

**FORECASTING**  
**AJUSTE DE UN MODELO**  
**SARIMA A UNA SERIE**  
**REAL**  
**PREDICCIONES**  
**MENSUALES**

## CONTENIDO

1	INTRODUCCIÓN .....	3
2	IDENTIFICACIÓN .....	3
2.1	REPRESENTACIÓN Y EXPLORACIÓN INICIAL DE LA SERIE .....	4
2.2	TRANSFORMACIONES PARA ESTABILIZAR LA VARIANZA $X_t = g(X_t)$ .....	7
2.3	TRANSFORMACIONES PARA ESTABILIZAR LA MEDIA DE LA SERIE .....	9
2.4	IDENTIFICACIÓN FINAL: GRADOS POLINOMIALES .....	14
3	ESTIMACIÓN Y SELECCIÓN DE MODELOS .....	17
3.1	ESTIMACIÓN DE MODELOS (Peña, capítulo 10: páginas 285-304) .....	17
3.2	SELECCIÓN DE MODELOS (Peña, capítulo 10: páginas 305-311, 317-319) .....	20
	<i>CRITERIOS AIC DE AKAIKE, AICc y BIC</i> .....	20
	<i>CROSS VALIDATION</i> .....	24
	<i>DATA-SPLITTING</i> .....	26
4	DIAGNOSIS .....	30
4.1	¿LOS RESIDUOS SON RUIDO ALEATORIO? .....	31
	<i>Contraste basado en la ACF muestral</i> .....	32
	<i>Contraste de Ljung-Box</i> .....	33
	<i>Contraste Turning Point Test</i> .....	34
	<i>Contraste Difference sign test</i> .....	35
4.2	¿LOS RESIDUOS TIENEN DISTRIBUCIÓN NORMAL? .....	37
5	AJUSTE DEL MODELO Y PREDICCIONES .....	39
6	SOBRE STRATEBI .....	41

## 1 INTRODUCCIÓN

En este documento creado por Jose Antonio Esteban de Stratebi se aplicará el método Box and Jenkins para el análisis de una serie temporal, en concreto, para las ventas de comida para llevar (en billones) en Australia durante el periodo abril/1980-abril/2014. Para ello nos serviremos del software estadístico R. Cabe decir que los modelos SARIMA están recomendados fundamentalmente para las predicciones mensuales, ya que las predicciones mensuales poseen habitualmente una estacionalidad simple y regular (única y sin alteraciones irregulares provocadas por factores añadidos como los festivos). Sin embargo, las series diarias o semanales suelen poseer estacionalidad compleja (e.g., semanal y anual) e irregular (más de una y con alteraciones). Por tanto, recomiendo los modelos SARIMA únicamente para los casos en los que se trate una serie de datos mensuales.

Es importante aclarar que, aunque para la mayoría de las partes del artículo no sea necesario, se asume que se disponen de los conocimientos adquiridos en los **11 primeros capítulos del libro de Daniel Peña, Análisis de Series temporales**. Este análisis no es otra cosa que la culminación de todos esos capítulos.

El método Box and Jenkins consiste en un procedimiento de ajuste dividido en 3 etapas:

1. **Identificación** de los posibles modelos SARIMA que podría seguir la serie. Esto es, identificar los parámetros  $(p, d, q) \times (P, D, Q)_s$  a partir de sus gráficas ACF y PACF (estas son, las gráficas de autocorrelación simple y parcial).
2. **Estimación y selección de los modelos**. Donde la estimación es la obtención de los parámetros constantes polinomiales y la varianza del ruido aleatorio involucrado.
3. **Diagnos**. Consiste en validar aquellos supuestos sobre los que construimos o ajustamos los modelos.

## 2 IDENTIFICACIÓN

Supongamos que dada una serie temporal  $\{X_t\}$  donde  $t$  son los tiempos (normalmente en meses) se ha observado una realización de longitud  $T$  de la serie, i.e., tenemos los datos  $\{x_1, x_2, \dots, x_T\}$ . Si  $T = 12$ , por ejemplo, tendríamos la observación correspondiente al mes después de un año de transcurso.

Una serie suele verse únicamente afectadas por 3 componentes:

1. **Varianza** (que comprende, en particular, la amplitud de las oscilaciones de la gráfica)
2. **Tendencia** (la media general de la serie)
3. **Estacionalidad** (en caso de ser una estacionalidad anual,  $s = 12$ , la relación existente entre los mismos meses de distintos años)

Estas 3 componentes las recoge el modelo SARIMA cuyas siglas significan "Seasonal Autorregresive Integrated Moving Average" y cuya expresión matemática es:

$$\Phi_P(B^s)\phi_p(B)\nabla_s^D\nabla^dX_t = c + \theta_q(B)\Theta_Q(B^s)Y_t$$

Donde

1.  $X_t$  es nuestra serie temporal,
2.  $\Phi_P(B^s)$  y  $\Theta_Q(B^s)$  son los polinomios de retardo de la parte estacional,
3.  $\phi_p(B)$  y  $\theta_q(B)$  son los polinomios de retardo de la parte regular,
4.  $\nabla_s^D, \nabla^d$  son los operadores diferencia estacional y regular, respectivamente.
5.  $Y_t$  es un ruido aleatorio con distribución normal de media 0 y varianza  $\sigma^2$ .
6. y  $c$  es una constante.

Esquemataremos la parte relativa a la **identificación** en la que estamos ahora interesados:

1. **Objetivo:** Identificar un conjunto de modelos compatibles con el gráfico de la serie  $\{X_t\}$  y sus funciones de autocorrelación simple ACF y parcial PACF.
2. **Este objetivo requiere:**
  - 2.1. Identificar la estructura NO ESTACIONARIA y decidir qué transformaciones aplicar para convertir la serie original en una serie estacionaria:
    - 2.1.1. Transformaciones para estabilizar la varianza:  $\hat{X}_t = g(X_t)$
    - 2.1.2. Transformaciones para estabilizar la media:  $\Rightarrow D, s, d \Rightarrow W_t = \nabla_s^D\nabla^d\hat{X}_t$
  - 2.2. Determinar un modelo ARMA para la serie estacionaria  $\nabla_s^D\nabla^d\hat{X}_t \Rightarrow p, q, P, Q$

En resumen, el método Box-Jenkins consiste en obtener una serie transformación estacionaria. La serie  $(\nabla_s^D\nabla^dX_t)$  es **estacionaria**. La estacionariedad es la cualidad deseada en una serie temporal, pues se puede reconocer dentro de un grupo de series conocido como series ARMA (estacional multiplicativo) (estacionales o no). De este grupo de series se tiene un conocimiento extenso de sus gráficas ACF y PACF (función de autocorrelación parcial) con el que se pueden identificar los parámetros  $(p, q) \times (P, Q)_s$  (identificación).

NOTA: estacionariedad y estacionalidad son conceptos distintos.

## 2.1 REPRESENTACIÓN Y EXPLORACIÓN INICIAL DE LA SERIE

Lo primero lógicamente es cargar los datos:

```
1. # Datos 9: Ventas de comida para llevar (en billones) en
   # Australia,
2. # abril 1980 - abril 2014, n=408, fichero: datos9.txt
3.
4. #####
5. # REPRESENTACIÓN GRÁFICA #
6. #####
7.
```

```

8. > library(forecast)
9. > VENTAS <- read.table("Datos9.txt",header=T)
10. > class(VENTAS)
11. [1] "data.frame"
12. > VENTAS <- VENTAS$x # x es el header del data.frame
13. > class(VENTAS)
14. [1] "numeric"
15. > head(VENTAS,10)
16. [1] 0.3424 0.3421 0.3287 0.3385 0.3315 0.3419 0.3584 0.3747 0.4
    331
17. [10] 0.3686
18. > L <- length(VENTAS)
19. > L
20. [1] 408
21. > L/12 # 34 años
22. [1] 34

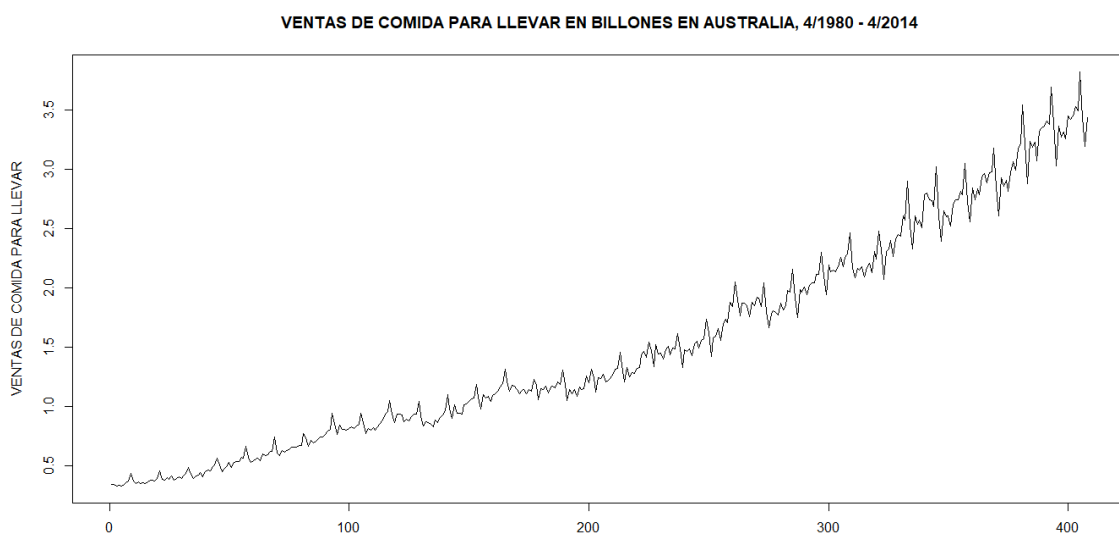
```

A continuación, representamos la gráfica:

```

1. plot(1:L, VENTAS, ylab="VENTAS DE COMIDA PARA LLEVAR",
2.       xlab="",type="l",
3.       main="VENTAS DE COMIDA PARA LLEVAR EN BILLONES EN AUSTRALIA,
    4/1980 - 4/2014")

```



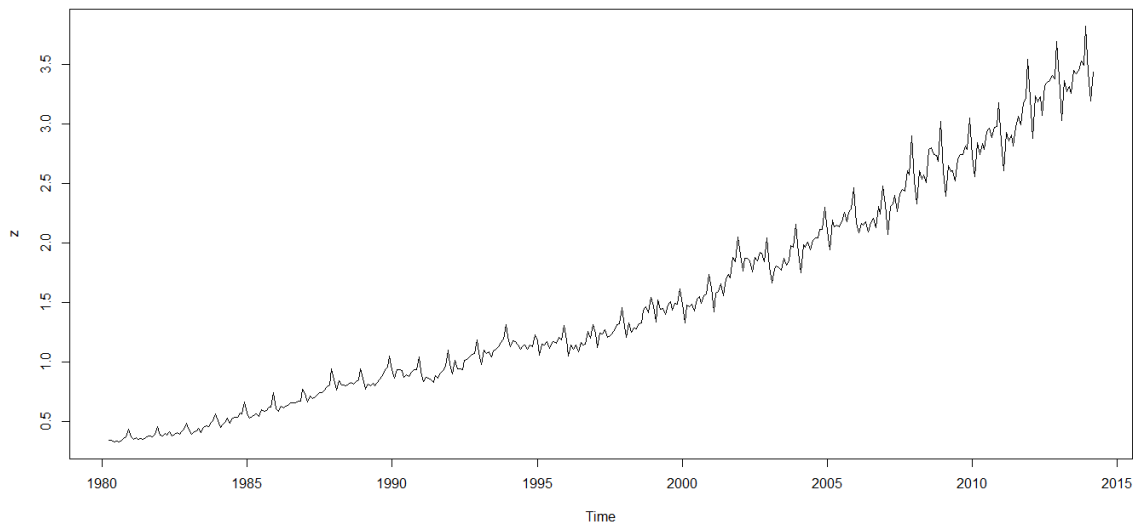
O, también, si queremos representar la serie con fechas en el eje de abscisas podemos usar los objetos ts:

```

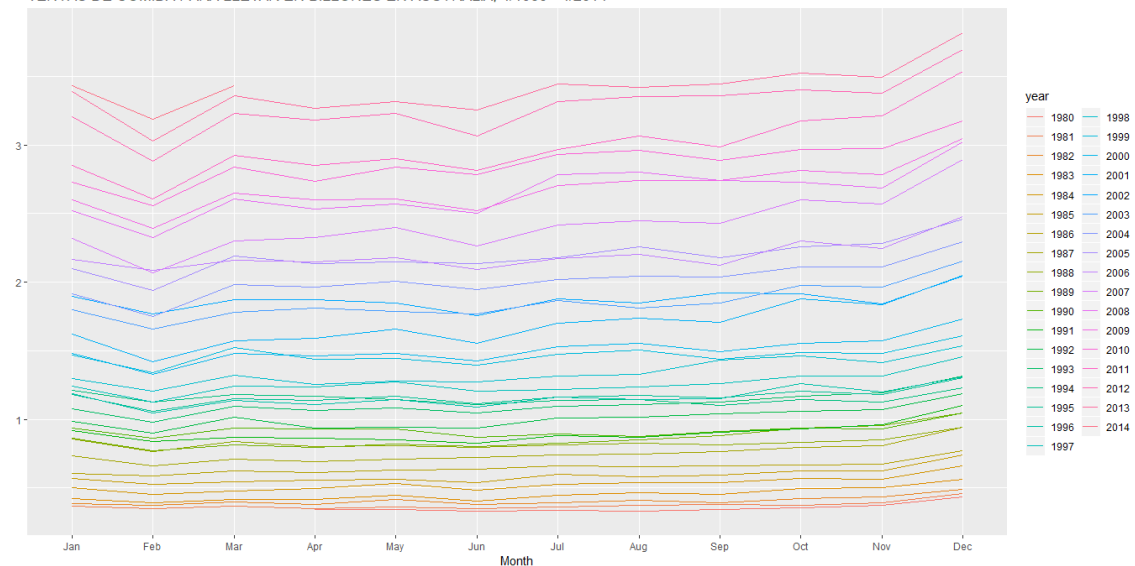
1. > z <- ts(VENTAS, start = c(1980, 4), frequency = 12) #creamos
    un objeto .ts
2. > ts.plot(z, main="VENTAS DE COMIDA PARA LLEVAR EN BILLONES EN
    AUSTRALIA, 4/1980 - 4/2014")
3. > ggseasonplot(z,main="VENTAS DE COMIDA PARA LLEVAR EN BILLONES
    EN AUSTRALIA, 4/1980 - 4/2014")

```

VENTAS DE COMIDA PARA LLEVAR EN BILLONES EN AUSTRALIA, 4/1980 - 4/2014



VENTAS DE COMIDA PARA LLEVAR EN BILLONES EN AUSTRALIA, 4/1980 - 4/2014



¿Qué nos sugieren estas dos representaciones iniciales de la serie?

Nos sugieren, sin atisbo de duda, tres hechos:

1. Un aumento de la varianza  $\sigma_t^2$  con el nivel de la serie (la media  $\mu_t$ ).
2. La presencia de una componente de tendencia al menos lineal o tal vez cuadrática.
3. El ggseasonplot nos sugiere una estacionalidad. Por ejemplo, para cada año, y dentro del crecimiento general de la serie, es indudable que en febrero siempre se produce un decrecimiento que evidencia la estacionalidad. Asimismo, esto ocurre para el resto de meses. En definitiva, lo que nos debe hacer sospechar de la estacionalidad es que la serie posea una especie de patrón anual.

Estos 3 inconvenientes hay que resolverlos, y se resuelven por medio de transformaciones.

## 2.2 TRANSFORMACIONES PARA ESTABILIZAR LA VARIANZA $\hat{X}_t = g(X_t)$

Lo primero es estabilizar la varianza. Para esto siempre hay solución si se verifica que

$$\sigma_t^2 = k\mu_t^{1-\lambda}$$

Equivalente a:

$$\log\sigma_t = a + (1 - \lambda)\log\mu_t, \quad a > 0$$

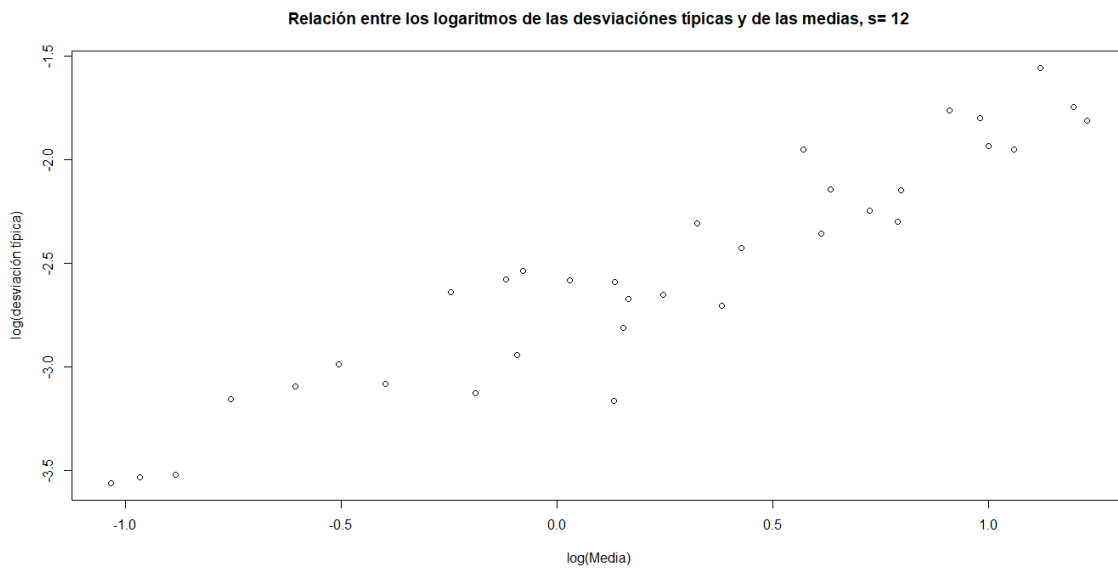
con  $k, a$  constantes (nótese que esta dependencia comprende numerosas relaciones entre la varianza y el nivel de la serie). Notemos que la condición suficiente para que se dé esta relación es que exista una relación lineal entre el logaritmo de la varianza de la serie y el logaritmo del nivel de la serie. Entonces, la llamada transformación de Box-Cox (Págs. 256-260, Daniel Peña) es de varianza constante:

$$\hat{X}_t = \begin{cases} \frac{X_t^\lambda - 1}{\lambda} & \text{si } \lambda \neq 0 \\ \log(X_t) & \text{si } \lambda = 0 \end{cases}$$

Y con esto resolvemos el problema de la varianza. Es habitual encontrarse con modelos en los que las transformaciones de Box-Cox no resuelvan el problema de la heterocedasticidad (varianza no constante). En estos casos, podríamos recurrir a los modelos ARCH/GARCH si la estacionalidad no supone un problema.

Comprobemos si podemos emplear Box-Cox:

```
1. # Determinación de la transformación para estabilizar la
   # varianza
2.
3. source ("BoxCoxTransformation.R", encoding = "UTF8")
4.
5. # Primero hay que comprobar que existe una relación lineal entre
6. # los logaritmos de las desviaciones típicas y de las medias en
   # grupos consecutivos de "s" observaciones
7. Plot.var (VENTAS, 12)
```

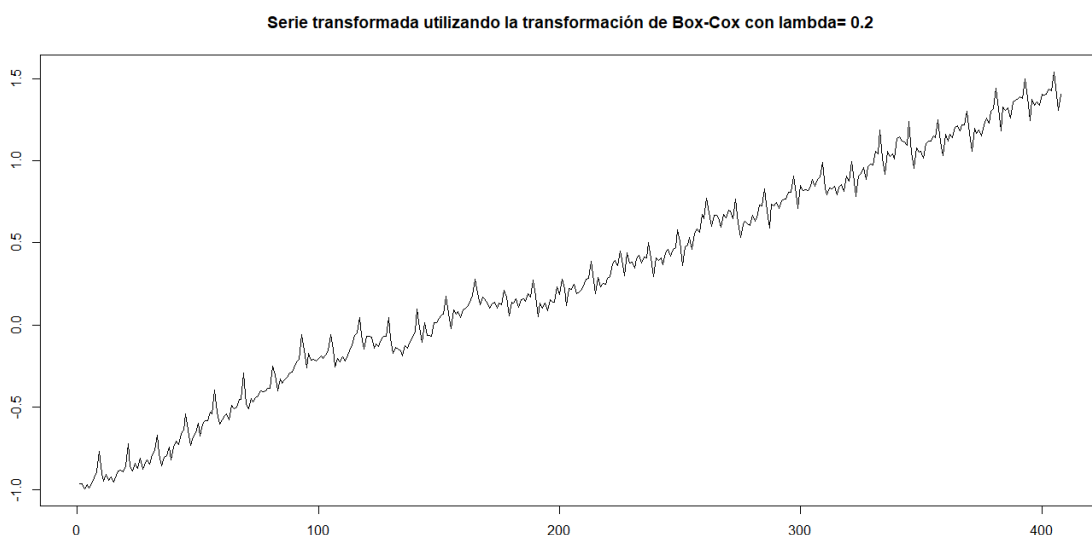


Puesto que tenemos una clara relación lineal entre los logaritmos podemos emplear Box-Cox:

```

1. > BoxCox (VENTAS, 12)
2. > BoxCox.out (VENTAS, 12) [[2]]
3.   log(xbar.vec)
4.   0.2031995
5. > BCVENTAS <- BoxCox.out (VENTAS, 12) [[1]]
6. > zBC <- ts(BCVENTAS, start = c(1980, 4), frequency = 12)
7. > ts.plot(zBC, main="TRANSFORMACIÓN DE BOX-COX DE LAS VENTAS DE
   COMIDA PARA LLEVAR EN BILLONES EN AUSTRALIA, 4/1980 - 4/2014")

```



Por lo tanto, la primera transformación de la serie original para obtener una varianza constante es:

$$\hat{X}_t = \frac{X_t^\lambda - 1}{\lambda}, \text{ con } \lambda = 0.2$$



Consideramos que 0.2 es un valor no suficientemente cercano a 0 como para contemplar el uso de la transformación logarítmica.

## 2.3 TRANSFORMACIONES PARA ESTABILIZAR LA MEDIA DE LA SERIE

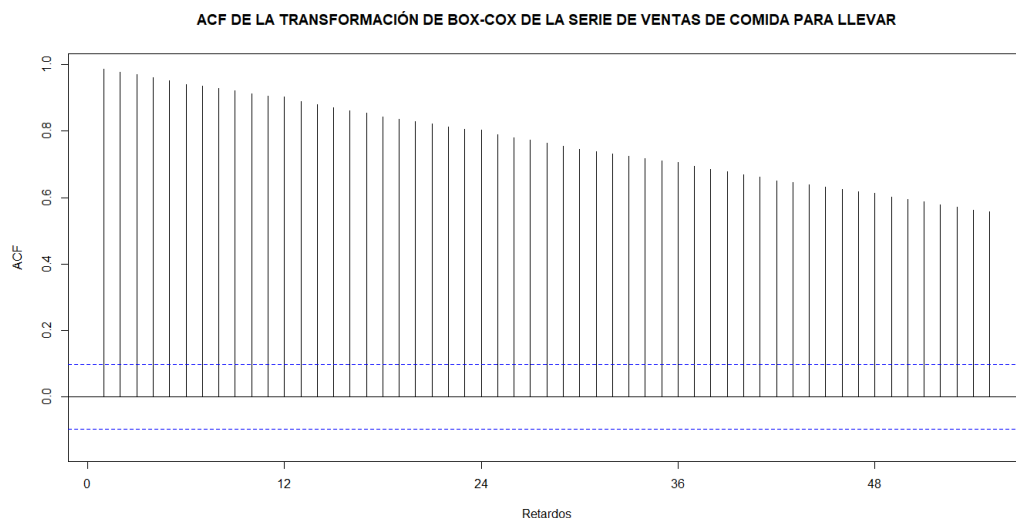
### 1) ELIMINACIÓN DE LA TENDENCIA GENERAL:

Hemos resuelto el problema de la varianza, ahora nos toca resolver el problema de la presencia de tendencia. Para ello tenemos que diferenciar la serie regularmente, esto es, aplicar el operador

$$\nabla^d = (1 - B)^d$$

Donde  $B$  es el operador de retardo, i.e.,  $BX_t = X_{t-1}$ . La forma de confirmar la presencia de tendencia en la serie (aunque sea evidente en nuestro caso) consiste en representar la ACF muestral de la serie en cuestión:

```
1. Acf(zBC,main="ACF DE LA TRANSFORMACIÓN DE BOX-COX DE LA SERIE DE VENTAS DE COMIDA PARA LLEVAR",xlab="Retardos",ylab="ACF",55)
```



Esto nos confirma que tenemos que diferenciar regularmente la serie ya que la ACF muestral presenta un decrecimiento lento y aproximadamente lineal, algo característico de un proceso ARIMA. Por lo tanto, necesitamos diferenciar regularmente al menos una vez la serie, es decir, al menos ha de ser  $d \geq 1$ . Notemos que esto lo que significa es que existe al menos una raíz unitaria en la serie. No siempre que una serie posea una raíz unitaria necesariamente hay tendencia, ya que un paseo aleatorio es un  $I(1)$  y no presenta tendencia. Por otra parte, podemos notar un poco la presencia de estacionalidad en los múltiplos de 12 (aunque la apreciación visual es muy sutil en este caso). Todo esto nos confirma que nos encontramos frente a un modelo SARIMA.

Diferenciemos regularmente una vez y veamos su representación gráfica:

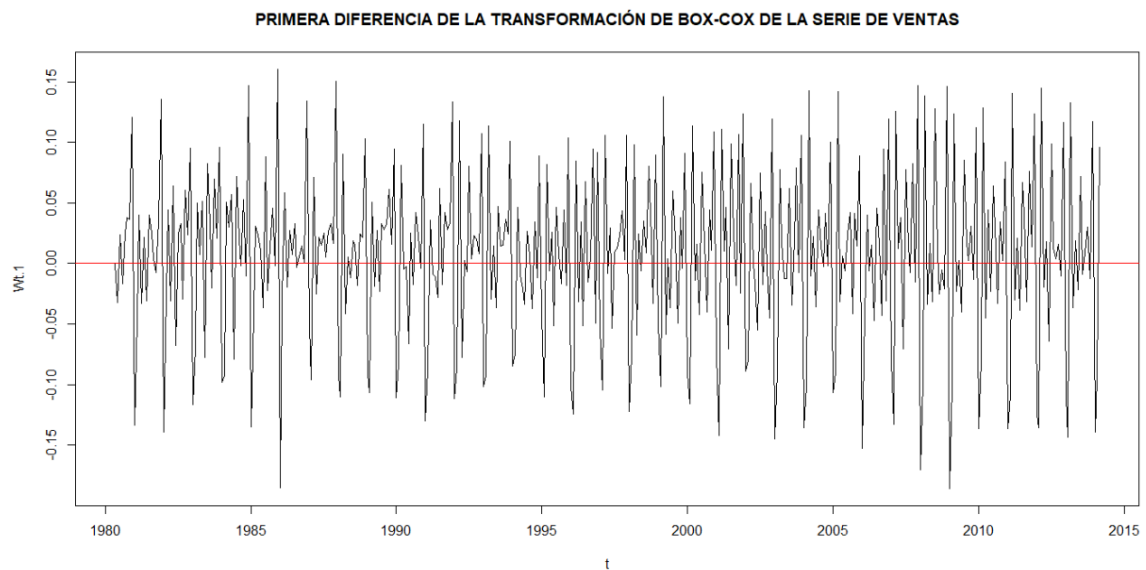
```
1. # Difference operator
```

```

2. #####
3. # diff(x, lag = 1, differences = 1, ...)
4.
5. Wt.1=diff(zBC,lag=1,differences=1)
6. ts.plot(Wt.1,xlab="t",main="PRIMERA DIFERENCIA DE LA
   TRANSFORMACIÓN DE BOX-COX DE LA SERIE DE VENTAS",type="l")
7. abline(h=0, col = "red")

```

Consultamos la representación:



A simple vista, parece que diferenciando regularmente una vez hemos conseguido estabilizar la media de la serie temporal en cuestión. No obstante, podemos confirmarlo de dos maneras:

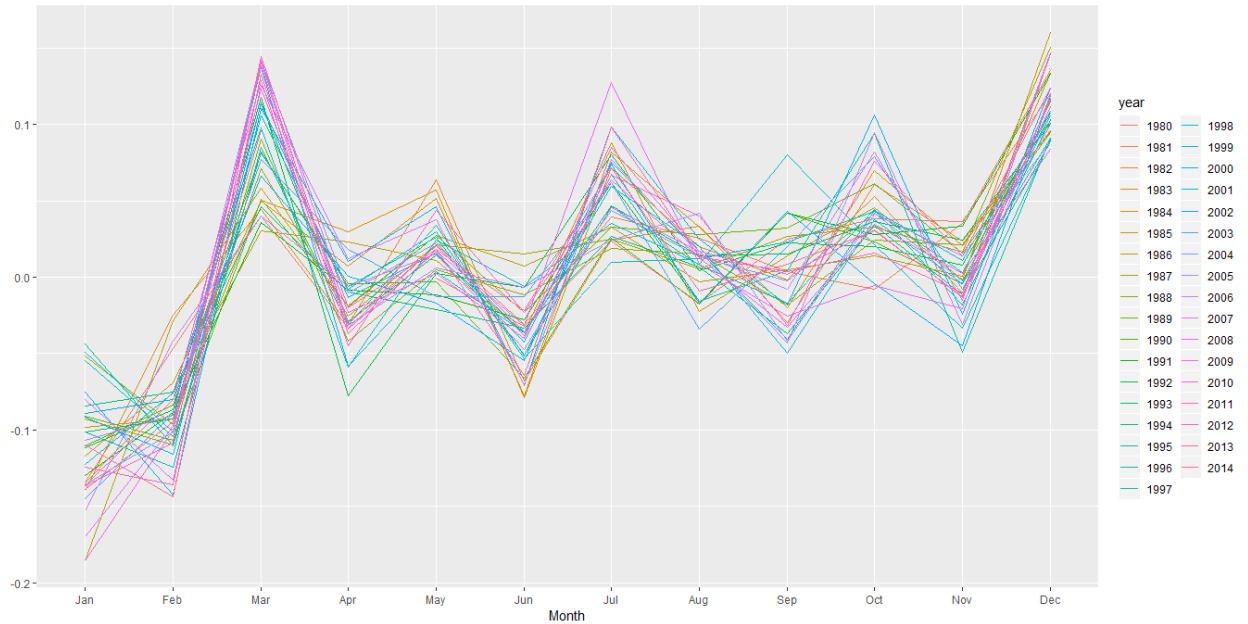
1. Podemos comprobar la ACF muestral y el ggseasonplot de la serie resultante:

```

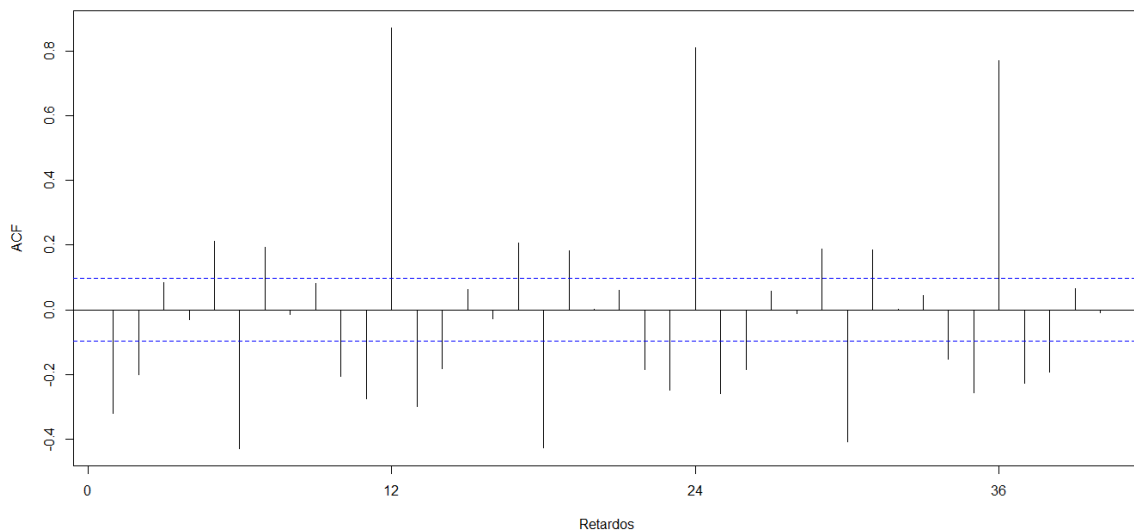
1. ggseasonplot(Wt.1,main="PRIMERA DIFERENCIA DE LA TRANSFORMACIÓN
   DE BOX-COX DE LA SERIE DE VENTAS")
2. Acf(Wt.1,main="ACF DE PRIMERA DIFERENCIA DE LA TRANSFORMACIÓN DE
   BOX-COX DE LA SERIE DE VENTAS", xlab="Retardos",ylab="ACF",40)

```

PRIMERA DIFERENCIA DE LA TRANSFORMACIÓN DE BOX-COX DE LA SERIE DE VENTAS



ACF DE PRIMERA DIFERENCIA DE LA TRANSFORMACIÓN DE BOX-COX DE LA SERIE DE VENTAS



Observemos que la ACF ya no presenta el decrecimiento aproximadamente lineal característico de los procesos  $I(d)$ . Además, en el ggseasonplot podemos apreciar cómo se agrupan en torno a un nivel general para la serie. Por lo tanto, podemos intuir que no necesitamos diferenciar más veces. Queda manifiesto aún los distintos niveles que caracterizan a los mismos meses de distintos años propios de la estacionalidad.

Notemos también que no tiene sentido interpretar el eje de abscisas como meses porque estamos diferenciando regularmente.

- Un método formal que se puede emplear para contrastar si hay presencia de raíces unitarias es el Test de Dickey-Fuller. *Análisis de Series Temporales (por Daniel Peña páginas 264-271)*.

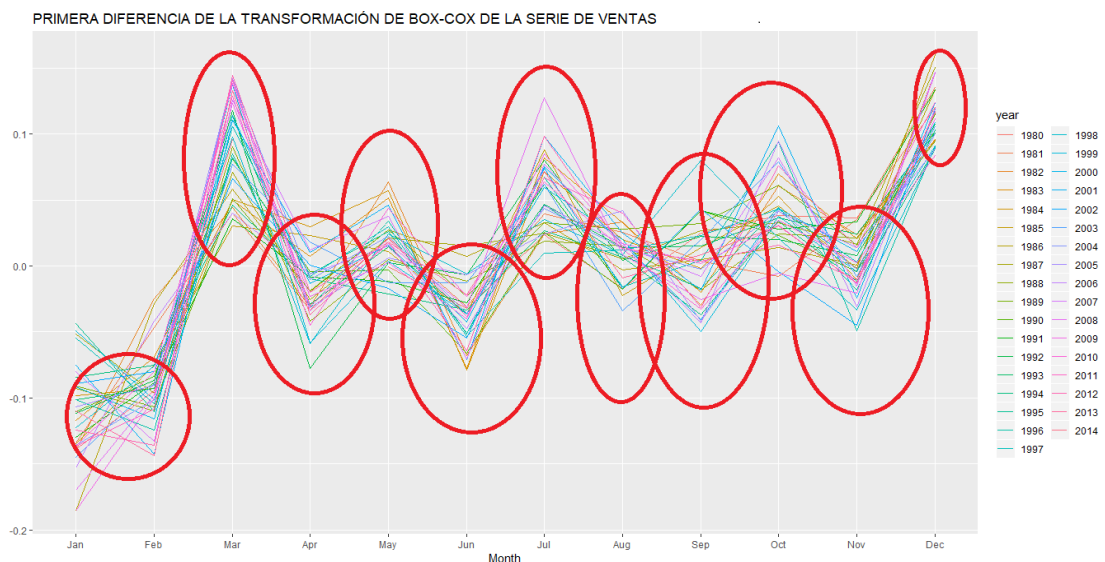
Perdemos una observación al tomar  $d = 1$ . Así, y hasta ahora, tenemos la transformación que sigue:

$$\nabla^d \hat{X}_t = \nabla \left( \frac{X_t^\lambda - 1}{\lambda} \right), \text{ con } \lambda = 0.2$$

## 2) ELIMINACIÓN DEL EFECTO ESTACIONAL:

En el mismo ggseasonplot evidenciamos que hay presente un efecto estacional:

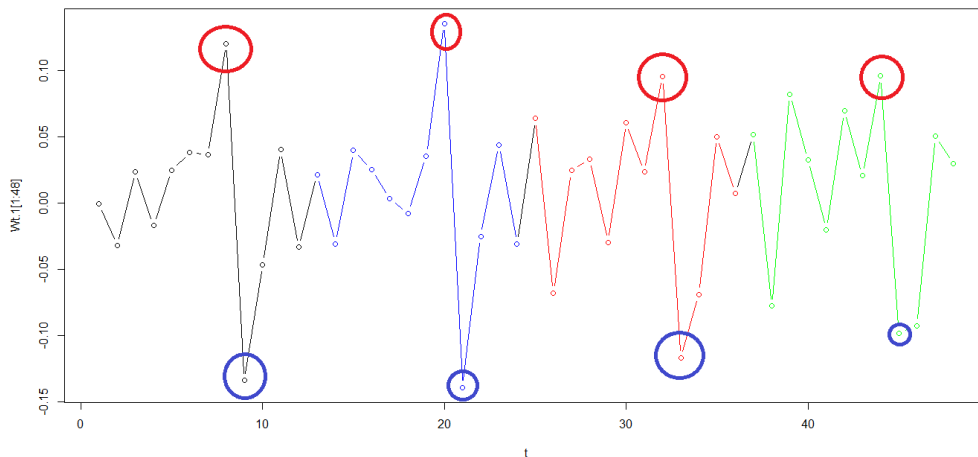
```
1. ggseasonplot(Wt.1, main="PRIMERA DIFERENCIA DE LA TRANSFORMACIÓN
  DE BOX-COX DE LA SERIE DE VENTAS")
2. ts.plot(Wt.1[1:48], xlab="t",
3.         main="48 PRIMEROS VALORES DE LA DIFERENCIA DE LA
  TRANSFORMACIÓN DE BOX-COX DE LA SERIE DE VENTAS", type="b")
4. points(seq(13, 24), Wt.1[13:24], type="b", col="blue", lwd=1.4)
5. points(seq(25, 36), Wt.1[25:36], type="b", col="red", lwd=1.4)
6. points(seq(37, 48), Wt.1[37:48], type="b", col="green", lwd=1.4)
```



Los círculos rojos en ggseasonplot evidencian un claro efecto estacional para cada mes. Quiero matizar que la presencia de estacionalidad significa que hay un patrón que se repite en todos los años. La estacionalidad puede estar presente, aunque los valores de cada mes no se agrupen en torno a un valor medio a causa de la tendencia general de la serie. Esto último únicamente queda reflejado cuando estabilizamos la media, donde queda claro que las subseries anuales poseen valores medios distintos propios del efecto estacional.

Asimismo, en la representación de la serie en sí también podemos evidenciar que hay claro un patrón que se repite. Aunque no se refleje en este plot, dicho patrón se repite a lo largo de toda la serie:

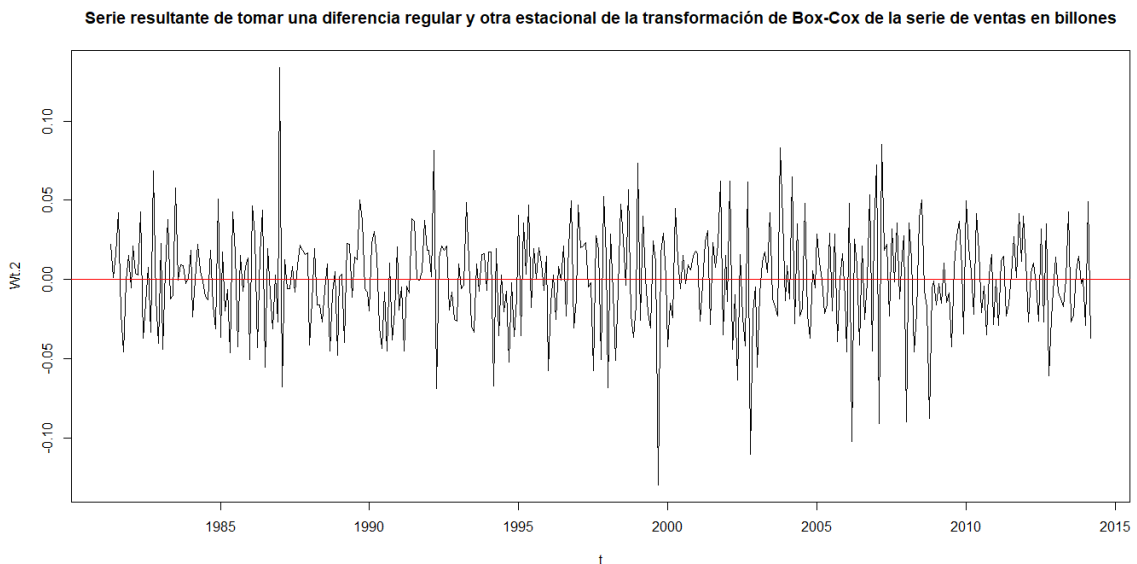
48 PRIMEROS VALORES DE LA DIFERENCIA DE LA TRANSFORMACIÓN DE BOX-COX DE LA SERIE DE VENTAS



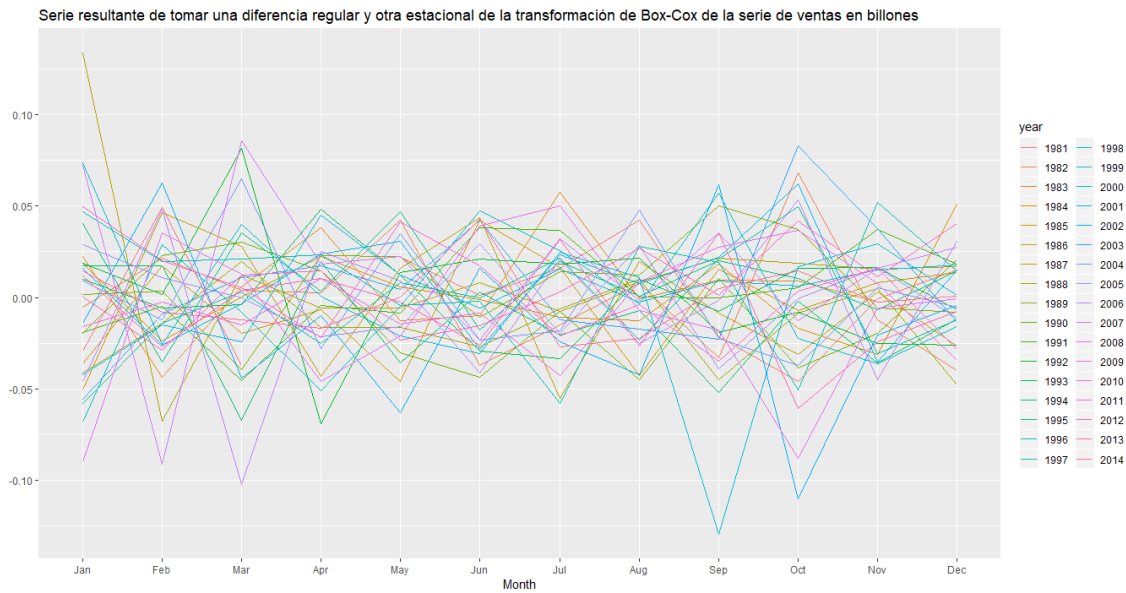
Tomamos  $s = 12$  e implementamos el cálculo en R:

```
1. Wt.2=diff(Wt.1, lag=12,differences=1)
2. ts.plot(Wt.2,xlab="t",main="Serie resultante de tomar una
diferencia regular y otra estacional de la transformación de
Box-Cox de la serie de ventas en billones",type="l")
3. abline(h=0, col = "red")
4. ggseasonplot(Wt.2,main="Serie resultante de tomar una diferencia
regular y otra estacional de la transformación de Box-Cox de la
serie de ventas en billones")
```

Perdemos otras 12 observaciones y se tiene la representación:



Sin embargo, resulta más evidente que hemos deshecho el efecto estacional en la representación del ggseasonplot:



Con todo esto, nos queda la serie transformada (ya estacionaria) que sigue:

$$\nabla_{12} \nabla^d \hat{X}_t = \nabla_{12} \nabla \left( \frac{X_t^\lambda - 1}{\lambda} \right), \text{ con } \lambda = 0.2$$

## 2.4 IDENTIFICACIÓN FINAL: GRADOS POLINOMIALES

La transformación  $\nabla_{12} \nabla^d \hat{X}_t$  ya es una serie estacionaria. De hecho, el método Box-Jenkins asume que dicha serie sigue un proceso ARMA estacional multiplicativo de parámetros por identificar  $(p, q) \times (P, Q)$ .

Podemos encontrar el comportamiento de la ACF de un modelo ARMA estacional multiplicativo en el libro *Análisis de Series Temporales* (por Daniel Peña **páginas 269-272**). Dicho comportamiento sigue la fórmula:

$$\rho_j = \frac{r_j + \sum_{i=1}^{\infty} R_{si} (r_{si+j} + r_{si-j})}{1 + 2 \sum_{i=1}^{\infty} r_{si} R_{si}}$$

Donde  $\rho_j$  representa la ACF del modelo ARMA estacional multiplicativo con retardo  $j$ ,  $r_j$  representa la ACF de la parte regular con retardo  $j$  y  $R_{si}$  representa la ACF de la parte estacional ( $s = 12$ , en nuestro caso).

En resumen, la **ACF de un ARMA estacional multiplicativo** presenta las siguientes características:

1. En los retardos bajos se observará la ACF de la parte regular.
2. En los retardos estacionales (múltiplos de  $s = 12$ ) se observará básicamente la ACF de la parte estacional.

- Alrededor de los retardos estacionales observaremos la interacción entre la parte regular y estacional, que se manifestará en la repetición a ambos lados de cada retardo estacional de la ACF de la parte regular.

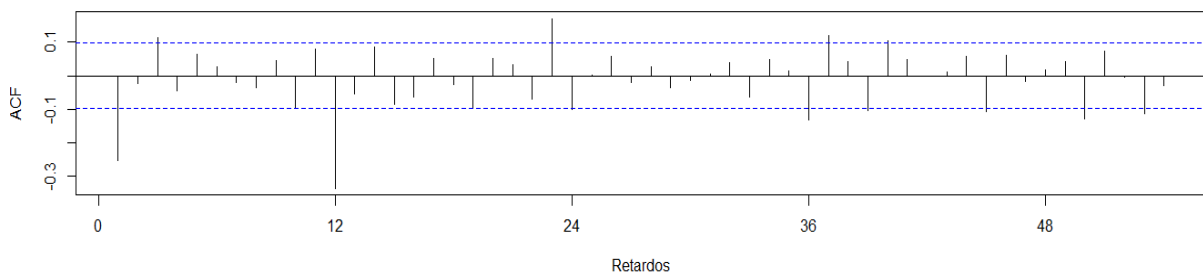
Por otra parte, la PACF de un ARMA estacional multiplicativo presenta características parecidas:

- En los retardos bajos se observará la PACF de la parte regular.
- En los retardos estacionales se observará básicamente la PACF de la parte estacional.
- A la derecha de cada coeficiente estacional (retardos  $js + 1, js + 2, \dots$ ) aparecerá la PACF de la parte regular. Si el coeficiente estacional es positivo, la PACF regular aparecerá invertida, mientras que si es negativo la PACF regular aparecerá con su signo.
- A la izquierda de cada coeficiente estacional (retardos  $js - 1, js - 2, \dots$ ) aparecerá la ACF de la parte regular.

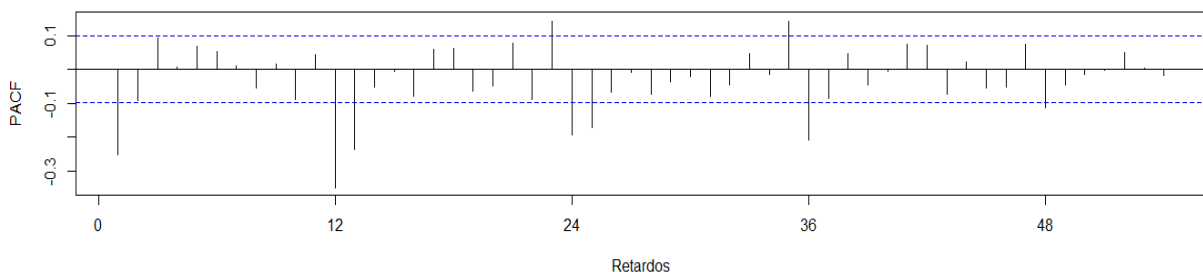
Con estas premisas somos capaces de proponer modelos candidatos que expliquen la serie temporal:

```
1. par(mfrow=c(2,1))
2. Acf(Wt.2,ylab="ACF",xlab="Retardos", main="ACF muestral de la
serie resultante de tomar d=1 y s=1 de la transformación de Box-
Cox de la serie de ventas en billones",54) # ACF muestral
3. Pacf(Wt.2,ylab="PACF",xlab="Retardos", main="PACF muestral de
la serie resultante de tomar d=1 y s=1 de la transformación de
Box-Cox de la serie de ventas en billones",54) # PACF
muestral
```

**ACF muestral de la serie resultante de tomar  $d=1$  y  $s=1$  de la transformación de Box-Cox de la serie de ventas en billones**



**PACF muestral de la serie resultante de tomar  $d=1$  y  $s=1$  de la transformación de Box-Cox de la serie de ventas en billones**



Nótese que lo que recaiga dentro de las bandas azules no debe utilizarse para establecer conclusiones. Estas bandas cuando la muestra es muy grande indican que lo que reste dentro de ellas debe considerarse 0. Por otra parte, no debe pasar inadvertido el hecho de que conforme aumentamos el

retardo en las gráficas el número de muestras para la estimación disminuye y, con ello, nuestra fiabilidad sobre las estimaciones obtenidas.

Analizando la parte regular: Tenemos bastante incertidumbre en la parte regular de la serie transformada. Notemos que para los primeros retardos de la ACF solamente se puede confirmar que el primer coeficiente es no nulo y que el resto de los primeros retardos son posiblemente nulos o muy cercanos a cero. Esto último es compatible para cualesquiera que sean  $p, q$ . Por otra parte, en la PACF el primer coeficiente es no nulo y el resto de los primeros retardos son nulos o muy cercanos a cero también.

Conclusión: Teniendo en cuenta que los primeros retardos tanto en la ACF como la PACF (excluyendo el primero que es claramente no nulo en ambos casos) pueden ser o nulos o cercanos a cero como el resultado de un decrecimiento geométrico abrupto, concluimos que los modelos candidatos para la parte regular son los siguientes (teniendo en cuenta que es frecuente que  $0 \leq p, q \leq 3$ ):

1. ARMA(0,1)
2. ARMA(1,0)
3. ARMA(1,1)
4. ARMA(2,2)
5. ARMA(3,3)

Analizando la parte estacional: En los retardos estacionales de la ACF se tiene que el primer retardo estacional es no nulo y que el resto podrían ser nulos. Por otro lado, en los retardos estacionales observamos un decrecimiento de términos no nulos. Concluimos, por consiguiente, que una estructura prometedora para la parte estacional sería  $ARMA(0, 1)_{12}$ . No obstante, el retardo 36 de la ACF podría sugerir considerar una parte estacional con otros parámetros, sin embargo, por ser pragmáticos e ilustrativos, nos dedicaremos a analizar solamente los modelos resultantes de considerar un  $ARMA(0, 1)_{12}$ .

Concluimos, por tanto, que nuestros modelos candidatos son 5 y son los siguientes:

1. ARIMA (0,1,1)  $\times$  (0,1,1)<sub>12</sub>
2. ARIMA (1,1,0)  $\times$  (0,1,1)<sub>12</sub>
3. ARIMA (1,1,1)  $\times$  (0,1,1)<sub>12</sub>
4. ARIMA (2,1,2)  $\times$  (0,1,1)<sub>12</sub>
5. ARIMA (3,1,3)  $\times$  (0,1,1)<sub>12</sub>

Con estos resultados pasamos a la estimación por método de máxima verosimilitud de los polinomios de retardo correspondientes en cada modelo  $\Phi_p(B^s)$ ,  $\Theta_q(B^s)$ ,  $\phi_p(B)$  y  $\theta_q(B)$ . Tras la estimación de tales polinomios ya disponemos de los modelos candidatos al completo y resta hacer que todos los modelos candidatos que hayamos propuesto compitan. Los métodos que emplearemos para hacerlos competir son 3:

Métodos de comparación/competición:

1. Penalización: AIC, AICc y BIC.
2. Cross validation.



3. Data-splitting.

## 3 ESTIMACIÓN Y SELECCIÓN DE MODELOS

### 3.1 ESTIMACIÓN DE MODELOS (Peña, capítulo 10: páginas 285-304)

Explicaremos brevemente la idea esencial sobre la estimación de los parámetros de un modelo  $X_t \sim \text{ARMA}(p,q)$  en el que la fórmula viene dada por:

$$X_t - \phi_1 X_{t-1} - \dots - \phi_p X_{t-p} = c + Y_t - \theta_1 Y_{t-1} - \dots - \theta_q Y_{t-q}$$

O, equivalentemente,

$$\phi(B)X_t = \theta(B)Y_t$$

Donde:

1.  $\phi(B) := 1 - \phi_1 B - \dots - \phi_p B^p$
2.  $\theta(B) := 1 - \theta_1 B - \dots - \theta_q B^q$
3.  $Y_t \sim \text{IID}(0, \sigma^2) \equiv N(0, \sigma^2)$  ruido aleatorio.
4.  $c$  es constante.

Supondremos que se dispone de una serie estacionaria de  $T$  valores observados (no necesariamente es el tamaño de la serie original, sino que puede ser el tamaño de la serie transformada)  $\{w_t\}$ ,  $t = 1, \dots, T$  y queremos estimar los parámetros de un modelo ARMA especificado. Utilizaremos la notación  $w_t$  porque es frecuente que la serie a estimar sea una transformación de la serie original,  $X_t$ , e.g.,  $w_t = \nabla \nabla_{12} \ln X_t$ . Se supone también que no se dispone de información a priori sobre los valores de los parámetros del proceso, esto es, sobre los valores denotados como el vector  $\beta = (\mu, \phi_1, \dots, \theta_q, \sigma^2)$ . El método de máxima verosimilitud consiste en encontrar aquellos parámetros  $\beta$  que maximizan el valor de la función de densidad conjunta  $f(w_1, \dots, w_T)$ . Gracias a la hipótesis de que los errores conformen un ruido aleatorio de distribución normal, y condicionando variables, la función de densidad conjunta de un ARMA se puede escribir (Peña, pág: 288) como:

$$\begin{aligned} f(w_1, \dots, w_T) &= f(w_1) f(w_2|w_1) \dots f(w_T|w_1, \dots, w_{T-1}) = \\ &= \prod_{t=1}^T \sigma^{-1} v_{t|t-1}^{-1/2} (2\pi)^{-1/2} \exp \left\{ -\frac{1}{2\sigma^2} \sum_{t=1}^T \frac{(w_t - w_{t|t-1})^2}{v_{t|t-1}} \right\} \end{aligned}$$

Donde

1.  $w_{t|t-1} := \hat{w}_{t-1}(1) = E(w_T|w_1, \dots, w_{T-1})$ , i.e., la predicción a un paso (origen  $t-1$  y horizonte 1)
2. y  $v_{t|t-1} := \frac{1}{\sigma^2} \text{Var}(w_T|w_1, \dots, w_{T-1})$  son las llamadas varianzas condicionales.

Quedando la función de verosimilitud como:

$$L(\boldsymbol{\beta}) := -\frac{T}{2} \ln \sigma^2 - \frac{1}{2} \sum_{t=1}^T \ln v_{t|t-1} - \frac{1}{2\sigma^2} \sum_{t=1}^T \frac{e_t^2}{v_{t|t-1}}$$

Donde  $e_t := w_t - w_{t|t-1}$  es el error de predicción. Dicha función de verosimilitud se maximiza sobre los parámetros  $\boldsymbol{\beta}$  mediante el algoritmo de optimización no lineal. Se utiliza frecuentemente el basado en el algoritmo de Gauss-Newton (Peña, Pág: 289) con una modificación debida a Marquardt. Esto se describe en el apéndice 10.1 del libro de Daniel Peña: Análisis de Series Temporales al que estamos haciendo referencia (Peña, Págs: 313- 317). La normalidad en la función de densidad conjunta la podemos asumir gracias a la hipótesis de que las innovaciones sigan una distribución de ruido aleatorio con normalidad. En definitiva, evaluar la función de verosimilitud se reduce a calcular los errores de predicción a un paso de cada observación dadas las anteriores ( $e_T$ ) y sus varianzas.

Recordemos que los errores de predicción a un paso coinciden con las innovaciones del proceso salvo algunos primeros valores a casusa de un problema de valores iniciales.

Para el caso AR(1) de media  $\mu = 0$ , donde  $w_t = \phi w_{t-1} + Y_t$ , se tiene que los errores de predicción son:

$$e_1 = w_1 - \phi E(w_0) = w_1 \neq Y_1 = w_1 - \phi w_0$$

$$e_2 = w_2 - \phi w_1 = Y_2$$

...

$$e_T = w_T - \phi w_{T-1} = Y_T$$

En tal caso:

$Var(w_T | w_1, \dots, w_{T-1}) := \sigma^2 v_{t|t-1}$ , con  $v_{t|t-1} = 1$  para  $t = 2, \dots, T$ , y para  $t = 1$ ,  $v_{1|0} = (1 - \phi^2)^{-1}$ , ya que  $Var(w_1) = \sigma^2(1 - \phi^2)^{-1}$ . Y la función de máxima verosimilitud estaría determinada con estos valores.

Ahora que ya conocemos un poco el fundamento teórico, nos disponemos a estimar los parámetros de los modelos que nosotros propusimos para la serie de ventas en billones de comidas para llevar en Australia:

1. ARIMA (0,1,1) × (0,1,1)<sub>12</sub>
2. ARIMA (1,1,0) × (0,1,1)<sub>12</sub>
3. ARIMA (1,1,1) × (0,1,1)<sub>12</sub>
4. ARIMA (2,1,2) × (0,1,1)<sub>12</sub>
5. ARIMA (3,1,3) × (0,1,1)<sub>12</sub>

```
1. > LBD <- BoxCox.out (VENTAS, 12) [[2]]
2. > LBD
3. log(xbar.vec)
4. 0.2031995
5. > MODELO_1=Arima (VENTAS, order=c(0, 1, 1), seasonal=list(order=c(0, 1, 1)
, period=12 ), lambda=LBD)
6. > MODELO_2=Arima (VENTAS, order=c(1, 1, 0), seasonal=list(order=c(0, 1, 1)
, period=12 ), lambda=LBD)
```

```

7. > MODELO_3=Arima(VENTAS,order=c(1,1,1),seasonal=list(order=c(0,1,1)
,period=12),lambda=LBD)
8. > MODELO_4=Arima(VENTAS,order=c(2,1,2),seasonal=list(order=c(0,1,1)
,period=12),lambda=LBD)
9. > MODELO_5=Arima(VENTAS,order=c(3,1,3),seasonal=list(order=c(0,1,1)
,period=12),lambda=LBD)
10.
11. > MODELO_1
12. Series: VENTAS
13. ARIMA(0,1,1)(0,1,1)[12]
14. Box Cox transformation: lambda= 0.2031995
15.
16. Coefficients:
17.          ma1          sma1
18.      -0.3631   -0.8138
19. s.e.   0.0439    0.0323
20.
21. sigma^2 estimated as 0.0006273: log likelihood=890.32
22. AIC=-1774.64  AICc=-1774.58  BIC=-1762.7
23.
24. > MODELO_2
25. Series: VENTAS
26. ARIMA(1,1,0)(0,1,1)[12]
27. Box Cox transformation: lambda= 0.2031995
28.
29. Coefficients:
30.          ar1          sma1
31.      -0.3333   -0.8094
32. s.e.   0.0476    0.0327
33.
34. sigma^2 estimated as 0.0006378: log likelihood=887.19
35. AIC=-1768.37  AICc=-1768.31  BIC=-1756.44
36.
37. > MODELO_3
38. Series: VENTAS
39. ARIMA(1,1,1)(0,1,1)[12]
40. Box Cox transformation: lambda= 0.2031995
41.
42. Coefficients:
43.          ar1          ma1          sma1
44.      -0.0836   -0.2956   -0.8134
45. s.e.   0.1134    0.1050    0.0323
46.
47. sigma^2 estimated as 0.0006281: log likelihood=890.58
48. AIC=-1773.17  AICc=-1773.07  BIC=-1757.25
49.
50. > MODELO_4
51. Series: VENTAS
52. ARIMA(2,1,2)(0,1,1)[12]
53. Box Cox transformation: lambda= 0.2031995
54.
55. Coefficients:
56.          ar1          ar2          ma1          ma2          sma1
57.      -0.9214   -0.3110   0.5531   -0.0333   -0.8105
58. s.e.   0.1854    0.1235   0.1912    0.1496    0.0327
59.
60. sigma^2 estimated as 0.0006197: log likelihood=894.3
61. AIC=-1776.6  AICc=-1776.39  BIC=-1752.73
62.

```

```

63. > MODELO_5
64. Series: VENTAS
65. ARIMA(3,1,3)(0,1,1)[12]
66. Box Cox transformation: lambda= 0.2031995
67.
68. Coefficients:
69.          ar1      ar2      ar3      ma1      ma2      ma3      sma1
70.         -0.2394  0.3102  0.1409  -0.1311  -0.4027  0.1080  -0.8089
71. s.e.         0.3636  0.3180  0.1650   0.3643   0.2328  0.1649   0.0331
72.
73. sigma^2 estimated as 0.0006224:  log likelihood=894.49
74.  AIC=-1772.99  AICc=-1772.62  BIC=-1741.16

```

Por ejemplo, para el modelo 2, se tendría la siguiente estimación:

```

1. > MODELO_2
2. Series: VENTAS
3. ARIMA(1,1,0)(0,1,1)[12]
4. Box Cox transformation: lambda= 0.2031995
5.
6. Coefficients:
7.          ar1      sma1
8.         -0.3333  -0.8094
9. s.e.         0.0476  0.0327
10.
11. sigma^2 estimated as 0.0006378:  log likelihood=887.19
12.  AIC=-1768.37  AICc=-1768.31  BIC=-1756.44

```

Siguiendo con nuestro caso, nuestro modelo candidato sería:

$$\phi_1(B)\nabla_{12}\nabla\hat{X}_t = \theta_1(B^{12})Y_t$$

Donde se tiene que:

- 1)  $\phi_1(B) = 1 + 0.3333B$
- 2)  $\theta_1(B^{12}) = 1 + 0.8094B^{12}$
- 3)  $Y_t$  es un ruido aleatorio de distribución normal  $N(0, 0.0006378)$  (a comprobar por diagnosis)
- 4)  $\hat{X}_t = \left(\frac{X_t^\lambda - 1}{\lambda}\right)$ , con  $\lambda = 0.2$
- 5) Log likelihood es  $L(\hat{\beta})$  y tiene valor 887.19

### 3.2 SELECCIÓN DE MODELOS (Peña, capítulo 10: páginas 305-311, 317-319)

#### CRITERIOS AIC DE AKAIKE, AICc y BIC

Tenemos que tener en cuenta que no es útil comparar las verosimilitudes obtenidas para cada modelo en las estimaciones, porque siempre el modelo con más parámetros tendrá mayor verosimilitud. Esto último significa que un modelo con más parámetros se ajustará mejor a

los valores observados de la serie, pero no generalizará y conseguiremos ruido. Por tanto, no debemos escoger el modelo únicamente en base a la serie observada, sino también en base a todas las posibles realizaciones. Sin embargo, para solucionar este inconveniente podemos calcular el valor esperado de la verosimilitud para cada uno de los modelos candidatos, es decir, el valor que esperamos obtener para la verosimilitud sobre los parámetros  $\beta$  y sobre todas las realizaciones posibles si en cada una estimamos los parámetros por máxima verosimilitud. Por consiguiente, **seleccionaremos aquel modelo que maximice el valor esperado de esta verosimilitud esperada** de la que hablamos.

Este último enfoque nos conduce al **criterio de información de Akaike** (AIC, Hirotugu Akaike), que ofrece el objetivo de conseguir un trade-off entre la bondad de ajuste del modelo y la complejidad del mismo:

$$AIC := E(-2L(\beta)) = T \ln \sigma^2 + \sum_{t=1}^T \ln v_{t|t-1} + E \left[ \sum_{t=1}^T \frac{e_t^2}{\sigma^2 v_{t|t-1}} \right]$$

Se puede probar (Daniel Peña, página 317-318) que si

1. Suponemos que los parámetros están estimados con los datos y
2. Se calcula la esperanza de esta verosimilitud respecto a futuras observaciones, se obtiene que:

$$AIC = E(-2L(\hat{\beta})) = T \ln \hat{\sigma}_{MV}^2 + 2k$$

Donde  $T$  es el tamaño muestral utilizado para estimar el modelo,  $\hat{\sigma}_{MV}^2$  es el estimador MV de la varianza de las innovaciones utilizando los  $T$  datos y  $k$  es el número de parámetros estimados para calcular las predicciones a un paso. Notemos que el **AIC se minimiza** (menos minimizar es maximizar). Notemos también que en la demostración referenciada se ignora una constante que no afecta para encontrar el valor óptimo a la hora de minimizar el AIC.

No obstante, esta fórmula AIC ofrecida en el Daniel Peña se trata con normalidad en la distribución, pero hay definiciones más generales.

- A) Es habitual ver la **fórmula general AIC** como

$$AIC := 2k - 2L(\hat{\beta}) = 2k - 2\log(\mathbf{Likelihood})$$

donde el primer sumando penaliza el número de parámetros utilizado para crear el modelo y  $k$  es el número de parámetros utilizados.

- B) El problema con el AIC es que tiende a sobrestimar el número de parámetros en el modelo y este afecto puede ser muy grande en pequeñas muestras. Una alternativa que corrige esta sobrestimación es el criterio AIC **corregido** denotado por AICc, dado por:

$$AICc := AIC + \frac{2k^2 + 2k}{T - k - 1}$$

- C) Por último, consideraremos el criterio BIC (criterio de información bayesiano). Este enfoque, al igual que el criterio AIC, busca penalizar el número de parámetros utilizados en el modelo. Su fórmula se define como:

$$BIC = k \log n - 2 \log (L(\hat{\beta})) = k \log n - 2 \log (\text{Likelihood})$$

Donde k es el número de parámetros utilizado para crear el modelo.

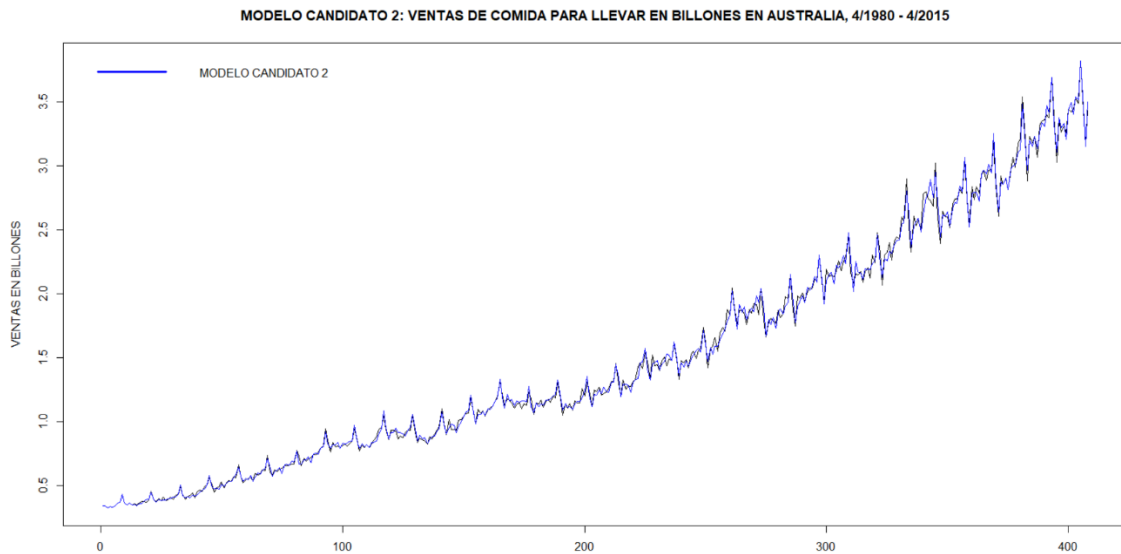
Para el modelo 2 se tenía:

```
1. > MODELO_2
2. Series: VENTAS
3. ARIMA (1,1,0) (0,1,1) [12]
4. Box Cox transformation: lambda= 0.2031995
5.
6. Coefficients:
7.      ar1      sma1
8.    -0.3333  -0.8094
9. s.e.  0.0476  0.0327
10.
11.      sigma^2 estimated as 0.0006378:  log likelihood=887.19
12.      AIC=-1768.37  AICc=-1768.31  BIC=-1756.44
```

Teniendo en cuenta que tenemos 3 parámetros estimados:

```
1. > MODELO_2$loglik
2. [1] 887.1864
3. > k = 3 # k parámetros estimados
4. > -2*MODELO_2$loglik+2*k # AIC
5. [1] -1768.373
6. > -2*MODELO_2$loglik+2*k + (2*k**2 + 2*k) / ((L-1-12)-3-1) #
  AICc
7. [1] -1768.311
8. > -2*MODELO_2$loglik+3*log(L-1-12) # BIC
9. [1] -1756.436
```

Por último, y como inciso, si quisiéramos ver cómo se adapta el modelo 2, i.e., cómo el modelo 2 ajusta los valores observados (fitted values mediante one-step forecasts) tendríamos:



```

1. # Modelo Candidato 2
2. par(mfrow=c(1,1))
3. plot(seq(1,L), VENTAS, ylab="VENTAS EN
   BILLONES", xlab="", type="l", main="MODELO CANDIDATO 2: VENTAS DE
   COMIDA PARA LLEVAR EN BILLONES EN AUSTRALIA, 4/1980 - 4/2014")
4. lines(seq(1,L), forecast(MODELO_2)$fitted, col="blue")
5. legend("topleft", col=c("blue"), legend =c("MODELO CANDIDATO 2"),
   lwd=3, bty = "n")

```

Sabido esto, comparemos los 5 modelos candidatos:

```

1. > #####
2. > ### COMPETICION DE MODELOS CANDIDATOS ###
3. > #####
4. >
5. >
6. > ### COMPARACIÓN AIC, AICc y BIC DE LOS TRES MODELOS
7. >
8. > AIC.BIC=rbind( c(MODELO_1$aic,MODELO_1$aicc,MODELO_1$bic),
9. + c(MODELO_2$aic,MODELO_2$aicc,MODELO_2$bic),
10. + c(MODELO_3$aic,MODELO_3$aicc,MODELO_3$bic),
11. + c(MODELO_4$aic,MODELO_4$aicc,MODELO_4$bic),
12. + c(MODELO_5$aic,MODELO_5$aicc,MODELO_5$bic))
13. >
14. > dimnames(AIC.BIC)=list(c("Model 1","Model 2", "Model
   3", "Model 4", "Model 5"),c("AIC", "AICc", "BIC"))
15. > AIC.BIC
16.           AIC      AICc      BIC
17. Model 1 -1774.641 -1774.580 -1762.705
18. Model 2 -1768.373 -1768.311 -1756.436
19. Model 3 -1773.168 -1773.066 -1757.253
20. Model 4 -1776.605 -1776.388 -1752.732
21. Model 5 -1772.990 -1772.617 -1741.159

```

A la luz de los datos, y teniendo en cuenta que se debe escoger aquel modelo que minimice los AIC, AICc y BIC, extraemos las siguientes conclusiones:

- 1) El modelo 1 minimiza el BIC
- 2) El modelo 4 minimiza los AIC, AICc

Puesto que la diferencia entre los AIC, AICc de los modelos 1 y 4 es muy pequeña parece que según los criterios de información la elección acertada es escoger el modelo 1. Cabe decir que es esperable que el modelo 4 posea un BIC superior al modelo 1, puesto que el modelo 1 posee menos parámetros y el criterio BIC penaliza mucho más el número de parámetros empleados en el modelo que el criterio AIC. Sin embargo, deberíamos tener en cuenta otros criterios para nuestra elección de modelo ideal como los que vamos a tratar a continuación.

### CROSS VALIDATION

Otra forma de hacer competir los modelos es mediante el método llamado Cross Validation. Este método consiste en (la dirección del algoritmo es a elección de uno mismo, e incluso podría hacerse de forma bidireccional):

1. Tomar una primera porción de la serie (normalmente los  $N$  primeros valores de la serie, para un determinado  $N$  suficientemente grande)
2. Se ajusta el modelo para los  $N$  primeros valores de la serie y se calcula la predicción a un paso (origen  $N$ , horizonte 1).
3. Se toman ahora los  $N+1$  primeros valores y se reiteran los pasos 1 y 2 hasta llegar al límite natural fijado por la serie observada.

Si ahora implementamos este algoritmo en R para  $N = 250$  con nuestra serie tenemos:

```

1. ### COMPARACIÓN CON CROSS-VALIDATION
2.
3. #Comparación de los cinco modelos utilizando "Cross-validation"
4.
5. n.min=250 # minimal sample size required to estimate a model
6. n.data=length(VENTAS)
7. MSE.vec1=numeric(n.data-n.min) # Mean Square Error model 1
8. MAE.vec1=numeric(n.data-n.min) # Mean Absolute Error model 1
9. MSE.vec2=numeric(n.data-n.min) # Mean Square Error model 2
10. MAE.vec2=numeric(n.data-n.min) # Mean Absolute Error model 2
11. MSE.vec3=numeric(n.data-n.min) # Mean Square Error model 3
12. MAE.vec3=numeric(n.data-n.min) # Mean Absolute Error model 3
13. MSE.vec4=numeric(n.data-n.min) # Mean Square Error model 4
14. MAE.vec4=numeric(n.data-n.min) # Mean Absolute Error model 4
15. MSE.vec5=numeric(n.data-n.min) # Mean Square Error model 5
16. MAE.vec5=numeric(n.data-n.min) # Mean Absolute Error model 5
17.
18. forecast.mod1=numeric(n.data-n.min)
19. forecast.mod2=numeric(n.data-n.min)
20. forecast.mod3=numeric(n.data-n.min)
21. forecast.mod4=numeric(n.data-n.min)
22. forecast.mod5=numeric(n.data-n.min)
23.
24. LBD <- BoxCox.out(VENTAS, 12)[[2]]
25. LBD
26.
27. for (k in 1:(n.data-n.min-1))

```



```

28. {
29.   fit.mod1=Arima (VENTAS[1:(n.min+k)],order=c(0,1,1),seasonal=list(o
   rder=c(0,1,1),period=12),lambda = LBD)
30.   forecast.mod1[k] <- forecast(fit.mod1, h=1)[['mean']] # horizonte
   1 --> h=1
31.   #
32.   fit.mod2=Arima (VENTAS[1:(n.min+k)],order=c(1,1,0),seasonal=list(o
   rder=c(0,1,1),period=12),lambda= LBD)
33.   forecast.mod2[k] <- forecast(fit.mod2, h=1)[['mean']]
34.   #
35.   fit.mod3=Arima (VENTAS[1:(n.min+k)],order=c(1,1,1),seasonal=list(o
   rder=c(0,1,1),period=12),lambda = LBD)
36.   forecast.mod3[k] <- forecast(fit.mod3, h=1)[['mean']]
37.   #
38.   fit.mod4=Arima (VENTAS[1:(n.min+k)],order=c(2,1,2),seasonal=list(o
   rder=c(0,1,1),period=12),lambda = LBD)
39.   forecast.mod4[k] <- forecast(fit.mod4, h=1)[['mean']]
40.   #
41.   fit.mod5=Arima (VENTAS[1:(n.min+k)],order=c(3,1,3),seasonal=list(o
   rder=c(0,1,1),period=12),lambda = LBD)
42.   forecast.mod5[k] <- forecast(fit.mod5, h=1)[['mean']]
43.
44.   MSE.vec1[k]=(VENTAS[(n.min+k)+1]-forecast.mod1[k])^2
45.   MAE.vec1[k]=abs(VENTAS[(n.min+k)+1]-forecast.mod1[k])
46.   MSE.vec2[k]=(VENTAS[(n.min+k)+1]-forecast.mod2[k])^2
47.   MAE.vec2[k]=abs(VENTAS[(n.min+k)+1]-forecast.mod2[k])
48.   MSE.vec3[k]=(VENTAS[(n.min+k)+1]-forecast.mod3[k])^2
49.   MAE.vec3[k]=abs(VENTAS[(n.min+k)+1]-forecast.mod3[k])
50.   MSE.vec4[k]=(VENTAS[(n.min+k)+1]-forecast.mod4[k])^2
51.   MAE.vec4[k]=abs(VENTAS[(n.min+k)+1]-forecast.mod4[k])
52.   MSE.vec5[k]=(VENTAS[(n.min+k)+1]-forecast.mod5[k])^2
53.   MAE.vec5[k]=abs(VENTAS[(n.min+k)+1]-forecast.mod5[k])
54.   print(k) # si queremos hacer seguimiento del progreso
55. }

```

**Observación:** las últimas 55 líneas de código no aprovechan el modelo ajustado y no se actualiza el modelo con un nuevo dato/punto en base al modelo anterior para ahorrar en tiempo y eficacia. He probado dos alternativas que funcionan, sin embargo, sugieren distintas conclusiones, por tanto, las descarto. Adjunto el código de las dos alternativas junto con este documento para que se compruebe por uno mismo si se desea. El método así es lento, es cierto, no obstante, las conclusiones que se establecen están más acordes con las conclusiones proporcionadas por la competición de modelos vía criterios de la información AIC, AICc y BIC.

Los resultados son los siguientes:

```

1. > # Summary Results
2. > Results.mat=rbind(
3. +   apply(cbind(MSE.vec1,MSE.vec2,MSE.vec3,MSE.vec4,MSE.vec5)[- (L-
   n.min),],2,mean),
4. +   apply(cbind(MAE.vec1,MAE.vec2,MAE.vec3,MSE.vec4,MSE.vec5)[- (L-
   n.min),],2,mean)
5. + )
6. > dimnames(Results.mat)=list(c("MSE", "MAE"),c("Model 1", "Model
   2", "Model 3", "Model 4", "Model 5"))
7. > Results.mat
8.           Model 1      Model 2      Model 3      Model 4      Model 5
9.   MSE 0.002572866 0.002599347 0.002575864 0.00263268 0.002611254

```

```

10. MAE 0.040720619 0.041086917 0.040754330 0.00263268 0.002611254
11. > data.frame(MIN_MSE = min(Results.mat[1,]),
  MIN_MAE = min(Results.mat[2,]))
12.      MIN_MSE      MIN_MAE
13.      0.002572866 0.002611254

```

Deberíamos notar que el MAE es más restrictivo en este caso que el MSE. Esto se debe a la naturaleza de los datos:

```

1. > head(VENTAS)
2. [1] 0.3424 0.3421 0.3287 0.3385 0.3315 0.3419
3. > summary(VENTAS)
4.      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
5.  0.3287  0.8192  1.2447  1.5194  2.1545  3.8199

```

Se tratan de valores muy pequeños porque se han considerado las ventas en billones, así que las diferencias involucradas en el MSE y el MAE también son muy pequeñas si los modelos son plausibles y predicen medianamente bien. Podríamos haber asignado `VENTAS <- VENTAS*1.000.000` y considerar las ventas en millones. De todas formas, esto no debe preocuparnos, ya que los objetos "numeric" en R se almacenan con precisión double, esto significa que hasta 16 dígitos de precisión en decimales los tenemos garantizados, y en nuestro caso no llegan a tal número y podemos estar tranquilos con el tema de la precisión. Aunque nos muestre R los números truncados (pero no trabaja con tales números truncados) solo tenemos que indicarle la opción "`options(digits=16)`" si queremos que nos muestre todos los decimales.

A la luz de los resultados, se tiene que **el modelo que presenta el menor error cuadrático medio (MSE) con Cross-Validation es el modelo 1**. Sin embargo, **el modelo que presenta el menor error absoluto medio (MAE) es el modelo 5**. Teniendo en cuenta ahora que el modelo 5 era el que tenía peor criterio BIC, **aconsejo tomar el modelo 1 de momento**. Aunque todavía nos queda hacer un data-splitting. Cabe añadir que resulta llamativo que el MAE del modelo 5 es mucho menor que el MAE del modelo 1.

## DATA-SPLITTING

Por último, para la competición de modelos vamos a ver el **data-splitting**. Esto no es otra cosa que particionar el conjunto de entrenamiento (en nuestro caso la serie con la que se ajustan los modelos) en **2 porciones**. Una **primera porción** en términos generales se dice que es el **conjunto de entrenamiento (parte de la serie ventas con la que construiremos el modelo)** y se deja una **segunda porción** llamada conjunto tipo test y **cuyos valores no se han visto involucrados a la hora de construir cada modelo**. De hecho, estos valores pueden considerarse como "**el futuro**". Este concepto de data-splitting es general para la mayoría de modelos que se construyen a partir de un conjunto previo de valores observados y se suelen escindir con las proporciones de 0.7 para el conjunto de entrenamiento y 0.3 para el conjunto tipo test. En nuestro caso, dado que tenemos una serie, lo que hacemos es **dejar las observaciones correspondientes a un año** (12 observaciones por ser una serie mensual) y predecimos estos 12 valores comprobando los MAE y MSE cometidos por cada modelo candidato:

```

1. LBD <- BoxCox.out(VENTAS,12)[[2]]
2. LBD
3.
4. n.data=length(VENTAS)-12*0 # = 408
5. lap = 12*1
6. n1=408-lap # hago data splitting sobre 1 año
7.

```

```

8. fit.mod1DS=Arima(VENTAS[1:n1], order=c(0,1,1), seasonal=list(order
=c(0,1,1), period=12), lambda = LBD)
9. forecast.mod1DS <- forecast(fit.mod1DS, h=(n.data-
n1)) #predicciones
10. #
11. fit.mod2DS=Arima(VENTAS[1:n1], order=c(1,1,0), seasonal=list(
order=c(0,1,1), period=12), lambda = LBD)
12. forecast.mod2DS <- forecast(fit.mod2DS, h=(n.data-
n1)) #predicciones
13. #
14. fit.mod3DS=Arima(VENTAS[1:n1], order=c(1,1,1), seasonal=list(
order=c(0,1,1), period=12), lambda = LBD)
15. forecast.mod3DS <- forecast(fit.mod3DS, h=(n.data-
n1)) #predicciones
16. #
17. fit.mod4DS=Arima(VENTAS[1:n1], order=c(2,1,2), seasonal=list(
order=c(0,1,1), period=12), lambda = LBD)
18. forecast.mod4DS <- forecast(fit.mod4DS, h=(n.data-
n1)) #predicciones
19. #
20. fit.mod5DS=Arima(VENTAS[1:n1], order=c(3,1,3), seasonal=list(
order=c(0,1,1), period=12), lambda = LBD)
21. forecast.mod5DS <- forecast(fit.mod5DS, h=(n.data-
n1)) #predicciones
22. #
23. Results.mat2=cbind(
24.   c(mean( (forecast.mod1DS$mean-
VENTAS[(n1+1):n.data])^2), mean(abs(forecast.mod1DS$mean-
VENTAS[(n1+1):n.data]))),
25.   c(mean( (forecast.mod2DS$mean-
VENTAS[(n1+1):n.data])^2), mean(abs(forecast.mod2DS$mean-
VENTAS[(n1+1):n.data]))),
26.   c(mean( (forecast.mod3DS$mean-
VENTAS[(n1+1):n.data])^2), mean(abs(forecast.mod3DS$mean-
VENTAS[(n1+1):n.data]))),
27.   c(mean( (forecast.mod4DS$mean-
VENTAS[(n1+1):n.data])^2), mean(abs(forecast.mod4DS$mean-
VENTAS[(n1+1):n.data]))),
28.   c(mean( (forecast.mod5DS$mean-
VENTAS[(n1+1):n.data])^2), mean(abs(forecast.mod5DS$mean-
VENTAS[(n1+1):n.data]))))
29. )

```

Y calculamos ahora los correspondientes MSE y MAE:

```

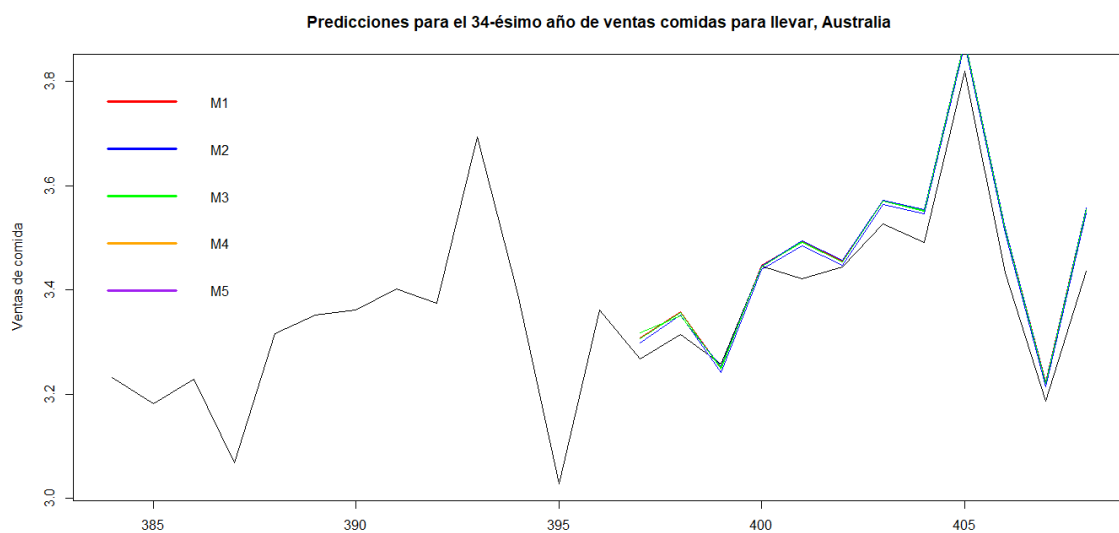
1. > dimnames(Results.mat2)=list(c("MSE", "MAE"), c("Model 1", "Model
2", "Model 3", "Model 4", "Model 5"))
2. > Results.mat2
3.      Model 1      Model 2      Model 3      Model 4      Model 5
4. MSE 0.003501122 0.002763423 0.003342793 0.003518211 0.003344743
5. MAE 0.049195119 0.043188729 0.047800296 0.048840421 0.047653501
6. > data.frame(MIN_MSE = min(Results.mat2[1,]),
MIN_MAE = min(Results.mat2[2,]))
7.      MIN_MSE      MIN_MAE
8. 1 0.002763423 0.04318873
9. > mean( (forecast.mod1DS$mean-VENTAS[397:408])^2)

```

```
10. [1] 0.003501122
```

Si quisiéramos representar las predicciones de cada modelo:

```
1. #plot
2. plot(seq((n1-lap), 408), VENTAS[(n1-lap):408], ylab="
", xlab="", type="l", main="Predicciones comidas para llevar")
3. points(seq((n1+1), 408), forcast.mod1DS$mean, type="l", col="red") #
predicciones
4. points(seq((n1+1), 408), forcast.mod2DS$mean, type="l", col="blue")
# predicciones
5. points(seq((n1+1), 408), forcast.mod3DS$mean, type="l", col="green")
# predicciones
6. points(seq((n1+1), 408), forcast.mod4DS$mean, type="l", col="blue")
# predicciones
7. points(seq((n1+1), 408), forcast.mod5DS$mean, type="l", col="green")
# predicciones
8. legend("topleft", col=c("red", "blue", "green", "orange", "purple")
, legend=c("M1", "M2", "M3", "M4", "M5"), lwd=3, bty="n")
9.
```



Podemos comprobar que todos los modelos predicen de una forma muy similar.

Los MSE y MAE correspondientes al modelo 2 son los más pequeños. Esto nos indica que **el modelo 2 es el que mejor se comporta de todos con el método data-splitting**. Sin embargo, el modelo 1 nos daba mejores resultados con el criterio de información y el criterio vía cross-validation. Podríamos considerar data-splitting para otros tramos. Podemos realizar data-splitting escogiendo 29 años y predecir el 30-ésimo año, escogiendo 30-años y predecir el 31-ésimo año, etc, y así comparar un poco más y no restringirnos a un único tramo de la serie:

```
1. > n.data=length(VENTAS)-12*0 # = 408
2. > lap = 12*1
3. > n1=408-lap # 33 años, predigo el 34-ésimo
4. > dimnames(Results.mat33_34)=list(c("MSE", "MAE"), c("Model 1", "Model
2", "Model 3", "Model 4", "Model 5"))
5. > Results.mat33_34
6.      Model 1      Model 2      Model 3      Model 4      Model 5
7. MSE 0.003501122 0.002763423 0.003342793 0.003518211 0.003344743
```

```

8. MAE 0.049195119 0.043188729 0.047800296 0.048840421 0.047653501
9. > data.frame(MIN_MSE = min(Results.mat33_34[1,]),
  MIN_MAE = min(Results.mat33_34[2,]))
10.      MIN_MSE      MIN_MAE
11. 1 0.002763423 0.04318873
12.
13. #####
14.
15. > n.data=length(VENTAS)-12*1 # = 396
16. > lap = 12*2
17. > n1=408-lap # 32 años, predigo el 33-ésimo
18. > dimnames(Results.mat32_33)=list(c("MSE", "MAE"), c("Model
  1", "Model 2", "Model 3", "Model 4", "Model 5"))
19. > Results.mat32_33
20.      Model 1      Model 2      Model 3      Model 4      Model 5
21. MSE 0.00165425 0.001609613 0.001638537 0.001723662 0.001784309
22. MAE 0.03539146 0.035657632 0.035468766 0.036548227 0.036571190
23. > data.frame(MIN_MSE = min(Results.mat32_33[1,]),
  MIN_MAE = min(Results.mat32_33[2,]))
24.      MIN_MSE      MIN_MAE
25. 1 0.001609613 0.03539146
26.
27. #####
28.
29. > n.data=length(VENTAS)-12*2 # = 384
30. > lap = 12*3
31. > n1=408-lap # 31 años, predigo el 32-ésimo
32. > dimnames(Results.mat31_32)=list(c("MSE", "MAE"), c("Model
  1", "Model 2", "Model 3", "Model 4", "Model 5"))
33. > Results.mat31_32
34.      Model 1      Model 2      Model 3      Model 4      Model 5
35. MSE 0.01267450 0.01258261 0.01281171 0.01252294 0.01312324
36. MAE 0.08353889 0.08305401 0.08374910 0.08328605 0.08530341
37. > data.frame(MIN_MSE = min(Results.mat31_32[1,]),
  MIN_MAE = min(Results.mat31_32[2,]))
38.      MIN_MSE      MIN_MAE
39. 1 0.01252294 0.08305401
40.
41. #####
42.
43. > n.data=length(VENTAS)-12*2 # = 372
44. > n1=408-lap # 30 años, predigo el 31-ésimo
45. > dimnames(Results.mat30_31)=list(c("MSE", "MAE"), c("Model
  1", "Model 2", "Model 3", "Model 4", "Model 5"))
46. > Results.mat30_31
47.      Model 1      Model 2      Model 3      Model 4      Model 5
48. MSE 0.003600163 0.004102761 0.003647235 0.08777168 0.003926388
49. MAE 0.050639808 0.053119843 0.050722245 0.26291101 0.052447517
50. > data.frame(MIN_MSE = min(Results.mat30_31[1,]),
  MIN_MAE = min(Results.mat30_31[2,]))
51.      MIN_MSE      MIN_MAE
52. 1 0.003600163 0.05063981
53.
54. #####
55.
56. > n.data=length(VENTAS)-12*4 # = 360
57. > lap = 12*5
58. > n1=408-lap # 29 años, predigo el 30-ésimo

```

```

59. > dimnames(Results.mat29_30)=list(c("MSE", "MAE"), c("Model
60. > Results.mat29_30
61.      Model 1      Model 2      Model 3      Model 4
62. MSE 0.0008207906 0.0007718474 0.0007878565 0.03737737
63. MAE 0.0231968173 0.0218052273 0.0222955934 0.19207390
64.      Model 5
65. MSE 0.0008250846
66. MAE 0.0219574502
67. > data.frame(MIN_MSE = min(Results.mat29_30[1,]),
68. MIN_MAE = min(Results.mat29_30[2,]))
69.      MIN_MSE      MIN_MAE
70. 1 0.0007718474 0.02180523
71. #####
72.
73. > n.data=length(VENTAS)-12*0 # = 408
74. > n1=408-lap # 30 años, predigo un lapso de 4 años
75. > lap = 12*4
76. > dimnames(Results.matDS)=list(c("MSE", "MAE"), c("Model 1", "Model
77. > Results.matDS
78.      Model 1      Model 2      Model 3      Model 4      Model 5
79. MSE 0.003837938 0.003828464 0.003815552 0.003777293 0.003783397
80. MAE 0.052062865 0.052670674 0.052054961 0.052216544 0.052456984
81. > data.frame(MIN_MSE = min(Results.matDS[1,]),
82. MIN_MAE = min(Results.matDS[2,]))
83.      MIN_MSE      MIN_MAE

```

Como podemos comprobar no se tiene todo claro. Siguiendo mi criterio, no decidiría en este caso, a partir de un data-splitting del último año, que el modelo 2 es mejor que los demás, porque como hemos visto para distintos años se obtienen conclusiones distintas con data-splitting. Si bien, es cierto que hemos ido quitando años y las muestras pueden ser menos fiables, pero a la vista de la exploración inicial, no considero que la serie tenga un cambio de patrón significativo o que se aprecie a simple vista. Por tanto, hacer una estimación sobre 30 años de la serie dudo que suponga, para las estimaciones, un impacto considerable en comparación a la estimación obtenida con 33 años. Las características principales de la serie se mantienen.

Podría resultar que un modelo haga predicciones mejor a largo plazo y que otro haga mejores predicciones a corto plazo.

#### CONCLUSIÓN:

**Mi conclusión es escoger el modelo 1.** Recordemos que por cuestiones de simplicidad en el análisis nos hemos restringido a unos parámetros estacionales muy concretos y que podemos estar ignorando algunos modelos que expliquen mejor la serie.

## 4 DIAGNOSIS

Vamos a escoger el modelo 1 como ganador. No obstante, teníamos motivos para dudar de nuestra elección. Una forma más de decantarnos en casos de duda sería descartar aquellos modelos que no verifiquen la diagnosis. Aun así, por ser prácticos e ilustrativos, escogeré únicamente el modelo 1 para tratar la diagnosis. Escoger más modelos extendería aún más el trabajo.

Hemos seleccionado el modelo 1 como propuesta. Sin embargo, no está todo hecho. Todavía resta la diagnosis. Esto último significa comprobar aquellas hipótesis (validar los supuestos) sobre las que se sostiene la construcción del modelo. En el caso de una regresión lineal múltiple estos supuestos que había que validar para garantizar el buen funcionamiento del modelo (y que no sea fruto de alguna casualidad) consistían en comprobar que el tamaño de la muestra fuera correcto, que las variables que se usaban como regresoras fueran independientes, que el fenómeno no presentara heterocedasticidad, que los residuos fueran normales e independientes, y que no hubiera valores atípicos influyentes. Sin duda, para la regresión lineal hay una gran cantidad de supuestos que había que comprobar para la regresión lineal múltiple, por lo que podemos entender que el rango de aplicación de una regresión lineal es muy exigente.

Podemos encontrar la información relativa a esta sección en el libro "*Testing the Estimated Noise Sequence, Brockwell and Davis 1.6, pag. 35-40*"

La diagnosis en un modelo SARIMA requiere comprobar las hipótesis básicas realizadas respecto a los residuos. Estas últimas son:

1.  $H_{0a}: \{Y_t\} \sim \text{IID}(\mathbf{0}, \sigma^2)$ ; (los residuos son ruido aleatorio)
2.  $H_{0b}: \{Y_t\} \sim \mathbf{N}(\mathbf{0}, \sigma^2)$ ; (los residuos siguen una distribución normal)

#### 4.1 ¿LOS RESIDUOS SON RUIDO ALEATORIO?

Para contrastar la hipótesis nula  $H_{0a}$  de que los residuos forman un ruido aleatorio podemos utilizar los siguientes contrastes:

1. Contraste basado en *la ACF muestral*
2. Contraste de *Ljung-Box*
3. Contraste del punto de cambio (*Turning point test*)
4. Contraste de la diferencia de signos (*Difference sign test*)
5. Contraste de rangos (*Rank test*)

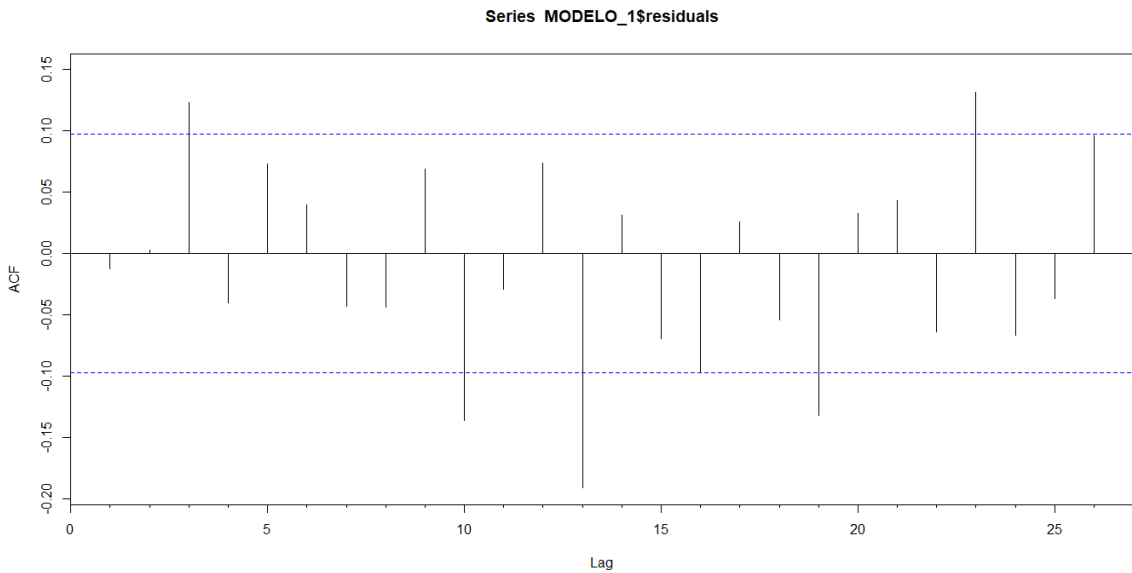
Antes que nada, cargamos nuestro el script llamado Diagnostic.R en el que vienen implementados todas las funciones de estos contrastes. Para poder cargar este script, hemos de cargar previamente unas cuantas librerías:

```
1. # Cargamos los paquetes necesarios
2.
3. # Diagnóstico (load the functions for "diagnostic" script)
4. require(astsa) # load astsa
5. require(portes) # load portes (Ljung-Box test, McLeod-Li_83 test)
```

6. `require(nortest)`
7. `source("Diagnostic.R")`

Contraste basado en la ACF muestral

La ACF muestral de los residuos en el modelo 1 es la siguiente:



Bajo la hipótesis nula,  $H_{0a}$ :  $\{Y_t\} \sim \text{IID}(0, \sigma^2)$ ; los coeficientes de la ACF muestral de los residuos siguen una distribución Normal de media cero y varianza asintótica IID (variables independientes e idénticamente distribuidas)  $1/n$ , i.e.,

$$\hat{\rho}_Y(h) | H_{0a} \sim N(0, \sigma^2)$$

Por lo tanto, si la hipótesis nula fuera verdadera, aproximadamente el 95% de los coeficientes de la ACF muestral deberían estar dentro del intervalo  $\left(-\frac{1.96}{\sqrt{n}}, \frac{1.96}{\sqrt{n}}\right)$

**PROCEDIMIENTO:** se rechaza la hipótesis nula  $H_{0a}$  si más del 5% de los coeficientes (para  $h \leq n/4$ ) de la ACF muestral de los residuos están fuera del intervalo  $\left(-\frac{1.96}{\sqrt{n}}, \frac{1.96}{\sqrt{n}}\right)$ . El tamaño muestral utilizado  $n$  viene determinado en la función:

**SCRIPT:**

```
1. S.ACF=function(x) # evaluate the % of sample autocorrelation out
of the interval [-1.96*sqrt(n), 1.96*sqrt(n)]
2. {
3.   l.max=round(length(x)/4,0)
4.   SampleACF=as.numeric(unlist(acf(x,l.max))[2:(l.max+1)])
5.   bound.l=1.96/sqrt(length(x))
6.   out.bounds=length(seq(1,l.max)[abs(SampleACF)>bound.l]) #
number of sample acf in (1:l.max) whose absolute values is greater
than 1.96*sqrt(n)
7.   p.out=round(100*out.bounds/l.max,2) #
percentage of sample ACF ( in 1:l.max) whose absolute value is
greater than 1.96*sqrt(n)
```



```

8.   out.A=matrix(numeric(5), ncol=1)
9.   out.A[,1]=round(c(length(x), 1.max, bound.1, out.bounds, p.out), 2)
10.  dimnames(out.A)=list(c("n", "h", "bound", "# out of bounds", "%
    out of bounds"))
11.  return(out.A)
12.  }

```

En nuestro caso:

```

1. > S.ACF(MODELO_1$residuals)
2.      [,1]
3. n      408.00
4. h      102.00
5. bound      0.10
6. # out of bounds 14.00
7. % out of bounds 13.73

```

Tenemos que aproximadamente el 14 % de los coeficientes exceden los límites ( $bound = 0.1$ ). Esto nos obliga a rechazar la hipótesis nula y descartar el modelo 1 como ideal.

### Contraste de Ljung-Box

Supongamos la hipótesis nula  $H_{0a}$  cierta otra vez, vamos a comprobar el contraste de Ljung-Box.

Suponiendo  $H_{0a}$  cierto, entonces podemos asumir que

$$i. i. d. \\ \hat{\rho}_Y(h) | H_{0a} \sim N(0, \sigma^2)$$

Por lo tanto, bajo la hipótesis nula  $H_{0a}$ , el estadístico  $Q = n \sum_{h=1}^k \frac{\hat{\rho}_Y(h)^2}{n-h}$ ; ( $k$  debe ser grande) se distribuye aproximadamente como una  $\chi^2$ . Un valor elevado de  $Q$  indica que los coeficientes de la ACF muestral de los residuos son demasiado grandes para ser los residuos ruido aleatorio. Ljung-Box han propuesto utilizar, en lugar de  $Q$ , el estadístico  $Q_{LB}$ ,

$$Q_{LB} = n(n+2) \sum_{h=1}^k \frac{\hat{\rho}_Y(h)^2}{n-h}$$

### PROCEDIMIENTO:

Se rechaza la hipótesis nula  $H_{0a}$ , al nivel de significación  $\alpha$ , si el valor calculado del estadístico  $Q_{LB}$  es mayor que el percentil  $(1 - \alpha)$  de la distribución  $\chi^2$  con  $(k - n_p)$  grados de libertad siendo  $n_p$  el número de parámetros del modelo.

### SCRIPT:

```

1. # Test3. (Ljung-Box test)
2.
3. # np= number of estimated parameters in the model
4. My.Ljung.Box=function(x, np)
5. {
6.   n=length(x)
7.   l.max=round(n/4, 0)

```

```

8. SampleACF=as.numeric(unlist(acf(x,l.max))[2:l.max])
9. lag.vec=seq(5,30,length=6)
10. Q.ML=numeric(6)
11. for(k in 1:6)
12. {
13.   Q.ML[k]=n*(n+2)*sum( ( (SampleACF[1:lag.vec[k]])^2)/(n-
14.   seq(1:lag.vec[k])) )
15. }
16. Q.ML.Pvalues=numeric(6)
17. for(k in 1:6)
18. {
19.   Q.ML.Pvalues[k]=1-pchisq(Q.ML[k], lag.vec[k]-np, ncp = 0,
20.   lower.tail = TRUE, log.p = FALSE) # percentil (1-\alpha)
21. }
22. Q.ML.out=cbind(lag.vec, Q.ML,Q.ML.Pvalues)
23. dimnames(Q.ML.out)=list(NULL,c("k", "Test Statistic", "P-
value"))
24. return(Q.ML.out)
25. }

```

En nuestro caso:

```

1. # Test. (Ljung-Box test)
2. > My.Ljung.Box(MODELO_1$residuals,3)
3.   k Test Statistic      P-value
4. [1,] 5           9.132881 1.039490e-02
5. [2,] 10          21.139680 3.568825e-03
6. [3,] 15          41.665369 3.791018e-05
7. [4,] 20          55.128906 6.533282e-06
8. [5,] 25          67.634123 1.550384e-06
9. [6,] 30          73.606040 3.381546e-06

```

Podemos ver que a partir de  $k = 20, 25, 30$  el P-value minora el nivel de significación  $\alpha < 0'05$ , por lo que rechazaríamos la hipótesis nula  $H_{0a}$ .

### Contraste Turning Point Test

Supongamos la hipótesis nula  $H_{0a}$  cierta otra vez. Sea  $y_1, y_2, \dots, y_n$  una realización del proceso  $\{Y_t\}$ . Diremos que hay un cambio en  $t = i$  si

$$(y_{i-1} < y_i) \cap (y_i > y_{i+1}) \text{ ó } (y_{i-1} > y_i) \cap (y_i < y_{i+1}), \text{ siendo } i = 2, \dots, n-1$$

Definimos ahora la v.a.  $T$  = "Número de puntos de cambio en una realización de longitud  $n$  del proceso  $\{Y_t\}$ ". Para  $n$  grande, se tiene que:

$$T|H_{0a} \sim N(\mu_T, \sigma_T^2), \text{ siendo } \mu_T = \frac{2(n-2)}{3}; \sigma_T^2 = \frac{16n-29}{90}$$

### PROCEDIMIENTO:

Se rechaza la hipótesis nula  $H_{0a}$  al nivel de significación  $\alpha$ , si  $\frac{|T-\mu_T|}{\sigma_T} > \phi_{1-\alpha/2}$ , donde  $\phi_{1-\alpha/2}$  es el percentil  $(1 - \alpha/2)$  de una distribución  $N(0, 1)$ .

Implementaremos este contraste junto con los 2 que siguen ya que los estadísticos implicados siguen la misma distribución normal  $N(0, 1)$ .

### Contraste Difference sign test

Supongamos la hipótesis nula  $H_{0a}$  cierta otra vez. Sea  $y_1, y_2, \dots, y_n$  una realización del proceso  $\{Y_t\}$ . Definimos la v.a.  $S$  = "Número tiempos  $i$  para los cuales  $y_i > y_{i-1}$ , siendo  $i = 1, \dots, n - 1$ ".

Para  $n$  grande, se tiene:

$$S|H_{0a} \sim N(\mu_S, \sigma_S^2), \text{ siendo } \mu_S = \frac{n-1}{2}; \sigma_S^2 = \frac{n+1}{12}$$

### PROCEDIMIENTO:

Se rechaza la hipótesis nula  $H_{0a}$  al nivel de significación  $\alpha$ , si  $\frac{|S-\mu_S|}{\sigma_S} > \phi_{1-\alpha/2}$ , donde  $\phi_{1-\alpha/2}$  es el percentil  $(1 - \alpha/2)$  de una distribución  $N(0, 1)$ .

### Contraste Rank test:

Supongamos la hipótesis nula  $H_{0a}$  cierta otra vez. Sea  $y_1, y_2, \dots, y_n$  una realización del proceso  $\{Y_t\}$ . Definimos la v.a.  $P$  = "Número de parejas  $(i,j)$  para las cuales  $y_j > y_i$ ", siendo  $i = 1, \dots, n - 1$  y  $j > i$ ".

Para  $n$  grande, se tiene:

$$P|H_{0a} \sim N(\mu_P, \sigma_P^2), \text{ siendo } \mu_P = \frac{n(n-1)}{4}; \sigma_P^2 = \frac{n(n-1)(2n+5)}{72}$$

### PROCEDIMIENTO:

Se rechaza la hipótesis nula  $H_{0a}$  al nivel de significación  $\alpha$ , si  $\frac{|P-\mu_P|}{\sigma_P} > \phi_{1-\alpha/2}$ , donde  $\phi_{1-\alpha/2}$  es el percentil  $(1 - \alpha/2)$  de una distribución  $N(0, 1)$ .

### SCRIPT DE LOS 3 ÚLTIMOS CONTRASTES:

```
1. # Test 2 "Turning Point test", "Difference-sign Test", "Rank Test"
2.
3. # Define the functions that provide the test statistics and the
   corresponding P-value
4. difference.sign.test <- function(x)
5. {
6.   DNAME <- deparse(substitute(x))
7.   n<-length(x)
8.   METHOD <- "Difference-Sign Test (NULL:iid)"
9.   X<-embed(x, 2)
10.   STATISTIC<-sum(X[, 2] > X[, 1])
11.   mu <- (n-1)/2
12.   sigma2 <- (n+1)/12
13.   PVAL<-2*(1-pnorm(abs(STATISTIC-mu)/sqrt(sigma2)))
14.   return(c(abs(STATISTIC-mu)/sqrt(sigma2),
15.           p.value = PVAL))
16. }
17.
18. turning.point.test <- function(x)
19. {
```

```

20. DNAME <- deparse (substitute (x))
21. n<-length (x)
22. METHOD <- "Turning Point Test (NULL:iid)"
23. X<-embed (x,3)
24. STATISTIC<-
    sum ((X[,2] > X[,1] & X[,2] > X[,3]) | (X[,2] < X[,1] & X[,2] < X[,3])
    )
25. mu <- 2*(n-2)/3
26. sigma2 <- (16*n-29)/90
27. PVAL<-2*(1-pnorm (abs (STATISTIC-mu)/sqrt (sigma2)))
28. return (c (abs (STATISTIC-mu)/sqrt (sigma2),
29.           p.value = PVAL))
30. }
31.
32. rank.test <- function (x)
33. {
34.     DNAME <- deparse (substitute (x))
35.     n<-length (x)
36.     METHOD <- "Rank Test (NULL:iid)"
37.     STATISTIC<-0
38.     for (i in 1:(n-1))
39.         for (j in i:n)
40.             if (x[j]>x[i]) STATISTIC<-STATISTIC+1
41.     mu <- n*(n-1)/4
42.     sigma2 <- n*(n-1)*(2*n+5)/72
43.     PVAL<-2*(1-pnorm (abs (STATISTIC-mu)/sqrt (sigma2)))
44.     return (c (abs (STATISTIC-mu)/sqrt (sigma2),
45.               p.value = PVAL))
46. }
47.
48.
49. ##### The three tests together #####
50.
51. NonParametric.Tests=function (x)
52. {
53.     A=rbind (turning.point.test (x),
54.             difference.sign.test (x),
55.             rank.test (x)
56.             )
57.     dimnames (A)=list (c ("Turning Point test", "Difference-sign
    Test", "Rank Test"), c ("Test Statistic", "P-value"))
58.     return (A)
59. }

```

Empleando el código anterior tenemos:

```

1. > NonParametric.Tests (MODELO_1$residuals)
2.           Test Statistic      P-value
3. Turning Point test      0.03922624 0.96871001
4. Difference-sign Test    1.96982121 0.04885886
5. Rank Test               0.18822060 0.85070373

```

Podemos observar que solo el contraste de la diferencia de signo nos motiva rechazar la hipótesis nula.

## 4.2 ¿LOS RESIDUOS TIENEN DISTRIBUCIÓN NORMAL?

Para contrastar la hipótesis nula  $H_{0b}$ :  $\{Y_t\} \sim N(0, \sigma^2)$  (si los residuos siguen una distribución normal) utilizaremos los siguientes contrastes que dividiremos en dos partes:

Una primera parte para verificar que se distribuyen como una normal:

1. **Contraste de Shapiro-Wilk** (preferible para muestras pequeñas,  $n \leq 30$ )
2. **Contraste de Kolmogorov-Smirnov-Lilliefors** (preferible para muestras grandes)
3. **Métodos gráficos**: QQ-plot, histograma de los residuos con la curva normal superpuesta.
4. **Contraste basado en los coeficientes de asimetría y curtosis**: (Peña, pág. 327).

Una segunda parte para:

- A) Contrastar la hipótesis nula de que la media de los residuos es cero (**contrastos de media cero**): Peña 11.3.1, pág. 326.
- B) Contrastar la hipótesis nula de que la varianza de los residuos es constante (**contraste de homocedasticidad**): Peña 11.3.2, pág. 327.

Para la primera parte el código implementado en R sería:

```

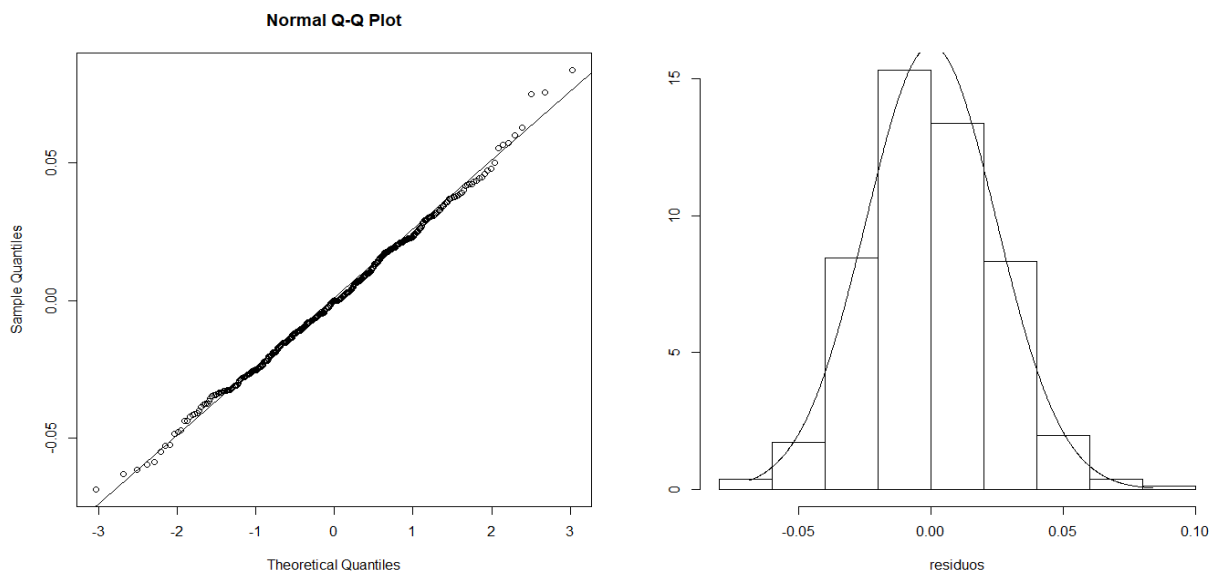
1. # Testing the hypothesis that residuals residuals follows a normal
   distribution
2.
3. # Contraste de normalidad
4.
5. TEST.normality=function(x)
6. {
7.   library(nortest)
8.   test.temp=matrix(
9.     round(c(
10.      c(shapiro.test(x)[[1]], lillie.test(x)[[1]],
11.        pearson.test(x)[[1]]),
12.      c(shapiro.test(x)[[2]], lillie.test(x)[[2]],
13.        pearson.test(x)[[2]]),
14.      ), 4), ncol=2,
15.      dimnames=list(c("Shapiro-Wilxs", "Lilliefors", "Pearson
16.      Chi.square"), c
17.      ("Statistics", "P-value"))
18.     return(test.temp)
19.   }
20.
21. Check.normality=function(x)
22. {
23.   par(mfrow=c(2, 2))
24.   qqnorm(x)
25.   qqline(x)
26.   hist(x, xlab="residuos", ylab="", main="", probability=T)
27.   range.d=c(min(x), max(x))
28.   param.d=c(mean(x), sqrt(var(x)))
29.   x.d=seq(range.d[1], range.d[2], length=1000)

```

```
29. dnorm.d=function(x){return(dnorm(x,mean=param.d[1],param.d[2]))}
30. points(x.d,dnorm.d(x.d),type="l")
31. return(TEST.normality(x))
32. }
```

Si ahora lo empleamos:

```
1. > Check.normality(MODELO_1$residuals)
2.           Statistics P-value
3. Shapiro-Wilk      0.9967  0.5780
4. Lilliefors        0.0273  0.6470
5. Pearson Chi.square 22.5735  0.3102
```



Puesto que nuestra muestra es grande y tenemos que  $n > 30$ , nos fijamos en el contraste de Lilliefors. En dicho contraste nos sale un  $P$ -value mayor que el nivel de significación. Por lo tanto, no podemos rechazar la hipótesis y asumimos la normalidad de los datos. A esta conclusión se suman con fuerza los gráficos QQ-PLOT y la gráfica de los residuos a la que le hemos superpuesto una curva normal.

Además, la gráfica QQ-PLOT nos sugiere que la media de la distribución es cero o un valor cercano a cero.

Hemos aplicado el contraste para ver si la media es cero, con esta última como hipótesis nula. El valor del  $P$ -value nos daba un rotundo 1. Deducimos que esto se debe a que la media de los residuos debe de ser según la gráfica que hemos representado arriba un valor cercano a 0 por la izquierda, e.g., -0,001. Al ser la muestra de la serie considerable el contraste detecta ese valor como distinto de 0 y niega la hipótesis nula, sin embargo, a efectos prácticos no supone un problema. Desaconsejamos el uso del contraste en este caso para sacar conclusiones dadas las evidencias que nos proporcionan las representaciones gráficas de arriba.

En cualquier caso, aquí dejo el código correspondiente:

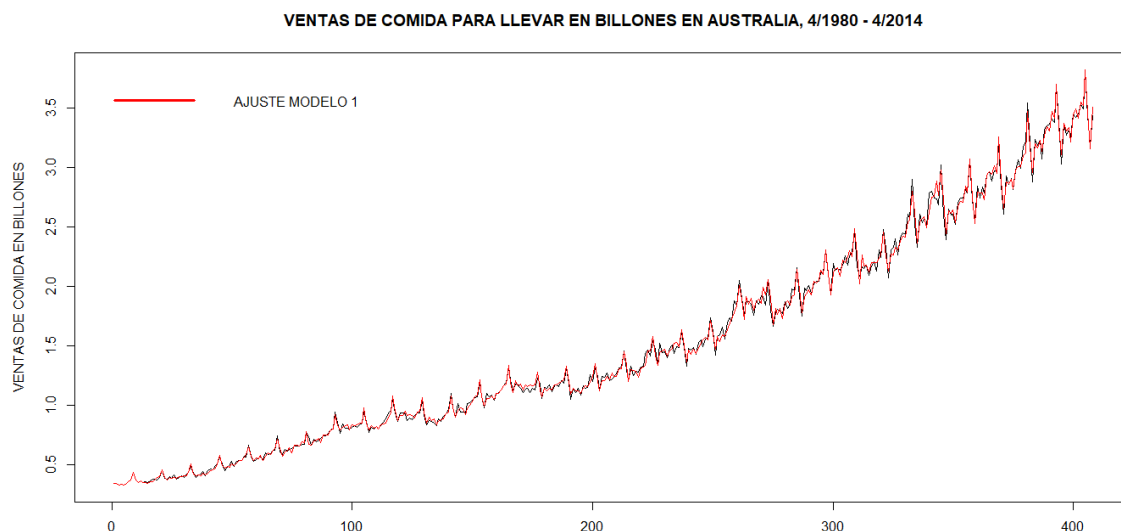
```
1. # contrastar la hipótesis que los residuos tienen media cero
2. n.res=length(MODELO_1$residuals)
3. t.obs=mean(MODELO_1$residuals)/(var(MODELO_1$residuals)/(n.res-1))
4. p.value=1-pnorm(t.obs)
```

## 5 AJUSTE DEL MODELO Y PREDICCIONES

Ajustamos el modelo:

```
1. # Ajuste del modelo a los datos
2. par(mfrow=c(1,1))
3. plot(seq(1,408), VENTAS,ylab="VENTAS DE COMIDA EN BILLONES",xlab="",type="l",main="VENTAS DE COMIDA PARA LLEVAR EN BILLONES EN AUSTRALIA, 4/1980 - 4/2014")
4. points(seq(1,408),forecast(MODELO_1)$fitted,col="red",type="l")
5. legend("topleft",col=c("red"),legend=c("AJUSTE MODELO 1"),lwd=3, bty = "n")
```

Y tenemos la imagen:



**OBSERVACIÓN IMPORTANTE:** este ajuste que se ve en la gráfica podría no ser un buen ajuste, aunque a primera vista lo parezca. Tenemos que tener en cuenta siempre la escala. En nuestro caso, medimos las ventas en billones, un error numérico pequeño, podría suponer un error muy grande cuando multipliquemos por un billón el error. Todo depende de la valoración del analista y el margen permitido para el error.

Si queremos proyectar predicciones durante los 12 meses siguientes:

```
1. > LBD <- BoxCox.out(VENTAS,12)[[2]]
```

```

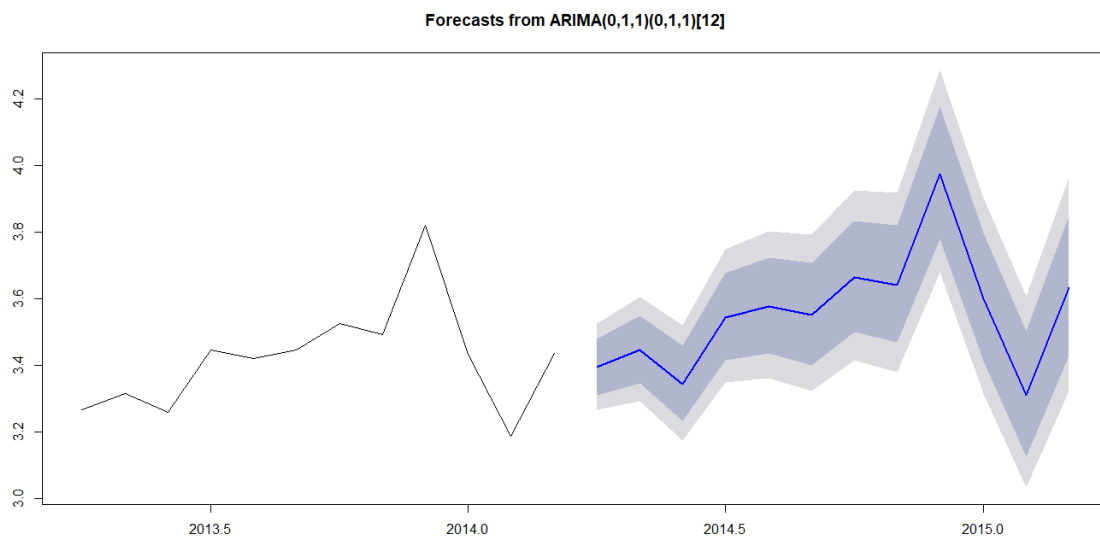
2. > LBD
3. log(xbar.vec)
4. 0.2031995
5. > z <- ts(VENTAS, start = c(1980, 4), frequency = 12) #creamos
un objeto .ts
6. > MODELO_1=Arima(z, order=c(0,1,1), seasonal=list(order=c(0,1,1), p
period=12 ), lambda=LBD)
7. > Predic.MODELO_1=forecast(MODELO_1,10)

```

```

1. > Predic.MODELO_1
2.      Point Forecast      Lo 80      Hi 80      Lo 95      Hi 95
3. Apr 2014      3.394086 3.309940 3.479927 3.266074 3.526063
4. May 2014      3.445363 3.344586 3.548545 3.292197 3.604155
5. Jun 2014      3.343814 3.232298 3.458374 3.174477 3.520271
6. Jul 2014      3.543361 3.414319 3.676258 3.347541 3.748200
7. Aug 2014      3.576020 3.434817 3.721810 3.361889 3.800880
8. Sep 2014      3.550029 3.399362 3.705970 3.321692 3.790701
9. Oct 2014      3.662731 3.498409 3.833145 3.413832 3.925880
10. Nov 2014      3.640733 3.467993 3.820261 3.379232 3.918112
11. Dec 2014      3.974232 3.779581 4.176788 3.679660 4.287295
12. Jan 2015      3.599835 3.411797 3.796040 3.315474 3.903300
13. > plot(Predic.MODELO_1,20)

```



También disponemos de los intervalos de confianza tanto al 80% como al 95%.

**Concluimos** que los residuos del modelo 1 siguen una distribución normal de media 0. Si acaso, dado que hemos negado que estos formen un ruido aleatorio, podríamos tener que los residuos estuvieran relacionados algunos entre sí, y que se fuera arrastrando un error en las predicciones a largo plazo ocasionado por alguna característica que no ha conseguido capturar nuestro modelo. En cualquier caso,



o bien podemos aceptarlo puesto que ofrecía buenas predicciones, o bien, podemos contemplar otros parámetros estacionales y ver si algún otro modelo que hayamos ignorado lo mejora.

Recomiendo por completo los modelos SARIMA para el análisis de series mensuales, ya que queda reflejado en este trabajo su alcance.

## 6 SOBRE STRATEBI



[Stratebi](#) es una empresa española, con sede en Madrid, y oficinas en Barcelona, Sevilla y Alicante, creada por un grupo de profesionales con amplia experiencia en sistemas de información, soluciones tecnológicas y procesos relacionados con soluciones de Open Source y de inteligencia de Negocio.

Esta experiencia, adquirida durante la participación en proyectos estratégicos en compañías de reconocido prestigio a nivel internacional, se ha puesto a disposición de nuestros clientes a través de Stratebi.

Stratebi es la única empresa española que ha estado presente en todos los Pentaho Developers celebrados en Europa, habiendo organizado el de Barcelona.

Stratebi es partner de grandes soluciones como Microsoft, Talend, Vertica, Apache Kylin, Hortonworks, LinceBI...

En Stratebi nos planteamos como **objetivo** dotar a las compañías e instituciones, de herramientas escalables y adaptadas a sus necesidades, que conformen una estrategia Business Intelligence capaz de rentabilizar la información disponible. Para ello, nos basamos en el desarrollo de soluciones de Inteligencia de Negocio, mediante tecnologías Open Source y Cloud

Stratebi **son** profesores y responsables de diferentes Masters en Business Intelligence y Big Data

Los profesionales de Stratebi son los creadores y autores del primer weblog en español sobre el mundo del Business Intelligence, Data Warehouse, CRM, Dashboards, Scorecard y Open Source.

Todo Bi, Portal líder de información sobre Analytics en español, se ha convertido en una referencia para el conocimiento y divulgación del Business Intelligence en español.