Trabajo autónomo I: Series Temporales

Nombre: Francisco Pérez Hernández
E-mail: herpefran92@gmail.com
Asignatura: Series temporales y minería de flujos de datos
Máster: Ciencia de Datos e Ingeniería de Computadores
Fecha de entrega: 19/05/2017

1 Parte teórica

1.1 Serie temporal

Una serie temporal es cualquier magnitud observada a lo largo del tiempo, en un intervalo regular ya sea cada minuto, hora, día, semana, etc. Además, tenemos series estacionarias o no estacionarias, en las que, se dice estacionaria cuando su media y varianza no varía con el tiempo. La estacionaridad indica que las propiedades estadísticas de la serie no varían en el tiempo y, por tanto, los datos pueden estudiarse bajo un mismo modelo paramétrico independiente del tiempo. Además, que sea estacionaria es un requisito para poder aplicar modelos paramétricos de análisis y predicción de series de datos. Las series no estacionarias pueden tener o no tener tendencia o estacionalidad.

1.2 Metodología Box-Jenkins para predicción de series temporales

La metodología Box-Jenkins para predicción de series temporales conlleva:

- Dividir la serie X(t) en tendencia T(t), estacionalidad S(t) y una componente irregular (E(t)). Enfoque aditivo: X(t) = T(t) + S(t) + E(t)
- **Cálculo de la estacionalidad**: valor medio de los valores de la serie dentro del mismo punto estacional.
- **Cálculo de la tendencia**: regresiones, filtrado de la señal, diferenciación, etc.
- **Metodología**:
    - Eliminar tendencia y estacionalidad de la serie.
    - Hacer E(t) estacionaria y aplicar métodos paramétricos.

1.3 Técnicas de modelado de tendencia

La tendencia es el incremento o decremento a largo plazo de los datos. Para modelarla tenemos diferentes técnicas como pueden ser:

- **Estimación funcional**: Aproximar la tendencia de la serie como una función. Requiere realizar hipótesis sobre el modelo que rige la tendencia como los modelos lineales, polinómicos, etc.
- **Filtrado**: Esta opción consiste en aplicar un filtro de medias móviles para estimar la tendencia.
- **Diferenciación**: Se diferencia la señal hasta que desaparezca la tendencia.

1.4 Técnicas de modelado de estacionalidad

La estacionalidad son los datos afectados por un patrón estacional tal como el día del año o el día de la semana. Para modelarla necesitamos:

- Encontrar el periodo de la estacionalidad. Para ello podemos ayudarnos de la ACF para calcular el periodo.
- Se calcula el valor promedio de cada punto de la estación (si es mensual, la media para cada mes; si es semanal, la media para cada día de la semana, etc.).
- Otro método consiste en utilizar la diferenciación, con un desfase d igual al periodo de la estacionalidad: X'(t) = X(t) – X(t-d)
- Existen métodos como la transformada de Fourier o el método de Holts-Winter de alisado exponencial para estacionalidad.

1.5 Proceso para obtener los parámetros de un modelo ARIMA

Un modelo ARIMA está compuesto de 3 componentes p, d y q: ARIMA(p,d,q). Lo primero que tenemos que hacer es saber si el modelo es AR o MA y en qué orden:

- Los modelos AR tienen un ACF que decrece a 0 (con diferentes posibles formas: regulares, sinusoidales, alternando +/-). El número del orden "p" (AR(p)) es tantos valores "distintos de 0 como haya en el PACF".
- Los modelos MA tienen un PACF que decrece a 0 (con diferentes posibles formas: regulares, sinusoidales, alternando +/-). El número del orden "q" (MA(q)) en tantos "valores distintos de 0" como haya en el ACF.
- Un valor se considera "distinto de cero" si no está en el rango (-2/sqrt(N), 2/sqrt(N)), con N=longitud de la serie.

También tenemos que hallar el parámetro d de la siguiente forma:

- Cuando la serie no es estacionaria, el ACF decrece lentamente a 0.
- La parte integrada es necesaria normalmente para corregir la estacionaridad en la varianza.
- Si la serie presenta tendencia lineal, normalmente con d=1 es suficiente. Si la tendencia no es lineal, puede ser necesario usar d>1.
- Si la serie presenta estacionalidad, puede ser necesario un d=periodo de estacionalidad.

2 Parte Práctica

2.1 Pasos seguidos

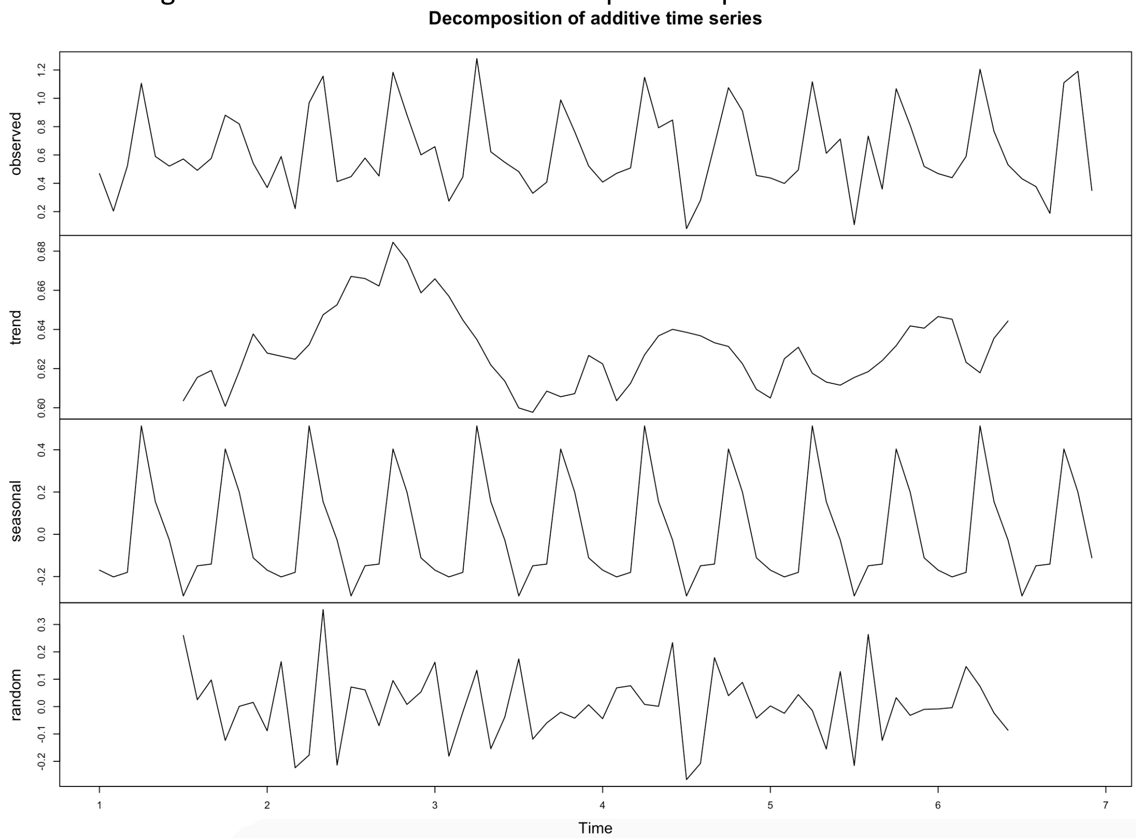
Los pasos que se van a seguir durante esta parte práctica serán:

1. Analizar el problema
2. Pre-procesamiento
3. Eliminación de tendencia
4. Eliminación de estacionalidad
5. Hacer la serie estacionaria
6. Obtención de los parámetros del modelo ARIMA
7. Selección del mejor modelo
8. Realizar predicción

2.2 Necesidad o no de pre-procesamiento

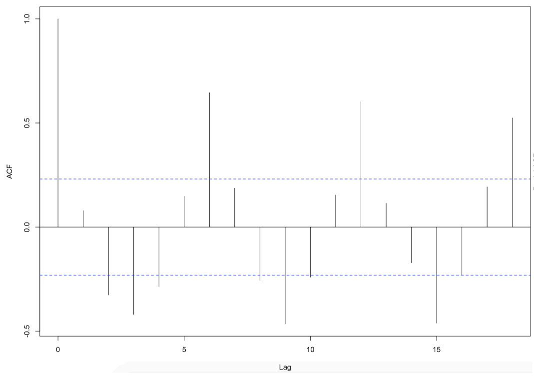
El fichero de datos que vamos a analizar es un fichero en el que tenemos los datos sobre el número de ventas (en miles) de un producto durante 6 años en los conocidos almacenes Guardo-To-Íto, desde enero de 2010 hasta 2015. Lo que queremos realizar, por tanto, es predecir las ventas para el año 2016.

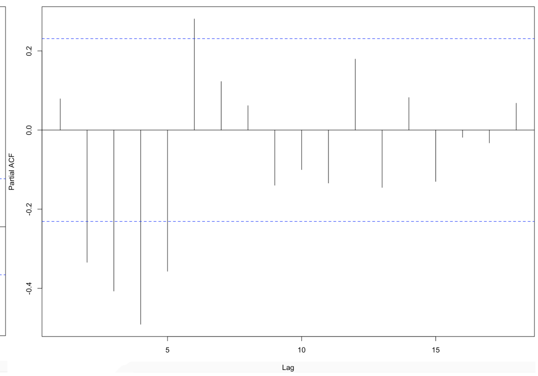
Lo primero que hacemos es cargar la serie como una serie temporal con una frecuencia de 12 ya que estamos tratando las ventas anuales. Pero cuando vemos la primera gráfica, vemos como la frecuencia puede ser de 6 meses por el comportamiento que se visualiza. Seguidamente hacemos una descomposición para ver más información:



Podemos ver 3 aspectos: 1 La variabilidad de la estacionalidad no aumenta ni decrece. 2 Puede que no haya tendencia en la serie. 3 Por la gráfica de estacionalidad, se puede ver que habrá estacionalidad en la serie.

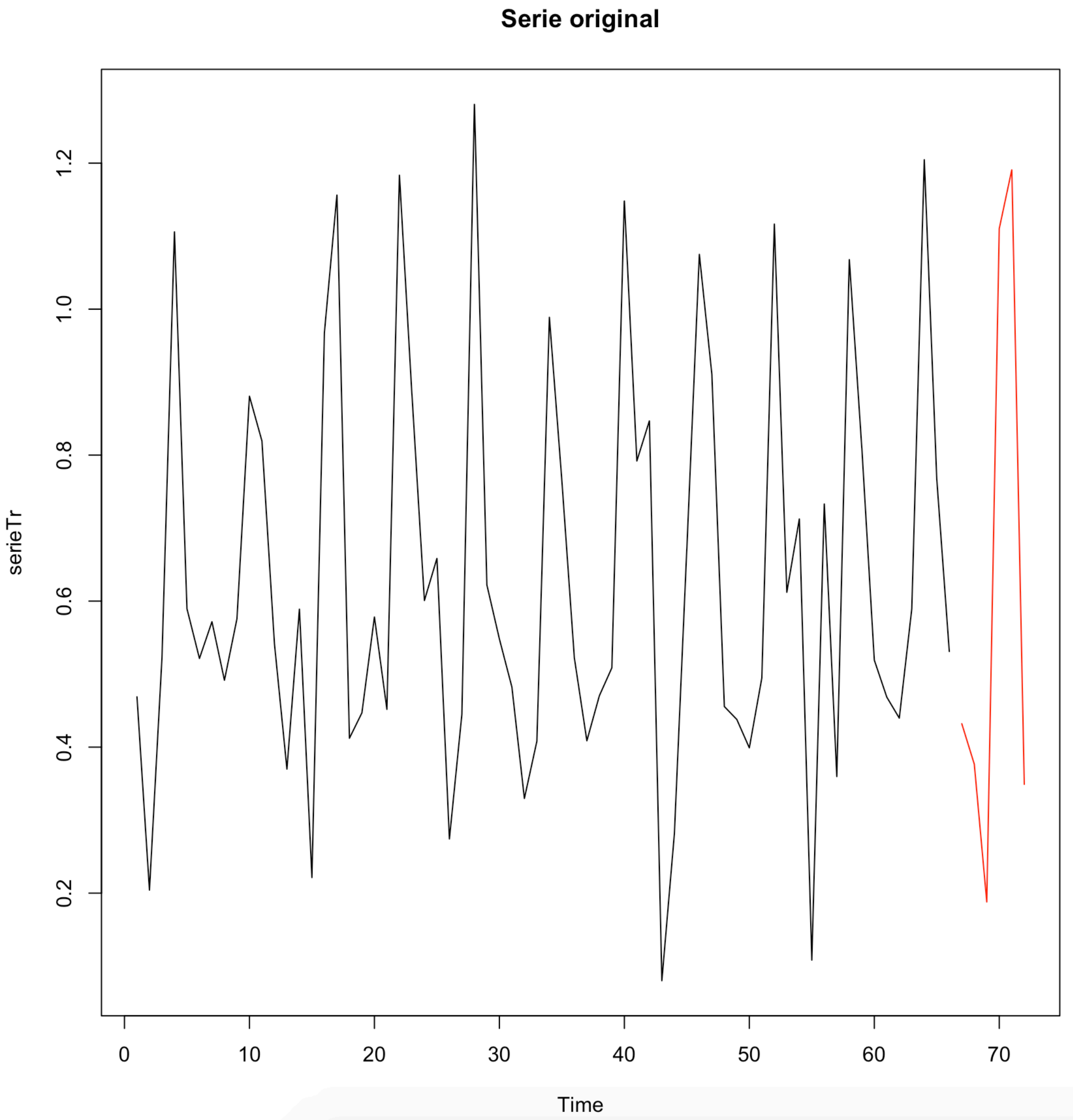
Mostramos además el gráfico ACF y PACF:





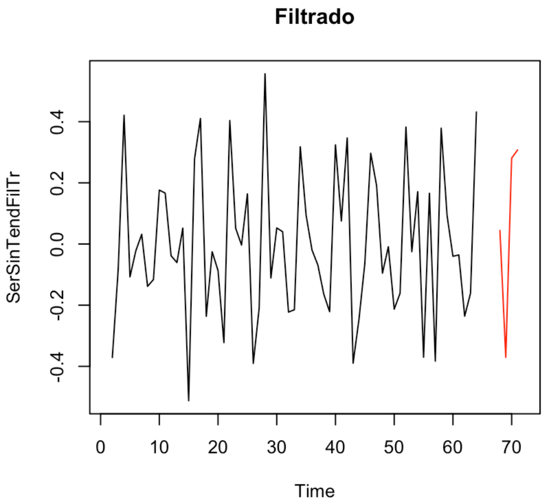
Si además, hacemos el test de Dickey-Fuller aumentado, nos da un valor de p-value = 0.01 por lo que podemos decir que la serie es Estacionaria con un nivel de confianza del 99%. Además, en el gráfico ACF vemos que el Lag es 6.

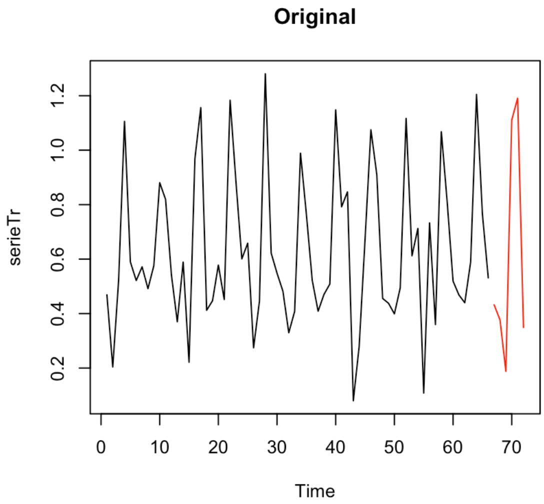
Vamos a dividir la serie original en train y test, dejando un total 6 elementos, para la parte de test.

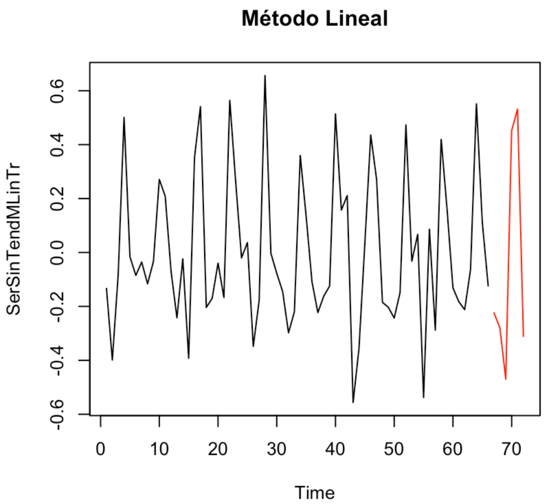


2.3 Necesidad o no de eliminación de tendencia

Para modelar la tendencia he decidido realizar dos modelos, el lineal y el filtrado.
El modelo lineal nos ha dado una suma de residuos RSS igual a 5.132. Además, pasa los test de jarque.bera y el t.test, por lo que podemos decir que la aproximación lineal es factible.
El modelo basado en filtrado, se ha quedado con unos valores de k para train de 4 y de 3 para test.
Los resultados de la serie sin tendencia serían:



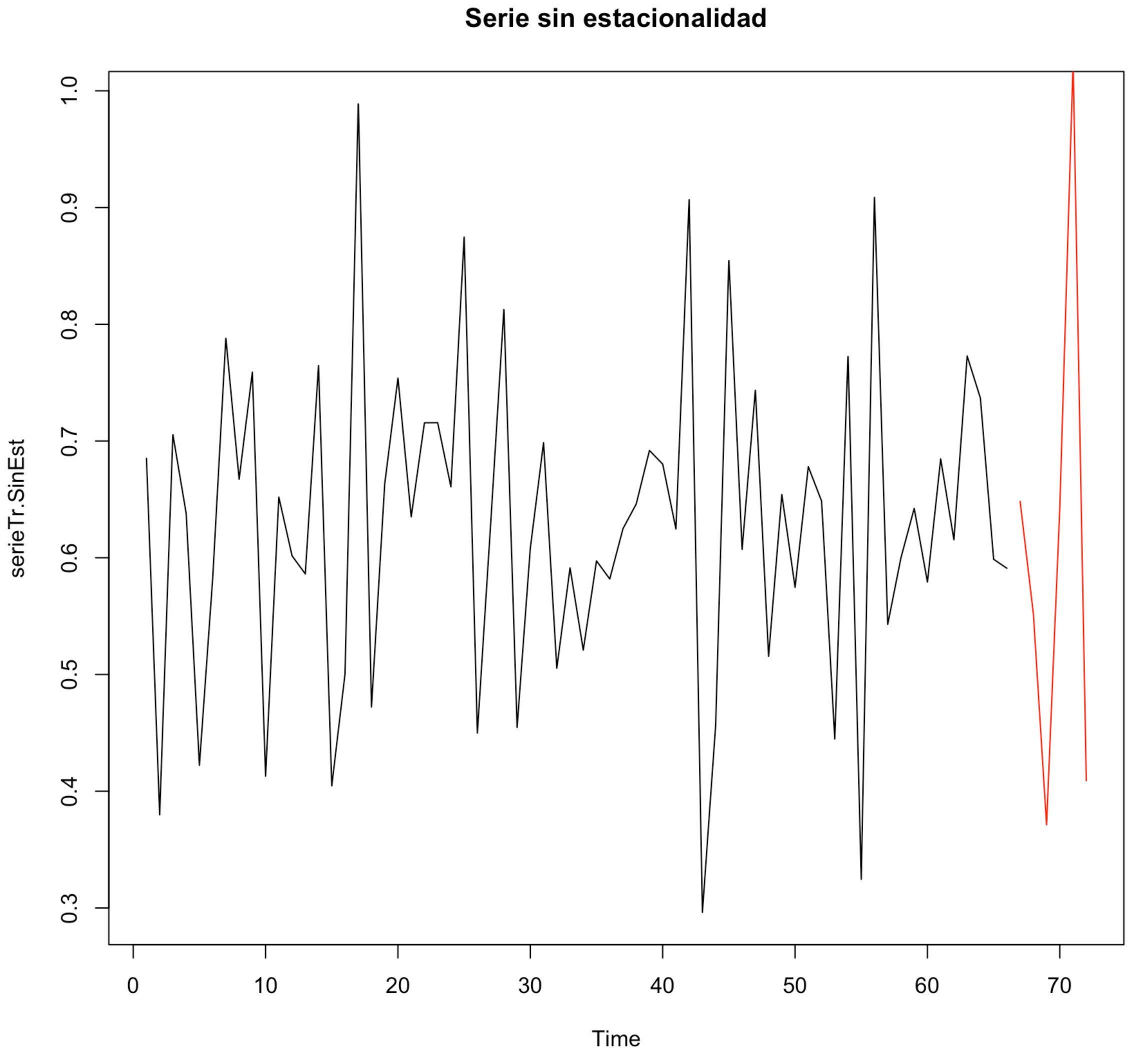




En base a estos resultados, he decidido no eliminar tendencia en la serie, ya que como habíamos dicho en el pre-procesamiento, la serie no presenta tendencia.

2.4 Necesidad o no de eliminación de estacionalidad

Como habíamos visto en el apartado 2.2, la serie presenta una estacionalidad que se repite cada 6 meses. Primero habíamos pensado en 12 al ser una serie de datos anual, pero hemos visto en las gráficas que se trataba de una de 6 meses. Por lo tanto, si eliminamos estacionalidad, sabiendo el periodo, quedando la serie de la siguiente forma:



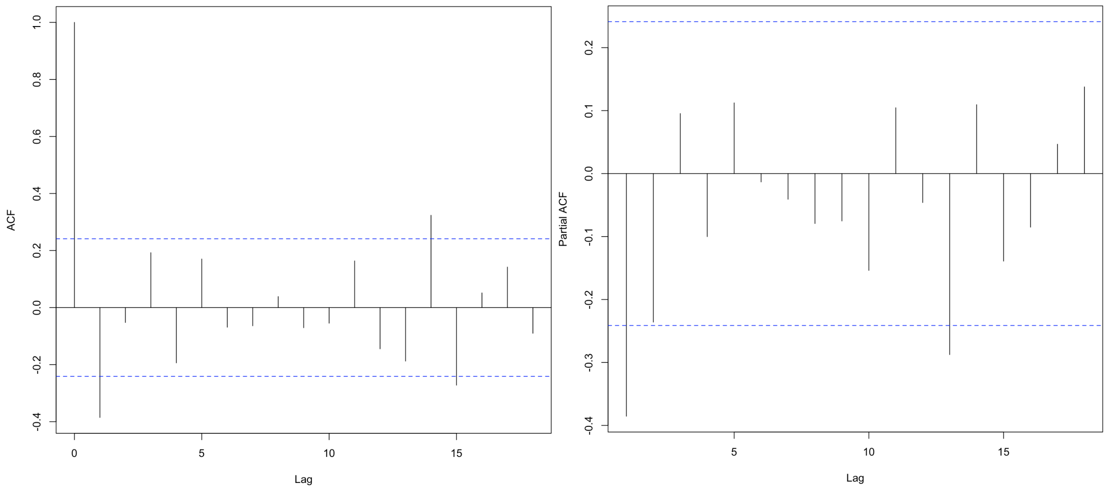
2.5 Necesidad o no de hacer la serie estacionaria

Para saber si la serie es estacionaria o no, como ya hemos visto en el apartado 2.2, realizamos el test de Dickey-Fuller aumentado, con el que hemos visto con un 96% de confianza, que la serie presenta estacionariedad. Por lo tanto, como ya sabemos que es estacionaria, no es necesario diferenciarla.

Por lo tanto, vamos a pasar a realizar nuestro modelo ARIMA.

2.6 Obtención de los parámetros del modelo ARIMA

Veamos para comenzar los gráficos ACF y PACF:



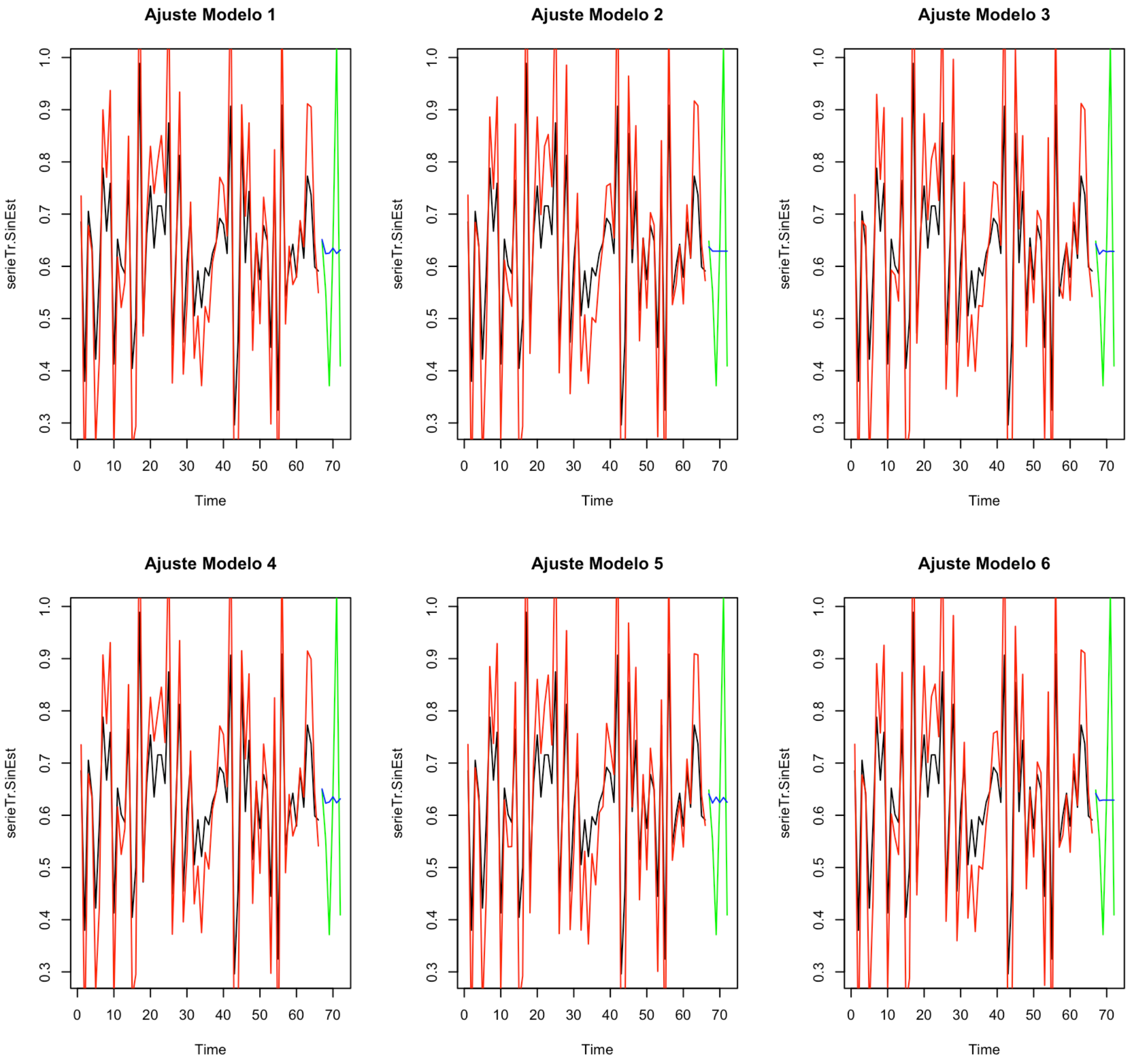
Viendo estos gráficos podemos decidir si tenemos modelos AR o MA y en qué orden. De forma que, para tener un AR, el gráfico ACF tiene que decrecer a 0, condición que se cumple. Además, su orden “p” será 1, ya que el gráfico PACF tiene un valor distinto a 0. Para tener un MA, el gráfico PACF decrece a 0, con un orden “q” igual a 2 ya que tiene 2 valores distintos a 0 en el ACF.

Ya solo falta el valor “d”, pero como no hemos realizado diferenciación, su valor será 0. Por lo tanto vamos a probar con los modelos ARIMA(2,0,2), ARIMA(0,0,1), ARIMA(1,0,0), ARIMA(2,0,1), ARIMA(1,0,2) y ARIMA(1,0,1) para poder tener una comparación amplia.

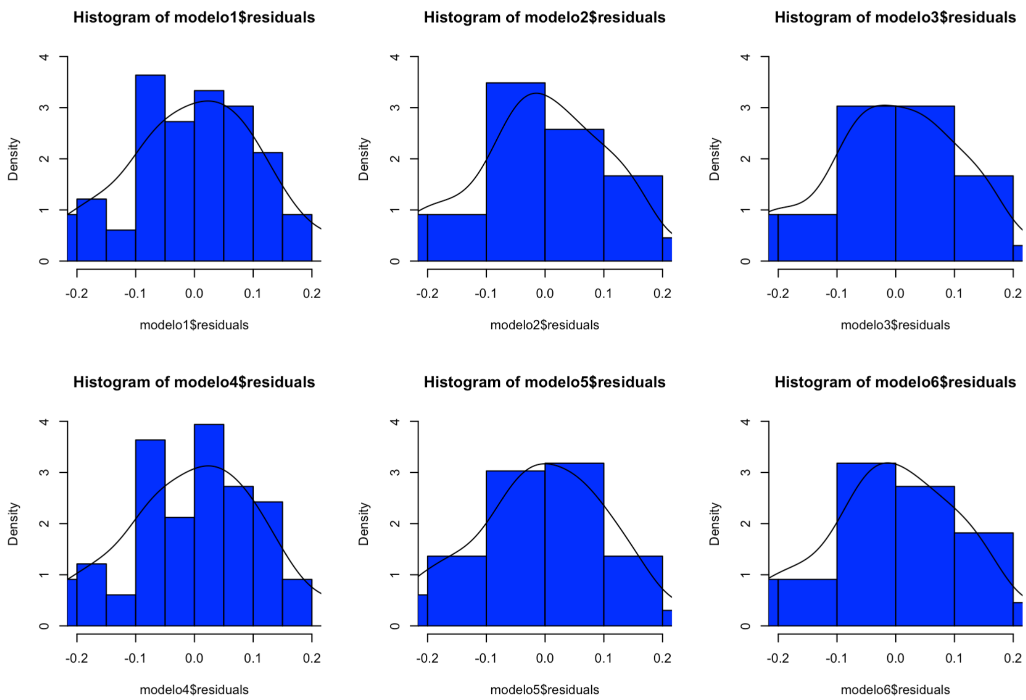
2.7 Selección del mejor modelo

Para cada uno de los 6 modelos que voy a comparar, le calculo los errores de ajuste en train, los errores de predicción de test, mostrando sus gráficas de ajuste y predicción en test. Además, realizamos los test Box-Pierce, Jarque Bera y el Shapiro-Wilk. Estos test son pasados por los 6 modelos en todos los test. Además, se muestra los histogramas de los residuos.

Si mostramos las gráficas de los 6 modelos tenemos:



Además, podemos ver los histogramas de los residuos de los 6 modelos:



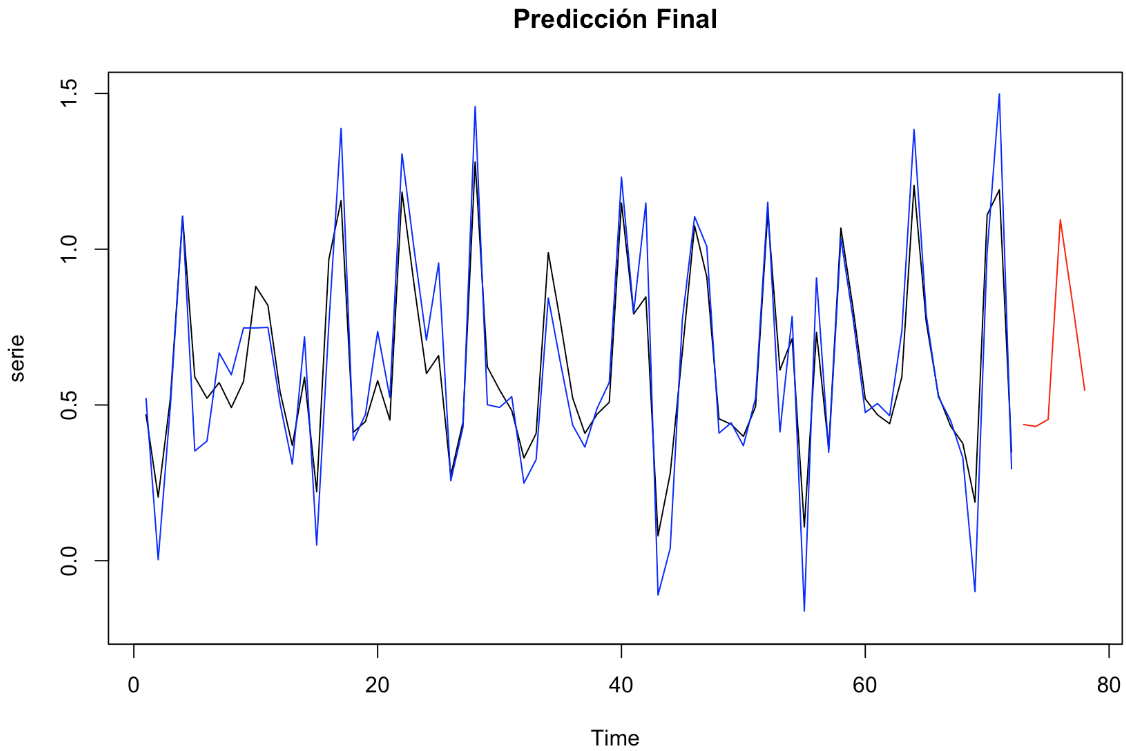
Para seleccionar el mejor modelo, nos vamos a basar en el criterio de Akaike (AIC), ya que los 3 test, que hemos comentado anteriormente, han pasado los resultados satisfactoriamente.
El resultado del criterio de Akaike es:

Modelo	df	AIC
ARIMA(2,0,2)	6	- 78.15619
ARIMA(0,0,1)	3	- 80.80363
ARIMA(1,0,0)	3	- 78.55963
ARIMA(2,0,1)	5	- 80.07062
ARIMA(1,0,2)	5	- 77.84187
ARIMA(1,0,1)	4	- 78.93667

Por lo que nos vamos a quedar con el modelo ARIMA(0,0,1) al presentar menos errores, con un valor de -80,80.

2.8 Obtención de los valores predichos para la serie

Para predecir la serie lo primero que vamos a hacer es aplicar lo hecho durante la práctica, es decir, vamos a quitarle la estacionalidad, vamos a aplicar el modelo ARIMA(0,0,1) y vamos a predecir las siguientes 6 resultados de estas ventas de las que trata el conjunto de datos. Seguidamente deshacemos los cambios e imprimimos la serie final con su predicción, quedando:



En esta gráfica vemos, en negro la serie original, en azul los valores ajustados y en rojo, los valores predichos para los siguientes 6 meses de esta serie.

*El código estará adjunto en la entrega de la práctica además de en el anexo 1.

Anexo 1: Código en R

```
#########################################
# 2.2 Pre-procesameinto
#########################################
rm(list=ls()) # Eliminamos lo que haya en el espacio de trabajo
library("tseries") # para el test ADF

#Leemos la serie
serie<-scan("SerieTrabajoPractico.dat")
# Echamos un primer vistazo a la serie.
# Como se trata de series anuales, de haber estacionalidad, posiblemente aparezca cada 12 meses.
# Por tanto, inicialmente creamos la serie temporal con periodo de estacionalidad 12, para visualizar
serie.ts<- ts(serie, frequency = 12)
# Visualizamos la descomposición
plot(decompose(serie.ts))
print("Vemos como la frecuencia no es de 12 meses sinó cada 6 meses, así que vamos a modificar esta serie")
serie.ts <- ts(serie, frequency = 6)
cat("Visualizamos primero la descomposición de la serie, buscando patrones visuales que nos den idea de por dónde empezar (tendencia, estacionalidad)\n");
cat("Vemos 3 cosas: \n");
cat("  1. Que la variabilidad de la estacionalidad no aumenta ni decrece. ");
cat("  2. Que puede que la tendencia no juegue un papel importante.\n")
cat("  3. Que posiblemente hay una estacionalidad.\n")
print("Como con esto no estamos seguros de si la serie es estacionaria o no, vamos a ver el ACF, PACF y pasaremos el test ADF:")
print("Mostrando el ACF")
acf(serie) # Mostramos ACF
print("Mostrando el PACF")
pacf(serie) # Mostramos PACF
resul <- adf.test(serie) # Pasamos el test de ADF
cat("El resultado del test de Dickey-Fuller aumentado es un p-value=", resul$p.value, "\n")
print("Podemos decir debido a este resultado, que la serie es Estacionaria con un nivel de confianza del 99%. Además en el gráfico ACF vemos como con un Lag = 6, que la serie va a ser Estacionaria")

# Dividimos la serie en training y test (nos quedamos con los NTest últimos para el test)
NPred= 6; # Valores a predecir
NTest= 6; # Valores que vamos a dejar para test
serieTr<- serie[1:(length(serie)-NTest)];
tiempoTr<- 1:length(serieTr)
serieTs<- serie[(length(serie)-NTest+1):length(serie)];
tiempoTs<-
(tiempoTr[length(tiempoTr)]+1):(tiempoTr[length(tiempoTr)]+NTest);
plot.ts(serieTr, xlim=c(1,
tiempoTs[length(tiempoTs)]))
title("Serie original")
lines(tiempoTs, serieTs, col="red")
cat("Dividimos el conjunto de datos en entrenamiento (para ajuste, negro) y test (para comprobar los modelos, rojo).\n");

#########################################
#2.3 Tendencia
#########################################
# Modelado y eliminación de tendencia
# Hipótesis: Modelo Lineal x= a + b*t (x=serie; t=tiempo; a,b=parámetros a estimar)
parametros.MLineal <- lm (serieTr ~ tiempoTr) # Ajustamos modelo
# Calculamos la estimación de la tendencia
TendEstimadaTr.MLineal<-
parametros.MLineal$coefficients[1]+tiempoTr*parametros.MLineal$coefficients[2]
TendEstimadaTs.MLineal<-
parametros.MLineal$coefficients[1]+tiempoTs*parametros.MLineal$coefficients[2]
# Mostramos en la misma figura la serie y la tendencia estimada
plot.ts(serieTr, xlim=c(1,
tiempoTs[length(tiempoTs)]))
title("Modelo lineal")
lines(tiempoTr, TendEstimadaTr.MLineal, col="blue")
lines(tiempoTs, serieTs, col="red")
lines(tiempoTs, TendEstimadaTs.MLineal,
col="green")
RSS.tendencia.MLineal= sum(
(parametros.MLineal$residuals)^2); # Calculamos suma de errores al cuadrado
print("Metodo de eliminacion de tendencia por aproximacion lineal.")
cat(c("El modelo presenta un RSS (Residual Sum of Squares)=", RSS.tendencia.MLineal, "\n"));
JB <- jarque.bera.test(parametros.MLineal$residuals)
JB$p.value
JB <- jarque.bera.test((TendEstimadaTs.MLineal-serieTs))
JB$p.value
TT <- t.test(c(parametros.MLineal$residuals,
TendEstimadaTs.MLineal-serieTs))
TT$p.value
print("Viendo estos resultados, podemos pensar, que al tener todos un p-value > 0.05, no existen diferencias significativas en los datos y la hipótesis lineal es factible")

print("Si observamos los resultados, no parece que sea el modelo más correcto para este caso, por lo que voy a probar con un flitrado")
# Estimación de tendencia por filtrado de medias moviles de orden k
for (k in 1:4) {
 filtro<-rep(1/k, k); # Creamos el filtro
 # Filtramos señal
 SerFiltradaTr<-
filter(serieTr,filter=filtro,sides=2,method="convolution")
 # Mostramos en la misma figura la serie y la tendencia estimada
```

```
  series<-matrix(c(t(serieTr), t(SerFiltradaTr)),
ncol=2);
  matplot(series, pch=1, type= "l")
  title("Método Filtrado train")
  cat("Calculo tendencia con filtro de orden k=", k,
"\n")
  print("Pulse una tecla para continuar...")
  pause<-readline(); # para pausar la ejecución
}
for (k in 1:3) {
  filtro<-rep(1/k, k); # Creamos el filtro
  # Filtramos señal
  SerFiltradaTs<-
filter(serieTs,filter=filtro,sides=2,method="convoluti
on")
  # Mostramos en la misma figura la serie y la
tendencia estimada
  series<-matrix(c(t(serieTs), t(SerFiltradaTs)),
ncol=2);
  matplot(series, pch=1, type= "l")
  title("Método Filtrado test")
  cat("Calculo tendencia con filtro de orden k=", k,
"\n")
  print("Pulse una tecla para continuar...")
  pause<-readline(); # para pausar la ejecución
}
print("Vemos que, a mayor k, más se suaviza la serie.
Vamos a quedarnos con el ultimo valor.")

# Eliminamos la tendencia
SerSinTendMLinTr <- serieTr -
TendEstimadaTr.MLineal
SerSinTendMlinTs <- serieTs -
TendEstimadaTs.MLineal
SerSinTendFilTr<-serieTr-SerFiltradaTr
SerSinTendFilTs<-serieTs-SerFiltradaTs
par(mfrow=c(2,2))
plot.ts(SerSinTendFilTr, xlim=c(1,
tiempoTs[length(tiempoTs)]))
title("Filtrado")
lines(tiempoTs, SerSinTendFilTs, col="red")
plot.ts(serieTr, xlim=c(1,
tiempoTs[length(tiempoTs)]))
title("Original")
lines(tiempoTs, serieTs, col="red")
plot.ts(SerSinTendMLinTr, xlim=c(1,
tiempoTs[length(tiempoTs)]))
title("Método Lineal")
lines(tiempoTs, SerSinTendMlinTs, col="red")
par(mfrow=c(1,1))
print("Vemos que la eliminación de tendencia es una
posibilidad, pero vamos a pasar al siguiente paso sin
eliminarla, ya que puedo asumir que no es
elemental")


########################################
#2.4 Estacionalidad
########################################
# Calculamos y eliminamos la estacionalidad
k<- 6; # Asumimos periodo de estacionalidad k= 6
estacionalidad<- decompose(serie.ts)$seasonal[1:k];
#Eliminamos estacionalidad para el modelo ya que
vemos que hay
aux<-rep(estacionalidad,
length(serieTr)/length(estacionalidad));
serieTr.SinEst<- serieTr-aux;
serieTs.SinEst<- serieTs-estacionalidad;
plot.ts(serieTr.SinEst, xlim=c(1,
tiempoTs[length(tiempoTs)]))
title("Serie sin estacionalidad")
lines(tiempoTs, serieTs.SinEst, col="red")
print("Serie sin la estacionalidad\n")

########################################
#2.5 Estacionaridad
########################################
#Vamos a ver si es estacionaria con el Test de
Dickey-Fuller aumentado
adftest<- adf.test(serieTr.SinEst);
cat(c("Resultados del test ADF (aproximación lineal
sin tendencia ni estacionalidad): ", adftest$p.value,
"\n"));
cat("No es necesario diferenciar.\n")


########################################
#2.6 Modelos ARIMA
########################################
# Vemos los ACF y PACF
acf(serieTr.SinEst) # Mostramos ACF
print("Mostrando el ACF de la serie sin tendencia
lineal y sin estacionalidad\n")
pacf(serieTr.SinEst) # Mostramos PACF
print("Mostrando el PACF de la serie sin tendencia
lineal y sin estacionalidad")

# Mostramos las series sin estacionalidad
plot.ts(serieTr.SinEst, xlim=c(1,
tiempoTs[length(tiempoTs)]))
title("Serie sin estacionalidad")
lines(tiempoTs[1:(length(tiempoTs))], serieTs.SinEst,
col="red")
print("Serie sin la estacionalidad.")
cat("Con esto, podemos comenzar probando un
modelo ARIMA(1, 0, 2):\n");

########################################
#2.6.1 Modelo ARIMA
########################################
# Ajustamos el modelo ARIMA(2,0,2)
modelo1<- arima(serieTr.SinEst, order=c(2, 0, 2))
# Cogemos los valores del ajuste y las predicciones
# Cogemos los valores que se han ajustado de la
serie
valoresAjustados1<-
serieTr.SinEst+modelo1$residuals;
# Calculamos las predicciones
Predicciones1<- predict(modelo1, n.ahead = NPred);
valoresPredichos1<- Predicciones1$pred; # Cogemos
las predicciones
# Calculamos el error cuadrático acumulado del
ajuste, en ajuste y en test
errorTr1<- sum((modelo1$residuals)^2);
```

```
errorTs1<- sum((valoresPredichos1-
serieTs.SinEst)^2);
cat("Error en ajuste con ARIMA(2, 0, 2): ", errorTr1,
"\n")
cat("Error en la predicción de test con ARIMA(2, 0,
2): ", errorTs1, "\n")

# Mostramos las gráficas del ajuste y predicción en
test
plot.ts(serieTr.SinEst, xlim=c(1,
tiempoTs[length(tiempoTs)]))
title("Ajuste y predicción")
lines(valoresAjustados1, col="blue")
lines(tiempoTs, serieTs.SinEst, col="red")
lines(tiempoTs, valoresPredichos1, col="blue")
cat("Predicción con el modelo\n");

# Tests para la selección del modelo y su validación
boxtest1<- Box.test(modelo1$residuals) # Test de
aleatoriedad de Box-Pierce
cat(c("El test de Box-Pierce da un p-value=",
boxtest1$p.value, " para el modelo. Lo pasa (los
errores son aleatorios)\n"))
JB1<- jarque.bera.test(modelo1$residuals); # Test de
normalidad de Jarque Bera
cat(c("El test de Jarque Bera da un p-value=",
JB1$p.value, " para el modelo. Lo pasa (p-
value>0.05)\n"))
SW1<- shapiro.test(modelo1$residuals); # Test de
normalidad de Shapiro-Wilk
cat(c("El test de Shapiro-Wilk da un p-value=",
SW1$p.value, " para el modelo. Lo pasa (p-
value>0.05)\n"))

# Mostramos histograma de residuos
cat("Mostramos el histograma de los residuos del
modelo.\n")
hist(modelo1$residuals, col="blue",
prob=T,ylim=c(0,20),xlim=c(-0.2,0.2))
lines(density(modelo1$residuals))

########################################
#2.6.2 Modelos ARIMA
########################################
# Ajustamos el modelo ARIMA(0,0,1)
modelo2<- arima(serieTr.SinEst, order=c(0, 0, 1))
# Cogemos los valores del ajuste y las predicciones
# Cogemos los valores que se han ajustado de la
serie
valoresAjustados2<-
serieTr.SinEst+modelo2$residuals;
# Calculamos las predicciones
Predicciones2<- predict(modelo2, n.ahead = NPred);
valoresPredichos2<- Predicciones2$pred; # Cogemos
las predicciones
# Calculamos el error cuadrático acumulado del
ajuste, en ajuste y en test
errorTr2<- sum((modelo2$residuals)^2);
errorTs2<- sum((valoresPredichos2-
serieTs.SinEst)^2);
cat("Error en ajuste con ARIMA(0, 0, 1): ", errorTr2,
"\n")
cat("Error en la predicción de test con ARIMA(0, 0,
1): ", errorTs2, "\n")

# Mostramos las gráficas del ajuste y predicción en
test
plot.ts(serieTr.SinEst, xlim=c(1,
tiempoTs[length(tiempoTs)]))
title("Ajuste y predicción")
lines(valoresAjustados2, col="blue")
lines(tiempoTs, serieTs.SinEst, col="red")
lines(tiempoTs, valoresPredichos2, col="blue")
cat("Predicción con el modelo\n");

# Tests para la selección del modelo y su validación
boxtest2<- Box.test(modelo2$residuals) # Test de
aleatoriedad de Box-Pierce
cat(c("El test de Box-Pierce da un p-value=",
boxtest2$p.value, " para el modelo. Lo pasa (los
errores son aleatorios)\n"))
JB2<- jarque.bera.test(modelo2$residuals); # Test de
normalidad de Jarque Bera
cat(c("El test de Jarque Bera da un p-value=",
JB2$p.value, " para el modelo. Lo pasa (p-
value>0.05)\n"))
SW2<- shapiro.test(modelo2$residuals); # Test de
normalidad de Shapiro-Wilk
cat(c("El test de Shapiro-Wilk da un p-value=",
SW2$p.value, " para el modelo. Lo pasa (p-
value>0.05)\n"))

# Mostramos histograma de residuos
cat("Mostramos el histograma de los residuos del
modelo.\n")
hist(modelo2$residuals, col="blue",
prob=T,ylim=c(0,20),xlim=c(-0.2,0.2))
lines(density(modelo2$residuals))

########################################
#2.6.3 Modelos ARIMA
########################################
# Ajustamos el modelo ARIMA(1,0,0)
modelo3<- arima(serieTr.SinEst, order=c(1, 0, 0))
# Cogemos los valores del ajuste y las predicciones
# Cogemos los valores que se han ajustado de la
serie
valoresAjustados3<-
serieTr.SinEst+modelo3$residuals;
# Calculamos las predicciones
Predicciones3<- predict(modelo3, n.ahead = NPred);
valoresPredichos3<- Predicciones3$pred; # Cogemos
las predicciones
# Calculamos el error cuadrático acumulado del
ajuste, en ajuste y en test
errorTr3<- sum((modelo3$residuals)^2);
errorTs3<- sum((valoresPredichos3-
serieTs.SinEst)^2);
cat("Error en ajuste con ARIMA(1, 0, 0): ", errorTr3,
"\n")
cat("Error en la predicción de test con ARIMA(1, 0,
0): ", errorTs3, "\n")

# Mostramos las gráficas del ajuste y predicción en
test
```

```
plot.ts(serieTr.SinEst, xlim=c(1,
tiempoTs[length(tiempoTs)]))
title("Ajuste y predicción")
lines(valoresAjustados3, col="blue")
lines(tiempoTs, serieTs.SinEst, col="red")
lines(tiempoTs, valoresPredichos3, col="blue")
cat("Predicción con el modelo\n");

# Tests para la selección del modelo y su validación
boxtest3<- Box.test(modelo3$residuals) # Test de
aleatoriedad de Box-Pierce
cat(c("El test de Box-Pierce da un p-value=",
boxtest3$p.value, " para el modelo. Lo pasa (los
errores son aleatorios)\n"))
JB3<- jarque.bera.test(modelo3$residuals); # Test de
normalidad de Jarque Bera
cat(c("El test de Jarque Bera da un p-value=",
JB3$p.value, " para el modelo. Lo pasa (p-
value>0.05)\n"))
SW3<- shapiro.test(modelo3$residuals); # Test de
normalidad de Shapiro-Wilk
cat(c("El test de Shapiro-Wilk da un p-value=",
SW3$p.value, " para el modelo. Lo pasa (p-
value>0.05)\n"))

# Mostramos histograma de residuos
cat("Mostramos el histograma de los residuos del
modelo.\n")
hist(modelo3$residuals, col="blue",
prob=T,ylim=c(0,20),xlim=c(-0.2,0.2))
lines(density(modelo3$residuals))

########################################
#2.6.4 Modelos ARIMA
########################################
# Ajustamos el modelo ARIMA(2,0,1)
modelo4<- arima(serieTr.SinEst, order=c(2, 0, 1))
# Cogemos los valores del ajuste y las predicciones
# Cogemos los valores que se han ajustado de la
serie
valoresAjustados4<-
serieTr.SinEst+modelo4$residuals;
# Calculamos las predicciones
Predicciones4<- predict(modelo4, n.ahead = NPred);
valoresPredichos4<- Predicciones4$pred; # Cogemos
las predicciones
# Calculamos el error cuadrático acumulado del
ajuste, en ajuste y en test
errorTr4<- sum((modelo4$residuals)^2);
errorTs4<- sum((valoresPredichos4-
serieTs.SinEst)^2);
cat("Error en ajuste con ARIMA(2, 0, 1): ", errorTr4,
"\n")
cat("Error en la predicción de test con ARIMA(2, 0,
1): ", errorTs4, "\n")

# Mostramos las gráficas del ajuste y predicción en
test
plot.ts(serieTr.SinEst, xlim=c(1,
tiempoTs[length(tiempoTs)]))
title("Ajuste y predicción")
lines(valoresAjustados4, col="blue")
lines(tiempoTs, serieTs.SinEst, col="red")
lines(tiempoTs, valoresPredichos4, col="blue")
cat("Predicción con el modelo\n");

# Tests para la selección del modelo y su validación
boxtest4<- Box.test(modelo4$residuals) # Test de
aleatoriedad de Box-Pierce
cat(c("El test de Box-Pierce da un p-value=",
boxtest4$p.value, " para el modelo. Lo pasa (los
errores son aleatorios)\n"))
JB4<- jarque.bera.test(modelo4$residuals); # Test de
normalidad de Jarque Bera
cat(c("El test de Jarque Bera da un p-value=",
JB4$p.value, " para el modelo. Lo pasa (p-
value>0.05)\n"))
SW4<- shapiro.test(modelo4$residuals); # Test de
normalidad de Shapiro-Wilk
cat(c("El test de Shapiro-Wilk da un p-value=",
SW4$p.value, " para el modelo. Lo pasa (p-
value>0.05)\n"))

# Mostramos histograma de residuos
cat("Mostramos el histograma de los residuos del
modelo.\n")
hist(modelo4$residuals, col="blue",
prob=T,ylim=c(0,20),xlim=c(-0.2,0.2))
lines(density(modelo4$residuals))

########################################
#2.6.5 Modelos ARIMA
########################################
# Ajustamos el modelo ARIMA(1,0,2)
modelo5<- arima(serieTr.SinEst, order=c(1, 0, 2))
# Cogemos los valores del ajuste y las predicciones
# Cogemos los valores que se han ajustado de la
serie
valoresAjustados5<-
serieTr.SinEst+modelo5$residuals;
# Calculamos las predicciones
Predicciones5<- predict(modelo5, n.ahead = NPred);
valoresPredichos5<- Predicciones5$pred; # Cogemos
las predicciones
# Calculamos el error cuadrático acumulado del
ajuste, en ajuste y en test
errorTr5<- sum((modelo5$residuals)^2);
errorTs5<- sum((valoresPredichos5-
serieTs.SinEst)^2);
cat("Error en ajuste con ARIMA(1, 0, 2): ", errorTr5,
"\n")
cat("Error en la predicción de test con ARIMA(1, 0,
2): ", errorTs5, "\n")

# Mostramos las gráficas del ajuste y predicción en
test
plot.ts(serieTr.SinEst, xlim=c(1,
tiempoTs[length(tiempoTs)]))
title("Ajuste y predicción")
lines(valoresAjustados5, col="blue")
lines(tiempoTs, serieTs.SinEst, col="red")
lines(tiempoTs, valoresPredichos5, col="blue")
cat("Predicción con el modelo\n");

# Tests para la selección del modelo y su validación
```

```
boxtest5<- Box.test(modelo5$residuals) # Test de
aleatoriedad de Box-Pierce
cat(c("El test de Box-Pierce da un p-value=",
boxtest5$p.value, " para el modelo. Lo pasa (los
errores son aleatorios)\n"))
JB5<- jarque.bera.test(modelo5$residuals); # Test de
normalidad de Jarque Bera
cat(c("El test de Jarque Bera da un p-value=",
JB5$p.value, " para el modelo. Lo pasa (p-
value>0.05)\n"))
SW5<- shapiro.test(modelo5$residuals); # Test de
normalidad de Shapiro-Wilk
cat(c("El test de Shapiro-Wilk da un p-value=",
SW5$p.value, " para el modelo. Lo pasa (p-
value>0.05)\n"))

# Mostramos histograma de residuos
cat("Mostramos el histograma de los residuos del
modelo.\n")
hist(modelo5$residuals, col="blue",
prob=T,ylim=c(0,20),xlim=c(-0.2,0.2))
lines(density(modelo5$residuals))

########################################
#2.6.6 Modelos ARIMA
########################################
# Ajustamos el modelo ARIMA(1,0,1)
modelo6<- arima(serieTr.SinEst, order=c(1, 0, 1))
# Cogemos los valores del ajuste y las predicciones
# Cogemos los valores que se han ajustado de la
serie
valoresAjustados6<-
serieTr.SinEst+modelo6$residuals;
# Calculamos las predicciones
Predicciones6<- predict(modelo6, n.ahead = NPred);
valoresPredichos6<- Predicciones6$pred; # Cogemos
las predicciones
# Calculamos el error cuadrático acumulado del
ajuste, en ajuste y en test
errorTr6<- sum((modelo6$residuals)^2);
errorTs6<- sum((valoresPredichos6-
serieTs.SinEst)^2);
cat("Error en ajuste con ARIMA(1, 0, 1): ", errorTr6,
"\n")
cat("Error en la predicción de test con ARIMA(1, 0,
1): ", errorTs6, "\n")

# Mostramos las gráficas del ajuste y predicción en
test
plot.ts(serieTr.SinEst, xlim=c(1,
tiempoTs[length(tiempoTs)]))
title("Ajuste y predicción")
lines(valoresAjustados6, col="blue")
lines(tiempoTs, serieTs.SinEst, col="red")
lines(tiempoTs, valoresPredichos6, col="blue")
cat("Predicción con el modelo\n");

# Tests para la selección del modelo y su validación
boxtest6<- Box.test(modelo6$residuals) # Test de
aleatoriedad de Box-Pierce
cat(c("El test de Box-Pierce da un p-value=",
boxtest6$p.value, " para el modelo. Lo pasa (los
errores son aleatorios)\n"))
JB6<- jarque.bera.test(modelo6$residuals); # Test de
normalidad de Jarque Bera
cat(c("El test de Jarque Bera da un p-value=",
JB6$p.value, " para el modelo. Lo pasa (p-
value>0.05)\n"))
SW6<- shapiro.test(modelo6$residuals); # Test de
normalidad de Shapiro-Wilk
cat(c("El test de Shapiro-Wilk da un p-value=",
SW6$p.value, " para el modelo. Lo pasa (p-
value>0.05)\n"))

# Mostramos histograma de residuos
cat("Mostramos el histograma de los residuos del
modelo.\n")
hist(modelo6$residuals, col="blue",
prob=T,ylim=c(0,20),xlim=c(-0.2,0.2))
lines(density(modelo6$residuals))


########################################
#2.7 Selección del modelo
########################################
print("Probamos con 6 modelos: ARIMA(2,0,2),
ARIMA(0,0,1), ARIMA(1,0,0), ARIMA(2,0,1),
ARIMA(1,0,2) y ARIMA(1,0,1).");
print("Calculamos las predicciones y el modelo en
si");

#Ajuste de los modelos
par(mfrow=c(2,3))
plot.ts(serieTr.SinEst, xlim=c(1,
tiempoTs[length(tiempoTs)]));
title("Ajuste Modelo 1")
lines(valoresAjustados1, col="red");
lines(tiempoTs, serieTs.SinEst, col="green")
lines(valoresPredichos1, col="blue")
plot.ts(serieTr.SinEst, xlim=c(1,
tiempoTs[length(tiempoTs)]));
title("Ajuste Modelo 2")
lines(valoresAjustados2, col="red");
lines(tiempoTs, serieTs.SinEst, col="green")
lines(valoresPredichos2, col="blue")
plot.ts(serieTr.SinEst, xlim=c(1,
tiempoTs[length(tiempoTs)]));
title("Ajuste Modelo 3")
lines(valoresAjustados3, col="red");
lines(tiempoTs, serieTs.SinEst, col="green")
lines(valoresPredichos3, col="blue")
plot.ts(serieTr.SinEst, xlim=c(1,
tiempoTs[length(tiempoTs)]));
title("Ajuste Modelo 4")
lines(valoresAjustados4, col="red");
lines(tiempoTs, serieTs.SinEst, col="green")
lines(valoresPredichos4, col="blue")
plot.ts(serieTr.SinEst, xlim=c(1,
tiempoTs[length(tiempoTs)]));
title("Ajuste Modelo 5")
lines(valoresAjustados5, col="red");
lines(tiempoTs, serieTs.SinEst, col="green")
lines(valoresPredichos5, col="blue")
plot.ts(serieTr.SinEst, xlim=c(1,
tiempoTs[length(tiempoTs)]));
title("Ajuste Modelo 6")
```

```
lines(valoresAjustados6, col="red");
lines(tiempoTs, serieTs.SinEst, col="green")
lines(valoresPredichos6, col="blue")
print("Serie original (negro y verde), ajustada y
predicha con los diferentes modelos")
par(mfrow=c(1,1))

#Histograma de los residuos
par(mfrow=c(2,3))
hist(modelo1$residuals, col="blue",
prob=T,ylim=c(0,4),xlim=c(-0.2,0.2))
lines(density(modelo1$residuals))
hist(modelo2$residuals, col="blue",
prob=T,ylim=c(0,4),xlim=c(-0.2,0.2))
lines(density(modelo2$residuals))
hist(modelo3$residuals, col="blue",
prob=T,ylim=c(0,4),xlim=c(-0.2,0.2))
lines(density(modelo3$residuals))
hist(modelo4$residuals, col="blue",
prob=T,ylim=c(0,4),xlim=c(-0.2,0.2))
lines(density(modelo4$residuals))
hist(modelo5$residuals, col="blue",
prob=T,ylim=c(0,4),xlim=c(-0.2,0.2))
lines(density(modelo5$residuals))
hist(modelo6$residuals, col="blue",
prob=T,ylim=c(0,4),xlim=c(-0.2,0.2))
lines(density(modelo6$residuals))
par(mfrow=c(1,1))
# Comparamos ambos modelos por el criterio de AIC
resultsAIC<-AIC(modelo1, modelo2, modelo3,
modelo4, modelo5, modelo6)
print(resultsAIC)

########################################
#2.8 Predicción de la serie
########################################
# Probamos ahora a calibrar el mejor modelo con la
serie completa
tiempo<- 1:length(serie)
# Calculamos estacionalidad
k<-6
estacionalidad<-rep(0, k);
for (i in 1:k) {
  secuencia<-seq(i, length(serie), by=k);
  for (j in secuencia) {
    estacionalidad[i]<- estacionalidad[i] + serie[j];
  }

  estacionalidad[i]<-
estacionalidad[i]/length(secuencia);
}
aux<-rep(estacionalidad,
length(serie)/length(estacionalidad));
#Eliminamos estacionalidad
serieSinEst<- serie-aux;
# Ajustamos el modelo que hemos seleccionado
modelo<- arima(serieSinEst, order=c(0, 0, 1))
# Obtenemos ajuste y predicción
valoresAjustados<- serieSinEst+modelo$residuals;
Predicciones<- predict(modelo, n.ahead = NPred);
valoresPredichos<- Predicciones$pred; # Cogemos
las predicciones
# Por último, deshacemos cambios
valoresAjustados<- valoresAjustados+aux; #
Estacionalidad
valoresPredichos<- valoresPredichos+estacionalidad;
tiempoPred<- (tiempo[length(tiempo)]+(1:NPred));
plot.ts(serie, xlim=c(1, max(tiempoPred)), ylim=c(-
0.2, 1.5))
title("Predicción Final")
lines(valoresAjustados, col="blue")
lines(valoresPredichos, col="red")
```