Is air pollution correlated to inter-state migration in the US

EDA_Final_Project

Leonardo Rueda & Theo Napitupulu

2022-12-07

OVERVIEW

This project looks at the effects of air pollution on inter-state migration in the United States using the Air Quality Index datasets from EPA (available at https://aqs.epa.gov/aqsweb/airdata/download_files.html) and the Population Migration data from the IRS (available at https://www.irs.gov/statistics/soi-tax-stats-migration-data) for the period 2012-2020.

Evidence from middle-income countries shows that air pollution has negative impacts on several health and economic outcomes, such as mortality rates, health expenditures, mental health, hours worked, labor productivity and income. Additionally, other studies have shown how migration decisions are affected by air pollution. For instance, Chen, S., Oliva, P., & Zhang, P. (2022) found that a 10 percent increase in air pollution, holding everything else constant, reduces population through net outmigration by about 2.8 percent in a given county in China (see Chen, S., Oliva, P., & Zhang, P. (2022). The effect of air pollution on migration: Evidence from China. Journal of Development Economics, 156, 102833. https://doi.org/10.1016/j.jdeveco.2022.102833)

Our hypothesis is that there is a positive relationship between high air pollution and migration outflows, that is, the highest the pollution registered by the Air Quality Index (AQI) in a state in a given year, the higher the number of people leaving that state the same year.

The Air Quality Index dataset provides annual information per county about the maximum values reached by the AQI, the number of days in which this index reached values considered unhealthy, and the number of days with PM2.5 particles recorded. The Air Quality Index dataset provides annual information at the State level about the number of people whose reported home address changed in their individual income tax returns.

Set up

- Get the working directory.
- Upload the packages for the analysis.
- Set the theme.

```r
getwd()
```

```
## [1] "C:/Users/leor9/OneDrive/Leonardo/MIDP Courses Fall 2022/R Class/Project/Rueda_Napitupulu_ENV872
```

```r
library("dplyr")
```

```
##
## Attaching package: 'dplyr'
## The following objects are masked from ’package:stats’:
##
##     filter, lag

## The following objects are masked from ’package:base’:
##
##     intersect, setdiff, setequal, union
```

```r
library("plyr")
```

```
## ------------------------------------------------------------------------------

## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)

## ------------------------------------------------------------------------------

##
## Attaching package: ’plyr’

## The following objects are masked from ’package:dplyr’:
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```r
library("ggplot2")
library("tidyverse")
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --

## v tibble  3.1.8     v purrr   0.3.4
## v tidyr   1.2.1     v stringr 1.4.1
## v readr   2.1.2     v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x plyr::arrange()   masks dplyr::arrange()
## x purrr::compact()  masks plyr::compact()
## x plyr::count()     masks dplyr::count()
## x plyr::failwith()  masks dplyr::failwith()
## x dplyr::filter()   masks stats::filter()
## x plyr::id()        masks dplyr::id()
## x dplyr::lag()      masks stats::lag()
## x plyr::mutate()    masks dplyr::mutate()
## x plyr::rename()    masks dplyr::rename()
## x plyr::summarise() masks dplyr::summarise()
## x plyr::summarize() masks dplyr::summarize()
```

```r
library("rvest")
```

```
##
## Attaching package: ’rvest’
##
## The following object is masked from ’package:readr’:
##
##     guess_encoding
```

```
mytheme <- theme_classic() + theme(axis.text = element_text(color = "black"),
    legend.position = "top")
theme_set(mytheme)
```

Data Wrangling

1. Air Quality Datasets

- The raw AQI datasets are available by year. Therefore, we create a dataset with the information from 2010 to 2020.
- This dataset is at the county level. We aggregate the information at the state level. Likewise, we calculate the state averages for the variables Unhealthy.Days, Max.AQI, and Days.PM2.5.

- We save the new dataset in the data/processed folder.

```
# Air quality datasets 2010-2020

mydir = "./Data/Raw/Annual_aqi_by_county"
files = list.files(path = mydir, pattern = "*.csv", full.names = TRUE)
files
```

```
##  [1] "./Data/Raw/Annual_aqi_by_county/annual_aqi_by_county_2010.csv"
##  [2] "./Data/Raw/Annual_aqi_by_county/annual_aqi_by_county_2011.csv"
##  [3] "./Data/Raw/Annual_aqi_by_county/annual_aqi_by_county_2012.csv"
##  [4] "./Data/Raw/Annual_aqi_by_county/annual_aqi_by_county_2013.csv"
##  [5] "./Data/Raw/Annual_aqi_by_county/annual_aqi_by_county_2014.csv"
##  [6] "./Data/Raw/Annual_aqi_by_county/annual_aqi_by_county_2015.csv"
##  [7] "./Data/Raw/Annual_aqi_by_county/annual_aqi_by_county_2016.csv"
##  [8] "./Data/Raw/Annual_aqi_by_county/annual_aqi_by_county_2017.csv"
##  [9] "./Data/Raw/Annual_aqi_by_county/annual_aqi_by_county_2018.csv"
## [10] "./Data/Raw/Annual_aqi_by_county/annual_aqi_by_county_2019.csv"
## [11] "./Data/Raw/Annual_aqi_by_county/annual_aqi_by_county_2020.csv"
```

```
Annual_AQI_by_county_2010_2020 <- ldply(files, read.csv)

# Dataset at the state level and averages for variables of interest

Annual_AQI_by_state_2010_2020 <- Annual_AQI_by_county_2010_2020 %>%
    group_by(Year, State) %>%
    dplyr::summarise(Year = first(Year), Avg.Unhealthy.Days = mean(Unhealthy.Days),
        Avg.Max.AQI = mean(Max.AQI), Avg.Days.PM2.5 = mean(Days.PM2.5))
```

```
## ‘summarise()‘ has grouped output by ’Year’. You can override using the
## ‘.groups‘ argument.
```

```
# Saving the new AQI dataset with information at the state level from
# 2010 to 2020

write.csv(Annual_AQI_by_state_2010_2020, file = "./Data/Processed/Annual_AQI_by_state_2010_2020_Processe
    row.names = FALSE)
```

2. Inter-state migration datasets

- The Inter-state migration datasets do not include the variable "Year", so we create it. Then, we save each dataset in the folder Data/Raw.
- We create a dataset with information from 2012 to 2020.
- According to the dictionary for this dataset, the variable "y2_statefips" and the code "96" refers to the total outflows of migrants for each state in a given year. Therefore, we filtered the previous dataset by that value.
- We change this variable's name to "FIPS_Code", so later we can merge this dataset with the AQI dataset.
- We save the dataset with migration outflows per state for years 2010-2020 in the folder Data/Processed.

```
# Creating the variable 'Year' for each migration dataset

read.csv("./Data/Raw/Outflows_by_state/stateoutflow1112.csv", stringsAsFactors = TRUE) %>%
    mutate(Year = 2012) %>%
    write.csv(file = "./Data/Raw/Outflows_by_state/stateoutflow1112.csv",
        row.names = FALSE)  #2012

read.csv("./Data/Raw/Outflows_by_state/stateoutflow1213.csv", stringsAsFactors = TRUE) %>%
    mutate(Year = 2013) %>%
    write.csv(file = "./Data/Raw/Outflows_by_state/stateoutflow1213.csv",
        row.names = FALSE)  #2013

read.csv("./Data/Raw/Outflows_by_state/stateoutflow1314.csv", stringsAsFactors = TRUE) %>%
    mutate(Year = 2014) %>%
    write.csv(file = "./Data/Raw/Outflows_by_state/stateoutflow1314.csv",
        row.names = FALSE)  #2014

read.csv("./Data/Raw/Outflows_by_state/stateoutflow1415.csv", stringsAsFactors = TRUE) %>%
    mutate(Year = 2015) %>%
    write.csv(file = "./Data/Raw/Outflows_by_state/stateoutflow1415.csv",
        row.names = FALSE)  #2015

read.csv("./Data/Raw/Outflows_by_state/stateoutflow1516.csv", stringsAsFactors = TRUE) %>%
    mutate(Year = 2016) %>%
    write.csv(file = "./Data/Raw/Outflows_by_state/stateoutflow1516.csv",
        row.names = FALSE)  #2016

read.csv("./Data/Raw/Outflows_by_state/stateoutflow1617.csv", stringsAsFactors = TRUE) %>%
    mutate(Year = 2017) %>%
    write.csv(file = "./Data/Raw/Outflows_by_state/stateoutflow1617.csv",
        row.names = FALSE)  #2017

read.csv("./Data/Raw/Outflows_by_state/stateoutflow1718.csv", stringsAsFactors = TRUE) %>%
    mutate(Year = 2018) %>%
    write.csv(file = "./Data/Raw/Outflows_by_state/stateoutflow1718.csv",
        row.names = FALSE)  #2018

read.csv("./Data/Raw/Outflows_by_state/stateoutflow1819.csv", stringsAsFactors = TRUE) %>%
    mutate(Year = 2019) %>%
    write.csv(file = "./Data/Raw/Outflows_by_state/stateoutflow1819.csv",
        row.names = FALSE)  #2019

read.csv("./Data/Raw/Outflows_by_state/stateoutflow1920.csv", stringsAsFactors = TRUE) %>%
    mutate(Year = 2020) %>%
    write.csv(file = "./Data/Raw/Outflows_by_state/stateoutflow1920.csv",
        row.names = FALSE)  #2020

# Merging the dataset for years 2012 - 2020

mydir = "./Data/Raw/Outflows_by_state"
files = list.files(path = mydir, pattern = "*.csv", full.names = TRUE)
files
```

```
## [1] "./Data/Raw/Outflows_by_state/stateoutflow1112.csv"
## [2] "./Data/Raw/Outflows_by_state/stateoutflow1213.csv"
## [3] "./Data/Raw/Outflows_by_state/stateoutflow1314.csv"
## [4] "./Data/Raw/Outflows_by_state/stateoutflow1415.csv"
## [5] "./Data/Raw/Outflows_by_state/stateoutflow1516.csv"
## [6] "./Data/Raw/Outflows_by_state/stateoutflow1617.csv"
## [7] "./Data/Raw/Outflows_by_state/stateoutflow1718.csv"
## [8] "./Data/Raw/Outflows_by_state/stateoutflow1819.csv"
## [9] "./Data/Raw/Outflows_by_state/stateoutflow1920.csv"
```

```
Mig_outflows_by_state_2012_2020 <- ldply(files, read.csv)

# Filtering by the variable 'y2_statefips == 96' (total outflows by
# state)

Mig_outflows_by_state_2012_2020_filtered <- filter(Mig_outflows_by_state_2012_2020,
    y2_statefips == 96)

# Changing the name of the variable with information of the code
# state

colnames(Mig_outflows_by_state_2012_2020_filtered)[1] = "FIPS_Code"

# Saving the dataset with migration information

write.csv(Mig_outflows_by_state_2012_2020_filtered, file = "./Data/Processed/Mig_outflows_by_state_2012
    row.names = FALSE)
```

3. Scraping the FIPS codes

- The AQI Dataset has the names of each State in the US, but it does not have the code, which is the variable we need to merge this data with the migration one.
- In the object "the_website" we store the website direction where the FIPS codes are available (https://www.bls.gov/respondents/mwr/electronic-data-interchange/appendix-d-usps-state-abbreviations-and-fips-codes.htm).

- We scrape from the website the information for the states and codes, and we create a data frame.
- We merge this dataset with the AQI dataset by the variable "State".
- We save the dataset in the folder Data/Processed.

```
# Scrapping the FIPS Codes

the_website <- read_html("https://www.bls.gov/respondents/mwr/electronic-data-interchange/appendix-d-usp

# Creating the variables for the states and codes

the_states <- the_website %>%
    html_nodes("tr+ tr td:nth-child(4) , tr+ tr td:nth-child(1)") %>%
    html_text()
the_states
```

```
##  [1] "Alabama"              "Nebraska"             "Alaska"
##  [4] "Nevada"               "Arizona"              "New Hampshire"
##  [7] "Arkansas"             "New Jersey"           "California"
## [10] "New Mexico"           "Colorado"             "New York"
## [13] "Connecticut"          "North Carolina"       "Delaware"
## [16] "North Dakota"         "District of Columbia" "Ohio"
## [19] "Florida"              "Oklahoma"             "Georgia"
## [22] "Oregon"               "Hawaii"               "Pennsylvania"
## [25] "Idaho"                "Puerto Rico"          "Illinois"
## [28] "Rhode Island"         "Indiana"              "South Carolina"
## [31] "Iowa"                 "South Dakota"         "Kansas"
## [34] "Tennessee"            "Kentucky"             "Texas"
## [37] "Louisiana"            "Utah"                 "Maine"
## [40] "Vermont"              "Maryland"             "Virginia"
## [43] "Massachusetts"        "Virgin Islands"       "Michigan"
## [46] "Washington"           "Minnesota"            "West Virginia"
## [49] "Mississippi"          "Wisconsin"            "Missouri"
## [52] "Wyoming"              "Montana"              " "
```

```
the_codes <- the_website %>%
    html_nodes("tr+ tr td:nth-child(6) , tr+ tr td:nth-child(3)") %>%
    html_text()
the_codes
```

```
##  [1] "01" "31" "02" "32" "04" "33" "05" "34" "06" "35" "08" "36" "09" "37" "10"
## [16] "38" "11" "39" "12" "40" "13" "41" "15" "42" "16" "72" "17" "44" "18" "45"
## [31] "19" "46" "20" "47" "21" "48" "22" "49" "23" "50" "24" "51" "25" "78" "26"
## [46] "53" "27" "54" "28" "55" "29" "56" "30" " "
```

```
df_states_codes <- data.frame(State = the_states, FIPS_Code = as.numeric(the_codes))
```

```
## Warning in data.frame(State = the_states, FIPS_Code = as.numeric(the_codes)):
## NAs introduced by coercion
```

```
# Paste the state code to the air quality dataset

Annual_AQI_by_state_2010_2020 <- merge(Annual_AQI_by_state_2010_2020, df_states_codes,
    by = "State")

# Save the new dataset

write.csv(Annual_AQI_by_state_2010_2020, file = "./Data/Processed/Annual_AQI_by_state_2010_2020_Process
    row.names = FALSE)
```

4. Merging the AQI and Migration datasets

- We merge and arrange the AQI and the migration datasets by the variables "FIPS_Code" and "Year". The resulting dataset has information from 2012 to 2020.
- According to the dictionary for the migration dataset, the variable "n2" refers to the number of individuals who migrated to other states. To facilitate the interpretation, we changed the name of the variable to "Migrants.outflows".
- We create a subset of the previous dataset with the variables of interest: FIPS_Code, Year, State, Avg.Unhealthy.DAys, Avg.Days.AQI, Avg.Days.PM2.5, and Migrants.outflows.
- We save the dataset in the folder Data/Processed. This is the dataset that we will use in our analysis.

```
# Paste air quality and migration data

AQI_Mig.outflows_by.state_2012_2020 <- merge(Annual_AQI_by_state_2010_2020,
    Mig_outflows_by_state_2012_2020_filtered, by = c("FIPS_Code", "Year"))

# Arranging the dataset in ascending order

AQI_Mig.outflows_by.state_2012_2020 <- AQI_Mig.outflows_by.state_2012_2020 %>%
    arrange(Year, FIPS_Code)

# Changing the name of the variable with information of the code
# state

colnames(AQI_Mig.outflows_by.state_2012_2020)[11] = "Migrants.outflows"

# Create a subset with the variables of interest for the analysis

AQI_Mig.outflows_by.state_2012_2020 = subset(AQI_Mig.outflows_by.state_2012_2020,
    select = -c(y2_statefips, y2_state_name, n1, AGI))

# Save the final dataset for the analysis

write.csv(AQI_Mig.outflows_by.state_2012_2020, file = "./Data/Processed/AQI_Mig.outflows_by.state_2012_
    row.names = FALSE)
```

5. Correlations

```
# Initial correlation tests

cor.test(AQI_Mig.outflows_by.state_2012_2020$Avg.Unhealthy.Days, AQI_Mig.outflows_by.state_2012_2020$Mi
```

```
##
##  Pearson's product-moment correlation
##
```

```
## data:  AQI_Mig.outflows_by.state_2012_2020$Avg.Unhealthy.Days and AQI_Mig.outflows_by.state_2012_2020
## t = 6.3515, df = 448, p-value = 5.243e-10
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.2003012 0.3700254
## sample estimates:
##      cor
## 0.287418
```

```
cor.test(AQI_Mig.outflows_by.state_2012_2020$Avg.Max.AQI, AQI_Mig.outflows_by.state_2012_2020$Migrants.
```

```
##
##  Pearson's product-moment correlation
##
## data:  AQI_Mig.outflows_by.state_2012_2020$Avg.Max.AQI and AQI_Mig.outflows_by.state_2012_2020$Migran
## t = 5.7931, df = 448, p-value = 1.303e-08
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.1758409 0.3479362
## sample estimates:
##       cor
## 0.2639883
```

```
cor.test(AQI_Mig.outflows_by.state_2012_2020$Avg.Days.PM2.5, AQI_Mig.outflows_by.state_2012_2020$Migrant
```

```
##
##  Pearson's product-moment correlation
##
## data:  AQI_Mig.outflows_by.state_2012_2020$Avg.Days.PM2.5 and AQI_Mig.outflows_by.state_2012_2020$Mig
## t = -0.48538, df = 448, p-value = 0.6276
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.11512036  0.06966023
## sample estimates:
##         cor
## -0.02292586
```