Data Appendix to "The Impact of Public Transit Introduction on Individual Air Pollutants in Houston, Texas"

Avery Hammond

Contents


```
knitr::opts_chunk$set(results = 'asis', cache = T)
library(tidyverse)
library(summarytools)
library(sf)
library(stargazer)

systemInfo <- Sys.info()
if (systemInfo[[1]] == "Windows") {
  run_summary <- F
} else {
    run_summary <- F}

if (run_summary) {
  st_options(plain.ascii = F,
             style = "rmarkdown",
             footnote = NA,
             subtitle.emphasis = F,
             dfSummary.silent = T,
             dfSummary.valid.col = F,
             dfSummary.style = "grid")

  export_summary_table <- function(dfSummary_output){
    data_info <- attr(dfSummary_output, "data_info")
    ds_name <- data_info$Data.frame
    print(dfSummary_output,
```

```
          file = file.path("output", str_c(ds_name, "_summary.html")),
          method = "browser",
          report.title = ds_name)
  }
}
```

1 Appendix description

This Data Appendix catalogs the data used in "The Impact of Public Transit Introduction on Individual Air Pollutants in Houston, Texas". It was prepared in a Rmarkdown document containing the documentation and R code used to prepare the data used in the final analysis. It additionally includes summary statistics for the final datasets, followed by a discussion of any issues or patterns of note.

The datasets used directly by the final analysis are saved in `processed-data/` at the end of this file.

2 Raw data

2.1 Historical Pollutant and Weather Data

Citation: Texas Commission on Environmental Quality. (2003-2004). "Historical Pollutant and Weather Data." Retrieved from https://www.tceq.texas.gov/airquality/monops/historical_data.html#red

Date Downloaded: March 10, 2020

Filename(s): raw_data/camswx_200x.csv raw_data/co_200x.csv raw_data/nox_200x.csv raw_data/oz_200x.csv raw_data/pm25x_200x.csv raw_data/so2_200x.csv raw_data/sitefile.xls

Unit of observation: parts per billion, micrograms per unit, meters per second, degrees compass, decrees Celsius

Dates covered: 01/01/2003-12/31/2004

2.1.1 To obtain a copy

Interested users should visit the Historical Pollutant and Weather Data on the Texas Commission on Environmental Quality website at https://www.tceq.texas.gov/airquality/monops/historical_data.html#red. To download data from 2003 and 2004, users should click on the corresponding years for each column, which will download the data setfor each column as a CSV file.

2.1.2 Variable descriptions

- **date:** Date of observation.
- **hour:** Hour of observation.
- **AQCR:** Air Quality Control Region code.
- **AIRS:** AIRS site identification number.

- **LONG:** Longitudinal coordinates of the air quality monitor location.
- **LATT:** Latitudinal coordinates of the air quality monitor location.
- **TNRCCRNM:** Texas Natural Resource Conservation Commission region name.
- **ST_CODE:** Federal Information Processing Standards state code.
- **nitrogen_oxides:** Concentration of nitrogen oxides in parts per billion.
- **carbon_monoxide:** Concentration of carbon monoxide in parts per billion.
- **sulfur_dioxide:** Concentration of sulfur dioxide in parts per billion.
- **PM_25:** Concentration of particulate matter_{2.5} in micrograms per cubic meter.
- **TMP1:** Temperature taken hourly, measured in degrees Celsius.
- **WDR1:** Wind direction taken hourly, measured in degrees compass.
- **WSR1:** Wind speed taken hourly, measured in meters per second.
- Note: These data sets include additional variables that were not relevant to this study and therefore omitted from this list.

2.1.3 Data import code and summary

```
years = c("2003","2004")

download_and_read_data <- function(filename){

  if (!file.exists(file.path("raw-data",str_c(filename, ".csv")))) {

    destfile = file.path("raw-data",str_c(filename, ".zip"))

    download.file(url =
                    str_c("https://www.tceq.texas.gov/assets/public/compliance/monops/air/ozonehist/",
                          filename, ".zip"), destfile = destfile)

    unzip(destfile, exdir = "raw-data", junkpaths = T)
  }
  this_data <- read_csv(file.path("raw-data", str_c(filename, ".csv")))
}
```

```
co_data <- lapply(str_c("co_",years), download_and_read_data) %>%
  bind_rows()
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   date = col_character(),
##   COunit = col_character(),
##   COmetxt = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   date = col_character(),
##   COunit = col_character(),
##   COmetxt = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

```
if (run_summary) {
  export_summary_table(dfSummary(co_data))
}

co_hourly <- co_data %>%
  select(airs, ST_CODE, AQCR, date, contains("CO1hr")) %>%
  select(-CO1hrvh, -CO1hrvd, -CO1hrpk, -CO1hrav) %>%
  group_by(airs, ST_CODE, AQCR, date) %>%
  pivot_longer(cols = contains("CO1"), names_to = "hour",
               values_to = "carbon_monoxide") %>%
  mutate(hour = as.numeric(str_remove(hour,"CO1hr"))) %>%
  mutate(Date = as.Date(date, "%m/%d/%Y")) %>%
  mutate(after = Date >="2004/01/01")
```

```
so2_data <- lapply(str_c("so2_",years), download_and_read_data) %>%
  bind_rows() %>%
  select(airs, ST_CODE, AQCR, date, contains("SO21hr")) %>%
  select(-SO21hrvh, -SO21hrvd, -SO21hrpk, -SO21hrav)
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   date = col_character(),
##   SO2unit = col_character(),
##   SO2metxt = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   date = col_character(),
##   SO2unit = col_character(),
##   SO2metxt = col_character()
## )

## See spec(...) for full column specifications.
```

```
if (run_summary) {
  export_summary_table(dfSummary(so2_data))
}

so2_hourly <- so2_data %>%
  group_by(airs, ST_CODE, AQCR, date) %>%
  pivot_longer(cols = contains("SO21"), names_to = "hour",
               values_to = "sulfur_dioxide") %>%
  mutate(hour = as.numeric(str_remove(hour,"SO21hr"))) %>%
  mutate(Date = as.Date(date, "%m/%d/%Y")) %>%
  mutate(after = Date >="2004/01/01")
```

```
nox_data <- lapply(str_c("nox_",years), download_and_read_data) %>%
  bind_rows() %>%
  select(airs, ST_CODE, AQCR, date, contains("NOX1hr")) %>%
  select(-NOX1hrvh, -NOX1hrvd, -NOX1hrpk, -NOX1hrav)
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   date = col_character(),
##   NOunit = col_character(),
##   NOmetxt = col_character(),
##   NO2unit = col_character(),
##   NO2metxt = col_character(),
##   NOXunit = col_character(),
##   NOXmetxt = col_character()
## )

## See spec(...) for full column specifications.

## Parsed with column specification:
## cols(
##   .default = col_double(),
##   date = col_character(),
##   NOunit = col_character(),
##   NOmetxt = col_character(),
##   NO2unit = col_character(),
##   NO2metxt = col_character(),
##   NOXunit = col_character(),
##   NOXmetxt = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

```
if (run_summary) {
  export_summary_table(dfSummary(nox_data))
}

nox_hourly <- nox_data %>%
  group_by(airs, ST_CODE, AQCR, date) %>%
  pivot_longer(cols = contains("NOX1"), names_to = "hour",
               values_to = "nitrogen_oxides") %>%
  mutate(hour = as.numeric(str_remove(hour,"NOX1hr"))) %>%
  mutate(Date = as.Date(date, "%m/%d/%Y")) %>%
  mutate(after = Date >="2004/01/01")
```

```
pm25_data <- lapply(str_c("pm25x_",years), download_and_read_data) %>%
  bind_rows() %>%
  select(AIRS, ST_CODE, AQCR, date, contains("PM251hr")) %>%
  select(-PM251hrvh, -PM251hrvd, -PM251hrpk, -PM251hrav)
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   date = col_character(),
##   PM25unit = col_character(),
##   PM25metxt = col_logical()
## )

## See spec(...) for full column specifications.

## Parsed with column specification:
## cols(
##   .default = col_double(),
##   date = col_character(),
##   PM25unit = col_character(),
##   PM25metxt = col_logical()
## )

## See spec(...) for full column specifications.
```

```
if (run_summary) {
  export_summary_table(dfSummary(pm25_data))
}

pm25_hourly <- pm25_data %>%
  group_by(AIRS, ST_CODE, AQCR, date) %>%
  pivot_longer(cols = contains("PM251"), names_to = "hour",
               values_to = "PM_25") %>%
  mutate(hour = as.numeric(str_remove(hour,"PM251hr"))) %>%
  mutate(Date = as.Date(date, "%m/%d/%Y")) %>%
  mutate(after = Date >="2004/01/01")
```

```r
rose_breaks <- c(0, 360/32, (1/32 + (1:15 / 16)) * 360, 360)

rose_labs <- c("North", "North-Northeast", "Northeast",
               "East-Northeast","East", "East-Southeast",
               "Southeast", "South-Southeast", "South",
               "South-Southwest", "Southwest",
               "West-Southwest", "West", "West-Northwest",
               "Northwest", "North-Northwest", "North")
```

```r
weather_data <- lapply(str_c("camswx_",years), download_and_read_data) %>%
  bind_rows() %>%
  select(AIRS, ST_CODE, AQCR, date, contains("WSR1hr"),
         contains("TMP1hr"), contains("WDR1hr"),
         contains("WHR1hr")) %>%
  select(-contains("hrvh"), -contains("hrvd"),
         -contains("hrav"), -contains("hrpk"))
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   date = col_character(),
##   WSRunit = col_character(),
##   WSRmetxt = col_character(),
##   WDRunit = col_character(),
##   WDRmetxt = col_character(),
##   WSAunit = col_character(),
##   WSAmetxt = col_character(),
##   WDA1hr0 = col_logical(),
##   WDA1hr1 = col_logical(),
##   WDA1hr2 = col_logical(),
##   WDA1hr3 = col_logical(),
##   WDA1hr4 = col_logical(),
##   WDA1hr5 = col_logical(),
##   WDA1hr6 = col_logical(),
##   WDA1hr7 = col_logical(),
##   WDA1hr8 = col_logical(),
##   WDA1hr9 = col_logical(),
##   WDA1hr10 = col_logical(),
##   WDA1hr11 = col_logical(),
##   WDA1hr12 = col_logical()
##   # ... with 18 more columns
## )
```

```
## See spec(...) for full column specifications.
```

```
## Warning: 10142 parsing failures.
##   row     col           expected actual                      file
## 10748 WDA1hr0 1/0/T/F/TRUE/FALSE  83.9  'raw-data/camswx_2003.csv'
## 10748 WDA1hr1 1/0/T/F/TRUE/FALSE  121.5 'raw-data/camswx_2003.csv'
## 10748 WDA1hr2 1/0/T/F/TRUE/FALSE  284.4 'raw-data/camswx_2003.csv'
## 10748 WDA1hr3 1/0/T/F/TRUE/FALSE  275.6 'raw-data/camswx_2003.csv'
## 10748 WDA1hr4 1/0/T/F/TRUE/FALSE  279.4 'raw-data/camswx_2003.csv'
```

```
## ..... ....... .................. ...... ......................
## See problems(...) for more details.

## Parsed with column specification:
## cols(
##   .default = col_double(),
##   date = col_character(),
##   WSRunit = col_character(),
##   WSRmetxt = col_character(),
##   WDRunit = col_character(),
##   WDRmetxt = col_character(),
##   WSAunit = col_character(),
##   WSAmetxt = col_character(),
##   WDA1hr0 = col_logical(),
##   WDA1hr1 = col_logical(),
##   WDA1hr2 = col_logical(),
##   WDA1hr3 = col_logical(),
##   WDA1hr4 = col_logical(),
##   WDA1hr5 = col_logical(),
##   WDA1hr6 = col_logical(),
##   WDA1hr7 = col_logical(),
##   WDA1hr8 = col_logical(),
##   WDA1hr9 = col_logical(),
##   WDA1hr10 = col_logical(),
##   WDA1hr11 = col_logical(),
##   WDA1hr12 = col_logical()
##   # ... with 18 more columns
## )
## See spec(...) for full column specifications.

## Warning: 10223 parsing failures.
##   row     col           expected actual                      file
## 10523 WDA1hr0 1/0/T/F/TRUE/FALSE  15.8  'raw-data/camswx_2004.csv'
## 10523 WDA1hr1 1/0/T/F/TRUE/FALSE  348.4 'raw-data/camswx_2004.csv'
## 10523 WDA1hr2 1/0/T/F/TRUE/FALSE  176.4 'raw-data/camswx_2004.csv'
## 10523 WDA1hr3 1/0/T/F/TRUE/FALSE  249.9 'raw-data/camswx_2004.csv'
## 10523 WDA1hr4 1/0/T/F/TRUE/FALSE  261.9 'raw-data/camswx_2004.csv'
## ..... ....... .................. ...... ......................
## See problems(...) for more details.
```

```
if (run_summary) {
  export_summary_table(dfSummary(weather_data))
}

weather_hourly <- weather_data %>%
  group_by(AIRS, ST_CODE, AQCR, date) %>%
  pivot_longer(cols = contains("hr"),
               names_to = c("measurement", "hour"),
               names_pattern = "(.*)hr(.*)",
               values_to = "value") %>%
  mutate(hour = as.numeric(hour)) %>%
  pivot_wider(names_from = "measurement", values_from = "value") %>%
  mutate(Date = as.Date(date, "%m/%d/%Y")) %>%
```

```
  mutate(airs = AIRS) %>%
  mutate(WDR = cut(WDR1, breaks = rose_breaks,
                   labels = rose_labs, right = FALSE,
                   include.lowest = TRUE)) %>%
  mutate(after = Date >="2004/01/01")
```

```
monitor_locations <- readxl::read_xls("raw-data/sitefile.xls") %>%
  mutate(longitude = -LONG, latitude = LATT) %>%
  select(AIRS,TNRCCRNM, longitude, latitude) %>%
  filter(TNRCCRNM == "Houston") %>%
  filter(!is.na(longitude)) %>%
  filter(longitude != 0) %>%
  st_as_sf(coords = c("longitude", "latitude"), crs = 4326)
```

2.2 Metro Rail Station Data

Citation: City of Houston. (2019). "Metro Rail Station (current)." Retrieved from https://cohgis-mycity.opendata.arcgis.com/datasets/coh-metro-rail-station-current-1?geometry=-95.616%2C29.649%2C-95.181%2C29.858

Date Downloaded: March 10, 2020

Filename(s): raw_data/COH_METRO_RAIL_STATION_current.csv

Unit of observation: degrees longitude, degrees latitude, year in service

Dates covered: 2004

2.2.1 To obtain a copy

Interested users should visit the Metro Rail Stations section of the City of Houston GIS data website at https://cohgis-mycity.opendata.arcgis.com/datasets/1dc7a23374ac44cdae8553044bfeaf22_14?geometry=-95.618%2C29.649%2C-95.179%2C29.858. Users should select "Download," and then select "Spreadsheet" beneath "Full Dataset," which will download the metro rail station data as a csv file.

2.2.2 Variable descriptions

- **latitude:** Latitudinal coordinates of the corresponding station.
- **longitude:** Longitudinal coordinates of the corresponding station.
- **Stat_Name:** Name of the METRO station.
- **Stat_Loc:** The street address of the METRO station.
- **STATUS:** Operative status of the METRO station, either existing or non-existing.
- **service_year:** The year the METRO station began operating.
- **OBJECTID:** Identification number corresponding to individual METRO stations.

2.2.3 Data import code and summary

```
metro_data <- read_csv("raw-data/COH_METRO_RAIL_STATION_current.csv") %>%
  rename(latitude = Y, longitude = X)
```

```
## Parsed with column specification:
## cols(
##   X = col_double(),
##   Y = col_double(),
##   OBJECTID = col_double(),
##   Corr_Name = col_character(),
##   Stat_Name = col_character(),
##   Stat_Loc = col_character(),
##   STATUS = col_character(),
##   Label = col_double(),
##   INSERVICE_Y = col_double()
## )
```

```
metro_data_geometry <- st_as_sf(metro_data, coords = c("longitude", "latitude"),
                                crs = 4326) %>%
  filter(Label == 1) %>%
  select(OBJECTID)

colnames(metro_data) [1] <- "latitude"

colnames(metro_data) [2] <- "longitude"

colnames(metro_data) [9] <- "service_year"
```

3 Data Processing and Combination

During the read-in process for all pollutant and weather data sets, the $after$ dummy variable was created by setting it equal to "TRUE" if the date of observation was on or after 01/01/2004. Additionally, compass rose breaks were created prior to reading in the weather data in order to mutate the original numeric wind direction variable (WDR1) into a factor variable (WDR) to account for the circular nature of wind direction measurement. During the read-in process for the metro data, a second data set (metro_data_geometry) was created containing only the coordinates of METRO stations and their OBJECTID variable. These code chunks are not repeated in this section. Compass rose break creation can be found in the code chunk directly before weather data read-in, $after$ variable creation can be found in the last line of all read-in code for pollutant and weather data sets, and metro_data_geometry data set creation can be found within the metro data read-in code.

All monitor locations from the monitor_locations data set and all METRO station locations from the metro_data_geometry data set were combined into a matrix in which distances could be calculated between monitors and METRO stations. The matrix was then transformed into a list of distances, which were converted from meters to kilometers. From this list, the data frame distance_measure was created containing minimum distances (min_dist) between air monitors and METRO stations,. The *exposed* dummy variable was created on this data frame by equaling "TRUE" if the min_dist variable was less than or equal to 10.5 kilometers. All final pollutant regression data was created by merging the weather_hourly, distance_measure, and the corresponding hourly pollutant data sets. The log pollutant variables were created by taking the log of the pollutant variable plus 1 in order to avoid taking the log of null or negative pollutant values.

```
distances <- st_distance(monitor_locations, metro_data_geometry)
colnames(distances) <- metro_data_geometry$OBJECTID
distance_data <- as_tibble(distances)
```

```
## Warning: Calling `as_tibble()` on a vector is discouraged, because the behavior is likely to change i
## This warning is displayed once per session.
```

```
distance_data$airs <- monitor_locations$AIRS

distance_list <- distance_data %>%
  pivot_longer(-airs, names_to = "metro_object", values_to = "distance") %>%
  mutate(distance = as.numeric(distance/1000)) %>%
  mutate(airs = as.numeric(airs))

distance_measure <- distance_list %>%
  group_by(airs) %>%
  summarize(min_dist = min(distance)) %>%
  mutate(airs = as.numeric(airs),
         exposed = min_dist <= 10.5)
```

```
co_regression_data <- merge(distance_measure, merge(co_hourly, weather_hourly)) %>%
  mutate(log_co = log(1 + carbon_monoxide))
```

```
nox_regression_data <- merge(distance_measure, merge(nox_hourly, weather_hourly)) %>%
  mutate(log_nox = log(1 + nitrogen_oxides))
```

```
pm25_regression_data <- merge(distance_measure, merge(pm25_hourly, weather_hourly)) %>%
  mutate(log_pm25 = log(1 + PM_25))
```

```
so2_regression_data <- merge(distance_measure, merge(so2_hourly, weather_hourly)) %>%
  mutate(log_so2 = log(1 + sulfur_dioxide))
```

```
nox_summary_data <- nox_regression_data %>%
   select(nitrogen_oxides, exposed, after, WSR1, TMP1, WDR)

stargazer(nox_summary_data, type = "latex",
          label = "NOXSummaryStats",
          title = "$NO_{x}$ Summary Statistics",
          header = F,
          digits = 2)
```

```
co_summary_data <- co_regression_data %>%
   select(carbon_monoxide, exposed, after, WSR1, TMP1, WDR)

stargazer(co_summary_data, type = "latex",
          label = "COSummaryStats",
          title = "$CO$ Summary Statistics",
          header = F,
          digits = 2)
```

Table 1: NO_x Summary Statistics

Statistic	N	Mean	St. Dev.	Min	Pctl(25)	Pctl(75)	Max
nitrogen_oxides	206,879	16.10	23.85	0.00	5.00	18.00	809.00
exposed	222,120	0.16	0.36	0	0	0	1
after	222,120	0.51	0.50	0	0	1	1
WSR1	217,826	2.69	1.73	0.00	1.39	3.67	15.42
TMP1	218,736	21.12	7.43	−3.50	16.11	26.72	40.33

Table 2: CO Summary Statistics

Statistic	N	Mean	St. Dev.	Min	Pctl(25)	Pctl(75)	Max
carbon_monoxide	49,144	473.99	334.45	0.00	300.00	500.00	6,100.00
exposed	52,560	0.33	0.47	0	0	1	1
after	52,560	0.50	0.50	0	0	1	1
WSR1	50,255	2.42	1.44	0.00	1.30	3.35	9.75
TMP1	52,181	20.92	7.42	−2.00	15.83	26.50	38.28

```
so2_summary_data <- so2_regression_data %>%
   select(sulfur_dioxide, exposed, after, WSR1, TMP1, WDR)

stargazer(so2_summary_data, type = "latex",
          label = "SO2SummaryStats",
          title = "$SO_{2}$ Summary Statistics",
          header = F,
          digits = 2)
```

Table 3: SO_2 Summary Statistics

Statistic	N	Mean	St. Dev.	Min	Pctl(25)	Pctl(75)	Max
sulfur_dioxide	84,090	2.99	6.26	0.00	0.00	4.00	135.00
exposed	87,672	0.60	0.49	0	0	1	1
after	87,672	0.50	0.50	0	0	1	1
WSR1	86,788	2.55	1.49	0.00	1.39	3.49	10.77
TMP1	87,015	21.25	7.25	−1.94	16.33	26.78	38.78

```
pm25_summary_data <- pm25_regression_data %>%
   select(PM_25, exposed, after, WSR1, TMP1, WDR)

stargazer(pm25_summary_data, type = "latex",
          label = "PM25SummaryStats",
          title = "$PM_{2.5}$ Summary Statistics",
          header = F,
          digits = 2)
```

Table 4: $PM_{2.5}$ Summary Statistics

Statistic	N	Mean	St. Dev.	Min	Pctl(25)	Pctl(75)	Max
PM_25	135,235	10.18	7.37	0.00	5.20	13.50	226.60
exposed	139,440	0.12	0.33	0	0	0	1
after	139,440	0.50	0.50	0	0	1	1
WSR1	136,775	2.63	1.68	0.00	1.39	3.58	15.42
TMP1	138,564	21.07	7.45	−3.50	16.00	26.72	39.78

4 Analysis Variables

- **exposed:** Air quality monitors located within 10.5 kilometers of a new subway station
- **after:** Dates on or following the introduction of the METRO Rail between 01/01/2004 and 12/31/2004
- **carbon_monoxide:** Hourly concentration of carbon monoxide in parts per billion
- **log_co:** Log of *carbon_monoxide*
- **nitrogen_oxides:** Hourly concentration of nitrogen oxides in parts per billion
- **log_nox:** Log of *nitrogen_oxides*
- **sulfur_dioxide:** Hourly concentration of sulfur dioxide in parts per billion
- **log_so2:** Log of $sulfur\_dioxide$
- **PM_25:** Hourly concentration of $PM_{2.5}$ in micrograms per cubic meter
- **log_pm25:** Log of PM_25
- **TMP1:** Temperature measured in degrees Celsius
- **WDR:** Wind direction measured in compass directions
- **WSR1:** Wind speed measured in meters per second

```
save("co_hourly", "so2_hourly", "weather_hourly", "nox_hourly", "pm25_hourly", "nox_regression_data", "
```

5 Discussion of Data

The hourly means of all pollutants appear to stay within a relatively low range. The maximum values of *nitrogen_oxides* and $sulfur\_dioxide$ indicate that the data is skewed towards higher concentrations by the outlier maximum values. The mean values of *exposed* show that 16% of air quality stations that measured NO_x were within 10.5 km, as were 33% of monitors that measured CO, 60% of monitors that measured SO_2, and 12% of monitors that measured $PM_{2.5}$. Because the data covers a full year, the wide range between minimum and maximum values of $TMP1$ can be explained by seasonal variation. WDR and *hour* were omitted from this table because they are factor variables, but are included in the final regression.