Universidad
Internacional
Menéndez Pelayo

Máster Universitario en Investigación en Inteligencia Artificial

TRABAJO SERIES TEMPORALES

Datos temporales y complejos

Laura Rodríguez Navas

Diciembre de 2020

rodrigueznavas@posgrado.uimp.es

Índice general

Capítulo 1

Análisis de la serie temporal

1.1. Introducción

Una empresa tecnológica cuya área de negocio es la inteligencia artificial es contratada por una empresa eléctrica para que diseñe un sistema de recomendación que haga ofertas personalizadas a sus clientes sobre los paquetes energéticos más adecuados a sus consumos. Para ello, la empresa primero debe llevar a cabo un análisis exhaustivo de los consumos energéticos y diseñar una técnica de predicción que sea capaz de predecir dichos consumos con un horizonte temporal dado. La empresa eléctrica suministra a la empresa tecnológica para dicho cometido los consumos eléctricos desde el 01 de enero de 2015 hasta el 31 de diciembre de 2015 medidos con una frecuencia temporal de 10 minutos.

Se pide:

1. Analizar la serie temporal de consumos eléctricos y describir brevemente las principales características de esta.
2. Realizar un estudio para determinar si la serie temporal presenta *outliers*. Describir brevemente el estudio realizado y las conclusiones alcanzadas.
3. Seleccionar un artículo publicado en una revista científica de prestigio internacional en el que se presente un método de predicción. Estudiar de manera detallada el método publicado y describirlo de forma resumida.
4. Aplicar el método seleccionado en el apartado anterior para obtener una predicción del consumo eléctrico con un horizonte temporal de 4 horas para los días de la semana desde el lunes 8 de junio hasta el domingo 14 de junio. Para ello, se debe implementar un método de predicción en el lenguaje de programación que se estime oportuno o bien usar software libre disponible como WEKA, R, KEEL, ... Una vez obtenidas las predicciones del periodo indicado, visualizar los resultados (predicciones, errores, ...).

El trabajo de investigación se realiza utilizando el lenguaje de programación R y todos los datos y el código necesarios para reproducir el estudio se proporcionan dentro de este documento para garantizar que éste sea completamente reproducible.

Se utilizan los siguientes paquetes para el análisis:

```
library(tidyverse)
library(lubridate)
library(xts)
library(forecast)
```

- tidyverse es un conjunto de librerías de R diseñadas para la *"Ciencia de datos"*.
- lubridate es una librería que permite manipular fechas e intervalos de tiempo.
- xts es una librería que permite convertir datos en series temporales.
- forecast es una librería que contiene métodos y herramientas para analizar series temporales

1.2. Datos de la serie temporal

Los datos están disponibles para descargar aquí. Una vez descargados, se cargan en el espacio de trabajo.

```
data <- read.csv("Demanda_2015.csv", header = FALSE, sep = ",")
```

Buscamos valores perdidos.

```
sum(is.na(data))
```

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

Vemos que no existen valores perdidos, por lo tanto no tendremos que eliminarlos.

```
# Establecemos nombres a las columnas
colnames(data) <- c("date", "time", "demand")
head(data, 10)
```

```
##          date time demand
## 1  01/01/2015 0:00  25459
## 2  01/01/2015 0:10  25591
## 3  01/01/2015 0:20  25531
## 4  01/01/2015 0:30  25453
## 5  01/01/2015 0:40  25329
## 6  01/01/2015 0:50  25247
## 7  01/01/2015 1:00  25093
## 8  01/01/2015 1:10  24853
## 9  01/01/2015 1:20  24678
## 10 01/01/2015 1:30  24391
```

```
summary(data)
```

```
##      date               time               demand
##  Length:52560       Length:52560       Min.   :17985
##  Class :character   Class :character   1st Qu.:24392
##  Mode  :character   Mode  :character   Median :28566
##                                        Mean   :28349
##                                        3rd Qu.:31664
##                                        Max.   :40648
```

Tenemos la variable *date* como *character*, la variable *time* como *character* y la variable *demand* como *integer*. Para su análisis posterior, el marco de datos de fechas y horas se formatea. Para ello, primero crearemos una nueva columna *datetime*, donde juntaremos las fechas con las horas usando la función *cbind()*. Luego procederemos a la conversión de tipos *character* a *POSIXct* para la fecha-hora usando la función *parse_date_time()*.

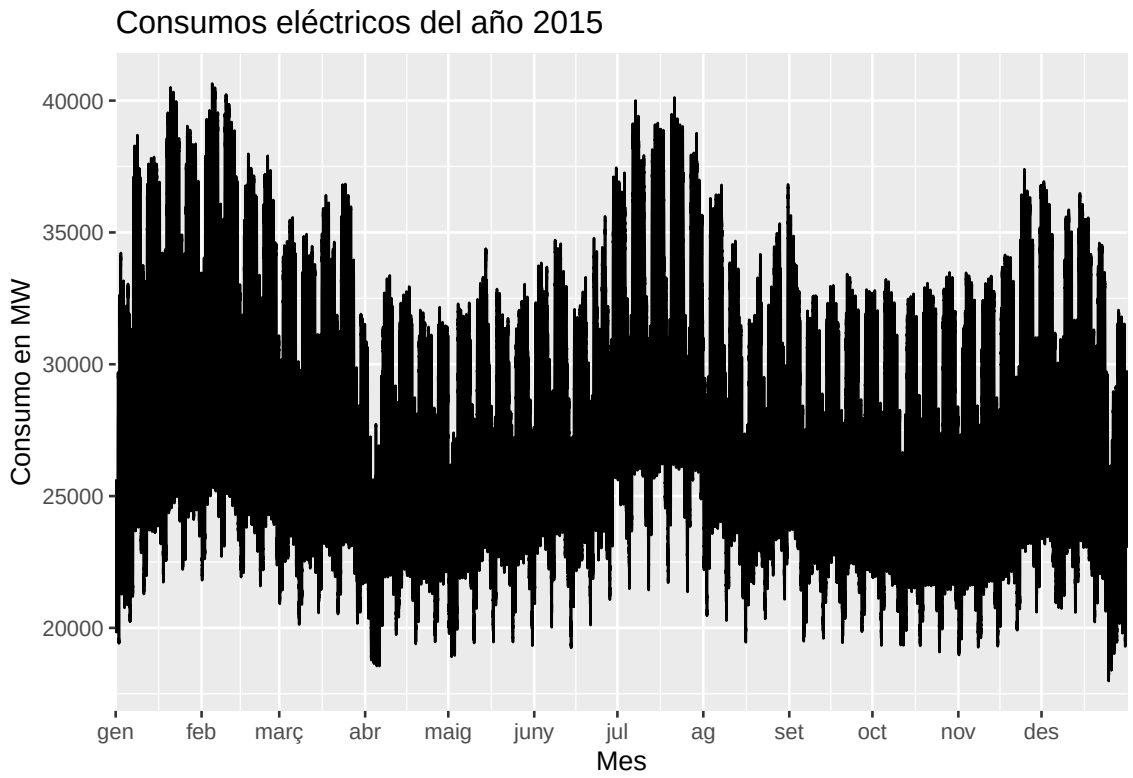
```
data <- cbind(datetime = paste(data$date, data$time), data)
data$datetime <- parse_date_time(data$datetime, "dmY HMS",
                                 truncated = 3, tz = "UTC")
head(data, 10)
```

```
##                datetime       date time demand
## 1  2015-01-01 00:00:00 01/01/2015 0:00  25459
## 2  2015-01-01 00:10:00 01/01/2015 0:10  25591
## 3  2015-01-01 00:20:00 01/01/2015 0:20  25531
## 4  2015-01-01 00:30:00 01/01/2015 0:30  25453
## 5  2015-01-01 00:40:00 01/01/2015 0:40  25329
## 6  2015-01-01 00:50:00 01/01/2015 0:50  25247
## 7  2015-01-01 01:00:00 01/01/2015 1:00  25093
## 8  2015-01-01 01:10:00 01/01/2015 1:10  24853
## 9  2015-01-01 01:20:00 01/01/2015 1:20  24678
## 10 2015-01-01 01:30:00 01/01/2015 1:30  24391
```

1.3. Visualización de la serie temporal

La siguiente figura muestra la distribución del consumo eléctrico a lo largo del 2015.

```
ggplot(data = data, aes(x = datetime, y = demand)) +
  geom_line() +
  scale_x_datetime(date_labels = "%b",
                   breaks = "1 month",
                   expand = c(0, 0)) +
  ggtitle("Consumos eléctricos del año 2015") +
  xlab("Mes") +
  ylab("Consumo en MW")
```

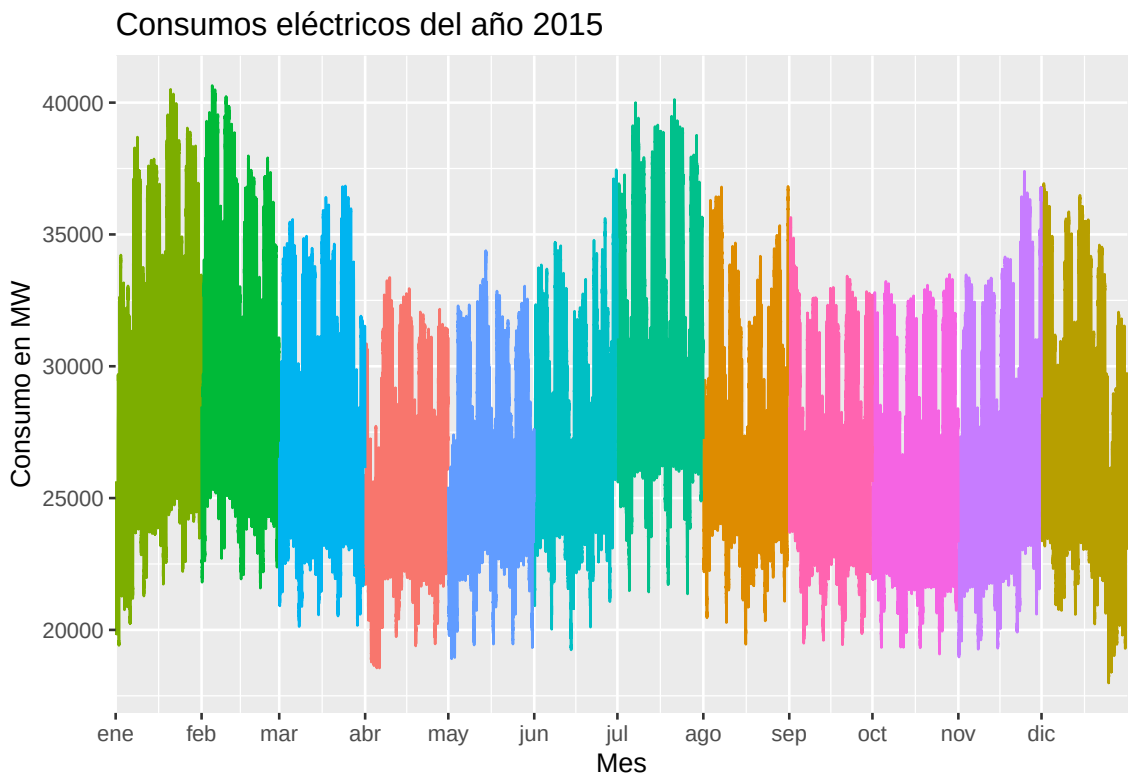


Observando la figura se puede detectar una dependencia estacional del consumo eléctrico, aunque pueden existir otros factores que pueden afectarla, como los días festivos, los fines de semana, ... Parece que el principal desafío introducido por la serie temporal podría ser su estacionalidad múltiple. En ese caso, las siguientes estacionalidades pueden estar presentes:

- Las personas usan la mayoría de sus electrodomésticos en determinadas horas del día, lo que resulta en una estacionalidad diaria.
- Las personas usan la electricidad de manera diferente los días entre semana y los fines de semana, lo que resulta en una estacionalidad semanal.
- Las personas usan la electricidad de manera diferente en diferentes épocas del año, lo que resulta en una estacionalidad anual.

Para mejorar la visualización de la distribución del consumo eléctrico a lo largo del 2015, la siguiente figura es muy útil.

```
ggplot(data, aes(x = datetime, y = demand)) +
  geom_line(aes(colour = month)) +
  scale_x_datetime(date_labels = "%b",
                   breaks = "1 month",
                   expand = c(0, 0)) +
  ggtitle("Consumos eléctricos del año 2015") +
  xlab("Mes") +
  ylab("Consumo en MW") +
  theme(legend.position = "none")
```



Es importante tener en cuenta cómo aumenta el consumo de la electricidad tanto en los meses de verano, de mayo a septiembre, como en los días de invierno, desde noviembre hasta febrero. Esto se debe a que las personas usan la electricidad para enfriar y calentar sus viviendas. Durante los períodos de invierno y de verano, el consumo eléctrico es de manera clara mayor salvo, probablemente, en los períodos vacacionales. Como se puede observar en el mes de agosto. En resumen, visualmente podemos deducir que los consumos eléctricos siguen un patrón de subidas y bajadas en una tendencia que varía de manera estacional.

1.4. Componentes de la serie temporal

Es frecuente analizar las series temporales des del punto de vista de sus componentes estructurales. Pero primero necesitamos que R trate los datos como una serie temporal, así que para ello tendremos que determinar apropiadamente sus características con la función *ts()*. Para definir la serie correctamente escribimos:

```
ts <- ts(data$demand, frequency = 24 * 60 / 10)
# El argumento frequency de la función *ts()* se utiliza para indicar
# la periodicidad de la serie (en este caso es de 10 minutos)
```

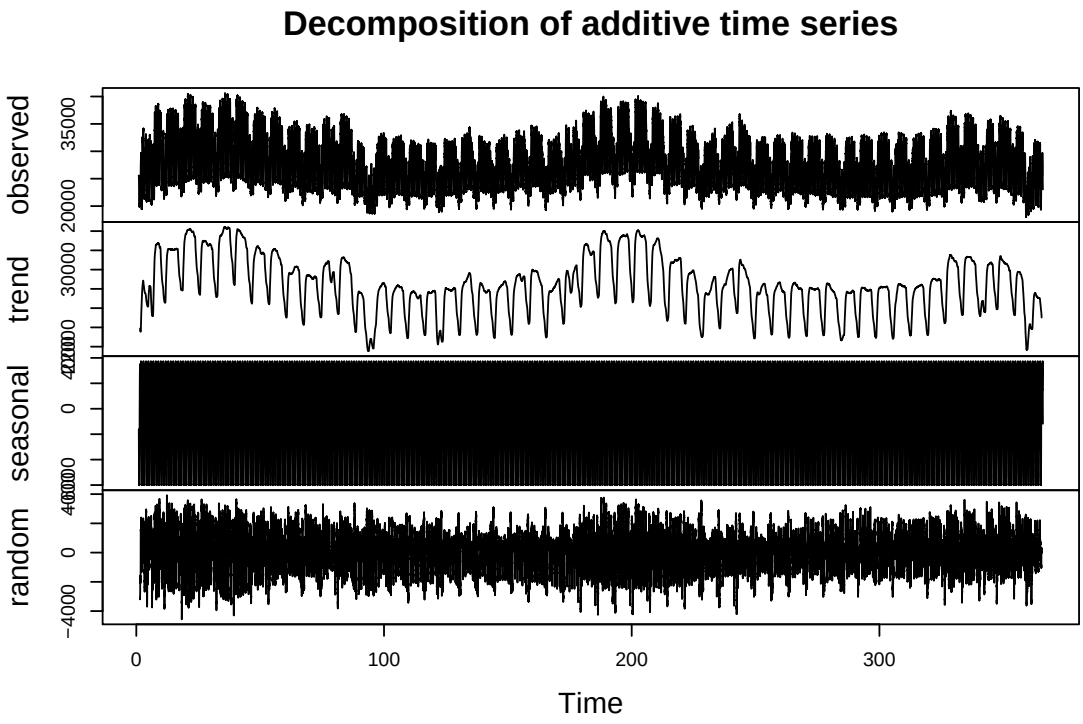
Los componentes de una serie temporal son:

- Tendencia: un aumento o disminución a largo plazo de los datos se denomina tendencia. No tiene por qué ser necesariamente lineal. Es el patrón subyacente en los datos a lo largo del tiempo.
- Estacional o Periódico: cuando una serie está influenciada por factores estacionales. Como en este caso y hemos podido visualizar en el apartado anterior.

- Cíclico: cuando los datos muestran subidas y bajadas que no son de un mismo período.

En R se pueden averiguar los componentes de la serie temporal con la siguiente función:

```
componentes.ts <- decompose(ts)
plot(componentes.ts)
```

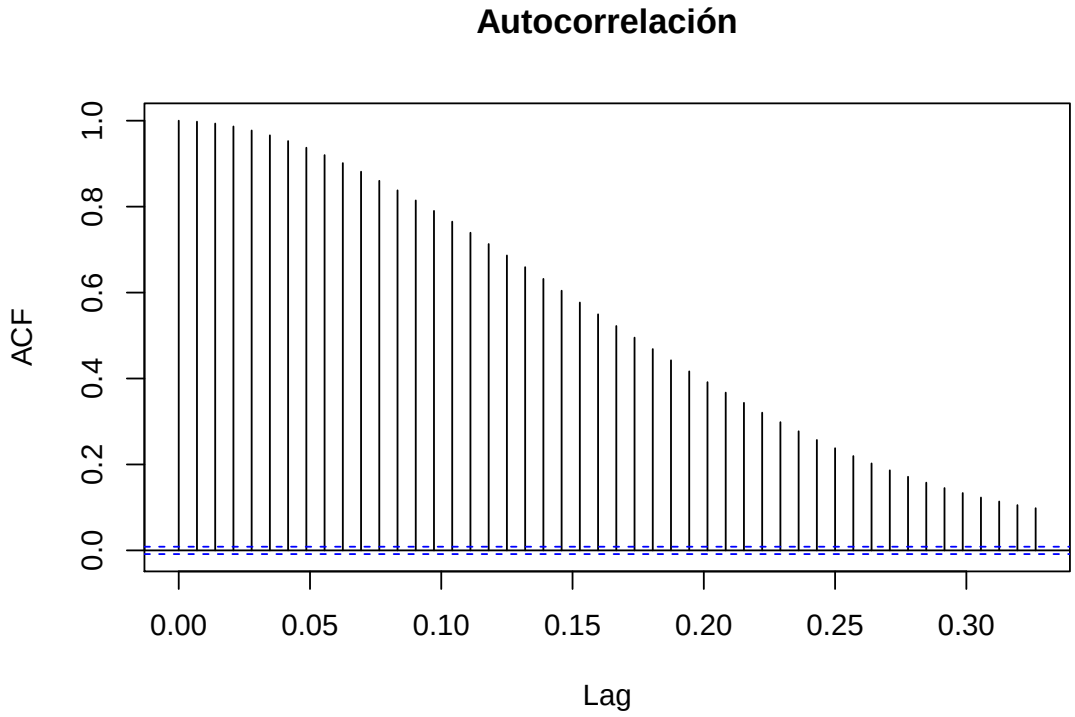


En la figura se observant cuatro sub-figuras:

- observed: los datos actuales.
- trend: el movimiento general hacia arriba o hacia abajo de los datos.
- seasonal: cualquier patrón mensual o anual de los os datos.
- random: parte inexplicable de los datos.

A partir de las sub-figuras se puede determinar que la serie temporal no presenta una tendencia. Con esto se puede afirmar que la serie es estacionaria. También se puede observar que la serie es estacionaria en cuando a la varianza, ya que no se aprecia gran variabilidad. En cambio, sí que presenta una estacionalidad muy marcada (el consumo aumenta durante los meses de invierno y verano). Por ese motivo, para mejorar las futuras predicciones se debe eliminar la estacionalidad de la serie temporal original. Pero antes, para confirmar que la serie es estacionaria trazaremos la gráfica de su función de autocorrelación, usando la función *acf()*.

```
acf(ts, main = "Autocorrelación")
```



Se puede confirmar que la serie es estacionaria, ya que el valor de la función de autocorrelación decae de manera exponencial a medida que aumentan los rezagos en el tiempo. Como es estacionaria será más fácil de predecir.

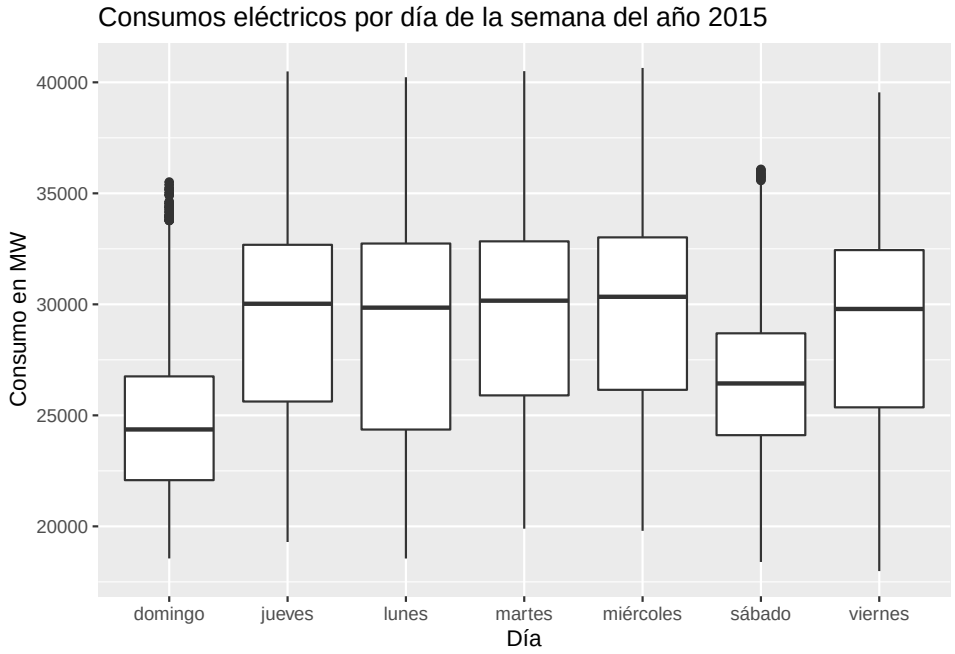
Capítulo 2

Outliers

2.1. Identificación de outliers

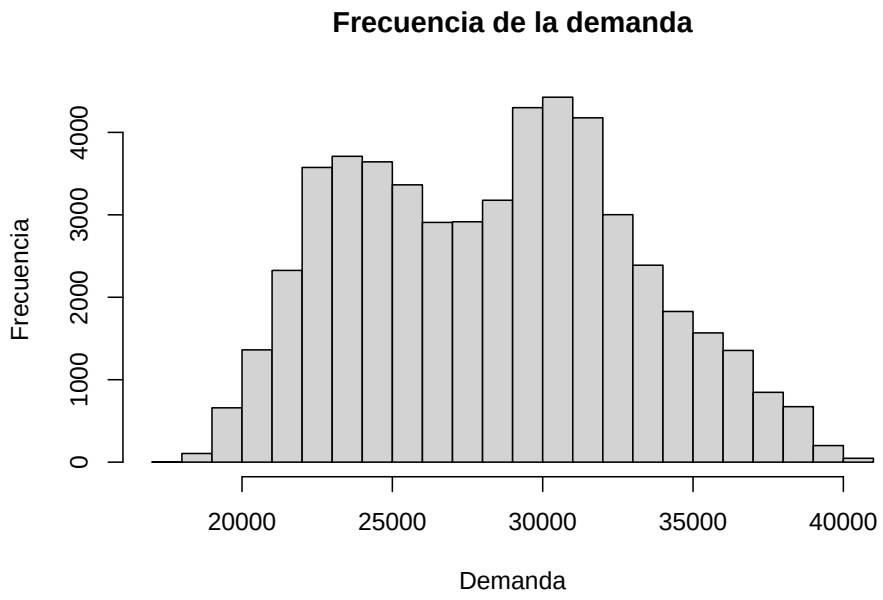
En el capítulo anterior se ha comentado que podrían existir factores que afecten a la serie temporal. Uno de esos factores podría ser que, durante los fines de semana, el consumo eléctrico disminuya considerablemente en comparación con el resto de los días de la semana, y eso provocaría la aparición de valores atípicos en la serie temporal. Otro factor que puede hacer disminuir el consumo eléctrico considerablemente y provocar la aparición de valores atípicos en la serie temporal, son los días festivos. Así, se han considerado estos dos factores para localizar valores atípicos en la serie temporal, si es que tiene. Primero se analiza el consumo eléctrico durante los fines de semana mediante la siguiente figura. Pero antes se añade a los datos una columna nueva *day*, para indicar los días de la semana.

```
data$day <- format(data$datetime, format = "%A")
ggplot(data, aes(day, demand)) +
  geom_boxplot() +
  ggtitle("Consumos eléctricos por día de la semana del año 2015") +
  xlab("Día") + ylab("Consumo en MW")
```



Como se observa en la figura durante los fines de semana (sábado y domingo), la demanda disminuye considerablemente en comparación con el resto de los días de la semana y aparecen valores atípicos. Por otro lado, ahora nos fijaremos en qué rangos se mueve la demanda del consumo eléctrico a lo largo del 2015. Se usa un histograma para que aporte esa información con un simple vistazo.

```
hist(data$demand,
     main = "Frecuencia de la demanda",
     ylab = "Frecuencia",
     xlab = "Demanda")
```



La figura anterior muestra una distribución extraña para la demanda.

- Hay poca frecuencia de la demanda, cuando la demanda es inferior a 22000 MW y superior a 34000 MW (aproximadamente).
- La frecuencia de la demanda es frecuente entre 22000 MW y 34000 MW.
- La baja demanda es más frecuente de lo normal dentro de conjunto de datos de la serie temporal, parece que predominan bastantes valores atípicos.

Después de este breve análisis, es interesante estudiar cuantos valores atípicos hay en la demanda. Para ello, el primer paso es convertir la variable demanda *demand*, una variable numérica en categórica. Se define la siguiente categorización, basándonos en el histograma, que a priori parece razonable:

- hasta 21000 MW la demanda es baja (*low*).
- de 21000 a 34000 MW la demanda no se consideraría un valor atípico (*medium*).
- mayor de 34000 MW la demanda es muy alta (*high*).

```
breakPoints <- c(0, 22000, 34000, Inf)
categories <- c("low", "medium", "high")
data$demand.C <- cut(data$demand, breaks = breakPoints, labels = categories)
summary(data$demand.C)
```

```
##    low medium   high
##   4453  41588   6519
```

Con esta categorización, podríamos decir que en la serie existen aproximadamente 10972 valores atípicos (4453 + 6519). Esta cantidad es bastante alta, y en ese caso sería recomendable aplicar un método de reemplazo de estos valores atípicos que parece que se encuentran fuera del rango medio de la demanda. Además, como curiosidad, antes de reemplazar los *outliers* de la serie, analizaremos si muchos de estos valores atípicos pertenecen a días festivos, fines de semana o días en períodos de vacaciones, como parece que hemos detectado en nuestro primer análisis.

```
demand.low <- data[data$demand.C == "low", ]
table(demand.low$day)
```

```
##
##   domingo    jueves     lunes    martes miércoles    sábado   viernes
##      1824       246       941       286       184       619       353
```

```
demand.high <- data[data$demand.C == "high", ]
table(demand.high$day)
```

```
##
##   domingo    jueves     lunes    martes miércoles    sábado   viernes
##        19      1272      1327      1378      1385        67      1071
```

Se puede observar que los días de menos demanda eléctrica son los fines de semana. Contrariamente los días de más demanda eléctrica son días entre semana. Pero se observa un dato extraño, hay muchos lunes donde la demanda es baja. Esto seguramente se debe a que bastantes días festivos del año 2015 fueron un lunes.

```
demand.monday <- demand.low[demand.low$day == "lunes", ]
sort(table(demand.monday$date))
```

```
##
## 12/01/2015 22/06/2015 08/06/2015 14/12/2015 02/03/2015 10/08/2015 24/08/2015
##          1          1          6          9         14         15         15
## 30/11/2015 23/03/2015 16/03/2015 05/01/2015 09/03/2015 21/12/2015 07/12/2015
##         15         16         18         20         20         22         23
## 17/08/2015 18/05/2015 23/11/2015 01/06/2015 11/05/2015 30/03/2015 14/09/2015
##         24         24         24         25         25         26         27
## 15/06/2015 25/05/2015 05/10/2015 27/04/2015 07/09/2015 13/04/2015 21/09/2015
##         27         27         28         28         29         29         29
## 28/12/2015 16/11/2015 19/10/2015 20/04/2015 28/09/2015 04/05/2015 09/11/2015
##         30         31         31         31         31         32         32
## 02/11/2015 26/10/2015 06/04/2015 12/10/2015
##         33         34         40         49
```

Observamos que la afirmación anterior es verdad y vemos algunos ejemplos:

- 12/10/2015: Día de la Hispanidad.
- 06/04/2015: Lunes de Pascua.
- 26/10/2015: Esta fecha es muy interesante, ya que coincide que es el día después del cambio de hora (horario de invierno).
- 09/11/2015: Almudena (sólo en Madrid).
- 28/12/2015: en período vacacional de Navidades.

En el apartado siguiente, se remplazarán los valores atípicos de la demanda. Porqué para la futura predicción del capítulo cuatro, si el conjunto de datos de la serie incluye bastantes valores atípicos la predicción será engañosa.

2.2. Reemplazo de outliers

Para reemplazar los *outliers*, se necesitan dos umbrales. Como se ha comentado, se opta por el reemplazo. No se opta por la eliminación de los valores atípicos para no perder datos representativos de la serie. La eliminación de bastantes datos representativos de la serie puede tener un impacto negativo en las futuras predicciones de la serie. Los umbrales que se definirán indicarán qué valores de la demanda serán reemplazados.

Se usa la función *summary()*, que nos devolverá información sobre los cuartiles de la demanda. Los cuartiles son valores que dividen el conjunto de datos de la serie temporal en diferentes partes. Estos se utilizarán para definir los umbrales.

```
summary(data$demand)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   17985   24392   28566   28349   31664   40648
```

En el reemplazo que nos traemos entre manos, se observa que el primer cuartil ($Q1$) está en 24392 MW y que el tercer cuartil ($Q3$) está en 31664 MW. El uso de los cuartiles en el reemplazo se debe a que entre $Q1$ y $Q3$ sabemos que se encuentran el 50 % de los valores de la serie. La distancia entre los valores de $Q1$ y $Q3$ se le llama rango intercuantílico (*IQR: InterQuantile Range*). En este caso, el valor de *IQR* es:

```
IQR <- 31664 - 24392
IQR
```

```
## [1] 7272
```

Así según los cuartiles sobre la demanda, se define como valor atípico leve aquel que dista 1.5 veces el rango intercuantílico por debajo de $Q1$ o por encima de $Q3$

$$*\text{outlier}* < \text{Q1} - 1.5 * \text{IQR} \text{ o bien } *\text{outlier}* > \text{Q3} + 1.5 * \text{IQR}$$

y valor atípico extremo aquel que dista 3 veces el rango intercantílico por debajo de *Q1* o por encima de *Q3*

$$*outlier* < Q1 - 3 * IQR \text{ o bien } *outlier* > Q2 + 3 * IQR$$

De hecho, con esta información, se calculan los umbrales *highLimit* y *lowLimit*:

```
highLimit <- 24392 + 1.5 * IQR
highLimit
```

```
## [1] 35300
```

```
lowLimit <- 31664 - 1.5 * IQR
lowLimit
```

```
## [1] 20756
```

Vemos que todos los valores de la demanda que superen los 35300 MW se consideran *outliers*. Y todos los valores de la demanda que sean inferiores a los 20756 MW también son *outliers*. Una vez identificados los *outliers* de la demanda, se procede al reemplazo de estos. Para ello, se define una función propia que recibe como parámetros el conjunto de datos de la demanda, el umbral inferior y el umbral superior. En la función se decide reemplazar por la media aquellos valores atípicos que estén por debajo del umbral inferior, y por la mediana aquellos que estén por encima del umbral superior, procedimiento muy utilizado en el reemplazo de *outliers*. La función y su aplicación se muestran a continuación:

```
outliersReplace <- function(data, lowLimit, highLimit) {
  data[data < lowLimit] <- as.integer(mean(data))
  data[data > highLimit] <- as.integer(median(data))
  data
}

data$demand.WO <- outliersReplace(data$demand, lowLimit, highLimit)
```

Veamos dos ejemplos donde se ha realizado el reemplazo.

```
data[752, c("demand", "demand.WO")]
```

```
##     demand demand.WO
## 752  20243     28348
```

```
data[1666, c("demand", "demand.WO")]
```

```
##      demand demand.WO
## 1666  35315     28566
```

Finalmente, substituiremos los valores de la demanda originales por los valores de la demanda sin *outliers*.

```
data$demand <- data$demand.WO
summary(data$demand)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   20756   24863   28566   27954   30720   35300
```

Es normal que la media, o la mediana o los cuartiles cambien después del reemplazo, pues los *outliers* tenían bastante peso y distorsionaban los datos de la serie.

Capítulo 3

Aplicación del modelo ARIMA

3.1. Transformación de la serie

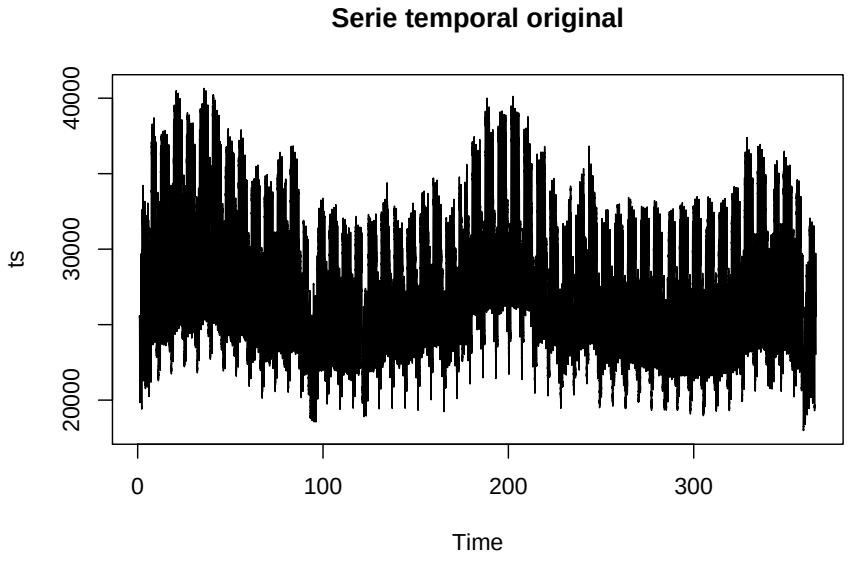
El análisis previo del primer capítulo nos ha revelado que la serie no es estacionaria por su estacionalidad, y como el modelo que vamos a utilizar es un modelo ARIMA, necesitamos que la serie sea estacionaria. Para ello, tendremos que eliminar la estacionalidad de la serie, diferenciando la serie lograremos que se convierta en estacionaria. Para empezar la transformación de la serie, primero se observa que resultado devuelve la función *ndiffs()*, que calcula el número de diferenciaciones estacionales que se necesitan llevar a cabo para que la serie sea estacionaria.

```
ndiffs(ts)
```

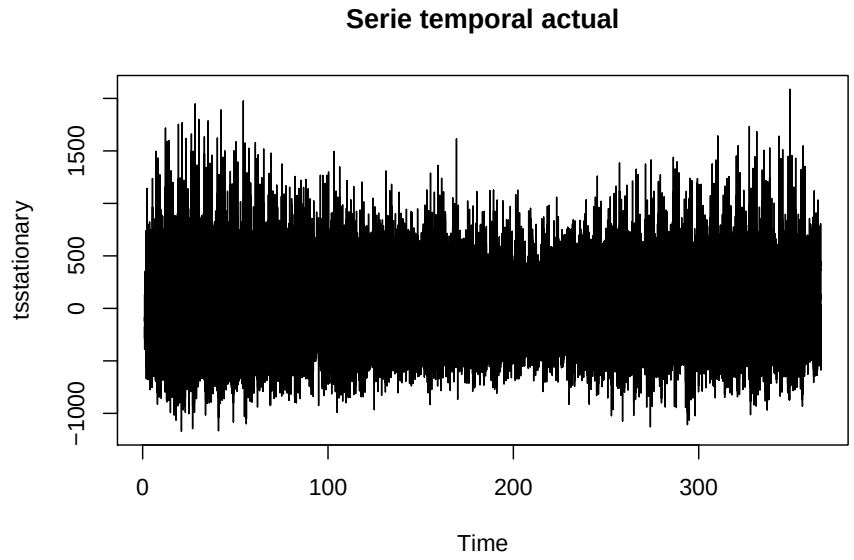
```
## [1] 1
```

El cálculo nos muestra que la serie necesita de una diferenciación estacional. Con la función *diff()* eliminaremos la estacionalidad. Compararemos el resultado gráficamente con la serie original.

```
tsstationary <- diff(ts, differences = 1)
plot(ts,  main = "Serie temporal original")
```

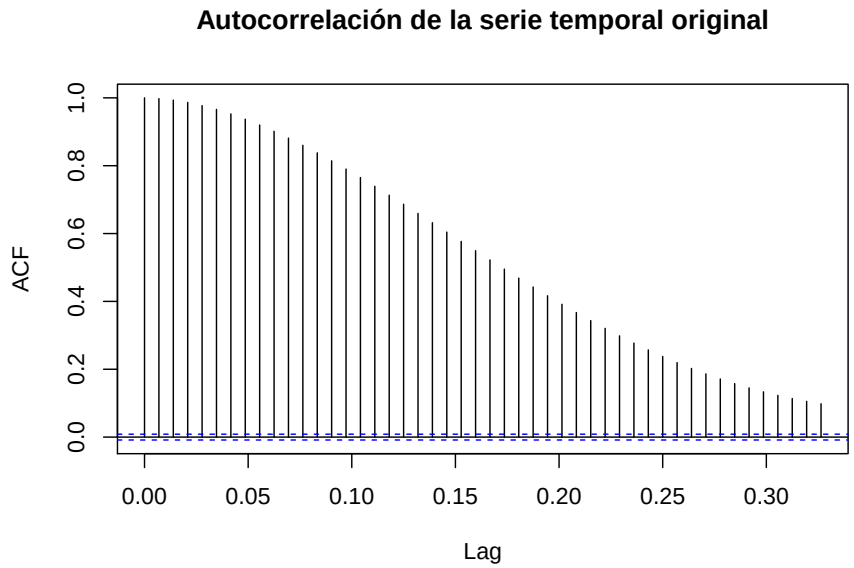


```
plot(tsstationary, main = "Serie temporal actual")
```

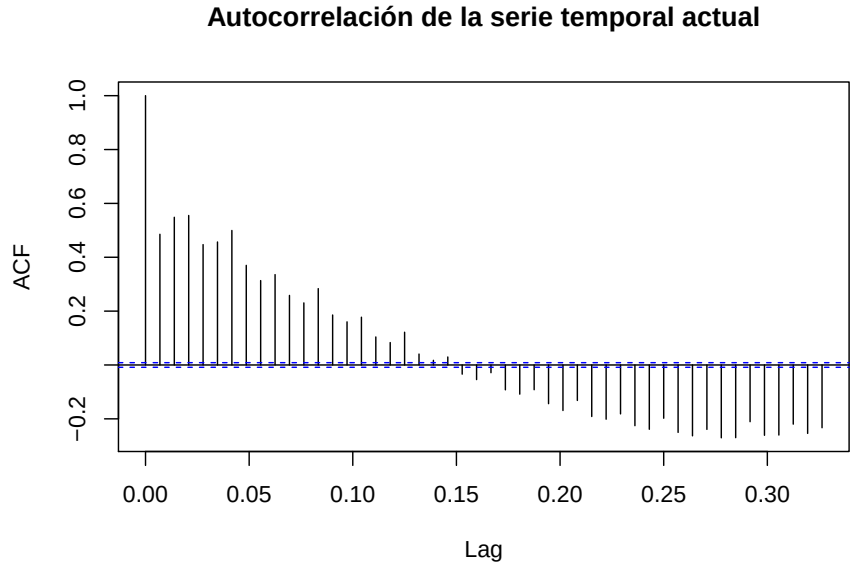


Como podemos ver se ha eliminado la componente de estacionalidad en la serie, también lo podemos ver en la función de autocorrelación.

```
acf(ts, main = "Autocorrelación de la serie temporal original")
```




```
acf(tsstationary, main = "Autocorrelación de la serie temporal actual")
```



Una vez eliminada la componente estacional, la serie se parece bastante a una serie estacionaria, ya que parece ser constante en media y varianza, pero para asegurarlo se aplica un test de estacionariedad como el test de Dickey-Fuller Aumentado (Augmented Dickey-Fuller Test (ADF), en inglés) y el test de Phillips–Perron (PP), una modificación de test de Dickey-Fuller.

En estos tests, se considera que la hipótesis nula es que la serie tiene raíces unitarias, por tanto, no es estacionaria. Al contrario, la serie es estacionaria. Con un *p-value* inferior a 0.05, la hipótesis nula se rechaza, confirmando que la serie es estacionaria. El motivo de basarse en la observación de raíces unitarias es porqué una raíz unitaria, es una tendencia estocástica en la serie temporal. Algunas veces se le llama *"paseo aleatorio con deriva"*. Por tanto, si la serie tiene una raíz unitaria, ésta presenta un patrón sistemático que es impredecible. Entonces, una serie temporal es estacionaria si un cambio en el tiempo no cambia la forma de la distribución; y las raíces unitarias son una causa de no estacionalidad.

Para aplicar los tests anteriores, se usa el paquete tseries.

```
# Cargamos el paquete tseries
library(tseries)
```

Se aplica el test de Dickey-Fuller Aumentado *adf.test*:

```
adf.test(tsstationary)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  tsstationary
## Dickey-Fuller = -48.343, Lag order = 37, p-value = 0.01
## alternative hypothesis: stationary
```

Como se puede observar, después de transformar la serie, ahora sí es estacionaria; el *p-value* de 0.01, indica que rechazamos la hipótesis nula de no estacionalidad.

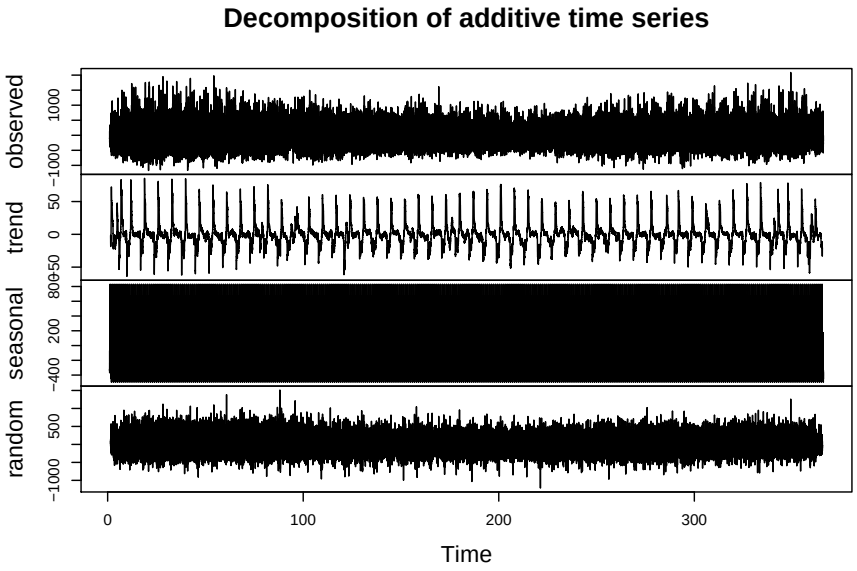
Ahora, se aplica el test de PP:

```
pp.test(tsstationary)
```

```
##
##  Phillips-Perron Unit Root Test
##
## data:  tsstationary
## Dickey-Fuller Z(alpha) = -65858, Truncation lag parameter = 19, p-value
## = 0.01
## alternative hypothesis: stationary
```

Con este test, también se obtiene un *p-value* de 0.01, por tanto, también rechazamos la hipótesis nula y podemos decir que la serie es estacionaria. Por último, descomponemos la serie temporal transformada:

```
componentes.tsstationary <- decompose(tsstationary)
plot(componentes.tsstationary)
```



Después de analizar los anteriores tests y observar la última figura, se puede afirmar que no se observa estacionalidad en la serie transformada, es estacionaria.

3.2. Predicción

Se aplica el método de predicción descrito en el capítulo anterior para obtener una predicción del consumo eléctrico con un horizonte temporal de 4 horas para los días de la semana desde el lunes 8 de junio hasta el domingo 14 de junio.

Bibliografía

Fernando Campos. Análisis de series temporales en r. arima. Disponible en https://www.diegocalvo.es/analisis-de-series-temporales-en-r-arima/, 2018.

Pedro L. Luque-Calvo. Escribir un trabajo fin de estudios con r markdown. Disponible en http://destio.us.es/calvo/post/escribir-un-trabajo-fin-de-estudios-con-rmarkdow/, 2017.

Juan Antonio Jiménez Torres. Detección y reemplazo de outliers con r. Disponible en https://www.adictosaltrabajo.com/2019/11/28/deteccion-y-reemplazo-de-outliers-con-r/, 2019.