Tidying ECCC Air Quality Data

David Hall

2021-01-11

Source

Air quality data is taken from Environment and Climate Change Canada's (ECCC) National Air Pollution Surveillance Program (NAPS). This specific applet uses *annual continuous hourly measurements*. Each of these files contains the hourly measurements of a single pollutant (i.e. O_3) across all operational NAPS station for the given year. There is at least a year delay between collection and publication of data.[1]

[1] At the time of writing, the most recent annual report is from 2018

Data organization

Each yearly pollutant dataset can be downloaded as a `.csv` from the ECCC website. As listed on the dataset preamble:

- Measurements are reported in parts-per-billion (ppb)
- Data is ending local standard time. (i.e. H01 is the hourly measurements from 00:00 to 01:00).
- Zeros are valid values
- `-999` denotes no data available.

However, the structure of this dataset have some incompatible elements with the `tidyverse` ecosystem, these include:

- Matrix style layout, with each row corresponding to a day, and columns for each hourly report.
- Bilingual headers separated by "//"
- Separation of *date* and *time*

Wide data format

A	B	C
1.1	4.2	5.6
1.0	4.5	5.8

Tidy data format

Condition	Value
A	1.1
A	1.0
B	4.2
B	4.5
C	5.6
C	5.8

Figure 1: Wide vs. tidy data layout in flat files (i.e. csv).

Why i'm not storing server data as tidy

My first idea was to combine two ECCC annual datasets into a single file, tidy, file. While R has some tricks to handling these types of files in memory, when it comes to actually saving, the resulting file was exceptionally large. For example, combining the 10.3 mb `O3_2018.csv` with the 7.5 mb `NO2_2018.csv` resulted in a 126 mb `.csv` file. Correspondingly, I'll store everything 'wide' and after subsetting in app, pivot to 'long'.

Converting ECCC NAPS data for Applet

Packages used include:

```
library(tidyverse)
```

```
## Warning: package 'tibble' was built under R version 4.0.3
```

```
library(anytime) # Quicker than tiverse's lubridate package
```

Importing and joining ECCC datasets

The goal is to combine multiple ECCC hourly datasets in their 'wide' format. In other words, so that each row is a day, and the hourly data is stored as a column.

```
# Adding pollutant as prefix to hour columns.
ECCCcolRename <- function(df, chem){

  df <- df %>% rename_all(funs(gsub("\\..*","", make.names(names(df))))) %>%
    rename_at(vars(starts_with("H")), ~paste0(chem,"_", .))

}


# Only worrying about O3 and NO2 for now
combineO3andNO2 <- function(O3, NO2, nrows = Inf){

  O3dat <-  read.csv(O3, skip = 7, nrows = nrows) %>%
    ECCCcolRename(., "O3") %>%
    select(-c("Pollutant"))

  NO2dat <- read.csv(NO2, skip = 7, nrows = nrows) %>%
    ECCCcolRename(., "NO2") %>%
    select(-c("Latitude", "Longitude", "P", "City", "Pollutant"))

  df <- O3dat %>%
    inner_join(NO2dat, by = c("NAPS", "Date")) %>%
    na_if(-999)

}

datWide <- combineO3andNO2(O3 = "O3_2018.csv", NO2 = "NO2_2018.csv")
```

Classifying population centres

A tentative question is for students to compare their urban (downtown Toronto) data to that of a rural/smaller population center. To this

end, I needed to identify the population of each NAPS ID assosciated City and use that vale to colour the map marker. StatsCan *Population and Dwelling Count Highlight Tables, 2016 Census* has population counts from all census populations. Data can be found here According to Stats Canada an " 'urban area' was defined as having a population of at least 1,000 and a density of 400 or more people per square kilometre. All territory outside an urban area was defined as rural area."

They also offer classification on urban population centres:

- **small population centres**, with a population of between 1,000 and 29,999
- **medium population centres**, with a population of between 30,000 and 99,999
- **large urban population centres**, consisting of a population of 100,000 and over

```
pops <- read.csv("NAPSPops.csv") # Population info for every NAPS City

# Adding population to wide data
mapInfo <- datWide %>%
  inner_join(pops, by = "City") %>%
  distinct(NAPS, .keep_all = TRUE) %>%
  select(c(City, P, NAPS, Latitude, Longitude, Population))  %>%
  mutate(PopSize = if_else(Population >= 100000, "large",
                    if_else(Population < 100000 & Population >= 30000, "medium",
                     if_else(Population < 30000 & Population >= 1000, "small","rural")))) %>%
    relocate(Population, .before = Latitude) %>%
    relocate(PopSize, .before = Latitude)

# Simplifying city names for map labels & dropdown menu
mapInfo <- mapInfo %>%
  unite("City", City:P, sep = ", ") %>%
  unite("NAPS", City:NAPS, sep = ", NAPS: ")

#write.csv(df, file = "www/ECCC2018_wideCombined.csv")
```

Saving Data files

```
# Stripping redundant data from ECCC data
dat4App <- datWide %>%
  unite("City", City:P, sep = ", ") %>%
  unite("NAPS", City:NAPS, sep = ", NAPS: ") %>%
  select(-c(Latitude, Longitude))
```

```
# Saving map info and App data

write.csv(dat4App,
          file = "www/ECCC2018_wideCombined.csv",
          row.names=FALSE)

write.csv(mapInfo,
          file = "www/ECCC2018_mapInfo.csv",
          row.names=FALSE)
```

Comparison between tidy and wide

Currently, combining and saving data in the 'wide' format *significantly* reduces file size. For example, saving the combined 2018 NO2 and O3 hourly data in the wide format takes 11.8 mb, whereas saving the same data in the 'long' tidy format takes 126 mb.

Importing and Tidying Data

I wrote a quick function called `ECCCTidy` which tidies a single ECCC NAPS `.csv` file. In other words, it converts the ECCC 'matrix' layout into a 'long' layout where each row is the measurement of that specific pollutant at a given date-time and location. *Note* that all columns start with a capital letter to maintain consistency with the original NAPS dataset.

```
ECCCTidy <- function(file, rows = Inf){

  # Getting pollutant from file name
  chem <- sub("\\_.*", "", file)


  # Skipping ECCC header when importing file
  df <- read_csv(file, skip = 7, n_max = rows)


  # Actually tidying ECCC file
  df <- df %>% rename_all(funs(gsub("\\..*","", make.names(names(df))))) %>%
    pivot_longer(
      cols = starts_with("H"),
      names_to = "Hour",
      values_to = chem)  %>%
    mutate(Date_time = anytime(paste(Date, str_sub(Hour, -2, -1), ":00"))) %>%
```

```
    select(-c(Date, Hour, Pollutant)) %>%
  relocate(Date_time, .before = chem)

  # Combining City w/ NAPS to simplify Shiny selection

  df <- df %>%
    unite("City_NAPS", City:NAPS, sep = ", NAPS: ")

  df
}
```

Transforming Data for Applet

I wrote another function which will combine tidied O_3 and NO_2 datasets,[2] and calculate the O_x value at each given time. All `-999` values are converted to `NA`, and therefore are not used in any subsequent plotting/calculations. I also added a `rows` input where you can specify the number of rows you want to import from the `.csv`. The default is `Inf` (read every row), but you can specify smaller numbers when testing stuff.

[2] Note this uses inner_join, so only O3 and NO2 values from stations found in BOTH datasets will be included.

```
# Test data to combine
NO2 <- ECCCTidy("NO2_2018.csv", rows = 2500)
O3 <- ECCCTidy("O3_2018.csv", rows = 2500)

# Row outline of function, will need to clean up and expand to include other pollutants (i.e. SO2)
ECCCCombine <- function(O3, NO2){
  df <- O3 %>%
  inner_join(NO2) %>%
  na_if(-999) %>%
  mutate(Ox = O3 + NO2)
}
```

Notes for the downstream stuff

- As it stands, combining all of the ECCC data into columns is easier than a massively long .csv (plus if I want to be able to export datasets to excel, this is the way (will need to fix date though…)). However., the 'long' format is easier w/ ggplot2, so it may be a good idea to convert to 'long' the subset of data to be plotted to take advantage of ggplot2 features…