Máster Ciencia de Datos:
Series Temporales y Minería de Flujo de Datos

Trabajo Autónomo I: Series Temporales

Antonio Manuel Milán Jiménez
email:antoniomj@correo.ugr.es

8 de Abril de 2019

Contents

Presentación del problema

En este problema se trabajará sobre los diferentes datos registrados por una estación meteorológica, concretamenta la estación ubicada en Lanjarón ,Granada (6258X). El objetivo será poder predecir la temperatura máxima de esta región para la primera semana de Marzo 2018 y los meses de Marzo y Abril 2018 utilizando los datos registrados por la estación desde Mayo de 2013 hasta Febrero de 2018.

Así, analizando el comportamiento de la temperatura máxima a lo largo de todo este periodo, se estudiará las diferentes componentes de la serie con el fin de poder realizar una predicción lo más realista posible.

Los datos utilizado se encuentran en el fichero '6258X.csv'. Para replicar este experimento únicamente habrá que ejecutar al completo los ficheros 'Estacion6258X_Diaria_77449165.R' y 'Estacion6258X_Mensual_77449165.R' donde se sitúe este fichero de datos.

Parte Teórica

Preprocesamiento

Valores perdidos

Dada la naturaleza del problema, es muy probable que existan valores perdidos en los datos, es decir, que por circunstancias externas no se pudiesen tomar ciertos valores y por lo tanto no aparezcan ahora en el conjunto de datos.

Para solventarlo se utilizará alguna técnica de **Imputación** considerando la naturaleza de un problema de series temporales, por lo que será más indicado que se establezca el valor perdido en función de valores conocidos de días cercanos al mes en el que se encuentre. De esta forma, el valor perdido que se recupere será lo más realista posible pues se considerará solamente registros cercanos al día más que tener en cuenta todo el conjunto.

Detección de anomalías

Similar a la idea anterior, puede suceder que por causas externas los sensores no realizasen una correcta medición en un momento dado y se tengan valores anómalos para ciertas observaciones. Si bien sería dificil detectar anomalías en las variables relacionadas con el viento o la lluvia dada la alta variación que puede darse entre dos días consecutivos, sí que es más intuitivo determinar que hay una anomalía en un registro de la temperatura en función del mes en el que se encuentra y de los registros para los días cercanos.

Con esta idea, calculando el rango intercuartil (IQR) de la variable para un mes concreto, se podrá determinar para dicho mes qué valores se consideran normales y cuáles se considerarían anómalos por sus valores extremos. Para aquellos registros anómalos dado el mes en el que se encuentran, se tratarán sustituyéndolos por la media de los días anteriores y posteriores, al igual que en el caso de los valores perdidos, para que sean lo más realistas posibles.

Además, en el caso de que se detectase una falsa anomalía, sería aun así aceptable establecer su registro de la temperatura en función de los días cercanos dada la naturaleza del problema.

Componentes de la serie temporal

Una serie temporal consta de las componentes: estacionalidad, tendencia y componente irregular. Es importante identificar cada una de ellas para poder eliminar la estacionalidad y la tendencia de la serie, facilitando así el modelado únicamente de la componente irregular para realizar la predicción.

Una vez realizada la predicción de la componente irregular, se sumarán la estacionalidad y la tendencia quitadas para conseguir la predicción final, pues se supone que esas dos componentes sí se mantienen a lo largo de la seríe y no hacía falta predecirlas.

Estacionalidad

Esta componente se refiere al comportamiento que se repite cada cierto tiempo en la serie. Por ejemplo, en el caso de este problema en el que se tratan aspectos atmosféricos, la temperatura está muy relacionada con esta componente pues su variación a lo largo del año es algo que debería repetirse en todos los años y seguramente esta componente sí este presente en el problema.

Una vez identificada, se eliminará en toda la serie y tras la predicción se volvera a incorporar.

Tendencia

Esta componente se refiere a que, por ejemplo, se presente un comportamiento ascendente o descendente a lo largo de toda la serie. También puede suceder que cambie la tendencia, es decir, que hasta la mitad de la serie sea ascendente y a continuación se convierta en una tendencia descendente.

Se realizarán hipótesis de funciones más o menos complejas (un grado de libertad, dos grados de libertad, etc.) para obtener aquella función más simple y que mejor se adapte a la componente si que haya un sobreajuste. Entonces, esta función será la que se elimine de la serie y se vuelva a sumar tras la predicción realizada.

Otro punto interesante sobre esta componente es estudiar su rango, es decir, saber en qué valores varía la componente para conocer cómo de influyente es en la serie y si realmente se debe considerar. Por ejemplo, es una serie de rango 0-40, una componente de tendencia que se "mueva" en 20-21, realmente no está influyendo en la serie sea cuál sea su comportamiento y se puede tomar como que no existe tendencia para facilitar la predicción pues no habrá diferencia.

Componente irregular

Esta componente es en sí la serie al haberle eliminado la estacionalidad y la tendencia. Como se ha comentado anteriormente, esta es la componente más interesante de poder modelar para realizar una predicción más adecuada y sencilla. Una vez modelada, añadiendo la estacionalidad y tendecia previamente quitadas, se tendrá la predicción final.

Estacionareidad

Para hacer un modelo predictivo sobre la componente irregular, ésta ha de ser estacionaria, con media y varianza en torno a 0. El test de Dickey Fuller aumentado puede determinar si la serie es estacionaria.

En el caso de no ser así, se puede convertir en estacionaria realizando algún tipo de transformación, por ejemplo, una transformación logarítmica sobre la serie que se deshacerá cuando vuelvan a añadirse la estacionalidad y la tendencia a la serie.

Modelado y predicción sobre la componente irregular

El modelo que se utilice en la predicción de la componente irregular puede ser Autorregresivo o de Medias moviles. Se utilizará un modelo ARIMA(p,d,q) en el que "p" es el grado de utilizar un modelo autorregresivo, "q" el grado de un modelo de medias móviles y "d" el número de diferenciaciones a hacer a la serie en el caso de que ésta no sea estacionaria.

Para determinar los parámetros del modelo ARIMA a utilizar, se estudian las gráficas ACF y PACF de la serie. El comportamiento de estas gráficas indicará que modelo será el más adecuado para la predicción:

Una gráfica ACF en la que la distribución presente un descenso a 0 de manera regular, sinusoidal o alternando entre positivo y negativo, será indicativo de que es preferible utilizar un modelo Autoregresivo cuyo grado será el número de valores por encima del umbral ("diferentes de 0") en la gráfica PACF.

De igual forma, una gráfica PACF que presente un descenso a 0 de manera regular, sinusoidal o alternando entre positivo y negativo, será indicativo de que es preferible utilizar un modelo de Medias móviles cuyo grado será el número de valores por encima del umbral ("diferentes de 0") en la gráfica ACF.

En el caso de que no sucedan ninguna de estas dos situaciones, se tendrán que predecir con ambos modelos y en función del error acumulado en la predicción determinar qué modelo es mejor para la serie.

Señalar que si la predicción que se obtenga de la componente irregular es una predicción constante, sin variaciones, se debe a que se ha detectado aleatoriedad y normalidad en los residuos, lo cual se traduce en que no se ha encontrado un comportamiento como tal en la componente irregular y simplementa con la estacionalidad y la tendencia se tendrá la predicción final.

Tests utilizados

Por último, se enumerarán los tests que se utilicen en el problema para guiar la predicción:

- **Test de Dickey Fuller aumentado** : Test utilizado para determinar si la serie es estacionaria.
- **Test de Jarque Bera** : Test utilizado para determinar la normalidad de los datos.
- **Test de Student** : Test utilizado para determinar si la hipótesis realizada sobre la tendencia es acertada en unos datos normalizados.
- **Test de Box-Pierce** : Test utilizado para determinar si existe aleatoriedad en los residuos del modelo ajustado.
- **Test de Shapiro-Wilk** : Test utilizado para determinar si existe normalidad en los residuos del modelo ajustado.

Parte Práctica

Estudio inicial de los datos

Se cargan los datos referentes a la estación meteorológica de Lanjarón, Granada.

```
setwd('/home/antonio/Documentos/SeriesTemporalesFlujosDatos/PracticasSeriesTemporales/datosParteAutonoma
datos <- read.csv("6258X.csv", sep=";")
```

Estas son las dimensiones del conjunto de datos:

```
dim(datos)
```

```
## [1] 1754   16
```

```
colnames(datos)
```

```
##  [1] "Id"     "Fecha"  "Tmax"   "HTmax"  "Tmin"   "HTmin"  "Tmed"
##  [8] "Racha"  "HRacha" "Vmax"   "HVmax"  "TPrec"  "Prec1"  "Prec2"
## [15] "Prec3"  "Prec4"
```

Se tiene un total de 1754 instancias (días) y 16 variables. Se muestra una descripción más detallada de los datos:

```
Hmisc::describe(datos)
```

```
## datos
##
##  16  Variables      1754  Observations
## ---------------------------------------------------------------------------
## Id
##        n  missing distinct    value
##     1754        0        1    6258X
##
## Value      6258X
## Frequency   1754
## Proportion     1
## ---------------------------------------------------------------------------
## Fecha
##        n  missing distinct
##     1754        0     1754
##
## lowest : 2013-05-07 2013-05-08 2013-05-09 2013-05-10 2013-05-11
## highest: 2018-02-24 2018-02-25 2018-02-26 2018-02-27 2018-02-28
## ---------------------------------------------------------------------------
## Tmax
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1647      107      280        1    21.55    7.428    11.53    13.10
##      .25      .50      .75      .90      .95
##    16.20    21.60    26.60    29.80    31.80
##
## lowest :  5.7  6.0  6.6  6.8  7.0, highest: 36.7 36.9 37.0 37.7 39.7
## ---------------------------------------------------------------------------
## HTmax
##        n  missing distinct
##     1754        0       79
##
```

```
## lowest :       00:00 00:10 00:30 00:40, highest: 20:40 22:30 23:10 23:30 23:40
## ---------------------------------------------------------------------------
## Tmin
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1647      107      226        1     9.94    6.146     1.23     2.66
##      .25      .50      .75      .90      .95
##     5.40    10.30    14.30    16.90    18.00
##
## lowest : -3.2 -2.1 -1.8 -1.7 -1.6, highest: 21.1 21.2 21.3 21.5 22.6
## ---------------------------------------------------------------------------
## HTmin
##        n  missing distinct
##     1754        0       98
##
## lowest :       00:00 00:10 00:20 00:30, highest: 23:20 23:30 23:40 23:50 23:59
## ---------------------------------------------------------------------------
## Tmed
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1647      107      241        1    15.75    6.626      6.9      8.1
##      .25      .50      .75      .90      .95
##     10.8     15.9     20.5     23.2     24.6
##
## lowest :  1.8  2.9  3.0  3.1  3.4, highest: 28.0 28.2 28.4 28.5 30.5
## ---------------------------------------------------------------------------
## TPrec
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1617      137       81    0.436   0.8294    1.568     0.00     0.00
##      .25      .50      .75      .90      .95
##     0.00     0.00     0.00     1.00     4.84
##
## lowest :  0.0  0.2  0.4  0.6  0.8, highest: 26.8 27.2 34.6 48.0 78.2
## ---------------------------------------------------------------------------
## Prec1
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1702       52       43    0.221    0.211   0.4092      0.0      0.0
##      .25      .50      .75      .90      .95
##      0.0      0.0      0.0      0.0      0.6
##
## lowest :  0.0  0.2  0.4  0.6  0.8, highest: 10.0 10.2 10.4 12.4 23.0
## ---------------------------------------------------------------------------
## Prec2
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1677       77       46    0.255   0.2937   0.5745      0.0      0.0
##      .25      .50      .75      .90      .95
##      0.0      0.0      0.0      0.0      0.4
##
## lowest :  0.0  0.2  0.4  0.6  0.8, highest: 19.2 21.2 28.6 40.6 70.4
## ---------------------------------------------------------------------------
## Prec3
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1707       47       40    0.205   0.1861   0.3608      0.0      0.0
##      .25      .50      .75      .90      .95
##      0.0      0.0      0.0      0.0      0.4
##
```

```
## lowest :  0.0  0.2  0.4  0.6  0.8, highest: 10.4 11.0 12.0 12.6 16.6
## ---------------------------------------------------------------------------
## Prec4
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1675       79       38    0.208   0.1939    0.378      0.0      0.0
##      .25      .50      .75      .90      .95
##      0.0      0.0      0.0      0.0      0.4
##
## lowest :  0.0  0.2  0.4  0.6  0.8, highest: 16.2 19.0 19.2 21.6 21.8
## ---------------------------------------------------------------------------
##
## Variables with all observations missing:
##
## [1] Racha  HRacha Vmax   HVmax
```

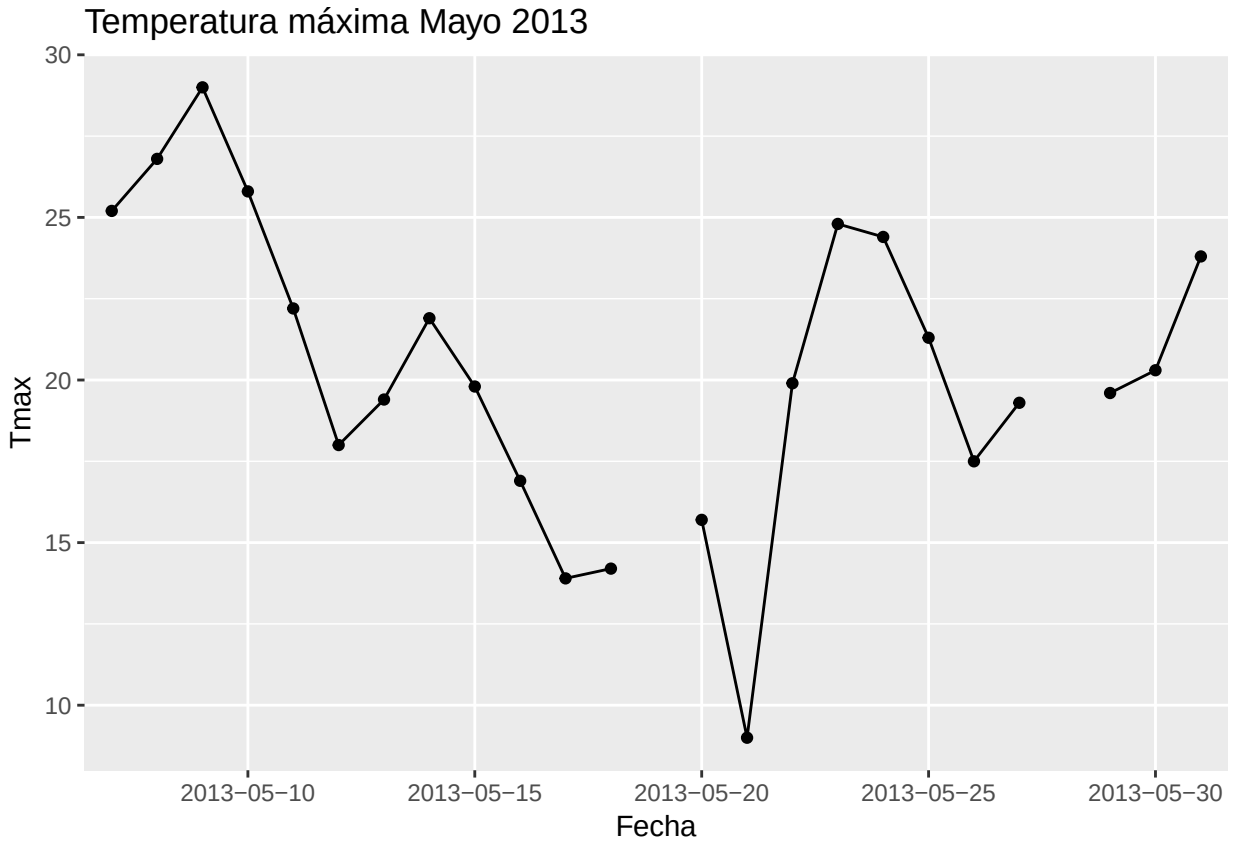
El hecho más destacable que se descubre es que no hay ningún valor para las variables "Racha", "HRacha", "Vmax", "HVmax". En los 1754 días de observaciones no se ha registrado nada respecto al viento, quizás porque no posea esta estación meteorológica los instrumentos de medición para ello. En cualquier caso, directamente se tendrán que eliminar estas variables que no aportan ninguna información.

Las dos primeras columnas son el código de identificación de la estación y el día de cada observación. Respecto al "Id" parece claro que haya que eliminarlo de los datos pues no aporta información útil para la construcción del modelo. Sucede algo parecido con la variable "Fecha"; si bien resulta muy útil saber la fecha para entender los datos de un problema meteorológico, no es una variable que vaya a ayudar a descubrir nuevos comportamientos interesantes e incluso puede llegar a ser contraproducente en el aprendizaje del modelo.

También se encuentra que para las diferentes variables predictoras existen varios valores perdidos que por alguna razón no se tomaron y que deberán ser tratados. Por ejemplo, podría tenerse en cuenta los valores conocidos para ese mes y sustituir el valor perdido por la media de ellos o por el dia anterior y posterior para tener una estimación lo más realista posible.

Ya respecto al contenido de las variables, se puede observar que todos los valores están dentro de unos rangos lógicos y aceptables. Sin embargo, sería interesante que las variables relacionadas con la temperatura se estudiasen detenidamente para cada estación del año pues, si bien una temperatura de 40ºC es lógica para los días de verano, no lo sería para un día de invierno y seguramente se tratase de un valor mal tomado (una anomalía) que tendría que ser tratado. Por ejemplo, para el mes de mayo de 2013:

```
ggplot(datos[1:25,],aes(x=Fecha, y=Tmax, group=1)) + geom_point() + geom_line() +
  scale_x_discrete(breaks=c("2013-05-05","2013-05-10","2013-05-15","2013-05-20",
  "2013-05-25","2013-05-30")) +labs(title="Temperatura máxima Mayo 2013")
```



La gráfica muestra en general unos valores normales para la temperatura máxima alcanzada en mayo (las discontinuidades se deben a los valores perdidos) excepto por el día "21 de mayo" que presenta una temperatura máxima por debajo de 10ºC, algo con menos sentido. Estudiando la hora en la que se registró esta temperatura:

```
datos[15,"HTmax"]
```

```
## [1] 00:10
## 79 Levels:  00:00 00:10 00:30 00:40 00:50 01:30 01:50 02:10 ... 23:40
```

```
datos[15,"HTmin"]
```

```
## [1] 05:20
## 98 Levels:  00:00 00:10 00:20 00:30 00:40 00:50 01:00 01:10 ... 23:59
```

```
datos[15,"Tmin"]
```

```
## [1] 7.4
```

Descubriendo que la temperatura máxima se registrase a las 00:10, parece indicar que hubiese algún fallo en la medición de temperatura ese día, que no se pudiese medir a lo largo del día, y que se tomase como temperatura máxima la más alta de las registradas que en este caso sería 9ºC a las 00:10.

Para automatizar este proceso de detección de anomalías se puede utilizar el rango intercuartil empleando la siguiente función utilizada en la asignatura del máster "Aprendizaje no supervisado y Detección de Anomalías":

```r
#Funciones encargadas de detectar outliers mediante el rango intercuartil con un coeficiente
#dado para una variable concreta
vector_es_outlier_IQR = function (datos, indice.de.columna, coef = 1.5){
  columna.datos = datos[,indice.de.columna]
  #Se obtienen cuartiles primero y tercero
  cuartil.primero = quantile(columna.datos,na.rm=TRUE)[2]
  cuartil.tercero = quantile(columna.datos,na.rm=TRUE)[4]
  #Rango intercuartil
  iqr = cuartil.tercero - cuartil.primero
  #Se definen los umbrales en función del rango intercuartil y el coeficient
  extremo.superior.outlier = (iqr * coef) + cuartil.tercero
  extremo.inferior.outlier = cuartil.primero - (iqr * coef)
  #Se determinan como outliers aquellos que superen algun umbral, superior o inferior
  es.outlier  = columna.datos > extremo.superior.outlier |
    columna.datos < extremo.inferior.outlier
  return (es.outlier)
}

vector_claves_outliers_IQR = function(datos, indice, coef = 1.5){
  columna.datos = datos[,indice]
  vector.de.outliers = vector_es_outlier_IQR(datos, indice, coef)
  #Se devuelven los indices de aquellos datos detectados como outliers
  return (which(vector.de.outliers  == TRUE))
}
```

Calculando la diferencia entre los cuartiles primero y tercero de los datos de la variable en cuestión y estableciendo un coeficiente sobre cómo de anómalo ha de ser el dato, se estiman las instancias que son anomalías para la variable que quiera. Probando su funcionamiento:

```r
outliers = vector_claves_outliers_IQR(datos = datos[1:25,], "Tmax",coef=1.3)
datos[outliers,2]
```

```
## [1] 2013-05-21
## 1754 Levels: 2013-05-07 2013-05-08 2013-05-09 2013-05-10 ... 2018-02-28
```

Efectivamente, el día 21 de mayo se ha detectado como anomalía para esta variable. Entonces, se podría cambiar ahora ese valor por la media de las temperaturas máximas del día anterior y posterior. Se eliminaría así la anomalía y en el caso de que no lo hubiese sido en un primer momento, no sería tan relevante pues estimar la temperatura de un día en función de sus días cercanos es un cálculo aceptable para el problema que se presenta.

Este proceso se automatizaría y se realizará para todos los meses de los datos para aquellas variables relacionadas con la temperatura. La razón por la que no se hará con el resto de variables es que fenómenos atmosféricos como la lluvia o el viento son mucho más variantes y no tan dependientes del mes, por lo que aquí sí que podrían detectarse demasiada anomalías que realmente no lo fuesen.

Como último apunte sobre este estudio inicial de los datos es que, dado que la predicción se realizará sobre los meses de Marzo y Abril de 2018 y no se tiene registros sobre ellos, sería una idea interesante utilizar el último año registrado como datos de validación y saber así como de bien o de mal se está realizando la predicción.

Preprocesamiento

Para empezar se agruparán los datos en función del mes y del año, para facilitar la detección de anomalías y la imputación de los valores perdidos. De este modo, se crea una columna auxiliar para facilitar el filtrado según el mes:

```
#Función encargada de devolver el año y el mes de la fecha
obtenerMes <- function(fecha){
  substr(toString(fecha),1,7)
}

#Se construye una columna auxiliar para hacer la agrupación que será eliminada posteriormente
listaAux <- unlist(list(datos[,2]))
nuevaColumna <- unlist(lapply(listaAux,obtenerMes))
datos$FechaSinDia = nuevaColumna
```

Una vez hecho esto, sobre cada grupo se detectarán y corregirán las anomalías mediante el rango intercuartil ya explicado para las variables "Tmax", "Tmin" y "Tmed":

```
#Se obtienen todos los meses en los que se han registrado los datos
meses = unique(datos$FechaSinDia)

#Función encargada de detectar y corregir las anomalías para un mes dado
corregirAnomalias <- function(mes){

  #Función encargada de corregir una anomalía
  eliminarAnomalias <- function(outlier,columna,miGrupo){

    ind1 = outlier
    ind2 = outlier

    #Se comprueba si el valor anterior existe
    if(outlier > 1){
      if(is.na(miGrupo[(outlier-1),columna]) == FALSE){
        ind1 = outlier-1
      }
    }
    #Se comprueba que existe el valor posterior
    if(outlier < length(miGrupo[,1])){
      if(is.na(miGrupo[(outlier+1),columna]) == FALSE){
        ind2 = outlier+1
      }
    }
    #Se obtiene la media de ambos valores
    valorMedio <- mean(c(miGrupo[ind1,columna],miGrupo[ind2,columna]))
    miGrupo[outlier,columna] <- valorMedio

    #Se corrige tambien la hora del registro en función del registro anterior
    if(columna=="Tmax"){
      miGrupo[outlier,"HTmax"] <- miGrupo[ind1,"HTmax"]
    }else if(columna=="Tmin"){
      miGrupo[outlier,"HTmin"] <- miGrupo[ind1,"HTmin"]
    }
    miGrupo
  }
```

```
  #Se obtienen los datos para un mes concreto
  grupo <- datos %>% filter(FechaSinDia == mes)

  #Se detectan y corrigen los outliers para la variable Tmax
  outliers = vector_claves_outliers_IQR(datos = grupo, "Tmax",coef=1.5)
  outTmax = length(outliers)
  for (outlier in outliers){
    grupo = eliminarAnomalias(outlier,"Tmax",grupo)
  }

  #Se detectan y corrigen los outliers para la variable Tmin
  outliers = vector_claves_outliers_IQR(datos = grupo, "Tmin",coef=1.5)
  outTmin = length(outliers)
  for (outlier in outliers){
    grupo = eliminarAnomalias(outlier,"Tmin",grupo)
  }

  #Se detectan y corrigen los outliers para la variable Tmed
  outliers = vector_claves_outliers_IQR(datos = grupo, "Tmed",coef=1.5)
  outTmed = length(outliers)
  for (outlier in outliers){
    grupo = eliminarAnomalias(outlier,"Tmed",grupo)
  }

  cat("Tratados", outTmax+outTmin+outTmed,"outliers para el mes",mes,"\n")
  grupo
}

#Se aplica la correción de anomalías para todos los meses de los datos
listaGrupos = lapply(meses,corregirAnomalias)
```

```
## Tratados 0 outliers para el mes 2013-05
## Tratados 0 outliers para el mes 2013-06
## Tratados 5 outliers para el mes 2013-07
## Tratados 4 outliers para el mes 2013-08
## Tratados 11 outliers para el mes 2013-09
## Tratados 10 outliers para el mes 2013-10
## Tratados 0 outliers para el mes 2013-11
## Tratados 0 outliers para el mes 2013-12
## Tratados 3 outliers para el mes 2014-01
## Tratados 0 outliers para el mes 2014-02
## Tratados 0 outliers para el mes 2014-03
## Tratados 1 outliers para el mes 2014-04
## Tratados 2 outliers para el mes 2014-05
## Tratados 3 outliers para el mes 2014-06
## Tratados 1 outliers para el mes 2014-07
## Tratados 0 outliers para el mes 2014-08
## Tratados 1 outliers para el mes 2014-09
## Tratados 1 outliers para el mes 2014-10
## Tratados 1 outliers para el mes 2014-11
## Tratados 6 outliers para el mes 2014-12
## Tratados 2 outliers para el mes 2015-01
## Tratados 1 outliers para el mes 2015-02
```

```
## Tratados 0 outliers para el mes 2015-03
## Tratados 11 outliers para el mes 2015-04
## Tratados 2 outliers para el mes 2015-05
## Tratados 4 outliers para el mes 2015-06
## Tratados 0 outliers para el mes 2015-07
## Tratados 2 outliers para el mes 2015-08
## Tratados 0 outliers para el mes 2015-09
## Tratados 5 outliers para el mes 2015-10
## Tratados 3 outliers para el mes 2015-11
## Tratados 2 outliers para el mes 2015-12
## Tratados 0 outliers para el mes 2016-01
## Tratados 2 outliers para el mes 2016-02
## Tratados 0 outliers para el mes 2016-03
## Tratados 0 outliers para el mes 2016-04
## Tratados 0 outliers para el mes 2016-05
## Tratados 0 outliers para el mes 2016-06
## Tratados 2 outliers para el mes 2016-07
## Tratados 0 outliers para el mes 2016-08
## Tratados 4 outliers para el mes 2016-09
## Tratados 1 outliers para el mes 2016-10
## Tratados 7 outliers para el mes 2016-11
## Tratados 1 outliers para el mes 2016-12
## Tratados 2 outliers para el mes 2017-01
## Tratados 2 outliers para el mes 2017-02
## Tratados 4 outliers para el mes 2017-03
## Tratados 2 outliers para el mes 2017-04
## Tratados 1 outliers para el mes 2017-05
## Tratados 0 outliers para el mes 2017-06
## Tratados 0 outliers para el mes 2017-07
## Tratados 2 outliers para el mes 2017-08
## Tratados 1 outliers para el mes 2017-09
## Tratados 3 outliers para el mes 2017-10
## Tratados 0 outliers para el mes 2017-11
## Tratados 2 outliers para el mes 2017-12
## Tratados 0 outliers para el mes 2018-01
## Tratados 0 outliers para el mes 2018-02
```

Con las anomalías corregidas, se procede a realizar la imputación de los valores perdidos de la misma forma, teniendo en cuenta el registro anterior y posterior:

```r
#Función encargada de realizar la imputación para un mes dado
imputacion <- function(grupo){

  #Función encargada de imputar un valor perdido
  imputarValor <- function(valorPerdido,columna, miGrupo){
    ind1 = valorPerdido
    ind2 = valorPerdido

    #Se comprueba si el valor anterior existe
    if(valorPerdido > 1){
      if(is.na(miGrupo[(valorPerdido-1),columna]) == FALSE){
        ind1 = valorPerdido-1
      }
    }
```

```
  #Se comprueba que existe el valor posterior
  if(valorPerdido < length(miGrupo[,1])){
    if(is.na(miGrupo[(valorPerdido+1),columna]) == FALSE){
      ind2 = valorPerdido+1
    }
  }

  #Se realiza la media de los valores anterior y posterior si no son valores perdidos también
  if(is.na(miGrupo[ind1,columna]) == FALSE & is.na(miGrupo[ind2,columna]) == FALSE){
    valorMedio <- mean(c(miGrupo[ind1,columna],miGrupo[ind2,columna]))
    #Se corrige también la columna de la hora registrada
    if(columna==3){
      miGrupo[valorPerdido,"HTmax"] <- miGrupo[ind1,"HTmax"]
    }else if(columna==5){
      miGrupo[valorPerdido,"HTmin"] <- miGrupo[ind1,"HTmin"]
    }
    #Se toma el valor anterior si el posterior es un valor perdido
  }else if(is.na(miGrupo[ind1,columna]) == FALSE){
    valorMedio <- miGrupo[ind1,columna]

    if(columna==3){
      miGrupo[valorPerdido,"HTmax"] <- miGrupo[ind1,"HTmax"]
    }else if(columna==5){
      miGrupo[valorPerdido,"HTmin"] <- miGrupo[ind1,"HTmin"]
    }
    #Se toma el valos posterior si el anterior es un valor perdido
  }else if(is.na(miGrupo[ind2,columna]) == FALSE){
    valorMedio <- miGrupo[ind2,columna]

    if(columna==3){
      miGrupo[valorPerdido,"HTmax"] <- miGrupo[ind2,"HTmax"]
    }else if(columna==5){
      miGrupo[valorPerdido,"HTmin"] <- miGrupo[ind2,"HTmin"]
    }

  }else{  #Ambos valores son indeterminados, se toma la media de toda la columna de este mes
    valorMedio <- mean(miGrupo[,columna],na.rm = TRUE)
    #En caso de no tener registro anterior o posterior se establece una hora por defecto
    if(columna==3){
      miGrupo[valorPerdido,"HTmax"] <- '15:00'
    }else if(columna==5){
      miGrupo[valorPerdido,"HTmin"] <- '05:00'
    }

  }

  #Se establece el valor corregido
  miGrupo[valorPerdido,columna] <- valorMedio
  miGrupo
}
```

```
  #La imputación se realiza para las variables "Tmax" "Tmin" "Tmed" "TPrec" "Prec1" "Prec2"
  #"Prec3" "Prec4"
  for (indice in c(3,5,7,12,13,14,15,16)){
    valoresPerdidos = which(is.na(grupo[,indice])==TRUE)
    for (valor in valoresPerdidos){
      grupo <- imputarValor(valor,indice,grupo)
    }
  }

  #Se devuelve el grupo sin valores perdidos
  grupo
}

#Se realiza la imputación de valores perdidos para cada mes
listaGrupos <- lapply(listaGrupos,imputacion)
```

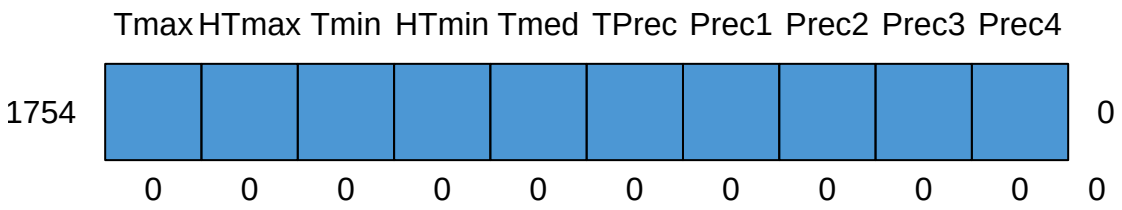
Una vez que todas las agrupaciones por meses están corregidas, se reagrupan en un mismo conjunto:

```
datos <- do.call("rbind", listaGrupos)
```

Teniendo así de nuevo el conjunto de datos sin anomalías y sin valores perdidos:

```
patron <- mice::md.pattern(x=datos[,-c(1,2,8,9,10,11,17)],plot = TRUE)
```

```
##  /\     /\
## {  `---'  }
## {  0   0  }
## ==>  V <==  No need for mice. This data set is completely observed.
##  \  \|/  /
##   `-----'
```



Ya solo queda eliminar las variables “Racha”, “HRacha”, “Vmax”, “HVmax”, “Id” y “Fecha” pues no serán de más utilizada por lo comentado en la sección anterior. Se convervará de momento la variable creada “FechaSinDia” pues servirá cuando se quiera agrupar nuevamente los datos en función del mes de su registro:

```
datos <- datos[,-c(1,2,8,9,10,11)]
colnames(datos)
```

```
##  [1] "Tmax"        "HTmax"       "Tmin"        "HTmin"       "Tmed"
##  [6] "TPrec"       "Prec1"       "Prec2"       "Prec3"       "Prec4"
## [11] "FechaSinDia"
```

Con este conjunto de datos final, sin anomalías ni valores perdidos, se construyen los datos en escalas diaria y mensual pues, dado que se quiere realizar una predicción sobre los meses de Marzo y Abril de 2018, y otra predicción para la primera semana de Marzo 2018, lo ideal es que se creen dos series diferentes, a escala diaria y a escala mensual (mediante la media de todos los días):

```r
#Datos en escala diaria
datosEscalaDiaria <- datos

meses = unique(datos$FechaSinDia)
#Función encargada de obtener los valores medios de las variables para un mes dado
mediaMes <-function(mes,datos){

  grupo <- datos %>% filter(FechaSinDia == mes)
  #Se calcula la media para Tmax, Tmin y Tmed
  mesCompleto <- colMeans(grupo[,-c(2,4,11)])
  #Añadimos valores por defecto para las variables restantes
  mesCompleto$FechaSinDia <- mes
  mesCompleto$HTmax <- "15:00"
  mesCompleto$HTmin <- "04:00"
  unlist(mesCompleto)
}

#Se obtiene la media de las variables para cada mes
datosEscalaMensual <- lapply(meses,mediaMes,datos)

#Se construyen los datos en escala mensual a partir de las medias calculadas
datosEscalaMensual <- data.frame(matrix(unlist(datosEscalaMensual),nrow=58,byrow=TRUE))
colnames(datosEscalaMensual) <- c("Tmax","Tmin","Tmed","TPrec","Prec1","Prec2","Prec3","Prec4",
                                  "FechaSinDia","HTmax","HTmin")
cat("Dimensiones de datos en escala mensual:",dim(datosEscalaMensual))
```

```
## Dimensiones de datos en escala mensual: 58 11
```

Así se tendría ya los dos conjuntos de datos.

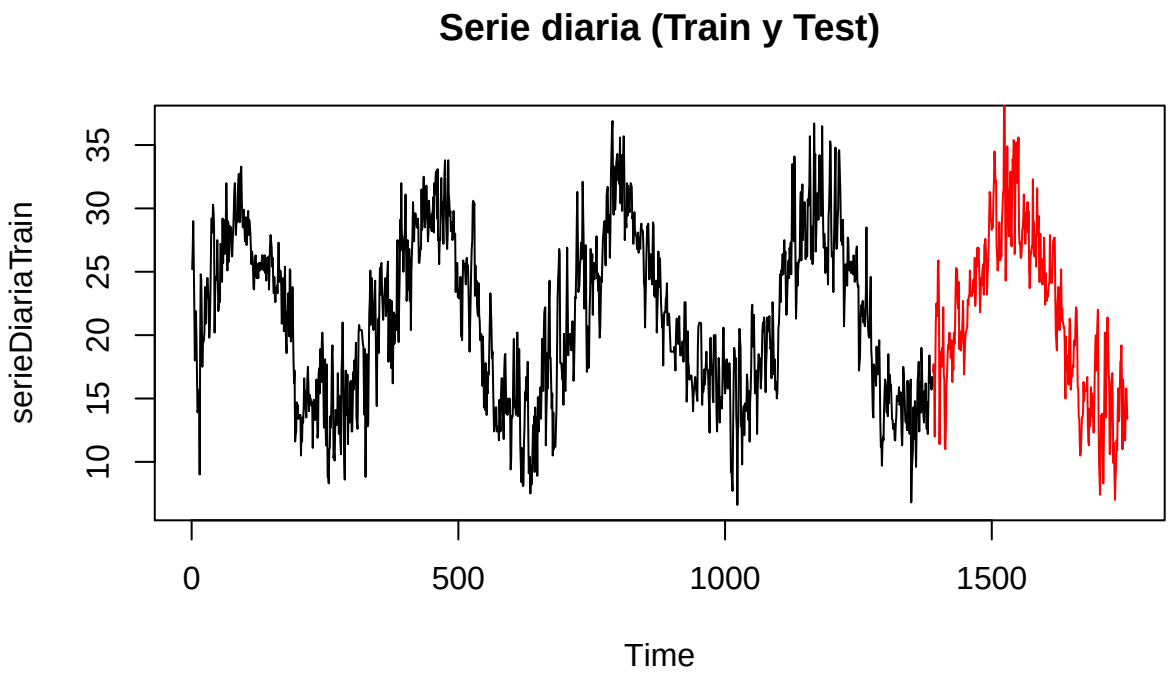
Serie temporal diaria

Como se comentó con anterioridad, se dividiran los datos en Train y Test (los últimos 365 días) para saber cómo de bien o mal se está realizando la predicción pues no se poseen los datos referentes al mes de Marzo 2018. Además, se seleccionan ya los datos correspondientes a la temperatura máxima:

```r
serieDiaria <- datosEscalaDiaria[,"Tmax"]

datosEscalaDiariaTrain <- datosEscalaDiaria[1:1389,]
datosEscalaDiariaTest <- datosEscalaDiaria[1390:1754,]

serieDiariaTrain <- datosEscalaDiariaTrain[,"Tmax"]
serieDiariaTest <- datosEscalaDiariaTest[,"Tmax"]
```

Visualizando la separación realizada en train y test:

```r
tiempoTrainDiaria <- 1:length(serieDiariaTrain)
tiempoTestDiaria <- (tiempoTrainDiaria[length(tiempoTrainDiaria)]+1):(tiempoTrainDiaria[
  length(tiempoTrainDiaria)]+length(serieDiariaTest))


#Se visualiza en rojo los datos de validación
plot.ts(serieDiariaTrain, xlim=c(1,tiempoTestDiaria[length(tiempoTestDiaria)]),
        main="Serie diaria (Train y Test)")
lines(tiempoTestDiaria, serieDiariaTest, col="red")
```

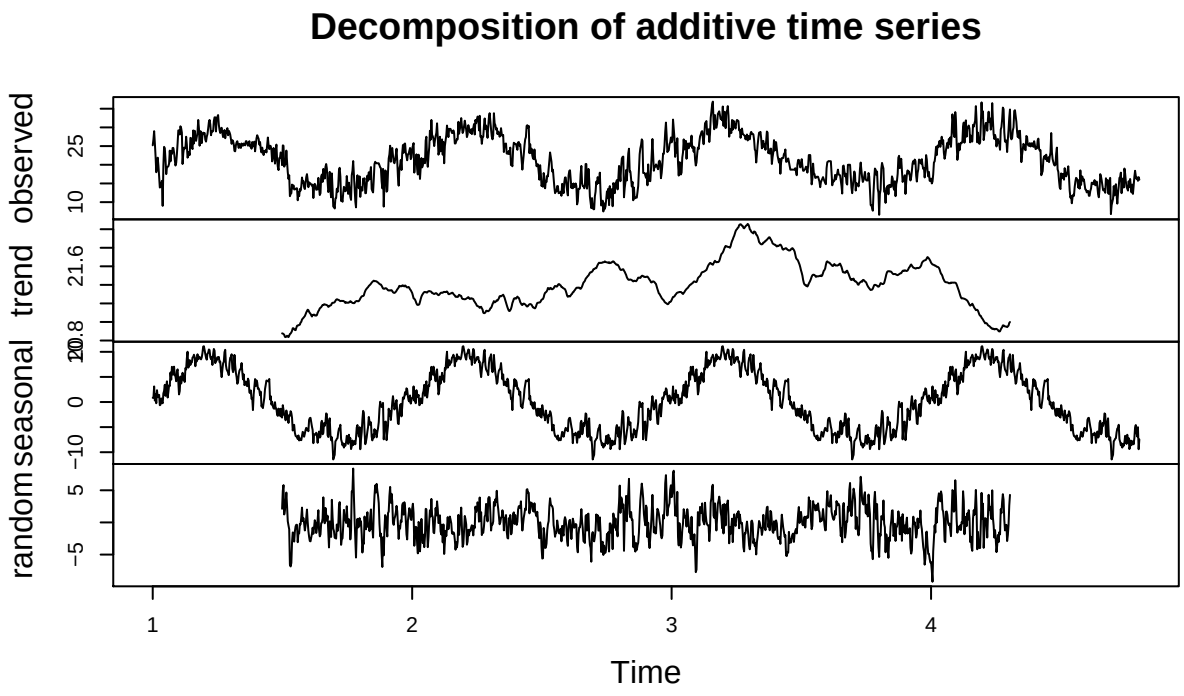


Ya organizados los datos como se requería, se construyen las series temporales correspondientes con los datos de temperatura máxima pues sobre esta variable se realizará la predicción:

```
library(tseries)
#Se establece una frecuencia de 365 correspondiente a los días del año
serieDiariaTrain.ts <- ts(serieDiariaTrain, frequency = 365)
```

Se visualizan las componentes para la serie:

```
plot(decompose(serieDiariaTrain.ts))
```



Observando sus componentes, respecto a la tendencia, es dificil establecer que haya alguna pues, si bien se observa que en la primera parte se tiene una tendencia ascendente que se convierte en una tendencia descente hasta aproximadamente el último año, el rango de esta gráfica de la tendencia se situa entre 20.8 y 22, mientras que el rango de la serie está entre 5 y 40, lo cual indica que apenas es considerable esa tendencia en la serie. No obstante, se intentará ajustar algún modelo más complejo a esta "tendencia".

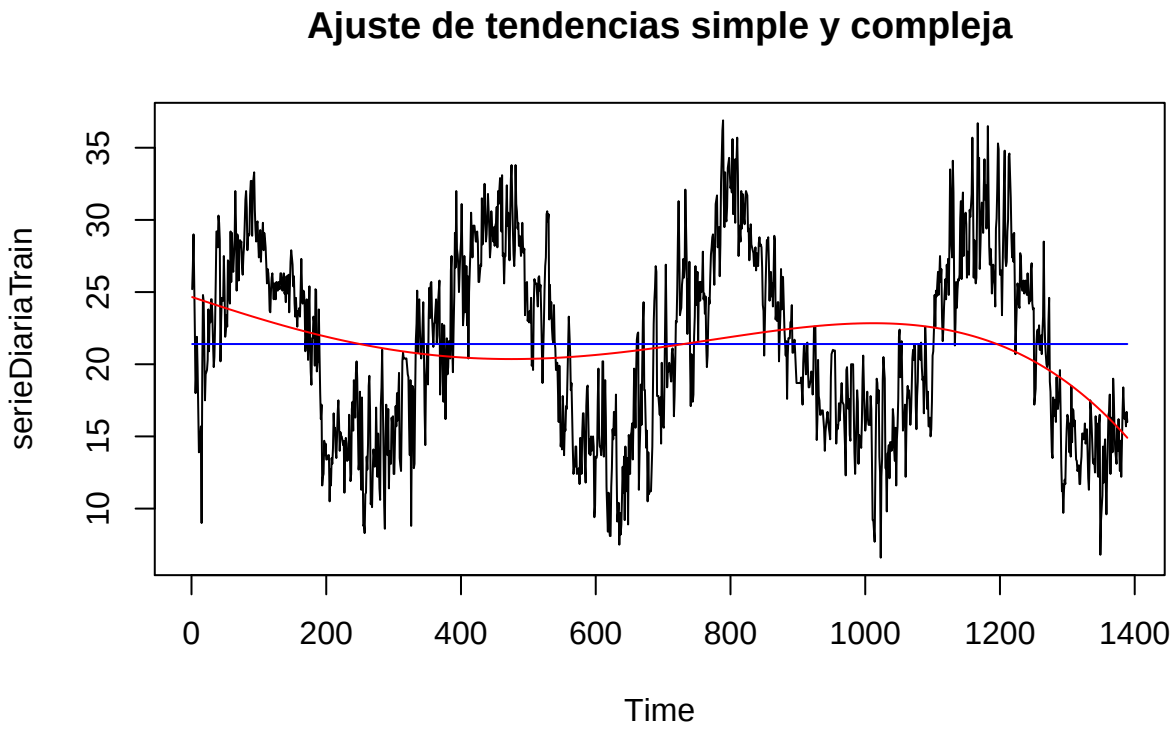
Se observa una clara estacionalidad y se intuye ya que es estacionaria pues a lo largo de toda la serie, la componente irregular presenta una media y una varianza en torno a 0.

Estudio de las componentes de la serie temporal diaria

Tratamiento de la tendencia

Dado el comportamiento descubierto por parte de la componente de la tendencia y su pequeño rango de influencia, se puede determinar que no exista una tendencia como tál en la serie. Aunque se puede intentar algún ajuste más complejo sobre la tendencia respecto a su máximo y mínimo, su comportamiento final en la serie será prácticamente el mismo que una tendencia constante (y = 21.4) dado el rango de la tendencia:

```
#Se ajusta la tendencia con un ajuste más complejo
parametros.H1 <- lm(serieDiariaTrain ~ tiempoTrainDiaria+I(tiempoTrainDiaria^2)+
                     I(tiempoTrainDiaria^3)+I(tiempoTrainDiaria^4))
#Se obtiene la tendencia compleja
TendEstimadaTrComplex.H1 <- parametros.H1$coefficients[1]+tiempoTrainDiaria*
  parametros.H1$coefficients[2]+tiempoTrainDiaria^2*parametros.H1$coefficients[3]+
  tiempoTrainDiaria^3*parametros.H1$coefficients[4]+tiempoTrainDiaria^4*
  parametros.H1$coefficients[5]
#Tendencia simple
TendEstimadaTr.H1 <- 21.4
#Visualización de ambas tendencias
plot.ts(serieDiariaTrain,main="Ajuste de tendencias simple y compleja")
lines(tiempoTrainDiaria, rep(TendEstimadaTr.H1,length(tiempoTrainDiaria)), col = "blue")
lines(tiempoTrainDiaria, TendEstimadaTrComplex.H1, col = "red")
```



Como se podía preveer, se visualizan ambos ajustes prácticamente iguales respecto a su relevancia en la serie (un rango no relevante). Según el principio de la navaja de Ockham, se tomaría el modelo más simple, en este caso la constante y=21.4 que equivale a determinar que no hay tendencia en la serie.

Tratamiento de la estacionalidad

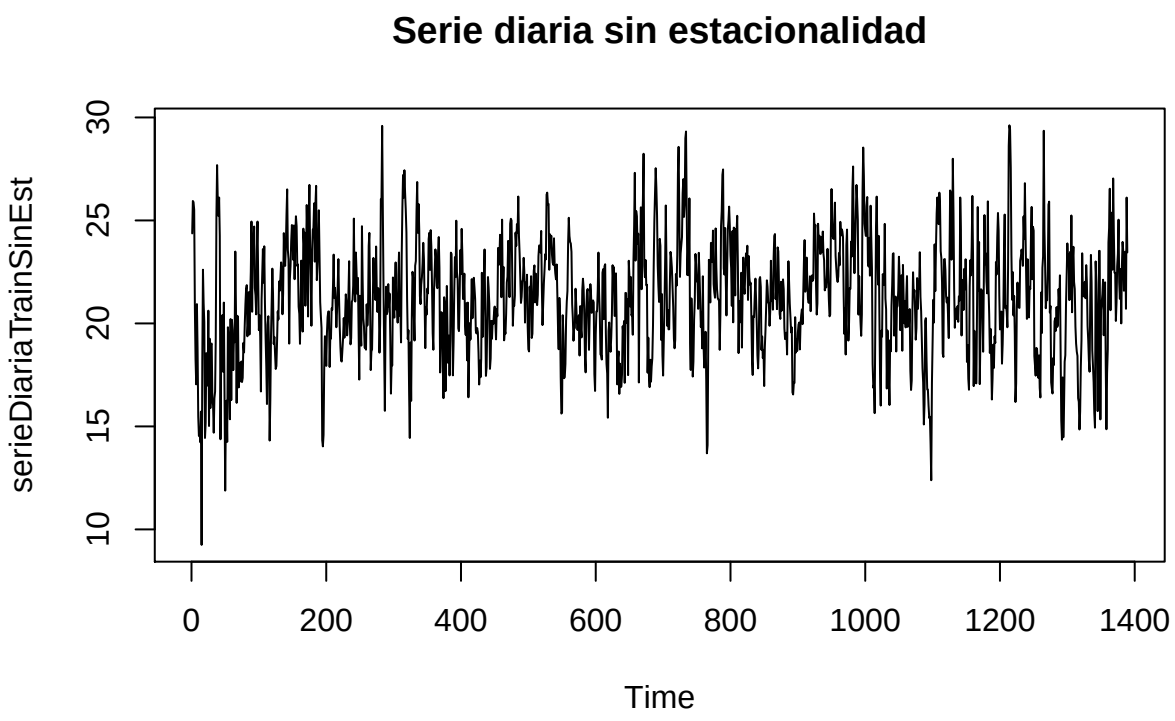
El siguiente paso consiste en tratar y eliminar la estacionalidad de la serie que se ha descubierto que presentan una fuerte componente de estacionalidad.

Lo primero es obtener la estacionalidad de las serie que se estableció en un periodo de 365:

```
estacionalidadDiaria.H1 <- decompose(serieDiariaTrain.ts)$seasonal[1:365]

#Dado que la división entre la longitud de la serie y de la estacionalidad  no es exacta,
#se utiliza "rep_len" para repetir la estacionalidad hasta la longitud indicada
estacionalidadDiariaRepetida <-
  rep_len(estacionalidadDiaria.H1,length(serieDiaria))[1:length(serieDiariaTrain)]

#Se calculan también la estacionalidad para los datos de validación
estacionalidadDiariaRepetidaTest <-
  rep_len(estacionalidadDiaria.H1,length(serieDiaria))[(length(serieDiariaTrain)+1):length(
    serieDiaria)]
```

Una vez determinada la estacionalidad a lo largo de la serie, se elimina de ésta y se muestra:

```
serieDiariaTrainSinEst <- serieDiariaTrain - estacionalidadDiariaRepetida
serieDiariaTestSinEst <- serieDiariaTest - estacionalidadDiariaRepetidaTest

#Se muestra la serie sin estacionalidad
plot.ts(serieDiariaTrainSinEst,main="Serie diaria sin estacionalidad")
```



Estas serían finalmente la serie sin estacionalidad (y sin tendencia pues se descubrió que era despreciable).

Determinación de estacionareidad

Antes de realizar la predicción sobre la serie es necesario saber si ésta es estacionaria. Para ello se utiliza el test de Dickey-Fuller ampliado:

```
#Test de Dickey-Fuller aumentado sobre serie diaria
adf.test(serieDiariaTrainSinEst)
```

```
## Warning in adf.test(serieDiariaTrainSinEst): p-value smaller than printed
## p-value
```

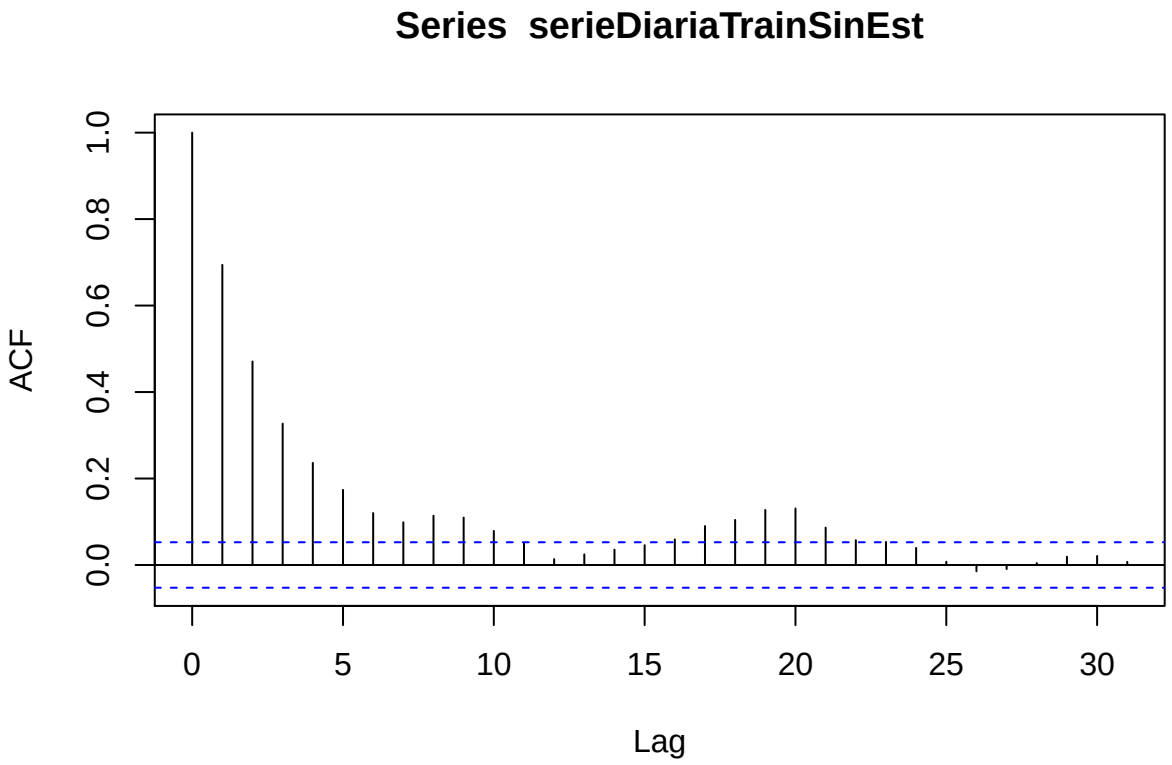
```
##
##  Augmented Dickey-Fuller Test
##
## data:  serieDiariaTrainSinEst
## Dickey-Fuller = -9.1881, Lag order = 11, p-value = 0.01
## alternative hypothesis: stationary
```

Se obtiene un p-value menor a 0.05 por lo que se puede determinar que es estacionaria y que no es necesario diferenciarla.

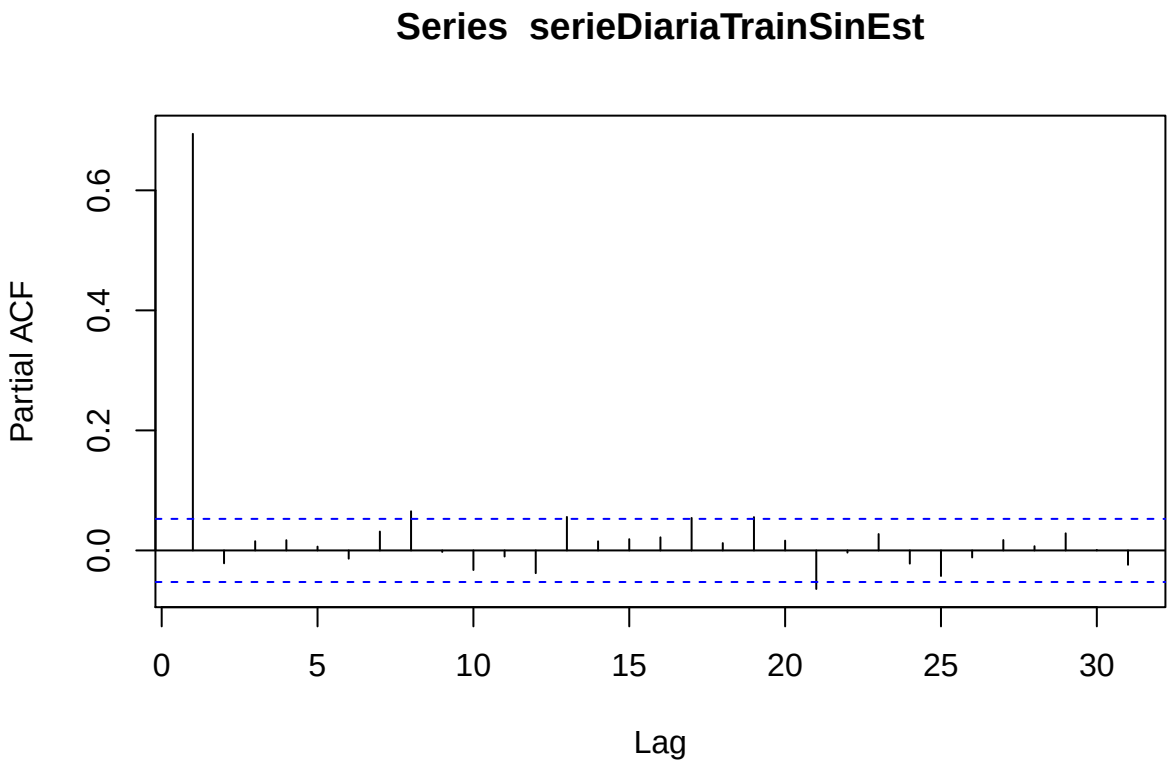
Predicción

Lo primero es obtener las gráficas ACF y PACF pues en función de ellas se podrá estimar qué tipo de modelado es recomendable para la predicción:

```
acf(serieDiariaTrainSinEst)
```




```
pacf(serieDiariaTrainSinEst)
```



Analizando la gráfica ACF se observe un decrecimiento regular a 0. Por otra parte, en la gráfica PACF decrece a 0 inmediatamente, sin que sea un decrecimiento regular, sinusoidal o alternando entre valores positivos y negativos. Por lo tanto, dado el comportamiento de ambas gráficas se puede determinar que se trata de un modelo autoregresivo y no de medias móviles para realizar la predicción. Además, dado que en la gráfica PACF hay hasta 3 valores por encima del umbral, "distintos de cero", se sabe que el orden del modelo autoregresivo será 3: AR(3).

Predicción de la serie diaria sobre validación

Realizando el ajuste mediante "ARIMA" de la serie diaria se tendría ARIMA(3,0,0) dadas las gráficas ACF y PACF, y dado que no fue necesario diferenciar la serie ninguna vez:

```
modelo <- arima(serieDiariaTrainSinEst, order=c(3,0,0))
valoresAjustados <- serieDiariaTrainSinEst + modelo$residuals
```

Se realizará ahora la predicción para los siguientes 365 días que se almacenaron para la validación, para tener así una idea del error que se ha tenido:

```
#Se calculan las predicciones
predicciones <- predict(modelo, n.ahead = 365)
valoresPredichos <- predicciones$pred

#Se calcula el error cuadrático acumulado del ajuste en train y test
sum((modelo$residuals)^2)
```

```
## [1] 5480.18
```

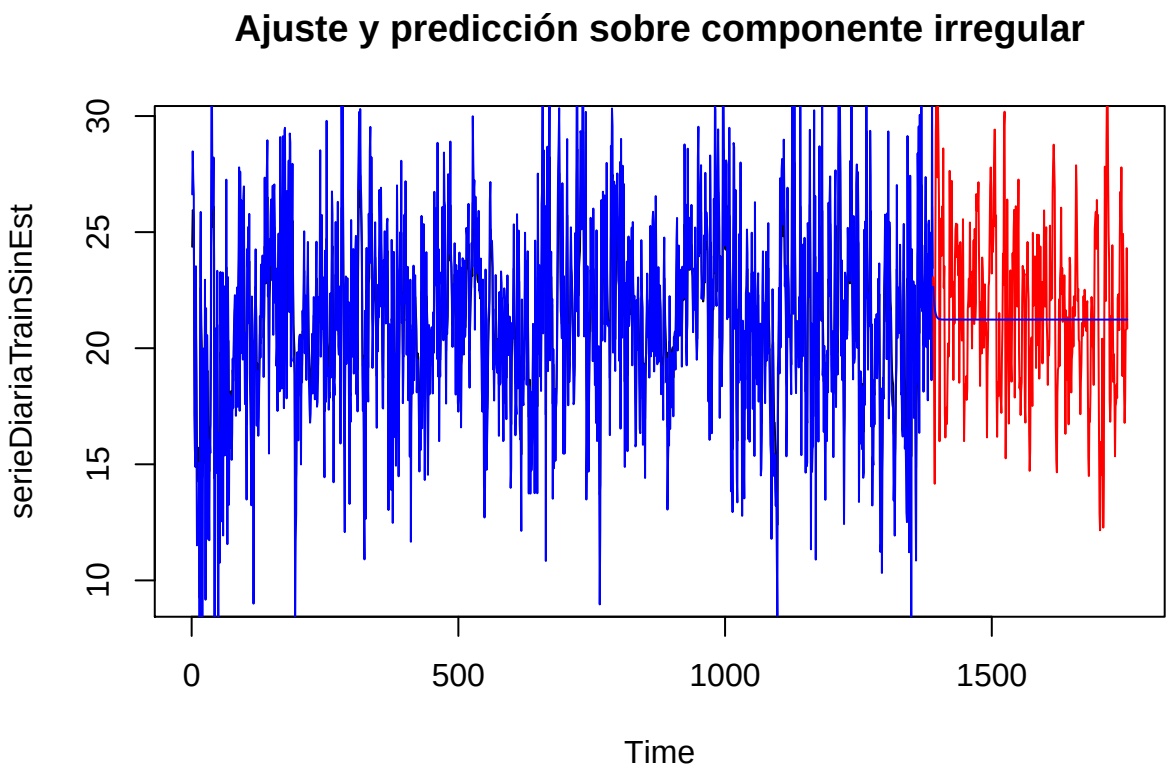
```
sum((valoresPredichos-serieDiariaTestSinEst)^2)
```

```
## [1] 4695.958
```

Se obtiene un error cuadrático acumulado de 5480 en train y de 4695 en test.

Se visualizan el ajuste realizado, los valores predichos y los reales para los datos de validación:

```
plot.ts(serieDiariaTrainSinEst, xlim=c(1,tiempoTestDiaria[length(tiempoTestDiaria)]),
        main="Ajuste y predicción sobre componente irregular")
lines(valoresAjustados, col="blue")
lines(tiempoTestDiaria, serieDiariaTestSinEst, col="red")
lines(tiempoTestDiaria, valoresPredichos, col="blue")
```

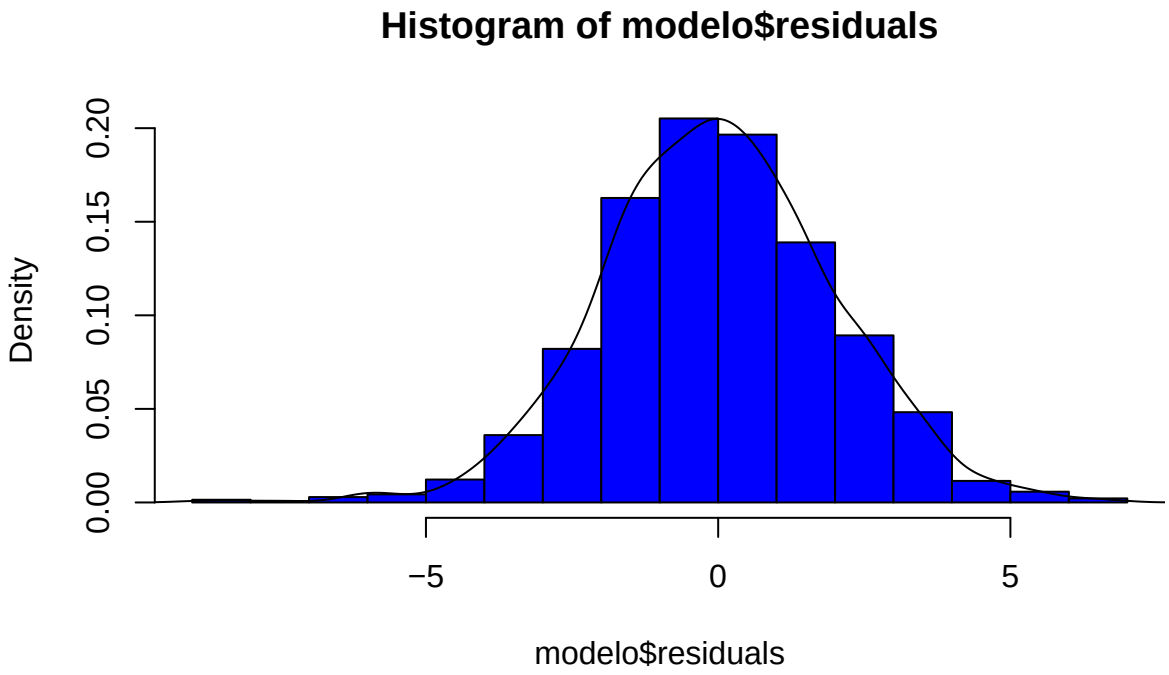


Se descubre que la predicción realizada (último tramo en azul) respecto a los valores reales (tramo en rojo) no se correponden. Sin embargo, el hecho de que los valores predichos sean constantes, sin variación, indica que posiblemente los residuos (componente irregular) sean aleatorios y con una distribución normal. Para validar esto se utiliza el Test de Box-Pierce para determinar la aleatoriedad de los residuos y se visualiza la distribución de los reisudos:

```
Box.test(modelo$residuals)
```

```
##
##  Box-Pierce test
##
## data:  modelo$residuals
## X-squared = 1.8865e-05, df = 1, p-value = 0.9965
```

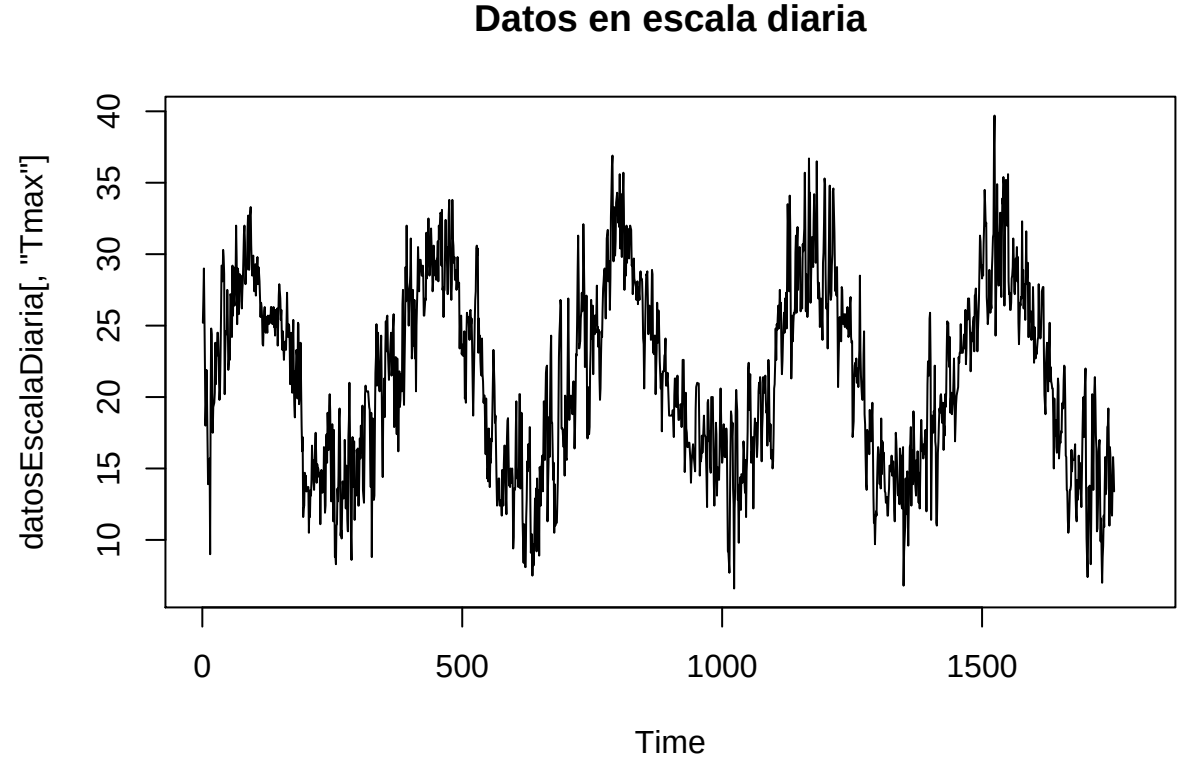
```
hist(modelo$residuals, col="blue", prob=T)
lines(density(modelo$residuals))
```



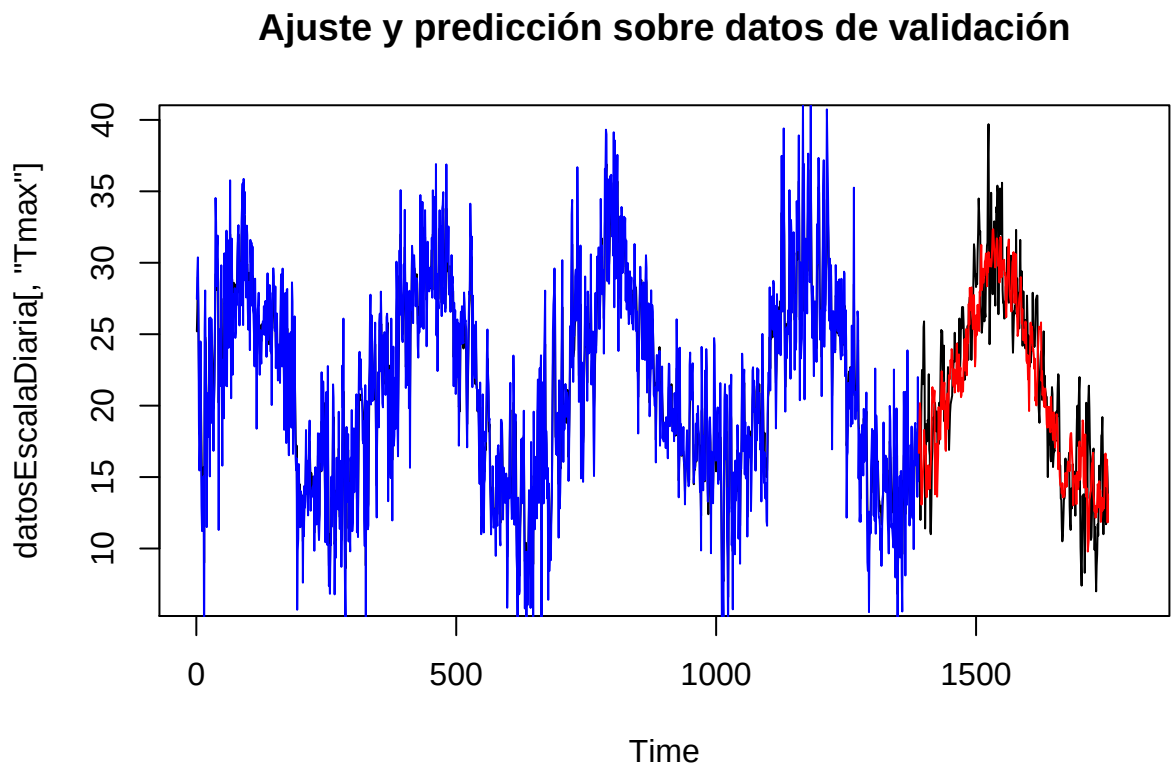
Efectivamente, existe aleatoriedad en los residuos y éstos siguen una distribución normal, lo cuál explicaría la predicción constante obtenida para la componente irregular.

Visualizando la predicción para los datos de validación añadiendo la componente de estacionalidad que se quitó:

```
plot.ts(datosEscalaDiaria[,"Tmax"],xlim=c(1,1800),main="Datos en escala diaria")
```



```r
plot.ts(datosEscalaDiaria[,"Tmax"],xlim=c(1,1800),
        main="Ajuste y predicción sobre datos de validación")
lines(valoresAjustados + estacionalidadDiariaRepetida, col="blue")
lines(valoresPredichos + estacionalidadDiariaRepetidaTest, col="red")
```



Se puede observar que el ajuste realizado (en azul) se corresponde bastante bien con los datos y la predicción realizada (en rojo) para los datos de validación presenta el mismo comportamiento por lo que se puede decir que se han logrado predecir los datos de validación. Por lo tanto, ésta será la predicción que se realice ya para los datos reales a predecir (primera semana de marzo 2018) utilizando todos los datos disponibles para tener la predicción más acertada posbile.

Predicción de la primera semana de Marzo 2018

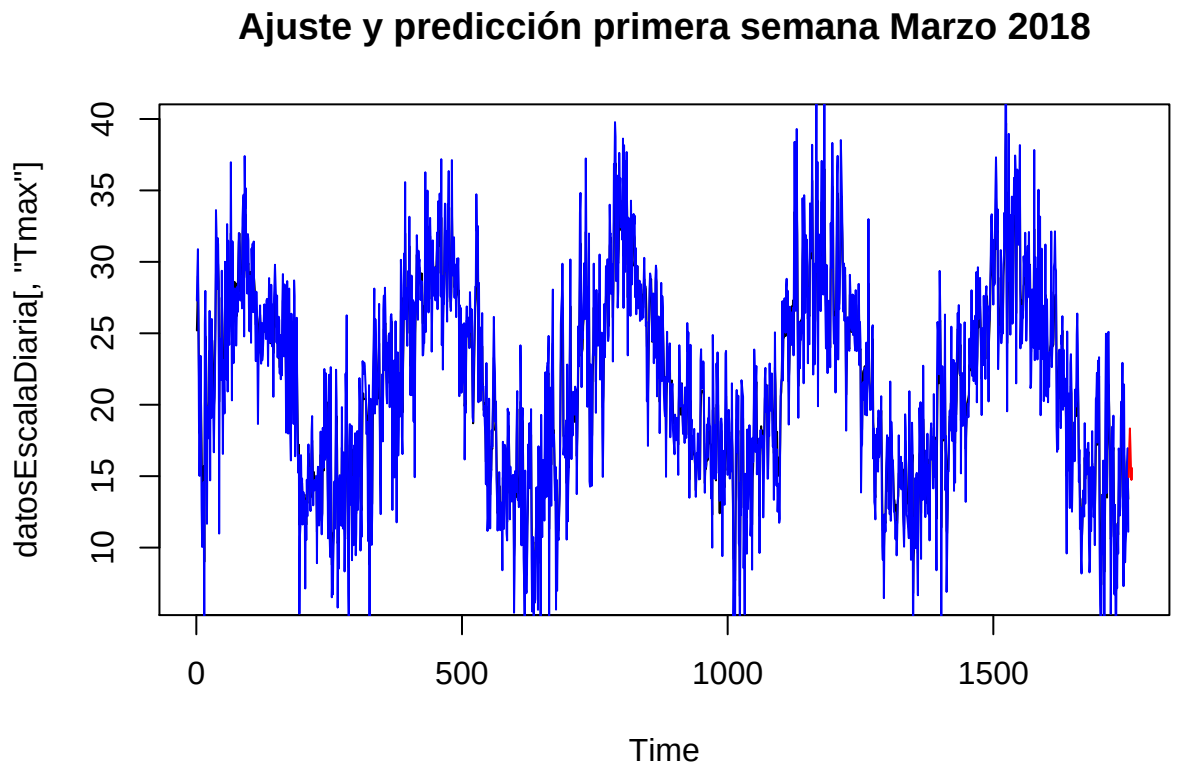
Ya estudiada la predicción realizada sobre los datos de validación, se toman todos los registros que se tienen y se realizan las mismas transformaciones sobre la serie para predecir finalmente la primera semana de Marzo 2018.

```r
#Se construye la serie diaria completa
serieDiaria <- datosEscalaDiaria[,"Tmax"]
serieDiaria.ts <- ts(serieDiaria, frequency = 365)

#Se calcula la estacionalidad
estacionalidadDiaria <- decompose(serieDiaria.ts)$seasonal[1:365]
estacionalidadDiariaRepetida <- rep_len(estacionalidadDiaria,length(serieDiaria))

#Se elimina la estacionalidad
serieDiaria <- serieDiaria - estacionalidadDiariaRepetida

#Se construye el modelo para la serie diaria según lo estudiado anteriormente
modeloDiario <- arima(serieDiaria,order=c(3,0,0))
```

```r
valoresAjustadosDiario <- serieDiaria + modeloDiario$residuals
#Se realiza la predicción para los próximos 7 días
valoresPredichosDiario <- predict(modeloDiario, n.ahead=7)$pred
```

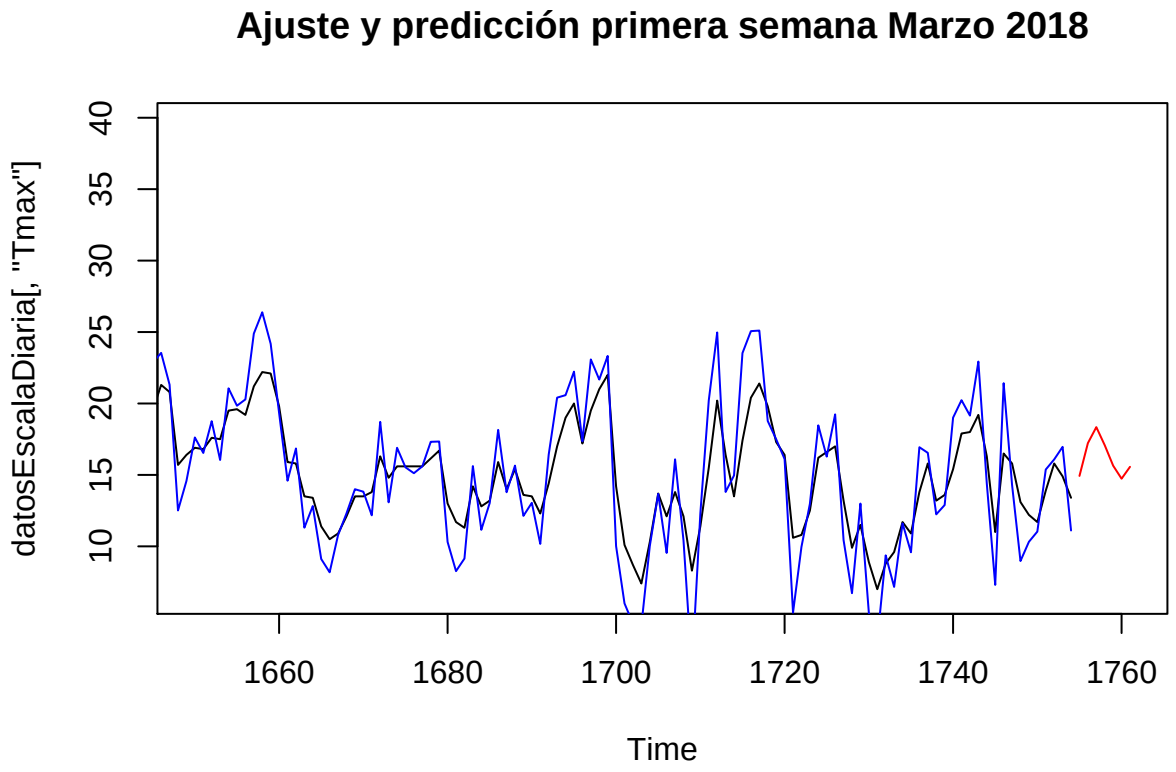
Ahora se deshacen los ajustes realizados para tener así finalmente los valores predichos reales (en rojo):

```r
#Se suma la estacionalidad teniendo en cuenta los 7 días proximos predichos
valoresAjustadosDiario <- valoresAjustadosDiario + rep_len(estacionalidadDiaria,length(
  serieDiaria)+7)[1:length(serieDiaria)]
valoresPredichosDiario <- valoresPredichosDiario + rep_len(estacionalidadDiaria,length(
  serieDiaria)+7)[(length(serieDiaria)+1):(length(serieDiaria)+7)]

plot.ts(datosEscalaDiaria[,"Tmax"], xlim=c(1,length(serieDiaria)+7),
        main="Ajuste y predicción primera semana Marzo 2018")
lines(valoresAjustadosDiario, col="blue")
lines(valoresPredichosDiario, col="red")
```



Centrando la visualización en los últimos meses:

```r
plot.ts(datosEscalaDiaria[,"Tmax"], xlim=c(1650,length(serieDiaria)+7),
        main="Ajuste y predicción primera semana Marzo 2018")
lines(valoresAjustadosDiario, col="blue")
lines(valoresPredichosDiario, col="red")
```



Esa sería la predicción realizada para la primera semana de Marzo 2018 con lo valores:

```
cat("Valores predichos de temperatura máxima para la primera semana de Marzo 2018 en Lanjarón
    -->",valoresPredichosDiario)
```

```
## Valores predichos de temperatura máxima para la primera semana de Marzo 2018 en Lanjarón
##     --> 14.93037 17.22477 18.3451 17.07632 15.64523 14.73646 15.56605
```

Serie temporal mensual

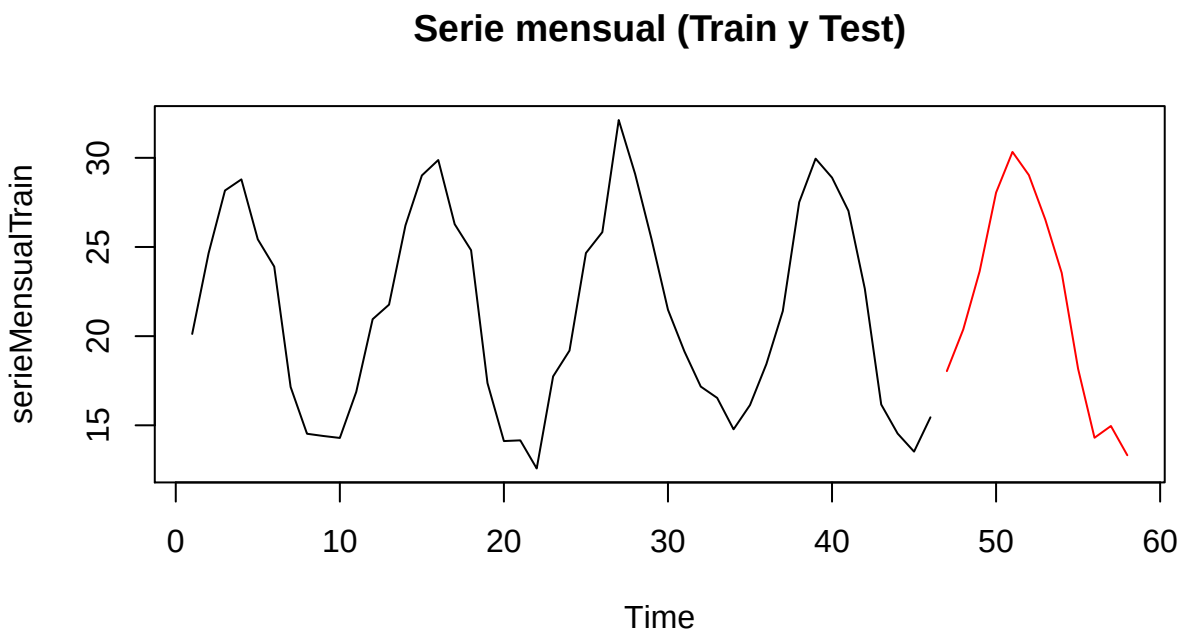
Se dividen los datos en Train y Test (los últimos 12 meses) para saber cómo de bien o mal se está realizando la predicción pues no se poseen los datos referentes a los meses de Marzo y Abril 2018:

```
serieMensual <- datosEscalaMensual[,"Tmax"]

datosEscalaMensualTrain <- datosEscalaMensual[1:46,]
datosEscalaMensualTest <- datosEscalaMensual[47:58,]

serieMensualTrain <- as.numeric(as.character(datosEscalaMensualTrain[,"Tmax"]))
serieMensualTest <- as.numeric(as.character(datosEscalaMensualTest[,"Tmax"]))
```

Visualizando la separación realizada en train y test:

```
tiempoTrainMensual <- 1:length(serieMensualTrain)
tiempoTestMensual <- (tiempoTrainMensual[length(tiempoTrainMensual)]+1):(tiempoTrainMensual[
  length(tiempoTrainMensual)]+length(serieMensualTest))
```

```r
#Se visualiza en rojo los datos de validación
plot.ts(serieMensualTrain, xlim=c(1,tiempoTestMensual[length(tiempoTestMensual)]),
        main="Serie mensual (Train y Test)")
lines(tiempoTestMensual, serieMensualTest, col="red")
```

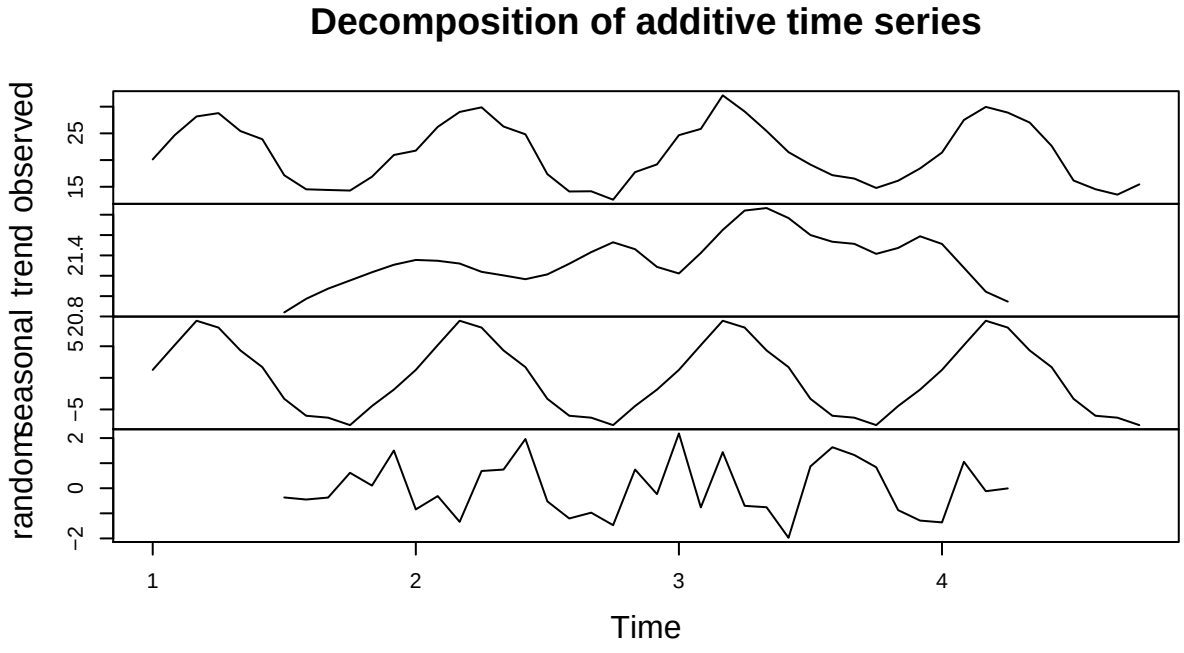


Se construye la serie temporal mensual correspondiente con los datos de temperatura máxima pues sobre esta variable se realizará la predicción:

```r
#Se establece una frecuencia de 12 correspondiente a los meses del año
serieMensualTrain.ts <- ts(serieMensualTrain, frequency = 12)
```

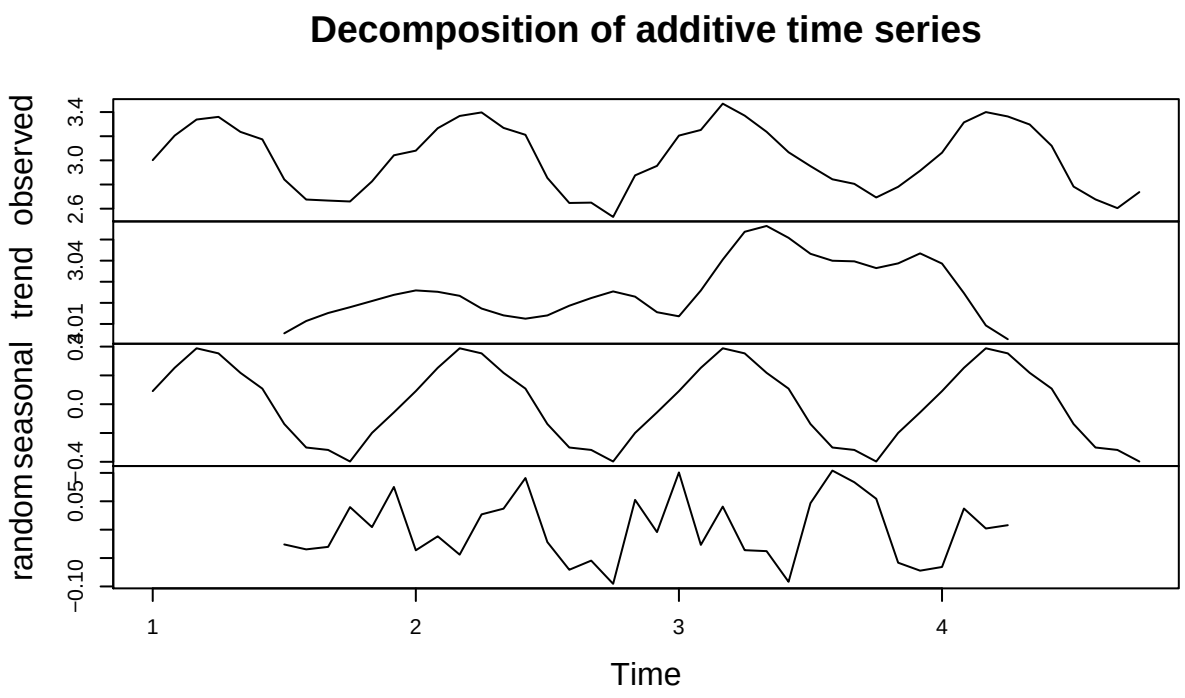
Descomponiendo la serie mensual:

```r
plot(decompose(serieMensualTrain.ts))
```



Aunque al tener muchos menos datos no se tenga la misma precisión que en la seie diaria, sí que vuelve a aparecer la misma "tendencia" con una influencia bastante reducida. La estacionalidad sigue siendo bastante clara pero no se está del todo seguro que la varianza en la componente irregular sea en torno a 0 y se trate de una serie estacionaria por lo que es recomendable que se realice una transformación logarítmica sobre ella para que la varianza sea más cercana a 0:

```
serieMensualTrain.log <- log(as.numeric(serieMensualTrain))
serieMensualTest.log <- log(as.numeric(serieMensualTest))
serieMensualTrain.ts <- log(serieMensualTrain.ts)
plot(decompose(serieMensualTrain.ts))
```



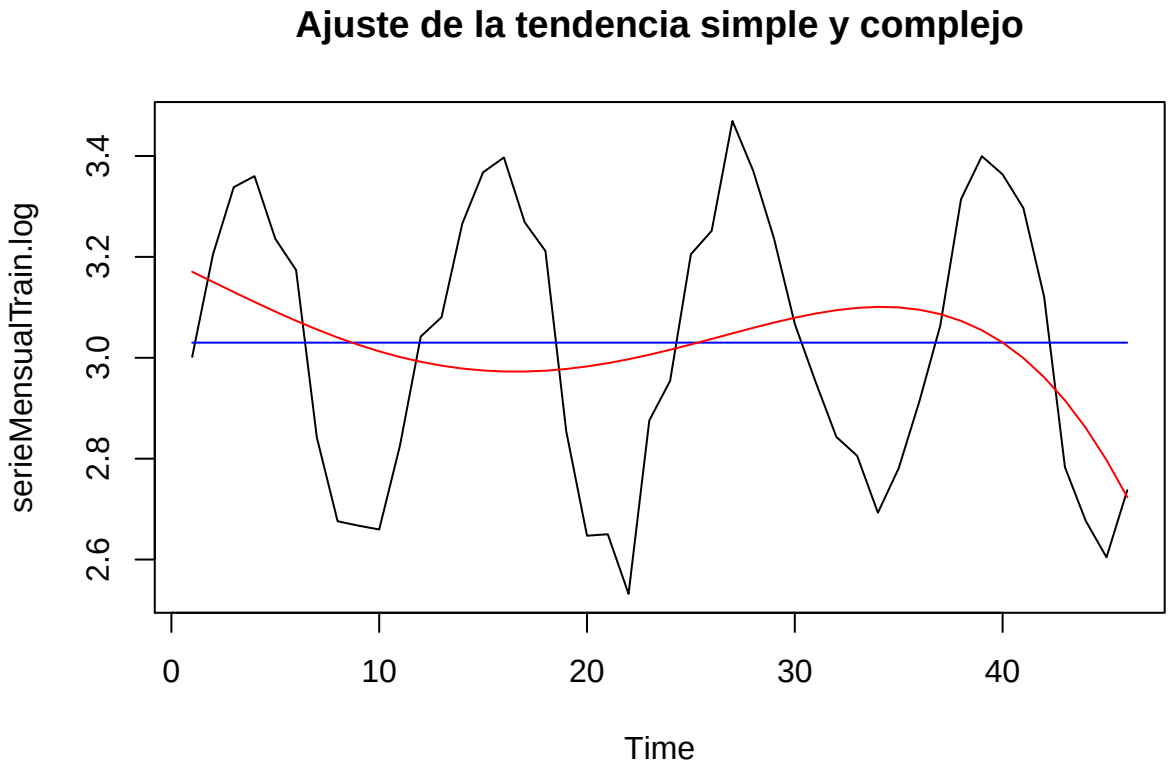
Estudio de las componentes de la serie temporal mensual

Tratamiento de la tendencia

Respecto a la serie mensual se descubre el mismo comportamiento que en la serie diaria, pues al haber hecho una transformación logarítmica sobre ella, el rango de la tendencia está entre 3 y 3.06, algo nuevamente irrelevante en el comportamiento de la serie:

```
#Se ajusta un modelo más complejo sobre la tendencia
parametros.H1 <- lm(serieMensualTrain.log ~ tiempoTrainMensual+I(tiempoTrainMensual^2)
                    +I(tiempoTrainMensual^3)+I(tiempoTrainMensual^4))

#Ajuste complejo de la tendencia
TendEstimadaTrComplex.H1 <- parametros.H1$coefficients[1]+tiempoTrainMensual*
  parametros.H1$coefficients[2]+tiempoTrainMensual^2*parametros.H1$coefficients[3]+
  tiempoTrainMensual^3*parametros.H1$coefficients[4]+tiempoTrainMensual^4*
  parametros.H1$coefficients[5]

#Ajuste simple
TendEstimadaTr.H1 <- 3.03
```

```r
#Se visualizan ambos ajustes
plot.ts(serieMensualTrain.log,main="Ajuste de la tendencia simple y complejo")
lines(tiempoTrainMensual, rep(TendEstimadaTr.H1,length(tiempoTrainMensual)), col = "blue")
lines(tiempoTrainMensual, TendEstimadaTrComplex.H1, col = "red")
```



Efectivamente, vuelven a ser prácticamente iguales dado el rango de su influencia por lo que se asume nuevamente que para la serie mensual tampoco existe una tendencia.

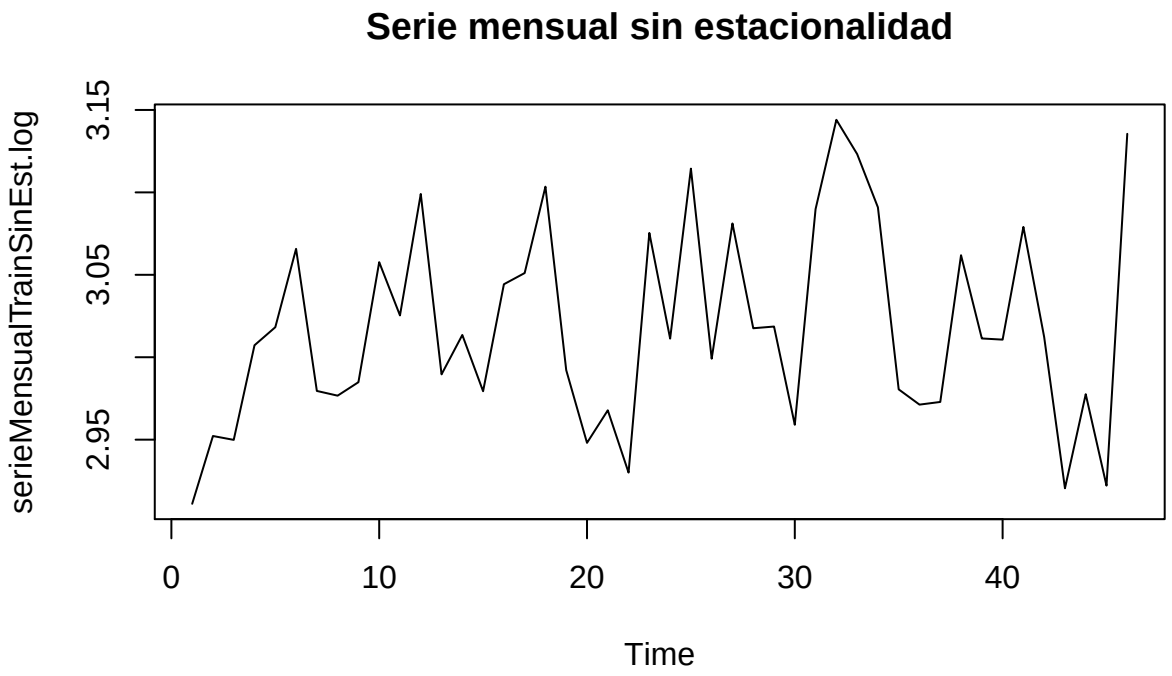
Tratamiento de la estacionalidad

Se obtiene la estacionalidad de la serie mensual que se fijó en 12 para la serie mensual:

```r
estacionalidadMensual.H1 <- decompose(serieMensualTrain.ts)$seasonal[1:12]

#Dado que la división entre la longitud de la serie y de la estacionalidad  no es exacta,
#se utiliza "rep_len" para repetir la estacinalidad hasta la longitud indicada
estacionalidadMensualRepetida <-
  rep_len(estacionalidadMensual.H1,length(serieMensual))[1:length(serieMensualTrain)]

#Se calculan las estacionalidades para los datos de validación
estacionalidadMensualRepetidaTest <-
  rep_len(estacionalidadMensual.H1,length(serieMensual))[(length(serieMensualTrain)+1):length(
    serieMensual)]
```

Se elimina la estacionalidad determinada de la serie y se muestra:

```r
serieMensualTrainSinEst.log <- serieMensualTrain.log - estacionalidadMensualRepetida
serieMensualTestSinEst.log <- serieMensualTest.log - estacionalidadMensualRepetidaTest

#Se muestra las serie sin estacionalidad
plot.ts(serieMensualTrainSinEst.log,main="Serie mensual sin estacionalidad")
```



Esta sería finalmente la serie mensual sin estacionalidad y sin tendencia dado que también resultaba ser despreciable.

Determinación de estacionareidad

Se aplica el test de Dickey Fuller aumentado para conocer si la serie es estacionaria:

```
#Test de Dickey-Fuller aumentado sobre serie mensual
adf.test(serieMensualTrainSinEst.log)
```

```
## Warning in adf.test(serieMensualTrainSinEst.log): p-value smaller than
## printed p-value
```

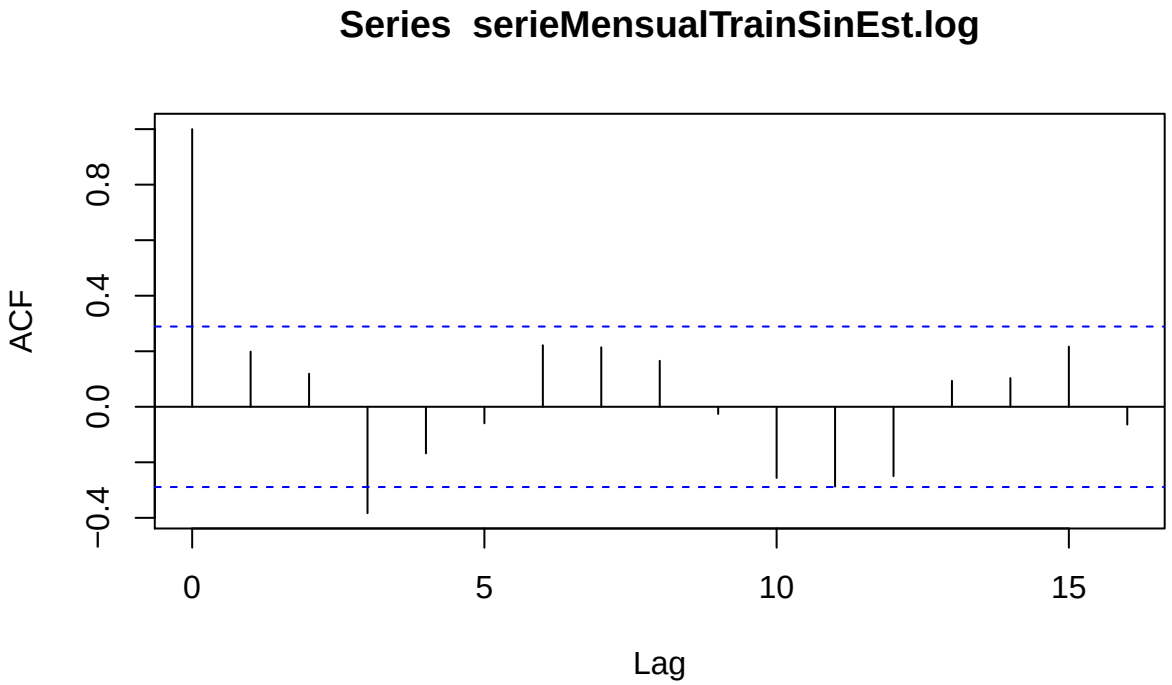
```
##
##  Augmented Dickey-Fuller Test
##
## data:  serieMensualTrainSinEst.log
## Dickey-Fuller = -4.4704, Lag order = 3, p-value = 0.01
## alternative hypothesis: stationary
```

Con un p-value menor a 0.05 se vuelve a determinar que es estacionaria y que no es necesario diferenciarla.

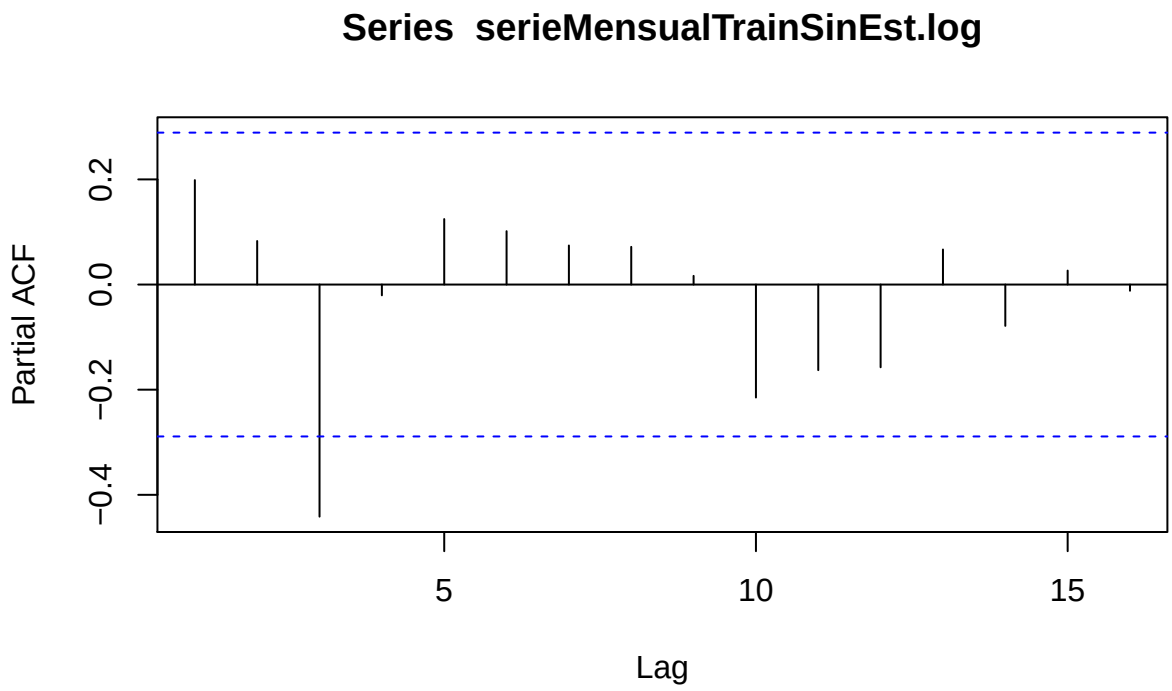
Predicción

Se obtienen las gráficas ACF y PACF de la serie mensual:

```
acf(serieMensualTrainSinEst.log)
```




```
pacf(serieMensualTrainSinEst.log)
```



En el caso de la serie mensual no son tan deterministas las gráficas como lo fueron en la serie diaria. Por una parte, en la gráfica ACF se puede intuir una decrecimiento sinusoidal aunque quizás demasiado precipitado. Por otra parte, en la gráfica PACF realmente es difícil encontrar un comportamiento de interés. Si no se tuviesen valores para validar la predicción de la series, se tomaría el primer caso como el más realista y se utilizaría un modelo autoregresivo. Sin embargo, dado que sí se poseen esos valores de validación, se puede igualmente ajustar también un modelo de medias móviles y descubrir cuál de los dos se comporta mejor.

Así, teniendo en cuenta el número de valores "distintos de 0" en ambas gráficas, lo modelos a construir serían: AR(1) y MA(2).

Predicción de la serie mensual sobre validación

Dado que las gráficas ACF y PACF no fueron demasiado deterministas y se considera el modelo AR(1) aunque no se descarta del todo un modelo MA(2), se compararán los errores acumulados de ambos modelos:

```
modeloAR <- arima(serieMensualTrainSinEst.log, order=c(1,0,0))
valoresAjustadosAR <- serieMensualTrainSinEst.log + modeloAR$residuals

#Se calculan las predicciones para los proximos 12 meses
prediccionesAR <- predict(modeloAR, n.ahead = 12)
valoresPredichosAR <- prediccionesAR$pred

#Se calcula el error cuadrático acumulado del ajuste en train y test
sum((modeloAR$residuals)^2)
```

```
## [1] 0.1582285
```

```
sum((valoresPredichosAR-serieMensualTestSinEst.log)^2)
```

```
## [1] 0.01871419
```

El error cuadrático acumulado para train es 0.158 y para test 0.018. Calculando ahora el error para el modelo de medias móviles:

```
modeloMA <- arima(serieMensualTrainSinEst.log, order=c(0,0,2))
valoresAjustadosMA <- serieMensualTrainSinEst.log + modeloMA$residuals

#Se calculan las predicciones para los proximos 12 meses
prediccionesMA <- predict(modeloMA, n.ahead = 12)
valoresPredichosMA <- prediccionesMA$pred

#Se calcula el error cuadrático acumulado del ajuste en train y test
sum((modeloMA$residuals)^2)
```

```
## [1] 0.1333083
```

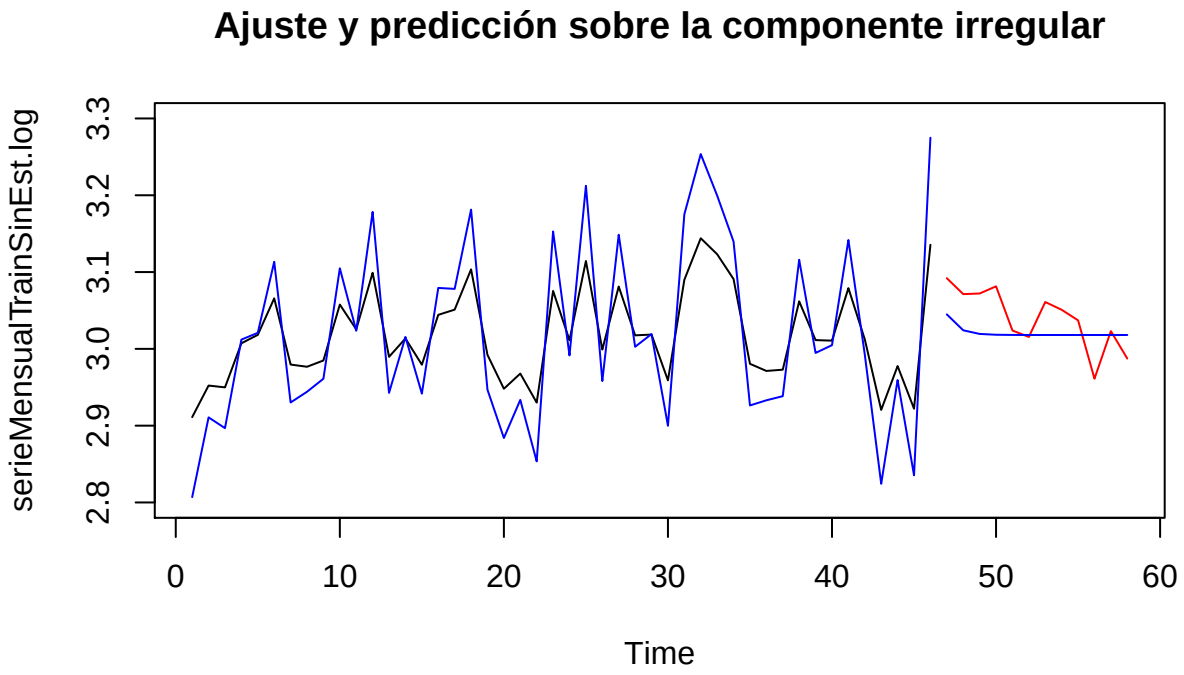
```
sum((valoresPredichosMA-serieMensualTestSinEst.log)^2)
```

```
## [1] 0.01668197
```

Se descubre que los valores obtenidos tanto para train como para test son muy parecidos. Dado que según las gráficas ACF y PACF el modelo autorregresivo parecía el más indicado, éste será el que se escoja finalmente.

Visualizando la predicción realizada (azul), junto con el ajuste y los valores reales (rojo) respecto a la componente irregular:

```
plot.ts(serieMensualTrainSinEst.log, xlim=c(1,tiempoTestMensual[length(tiempoTestMensual)]),
        ylim=c(2.8,3.3),main="Ajuste y predicción sobre la componente irregular")
lines(valoresAjustadosAR, col="blue")
lines(tiempoTestMensual, serieMensualTestSinEst.log, col="red")
lines(tiempoTestMensual, valoresPredichosAR, col="blue")
```

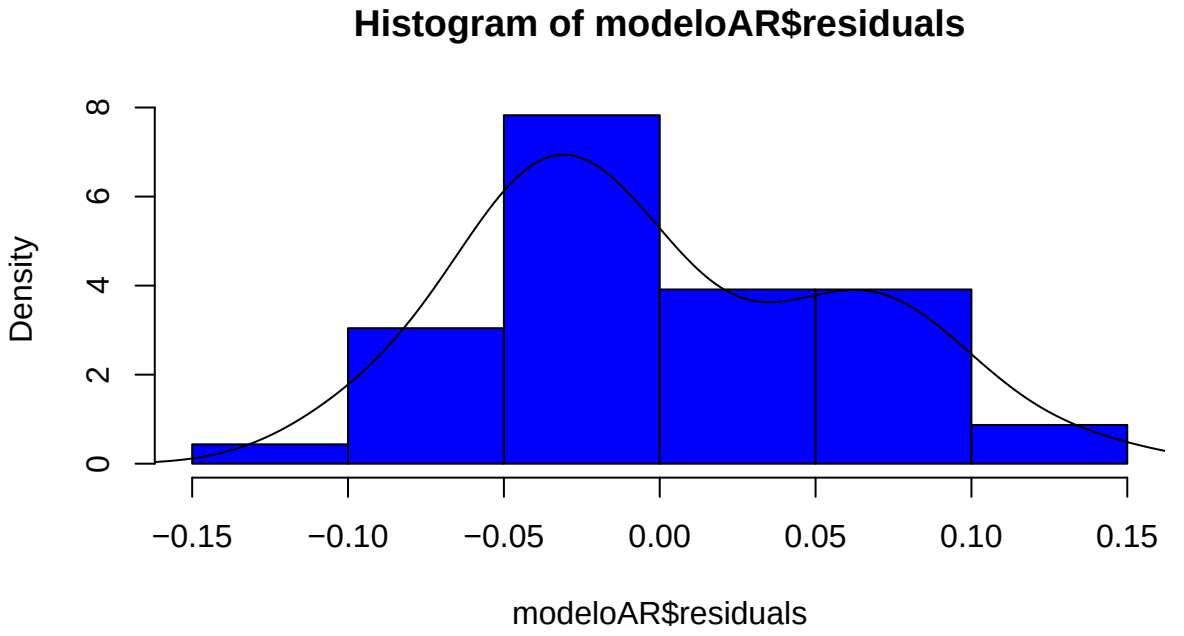


De forma similar a la serie diaria, se vuelve a realizar una predicción constante para la serie mensual; posiblemente indicando que vuelve a presentarse aleatoriedad y normalidad en los residuos:

```
Box.test(modeloAR$residuals)
```

```
##
##  Box-Pierce test
##
## data:  modeloAR$residuals
## X-squared = 0.030863, df = 1, p-value = 0.8605
```

```
hist(modeloAR$residuals, col="blue", prob=T)
lines(density(modeloAR$residuals))
```



```r
#Tests de Jarque Bera y Shapiro
jarque.bera.test(modeloAR$residuals)
```

```
##
##  Jarque Bera Test
##
## data:  modeloAR$residuals
## X-squared = 2.2129, df = 2, p-value = 0.3307
```

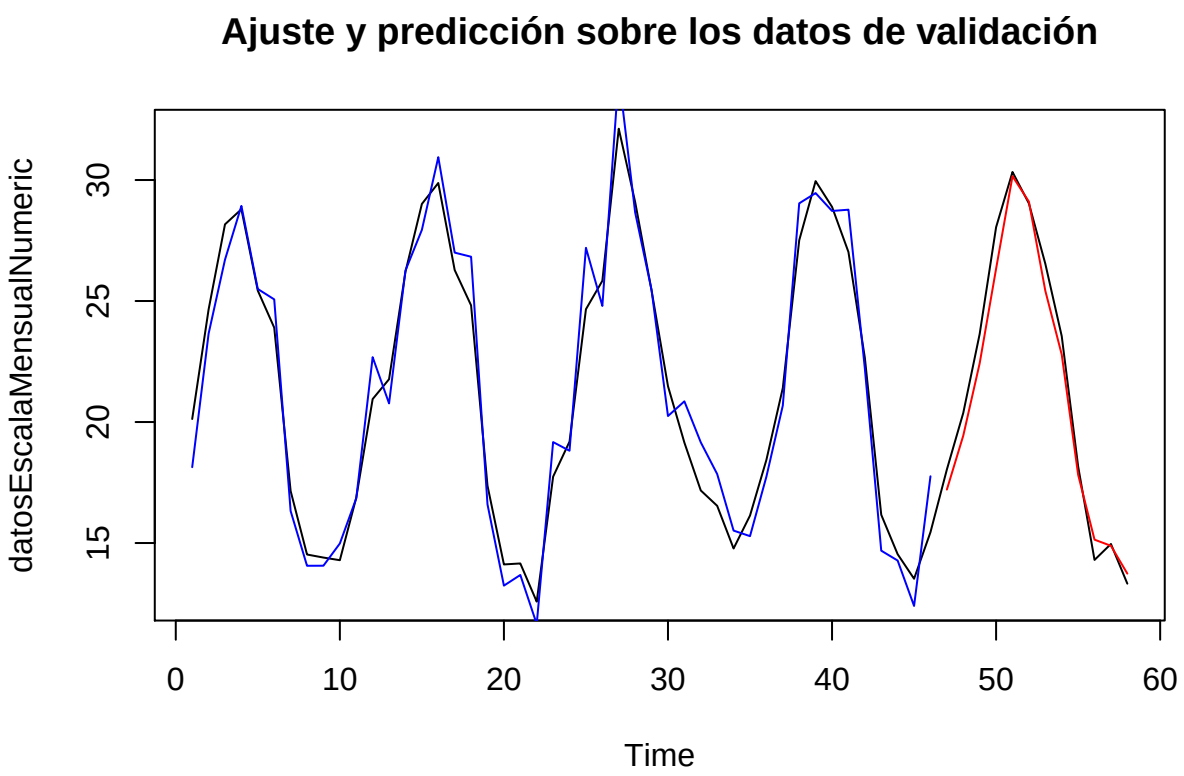
```r
shapiro.test(modeloAR$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  modeloAR$residuals
## W = 0.95941, p-value = 0.1088
```

Vuelve a determinarse aleatoriedad y normalidad en los residuos (apoyado por los test de Jarque Bera y Shapiro sobre normalidad), lo cuál explica la predicción realizada.

Mostrando ahora la predicción de los datos de validación sobre los datos reales:

```r
datosEscalaMensualNumeric <- as.numeric(as.character(datosEscalaMensual[,"Tmax"]))
plot.ts(datosEscalaMensualNumeric,main="Ajuste y predicción sobre los datos de validación")
lines(exp(valoresAjustadosAR + estacionalidadMensualRepetida), col="blue")
lines(exp(valoresPredichosAR + estacionalidadMensualRepetidaTest), col="red")
```



Se observa que para la serie mensual se ha conseguido también un gran ajuste tanto para los datos de entrenamiento como para los de validación. De igual forma, se utilizará ya toda la serie registrada para la predicción más realista posible de los meses de Marzo y Abril 2018.

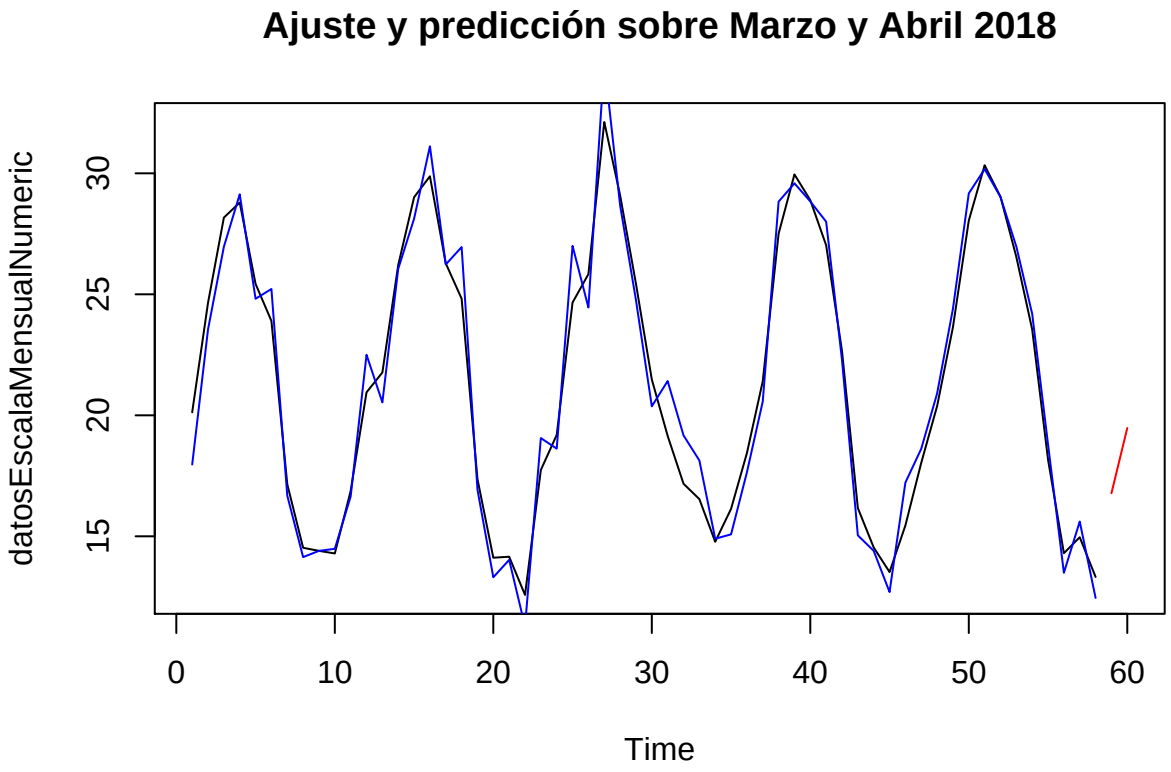
Predicción de Marzo 2018 y Abril 2018

Finalmente se realiza la predicción para los meses de Marzo 2018 y Abril 2018 aplicando los ajustes realizados para toda la serie mensual registrada:

```
#Se construye la serie mensual completa
serieMensual <- as.numeric(as.character(datosEscalaMensual[,"Tmax"]))
serieMensual.ts <- ts(serieMensual, frequency = 12)

#Se realiza la transformación logarítmica necesaria para la serie mensual
serieMensual.log <- log(serieMensual)
serieMensual.ts <- log(serieMensual.ts)

#Se calcula la estacionalidad
estacionalidadMensual <- decompose(serieMensual.ts)$seasonal[1:12]
estacionalidadMensualRepetida <- rep_len(estacionalidadMensual,length(serieMensual.log))

#Se eliminan la estacionalidad
serieMensual.log <- serieMensual.log - estacionalidadMensualRepetida


#Se construye el modelo para la serie mensual
modeloMensual <- arima(serieMensual.log,order=c(1,0,0))
valoresAjustadosMensual <- serieMensual.log + modeloMensual$residuals
#Se predicen los próximos 2 meses
valoresPredichosMensual <- predict(modeloMensual, n.ahead=2)$pred
```

Se deshacen los ajustes para obtener los valores predichos finales (en rojo):

```
#Se suman las estacionalidades teniendo en cuenta los 2 meses proximos predichos
valoresAjustadosMensual <- valoresAjustadosMensual + rep_len(estacionalidadMensual,length(
  serieMensual)+2)[1:length(serieMensual)]
valoresPredichosMensual <- valoresPredichosMensual + rep_len(estacionalidadMensual,length(
  serieMensual)+2)[(length(serieMensual)+1):(length(serieMensual)+2)]

#Se deshace la transformación logarítmica realizada
valoresAjustadosMensual <- exp(valoresAjustadosMensual)
valoresPredichosMensual <- exp(valoresPredichosMensual)

datosEscalaMensualNumeric <- as.numeric(as.character(datosEscalaMensual[,"Tmax"]))
plot.ts(datosEscalaMensualNumeric, xlim=c(1,length(serieMensual)+2),
        main="Ajuste y predicción sobre Marzo y Abril 2018")
lines(valoresAjustadosMensual, col="blue")
lines(valoresPredichosMensual, col="red")
```



Así, estos serían los valores predichos:

```
cat("Valores predichos de temperatura máxima para los meses de Marzo y Abril 2018 en Lanjarón
    -->",valoresPredichosMensual)
```

```
## Valores predichos de temperatura máxima para los meses de Marzo y Abril 2018 en Lanjarón
##     --> 16.77855 19.47462
```