SERIES TEMPORALES Y MINERÍA DE FLUJOS DE DATOS

MÁSTER EN CIENCIA DE DATOS E INGENIERÍA DE COMPUTADORES

Trabajo autónomo I: Series Temporales

Curso 2021-2022

Autora: Lidia Sánchez Mérida

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación\

Granada, 15 de Abril de 2022

Parte teórica

Preprocesamiento

Tras realizar un análisis general sobre la serie temporal en cuestión, hemos podido comprobar que no dispone de valores perdidos y que su componente ruidosa es totalmente aleatoria. Por lo tanto, no se han aplicado técnicas de preprocesamiento.

Análisis de tendencia y de estacionalidad

Una **serie temporal** es un conjunto de datos que representa una magnitud observada durante un período de tiempo determinado. Para facilitar el análisis de su comportamiento **descomponemos la serie temporal** en tres componentes principales:

1. **Tendencia**. Representa el incremento o decremento de los valores durante el tiempo observado. Existen diversas técnicas para modelar la tendencia de una serie temporal, entre las cuales se encuentran las dos que se emplean en este trabajo:
   - **Estimación funcional**. Este método púramente visual consiste en modelar la tendencia a partir de su aproximación a la función matemática cuyo comportamiento sea más similar a su representación gráfica.
   - **Filtrado**. Aplica un filtro de **Medias Móviles** para facilitar la detección de la tendencia y la existencia de ciclos en una serie temporal estacional. El objetivo consiste en calcular la media aritmética de un conjunto de puntos en el tiempo para intentar aproximar la serie temporal a una función matemática.
2. **Estacionalidad**. La componente estacional indica que existen períodos de tiempo determinados en los que existe un determinado patrón de datos, como puede ser mensual o anual. Para simplificar el análisis y modelado de la serie temporal, se pretende aplicar el método de **diferenciación** para obtener cada una de las componentes por separado de modo que, posteriormente, se pueda restar la estacional a la serie temporal original.
3. **Ruido**. Contiene los datos restantes de la serie temporal que no pueden ser representados con ninguna de las componentes anteriores.

Estacionariedad

Una serie temporal se caracteriza por ser estacionaria si sus **propiedades se mantienen constantes durante el tiempo** observado. Existen tanto técnicas visuales como métodos estadísticos que permiten conocer si una serie temporal es estacionaria.

- **Test de *Dickery-Fuller* aumentado**. Se trata de un test estadístico cuya **hipótesis nula** representa la teoría de que la **serie no es estacionaria**. Como en cualquier test estadístico existen dos posibles situaciones:
  - Estableciendo un umbral máximo de 0.05, si el p-valor resultante del test es mayor entonces no existe evidencia estadística de que la serie sea estacionaria. En tal caso deberemos aplicar tantas **diferenciaciones sobre la serie temporal** como sean necesarias hasta obtener una serie estacionaria.
  - Si el p-valor resultante es menor que el umbral fijado, entonces el test rechaza la hipótesis nula al 95% de confianza y por lo tanto se puede confirmar que **la serie temporal es estacionaria**.
- **Gráficos ACF y PACF**. Representan la **función de autocorrelación** total y parcial, respectivamente, asociadas a una serie temporal. Si se trata de una serie estacionaria, entonces la gráfica ACF se aproximará rápidamente a cero.

Modelado de series temporales

Uno de los modelos más utilizados para la predicción de series temporales es el conocido **ARIMA(p, d, q)** (*AutoRegressive Integrated Moving Average*) **no estacional**. Su origen reside en la combinación de dos tipos de modelos diferentes:

- **Modelos Autorregresivos** (AR). Aplican una regresión múltiple sobre la propia serie temporal con el objetivo de predecir los valores futuros a partir de un **conjunto de p valores relacionados en el tiempo**, y por ende, con una correlación muy alta entre sí.
- **Modelos de Medias Móviles** (MA). Representa una variante del modelo AR anterior cuyo enfoque consiste en calcular las predicciones futuras de una serie temporal en base a los **errores cometidos al predecir los q retardos anteriores** desde el instante actual.

Adicionalmente, un modelo $ARIMA$ también considera el **número de diferenciaciones** que han sido aplicadas sobre la serie temporal, siendo expresado mediante el parámetro d.

Selección del tipo de modelo

Para conocer qué tipo de modelo podemos utilizar para realizar las predicciones sobre una serie temporal, se pueden aplicar las siguientes reglas tras representar los gráficos ACF y PACF:

- Los modelos AR se caracterizan por disponer de un **gráfico ACF que tiende a cero**, con un orden p igual al número de valores diferentes que no se encuentran dentro de los umbrales definidos en un gráfico PACF.
- Los modelos MA se definen a partir de un **gráfico PACF cuyos valores tienden a disminuir**, y cuyo orden q es igual al número de valores distintos fuera de los umbrales establecidos en un gráfico ACF.

Validación de los modelos

Uno de los métodos más popularmente utilizados para validar un modelo entrenado sobre una serie temporal consiste en **analizar la autocorrelación de los residuos**, entendiendo como residuo el error cometido por el clasificador al generar las predicciones. Podemos determinar que la capacidad de generalización de un modelo es máxima si cumple las siguientes condiciones:

- Sus residuos **no están correlados**, lo que significa que los datos no modelados son aleatorios y no existe conocimiento útil en ellos para mejorar el clasificador. Para demostrar esta teoría se puede aplicar el **test de Box-Pierce cuya hipótesis nula afirma que los datos son independientes**. Estableciendo una confianza del 95%, si el p-valor resultante es menor que 0.05 entonces podríamos determinar que los residuos están relacionados entre sí y por ende el modelo es mejorable.
- Los residuos siguen una **distribución normal**, en cuyo caso simboliza que el modelo es estable y dispone de una varianza constante, por lo que es suficientemente robusto como para que la tasa de error no se dispare en ninguna ocasión. Los **tests de Jarque Bera y Shapiro-Wilk** son popularmente usados para verificar esta condición. Como **hipótesis nula se establece que los datos siguen una distribución normal** por lo que, si el p-valor resultante es menor que el umbral fijado a 0.05, se rechazaría esta teoría y por ende los datos no siguen una distribución normal.

Comparación de varios modelos

Existen diversas métricas que permiten comparar la bondad de diversos modelos entrenados y validados sobre el mismo conjunto de datos. Una de las más popularmente utilizadas en problemas de regresión es el **error cuadrático acumulado del ajuste**, aplicable tanto en el conjunto de entrenamiento a partir de los residuos del modelo, como en el conjunto de test mediante la diferencia entre las predicciones sobre el conjunto de test y los valores reales del mismo. De este modo, podemos comparar la capacidad de generalización de varios modelos en función del **rendimiento que demuestran al generar valores de la serie temporal en el futuro** a partir de los datos disponibles de momentos anteriores.

Predicciones reales

Una vez disponemos disponemos de la formulación de un modelo entrenado y validado para resolver el problema en cuestión, **repetimos la descomposición de la serie además del entrenamiento del modelo con todo el conjunto de datos** para generar las predicciones requeridas. A continuación, deshacemos las modificaciones realizadas sobre la serie temporal en orden inverso para conseguir adaptar las predicciones generadas a la serie temporal original. Así, para esta serie temporal los pasos a seguir se describen a continuación:

1. **Deshacer la diferenciación** realizada para convertir la serie temporal original en estacionaria.
2. **Añadir la componente estacional** que se eliminó al comienzo del análisis para facilitar el modelado de la serie.

Parte práctica

Análisis de la serie temporal

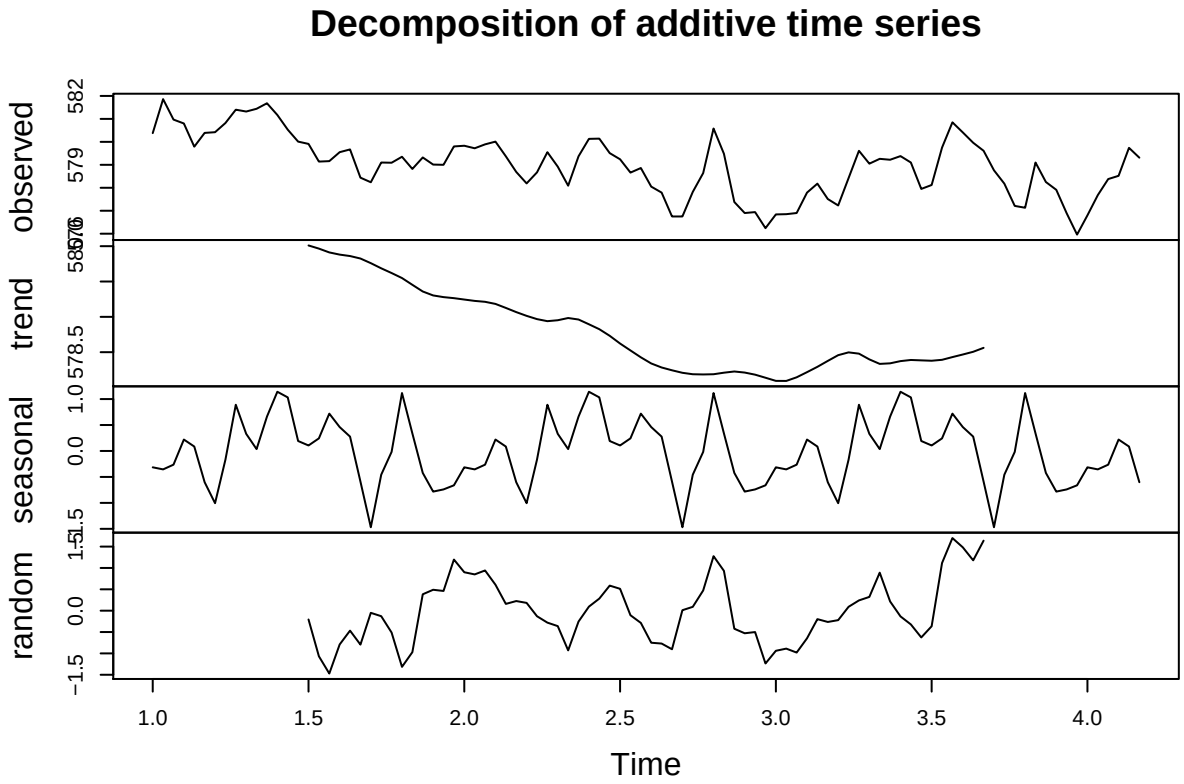
En primer lugar cargamos los datos del fichero proporcionado utilizando la función `scan` y creamos un objeto específico para el tratamiento de series temporales. La primera verificación realizada consiste en conocer si existen valores perdidos en la serie para aplicar técnicas de imputación o eliminación. Sin embargo, tal y como podemos observar, **no existen datos faltantes** en esta serie temporal.

```
# Cargamos los datos de la serie temporal
serie <- scan("dataseries.dat")
# Comprobamos si existen valores perdidos
sum(is.na(serie))
```

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

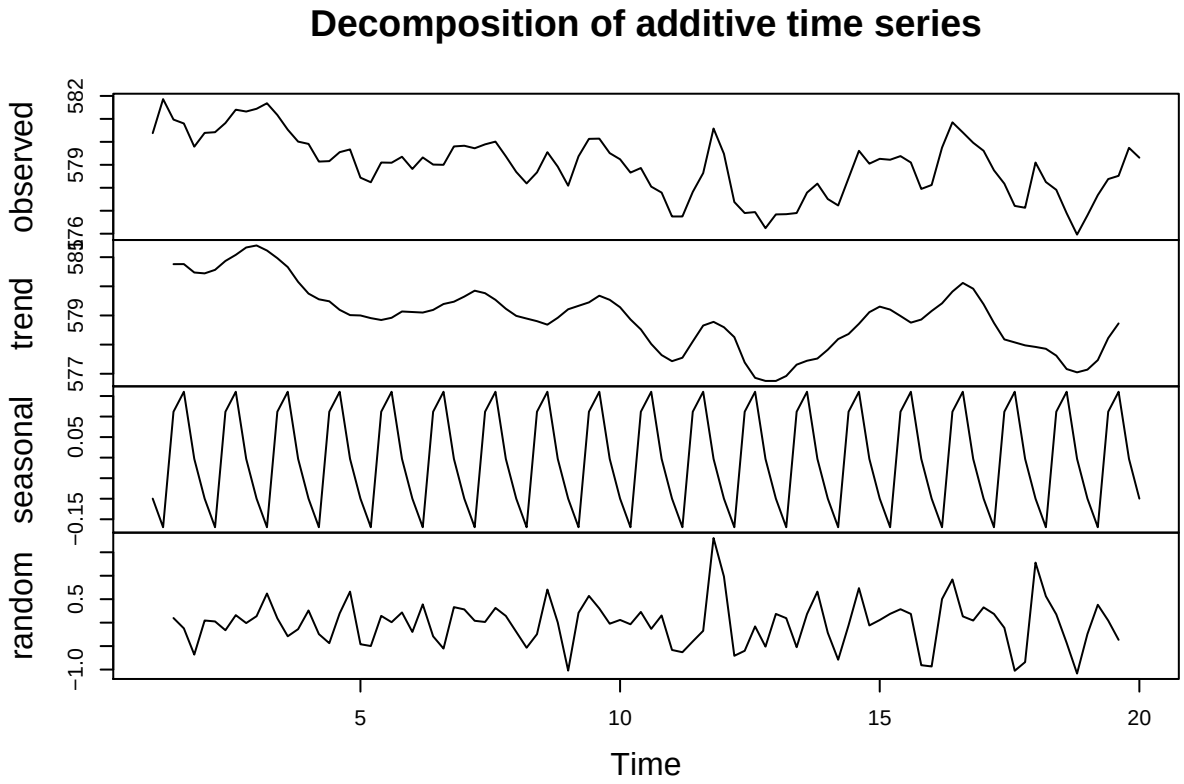
Continuamos su análisis aplicando una **descomposición aditiva** para dividir la serie temporal en las tres componentes explicadas en la parte teórica: tendencia, estacionalidad y ruido. Un paso previo a la descomposición consiste en determinar **la frecuencia de la serie**. Al no conocer su valor real ni disponer de ningún tipo de contexto sobre la procedencia de los datos, considerando únicamente la existencia de 96 valores podemos intuir que se trata de una **serie mensual**. Si observamos las diferentes componentes representadas en el gráfico, a simple vista parece que la serie temporal con una frecuencia de 30 días **no tiene tendencia**, ya que su comportamiento no se asemeja a ninguna función matemática conocida. En cambio, **sí parece tratarse de una serie estacional** puesto que en la tercera gráfica aparece un mismo patrón repetitivo durante el mismo período de tiempo. Adicionalmente, en la última gráfica podemos observar que **el ruido parece totalmente aleatorio** por lo que estos datos no aportan conocimiento útil para el problema de predicción.

```
# Creamos un objeto time-series suponiendo una frecuencia mensual (30 días)
serie.ts <- ts(serie, frequency=30)
# Descomposición aditiva de la serie temporal
plot(decompose(serie.ts))
```



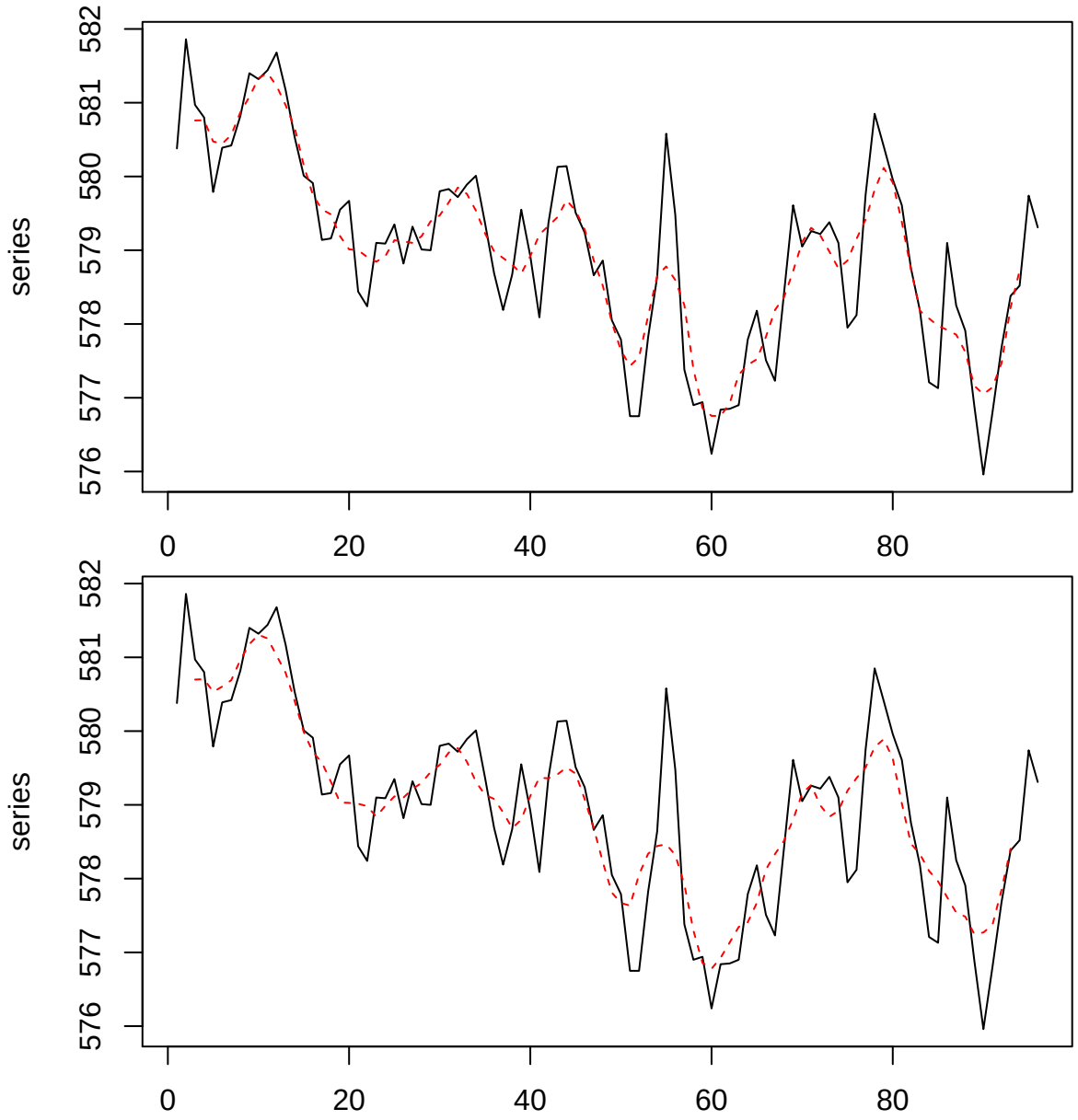
Sin embargo, visualizando el contenido del fichero parece existir un patrón divisorio de cinco valores, a excepción de la última línea, por lo que su **frecuencia podría ser de cinco días**. Si realizamos una segunda descomposición aditiva con este nuevo valor, podemos observar en la siguiente representación que la serie temporal **tampoco parece disponer de una tendencia reconocible, aunque su patrón estacional es más fácil de identificar**, comparado con el obtenido en el caso anterior asumiendo una frecuencia mensual. En relación a la componente ruidosa podemos observar que sus datos siguen pareciendo totalmente aleatorios, por lo que no se puede extraer información útil que nos ayude en el análisis y modelado de esta serie temporal.

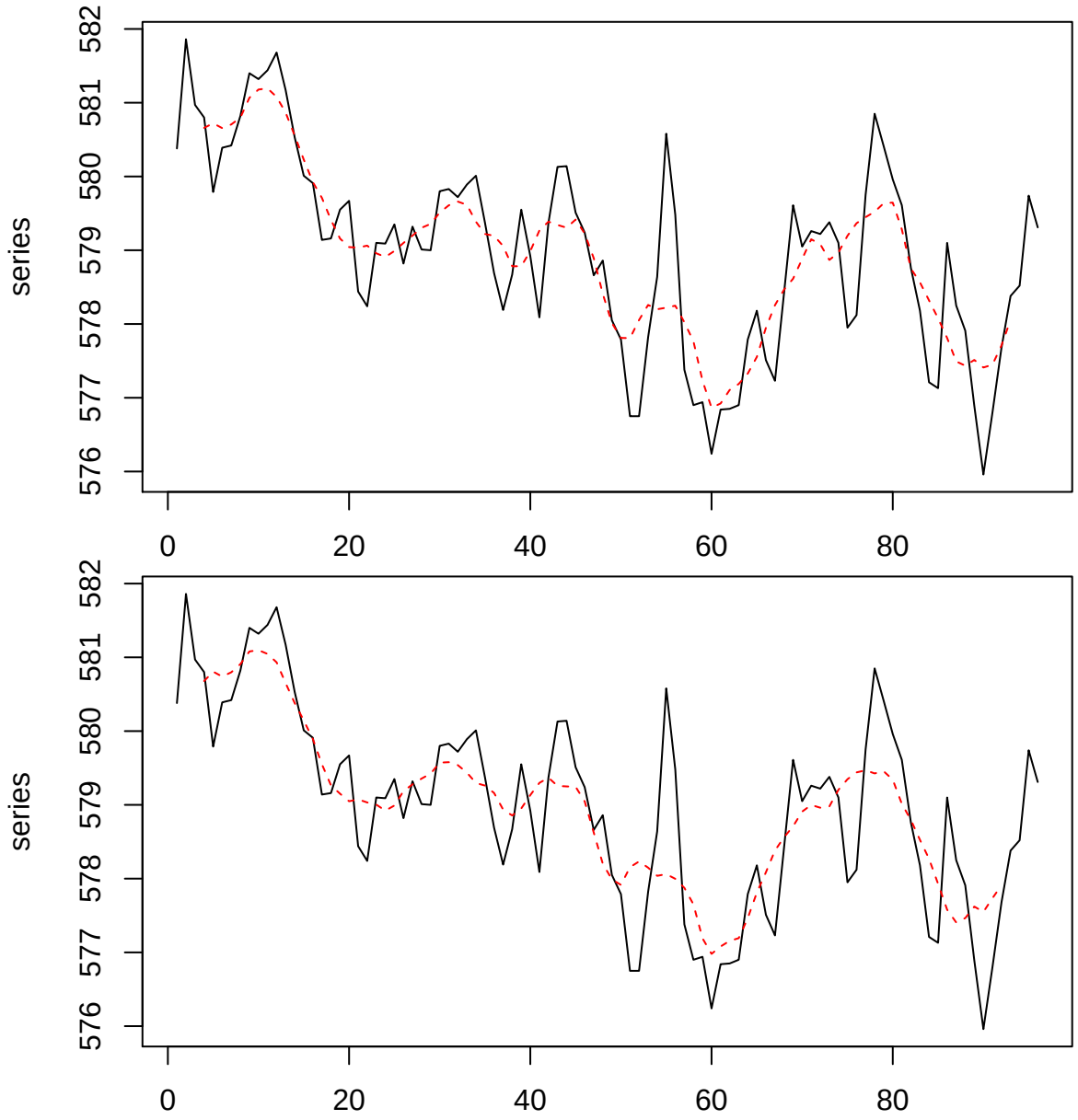
```
# Cargamos los datos de la serie temporal
serie <- scan("dataseries.dat")
# Creamos un objeto time-series suponiendo una frecuencia de 5 días
serie.ts <- ts(serie, frequency=5)
# Descomposición aditiva de la serie temporal
plot(decompose(serie.ts))
```

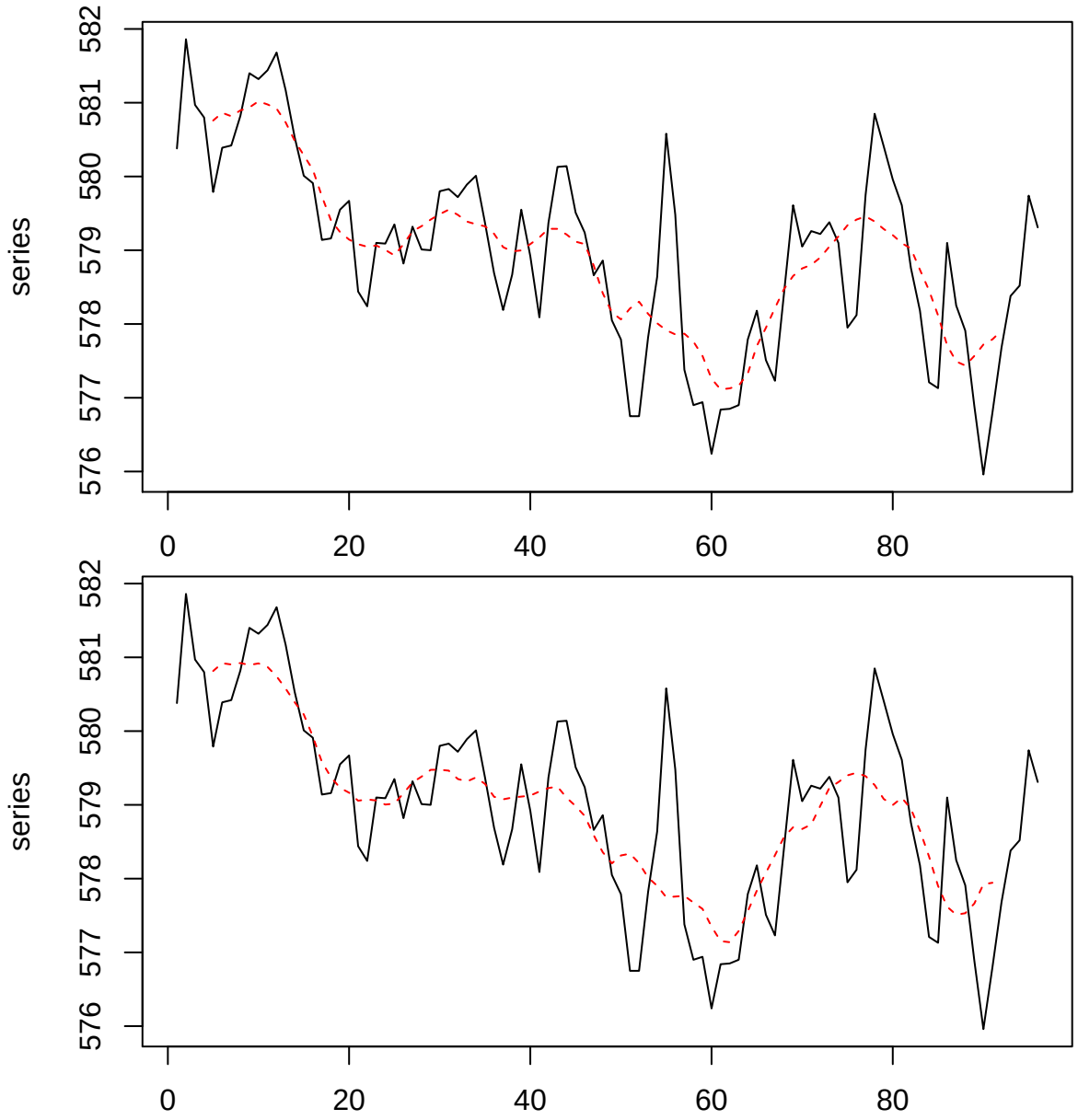


Tras comparar los resultados gráficos de sendos experimentos, en mi opinión el comportamiento de esta serie temporal parece ser más fácil de explicar al establecer una **frecuencia de cinco días** y por ende será la que considere en este trabajo. A continuación procedemos a aplicar la **técnica de filtrado** explicada anteriormente para confirmar que esta serie temporal no se caracteriza por una tendencia concreta. Para ello se calculan **las medidas móviles con un número diferente de puntos** con los que generar distintas aproximaciones a la serie temporal original y así intentar establecer alguna similitud con una función matemática. No obstante, tal y como podemos apreciar en las diferentes gráficas, no ha sido posible por lo que **esta serie temporal no tiene una tendencia definida**.

```
for (k in 5:10) {
  # Filtramos la serie calculando las Medias Móviles considerando un conjunto
  # de puntos k
  serie.filter <- rep(1/k, k)
  filtered.serie <- filter(serie, filter=serie.filter,
                           sides=2, method="convolution")
  # Representamos la serie original y la estimación
  series <- matrix(c(t(serie), t(filtered.serie)), ncol=2)
  matplot(series, pch=1, type="l")
}
```


series
576
577
578
579
580
581
582
0
20
40
60
80
series
576
577
578
579
580
581
582
0
20
40
60
80


series
576
577
578
579
580
581
582
0
20
40
60
80
series
576
577
578
579
580
581
582
0
20
40
60
80



Conjuntos de entrenamiento y test

El modelado de una serie temporal se reduce básicamente a la resolución de un problema de clasificación. Por ello, es necesario **dividir el conjunto de datos en entrenamiento y test** con los que construir uno o varios modelos y validar su capacidad de generalización al realizar predicciones futuras. No obstante, al no disponer de ningún tipo de información o contexto sobre el problema de clasificación o la serie temporal, se ha optado por realizar una partición tradicional basada en un **90% de datos para entrenamiento y el 10% restante para test**. Cabe destacar la importancia de **preservar el orden temporal** de los datos de modo que el conjunto de entrenamiento estará compuesto por los primeros 86 datos, mientras que el de test dispondrá de los últimos 10 valores de la serie temporal.

```
# Calculamos el número de instancias que representan el 90% de los datos
perc <- round(length(serie)*0.9)
# 90% de los datos totales para entrenamiento
serie.train <- serie[1:perc]
train.time <- 1:length(serie.train)
```

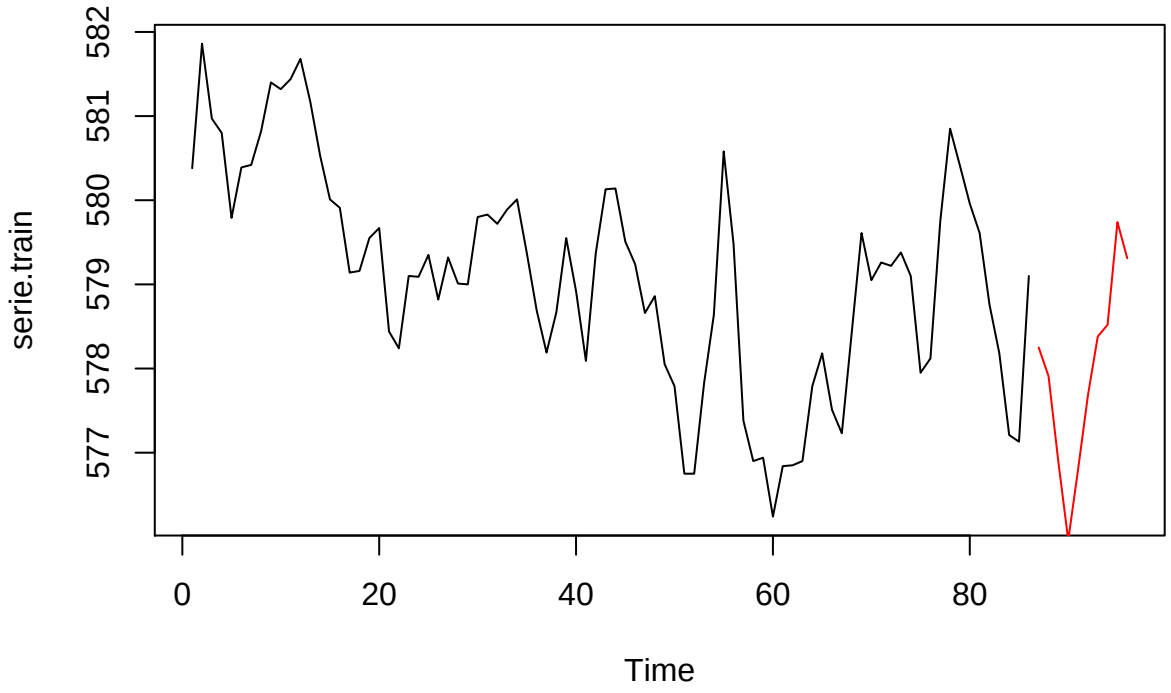
```
# 10% restante de los datos para test
serie.test <- serie[(perc+1):length(serie)]
test.time <- (train.time[length(train.time)]+1) :
             (train.time[length(train.time)]+length(serie.test))
# Comprobamos las dimensiones de sendos conjuntos
length(serie.train)
```

```
## [1] 86
```

```
length(serie.test)
```

```
## [1] 10
```

```
# Representamos la serie temporal indicando el conjunto de entrenamiento y test
plot.ts(serie.train, xlim=c(1, test.time[length(test.time)]))
lines(test.time, serie.test, col="red")
```



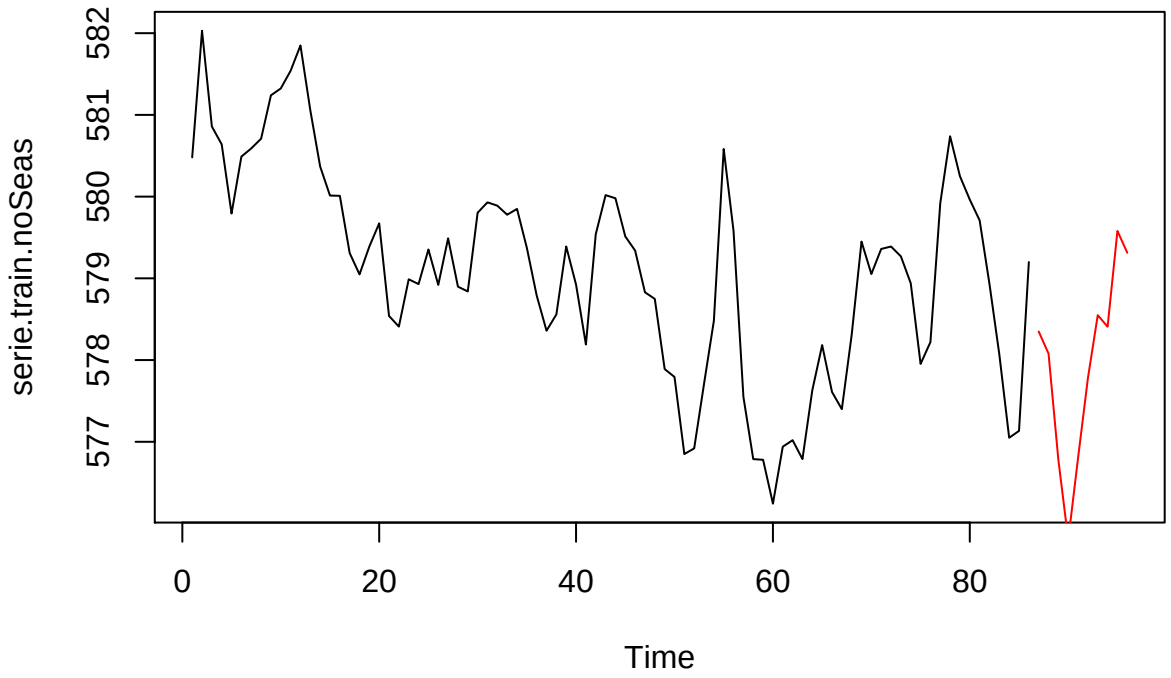
Eliminación de la Estacionalidad

En esta sección el objetivo consiste en **eliminar la componente estacional** de la serie temporal, tanto del conjunto de entrenamiento como del conjunto de test. Para separar cada una de las componentes explicadas en la parte teórica, hacemos uso de la **diferenciación aditiva** que realiza la función `decompose` para luego restar la componente estacional a sendos conjuntos de datos.

```
# Calculamos la estacionalidad asumiendo una frecuencia de 5 días
k <- 5
seasonality <- decompose(serie.ts)$seasonal[1:k]
# Eliminamos la estacionalidad en los conjuntos de entrenamiento y test
aux <- rep(seasonality, length(serie.train)/length(seasonality))
serie.train.noSeas <- serie.train - aux
```

```
## Warning in serie.train - aux: longitud de objeto mayor no es múltiplo de la
## longitud de uno menor
```

```
serie.test.noSeas <- serie.test - seasonality
# Representamos los conjuntos de entrenamiento y test sin la componente estacional
plot.ts(serie.train.noSeas, xlim=c(1, test.time[length(test.time)]))
lines(test.time, serie.test.noSeas, col = "red")
```



Análisis de la Estacionariedad

Una vez disponemos de la serie temporal sin estacionalidad, puesto que no existía tendencia tal y como se demostró en su respectivo análisis, a continuación comenzamos el estudio de la estacionariedad. Para ello aplicamos directamente el **test estadístico de Dickey-Fuller aumentado** estableciendo un **umbral del 95% de confianza**. Como podemos observar en el siguiente *chunk*, el p-valor obtenido es mayor que dicho umbral por lo que no se rechaza la hipótesis nula y como consecuencia **la serie no es estacionaria**.

```
# Primer test de Dickey-Fuller aumentado
adf.test(serie.train.noSeas)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  serie.train.noSeas
## Dickey-Fuller = -2.7686, Lag order = 4, p-value = 0.2603
## alternative hypothesis: stationary
```

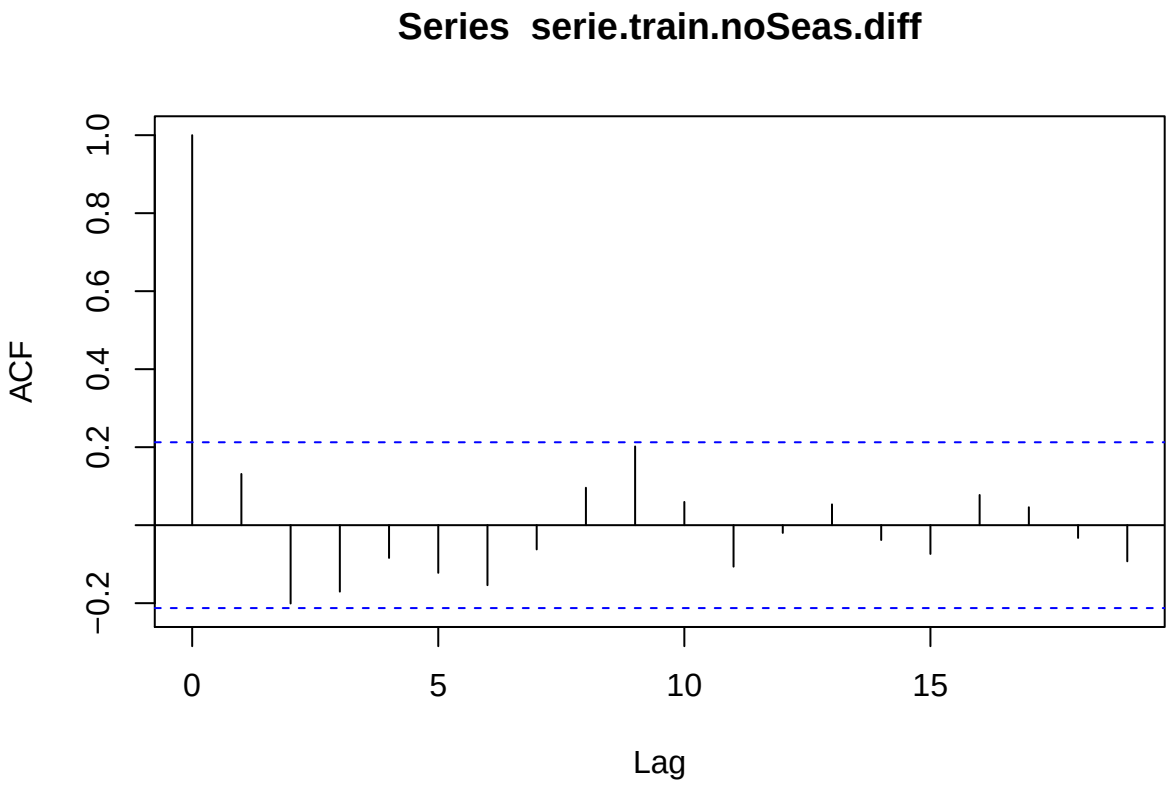
El siguiente paso consiste en **diferenciar la serie**, tanto en el conjunto de entrenamiento como en el de test para intentar convertirla en estacionaria. Al aplicar de nuevo el test sobre la serie diferenciada resultante podemos observar que, en esta ocasión, el p-valor devuelto es menor que el umbral establecido. Por lo tanto, el test estadístico rechaza la hipótesis nula y podemos confirmar que la **serie diferenciada es estacionaria al 95% de confianza**.

```
# Diferenciamos la serie temporal tanto en entrenamiento como en test
serie.train.noSeas.diff <- diff(serie.train.noSeas)
serie.test.noSeas.diff <- diff(serie.test.noSeas)
# Aplicamos por segunda vez el test de Dikcey-Fuller aumentado
adf.test(serie.train.noSeas.diff)
```

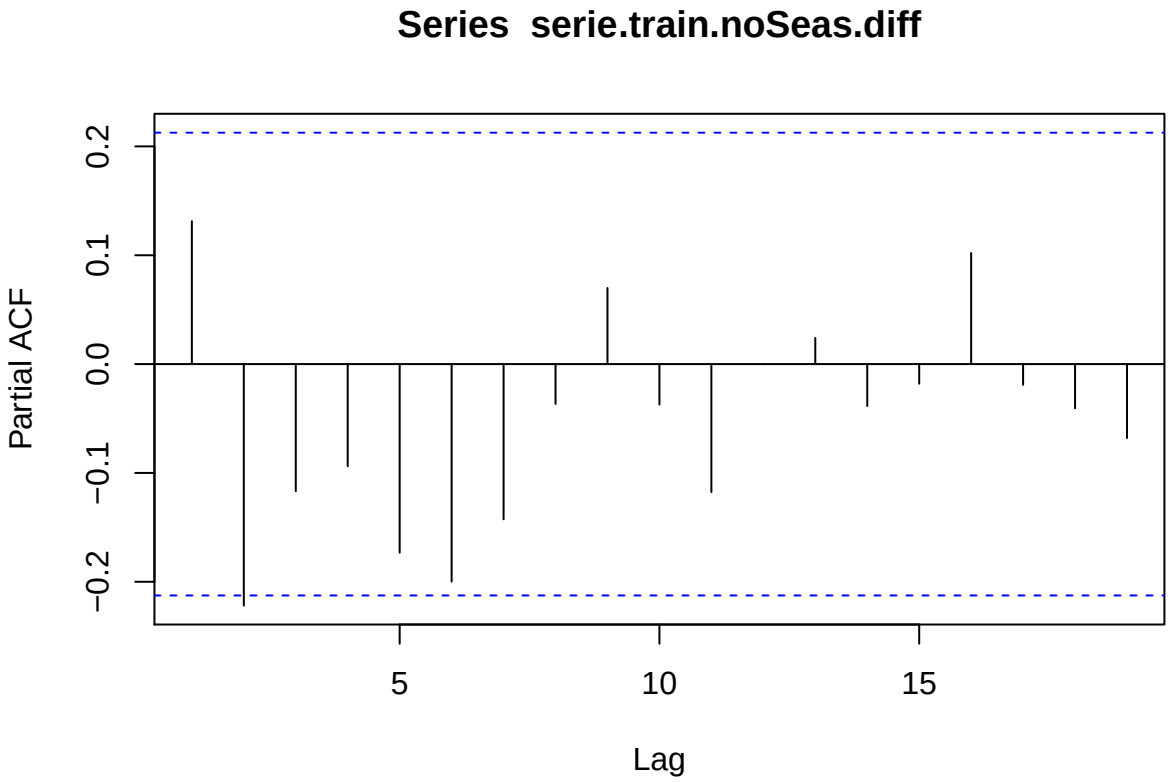
```
## Warning in adf.test(serie.train.noSeas.diff): p-value smaller than printed p-
## value

##
##  Augmented Dickey-Fuller Test
##
## data:  serie.train.noSeas.diff
## Dickey-Fuller = -5.5854, Lag order = 4, p-value = 0.01
## alternative hypothesis: stationary
```

Una vez disponemos de una considerable certeza estadística de que la serie diferenciada es estacionaria, procedemos a visualizar sus respectivos **gráficos ACF y PACF** para comprobar esta característica visualmente. Tal y como se puede observar, en sendas representaciones los **valores de autocorrelación se encuentran dentro de los umbrales** denotados por una línea discontinua azul muy cercana a 0.

```
# Representamos los gráficos ACF y PACFa partir de la serie estacionaria
# sin estacionalidad
acf(serie.train.noSeas.diff)
```




```
pacf(serie.train.noSeas.diff)
```



Modelos ARIMA

En función de las reglas explicadas anteriormente en la parte teórica y a los gráficos ACF y PACF obtenidos, procedemos a generar un clasificador por cada tipo de modelo descrito para posteriormente comparar su eficacia y calidad.

Modelo Autorregresivo

Comenzamos por un **Modelo Autorregresivo de orden 1** puesto que únicamente el dato situado en *lag=2* es el que supera ligeramente el umbral superior en la gráfica PACF. Para finalizar la formulación del modelo **añadimos el grado de las diferencias** que se han realizado sobre la serie temporal para convertirla en estacionaria. El modelo resultante sería un **ARIMA(1, 1, 0)** no estacional aplicado sobre la serie sin estacionalidad. Con el objetivo de comparar su bondad se calculan el **error de ajuste y de test**. Observando sendos valores podemos apreciar que la **capacidad de predicción del modelo no es muy buena**, ya que presenta casi un 40% de instancias mal clasificadas sobre el conjunto de test.

```
# Construimos un primer modelo ARIMA(1, 1, 0)
arima.ar <- arima(serie.train.noSeas, order=c(1, 1, 0))
# Calculamos el error de ajuste sobre el conjunto de entrenamiento
arima.ar.res <- serie.train.noSeas + arima.ar$residuals
# Obtenemos las predicciones sobre el conjunto de test
arima.ar.preds <- predict(arima.ar, n.ahead=length(serie.test.noSeas))$pred
# Calculamos el error cuadrático acumulado en entrenamiento y test
arima.ar.train_error <- sum((arima.ar$residuals)^2)
arima.ar.test_error <- sum((arima.ar.preds - serie.test.noSeas)^2)
# Mostramos los errores calculados de entrenamiento y test
arima.ar.train_error
```

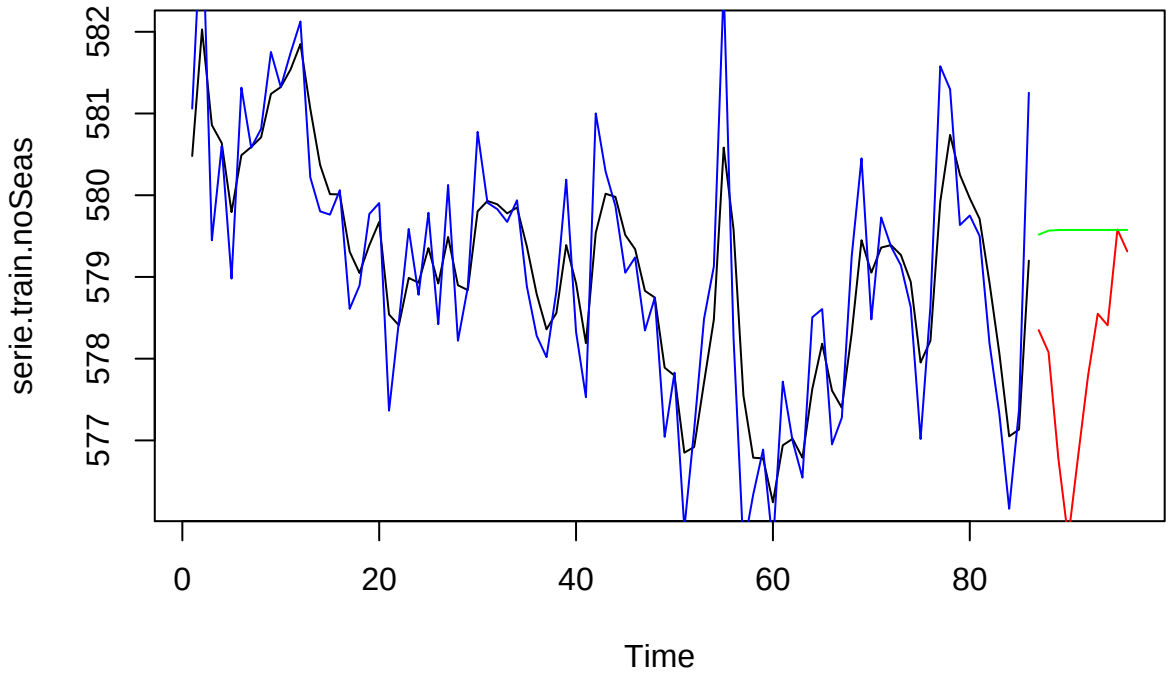
```
## [1] 44.68466
```

```
arima.ar.test_error
```

```
## [1] 39.03548
```

A continuación representamos la serie temporal sin la componente estacional, diferenciando el conjunto de entrenamiento de color azul y el conjunto de test en color rojo. Sobre esta línea podemos observar de color verde que las predicciones realizadas por el modelo apenas son **mínimamente similares a los valores reales** del conjunto de test.

```
# Gráfica de ajuste y predicción sobre test
plot.ts(serie.train.noSeas, xlim=c(1, test.time[length(test.time)]))
lines(arima.ar.res, col="blue")
lines(test.time, serie.test.noSeas, col="red")
lines(test.time, arima.ar.preds, col="green")
```



Procedemos a evaluar la bondad del modelo construido mediante un **análisis de la autocorrelación de sus residuos** verificando que no se encuentran relacionados entre sí y que siguen una distribución normal. Aplicando los tests estadísticos explicados en la parte teórica, podemos observar que todos devuelven un p-valor mayor que el umbral fijado a 0.05, por lo que no se rechazan ninguna de sus hipótesis nulas. Así, podemos determinar que los residuos de este primer modelo **no están correlados entre sí y siguen una distribución normal**.

```
# Test de Box-Pierce para comprobar las correlaciones entre los residuos
Box.test(arima.ar$residuals)
```

```
##
##  Box-Pierce test
##
## data:  arima.ar$residuals
## X-squared = 0.19854, df = 1, p-value = 0.6559
```

```
# Test de Jarque Bera para comprobar la normalidad los residuos del modelo
jarque.bera.test(arima.ar$residuals)
```

```
##
```

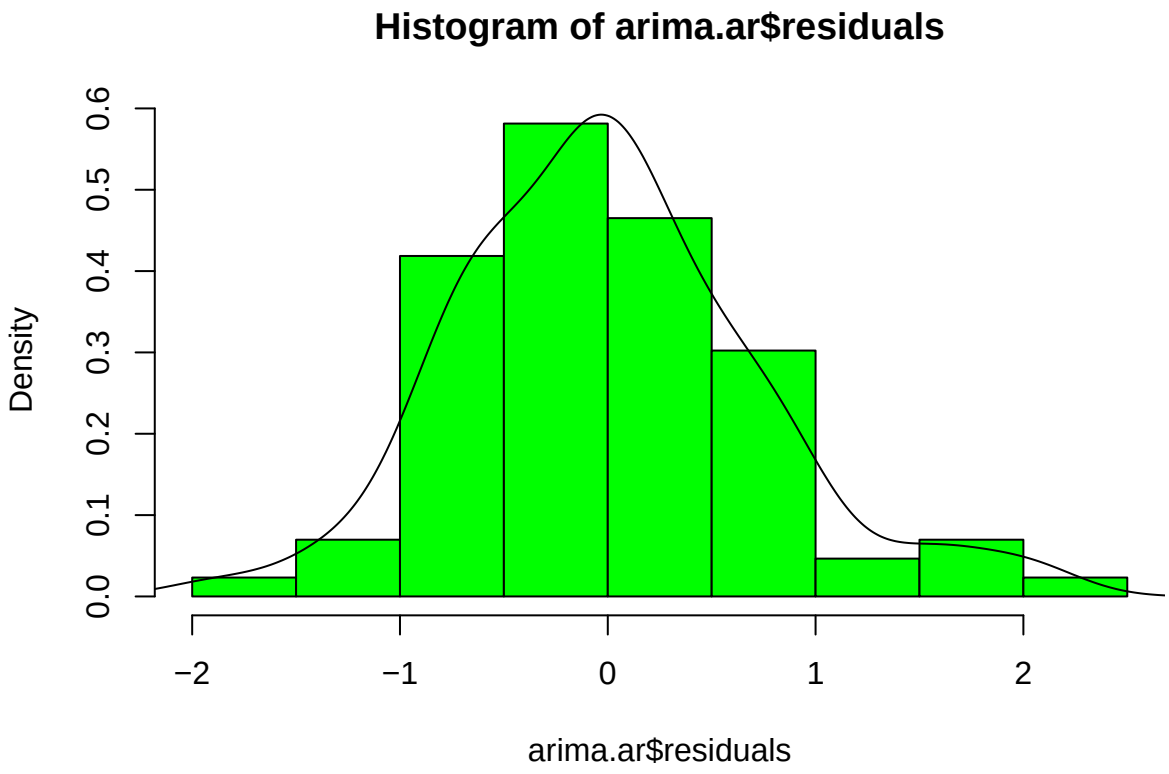
```
##  Jarque Bera Test
##
## data:  arima.ar$residuals
## X-squared = 4.1085, df = 2, p-value = 0.1282
```

```
# Test de Shapiro-Wilk para comprobar la normalidad de los residuos del modelo
shapiro.test(arima.ar$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  arima.ar$residuals
## W = 0.97884, p-value = 0.1705
```

Finalmente representamos un histograma de los residuos del modelo junto con su curva de densidad para afirmar gráficamente los resultados obtenidos de los tests estadísticos. Tal y como se puede apreciar, su distribución es muy similar a una normal con media cero. Por lo tanto, podemos determinar que la ínfima capacidad de generalización del modelo se debe en su totalidad a la **baja calidad y/o cantidad de datos disponibles**.

```
# Histograma y curva de densidad para visualizar los residuos del modelo
hist(arima.ar$residuals, col="green", prob=T)
lines(density(arima.ar$residuals))
```



Modelo de Medias Móviles

En esta sección se pretende construir un **Modelo de Medias Móviles de orden 1** ya que si observamos el gráfico ACF solo existe un único valor por encima del límite superior. Del mismo modo que en el caso anterior, añadimos en la formulación el **grado de las diferencias** realizadas sobre la serie temporal para convertirla en estacionaria. Por lo tanto, el segundo modelo que se pretende construir es un **ARIMA(0, 1, 1)** no estacional aplicado sobre la serie sin estacionalidad. Como en el procedimiento anterior, calculamos el **error de ajuste y sobre test** para luego poder realizar una comparación entre sendos modelos. Según los

resultados visualizados en el siguiente *chunk*, este segundo modelo **tampoco parece disponer de una buena capacidad de predicción**, inclusive puede llegar a ser menor puesto que su error sobre el conjunto de test es superior al Modelo Autorregresivo anterior.

```
# Construimos un modelo ARIMA(0, 1, 1)
arima.ma <- arima(serie.train.noSeas, order=c(0, 1, 1))
# Calculamos el error de ajuste sobre el conjunto de entrenamiento
arima.ma.res <- serie.train.noSeas + arima.ma$residuals
# Obtenemos las predicciones sobre el conjunto de test
arima.ma.preds <- predict(arima.ma, n.ahead=length(serie.test.noSeas))$pred
# Calculamos el error cuadrático acumulado en entrenamiento y test
arima.ma.train_error <- sum((arima.ma$residuals)^2)
arima.ma.test_error <- sum((arima.ma.preds - serie.test.noSeas)^2)
# Mostramos los errores calculados de entrenamiento y test
arima.ma.train_error
```

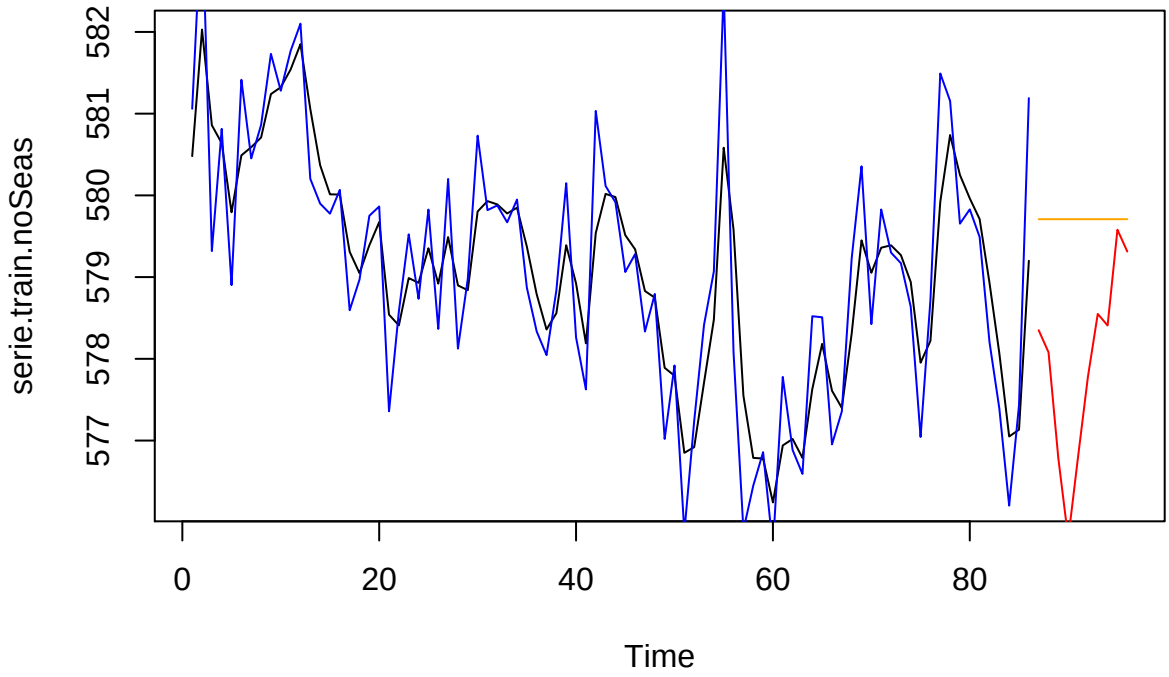
```
## [1] 43.96589
```

```
arima.ma.test_error
```

```
## [1] 43.73956
```

Si repetimos la misma representación del modelo anterior pero considerando las predicciones realizadas por este segundo clasificador basado en Medias Móviles, podemos apreciar que **no se ajustan al conjunto de test** que se intenta predecir, si comparamos las líneas naranja y roja, respectivamente.

```
# Gráfica de ajuste y predicción sobre test
plot.ts(serie.train.noSeas, xlim=c(1, test.time[length(test.time)]))
lines(arima.ma.res, col="blue")
lines(test.time, serie.test.noSeas, col="red")
lines(test.time, arima.ma.preds, col="orange")
```



Evaluamos la calidad del modelo aplicando los mismos tests de correlación y normalidad de residuos. Nuevamente podemos observar que todos los p-valores resultantes son mayores que el umbral fijado en 0.05, por lo que no se rechazan las hipótesis nulas y podemos determinar que los residuos del segundo modelo **no**

están correlados y siguen una distribución normal.

```
# Test de Box-Pierce para comprobar las correlaciones entre los residuos
Box.test(arima.ma$residuals)
```

```
##
##  Box-Pierce test
##
## data:  arima.ma$residuals
## X-squared = 0.077415, df = 1, p-value = 0.7808
```

```
# Test de Jarque Bera para comprobar la normalidad los residuos del modelo
jarque.bera.test(arima.ma$residuals)
```

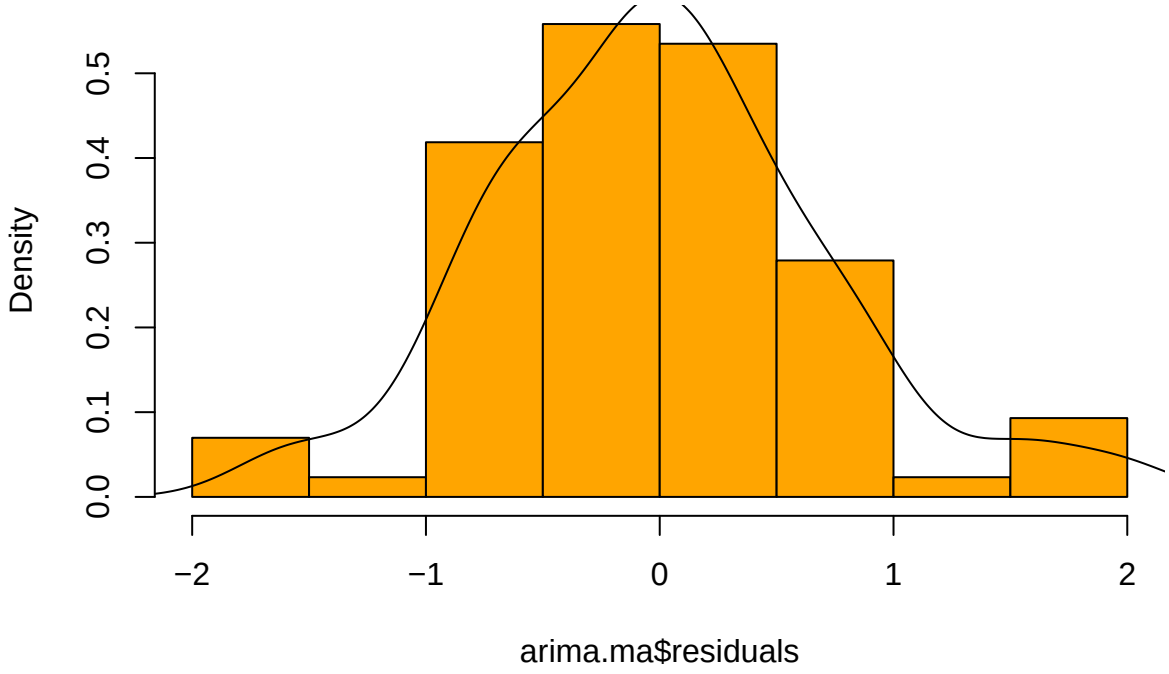
```
##
##  Jarque Bera Test
##
## data:  arima.ma$residuals
## X-squared = 2.7875, df = 2, p-value = 0.2481
```

```
# Test de Shapiro-Wilk para comprobar la normalidad de los residuos del modelo
shapiro.test(arima.ma$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  arima.ma$residuals
## W = 0.98024, p-value = 0.2114
```

Visualizando el histograma y la curva de densidad de los residuos de este clasificador basado en Medias Móviles, podemos apreciar que de nuevo su distribución parece ser una **normal centrada en media cero**. Por lo tanto, parece ser que la razón explicativa de la baja capacidad de predicción de sendos clasificadores se concentra en la **pésima calidad y/o insuficiencia de los datos** disponibles para su entrenamiento y validación.

```
# Histograma y curva de densidad para visualizar los residuos del modelo
hist(arima.ma$residuals, col="orange", prob=T)
lines(density(arima.ma$residuals))
```



Predicciones reales

Tras la construcción, validación y análisis de un Modelo Autorregresivo y otro basado en Medias Móviles, en función de los errores cuadráticos acumulados del ajuste en entrenamiento y test, podemos determinar que el **primer clasificador parece proporcionar resultados algo más certeros**, aunque ambos disponen de una capacidad de predicción bastante inferior. Por lo tanto, las dos predicciones que se pretenden calcular a partir de la serie temporal se realizarán con el Modelo Autorregresivo.

Tal y como se ha explicado en la parte teórica, repetimos el mismo procedimiento anterior para **eliminar la componente estacional y construir un modelo ARIMA(1, 1, 0)** pero a partir de la serie temporal al completo. Tras construir el clasificador, podemos **generar los dos nuevos valores futuros** de la serie volviendo a aplicar la componente estacional eliminada al comienzo de este proceso. En la siguiente gráfica podemos observar en azul el conjunto de datos disponible para ajustar el clasificador y en rojo las dos predicciones realizadas por el modelo.

```
# Eliminamos la estacionalidad de la serie completa
seasonality <- as.numeric(decompose(serie.ts)$seasonal[1:2])
aux <- rep(seasonality, length(serie)/length(seasonality))
serie.noSeas <- serie - aux
# Construimos el Modelo Autorregresivo ARIMA(1, 1, 0)
arima.model <- arima(serie.noSeas, order=c(1, 1, 0))
arima.train.preds <- serie.noSeas + arima.model$residuals
arima.test.preds <- predict(arima.model, n.ahead=2)$pred

# Añadimos la estacionalidad eliminada a los valores ajustados de
# entrenamiento y las predicciones sobre test
arima.train.preds <- arima.train.preds + aux
arima.test.preds <- arima.test.preds + seasonality
# Representamos la serie temporal ajustada con los dos nuevos valores predichos
plot.ts(serie)
```

```
lines(arima.train.preds, col="blue")
lines(arima.test.preds, col="red")
```