Documento de cambios

Ángela Alarcón Ballester, Lucía Ponce Salmerón y Carlos Sánchez Polo

2023-11-14

Introducción

En este proyecto analizaremos y trataremos los datos por horas de calidad del aire de las estaciones de la red de vigilancia de la ciudad de Valencia entre los años 2016 y 2020. El conjunto de datos que se utiliza a lo largo de este proyecto se ha obtenido a través del portal de datos abiertos del Ayuntamiento de Valencia (https://valencia.opendatasoft.com/explore/embed/dataset/rvvcca_d_horarios_2016-2020/table/ link).

Comenzaremos realizando una exploración inicial de los datos y responderemos las preguntas que se deriven de ellos.

Importación y acondicionamiento de los datos

Cargamos la librerías que necesitaremos a lo largo del proyecto:

```
library(readr) # Entrada de datos
library(dplyr) # Manipulación de datos
library(tidyr) # Manipulación de datos
library(ggplot2) # Gráficas
library(gridExtra) # Gráficas
library(pheatmap) # Para realizar operaciones
library(purrr) # Mapas de calor
library(knitr) # Tablas
library(GGally)
```

Cargamos y visualizamos los datos:

```
rvvcca_d_horarios_2016_2020 <-
  read_delim("ProyectoAED2023_files/data/rvvcca_d_horarios_2016-2020.csv",
  delim = ";", escape_double = FALSE,
  col_types = cols(Fecha = col_date(format = "%Y-%m-%d"),
  `Fecha creacion` = col_date(format = "%Y-%m-%d"), ), trim_ws = TRUE)
```

Como se observa, las columnas `Fecha` y `Fecha creación` las definimos como fechas en formato "YYYY-MM-DD".

```
head(rvvcca_d_horarios_2016_2020)
```

Convertimos la columna `Estacion` y `Dia de la semana` en factor:

```
rvvcca_d_horarios_2016_2020$Estacion <- as.factor(rvvcca_d_horarios_2016_2020$Estacion)
rvvcca_d_horarios_2016_2020$`Dia de la semana` <- factor(rvvcca_d_horarios_2016_2020$`Dia de la semana`
```

Nos damos cuenta de que la columna `Hora` está en un formato complejo de manejar:

```
unique(rvvcca_d_horarios_2016_2020$Hora)
```

Cambiamos el formato de la columna `Hora` a uno más manejable:

```
rvvcca_d_horarios_2016_2020$Hora <- format(rvvcca_d_horarios_2016_2020$Hora,
                                           format="%H:%M:%S")
unique(rvvcca_d_horarios_2016_2020$Hora)
```

Ordenamos por fecha:

```
indices_orden <- order(rvvcca_d_horarios_2016_2020$Fecha)
rvvcca_d_horarios_2016_2020 <- rvvcca_d_horarios_2016_2020[indices_orden, ]
```

Eliminamos todas las columnas vacías ya que no nos aportan ninguna información. En este caso, solo se eliminará la columna `Fecha baja`.

```
rvvcca_d_horarios_2016_2020 <- rvvcca_d_horarios_2016_2020[, colSums(is.na(rvvcca_d_horarios_2016_2020)
```

Utilizamos la función `glimpse` para obtener una visión general de los datos:

```
glimpse(rvvcca_d_horarios_2016_2020)
```

Observamos que contamos con las siguientes variables:

- Información general (Id, Fecha, Día de la semana, Día del mes, Hora, Fecha de creación). Son de tipo double, factor, fecha y caracter.
- Contaminantes atmosféricos (PM1, PM2.5, PM10, NO, NO2, NOx, O3, SO2, CO, NH3, C7H8, C6H6, C8H10) y Ruido. Estas variables miden la concentración de partículas en suspensión de diferentes tamaños, la concentración de óxidos de nitrógeno, ozono y azufre, entre otros. Todos son de tipo double.
- Información meteorológica (Velocidad del viento, Dirección del viento, Temperatura, Humedad relativa, Presión, Radiación, Precipitación, Velocidad máxima del viento). Todos son de tipo double.
- Estacion (Avda. Francia, Bulevard Sur, Molino del Sol, Pista Silla, Politécnico, Viveros, Centro, Consellería Meteo, Nazaret Meteo, Puerto València), la cual indica la ubicación de la estación de monitoreo. Es de tipo factor.

Valores faltantes

El estudio de los valores faltantes es esencial para garantizar que los análisis de datos sean sólidos y confiables. Al visualizar los datos inicialmente se han identificado valores faltantes, por lo que en este apartado procederemos a analizarlos. Primero, realizamos un resumen de la cantidad de valores faltantes en cada columna:

Podemos ver en la Tabla 1 como hay variables con un gran número de valores faltantes, en especial, en los contaminantes atmosféricos y la información meteorológica. Esto es debido a que cada estación dispone de

Id	0	**Velocidad del viento**	7874
Fecha	0	**Direccion del viento**	6020
Dia de la semana	0	**NH3**	313088
Dia del mes	0	**C7H8**	313156
Hora	0	**C6H6**	313635
Estacion	0	**Ruido**	309508
PM1	286441	**C8H10**	313053
PM2.5	183359	**Temperatura**	9033
PM10	183354	**Humedad relativa**	16357
NO	88740	**Presion**	6276
NO2	71383	**Radiacion**	6814
NOx	88738	**Precipitacion**	7414
O3	87741	**Velocidad maxima del viento**	8045
SO2	91245	**Fecha creacion**	0
CO	236277		

Table 1: Número de valores faltantes de cada variable.

unos sensores distintos de medición de estos fenómenos. Además, el ruido también dispone de muchos valores faltantes, ya que depende de lo tranquila que sea cada estación. Por ello, consideramos oportuno dividir nuestros datos en función de la estación meteorológica asociada, para poder hacer una mejor comparativa, ya que en cada una de ellas se han medido unos valores concretos.

Por tanto, para poder realizar un mejor análisis vamos a agrupar nuestro dataframe por `Estacion`:

```
grupos <- rvvcca_d_horarios_2016_2020 %>% group_by(Estacion)

lista_dataframes <- split(grupos, f = grupos$Estacion)

nombres_dataframes <- unique(rvvcca_d_horarios_2016_2020$Estacion)
```

Limpiamos las columnas vacías de cada dataframe:

```
for (estacion in nombres_dataframes) {
  lista_dataframes[[estacion]] <- lista_dataframes[[estacion]] %>%
    select_if(~!all(is.na(.)))
}
```

Convertimos la lista de dataframes en dataframes independientes:

```
list2env(lista_dataframes, envir = .GlobalEnv)
```

```
## <environment: R_GlobalEnv>
```

Ahora, veamos que parámetros se han medido en cada estación:

```
parametros <- names(rvvcca_d_horarios_2016_2020)

#Creamos dataframe vacío
tabla_zonas <- data.frame(Zona = character(0), stringsAsFactors = FALSE)

for (zona in 1:length(lista_dataframes)) {
```

```
  fila_zona <- data.frame(Zona = names(lista_dataframes)[zona], stringsAsFactors = FALSE)

  for (parametro in parametros) {
    fila_zona[[parametro]] <- parametro %in% colnames(lista_dataframes[[zona]])
  }

  tabla_zonas <- rbind(tabla_zonas, fila_zona)
}

rownames(tabla_zonas) <-tabla_zonas$Zona
colnames(tabla_zonas)[17] <- "V viento"
colnames(tabla_zonas)[29] <- "V_máx viento"
tabla_zonas <- subset(tabla_zonas, select = -c(Zona,Id,Fecha,`Dia de la semana`,`Dia del mes`,Hora,Esta
tabla_numeric <- as.data.frame(sapply(tabla_zonas, as.numeric))
```

```
# Crea el heatmap con colores azul y amarillo
pheatmap(tabla_numeric, color=hcl.colors(2,palette = "BluYl"),
         labels_row =rownames(tabla_zonas),cluster_rows = F,
         cluster_cols = F, legend_breaks = c(0,0.25,0.75, 1),
         legend_labels = c("","No se mide", "Sí se mide",""),
         border_color = "black")
```

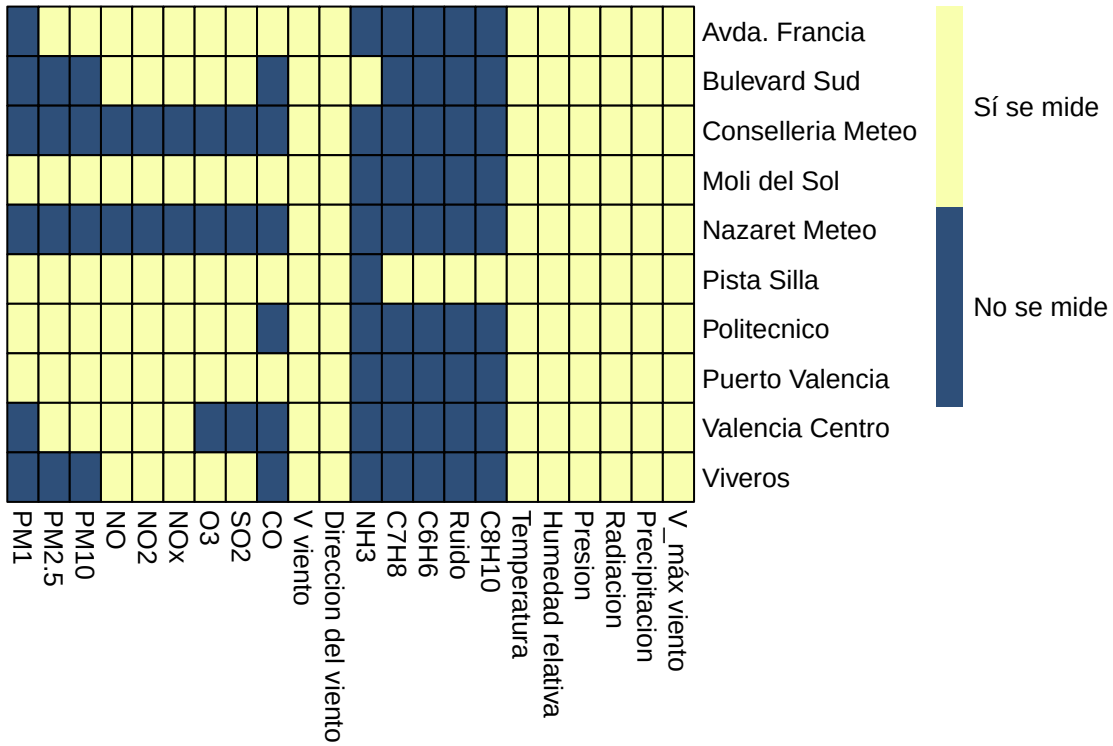


Figure 1: Variables medidas en cada estación.

Como la estación "Pista Silla" es la que menos columnas vacías tiene, es la que vamos a considerar para nuestro estudio. Veamos cuál es el porcentaje de valores faltantes en cada una de sus columnas, para ver qué variables pueden sernos de utilidad:

PM1	0.636	**C8H10**	0.097
PM2.5	0.026	**Temperatura**	0.023
PM10	0.026	**Humedad relativa**	0.045
NO	0.026	**Presion**	0.016
NO2	0.026	**Radiacion**	0.015
NOx	0.026	**Precipitacion**	0.019
O3	0.065	**Velocidad maxima del viento**	0.021
SO2	0.070	**CO**	0.092

Table 2: Porcentaje de valores faltantes de cada variable para la estación "Pista Silla".

En la Tabla 2 observamos que el porcentaje de datos faltantes no es significativo en ninguna de las variables, a excepción de `PM1`. Por ello, podemos considerar que esta variable no nos aportará información suficiente y podemos eliminarla de nuestro análisis.

Una vez que hemos visualizado y analizado los datos, podemos plantearnos ciertas preguntas. En nuestro caso, hemos considerado que, teniendo en cuenta las variables de las que disponemos, sería interesante realizar un estudio que analice la relación entre la calidad del aire y las horas de mayor cantidad de desplazamientos en automóvil, es decir, las horas de más tráfico en carreteras. ¿Se observará un aumento en la concentración de contaminantes atmosféricos en las horas puntas? ¿Dependerá esto de la estación del año en la que nos encontremos? ¿Y los fines de semana nos encontraremos ante un aumento o una disminución?

Análisis de las variables

Análisis univariante

Como hemos mencionado, vamos a hacer un estudio de los contaminantes atmosféricos, por lo que únicamente consideraremos estos valores de ahora en adelante. Hemos visto anteriormente que todos los contaminantes atmosféricos sí se miden en la estación Pista Silla. Por tanto, vamos a calcular los estadísticos básicos de éstos:

```
resultados_totales <- `Pista Silla` %>%
  summarise(across(c("NO", "NO2", "NOx", "O3", "SO2", "CO", "C6H6", "C7H8", "C8H10"),
                   list(mean = mean, sd = sd,
    median = median, IQR = IQR), na.rm = TRUE)) %>%
  pivot_longer(cols = -Estacion, names_to = "variable", values_to = "valor") %>%
  separate(variable, into = c("Contaminante", "stat"), sep = "_") %>%
  pivot_wider(names_from = "stat", values_from = "valor") %>%
  mutate(Estacion = "Pista Silla") %>%
  select(Contaminante, mean, sd, median, IQR) %>%
  arrange(Contaminante)

kable(resultados_totales, caption="Estadísticos básicos de la estación Pista Silla.")
```

Table 3: Estadísticos básicos de la estación Pista Silla.

Contaminante	mean	sd	median	IQR
C6H6	1.5749097	1.6196948	1.1	1.6
C7H8	5.1178385	6.1959639	3.4	4.4
C8H10	2.3844622	3.3573891	1.3	2.5
CO	0.1735941	0.1138526	0.1	0.1

Contaminante	mean	sd	median	IQR
NO	19.7068219	28.4383802	10.0	20.0
NO2	33.0132081	23.6815240	28.0	33.0
NOx	63.0690382	62.9793882	44.0	63.0
O3	45.0142906	26.7762457	46.0	42.0
SO2	4.3111563	2.2229034	4.0	1.0

La tabla de estadísticas descriptivas para los contaminantes del aire destaca diferencias notables entre ellos. Mientras el monóxido de carbono (CO) muestra niveles bajos y poca variabilidad, el dióxido de nitrógeno (NO) exhibe una variabilidad significativa con una media más alta. El dióxido de nitrógeno (NO2) presenta una distribución asimétrica, y el óxido de nitrógeno (NOx) muestra una amplia dispersión de datos. El ozono (O3) tiene una distribución relativamente simétrica, y el dióxido de azufre (SO2) muestra concentración cercana a la media con moderada variabilidad.

—-comentar variables nuevas y cambiar lo que hay——

—-recolocar———

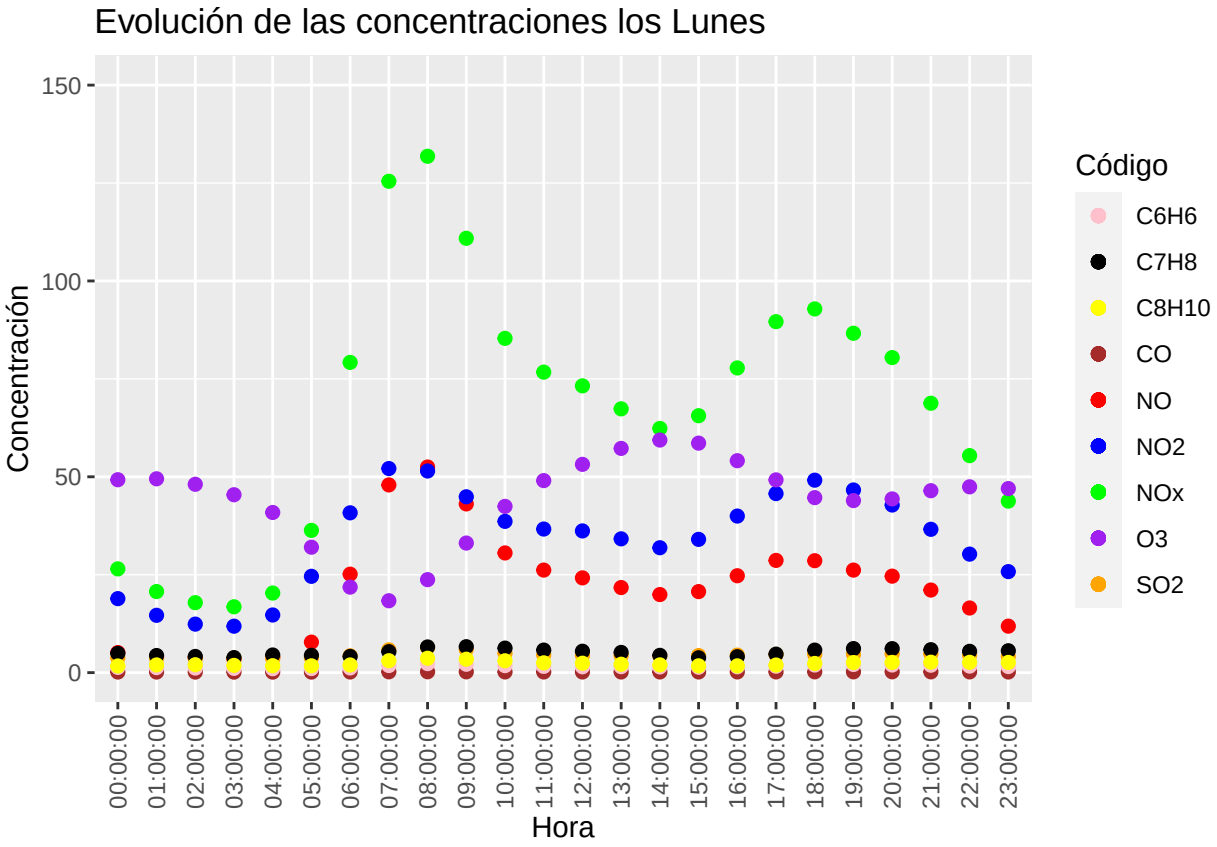
```
require("ggplot2")

dias_semana <- unique(`Pista Silla`$`Dia de la semana`)
par(mfrow = c(5, 2))

if (!dir.exists("./grafs")) {
  dir.create("./grafs")
}
```

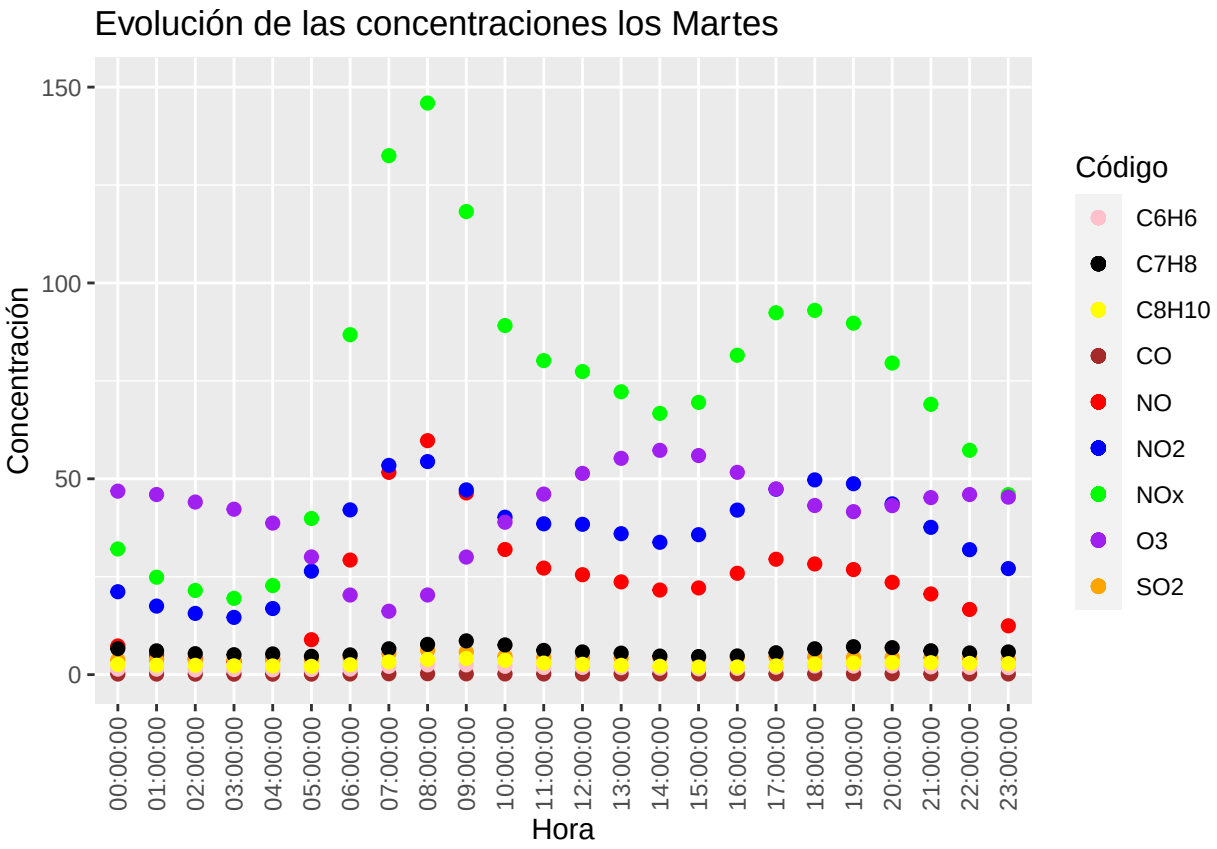
```
library(dplyr)

orden_dias <- c("Lunes", "Martes", "Miercoles", "Jueves", "Viernes", "Sabado", "Domingo")

df_media <- `Pista Silla` %>%
  group_by(`Dia de la semana`, Hora) %>%
  summarise(
    media_NO = mean(NO, na.rm = TRUE),
    media_NO2 = mean(NO2, na.rm = TRUE),
    media_NOx = mean(NOx, na.rm = TRUE),
    media_O3 = mean(O3, na.rm = TRUE),
    media_SO2 = mean(SO2, na.rm = TRUE),
    media_CO = mean(CO, na.rm = TRUE),
    media_C6H6 = mean(C6H6, na.rm = TRUE),
    media_C7H8 = mean(C7H8, na.rm = TRUE),
    media_C8H10 = mean(C8H10, na.rm = TRUE)
  )
```

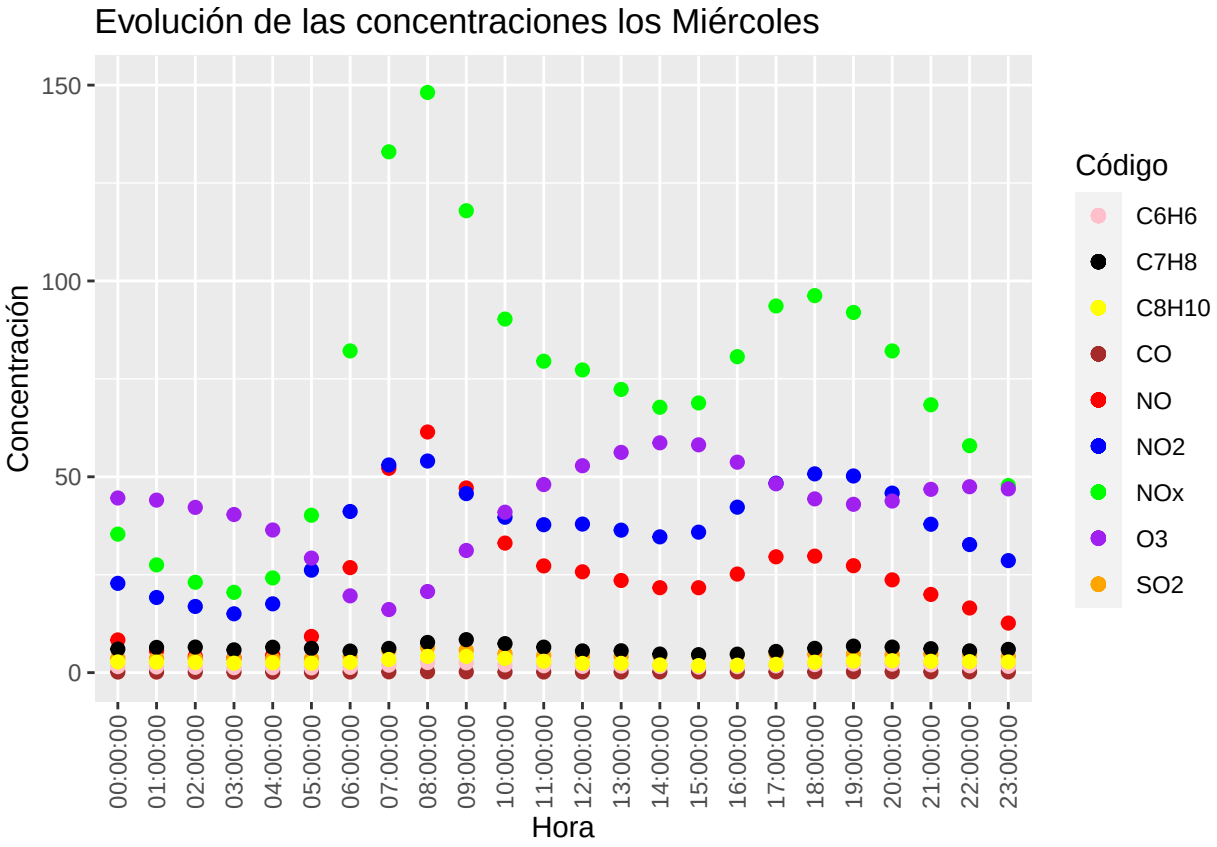
```
## `summarise()` has grouped output by 'Dia de la semana'. You can override using
## the `.groups` argument.
```

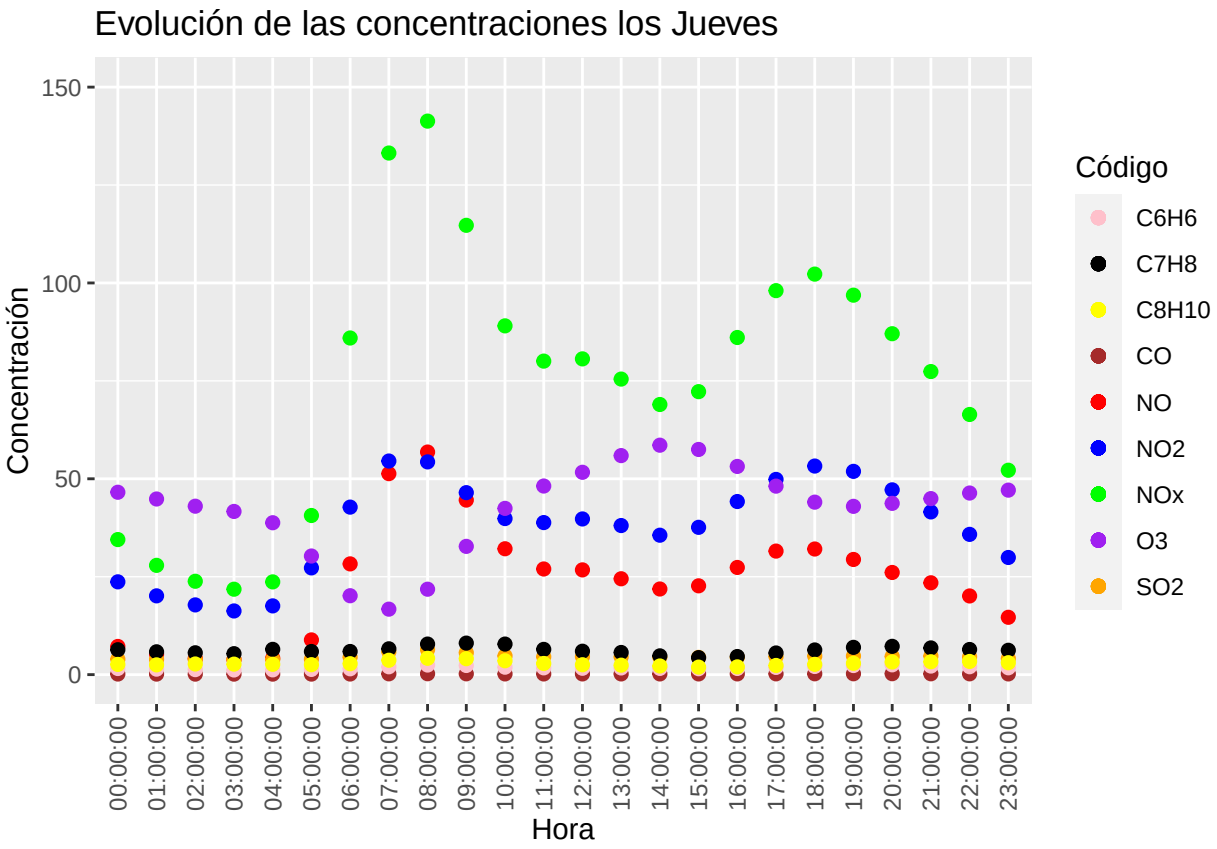
```
require("ggplot2")
dias_semana <- unique(`Pista Silla`$`Dia de la semana`)
par(mfrow = c(5, 2))
```

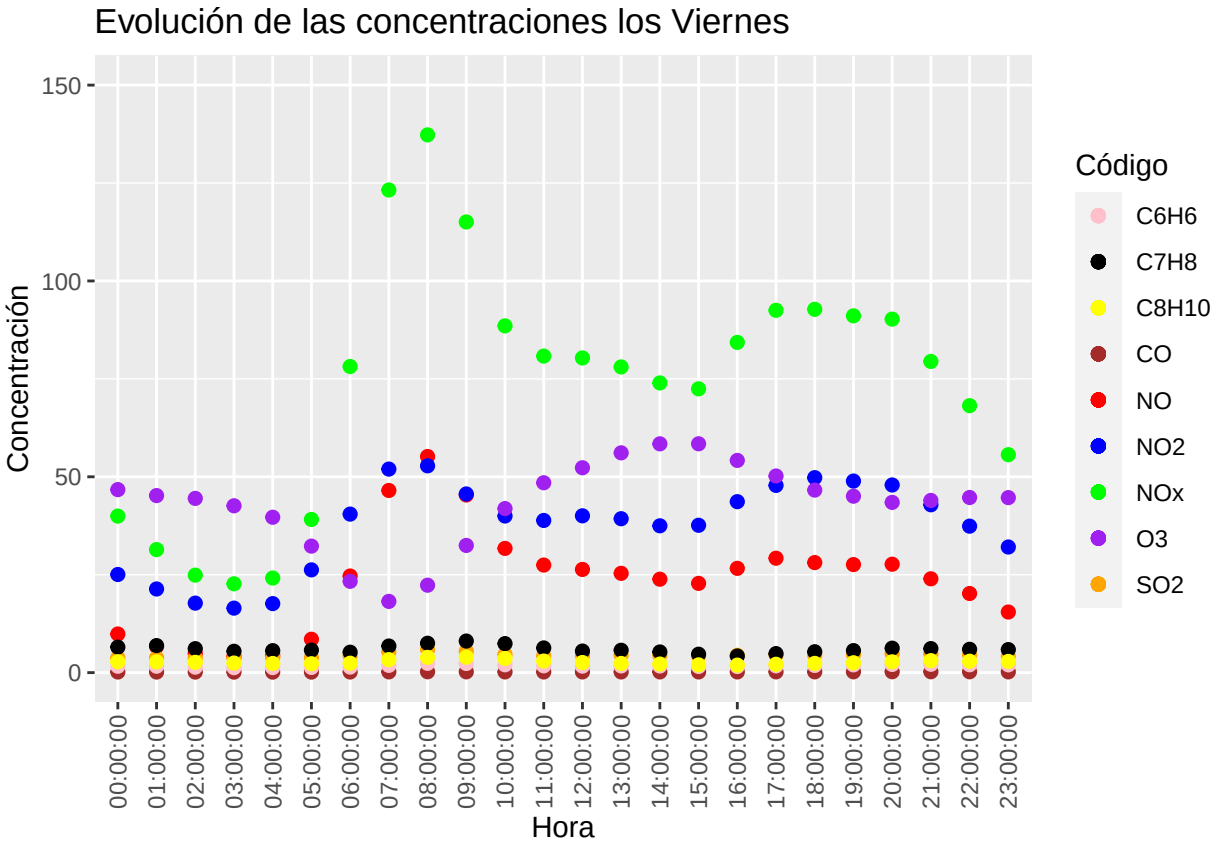
```r
if (!dir.exists("./grafs")) {
  dir.create("./grafs")
}
orden_dias <- c("Lunes", "Martes", "Miercoles", "Jueves", "Viernes", "Sabado", "Domingo")

for (dia in orden_dias) {
  df_media_dia <- df_media[df_media$`Dia de la semana` == dia, ]

  if (nrow(df_media_dia) > 0) {
    # Ajusta el título para manejar correctamente las tildes y la s del plural
    titulo <- ifelse(dia == "Sabado", "Sábados",
                     ifelse(dia == "Domingo", "Domingos",
                            ifelse(dia == "Miercoles", "Miércoles", dia)))

    p <- ggplot(df_media_dia, aes(x = Hora)) +
      geom_point(aes(y = media_NO, color = "NO"), size = 2) +
      geom_point(aes(y = media_NO2, color = "NO2"), size = 2) +
      geom_point(aes(y = media_NOx, color = "NOx"), size = 2) +
      geom_point(aes(y = media_O3, color = "O3"), size = 2) +
      geom_point(aes(y = media_SO2, color = "SO2"), size = 2) +
      geom_point(aes(y = media_CO, color = "CO"), size = 2) +
      geom_point(aes(y = media_C6H6, color = "C6H6"), size = 2) +
      geom_point(aes(y = media_C7H8, color = "C7H8"), size = 2) +
      geom_point(aes(y = media_C8H10, color = "C8H10"), size = 2) +
      ggtitle(paste("Evolución de las concentraciones los", titulo)) +
      xlab("Hora") +
      ylab("Concentración") +
      scale_color_manual(values = c("NO" = "red", "NO2" = "blue", "NOx" = "green", "O3" = "purple", "SO
      labs(color = "Código") +
      coord_cartesian(ylim = c(0, 150)) +
      theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust = 1))

    print(p)

    ggsave(paste("./grafs/grafico_", dia, ".png"), plot = p, device = "png", width = 8, height = 6, uni
  }
}
```

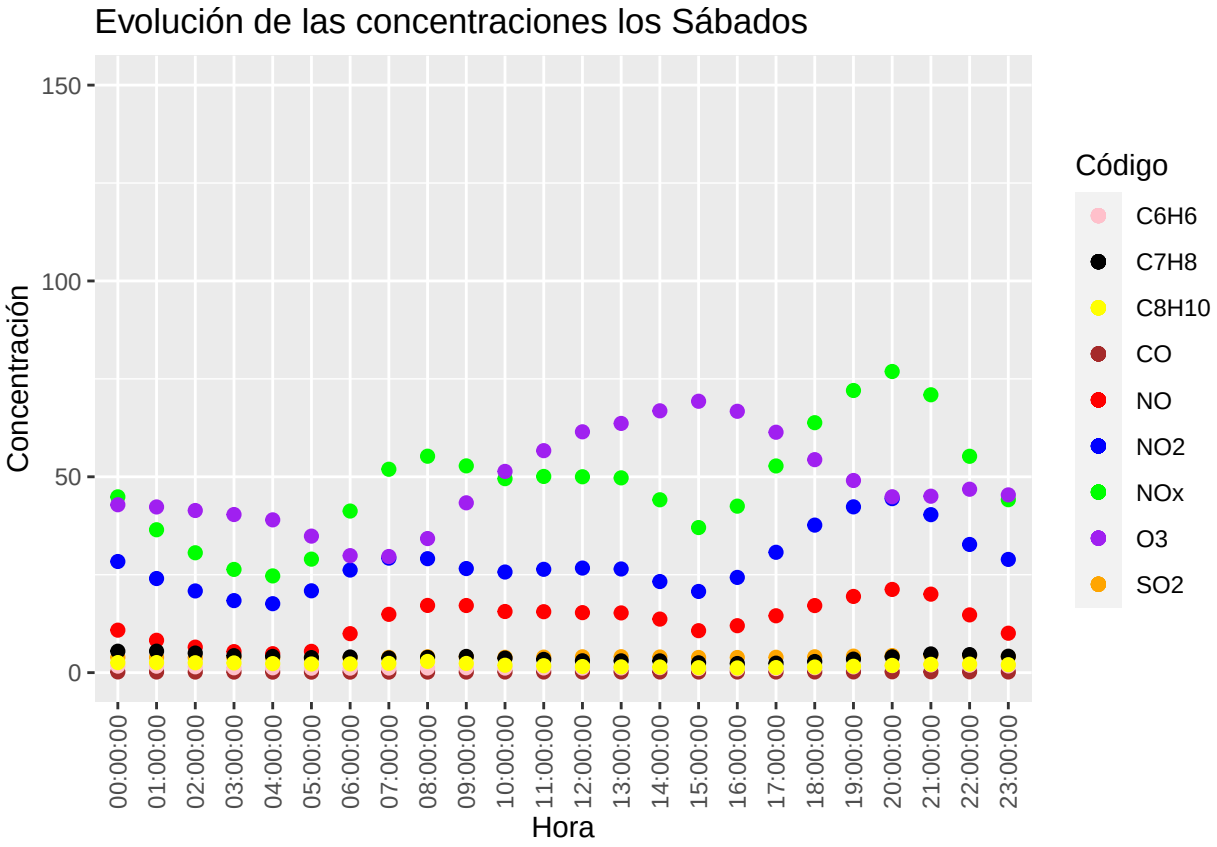

Evolución de las concentraciones los Lunes
Concentración
150
100
50
0
00:00:00
01:00:00
02:00:00
03:00:00
04:00:00
05:00:00
06:00:00
07:00:00
08:00:00
09:00:00
10:00:00
11:00:00
12:00:00
13:00:00
14:00:00
15:00:00
16:00:00
17:00:00
18:00:00
19:00:00
20:00:00
21:00:00
22:00:00
23:00:00
Hora
Código
C6H6
C7H8
C8H10
CO
NO
NO2
NOx
O3
SO2

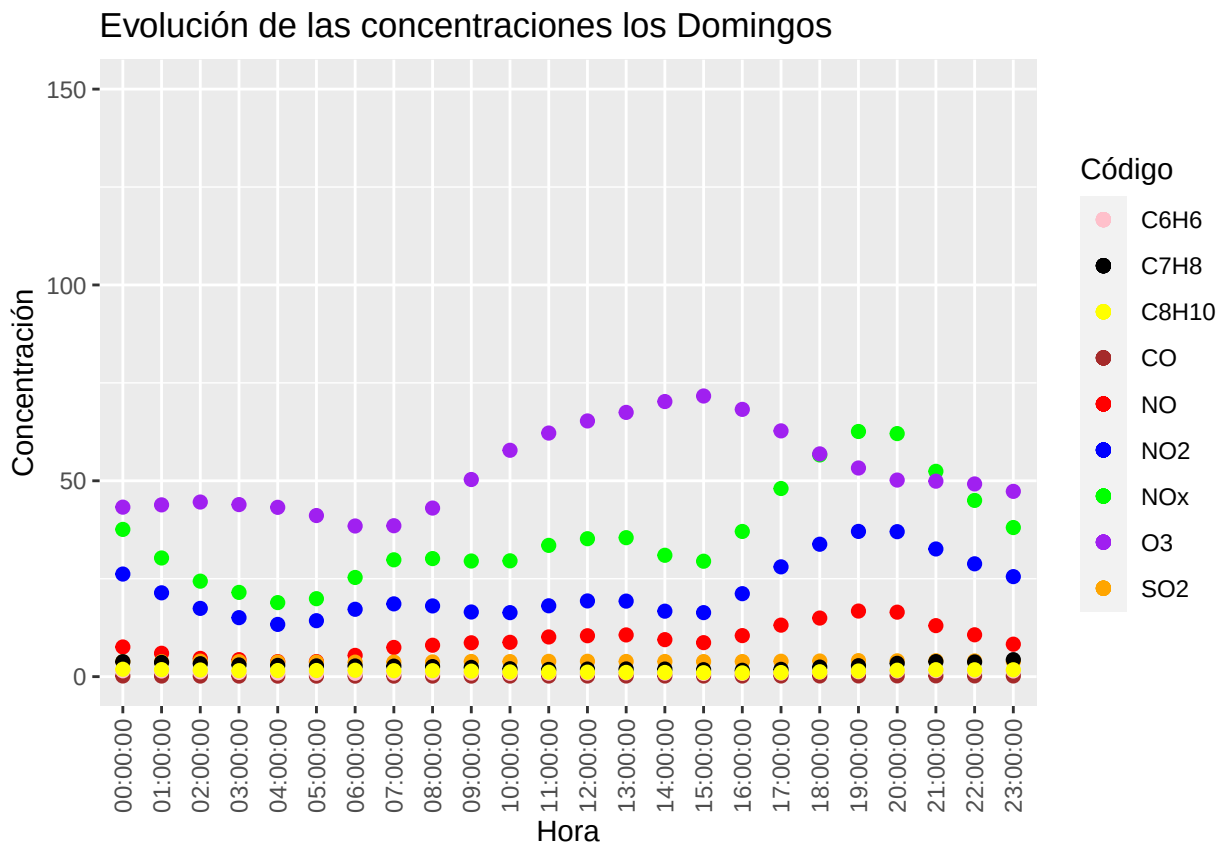

Evolución de las concentraciones los Martes
Concentración
150
100
50
0
00:00:00
01:00:00
02:00:00
03:00:00
04:00:00
05:00:00
06:00:00
07:00:00
08:00:00
09:00:00
10:00:00
11:00:00
12:00:00
13:00:00
14:00:00
15:00:00
16:00:00
17:00:00
18:00:00
19:00:00
20:00:00
21:00:00
22:00:00
23:00:00
Hora
Código
C6H6
C7H8
C8H10
CO
NO
NO2
NOx
O3
SO2


Evolución de las concentraciones los Miércoles
Concentración
150
100
50
0
00:00:00
01:00:00
02:00:00
03:00:00
04:00:00
05:00:00
06:00:00
07:00:00
08:00:00
09:00:00
10:00:00
11:00:00
12:00:00
13:00:00
14:00:00
15:00:00
16:00:00
17:00:00
18:00:00
19:00:00
20:00:00
21:00:00
22:00:00
23:00:00
Hora
Código
C6H6
C7H8
C8H10
CO
NO
NO2
NOx
O3
SO2


Evolución de las concentraciones los Jueves
Concentración
150
100
50
0
00:00:00
01:00:00
02:00:00
03:00:00
04:00:00
05:00:00
06:00:00
07:00:00
08:00:00
09:00:00
10:00:00
11:00:00
12:00:00
13:00:00
14:00:00
15:00:00
16:00:00
17:00:00
18:00:00
19:00:00
20:00:00
21:00:00
22:00:00
23:00:00
Hora
Código
C6H6
C7H8
C8H10
CO
NO
NO2
NOx
O3
SO2


Evolución de las concentraciones los Viernes
Concentración
150
100
50
0
00:00:00
01:00:00
02:00:00
03:00:00
04:00:00
05:00:00
06:00:00
07:00:00
08:00:00
09:00:00
10:00:00
11:00:00
12:00:00
13:00:00
14:00:00
15:00:00
16:00:00
17:00:00
18:00:00
19:00:00
20:00:00
21:00:00
22:00:00
23:00:00
Hora
Código
C6H6
C7H8
C8H10
CO
NO
NO2
NOx
O3
SO2


Evolución de las concentraciones los Sábados
Concentración
150
100
50
0
00:00:00
01:00:00
02:00:00
03:00:00
04:00:00
05:00:00
06:00:00
07:00:00
08:00:00
09:00:00
10:00:00
11:00:00
12:00:00
13:00:00
14:00:00
15:00:00
16:00:00
17:00:00
18:00:00
19:00:00
20:00:00
21:00:00
22:00:00
23:00:00
Hora
Código
C6H6
C7H8
C8H10
CO
NO
NO2
NOx
O3
SO2

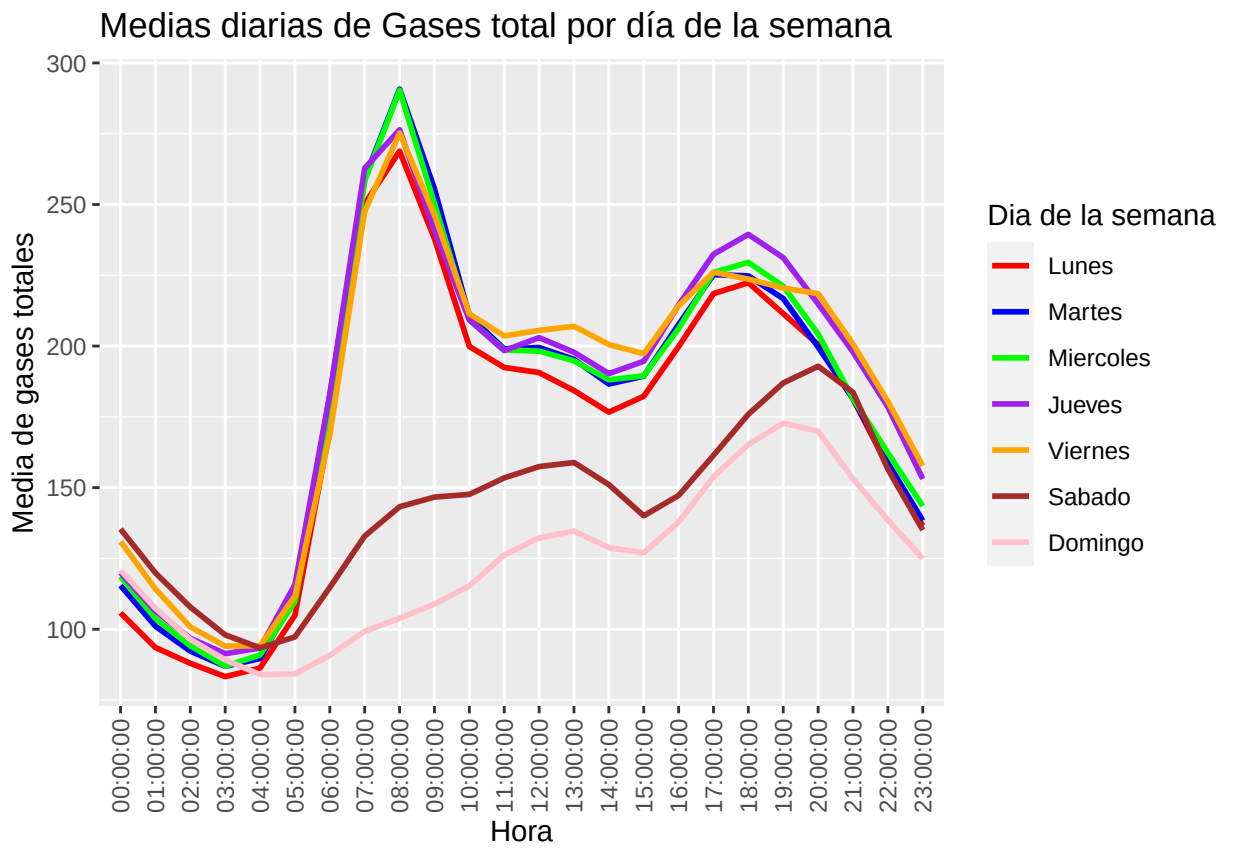



```
require("dplyr")

`Pista Silla`$gases_total <- rowSums(`Pista Silla`[, c('NO', 'NO2', 'NOx', 'O3', 'SO2', 'CO', 'C6H6', '
```

```
media_diaria <- `Pista Silla` %>%
  group_by(`Dia de la semana`, Hora) %>%
  summarise(media_gases_total = mean(gases_total, na.rm = TRUE))
```

```
## `summarise()` has grouped output by 'Dia de la semana'. You can override using
## the `.groups` argument.
```

```
# Crea la gráfica de las medias diarias de 'gases_total' por día de la semana con puntos y dispersión h
p3<-ggplot(media_diaria, aes(x = Hora, y = media_gases_total, color = `Dia de la semana`, group = `Dia d
  geom_line(size = 1) +
  ggtitle("Medias diarias de Gases total por día de la semana") +
  xlab("Hora") +
  ylab("Media de gases totales") +
  scale_color_manual(values = c("Lunes" = "red", "Martes" = "blue", "Miercoles" = "green", "Jueves" = "
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust = 1))

print(p3)
```




```
ggsave("./grafs/medias_gases_total.png", p3, width = 10, height = 6, units = "in", dpi = 300)
```

—-recolocar———

Análisis bivariante

```
#vars_of_interest <- c('Dia de la semana', 'NO', 'NO2', 'NOx', 'O3', 'SO2', 'CO', 'C6H6', 'C7H8', 'C8H1
#df_bivariante <- `Pista Silla`[, vars_of_interest]
#media_por_dia <- aggregate(. ~ `Dia de la semana`, data = df_bivariante, mean, na.rm = TRUE)
#ggpairs(media_por_dia[c('NO', 'NO2', 'NOx', 'O3', 'SO2', 'CO', 'C6H6', 'C7H8', 'C8H10')])
```

Análisis de outliers

A continuación, exploraremos la calidad del aire mediante boxplots que representan las concentraciones de dos sustancias fundamentales: dióxido de nitrógeno (NO2) y ozono (O3). Cada gráfico proporciona una visión detallada de la distribución de estas sustancias en las distintas estaciones de la red de vigilancia atmosférica de la ciudad de València.

```
generate_boxplot <- function(data, title) {
  boxplot(data, main = title,col = "lightpink", border = "darkblue")
}
```

```r
par(mfrow = c(2, 4), mar = c(4, 4, 2, 1),cex.main = 0.8)

for (nombre_df in nombres_dataframes) {
  df <- get(nombre_df)

  if ("NO2" %in% names(df) && length(df$NO2) > 0) {
    generate_boxplot(df$NO2, paste("Outliers NO2", nombre_df, sep = " "))
  } else {
    cat("No se encontró la columna 'NO2' en", nombre_df, "\n")
  }
}
```

```
## No se encontró la columna 'NO2' en Conselleria Meteo
```

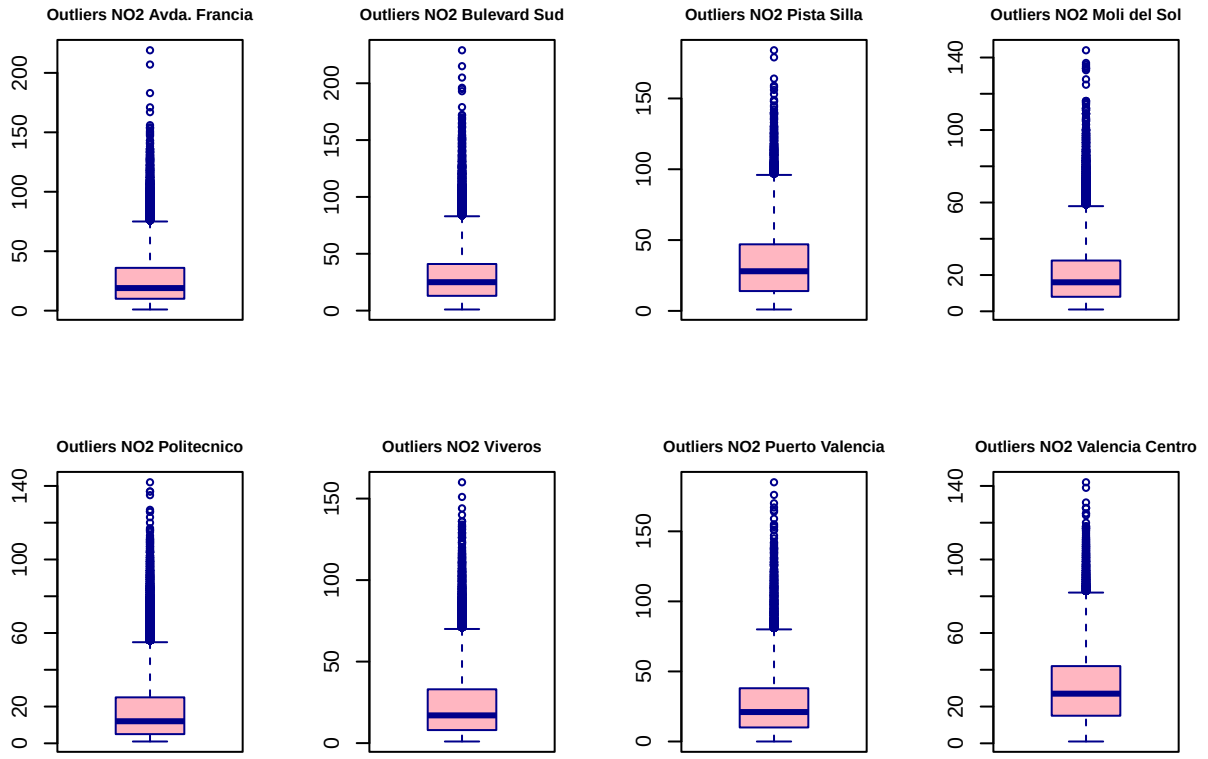


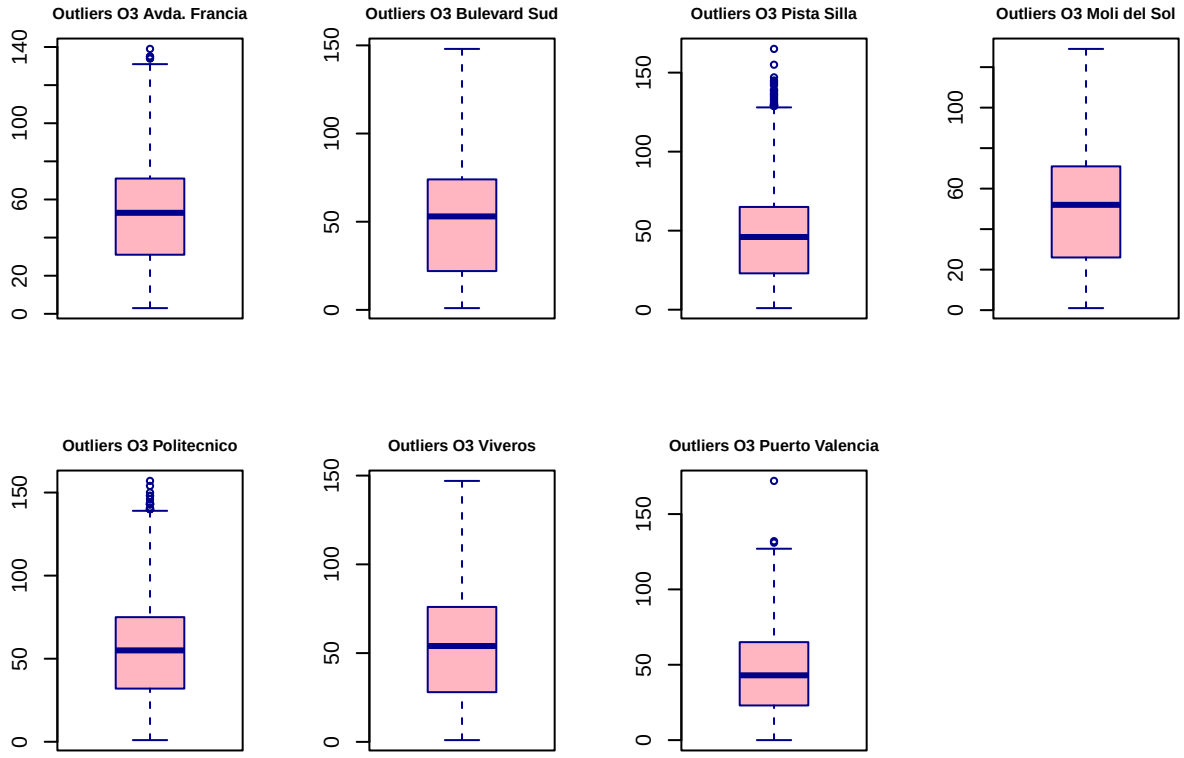
Figure 2: Diagramas de caja de NO2 en cada estación.

```
## No se encontró la columna 'NO2' en Nazaret Meteo
```

Observamos una notable presencia de outliers en la variable `NO2` en todas las estaciones, destacando en la parte superior de los boxplots. Estos boxplots son estrechos, lo que sugiere que la mayor parte de los datos se concentra en un rango reducido de concentraciones de dióxido de nitrógeno. Sin embargo, la presencia de numerosos outliers en la parte superior indica que a menudo se generan concentraciones inusualmente altas de esta sustancia en el aire.

```r
par(mfrow = c(2, 4), mar = c(4, 4, 2, 1),cex.main = 0.8)

for (nombre_df in nombres_dataframes) {
  df <- get(nombre_df)

  if ("O3" %in% names(df) && length(df$O3) > 0) {
    generate_boxplot(df$O3, paste("Outliers O3", nombre_df, sep = " "))
  } else {
    cat("No se encontró la columna 'O3' en", nombre_df, "\n")
  }
}
```

```
## No se encontró la columna 'O3' en Conselleria Meteo

## No se encontró la columna 'O3' en Valencia Centro
## No se encontró la columna 'O3' en Nazaret Meteo
```



En contraste, la presencia de outliers en la variable de ozono es menos pronunciada. Aunque en las estaciones de Viveros y Consellería Meteo se observa una mayor proporción de outliers, en general, los valores presentan una distribución más uniforme. Los boxplots son más anchos, indicando una dispersión más amplia de las concentraciones de ozono.

También hemos considerado oportuno implementar funciones específicas para la identificación de outliers utilizando distintos métodos estadísticos, como la regla de 3 sigma, el identificador Hampel y la regla de percentiles. Este es otro modo de resaltar valores que podrían considerarse atípicos y podemos comparar con los resultados obtenidos mediante el método boxplot.

Función para la regla de 3 sigma:

```
reglasigma <- function(x) {
  media <- mean(x, na.rm = TRUE)
  desviacion <- sd(x, na.rm = TRUE)
  umbral_superior <- media + 3 * desviacion
  umbral_inferior <- media - 3 * desviacion
  outliers <- x[x > umbral_superior | x < umbral_inferior]
  return(outliers)
}
```

Función para el identificador Hampel:

```
reglahampel <- function(x) {
  mediana <- median(x, na.rm = TRUE)
  mad <- mad(x, na.rm = TRUE)
  umbral_superior <- mediana + 3 * mad
  umbral_inferior <- mediana - 3 * mad
  outliers <- x[x > umbral_superior | x < umbral_inferior]
  return(outliers)
}
```

Función para la regla del boxplot:

```
reglaboxplot <- function(x) {
  cuartil_75 <- quantile(x, 0.75, na.rm = TRUE)
  cuartil_25 <- quantile(x, 0.25, na.rm = TRUE)
  iqr <- cuartil_75 - cuartil_25
  umbral_superior <- cuartil_75 + 1.5 * iqr
  umbral_inferior <- cuartil_25 - 1.5 * iqr
  outliers <- x[x > umbral_superior | x < umbral_inferior]
  return(outliers)
}
```

Función para la regla de percentiles:

```
reglapercentil <- function(x) {
  percentil_5 <- quantile(x, 0.05, na.rm = TRUE)
  percentil_95 <- quantile(x, 0.95, na.rm = TRUE)
  outliers <- x[x < percentil_5 | x > percentil_95]
  return(outliers)
}
```

```
detectar_outliers <- function(x) {
  sol <- list(
    r_sigma = data.frame(Variable = colnames(x), Metodo = "Regla 3 sigma",
              Outliers = sapply(x, function(col) length(reglasigma(col)))),
    r_hampel = data.frame(Variable = colnames(x), Metodo = "Identificador Hampel",
               Outliers = sapply(x, function(col) length(reglahampel(col)))),
    r_boxplot = data.frame(Variable = colnames(x), Metodo = "Regla Boxplot",
                Outliers = sapply(x, function(col) length(reglaboxplot(col)))),
    r_percentiles = data.frame(Variable = colnames(x), Metodo = "Regla Percentiles",
```

```
                    Outliers = sapply(x, function(col) length(reglapercentil(col))))
  )

  return(sol)
}


columnas_a_excluir <- c(1:6, ncol(df))
col <- setdiff(names(df), names(df)[columnas_a_excluir])
outliers_resultados <- detectar_outliers(Viveros[col])

plots <- lapply(outliers_resultados, function(result) {
  ggplot(result, aes(x = Variable, y = Outliers, fill = Variable)) +
    geom_bar(stat = "identity") +
    labs(title = unique(result$Metodo),
         x = "Variable",
         y = "Cantidad de Outliers") +
    theme_minimal() + theme(axis.text.x = element_blank(), legend.position = "none")
})

grid.arrange(grobs = plots, ncol = 2)
```

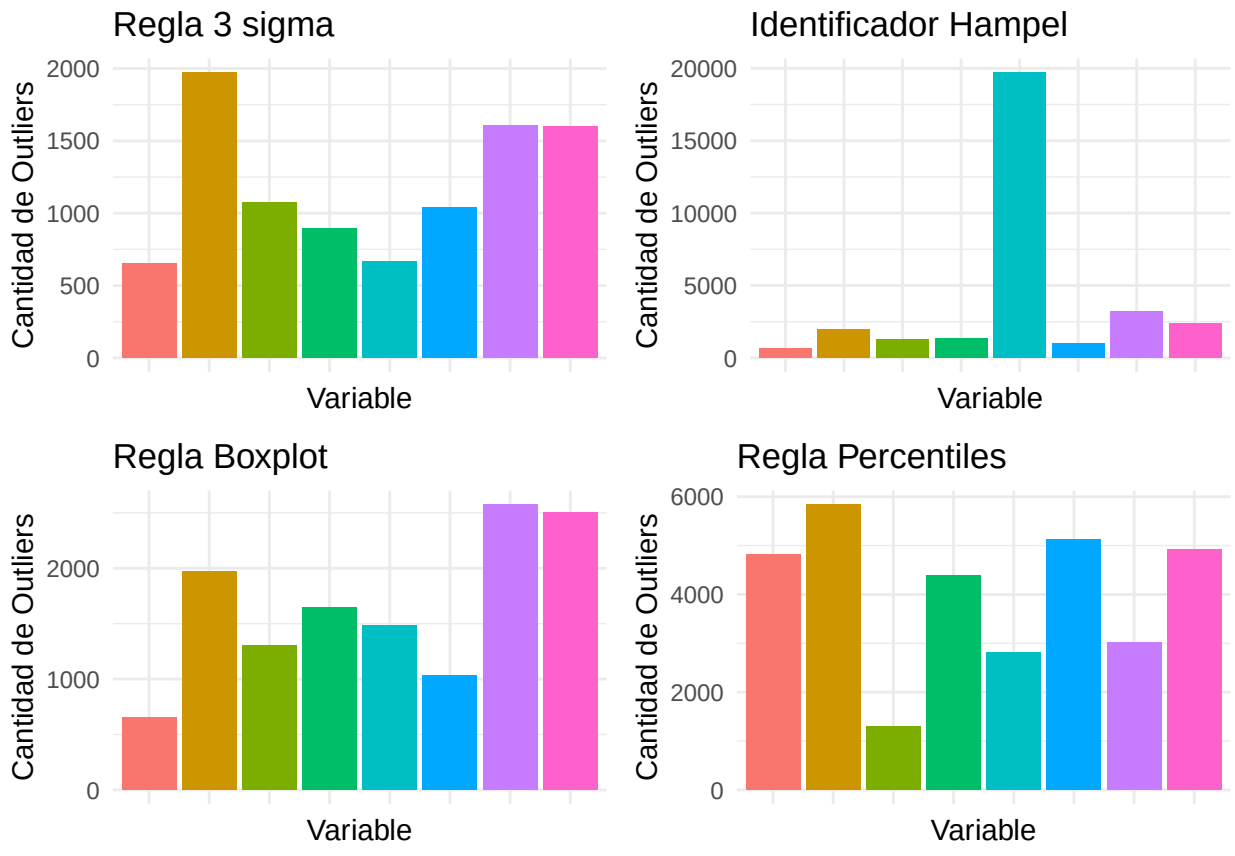


Figure 3: Gráficos de barras para detectar outliers.

La información resultante se presenta de manera clara y comparativa a través de gráficos de barras. Cada

barra en estos gráficos representa la cantidad de outliers detectados para una variable específica, empleando diferentes métodos de detección. En este caso, lo hemos aplicado sobre los datos de la estación Viveros.

Conclusiones