Demo-DH. Vergelijking meetgegevens RIVM (AVK, BL, RS)

Gerrit Versteeg

January 31th, 2017

Synopsis

In dit rapport onderzoeken we de verschillen in de metingen voor drie RIVM-meetstations locaties in Den Haag (Amsterdamse Veerkade, Bleriotlaat en Rebecquestraat).

De analyse leidt tot de volgende conclusies. Er lijkt een toename in emissie te zijn voor NO, NO2 en PM10 op de drie meetstations over de periode van 2015 tot nu. Voor de trend is een standaard lineair model gebruikt. De meetmomenten liggen nogal onregelmatig verspreid in clusters, waarvoor verder gecompenseerd moet worden. Qua gemiddeld dagpatroon is een duidelijke ochtendspits zichtbaar, maar vrijwel geen avondspits. De reden hiervoor is nog onduidelijk.

Data Processing

Loading the data

De data voor dit onderzoek is opgevraagd bij: https://www.luchtmeetnet.nl/download door selectie van telkens één van de vier RIVM-meetstations in Den Haag, gevolgd door de selectie van de breedste periode van aanwezige gevalideerde meetdata (groen gemarkeerde datums) en de invoer van een email-adres voor het resulterende csv-bestand (extract.csv). De csv-bestanden zijn vanuit de eMails opgeslagen in de werkdirectory onder de subdirectory 'data'.

Dit proces is uitgevoerd voor alle (4) meetstations in Den Haag: Export_AVK.csv (312 kB) - Amsterdamse Veerkade Export_BL.csv (353 kB) - Bleriotlaan Export_RS.csv (468 kB) - Rebecquestraat Vaillantlaan, kent geen datums met gevalideerde meetdata en levert derhalve een leeg bestand op.

De drie bestande met gevalideerde inhoud zijn in Nederlands csv-formaat (sep=; en dec=,) daarom gebruiken we read.csv2. De files worden in tibbles geplaatst en daarna samengevoegd tot één tibble (DF_RIVM) gebruikmakend van dplyr voor snellere en eenvoudigere datamanipulatie.

```
library("plyr", warn.conflicts=FALSE)           ## load plyr silently
library("dplyr", warn.conflicts=FALSE)          ## load dplyr silently
library("ggplot2", warn.conflicts=FALSE)        ## load ggplot2 silently

dateDownloaded <- date()                        ## register date of download
DF_AVK <- tbl_df(read.csv2("./data/Export_AVK.csv"))
DF_BL <- tbl_df(read.csv2("./data/Export_BL.csv"))
DF_RS <- tbl_df(read.csv2("./data/Export_RS.csv"))
DF_RIVM <- bind_rows(DF_AVK, DF_BL, DF_RS)      ## concatenate station data
```

```
## Warning in bind_rows_(x, .id): Unequal factor levels: coercing to character

## Warning in bind_rows_(x, .id): Unequal factor levels: coercing to character

## Warning in bind_rows_(x, .id): Unequal factor levels: coercing to character
```

```
DF_RIVM                                          ## take a peek
```

```
## # A tibble: 18,378 x 5
##                      tijdstip                       locatie component
##                         <chr>                         <chr>     <chr>
## 1  2015-08-01 02:00:00+02:00 Den Haag-Amsterdamse Veerkade        NO
## 2  2015-08-01 02:00:00+02:00 Den Haag-Amsterdamse Veerkade       NO2
## 3  2015-08-01 03:00:00+02:00 Den Haag-Amsterdamse Veerkade        NO
## 4  2015-08-01 03:00:00+02:00 Den Haag-Amsterdamse Veerkade       NO2
## 5  2015-08-01 04:00:00+02:00 Den Haag-Amsterdamse Veerkade        NO
## 6  2015-08-01 04:00:00+02:00 Den Haag-Amsterdamse Veerkade       NO2
## 7  2015-08-01 05:00:00+02:00 Den Haag-Amsterdamse Veerkade       NO2
## 8  2015-08-01 05:00:00+02:00 Den Haag-Amsterdamse Veerkade        NO
## 9  2015-08-01 06:00:00+02:00 Den Haag-Amsterdamse Veerkade        NO
## 10 2015-08-01 06:00:00+02:00 Den Haag-Amsterdamse Veerkade       NO2
## # ... with 18,368 more rows, and 2 more variables: waarde <dbl>, LKI <int>
```

De 'coercion'-warnings vertellen ons dat er verschillen in de levels (i.e. factors) zijn tussen de drie verschillende bestanden. Daarom heeft dplyr de formaten van de variabelen met verschillende factors terugegzet naar "character"

Afgeleid uit de door RIVM/LMN meegestuurde data-sheet bevat het bestand gevalideerde uurgemiddelde meetwaarden in csv-format (punt-komma gescheiden):

- Tijdstip : eindtijdstip van metingen van metingen waarmee uurgemiddelde is bepaald
- Locatie : naam van meetpunt
- Component : naam van component
- Waarde : uurgemiddelde meetwaarde (decimaalteken is een komma)
- LKI : luchtkwaliteitsindex; meetwaarde is omgezet tot een getal van 1 (weinig luchtverontreiniging) tot 11 (veel luchtverontreiniging), voor meer informatie zie https://www.luchtmeetnet.nl/uitleg#

Preprocessing the data

De volgende stap is het voorbereiden van de data voor analyse. Daarvoor gaan we data schoonmaken, filteren en verrijken. Allereerst even de de verschillende variabelen (kolommen) nader beschouwen.

tijdstip

Deze variabele geeft ons het meetmoment. Hierbij geldt dat: 2016-01-01 01:00:00+01:00 het gemeten resultaat is van de periode tussen 2016-01-01 00:00:00 en 2016-01-01 01:00:00 in wintertijd in Nederland.

Laten we eerst kijken naar de hoeveelheid unieke meetmomenten per meetstation. Dat doen we door voor variabele 'tijdstip' voor elk bestand een vector te maken met de factors en daarvan de lengte te bepalen, dan weten we het aantal unieke waarden.

```
length(levels(DF_AVK$tijdstip))         ## levels() geeft een vector van de unieke
```

```
## [1] 1730
```

```
length(levels(DF_BL$tijdstip))          ## factors voor variabele 'tijdstip', de
```

```
## [1] 1751
```

```
length(levels(DF_RS$tijdstip))          ## lengte ervan geeft # unieke waarden.
```

```
## [1] 1806
```

Het aantal meetmomenten verschilt per meetstation en daar moeten we rekening mee houden want voor onderlinge vergelijking moeten we alleen de meetmomenten hebben met registraties voor elk van de drie stations.

Vanuit de variabele 'tijdstip' gaan we in verband met plotting in de tijd drie nieuwe variabelen maken, namelijk: datetime (POSIXct-formaat), date (Date-formaat) en time (Time-formaat). Hiermee kunnen individueel meetmomenten herkennen en ook gemiddelden per dag met elkaar vergelijken en de meetwaarden over de verschillende momenten op een dag.

locatie

Het betrokken meetstation gaan we afkorten tot een handzamer formaat, waarbij het deel 'Den Haag -' wordt verwijderd. Tevens zetten de variabele locatie in DF_RIVM om naar een factor-formaat om plotting eenvoudiger te maken.

component

De variabele "component" verwijst naar de gemeten indicatoren (NO, NO2, O3, PM10 en PM2.5). Laten we even kijken naar welke indicatoren door welk meetstation worden gemeten. Ook hier gebruiken we levels() om de factors in de originele bestanden te bekijken

```
levels(DF_AVK$component)
```

```
## [1] "NO"   "NO2"  "PM10"
```

```
levels(DF_BL$component)
```

```
## [1] "NO"   "NO2"  "O3"   "PM10"
```

```
levels(DF_RS$component)
```

```
## [1] "NO"   "NO2"  "O3"   "PM10" "PM25"
```

Alle drie meten NO, NO2 en PM10. Bleriotlaan meet ook nog O3, terwijl de Rebecquestraat ook O3 en PM2.5 meet. Ook hier moeten we rekening mee houden. In dit voorbeeld kiezen we ervoor om alleen de componenten en meetmomenten te nemen die in alle drie meetstations voorkomen. Verder zetten we 'locatie' terug naar het formaat 'factor'.

waarde

Is de werkelijke meetwaarde als een uurgemiddelde over het meetmoment in ug/m3. Formaat houden we 'numeric'.

LKI

Is een categorisering van de meetwaarden in waarden van 1 (goed) tot 11 (slecht). Deze variabele gebruiken we voorlopig niet.

```
gmm <- intersect(levels(DF_AVK$tijdstip),           ## vul gmm met de gemeen-
              intersect(levels(DF_BL$tijdstip),     ## schappelijke meetmomenten
                levels(DF_RS$tijdstip)))            ## in alle drie meetstations

gcp <- intersect(levels(DF_AVK$component),          ## vul gcp met de gemeen-
              intersect(levels(DF_BL$component),    ## schappelijke indicatoren
                levels(DF_RS$component)))           ## in alle drie meetstations

colnames(DF_RIVM)[3] <- "indicator"                 ## varname to 'indicator'
DF_RIVM$locatie <- as.factor(DF_RIVM$locatie)       ## format 'locatie' as factor
levels(DF_RIVM$locatie) <- sub("Den Haag-","",
                levels(DF_RIVM$locatie))            ## en skip 'Den Haag-' deel
```

```
DF_prep <-                                          ## Create DF_prep
        DF_RIVM %>%                                 ## using DF-RIVM to filter
        filter(tijdstip %in% gmm) %>%               ## only common dates
        filter(indicator %in% gcp) %>%              ## only common indicators
        mutate(indicator = as.factor(indicator)) %>%  ## set to format: factor
        mutate(datetime = as.POSIXct(strptime(tijdstip,
                "%Y-%m-%d %H:%M:%S"))) %>%          ## add datetime column
        mutate(date = as.Date(strptime(tijdstip,
                "%Y-%m-%d"))) %>%                   ## add date column
        mutate(time = format(datetime,
                "%H:%M:%S")) %>%                    ## add time column
        print()                                     ## and let's have a look
```

```
## # A tibble: 13,581 x 8
##                      tijdstip              locatie indicator waarde   LKI
##                         <chr>               <fctr>    <fctr>  <dbl> <int>
## 1  2015-08-01 02:00:00+02:00 Amsterdamse Veerkade        NO   2.88     1
## 2  2015-08-01 02:00:00+02:00 Amsterdamse Veerkade       NO2  26.78     3
## 3  2015-08-01 03:00:00+02:00 Amsterdamse Veerkade        NO   4.76     1
## 4  2015-08-01 03:00:00+02:00 Amsterdamse Veerkade       NO2  25.42     3
## 5  2015-08-01 04:00:00+02:00 Amsterdamse Veerkade        NO   3.26     1
## 6  2015-08-01 04:00:00+02:00 Amsterdamse Veerkade       NO2  26.51     3
## 7  2015-08-01 05:00:00+02:00 Amsterdamse Veerkade       NO2  26.46     3
## 8  2015-08-01 05:00:00+02:00 Amsterdamse Veerkade        NO   3.16     1
## 9  2015-08-01 06:00:00+02:00 Amsterdamse Veerkade        NO   3.28     1
## 10 2015-08-01 06:00:00+02:00 Amsterdamse Veerkade       NO2  25.19     3
## # ... with 13,571 more rows, and 3 more variables: datetime <time>,
## #   date <date>, time <chr>
```

Even controleren of we geen missing values hebben door de combinatie van drie meetstations:

```
sum(is.na(DF_prep))                 ## to check for missing values in dataset
```

```
## [1] 0
```

Zoals te verwachten zijn er geen missing values, dus we kunnen starten met wat exploratief onderzoek van de emissie-waarden.

Results

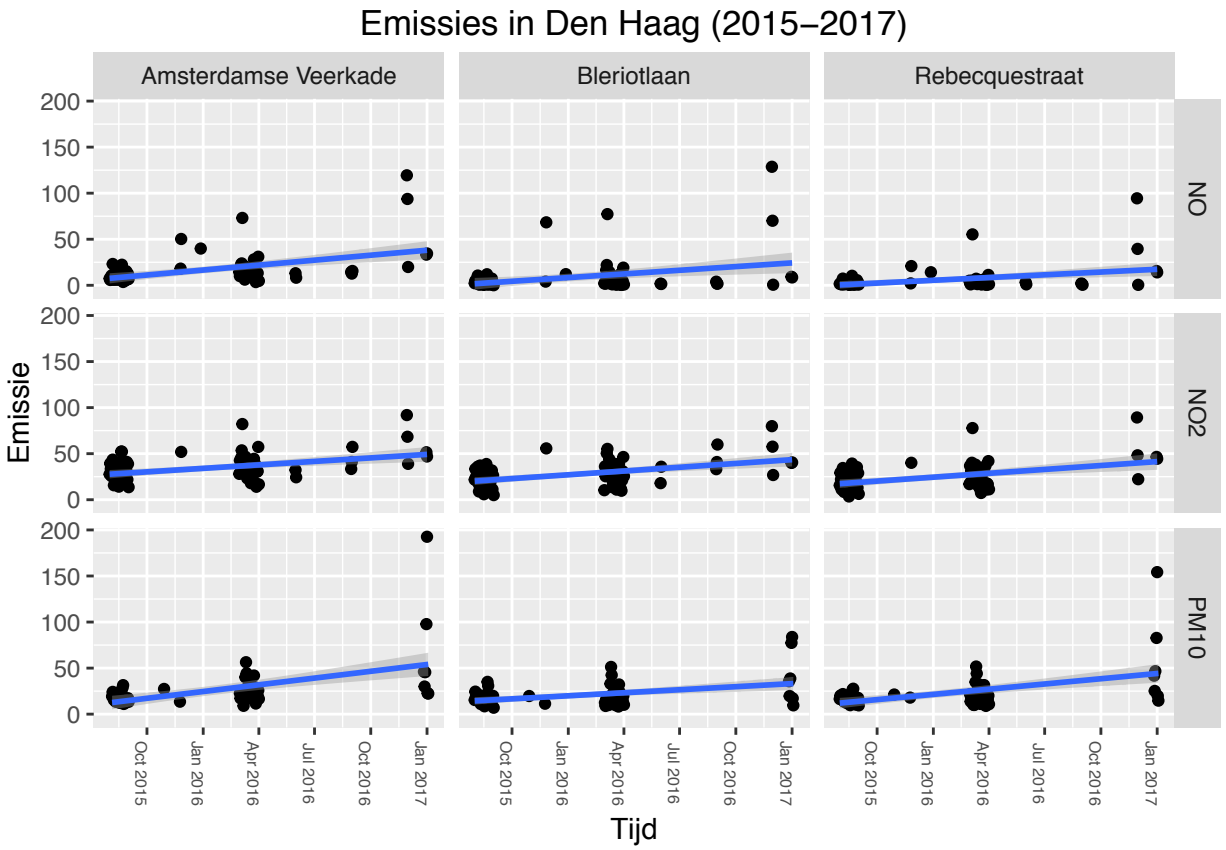
Plotting the indicators accross months

We gaan de drie indicatoren (NO, NO2 en PM10) in drie plots met verschillende kleuren voor de indicatoren per maand uitzetten in een plot, om de verschillen te bekijken. Voor de plot maken we een dedicated dataframe DF_month gebaseerd op DF_prep, die dan gebruikt wordt voor een plot met 9 facetten (indicatoren in de rijen en meetstations in de kolommen).

```
DF_month <-                                              ## create DF_month
        DF_prep %>%                                      ## using DF-prep to
        group_by(date, indicator, locatie) %>%           ## group by date,comp,loc
        summarize(avg = mean(waarde, na.rm = TRUE)) %>%  ## calc. avg waarde
        arrange(date)                                    ## sort ascending by date

g <- ggplot(DF_month,
        aes(x=date, y=avg))                              ## setup graphic object

g + geom_point() +
      geom_smooth(method="lm") +                         ## plot a trendline
      facet_grid(indicator~locatie) +                    ## plot facets per comp/loc
      theme(axis.text.x = element_text(size=6, angle=-90)) + ## adjust X-labels
      xlab("Tijd") +                                     ## lable X-axis
      ylab("Emissie") +                                  ## lable Y-axis
      ggtitle("Emissies in Den Haag (2015-2017)")        ## title plot
```



Plotting the indicator patterns during the day

We gaan de dagelijkse patronen van de drie indicatoren (NO, NO2 en PM10) in drie plots per meetstation met verschillende kleuren voor de indicatoren uitzetten in een plot, om het verloop per dag te bekijken.

ook voor deze plot maken we een dedicated dataframe DF_day gebaseerd op DF_prep, die dan gebruikt wordt voor een plot met 3 facetten (meetstations in de rijen, met kleuren voor de indicatoren).

```
DF_day <-                                                ## create DF_month
        DF_prep %>%                                      ## using DF-prep to
        group_by(time, indicator, locatie) %>%           ## group by date,comp,loc
        summarize(avg = mean(waarde, na.rm = TRUE)) %>%  ## calc. avg waarde
        arrange(time)                                    ## sort ascending by date

h <- ggplot(DF_day,
        aes(x=time, y=avg, color=locatie))               ## setup graphic object

h + geom_point() +
      geom_smooth(method="lm") +                         ## plot a trendline
      facet_grid(indicator~.) +                          ## plot facets per ind
      theme(axis.text.x = element_text(size=6, angle=-90)) + ## adjust X-labels
      xlab("Tijd gedurende dag") +                       ## lable X-axis
      ylab("Emissie") +                                  ## lable Y-axis
      ggtitle("Gemiddelde emissies Den Haag (2015-2017)") ## title plot
```