Minería de Datos y Modelización Predictiva (IV)

Fernández Hernández, Alberto. 54003003S

18/02/2021

Contents

1. Introducción. Presentación de la serie a analizar

El objetivo del presente proyecto consiste en el análisis y modelado predictivo de una serie temporal con la **estimación de ventas mensuales en tiendas de ropa en Estados Unidos**, conocido como *Monthly Retail Sales*. Los datos han sido obtenidos del repositorio de Investigación Económica de la Reserva Federal del Banco de Saint Louis. [1]

```
ventas.ropa <- read_excel("retail_sales.xls")
```

El fichero de datos contiene un total de dos variables: *observation_date*, con la fecha de estimación, así como las ventas o *sales* (en millones de dólares). Dicho conjunto abarca un total de 168 observaciones mensuales, **desde enero del año 2006 hasta diciembre del año 2019**:

```
min(ventas.ropa$observation_date) # Fecha min: Enero 2006
```

```
## [1] "2006-01"
```

```
max(ventas.ropa$observation_date) # Fecha max: Diciembre 2019
```

```
## [1] "2019-12"
```

```
# Analizamos las 6 primeras filas
head(ventas.ropa)
```

```
## # A tibble: 6 x 2
##   observation_date sales
##   <chr>            <dbl>
## 1 2006-01          12893
## 2 2006-02          14474
## 3 2006-03          16386
## 4 2006-04          16848
## 5 2006-05          17103
## 6 2006-06          16505
```

Por otro lado, analizando brevemente las estadísticas de ventas podemos comprobar que **existe un cierto contraste en los valores de ventas mínimo y máximo**. A modo de ejemplo, el valor de la mediana nos indica la presencia de meses en los que las ventas se sitúan por debajo de los 20 mil millones de dólares, situación contraria en otros meses, donde la estimación de ventas aumenta considerablemente, hasta llegar incluso a los 34 mil millones (valor máximo):

```
summary(ventas.ropa$sales)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   12893   17103   18994   19896   21341   34611
```

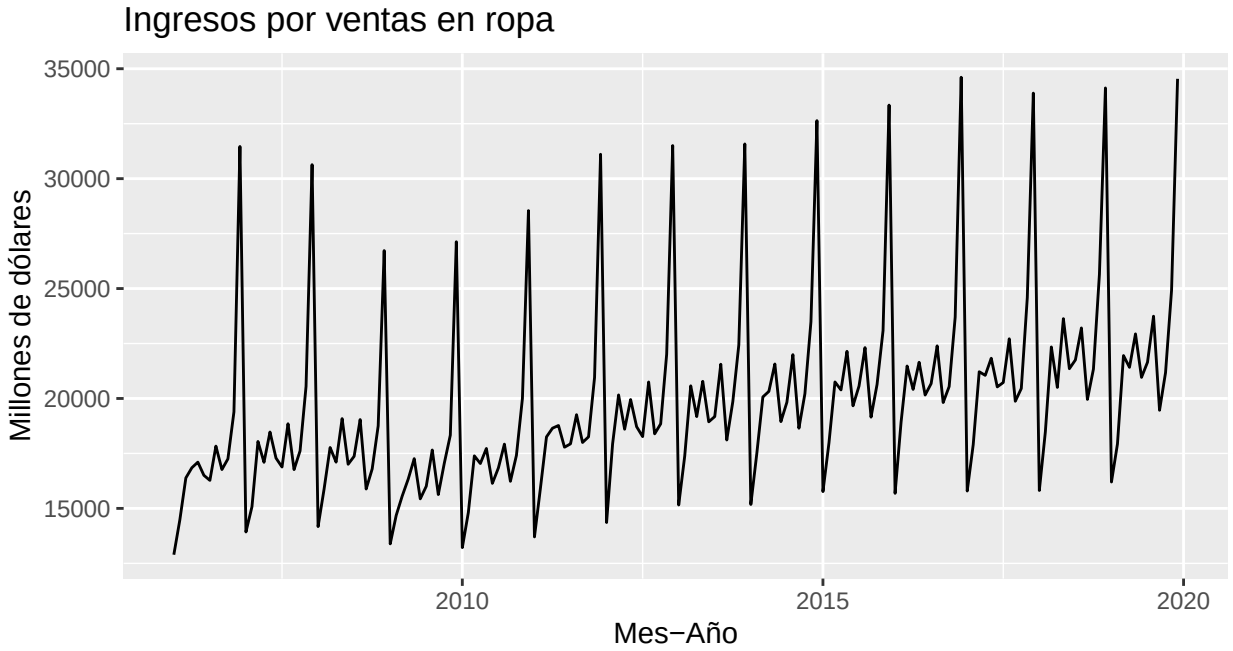
No obstante, con tan solo el *summary* no podemos aventurarnos a asegurar que la componente presenta estacionalidad, con valores mínimos y máximos de ventas anuales, por lo que necesitaremos de la representación gráfica para comprobarlo.

2. Representación gráfica y descomposición de la serie

De forma previa a los modelos predictivos, debemos representar gráficamente la serie temporal con el objetivo de estudiar su características tales como estacionalidad, tendencia y estacionariedad:

```
ventas.ropa.ts <- ts(ventas.ropa[,-1], start=c(2006,1), frequency=12)
ventas.ropa.test <- window(ventas.ropa.ts,start=c(2019,1), end=c(2019,12))
autoplot(ventas.ropa.ts) +  ggtitle("Ingresos por ventas en ropa") +
  xlab("Mes-Año") +  ylab("Millones de dólares")
```

[1] https://fred.stlouisfed.org/series/MRTSSM4481USN



Analizando la serie temporal, podemos extraer varias características: en primer lugar, la serie comienza con un decrecimiento en los ingresos desde el año 2006 hasta el año 2010, aproximadamente. Desde entonces, **la tendencia es prácticamente ascendente (media no constante)** hasta el año 2019 aproximadamente, momento en el que parece estabilizarse la serie. **En relación con la varianza, tampoco es constante**, presentando un aumento en la variabilidad de las ventas a lo largo de los años, tal y como podemos comprobar en el gráfico de la serie, donde la amplitud entre los valores de venta mínimo y máximo aumentan con el transcurso de los años, es decir, desde el año 2009-2010 existe cada vez un mayor constraste entre aquellos periodos donde se acentúa el número de ventas y momentos en los que se reduce al mínimo.

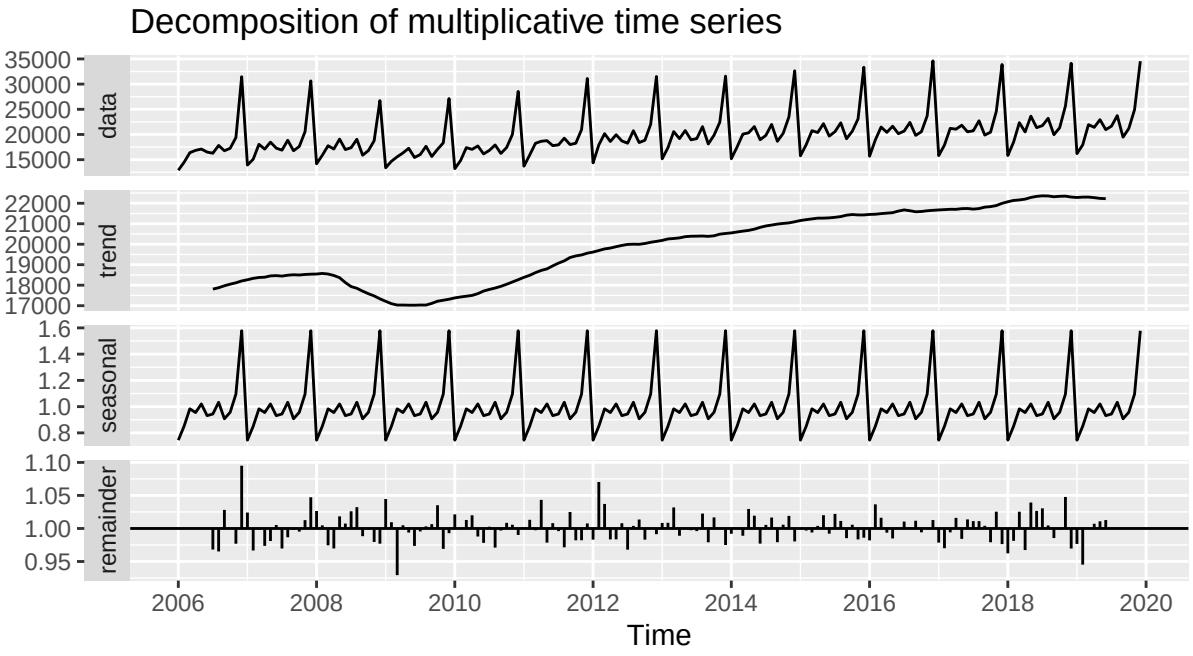
Por último, podemos observar que el número de ventas **presenta una variación estacional que parece repetirse anualmente**, con puntos de máximo y mínimo número de ventas. En conclusión, como primer análisis podemos extraer características de la serie como:

1. **Tendencia ascendente a partir del año 2010**.
2. **Aumento de la varianza con el transcurso del tiempo**.
3. **Estacionalidad anual en las ventas**.

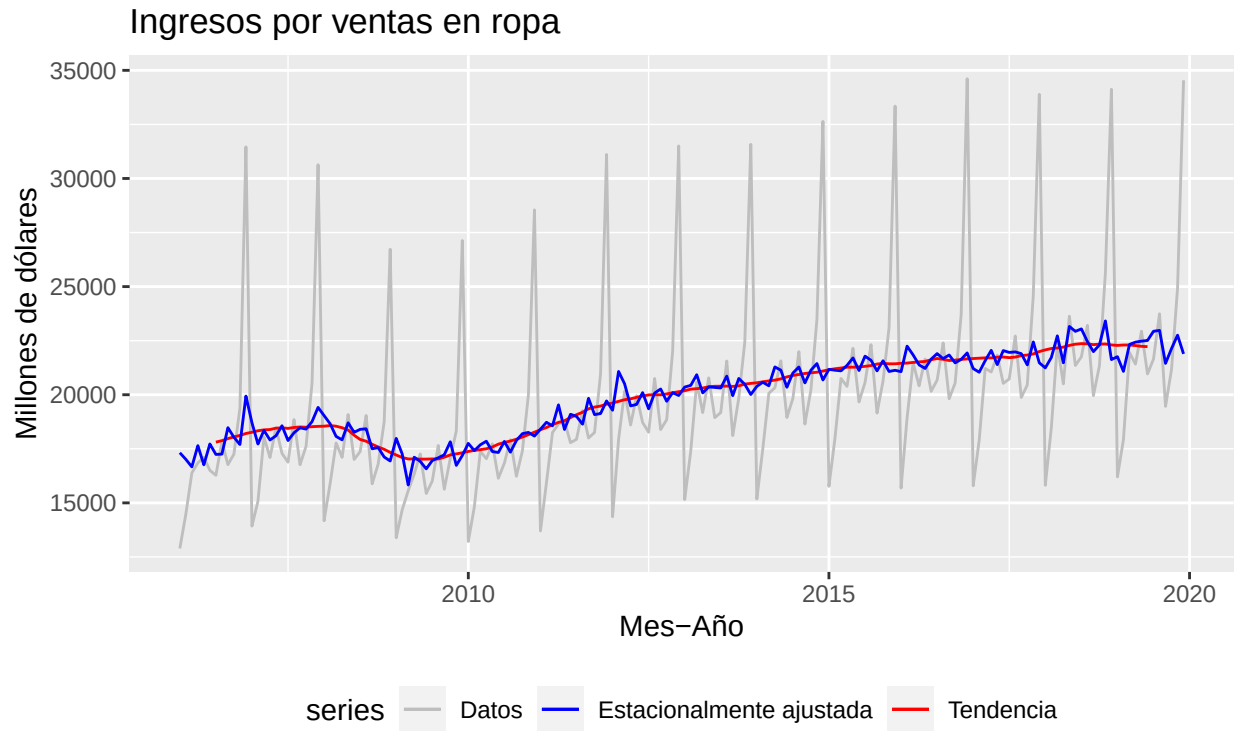
Por tanto, y en relación con las dos primeras características anteriores, podemos decir que **la serie presenta un esquema multiplicativo**.

Por otro lado, si analizamos la descomposición de la serie, podemos evidenciar tanto la tendencia ascendente desde el año 2009-2010, el aumento de la varianza con el paso de los años, así como la estacionalidad que presenta anualmente, donde los puntos con mayor número de ventas parecen corresponder con los meses de diciembre (ya que el "pico" de ingresos se sitúa al final de cada año), mientras que el instante inmediatamente posterior (posiblemente enero) es el mes con menor número de ingresos:

```
ventas.ropa.comp<- decompose(ventas.ropa.ts,type=c("multiplicative"))
autoplot(ventas.ropa.comp)
```



Nuevamente, observamos la tendencia claramente ascendente en el número de ventas a partir de la serie estacionalmente ajustada, pasando de una media de más de 15.000 millones de dólares en el año 2010 a más de 20.000 millones en el año 2019, año en el que el crecimiento comienza a dejar de ser lineal:



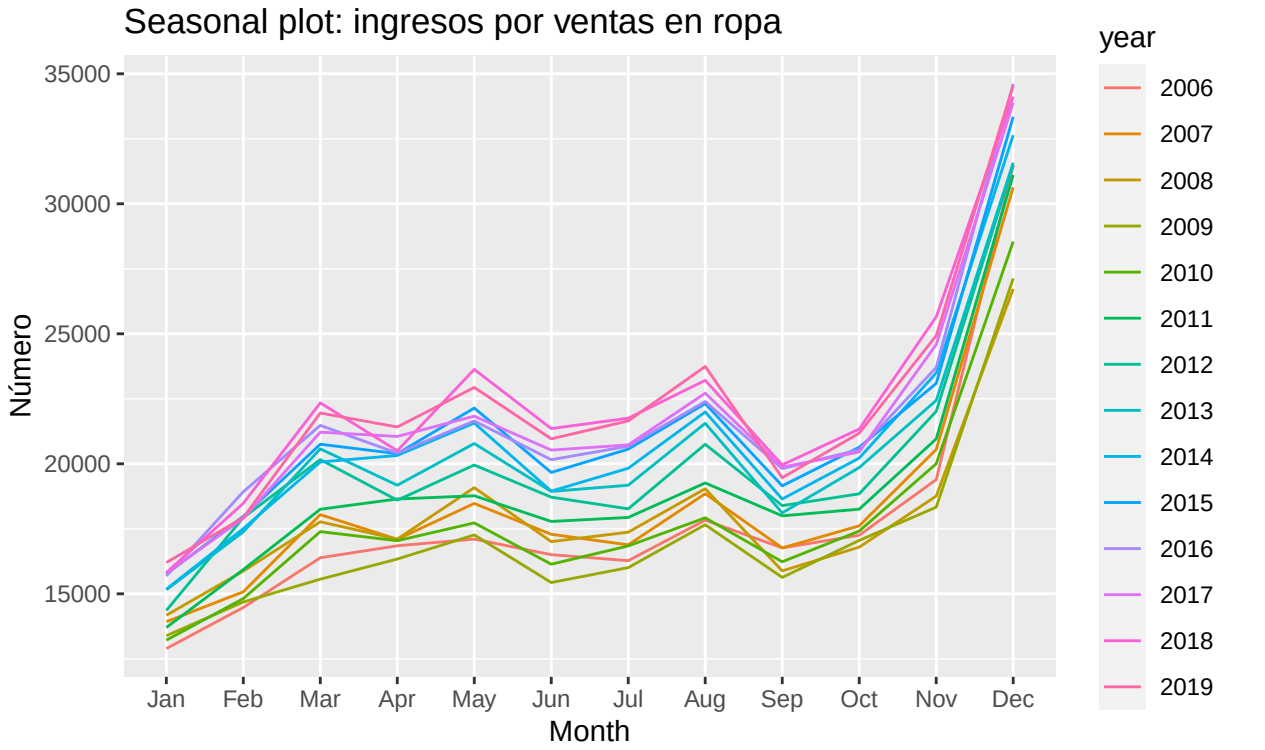
En relación con la estacionalidad, si analizamos los valores de la componente estacional:

```
comp.est <- data.frame(t(ventas.ropa.comp$seasonal[c(1:12)]))
```

```
##     Jan   Feb   Mar   Apr  May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
##  0.745 0.851 0.983 0.955 1.02 0.931 0.944 1.033 0.907 0.957 1.096 1.578
```

Efectivamente, observamos que **los ingresos por ventas en ropa en el mes de diciembre son un 57.8 %**

superior a la media anual, mientras que **los meses de enero y febrero concentran los porcentajes más bajos de ventas, con un 25.5 y un 14.9 % inferior en relación a la media anual, respectivamente**. Por otro lado, si analizamos las tendencias anuales:



Podemos comprobar, nuevamente, la tendencia ascendente de los ingresos por ventas en ropa con el transcurso del tiempo, siendo los últimos años, 2018 y 2019, los que presentan el mayor número de ingresos prácticamente en todos los meses.

Una vez realizado el primer análisis de la serie, de cara a los modelos tanto de suavizado exponencial como ARIMA **reservaremos los últimos 12 datos observados como conjunto de prueba para comprobar la eficacia de los métodos de predicción** (dado que la estacionalidad es anual, escogemos el último periodo, correspondiente a los ingresos del año 2019):

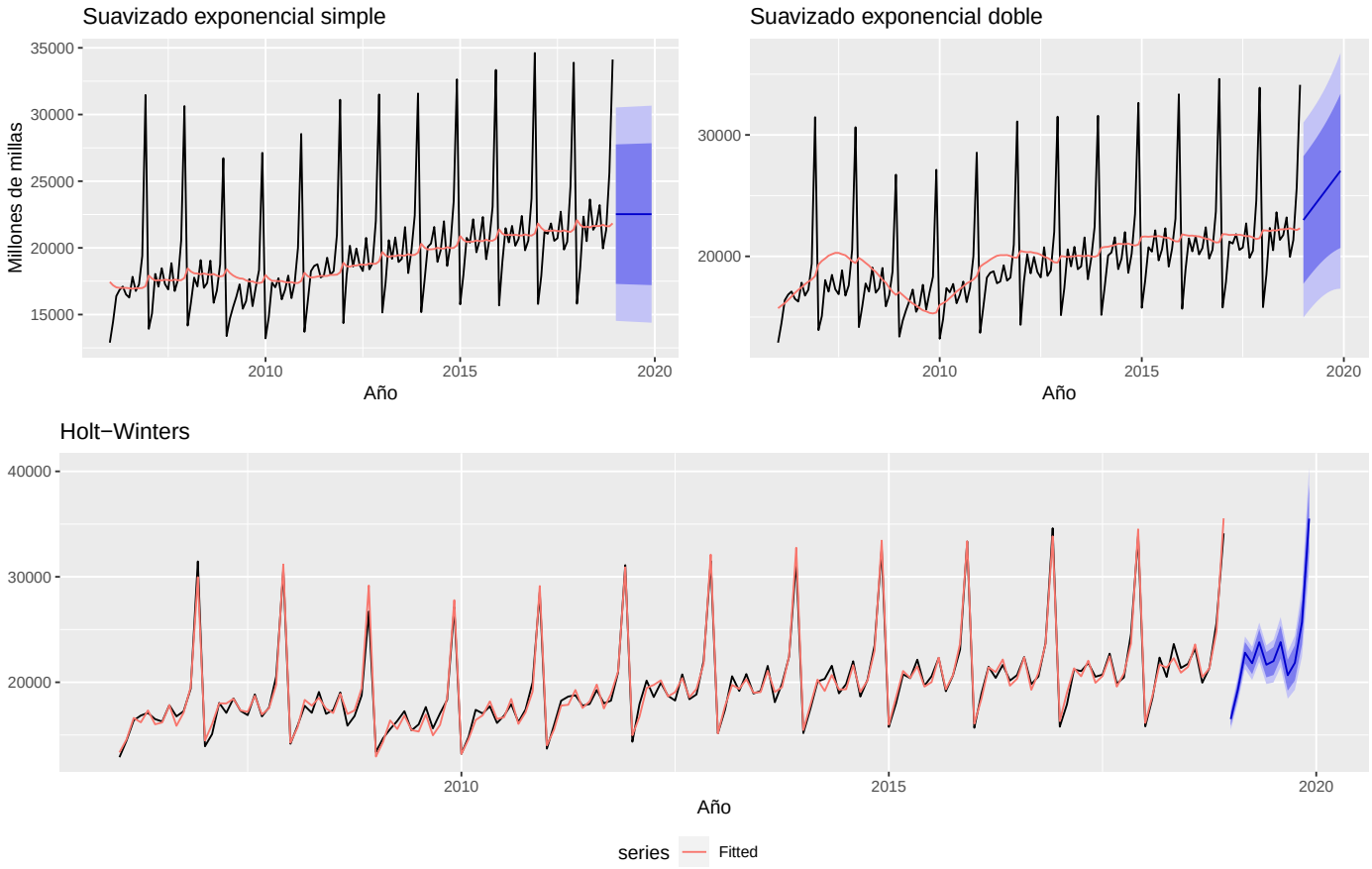
```
ventas.ropa.ts.transformado <- window(ventas.ropa.ts,start=c(2006,1),end=c(2018,12))
ventas.ropa.test <- window(ventas.ropa.ts,start=c(2019,1), end=c(2019,12))
```

3. Modelo de suavizado exponencial

Para determinar el mejor modelo de suavizado exponencial, realizamos una comparación de la precisión entre los diferentes modelos, tanto de alisado simple, doble, como de *Holt-Winters*. Dado que el conjunto de prueba empleado contiene los ingresos por venta del año 2019, la predicción calculada será para los siguientes 12 meses (h = 12). En el caso del modelo de *Holt-Winters*, dado que la serie es multiplicativa, debemos indicarlo a través del parámetro *seasonal*:

```
ventas.ropa.ss <- ses(ventas.ropa.ts.transformado, h=12) # Alisado simple
ventas.ropa.holt <- holt(ventas.ropa.ts.transformado, h=12) # Alisado doble (Holt)
ventas.ropa.hw <- hw(ventas.ropa.ts.transformado, h=12, seasonal="multiplicative") # Alisado Holt-Winters
estadisticas.suavizado <- rbind(round(accuracy(ventas.ropa.ss),3),
                                round(accuracy(ventas.ropa.holt),3), round(accuracy(ventas.ropa.hw),3))
```

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	AIC	BIC
Alisado Simple	581.130	4054.864	2392.677	-0.270	11.176	3.232	-0.077	3385.771	3394.921
Alisado Doble	0.774	4042.258	2604.200	-3.115	12.490	3.517	-0.053	3388.800	3404.049
Holt-Winters	-12.558	585.568	460.540	-0.074	2.347	0.622	-0.070	2796.921	2848.768



Analizando tanto la tabla como la salida gráfica, sin duda alguna **el método *Holt-Winters* ofrece un mejor modelo prácticamente en todos los sentidos**, desde los errores medios más bajos hasta valores AIC y BIC significativamente menores en comparación con los modelos de alisado simple y doble (2796 y 2848, respectivamente), **principalmente debido a que los modelos de *Holt-Winters* se emplean para series que presentan tendencia y estacionalidad**, mientras que el modelo de alisado simple devuelve la misma predicción para los siguientes 12 meses (series sin tendencia y estacionalidad) y el modelo de alisado doble realiza una predicción meramente lineal (series con tendencia pero sin estacionalidad). Como consecuencia, **el método de *Holt-Winters* se aproxima en mejor medida a los valores de la serie original**, lo que se traduce, además de un mejor ajuste en la predicción, en intervalos de confianza más cerrados, tal y como podemos comprobar en los gráficos anteriores.

Por tanto, dado el menor error, AIC y BIC obtenido, así como una mejor aproximación al comportamiento de la serie original, **elegimos como modelo ganador al obtenido por el método de *Holt-Winters***:

```
ventas.ropa.hw
```

```
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## Jan 2019       16513.22 15863.89 17162.54 15520.16 17506.28
## Feb 2019       19312.14 18511.94 20112.35 18088.33 20535.95
## Mar 2019       22804.55 21798.67 23810.43 21266.18 24342.91
## Apr 2019       21802.18 20770.20 22834.16 20223.90 23380.45
## May 2019       23799.02 22583.25 25014.80 21939.65 25658.39
## Jun 2019       21664.49 20465.84 22863.15 19831.31 23497.68
## Jul 2019       22010.00 20688.86 23331.15 19989.49 24030.52
```

```
## Aug 2019       23808.24 22257.44 25359.04 21436.50 26179.99
## Sep 2019       20668.68 19208.86 22128.50 18436.07 22901.28
## Oct 2019       21844.03 20173.60 23514.46 19289.33 24398.73
## Nov 2019       25698.13 23574.63 27821.63 22450.51 28945.74
## Dec 2019       35518.77 32354.38 38683.16 30679.26 40358.28
```

```
ventas.ropa.hw$model$par[1:3] # Obtenemos los parametros alpha, beta y gamma
```

```
##      alpha       beta      gamma
## 0.27962247 0.05314254 0.32077280
```

En base a los parámetros alfa, beta y gamma obtenidos, y dado que se trata de un modelo multiplicativo, la expresión del modelo final es la siguiente:

$$L_t = 0.2796\frac{x_t}{S_{t-s}} + (1-0.2796)(L_{t-1}+b_{t-1}) = 0.2796\frac{x_t}{S_{t-s}} + 0.7204(L_{t-1}+b_{t-1})$$

$$b_t = 0.0531(L_t - L_{t-1}) + (1-0.0531)b_{t-1} = 0.0531(L_t - L_{t-1}) + 0.9469b_{t-1}$$

$$S_t = 0.3207\frac{x_t}{L_t} + (1-0.3207)S_{t-s} = 0.3207\frac{x_t}{L_t} + 0.6793S_{t-s}$$

$$\hat{x}_{t+1} = (L_t + b_t)S_{t+1-s}$$

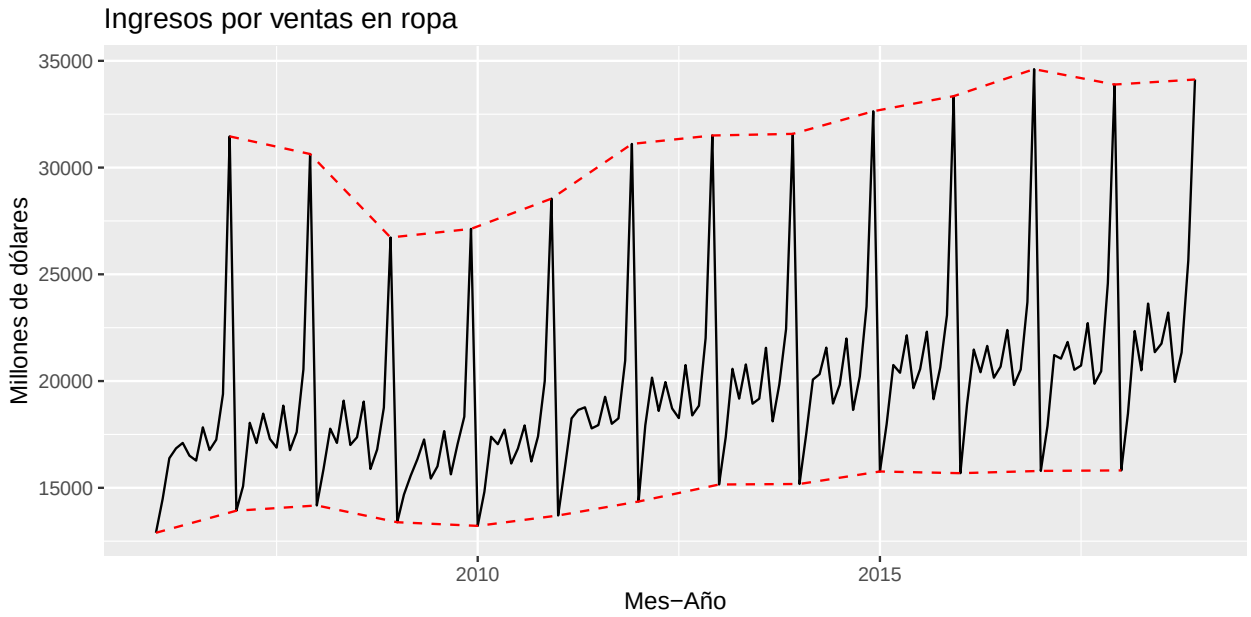
Donde L_t representa la componente de nivel de la serie temporal, b_t la tendencia y S_t la estacionalidad.

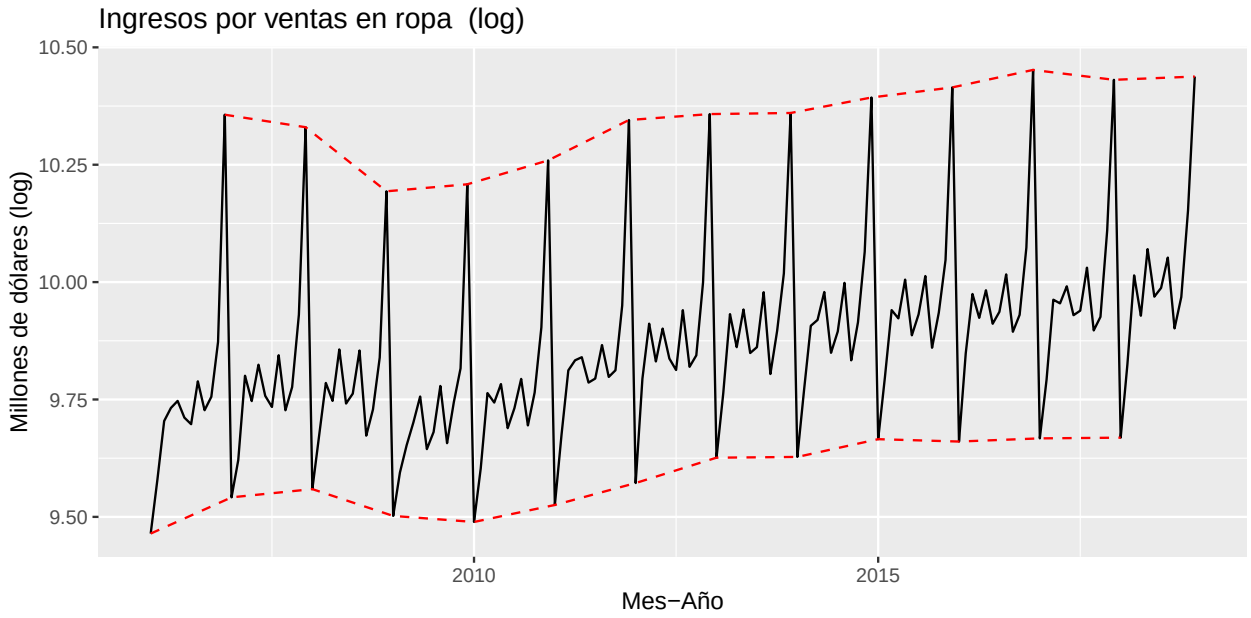
4. Modelo ARIMA

4.1 Transformaciones de la serie temporal

De forma previa a la elaboración del modelo ARIMA, así como a las funciones de autocorrelación y autocorrelación parcial, cabe recordar que la serie original **no es estacionaria en cuanto a varianza se refiere**, por lo que debemos comprobar si es posible, por medio de transformaciones Box-Cox, estabilizar dicha variabilidad. Como primera opción, realizamos una de las más comunes: **la logarítmica** (es decir, $\lambda = 0$).

```
serie.temp.original <- autoplot(ventas.ropa.ts.transformado)
transf.box.cox <-  log(ventas.ropa.ts.transformado)
serie.temp.log <- autoplot(transf.box.cox)
```





Como podemos observar en ambos gráficos, la transformación logarítmica **parece estabilizar la variabilidad de la serie, especialmente a partir del año 2010**, donde el aumento de la amplitud de la varianza ya no es tan significativa en comparación con la serie original, aunque conserva su tendencia ascendente. Por otro lado, la librería *forecast* dispone de una función denominada *BoxCox.lambda* que permite obtener el coeficiente *lambda* óptimo para la transformación de la serie. Dispone de dos métodos: *loglik*, que elige el valor de *lambda* que maximice la verosimilitud de la transformada con respecto a un modelo lineal; y *guerrero*, que escoge el parámetro *lambda* que minimice el coeficiente de variación para cada una de las sub-series del conjunto de datos. En caso de que λ sea 1, implicaría que la transformación no es necesaria:

```
BoxCox.lambda(ventas.ropa.ts.transformado, method = "guerrero")
```

```
## [1] 0.3614745
```

```
BoxCox.lambda(ventas.ropa.ts.transformado, method = "loglik")
```

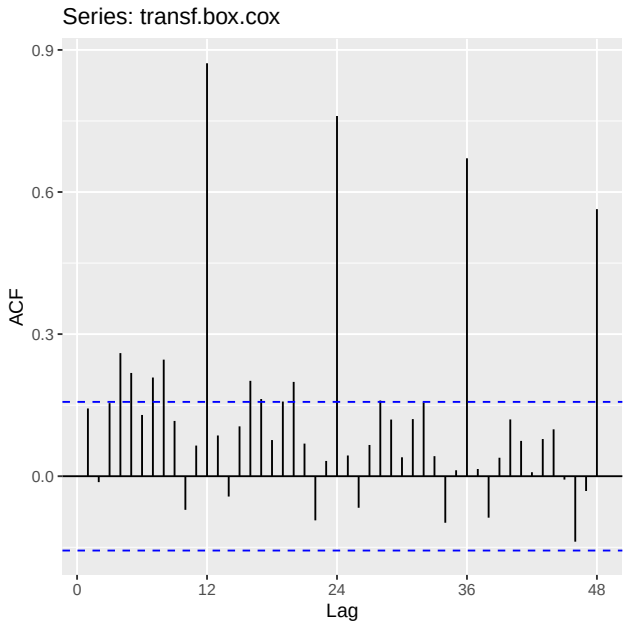
```
## [1] 0.05
```

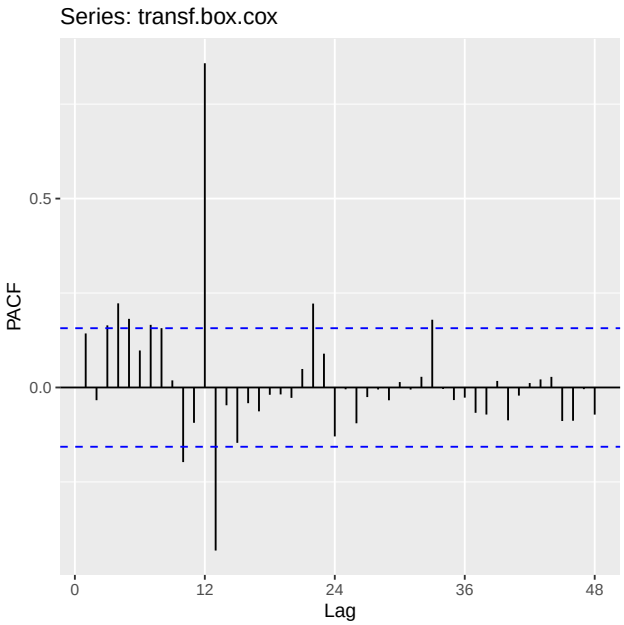
Como podemos observar, el valor de *lambda* en ambos casos es más cercano a 0 (0.36 y 0.05, respectivamente), **lo que evidencia nuevamente la necesidad de transformar la serie original**. De hecho, el propio método *loglik* sugiere un valor de *lambda* muy cercano a 0 (0.05), correspondiente con la transformada logarítmica. No obstante, se han comparado gráficamente la serie logarítmica junto con las series transformadas a partir de los valores *lambda* anteriores (0.36 y 0.05), pero la diferencia no entre ambas no es muy significativa. Por tanto, dado que la transformada logarítmica permite estabilizar en gran medida la varianza de la serie, además de que los valores *lambda* obtenidos mediante la función *BoxCox.lambda* no aportan apenas mejoría, **de aquí en adelante se trabajará con la serie logarítmica**.

4.2 Funciones de autocorrelación y autocorrelación parcial

Una vez transformada la serie temporal, representamos gráficamente tanto la función de autocorrelación como de autocorrelación parcial, con el objetivo no solo de analizar la estacionariedad de la serie, sino además de determinar el tipo de modelo ARIMA en función de los retardos que son significativamente distintos de cero. Dado que la serie presenta estacionalidad anual, como criterio personal se ha decidido calcular las autocorrelaciones hasta el retardo 48, es decir, hasta 4 años:

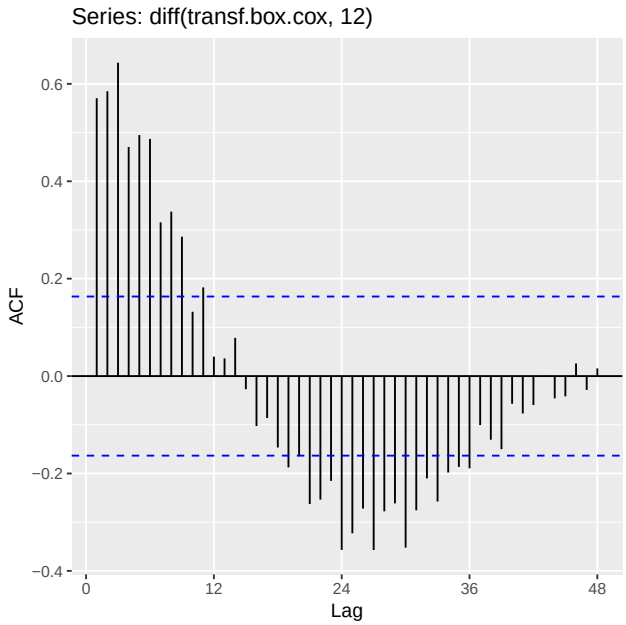
```
ggAcf(transf.box.cox, lag = 48) # Funcion de Autocorrelacion
ggPacf(transf.box.cox, lag = 48) # Funcion de Autocorrelacion Parcial
```

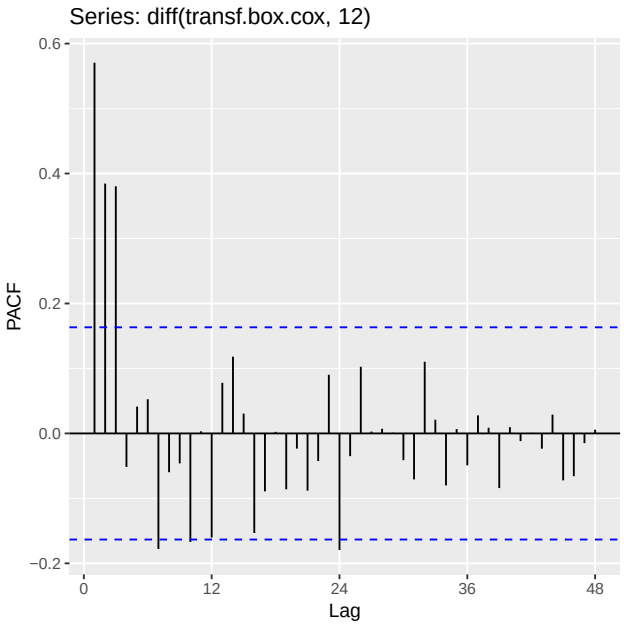




En los primeros pasos, debemos fijarnos sobretodo en la Función de Autocorrelación. En ella, detectamos la estacionalidad mencionada en los primeros apartados: por un lado, existe un patrón de autocorrelación que se repite anualmente y que disminuye conforme aumentan los retardos. De hecho, la estacionalidad se aprecia mejor en los **retardos múltiplos de la estacionalidad**: 12, 24, 36 y 48, con valores máximos anuales que van decreciendo lentamente conforme aumentan los retardos, claro indicio de que la serie no es estacionaria. Por tanto, para eliminar dicha estacionalidad **debemos aplicar una diferenciación de orden 12 (anual)**:

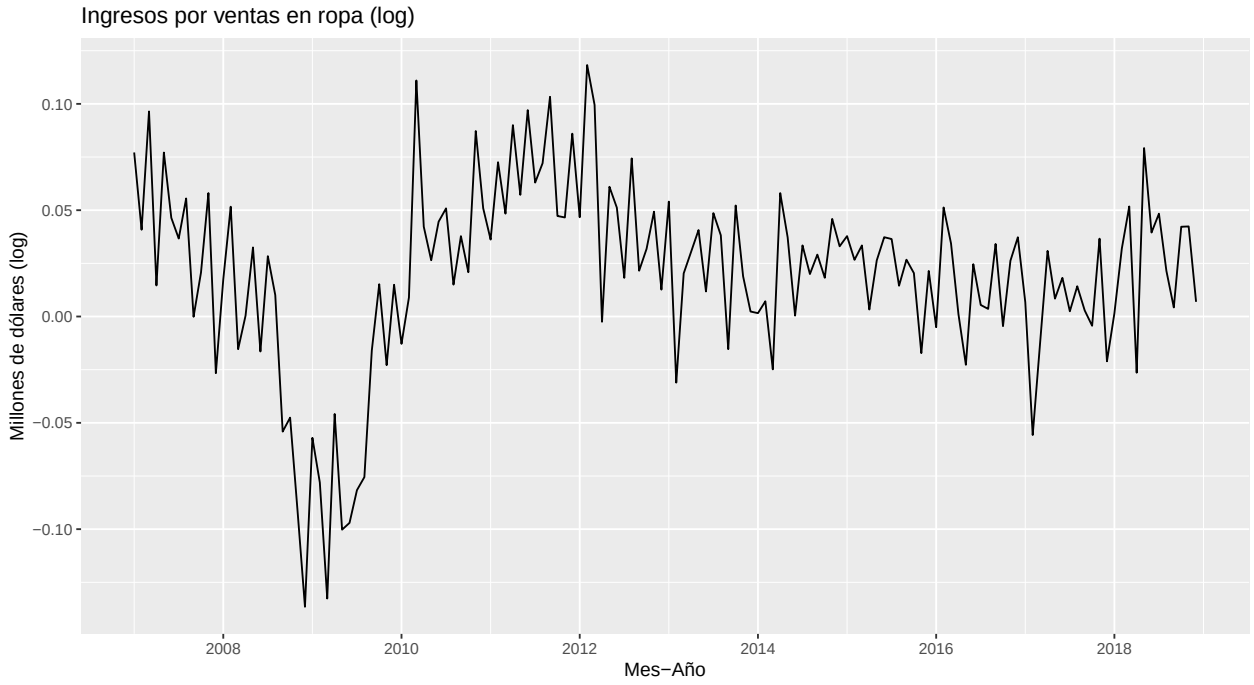
```
ggAcf(diff(transf.box.cox, 12), lag = 48) # Funcion de Autocorrelacion
ggPacf(diff(transf.box.cox, 12), lag = 48) # Funcion de Autocorrelacion Parcial
```





Una vez aplicada la diferenciación estacional, la serie continúa sin ser estacionaria, principalmente por un motivo: **sigue existiendo un decrecimiento lento de los valores de autocorrelación**, tal y como podemos observar en la función ACF, es decir, la media sigue sin ser constante. Una forma de comprobarlo sería gráficamente:

```
autoplot(diff(transf.box.cox, 12)) +  ggtitle("Ingresos por ventas en ropa (log)") +
  xlab("Mes-Año") +  ylab("Millones de dólares (log)")
```



Como podemos comprobar nuevamente, la estacionalidad anual se ha visto reducida considerablemente. No obstante, debido al decrecimiento en el número de ventas ocasionado entre los años 2007-2010 (periodo de recesión económica), provoca que la media no sea constante y, por tanto, **no podemos asegurar que la serie sea estacionaria en sentido débil**: *si escojo dos series en cualquier instante de tiempo, sus medias y varianzas deberán ser constantes*:

$$E(X_t) = E(X_{t+k}) = \mu$$

Sin embargo, el decrecimiento producido entre estos años lo impide. De hecho, existen contrastes de hipóstesis que permiten comprobar si la serie presenta o no estacionariedad. Uno de ellos es el método conocido como *adf* o *Augmenteed Dickey-Fuller test*, que toman como hipótesis nula que la serie no es estacionaria, también conocido como **raíz unitaria** (procesos que evolucionan a través del tiempo), en contraposición con la hipótesis alternativa [2]. Para ello, la librería *tseries* dispone de la función *adf.test* con el que realizar el contraste de hipótesis:

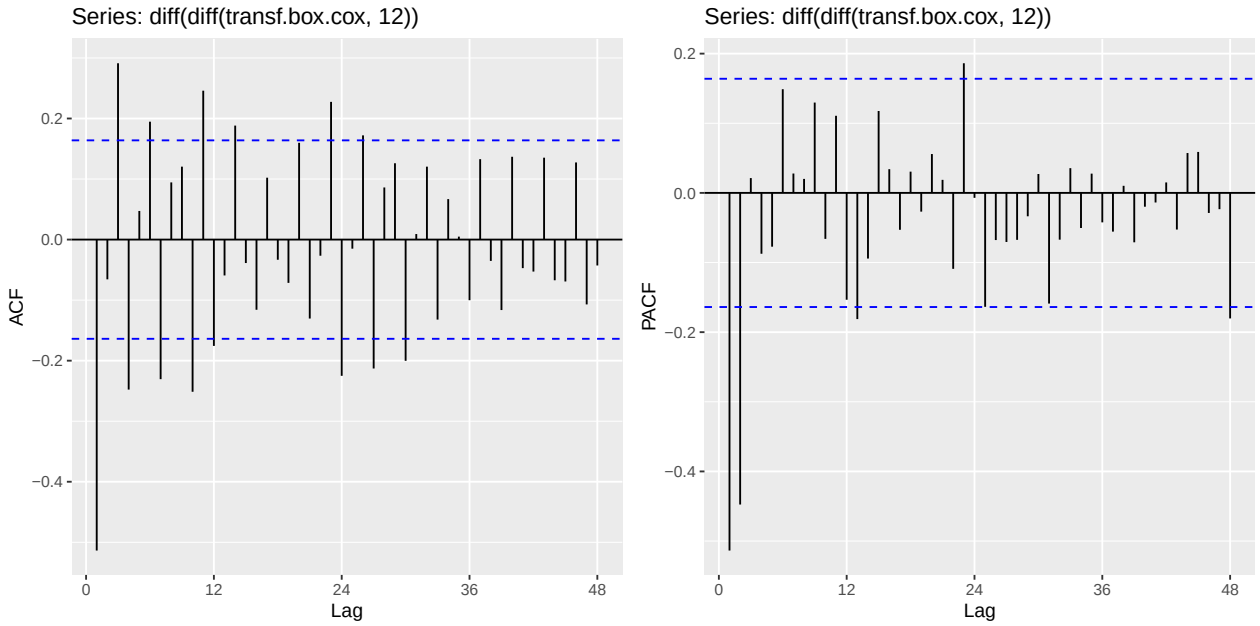
```
tseries::adf.test(diff(transf.box.cox, 12)) # Hipotesis nula: la serie NO es estacionaria
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff(transf.box.cox, 12)
## Dickey-Fuller = -2.1883, Lag order = 5, p-value = 0.498
## alternative hypothesis: stationary
```

En el caso anterior, el p-valor obtenido en el *test* es de 0.498, un valor significativamente superior a 0.05, por lo que **no podemos rechazar la hipótesis nula al 95 % de confianza**, es decir, estadísticamente no existe evidencia en contra de que la serie no sea estacionaria. Por tanto, dado que no solo gráfica, sino además estadísticamente hemos podido asegurar que la serie continúa sin ser estacionaria, **debemos aplicar nuevamente una diferenciación a la componente regular**:

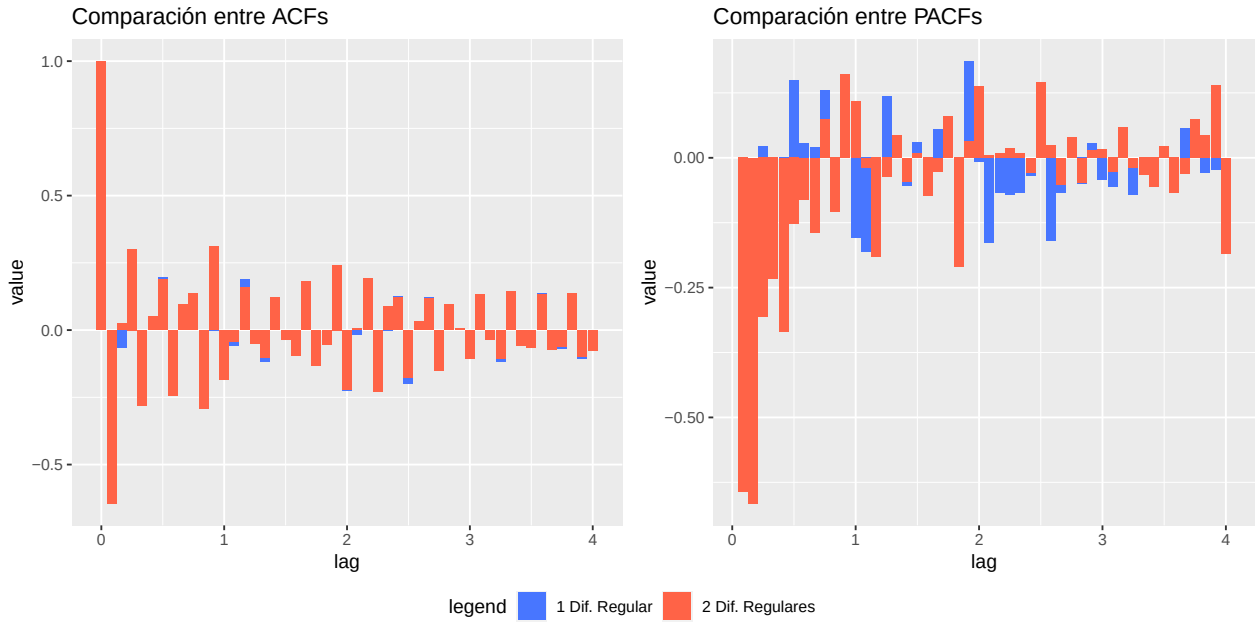
```
ggAcf(diff(diff(transf.box.cox, 12)), lag = 48) # Funcion de Autocorrelacion
ggPacf(diff(diff(transf.box.cox, 12)), lag = 48) # Funcion de Autocorrelacion Parcial
```

[2] https://nwfsc-timeseries.github.io/atsa-labs/sec-boxjenkins-aug-dickey-fuller.html



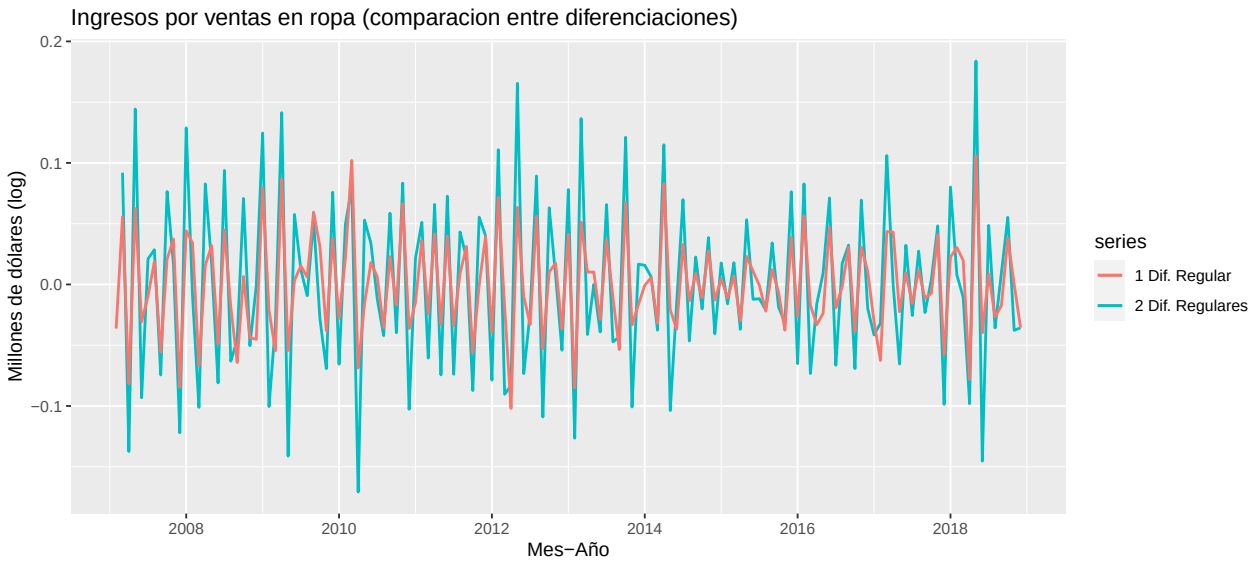
Como podemos comprobar, hemos conseguido eliminar "parcialmente" el decrecimiento de los retardos en la función de Autocorrelación. No obstante, pese a la diferenciación aplicada, continúa existiendo un decaimiento lento en los valores de autocorrelación. Por tanto, ¿Y si aplicamos una diferenciación de orden 2 a la parte regular para eliminar dicho decrecimiento? Analicemos el resultado, **comparando ambas funciones de autocorrelación, tanto con una diferenciación de orden 1 como de orden 2 en la parte regular**. Para ello, se ha elaborado una función denominada *comparar.autocorrelaciones*, que recibe como parámetros las series a utilizar, el tipo de función de autocorrelación a mostrar (ACF o PACF), así como el título del gráfico (empleando las funciones de *ggplot2*):

```
comparar.autocorrelaciones(diff(diff(transf.box.cox, 12)), # Funcion de Autocorrelacion
                diff(diff(diff(transf.box.cox, 12))), tipo = "acf", "Comparación entre ACFs")
comparar.autocorrelaciones(diff(diff(transf.box.cox, 12)), # Funcion de Autocorrelacion Parcial
                diff(diff(diff(transf.box.cox, 12))), tipo = "pacf", "Comparación entre PACFs")
```



Pese a aplicar una diferenciación adicional, **los valores de autocorrelación no se han visto prácticamente afectados**: en el diagrama de barras de la izquierda podemos comprobar que tan solo ve reducida la correlación en un pequeño subconjunto de retardos (donde la barra en rojo es menor a la barra azul). En el resto de autocorrelaciones,

el valor no ha disminuido practicamente. Con respecto a la función de autocorrelación parcial, bien es cierto que muchas de las autocorrelaciones se han visto reducidas. Sin embargo, debemos destacar una importante diferencia en la función de autocorrelación parcial: **mientras que con una diferenciación regular tan solo los dos primeros retardos son significativos, empleando dos diferenciaciones el número aumenta hasta 5**. Además, incluso si comparamos ambas series observamos un gran contraste en la varianza, con una variabilidad mucho mayor empleando dos diferenciaciones:



Por tanto, **dado que una diferenciación de orden 2 no reduce el decrecimiento en la función de autocorrelación parcial, además de aumentar incluso la variabilidad en la serie, optamos por una única diferenciación en la parte regular**. Si finalmente realizamos el contraste de hipótesis mediante la función *adf.test*, comprobamos que efectivamente la serie ya es estacionaria, dado que el p-valor es inferior a 0.05, rechazando con ello la hipótesis nula de la raíz unitaria:

```
tseries::adf.test(diff(diff(transf.box.cox, 12)))
```

```
## Warning in tseries::adf.test(diff(diff(transf.box.cox, 12))): p-value smaller
## than printed p-value
```

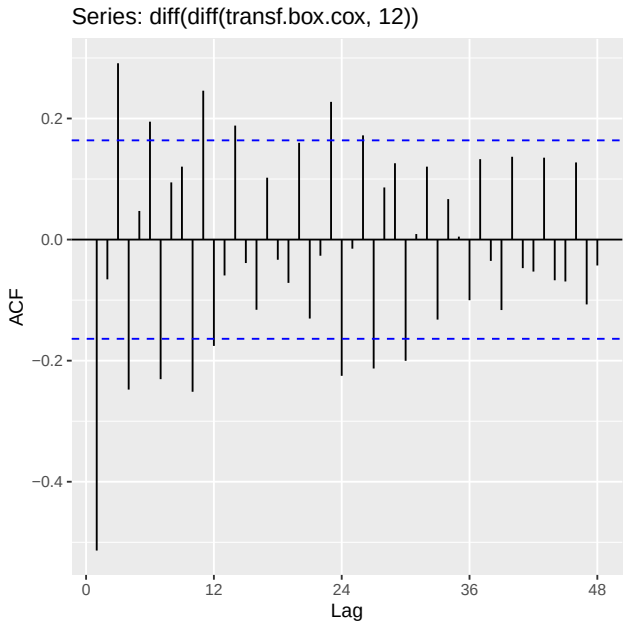
```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff(diff(transf.box.cox, 12))
## Dickey-Fuller = -5.1257, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

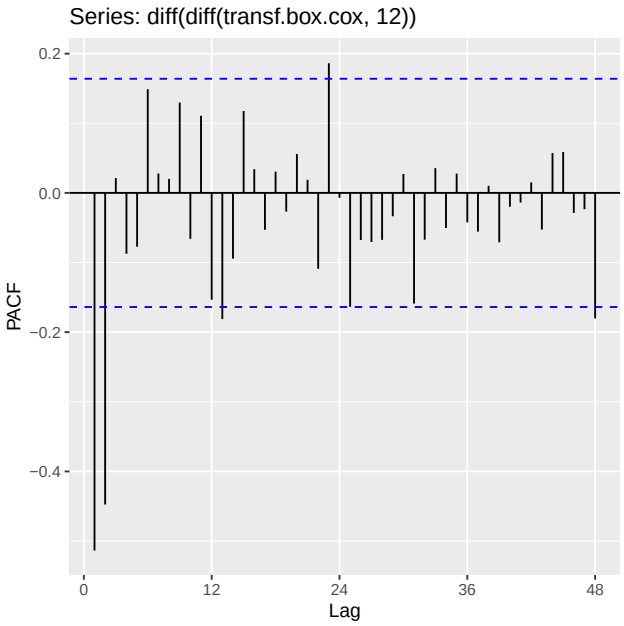
4.3 Selección de los parámetros del modelo

Una vez aplicadas las diferenciaciones, ya tenemos los coeficientes d y D del modelo ARIMA (1). Por tanto, debemos ajustar el resto de parámetros (autoregresivo y media móvil) tanto de la componente regular como estacional. Regresemos nuevamente con las funciones de autocorrelación y autocorrelación parcial:

PARTE INTEGRADA:

- **Componente Autoregresiva y Media Móvil (p y q)**: tal y como podemos observar en las funciones de autocorrelación, **nos encontramos con los primeros p = 2 parámetros significativos en la función de autocorrelación parcial**, así como un **decrecimiento atenuado (por medio de ondas sinusoidales) de los retardos en la función de autocorrelación parcial**. Por otro lado, dicha función no presenta aparentemente un decrecimiento atenuado de sus retardos u ondas sinusoidales, por lo que no parece ser necesario el uso de medias móviles. Por tanto, **de la parte integrada consideramos, junto con la diferenciación, una componente autoregresiva de orden 2**, es decir, ARIMA (2,1,0).





PARTE ESTACIONAL:

- **Componente Autoregresiva y Media Móvil (p y q)**: en relación con la componente estacional, nos encontramos antes varias opciones posibles:
  - **ARIMA (p = 0, d = 1, q = 2)**: por un lado, desde la función de autocorrelación (ACF) detectamos los primeros q = 2 retardos (múltiplos de la estacionalidad) significativos, concretamente en los retardos 12 y 24. Por otro lado, en relación con la función de autocorrelación parcial, bien es cierto que el decrecimiento de los retardos múltiplos de 12 no es tan claro, aunque si existe un comportamiento sinusoidal, especialmente en los retardos 12, 24 y 48. Por tanto, **una posibilidad sería considerar, junto con la diferenciación, una media móvil de orden 2**.
  - **ARIMA (p = 4, d = 1, q = 0)**: del mismo modo que consideramos únicamente una parte de media móvil, también podemos tener solamente en cuenta una componente autoregresiva, con los q = 4 primeros retardos significativos en la función de autocorrelación parcial (también tenemos en cuenta el retardo 48, aunque el retardo 36 no sea significativo), así como un cierto decrecimiento en los retardos de la función de autocorrelación, donde a partir del retardo 36 el valor se sitúa por debajo del umbral de aceptación. Por tanto, **otra opción sería considerar únicamente una componente autoregresiva de orden 4**.
  - **ARIMA (p = 4, d = 1, q = 2)**: por último, podría incluso considerarse un modelo ARIMA completo, dado que en la función de autocorrelación parcial los retardos múltiplos de 12 no "caen" repentinamente por debajo del umbral de aceptación (salvo el retardo 36); además de que los q = 2 primeros retardos son significativos en la función de autocorrelación. Por tanto, **otra posibilidad sería considerar tanto una componente autoregresiva de orden 4 como media móvil de orden 2**.

Bien es cierto que los dos últimos modelos estacionales pueden no ser lo más adecuados, principalmente porque no todas las componentes autoregresivas son significativas, concretamente el tercer valor, tal y como hemos podido comprobar gráficamente. Sin embargo, considerar ambas posibilidades puede servirnos de gran ayuda a la hora de determinar qué componente o componentes aportan un mayor peso a la parte estacional, esto es, si la componente autoregresiva, la media móvil o ambas. Por tanto, mediante la función *arima* realizamos un primer análisis, comparando los resultados obtenidos en los tres modelos:

```
fitARIMA.1<-arima(transf.box.cox,order = c(2,1,0), seasonal=c(0,1,2)) # ARIMA (2,1,0) (0,1,2)
fitARIMA.1.1 <- arima(transf.box.cox,order = c(2,1,0), seasonal=c(4,1,0)) # ARIMA (2,1,0) (4,1,0)
fitARIMA.1.2 <- arima(transf.box.cox,order = c(2,1,0), seasonal=c(4,1,2)) # ARIMA (2,1,0) (4,1,2)
```

Una vez creados los modelos, recuperamos las estadísticas de cada uno, incluyendo los errores obtenidos, los criterios AIC y BIC, así como el p-valor resultante del test de *Lljung-Box*:

```
estadisticas <- rbind(cbind(round(accuracy(fitARIMA.1), 3), "AIC" = AIC(fitARIMA.1),
                      "BIC" = BIC(fitARIMA.1), "p-valor " = checkresiduals(fitARIMA.1,
```

```
                    plot = FALSE)$p.value),
      cbind(round(accuracy(fitARIMA.1.1), 3), "AIC" = AIC(fitARIMA.1.1), "BIC" = BIC(fitARIMA.1.1),
          "p-valor " = checkresiduals(fitARIMA.1.1, plot = FALSE)$p.value),
      cbind(round(accuracy(fitARIMA.1.2), 3), "AIC" = AIC(fitARIMA.1.2), "BIC" = BIC(fitARIMA.1.2),
          "p-valor " = checkresiduals(fitARIMA.1.2, plot = FALSE)$p.value))
```

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	AIC	BIC	p-valor
ARIMA (2,1,0)(0,1,2)	-0.001	0.027	0.020	-0.012	0.204	0.122	0.000	-603.412	-588.598	0.474
ARIMA (2,1,0)(4,1,0)	-0.001	0.026	0.020	-0.011	0.199	0.119	-0.015	-603.153	-582.413	0.324
ARIMA (2,1,0)(4,1,2)	-0.001	0.025	0.019	-0.012	0.190	0.114	-0.013	-601.435	-574.769	0.309

Comenzando con los errores promedio obtenidos, **observamos que la diferencia es prácticamente pequeña en cualquiera de los modelos**, donde incluso en el tercer modelo, con componentes tanto autoregresiva como de media móvil, la diferencia de error con respecto al resto de modelos es de tan solo unas milésimas. Por otro lado, tanto el criterio AIC como BIC dan una ligera ventaja al tercer modelo ARIMA, aunque nuevamente por muy poca diferencia: de tan solo 2 puntos en el AIC y menos de 20 puntos en el BIC. No obstante, llama especialmente la atención la diferencia en el p-valor obtenido en el test de *Llunj-Box*: **empleando tan solo 2 medias móviles, el primer modelo obtiene un valor de 0.47, significativamente mayor que el resto de modelos, es decir, empleando únicamente medias móviles conseguimos que el ruido sea lo más incorrelado posible (ruido blanco)**. A simple vista, las medias móviles parecen ser mucho más relevantes que la componente autoregresiva o un modelo ARIMA completo. Sin embargo, ¿Son todas las variables significativas?

```
coeftest(fitARIMA.1); coeftest(fitARIMA.1.1); coeftest(fitARIMA.1.2)
```

```
ARIMA (2,1,0)(0,1,2)                                    ARIMA (2,1,0)(4,1,0)                                    ARIMA (2,1,0)(4,1,2)
z test of coefficients:

     Estimate  Std. Error z value  Pr(>|z|)     Estimate  Std. Error z value  Pr(>|z|)     Estimate Std. Error z value  Pr(>|z|)
ar1 -0.713482   0.075412 -9.4612 < 2.2e-16 ***  ar1 -0.661535   0.087315 -7.5764 3.552e-14 ***  ar1 -0.624438   0.087183 -7.1624 7.929e-13 ***
ar2 -0.453952   0.075472 -6.0149 1.800e-09 ***  ar2 -0.420648   0.079274 -5.3062 1.119e-07 ***  ar2 -0.375441   0.083968 -4.4712 7.778e-06 ***
sma1 -0.462673  0.099281 -4.6602 3.159e-06 ***  sar1 -0.503937  0.095562 -5.2734 1.339e-07 ***  sar1  0.301381   0.294014  1.0251 0.3053360
sma2 -0.233735  0.094450 -2.4747   0.01333 *    sar2 -0.533347  0.106884 -4.9900 6.039e-07 ***  sar2 -0.880520   0.255302 -3.4489 0.0005628 ***
                                                sar3 -0.344883  0.112617 -3.0624  0.002195 **   sar3 -0.228808   0.120563 -1.8978 0.0577189 .
                                                sar4 -0.192323  0.115060 -1.6715  0.094624 .    sar4 -0.276078   0.163381 -1.6898 0.0910703 .
                                                                                             sma1 -0.818056   0.340178 -2.4048 0.0161819 *
                                                                                             sma2  0.846148   0.568193  1.4892 0.1364371
```

Figure 1: Importancia de los coeficientes en los modelos ARIMA (I)

Tal y como podemos observar en la imagen anterior, pese a incluir un mayor número de parámetros, tanto el segundo como especialmente el tercer modelo ARIMA **presenta algunas de las variables con poca o ninguna importancia**. A modo de ejemplo, la cuarta componente autoregresiva apenas es relevante, con un p-valor por encima de 0.05. Por tanto, ¿Que ocurriría si eliminamos esta última componente? Es decir, mantener una componente autoregresiva de orden 3:

```
fitARIMA.1.1 <- arima(transf.box.cox,order = c(2,1,0), seasonal=c(3,1,0)) # ARIMA (2,1,0) (3,1,0)
fitARIMA.1.2 <- arima(transf.box.cox,order = c(2,1,0), seasonal=c(3,1,2)) # ARIMA (2,1,0) (3,1,2)
```

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	AIC	BIC	p-valor
ARIMA (2,1,0)(0,1,2)	-0.001	0.027	0.020	-0.012	0.204	0.122	0.000	-603.412	-588.598	0.474
ARIMA (2,1,0)(3,1,0)	-0.001	0.027	0.020	-0.012	0.203	0.122	-0.005	-602.483	-584.706	0.378
ARIMA (2,1,0)(3,1,2)	-0.001	0.026	0.019	-0.012	0.197	0.118	-0.017	-601.667	-577.964	0.276

Analizando nuevamente la tabla, tras eliminar la cuarta componente autoregresiva, el p-valor aumenta significativamente en el segundo modelo ARIMA, pasando de 0.32 a 0.37. Por el contrario, el p-valor en el tercer modelo si se ha visto reducido (de 0.30 a 0.27 al eliminar la cuarta componente). Por otro lado, si comparamos los dos primeros modelos, pese a tener valores de error, AIC y BIC prácticamente idénticos, el p-valor del primer modelo ARIMA continua aportando una mayor ventaja (0.47 frente a 0.37). En relación con la importancia de las variables:

```
coeftest(fitARIMA.1); coeftest(fitARIMA.1.1); coeftest(fitARIMA.1.2)
```

```
ARIMA (2,1,0)(0,1,2)
z test of coefficients:

     Estimate  Std. Error z value  Pr(>|z|)
ar1 -0.713482   0.075412 -9.4612 < 2.2e-16 ***
ar2 -0.453952   0.075472 -6.0149 1.800e-09 ***
sma1 -0.462673   0.099281 -4.6602 3.159e-06 ***
sma2 -0.233735   0.094450 -2.4747   0.01333 *
```

```
ARIMA (2,1,0)(3,1,0)
     Estimate  Std. Error z value  Pr(>|z|)
ar1 -0.728135   0.075642 -9.6261 < 2.2e-16 ***
ar2 -0.449319   0.076842 -5.8473 4.995e-09 ***
sar1 -0.446675   0.090184 -4.9530 7.309e-07 ***
sar2 -0.449873   0.093077 -4.8333 1.343e-06 ***
sar3 -0.344883   0.112617 -3.0624  0.002195 **
```

```
ARIMA (2,1,0)(3,1,2)
     Estimate Std. Error z value  Pr(>|z|)
ar1  -0.659417   0.087105 -7.5704 3.722e-14 ***
ar2  -0.406662   0.083983 -4.8422 1.284e-06 ***
sar1  0.488876   0.381359  1.2819 0.199866
sar2 -0.478714   0.190607 -2.5115 0.012021 *
sar3 -0.064972   0.210185 -0.3091 0.757232
sma1 -0.991825   0.376146 -2.6368 0.008369 **
sma2  0.459873   0.309310  1.4868  0.137076
```

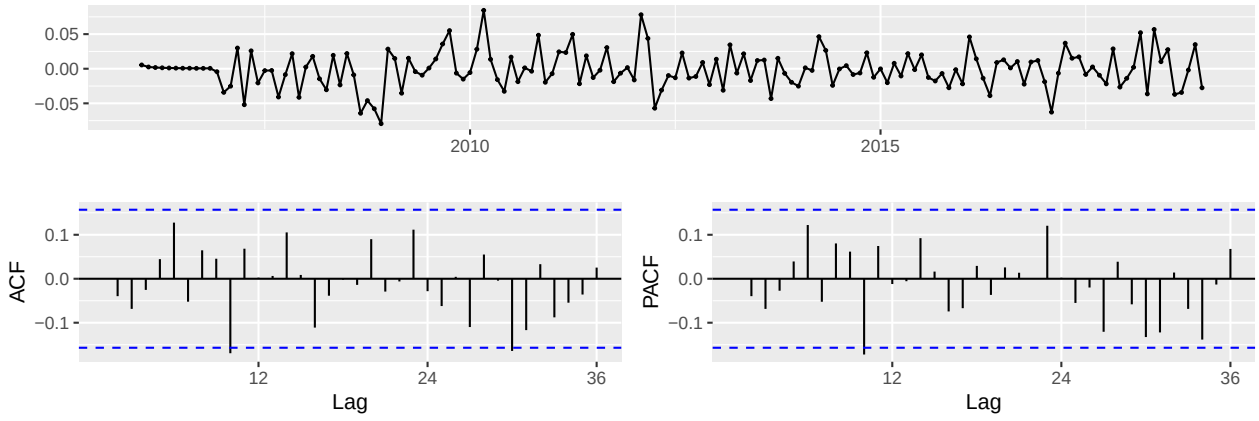
Figure 2: Importancia de los coeficientes en los modelos ARIMA (II)

Observamos que el modelo ARIMA, tanto con componente autoregresiva como media móvil, continua sin ser la mejor alternativa, dada la poca importancia que presentan la mayoría de sus coeficientes. Por el contrario, empleando únicamente la componente autoregresiva o media móvil, su importancia mejora significativamente. No obstante, **un modelo ARIMA, con tan solo la diferenciación y la componente autoregresiva, continua siendo la mejor alternativa**, principalmente por dos motivos:

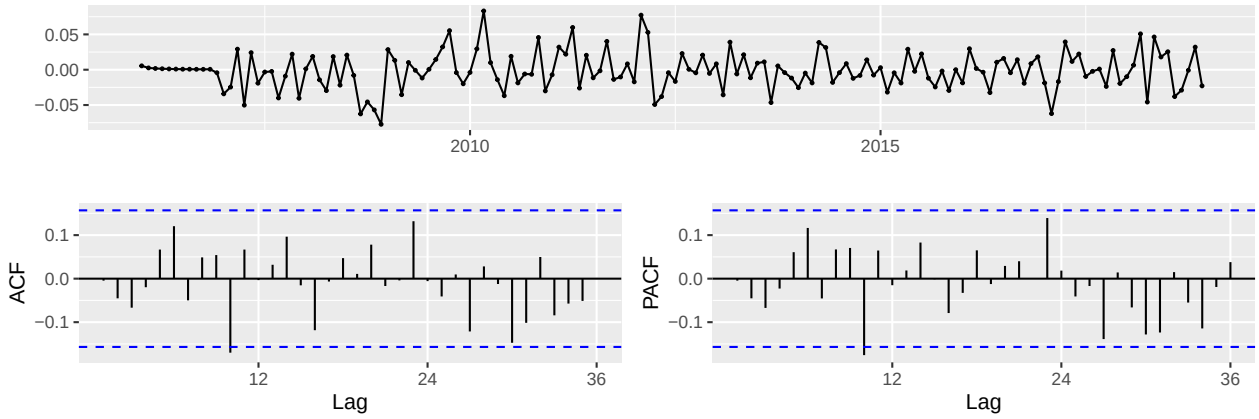
- Menor número de parámetros, en comparación con el modelo ARIMA (3,1,0)
- Mayor p-valor en el *test* de *Lljung-Box*, reforzando aún más la veracidad de la hipótesis nula (ruidos incorrelados)

No obstante, antes de decidir cual es el modelo ganador, representamos gráficamente la distribución de los residuos de ambos modelos:

```
ggtsdisplay(residuals(fitARIMA.1)) # ARIMA (2,1,0) (0,1,2)
```




```
ggtsdisplay(residuals(fitARIMA.1.1)) # ARIMA (2,1,0) (3,1,0)
```



A la vista de los resultados obtenidos en ambos casos, **los retardos múltiplos de la estacionalidad (12, 24 y 36) se sitúan dentro del umbral de aceptación**, por lo que la componente estacional parece estar, aparentemente, ajustada. Por el contrario, en ambos modelos ARIMA nos encontramos con un retardo, al comienzo de la función de autocorrelación, que supera el umbral definido. Esto último puede suponer que sea necesario añadir componentes a la parte regular del modelo. Dado que los únicos coeficientes significativos en la función de autocorrelación parcial corresponden con los dos primeros retardos, no tendría sentido aumentar el orden de media móvil. Por el contrario, si podríamos incrementar el orden autoregresivo de ambos modelos, concretamente a orden 3 (correspondientes con los primeros retardos más significativos):

```
fitARIMA.2<-arima(transf.box.cox,order = c(2,1,3), seasonal=c(0,1,2)) # ARIMA (2,1,3) (0,1,2)
fitARIMA.2.1<-arima(transf.box.cox,order = c(2,1,3), seasonal=c(3,1,0)) # ARIMA (2,1,3) (3,1,0)
```

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	AIC	BIC	p-valor
ARIMA (2,1,3)(0,1,2)	-0.001	0.025	0.018	-0.013	0.180	0.108	-0.030	-613.011	-589.308	0.948
ARIMA (2,1,3)(3,1,0)	-0.001	0.024	0.018	-0.013	0.182	0.109	-0.041	-612.698	-586.032	0.887

Analizando la salida obtenida, pese al aumento en los valores AIC y BIC, **el hecho de añadir una media móvil de orden 3 en ambos modelos ha supuesto una mejoría significativa**, principalmente en dos aspectos:

- En primer lugar, **el valor del Error absoluto medio porcentual o MAPE**, donde en ambos casos ha disminuido (de 0.204 a 0.180, así como 0.203 a 0.182, respectivamente).
- Sin embargo, **la principal mejoría se ve reflejada en el p-valor**, aumentando considerablemente en ambos modelos:
  - ARIMA (2,1,3)(0,1,2): de 0.47 a 0.94
  - ARIMA (2,1,3)(3,1,0): de 0.37 a 0.88

No obstante, si nos fijamos en la importancia de los parámetros:

```
coeftest(fitARIMA.2); coeftest(fitARIMA.2.1)
```

```
ARIMA (2,1,3)(0,1,2)                                                ARIMA (2,1,3)(3,1,0)
z test of coefficients:

     Estimate    Std. Error z value   Pr(>|z|)      Estimate  Std. Error   z value  Pr(>|z|)
ar1 -1.1686527  0.0074821  -156.1928 < 2.2e-16 ***  ar1 -1.1677240  0.0072746 -160.5204 < 2.2e-16 ***
ar2 -0.9981428  0.0031772  -314.1548 < 2.2e-16 ***  ar2 -0.9985627  0.0027897 -357.9525 < 2.2e-16 ***
ma1  0.5769293  0.0690274  8.3580    < 2.2e-16 ***  ma1  0.5754546  0.0689334    8.3480 < 2.2e-16 ***
ma2  0.3520325  0.0789287  4.4601    8.191e-06 ***  ma2  0.3494064  0.0792407    4.4094 1.036e-05 ***
ma3 -0.5666082  0.0668679  -8.4735   < 2.2e-16 ***  ma3 -0.5686437  0.0665060   -8.5503 < 2.2e-16 ***
sma1 -0.5059360 0.1037074  -4.8785   1.069e-06 ***  sar1 -0.5140580 0.0936740   -5.4877 4.071e-08 ***
sma2 -0.1571157 0.1021866  -1.5375   0.1242         sar2 -0.4597150 0.0986025   -4.6623 3.127e-06 ***
                                                   sar3 -0.1851065 0.0975304   -1.8979    0.0577 .
---------------------------------------------------------------------------------------------------
```

Figure 3: Importancia de los coeficientes en los modelos ARIMA (III)

Observamos que tanto la segunda media móvil como la tercera componente autoregresiva no son parámetros significativos en sus respectivos modelos, por lo que debemos descartarlos:

```
fitARIMA.2<-arima(transf.box.cox,order = c(2,1,3), seasonal=c(0,1,1)) # ARIMA (2,1,3) (0,1,1)
fitARIMA.2.1<-arima(transf.box.cox,order = c(2,1,3), seasonal=c(2,1,0)) # ARIMA (2,1,3) (2,1,0)
```

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	AIC	BIC	p-valor
ARIMA (2,1,3)(0,1,1)	-0.001	0.025	0.018	-0.014	0.184	0.110	-0.028	-612.732	-591.992	0.850
ARIMA (2,1,3)(2,1,0)	-0.001	0.025	0.018	-0.012	0.185	0.111	-0.041	-611.206	-587.504	0.848

Una vez eliminados ambos parámetros, aunque el p-valor disminuya en ambos casos, el valor obtenido continua siendo muy significativo (del orden de 0.85). Además, si comprobamos nuevamente la importancia de los parámetros:

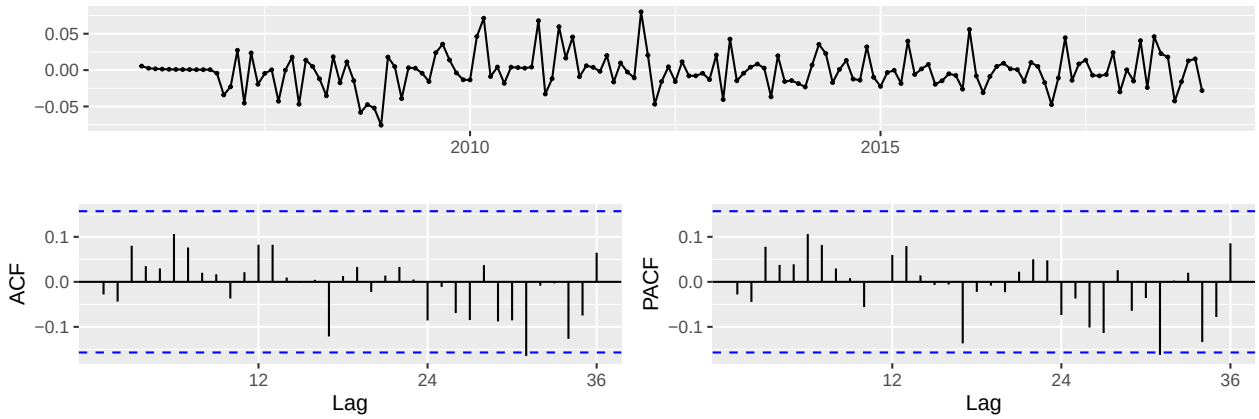
```
ARIMA (2,1,3)(0,1,1)                                    ARIMA (2,1,3)(2,1,0)
z test of coefficients:

     Estimate    Std. Error z value   Pr(>|z|)          Estimate  Std. Error   z value  Pr(>|z|)
ar1  -1.1683188  0.0064937 -179.9163 < 2.2e-16 ***  ar1  -1.1662704  0.0072069 -161.8270 < 2.2e-16 ***
ar2  -0.9988179  0.0021658 -461.1835 < 2.2e-16 ***  ar2  -0.9982861  0.0030221 -330.3273 < 2.2e-16 ***
ma1   0.5971246  0.0698892    8.5439 < 2.2e-16 ***  ma1   0.5813005  0.0694426    8.3709 < 2.2e-16 ***
ma2   0.3716549  0.0810181    4.5873 4.490e-06 ***  ma2   0.3674038  0.0786446    4.6717 2.987e-06 ***
ma3  -0.5484180  0.0673788   -8.1393 3.975e-16 ***  ma3  -0.5561234  0.0666288   -8.3466 < 2.2e-16 ***
sma1 -0.6155707  0.0859577   -7.1613 7.991e-13 ***  sar1 -0.4504021  0.0884115   -5.0944 3.499e-07 ***
                                                    sar2 -0.3732303  0.0886177   -4.2117 2.535e-05 ***
---------------------------------------------------------------------------------------------------
```

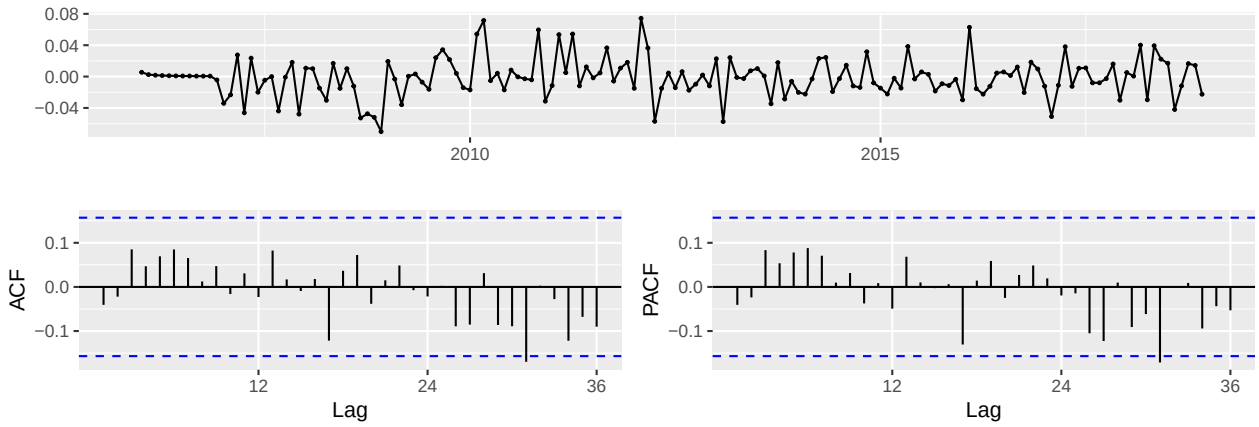
Figure 4: Importancia de los coeficientes en los modelos ARIMA (IV)

Podemos comprobar que todos los parámetros son prácticamente significativos en ambos modelos. Por otra parte, si analizamos los valores residuales en las funciones de autocorrelación:

```
ggtsdisplay(residuals(fitARIMA.2)) # ARIMA (2,1,3) (0,1,1)
```




```
ggtsdisplay(residuals(fitARIMA.2.1)) # ARIMA (2,1,3) (2,1,0)
```



Podemos comprobar como el retardo situado al comienzo de la función de autocorrelación se reduce drásticamente en ambos modelos, donde destaca únicamente un solo retardo en ambos modelos, aunque se situa en los límites del umbral de aceptación.

4.4 Selección y justificación del modelo ganador

Por tanto, ¿Qué modelo debemos escoger? En ambos casos, nos encontramos con dos modelos con valores de error muy similares. Sin embargo, pese a que el segundo modelo ARIMA presente valores AIC y BIC relativamente menores, la diferencia no es muy significativa, aún teniendo un mayor número de parámetros (-611 en comparación con -612 en

el AIC, así como -587 en comparación con -591 en el BIC). Por otro lado, el p-valor obtenido en el *test* de *Lljung-Box* proporciona una ligera ventaja al primer modelo ARIMA (0.849 frente a 0.847), incluso empleando un parámetro menos.

En relación con los valores residuales, a excepción de un valor de autocorrelación situado entre los retardos 24 y 36, **el resto se sitúan por debajo del umbral de aceptación, con valores muy similares en ambos modelos**. Por otro lado, ¿Y en relación con el modelo auto.arima? Es decir, el modelo ARIMA generado automáticamente ¿Supone mejoría alguna en relación con los modelo candidatos?

```
fitARIMA.auto <- auto.arima(transf.box.cox, seasonal = TRUE)
round(coef(fitARIMA.auto), 3) # Coeficientes del modelo auto.arima
```

```
##    ar1    ar2    ma1    ma2   sar1   sar2   sma1  drift
##  1.524 -0.561 -1.314  0.541  0.183 -0.313 -0.616  0.002
```

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	AIC	BIC	p-valor
ARIMA (2,1,3)(0,1,1)	-0.001	0.025	0.018	-0.014	0.184	0.110	-0.028	-612.732	-591.992	0.850
ARIMA (2,1,3)(2,1,0)	-0.001	0.025	0.018	-0.012	0.185	0.111	-0.041	-611.206	-587.504	0.848
auto.arima	0.000	0.026	0.020	0.001	0.205	0.526	0.004	-604.110	-577.381	0.100

Incluso el modelo automático, sin aplicar una diferenciación a la parte regular, no aporta mejoría alguna al modelo, principalmente por dos motivos:

- En primer lugar, tan solo aplicando una diferenciación en la componente estacional, todos los errores (a excepción del Error Medio o ME y el Error Porcentual Medio o MPE) aumentan con respecto a los otros dos modelos ARIMA.
- Por otro lado, el p-valor de los valores residuales es significativamente menor a los modelos ARIMA (0.1 frente a 0.84), por lo que, aunque estadísticamente no existiría evidencia en contra de que los residuos estén incorrelados, el mayor p-valor obtenido en los dos modelos anteriores, con una diferenciación en la parte regular, rechazan con "mayor fuerza" la hipótesis nula.

Es decir, el modelo *auto.arima*, en comparación con los modelos ARIMA generados manualmente, no aporta mejoría alguna. Finalmente, aplicando el principio de parsimonía: *en igualdad de condiciones, ante dos explicaciones de un suceso, la más sencilla suele ser la más probable*. Por tanto, **ante dos modelos ARIMA con resultados muy similares, nos decantamos por el modelo más sencillo**, concretamente el modelo ARIMA (2,1,3) (0,1,1):

```
fitARIMA.2 # ARIMA (2,1,3) (0,1,1)
```

```
##
## Call:
## arima(x = transf.box.cox, order = c(2, 1, 3), seasonal = c(0, 1, 1))
##
## Coefficients:
##           ar1      ar2     ma1     ma2      ma3     sma1
##       -1.1683  -0.9988  0.5971  0.3717  -0.5484  -0.6156
## s.e.   0.0065   0.0022  0.0699  0.0810   0.0674   0.0860
##
## sigma^2 estimated as 0.0006674:  log likelihood = 313.37,  aic = -612.73
```

Por tanto, el modelo ARIMA final será el siguiente:

$$(1-\phi_1 B-\phi_2 B^2)(1-B^{12})(1-B)X_t=(1-\Theta_1 B^{12})(1-\theta_1 B-\theta_2 B^2-\theta_3 B^3)Z_t$$

Con los parámetros estimados:

$$(1+1.1683B+0.9988B^2)(1-B^{12})(1-B)X_t=(1+0.6156B^{12})(1-0.5971B-0.3717B^2+0.5484B^3)Z_t$$

4.5 Predicción e intervalos de confianza

A continuación, una vez estimado el modelo ARIMA realizamos el cálculo de las predicciones e intervalos de confianza para los siguientes doce meses, es decir, para el año 2019, principalmente por dos motivos:

- Dado que los datos de la serie son mensuales, además de que disponemos de los valores del año 2019, **nos permitirá contrastar los resultados obtenidos en un periodo completo**, comparando las ventas previstas en cada mes con los valores reales en dicho año, junto con los obtenidos en el modelo de *Holt-Winters*.
- Por otro lado, aunque no es habitual realizar predicciones a medio/largo plazo, **una posibilidad podría haber sido no solo predecir los valores del año 2019, sino además del año 2020**, dado que desde el repositorio es posible recuperar dicho datos. Sin embargo, debido a la situación económica generada por la pandemia del COVID-19, las previsiones del modelo contrastarían con la "gran caída" en el número de ventas producido durante este año, lo que dificultaría en gran medida la medición de la calidad de los modelos.

Por tanto, **realizamos la predicción para los valores de venta del año 2019**:

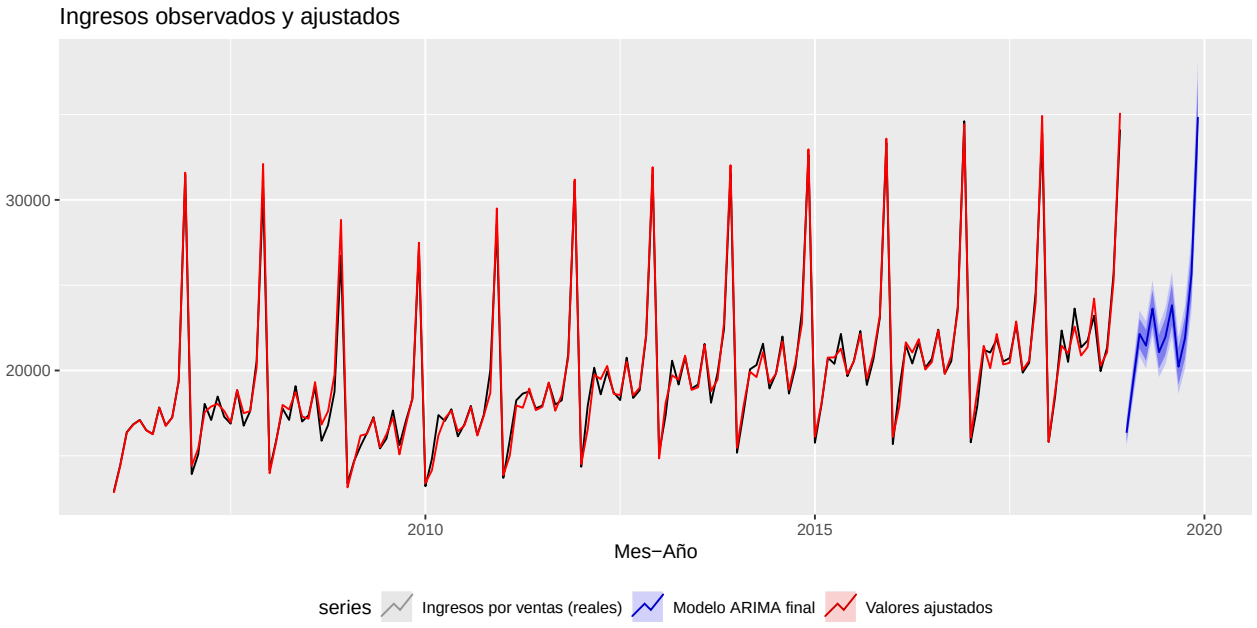
```
prediccion <- forecast(fitARIMA.2, h = 12)
```

Una vez realizada la predicción, debemos recordar que la serie que hemos empleado para la elaboración del modelo ARIMA **ha sido transformada mediante la función logarítmica**. Por tanto, y especialmente de cara al siguiente apartado, aplicamos la función *exp* a cada uno de los resultados obtenidos, tanto los valores predichos como sus intervalos de confianza al 80 y 95 %, con el objetivo de eliminar la transformación logarítmica:

```
prediccion$x <- exp(1) ** prediccion$x # Valores obtenidos en el entrenamiento
prediccion$mean <- exp(1) ** prediccion$mean # Valores de la prediccion para el año 2019
prediccion$lower <- exp(1) ** prediccion$lower;  prediccion$upper <- exp(1) ** prediccion$upper # Int. confianza
```

A continuación, mostramos tanto las predicciones e intervalos de confianza como su representación gráfica:

```
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## Jan 2019       16350.74 15814.81 16904.82 15538.25 17205.70
## Feb 2019       19295.49 18610.19 20006.03 18257.32 20392.69
## Mar 2019       22141.49 21288.33 23028.84 20850.09 23512.88
## Apr 2019       21461.75 20577.52 22383.98 20124.28 22888.11
## May 2019       23629.00 22603.45 24701.08 22078.70 25288.16
## Jun 2019       21073.35 20105.58 22087.70 19611.40 22644.29
## Jul 2019       21973.62 20916.92 23083.70 20378.26 23693.87
## Aug 2019       23823.78 22632.53 25077.73 22026.23 25768.03
## Sep 2019       20220.31 19164.58 21334.19 18628.21 21948.48
## Oct 2019       21893.49 20710.18 23144.41 20109.89 23835.28
## Nov 2019       25627.32 24197.71 27141.39 23473.50 27978.77
## Dec 2019       34855.90 32842.57 36992.65 31824.29 38176.31
```



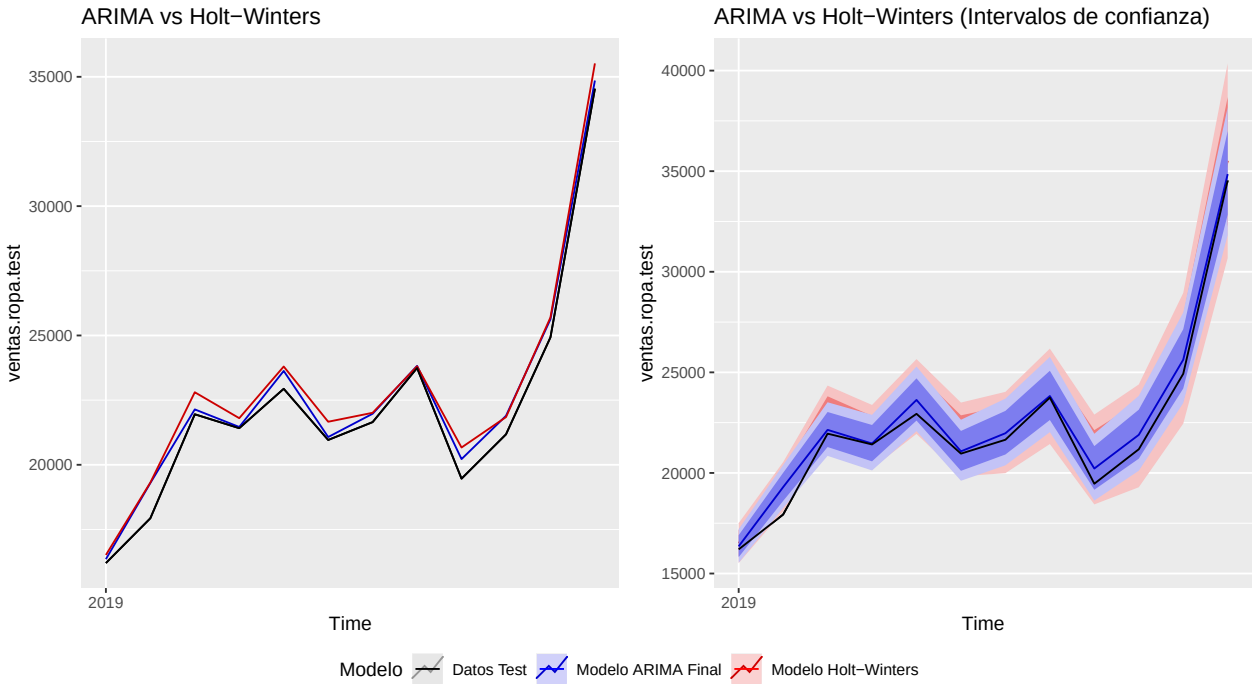
Tal y como podemos observar en el gráfico anterior, los valores de ajuste obtenidos con el conjunto de "entrenamiento" son bastante similares a las ventas reales. No obstante, en los periodos en los que hay un mayor descenso en el número de ventas, **las predicciones experimentan un mayor desajuste**, sobretodo en los "picos" de ventas correspondientes al mes de diciembre, donde los valores previstos son significativamente mayores a los ingresos reales. En relación con el resto de años, el ajuste se asemeja en mayor o menor medida a las ventas reales. Con respecto a los valores previstos, los resultados obtenidos gráficamente se asemejan a los obtenidos en el modelo *Holt-Winters*. Por ello, a simple vista no encontramos una diferencia significativa entre la previsión de ambos modelos, por lo que en el siguiente apartado no solo compararemos las predicciones gráficamente, sino también numericamente.

5. Comparación predicciones modelo ARIMA y suavizado exponencial. Conclusiones

Por último, realizamos una comparación de las predicciones, tanto gráfica como numericamente, a partir del conjunto de datos previamente reservado correspondiente al año 2019. En primer lugar, comparamos la diferencia entre los valores reales y previstos en ambos modelos:

```
##          ventas.ropa.test ARIMA (dif) Holt-Winters (dif)
## Jan 2019            16201   149.73572          312.21685
## Feb 2019            17932  1363.49236         1380.14167
## Mar 2019            21953   188.49366          851.54896
## Apr 2019            21416    45.75239          386.17856
## May 2019            22938   691.00213          861.02227
## Jun 2019            20960   113.35030          704.49438
## Jul 2019            21650   323.61842          360.00412
## Aug 2019            23743    80.77879           65.24272
## Sep 2019            19464   756.30825         1204.67902
## Oct 2019            21177   716.49009          667.03090
## Nov 2019            24928   699.32252          770.12757
## Dec 2019            34541   314.89898          977.76952
```

En primer lugar, analizando las diferencias entre los valores reales y previstos, observamos que las predicciones realizadas por el modelo ARIMA **se acercan en mayor medida a los valores originales prácticamente en todos los meses, a excepción de los meses de agosto y octubre, donde los valores obtenidos en el modelo *Holt-Winters* se acercan en mayor medida a las ventas reales**. Por otro lado, si analizamos gráficamente tanto las predicciones obtenidas como los intervalos de confianza:



No solo las predicciones del modelo ARIMA se aproximan en mejor medida a los valores reales, sino que además la

amplitud de los intervalos de confianza, tanto al 80 como al 95 %, **son mucho menores en comparación con el modelo de *Holt-Winters***. De hecho, si analizamos las estadísticas de error en cada modelo:

```
cbind(accuracy(prediccion, ventas.ropa.test),"AIC" = AIC(fitARIMA.2),"BIC" = BIC(fitARIMA.2))
```

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	Theil's U	AIC	BIC
Train	19705.162	20172.739	19705.162	99.948	99.948	26.614	0.102	NA	-612.732	-591.992
Test	-453.604	591.207	453.604	-2.165	2.165	0.613	-0.288	0.199	-612.732	-591.992

```
cbind(accuracy(ventas.ropa.hw, ventas.ropa.test),"AIC" = ventas.ropa.hw$model$aic,
      "BIC" = ventas.ropa.hw$model$bic)
```

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	Theil's U	AIC	BIC
Train	-12.558	585.568	460.540	-0.074	2.347	0.622	-0.070	NA	2796.921	2848.768
Test	-711.705	800.697	711.705	-3.301	3.301	0.961	-0.227	0.251	2796.921	2848.768

Observamos que un intervalo de confianza más pequeño ofrece resultados más precisos, aunque aumenta su probabilidad de error. Esto último lo podemos observar en los valores de error del conjunto de entrenamiento, **los cuales son significativamente mayores en el modelo ARIMA en comparación con el modelo *Holt-Winters***. Sin embargo, al contrastar los valores de error en el conjunto de prueba, cambian las tornas completamente: no solo los valores de error son mucho más pequeños en el modelo ARIMA, sino además del coeficiente de incertidumbre, conocido como *Theil's U* (0.199 frente a 0.251), encargado de medir la exactitud de un pronóstico, de forma que cuanto más cercano a 0, más precisa será la predicción. Por otro lado, los criterios de error AIC y BIC también muestran una mayor ventaja del modelo ARIMA frente al modelo de *Holt-Winters*:

- AIC: -612 frente a 2795
- BIC: -591 frente a 2848

Además, incluso si comparamos el p-valor de los valores residuales en ambos modelos:

```
p.valor.arima <- checkresiduals(fitARIMA.2, plot = FALSE)$p.value # Modelo ARIMA final
```

```
## [1] 0.8497833
```

```
p.valor.hw <- checkresiduals(ventas.ropa.hw, plot = FALSE)$p.value # Modelo Holt-Winters
```

```
## [1] 2.914335e-13
```

La diferencia es muy significativa: 0.85 frente a un p-valor extremadamente pequeño (del orden de 10^{-13}). Por tanto, y en base a los resultados obtenidos, **el modelo ARIMA presenta un mejor resultado en todos los aspectos**:

- En primer lugar, no solo unas predicciones mucho más cercanas a los valores reales, sino además una amplitud significativamente menor en sus intervalos de confianza, tanto al 80 como al 95 %.
- Por otro lado, pese a que el error obtenido en el conjunto de entrenamiento ha sido mucho mayor en el modelo ARIMA, los valores de error en el conjunto de prueba marcan la diferencia, con errores mucho más pequeños. Del mismo modo sucede con los criterios AIC y BIC.
- Por su parte, el criterio de incertidumbre muestra un valor mucho más cercano a 0, lo que evidencia que los pronósticos obtenidos en el modelo ARIMA son más fiables en el modelo de *Holt-Winters*.
- Además, el p-valor obtenido en la función *checkresiduals* demuestra la clara incorrelación del ruido en el modelo ARIMA, en contraposición al modelo de *Holt-Winters*, donde el p-valor se sitúa por debajo de 0.05, es decir, rechazamos la hipótesis nula de que la serie presenta "ruido blanco".

Como conclusión final, pese a la mejoría que supone el modelo ARIMA, **los valores de error obtenidos continuan siendo demasiado elevados**, teniendo en cuenta que las unidades de medida se miden en el orden de "millones de dólares", por lo que cabría destacar, como posible futura línea de investigación, contrastar dicho modelo ARIMA con otras técnicas predictivas, tales como algoritmos de *Machine Learning*, redes neuronales, algoritmos genéticos, e incluso la inclusión de variables externas al modelo, derivando en una serie temporal multivariable.