R_Básico_6

J.Ballesta

06/12/25

Resumen

En esta publicación veremos de forma práctica la presentación de series temporales en R y empezaremos a usar las librerias Tydyverse y Lubridate.

In diesem Beitrag werden wir die Darstellung von Zeitreihen in R auf praktische Weise betrachten und mit der Verwendung der Bibliotheken Tydiverse und Lubridate beginnen.

In this post, we will practically explore the presentation of time series in R and begin using the Tydiverse and Lubridate Libraries.

Introducción

Retomamos esta serie de publicaciones, para seguir descubriendo aspectos eminentemente prácticos del uso del lenguaje de programación R(2024), el IDE de programación RStudio(Posit team 2024) y Quarto(RStudio Team 2022) para la generación de informes de resultados.

En la mayoría de las ocasiones, las variables de nuestro interés evolucionarán de alguna forma en el tiempo, por ello es importante disponer de herramientas que nos permitan de forma efectiva presentar esta evolución en el tiempo y preparar los datos para su análisis. En esta publicación veremos que posibilidades nos brinda el package *ggplot2*(Wickham 2016a) para tratar datos en el tiempo, empezaremos a usar *tidyverse* (Wickham et al. 2019) y *lubridate*(Grolemund y Wickham 2011a).

Es esta publicación, haremos uso de los conjuntos de datos (en inglés: datasets) disponibles en la página web de la Universidad de Irvine (Univ. Irvine Machine Learning Repository), donde encontramos un extenso conjunto de datos para usar en nuestro aprendizaje de R. En concreto, usaremos “Power Consumption of Tetouan City” (Abdulwahed Salam 2018). Este conjunto de datos se refiere al consumo de electricidad de tres redes diferentes de la ciudad de Tetuan (Marruecos).

Disponemos de las siguientes variables:

- *$DateTime* : Fecha y hora de la recogida de datos
- *$Temperature*: Temperatura en la ciudad
- *$Humidity* : Humedad en la ciudad
- *$Windspeed*: Velocidad del viento
- *$General diffuse flows*
- *$Diffuse flows*
- *$Zone 1 Power consumption*

- *$Zone 2 Power consumption*
- *$Zone 3 Power consumption*

Librerías

Como siempre, en primer lugar cargaremos las librerías necesarias para nuestra publicación:

```
#
#    cargamos el package tidyverse que incluye a ggplot2, dplyr, ....
library(tidyverse)
```

```
── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
✔ dplyr     1.1.4     ✔ readr     2.1.5
✔ forcats   1.0.0     ✔ stringr   1.5.1
✔ ggplot2   3.5.1     ✔ tibble    3.2.1
✔ lubridate 1.9.4     ✔ tidyr     1.3.1
✔ purrr     1.0.2
── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
#    para el manejo de fechas usaremos Lubridate
library(lubridate)
#    para el análisis exploratorio de datos usamos DataExplorer
library(DataExplorer)
#
#      para las salidas de ggplot2, fijamos el tema del gráfico en B/N y la leyenda en la parte
#inferior de los gráficos
theme_set(theme_bw()+
            theme(legend.position="bottom"))
```

Conjunto de datos

Como hemos comentado en la introducción usaremos el conjunto de datos disponible en Power Consumption of Tetouan City, para ello descargamos los datos de la web y los cargamos en R, mediante la función *read.csv()* :

```
#
#    leemos los datos del fichero.csv y los almacenamos en una variable de trabajo
datos <- read.csv("Tetuan City power consumption.csv")
```

Comprobamos la subida de los datos y hacemos el análisis exploratorio inicial de los datos

```
#
#    cabecera y final de la tabla de datos
head(datos, 5)
```

```
      DateTime Temperature Humidity Wind.Speed general.diffuse.flows
1 1/1/2017 0:00       6.559     73.8      0.083               0.051
2 1/1/2017 0:10       6.414     74.5      0.083               0.070
3 1/1/2017 0:20       6.313     74.5      0.080               0.062
4 1/1/2017 0:30       6.121     75.0      0.083               0.091
5 1/1/2017 0:40       5.921     75.7      0.081               0.048
  diffuse.flows Zone.1.Power.Consumption Zone.2..Power.Consumption
1         0.119                34055.70                 16128.88
2         0.085                29814.68                 19375.08
3         0.100                29128.10                 19006.69
4         0.096                28228.86                 18361.09
5         0.085                27335.70                 17872.34
  Zone.3..Power.Consumption
1                  20240.96
2                  20131.08
3                  19668.43
4                  18899.28
5                  18442.41
```

```
tail(datos, 5)
```

```
           DateTime Temperature Humidity Wind.Speed general.diffuse.flows
52412 12/30/2017 23:10       7.010     72.4      0.080               0.040
52413 12/30/2017 23:20       6.947     72.6      0.082               0.051
52414 12/30/2017 23:30       6.900     72.8      0.086               0.084
52415 12/30/2017 23:40       6.758     73.0      0.080               0.066
52416 12/30/2017 23:50       6.580     74.1      0.081               0.062
      diffuse.flows Zone.1.Power.Consumption Zone.2..Power.Consumption
52412         0.096                31160.46                 26857.32
52413         0.093                30430.42                 26124.58
52414         0.074                29590.87                 25277.69
52415         0.089                28958.17                 24692.24
52416         0.111                28349.81                 24055.23
      Zone.3..Power.Consumption
52412                  14780.31
52413                  14428.81
52414                  13806.48
52415                  13512.61
52416                  13345.50
```

```
#    vemos la estructura de los datos
print("Estructura de datos:")
```

```
[1] "Estructura de datos:"
```

```
str(datos)
```

```
'data.frame':   52416 obs. of  9 variables:
 $ DateTime                : chr  "1/1/2017 0:00" "1/1/2017 0:10" "1/1/2017 0:20" "1/1/2017 0:30" ...
 $ Temperature             : num  6.56 6.41 6.31 6.12 5.92 ...
 $ Humidity                : num  73.8 74.5 74.5 75 75.7 76.9 77.7 78.2 78.1 77.3 ...
 $ Wind.Speed              : num  0.083 0.083 0.08 0.083 0.081 0.081 0.08 0.085 0.081 0.082 ...
```

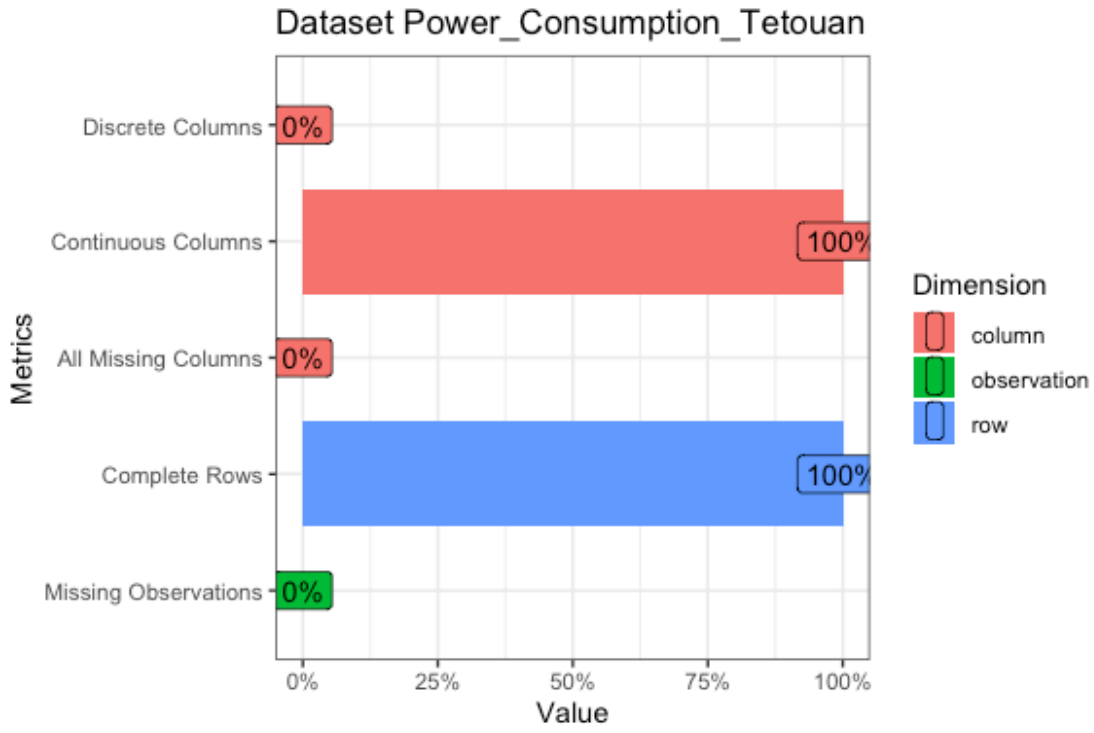
```
 $ general.diffuse.flows        : num    0.051  0.07  0.062  0.091  0.048  0.059  0.048  0.055  0.066
0.062 ...
 $ diffuse.flows           : num  0.119 0.085 0.1 0.096 0.085 0.108 0.096 0.093 0.141 0.111 ...
 $ Zone.1.Power.Consumption : num  34056 29815 29128 28229 27336 ...
 $ Zone.2..Power.Consumption: num  16129 19375 19007 18361 17872 ...
 $ Zone.3..Power.Consumption: num  20241 20131 19668 18899 18442 ...
```

```
#     vemos algunas magnitudes estadísticas de los datos, excepto la primera columna que es
de #fecha y hora.
summary(datos[,-1])
```

```
 Temperature       Humidity       Wind.Speed    general.diffuse.flows
 Min.   : 3.247   Min.   :11.34   Min.   :0.050   Min.   :   0.004
 1st Qu.:14.410   1st Qu.:58.31   1st Qu.:0.078   1st Qu.:   0.062
 Median :18.780   Median :69.86   Median :0.086   Median :   5.035
 Mean   :18.810   Mean   :68.26   Mean   :1.959   Mean   : 182.697
 3rd Qu.:22.890   3rd Qu.:81.40   3rd Qu.:4.915   3rd Qu.: 319.600
 Max.   :40.010   Max.   :94.80   Max.   :6.483   Max.   :1163.000
 diffuse.flows     Zone.1.Power.Consumption Zone.2..Power.Consumption
 Min.   :  0.011   Min.   :13896            Min.   : 8560
 1st Qu.:  0.122   1st Qu.:26311            1st Qu.:16981
 Median :  4.456   Median :32266            Median :20823
 Mean   : 75.028   Mean   :32345            Mean   :21043
 3rd Qu.:101.000   3rd Qu.:37309            3rd Qu.:24714
 Max.   :936.000   Max.   :52204            Max.   :37409
 Zone.3..Power.Consumption
 Min.   : 5935
 1st Qu.:13129
 Median :16415
 Mean   :17835
 3rd Qu.:21624
 Max.   :47598
```

Hacemos el análisis exploratorio inicial para ver que estado de datos tenemos:

```
#
#    vemos los datos, para ver como están distribuidas las variables y si hay datos faltantes
plot_intro(datos[,-1],
         title = "Dataset Power_Consumption_Tetouan",
         ggtheme = theme_bw())
```



De la salida de la función *str()*, vemos que el campo de fecha y hora (corresponde con la variable *$Datetime*) es una cadena de texto (tipo de datos chr), vamos a separar la columna *$DateTime* en dos columnas y formatear la columna de fecha como tipo de datos Date

```
#       para mantenerlo como referencia dejamos la variable original (datos) sin modificar y
#pasamos a otra variable (df) y convertimos la columna de fecha y hora a dos columnas
#separadas siguen siendo del tipo chr
df <- datos %>%
    separate(DateTime,
             into = c("fecha_original", "hora"),
             sep = " ")
#     formateamos la columna separada, mediante la función mdy() como  un tipo de datos de
fecha (Date)
df$fecha_original <- mdy(df$fecha_original)
#    añadimos algunas columnas adicionales que nos permitan hacer varios tipos de gráficos
df <- df %>%
    mutate(
        dia = day(fecha_original),
        semana = week(fecha_original),
        num_mes = month(fecha_original),
        nombre_mes = month(fecha_original, label=TRUE)
        )
#    comprobamos la salida de los datos y comprobamos efectivamente que el tipo de datos de
#la fecha ha cambiado a tipo Date
head (df, 5)
```

```
  fecha_original hora Temperature Humidity Wind.Speed general.diffuse.flows
1    2017-01-01 0:00       6.559    73.8      0.083               0.051
2    2017-01-01 0:10       6.414    74.5      0.083               0.070
```

```
3    2017-01-01 0:20      6.313    74.5     0.080             0.062
4    2017-01-01 0:30      6.121    75.0     0.083             0.091
5    2017-01-01 0:40      5.921    75.7     0.081             0.048
  diffuse.flows Zone.1.Power.Consumption Zone.2..Power.Consumption
1         0.119               34055.70               16128.88
2         0.085               29814.68               19375.08
3         0.100               29128.10               19006.69
4         0.096               28228.86               18361.09
5         0.085               27335.70               17872.34
  Zone.3..Power.Consumption dia semana num_mes nombre_mes
1                  20240.96   1     1       1        Jan
2                  20131.08   1     1       1        Jan
3                  19668.43   1     1       1        Jan
4                  18899.28   1     1       1        Jan
5                  18442.41   1     1       1        Jan
```

```
tail(df, 5)
```

```
      fecha_original  hora Temperature Humidity Wind.Speed
52412     2017-12-30 23:10       7.010    72.4     0.080
52413     2017-12-30 23:20       6.947    72.6     0.082
52414     2017-12-30 23:30       6.900    72.8     0.086
52415     2017-12-30 23:40       6.758    73.0     0.080
52416     2017-12-30 23:50       6.580    74.1     0.081
      general.diffuse.flows diffuse.flows Zone.1.Power.Consumption
52412                 0.040         0.096                31160.46
52413                 0.051         0.093                30430.42
52414                 0.084         0.074                29590.87
52415                 0.066         0.089                28958.17
52416                 0.062         0.111                28349.81
      Zone.2..Power.Consumption Zone.3..Power.Consumption dia semana num_mes
52412                  26857.32                  14780.31  30     52      12
52413                  26124.58                  14428.81  30     52      12
52414                  25277.69                  13806.48  30     52      12
52415                  24692.24                  13512.61  30     52      12
52416                  24055.23                  13345.50  30     52      12
      nombre_mes
52412        Dec
52413        Dec
52414        Dec
52415        Dec
52416        Dec
```

```
#
str(df)
```

```
'data.frame':   52416 obs. of  14 variables:
 $ fecha_original          : Date, format: "2017-01-01" "2017-01-01" ...
 $ hora                    : chr  "0:00" "0:10" "0:20" "0:30" ...
 $ Temperature             : num  6.56 6.41 6.31 6.12 5.92 ...
 $ Humidity                : num  73.8 74.5 74.5 75 75.7 76.9 77.7 78.2 78.1 77.3 ...
```

```
$ Wind.Speed              : num  0.083 0.083 0.08 0.083 0.081 0.081 0.08 0.085 0.081 0.082 ...
 $ general.diffuse.flows  : num  0.051 0.07 0.062 0.091 0.048 0.059 0.048 0.055 0.066
0.062 ...
$ diffuse.flows           : num  0.119 0.085 0.1 0.096 0.085 0.108 0.096 0.093 0.141 0.111 ...
$ Zone.1.Power.Consumption : num  34056 29815 29128 28229 27336 ...
$ Zone.2..Power.Consumption: num  16129 19375 19007 18361 17872 ...
$ Zone.3..Power.Consumption: num  20241 20131 19668 18899 18442 ...
$ dia                     : int  1 1 1 1 1 1 1 1 1 1 ...
$ semana                  : num  1 1 1 1 1 1 1 1 1 1 ...
$ num_mes                 : num  1 1 1 1 1 1 1 1 1 1 ...
$ nombre_mes              : Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 1 ...
```

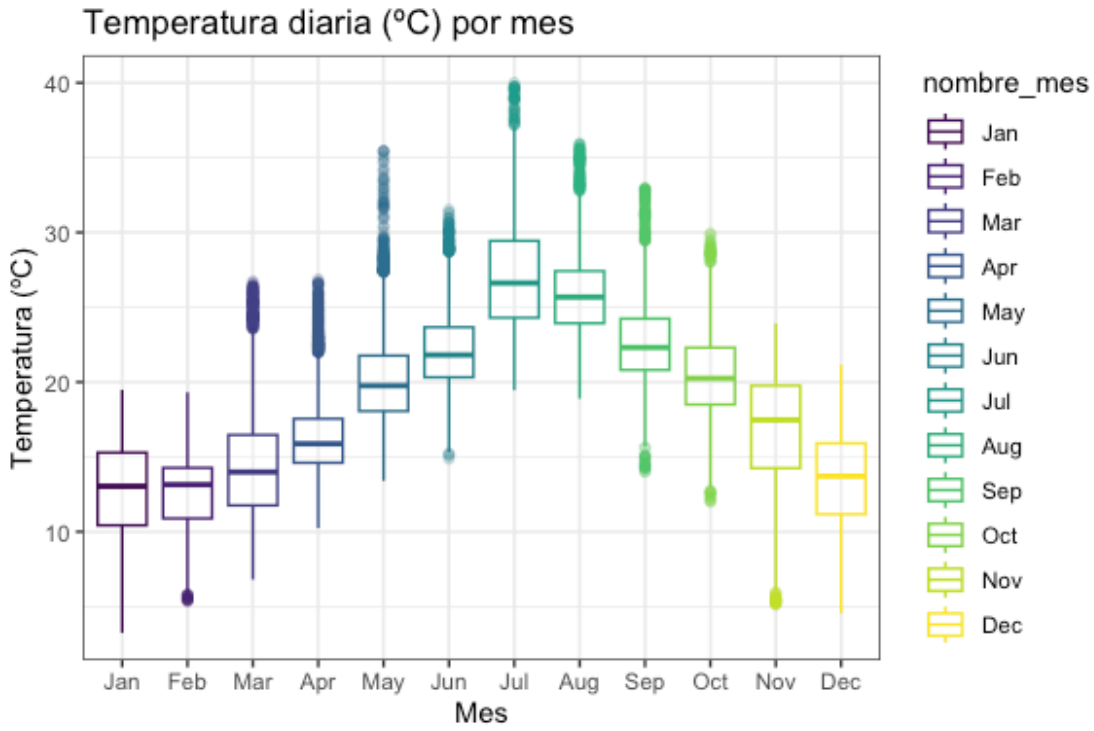
Verificamos que el rango de fechas esté completo.

```
#
#   Verificar la completitud temporal
fechas_completas <- seq(from = min(df$fecha),
                        to = max(df$fecha),
                        by = "day")
fechas_faltantes <- setdiff(fechas_completas, unique(df$fecha))
#
if(length(fechas_faltantes) > 0) {
    warning("Faltan datos para las fechas: ", paste(fechas_faltantes, collapse = ", "))
}
```

Gráficas en función del tiempo.

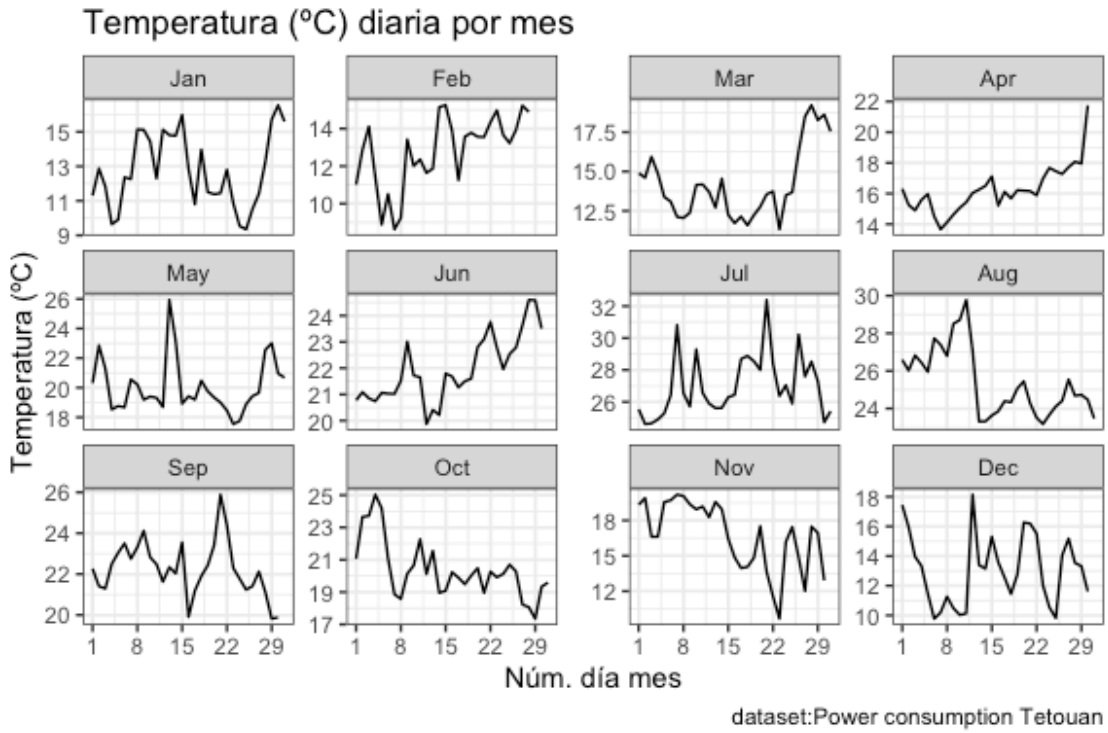
A continuación presentaremos los datos en función del tiempo, para ello una de las primeras gráficas puede ser un *geom_boxplot()* de la *$Temperature* en función de los meses :

```
#
#     Pasamos los datos a ggplot2 y podemos ver que en los meses de marzo a octubre hay
muchos #valores atípicos (outliers) por alta temperatura.
df %>%
    ggplot()+
    geom_boxplot(aes(x=nombre_mes, y= Temperature, color=nombre_mes), alpha=0.2)+
    labs(title = "Temperatura diaria (ºC) por mes")+
    xlab("Mes")+
    ylab("Temperatura (ºC)")+
    theme(legend.position = "right")
```



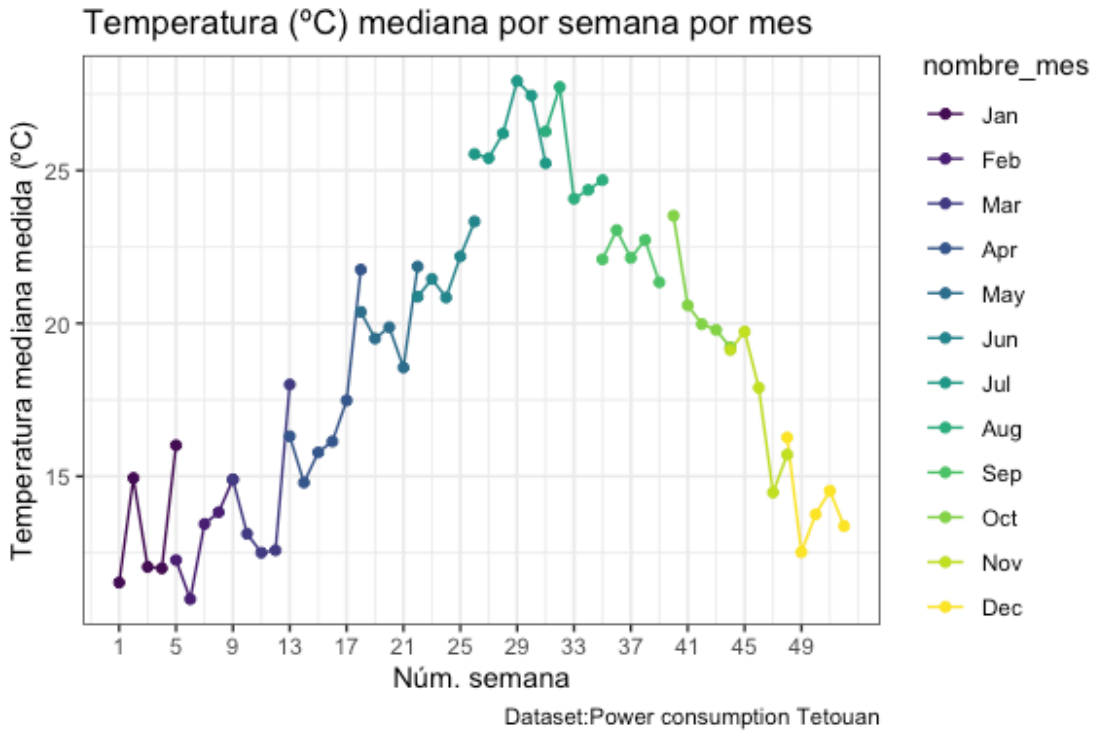
Los datos están compuestos por por varias mediciones en el día, para simplificar la presentación optamos por usar la función *median()* y obtener el valor que centra la distribución de la medida:

```
#
#         los datos están compuestos de varios registros en el mismo día, presentamos la
#temperatura mediana por día y separada por meses(facet_wrap)
df %>%
    group_by(nombre_mes, dia)%>%
    summarise(temp_mediana=median(Temperature)) %>%
     ggplot()+
     geom_line(aes(x=dia, y=temp_mediana))+
     labs(title="Temperatura (ºC) diaria por mes",
          caption = "dataset:Power consumption Tetouan")+
     xlab("Núm. día mes")+
     ylab("Temperatura (ºC)")+
     scale_x_continuous(limits=c(1,31), breaks=seq(1,31, by=7))+
       facet_wrap(~nombre_mes, scales="free_y") #mediante el parámetro scales dejamos que
ggplot2 ajuste la escala en el eje Y
```



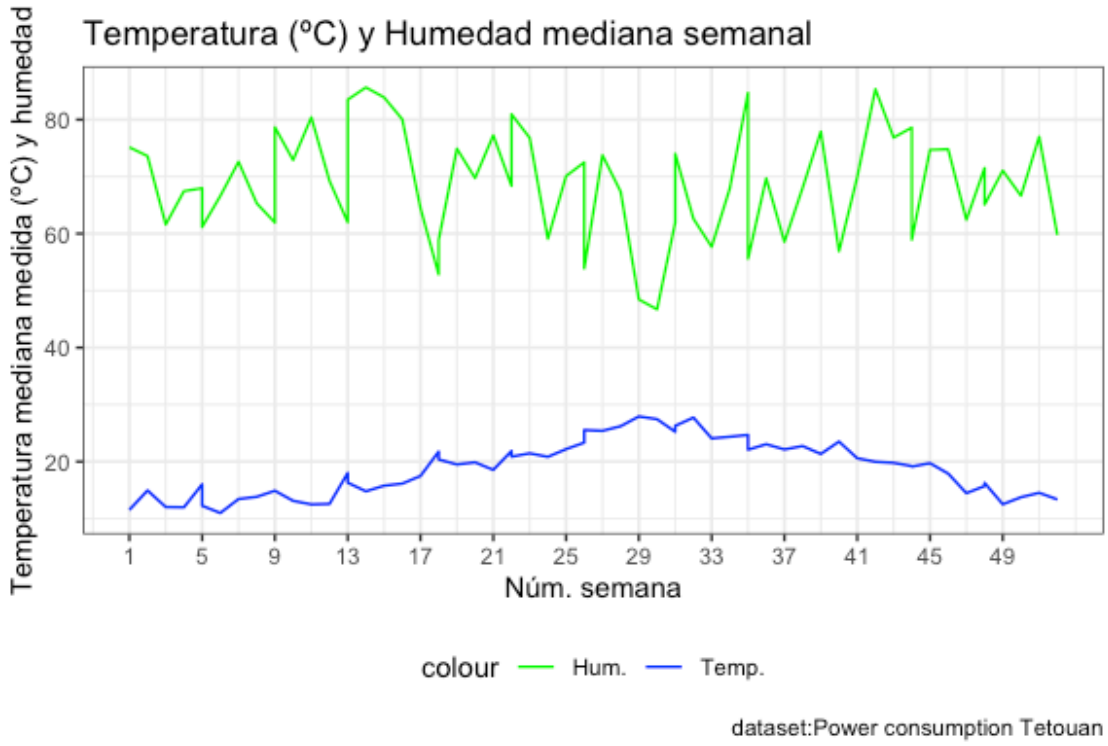
También podemos optar por presentar la evolución global de la temperatura mediana semanal en el año distinguiendo cada mes, para ello:

```
#
#    en este caso presentaremos la temperatura mediana en el año, distinguiendo por meses
df %>%
    group_by(nombre_mes, semana)%>%
    summarise(temp_mediana=median(Temperature)) %>%
    ggplot()+
    geom_line(aes(x=semana, y=temp_mediana, color=nombre_mes))+
    geom_point(aes(x=semana, y=temp_mediana, color=nombre_mes))+
    labs(title="Temperatura (ºC) mediana por semana por mes",
         caption = "Dataset:Power consumption Tetouan")+
    xlab("Núm. semana")+
    ylab("Temperatura mediana medida (ºC)")+
    scale_x_continuous(limits=c(1,52), breaks=seq(1,52, by=4))+
    theme(legend.position = "right")
```



Podemos combinar en el mismo gráfico varias variables, en este caso optamos por combinar en un mismo gráfico *$Temperature* y *$Humidity*

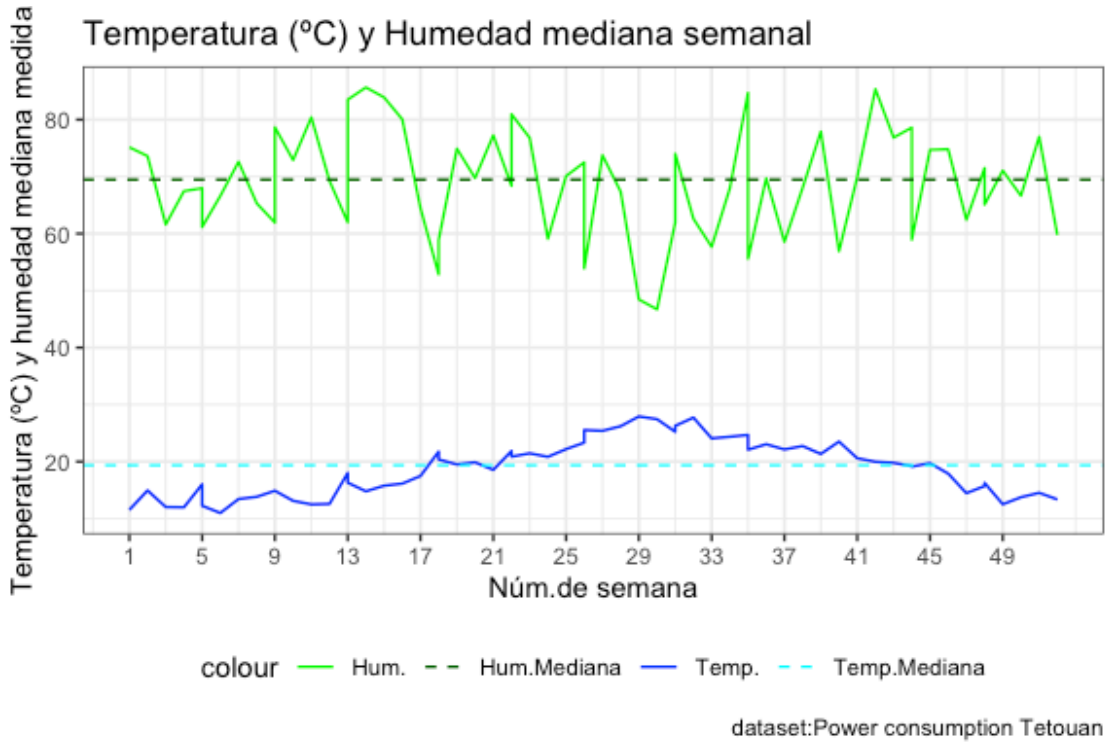
```
#
#
df %>%
    group_by(nombre_mes, semana)%>%
    summarise(temp_mediana=median(Temperature),
              hum_mediana=median(Humidity)) %>%
    ggplot()+
    geom_line(aes(x=semana, y=temp_mediana, color="Temp."))+
    geom_line(aes(x=semana, y=hum_mediana, color="Hum."))+
    scale_color_manual(values = c("Temp."="blue", "Hum."="green"))+
    labs(title="Temperatura (ºC) y Humedad mediana semanal",
         caption = "dataset:Power consumption Tetouan")+
     xlab("Núm. semana")+
     ylab("Temperatura mediana medida (ºC) y humedad")+
    scale_x_continuous(limits=c(1,52), breaks=seq(1,52, by=4))
```



Para terminar podemos dotar el gráfico de alguna linea de referencia adicional como el valor mediano anual, tanto en *$Temperature* como *$Humidity*:

```
df %>%
    group_by(nombre_mes, semana)%>%
    summarise(temp_mediana=median(Temperature),
              hum_mediana=median(Humidity)) %>%
    ggplot()+
    geom_line(aes(x=semana, y=temp_mediana, color="Temp."))+
    geom_line(aes(x=semana, y=hum_mediana, color="Hum."))+
    geom_hline(aes(yintercept=median(temp_mediana), color="Temp.Mediana"),
               linetype="dashed")+
    geom_hline(aes(yintercept=median(hum_mediana), color="Hum.Mediana"),
               linetype="dashed")+
    scale_color_manual(values = c("Temp."="blue", "Hum."="green",
                                  "Temp.Mediana"="cyan","Hum.Mediana"="darkgreen"))+
    labs(title="Temperatura (ºC) y Humedad mediana semanal",
         caption = "dataset:Power consumption Tetouan")+
    xlab("Núm.de semana")+
    ylab("Temperatura (ºC) y humedad mediana medida")+
    scale_x_continuous(limits=c(1,52), breaks=seq(1,52, by=4))
```

```
`summarise()` has grouped output by 'nombre_mes'. You can override using the
`.groups` argument.
```



En esta publicación hemos visto como preparar de una forma bastante sencilla los datos para representar su evolución en el tiempo, con los packages *dplyr*(Wickham et al. 2023), *lubridate*(Grolemund y Wickham 2011b) y *ggplot2*(Wickham 2016b).

Este tipo de preparación y presentación de datos en un software “standard”, en el sentido de habitualmente disponible, como Excel requiere de un mayor esfuerzo, dado que habría que recurrir a soluciones como PowerQuery para las distintas tablas de agrupamiento de los datos por mes o semana (el tamaño del fichero resultante en estos casos crece mucho) y la salida gráfica en Excel está bastante más limitada. La posibilidad de recurrir a soluciones como el uso de VBA, sólo estarían disponibles para gente con mucha experiencia en programación en VBA y seguramente tomaría su tiempo finalizar este tipo de resumen de los datos.

Referencias

Abdulwahed Salam, Abdelaaziz El Hibaoui. 2018. «Power Consumption of Tetouan City». UCI Machine Learning Repository. https://doi.org/10.24432/C5B034.

Grolemund, Garrett, y Hadley Wickham. 2011a. «Dates and Times Made Easy with lubridate» 40. https://www.jstatsoft.org/v40/i03/.

———. 2011b. «Dates and Times Made Easy with lubridate» 40. https://www.jstatsoft.org/v40/i03/.

Posit team. 2024. *RStudio: Integrated Development Environment for R*. Boston, MA: Posit Software, PBC. http://www.posit.co/.

R Core Team. 2024. «R: A Language and Environment for Statistical Computing». https://www.R-project.org/.

RStudio Team. 2022. *Quarto: A next-generation publishing system*. RStudio, PBC. https://quarto.org/.

Wickham, Hadley. 2016a. «ggplot2: Elegant Graphics for Data Analysis». https://ggplot2.tidyverse.org.

———. 2016b. «ggplot2: Elegant Graphics for Data Analysis». https://ggplot2.tidyverse.org.

Wickham, Hadley, Mara Averick, Jennifer Bryan, Winston Chang, Lucy D'Agostino McGowan, Romain François, Garrett Grolemund, et al. 2019. «Welcome to the tidyverse» 4: 1686. https://doi.org/10.21105/joss.01686.

Wickham, Hadley, Romain François, Lionel Henry, Kirill Müller, y Davis Vaughan. 2023. «dplyr: A Grammar of Data Manipulation». https://CRAN.R-project.org/package=dplyr.