` = `PM2.5 Post` - `PM2.5 Pre`) %>%   
 mutate(`% Change in IHD` = `Change in PM2.5 (ug/m3)`/10\*dr\_pm\_ischemic) %>%   
 mutate(`IHD Post` = `IHD Pre`\*(1 +`% Change in IHD`)) %>%   
 mutate(`IHD Post - Pre` = `IHD Post` - `IHD Pre`)  
  
# add total cases prevented - rate per 100,000 people, each family assumed to have 4 individuals   
sum(town\_moved\_pre\_post$`IHD Post - Pre`)/100000\*4

## [1] -0.009566417

**To-do/to-figure-out** \* Should we randomize family size? Or use average family size specific to each town? \* Population of interest - use 80% AMI (HUD definition but may require calculations), families in poverty (currently - ACS variable B17013\_002), or rent burden (can use ACS variables B25095\_006 et al. and add up rent burdened estimates across income groups)? \* Estimating change in incidence - do we need to know start and end town for each individual/family unit (I’m assuming yes)? Or is there a way to analyze just looking at pre/post distributions? \* Better way to aggregate tracts into towns (also need to check if 2019 tracts = 2010 tracts)? + Best way to deal with tracts containing/contained in multiple townships? Specifically three tracts with 0 families, should I split into multiple observations so pollution estimate can be input into town average? \* Mean vs median to average pollution exposure across census tracts when aggregating to towns? \* Identify/quantify cause-specific mortality endpoints and dose-response relationships (refer to BenMAP) \* Verify - we only need to simulate movement and calculate exposure/incidence changes for this specific population right? Since others not in this population are presumed to stay in place? Is it worth adding random movement of all CT residents? \* Percent affordable based on total assisted units / total units - total assisted currently includes government assisted, tenant rental assistance, SF CHFA USDA mortgages, and deed restricted units - good definition? \* Add inverse distance weighting penalty to movement simulation \* Crude vs. age-adjusted (or otherwise adjusted) incidence rates? \* Where to get incidence rates specific to population?