



Minería de datos y Modelización predictiva

Series Temporales I: Descripción y Modelos de suavizado



Tema 1 Series Temporales

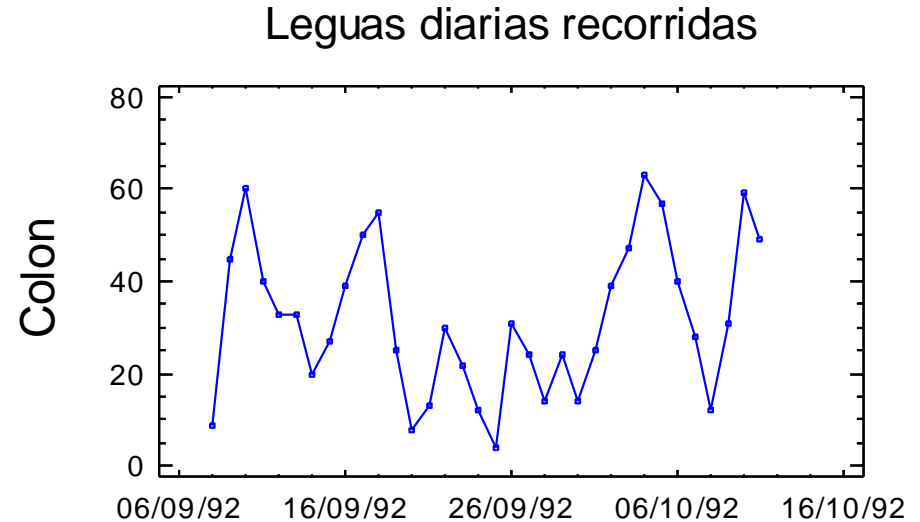
1. Introducción.
2. Representación gráfica.
3. Descomposición Estacional.
4. Métodos de suavizado.
 - 4.1 El modelo de alisado simple.
 - 4.2 Método de alisado doble de Holt.
 - 4.3 Método de suavizado para series con estacionalidad: Holt-Winters.



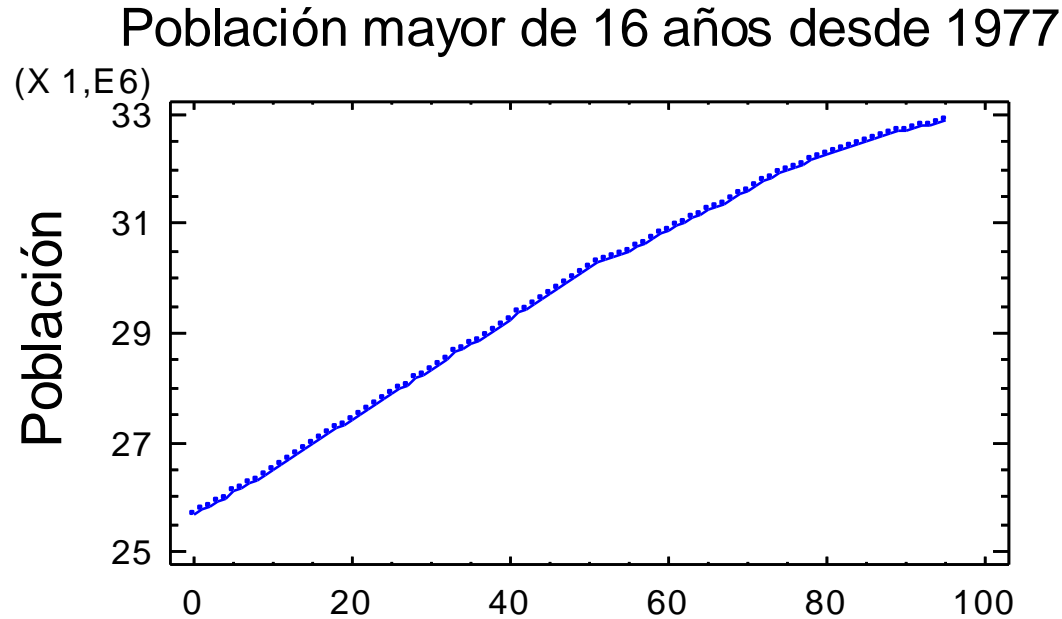
1.1 INTRODUCCIÓN

Una serie temporal es el resultado de observar los valores de una variable a lo largo del tiempo en intervalos regulares (cada día, cada mes, cada año,...)

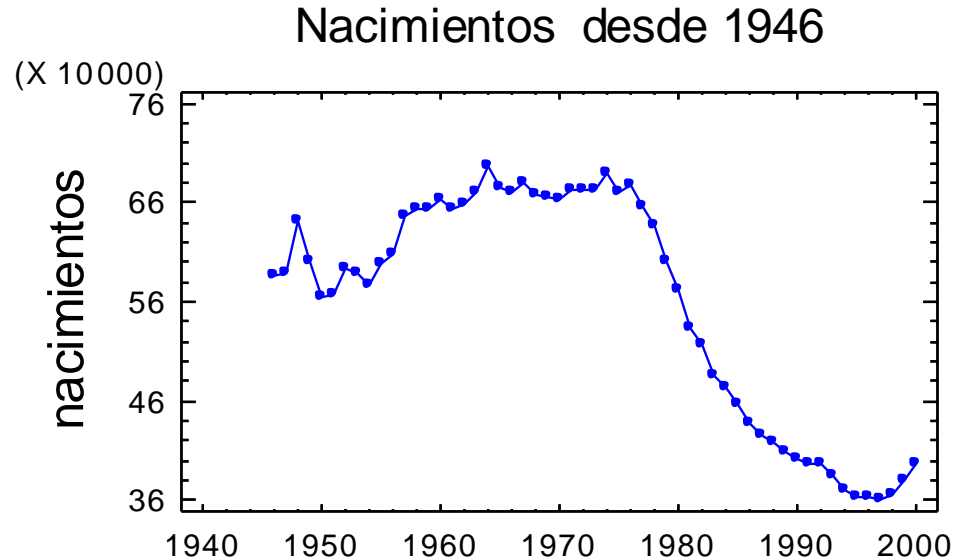
Diremos que una serie es **estacionaria** cuando **fluctúa alrededor de un nivel constante**. Por ejemplo si representamos las leguas recorridas diariamente por la flota de Colón en su primer viaje se observa que la serie es estable con valores que oscilan alrededor de un nivel fijo que en este caso son 30 leguas marinas.



Algunas series no son estacionarias sino que presentan una **tendencia** clara que **podemos modelizar con alguna función matemática (lineal, cuadrática exp,..)**, como la población mayor de 16 años en España desde el primer cuatrimestre de 1977 al cuarto trimestre del 2000.



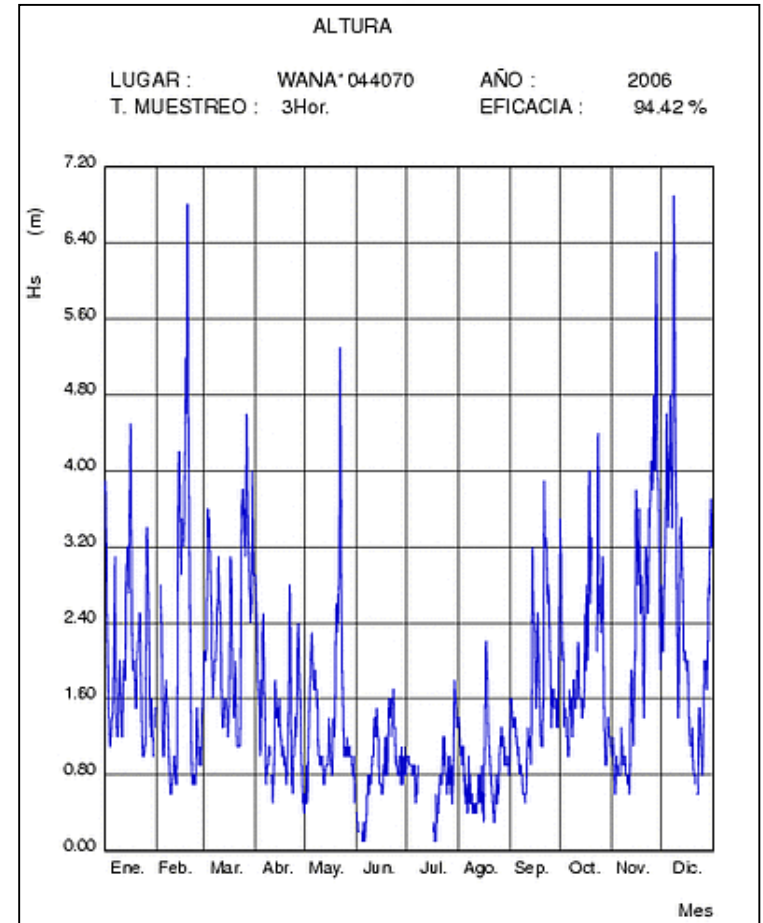
Sin embargo, la serie siguiente que representa el número de nacimientos en España entre los años 1946 al 2000 **no es estacionaria y además no presenta una tendencia clara** en todo el periodo de estudio.



La mayoría de las series económicas y sociales son no estacionarias.



En muchas ocasiones las series oscilan alrededor de un valor central pero **se comportan de forma diferente dependiendo del periodo** esto es lo que llamaremos **estacionalidad**. Un ejemplo típico es la altura de ola medida en las boyas marinas. El siguiente gráfico recoge la altura de ola recogida en la boya de Villagarcía de Arosa de Enero a Diciembre del 2006 con una frecuencia horaria.



Aún así existen otras series con periodos de repetición aún más claros, como el nivel del mar, con **fluctuaciones periódicas debidas a las mareas**. Si representamos 24 horas de la variable nivel del mar en un puerto del Atlántico, como por ejemplo en el Ferrol, observamos claramente que presenta un comportamiento estacional cuyo periodo es de 12 horas.

Gráfico de Series Temporales para Hs

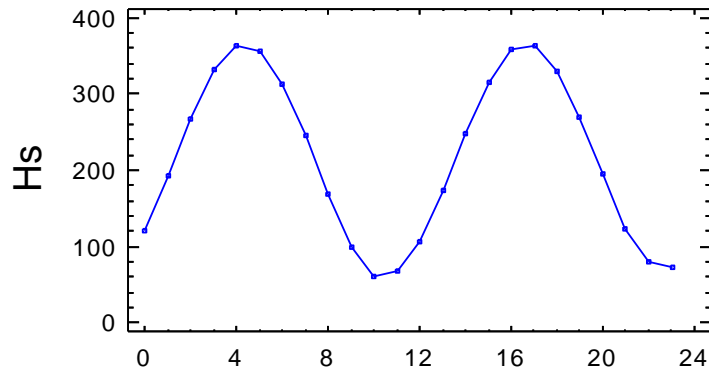
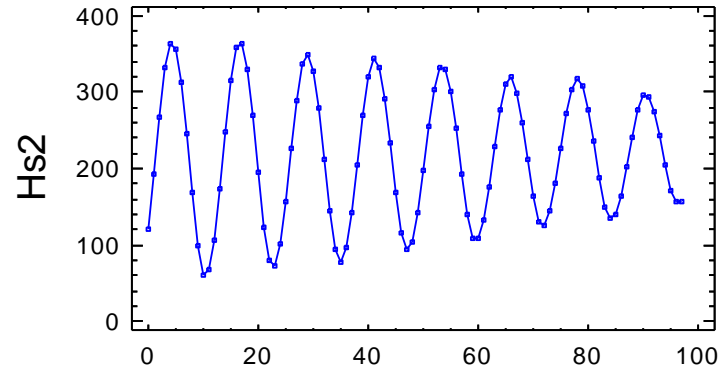
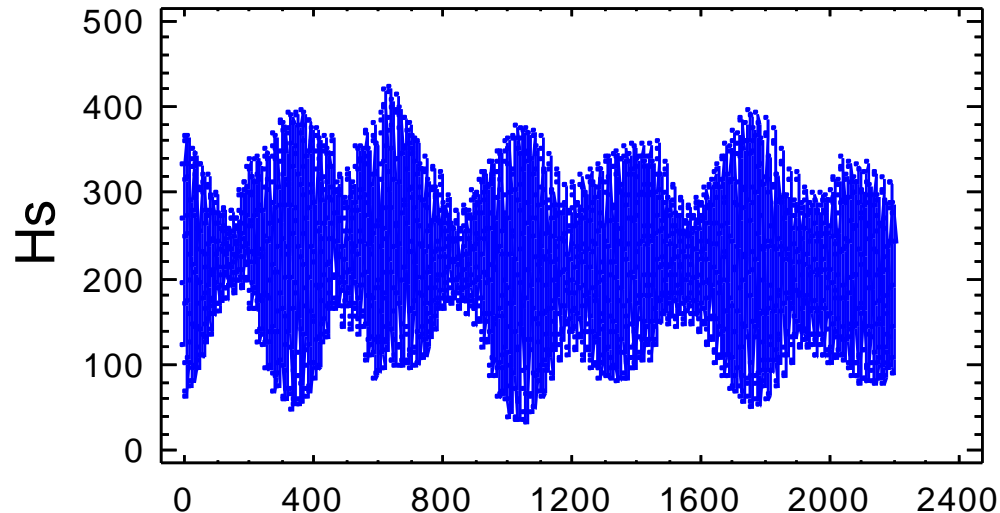


Gráfico de Series Temporales para Hs2



En ocasiones además de la estacionalidad, que se repite en periodos relativamente cortos (horas, días, meses,..), se presentan repeticiones del comportamiento de la serie en **periodos más largos que se denominan ciclos**. En la serie del nivel del mar se observa que en un trimestre tenemos seis repeticiones de ciclo que corresponden con los ciclos lunares (aproximadamente cada 28 días).

Altura de ola 4º Trimestre



1.2. REPRESENTACIÓN DE SERIES TEMPORALES.

Para cualquier análisis de series lo primero es crear un objeto **timeseries**, esto lo haremos con la función **ts** a partir de una tabla de datos.

La tabla de datos Bitcoin_A es importada desde Excel y a partir de ella creamos tres objetos time series, uno bidimensional y dos unidimensionales.

```
#Convert the data to time series
```

```
Bitcoin <- ts(bitcoin_A[,-1], start=c(2018,258), frequency=365)
```

```
precio <- ts(bitcoin_A[,2], start=c(2018,258), frequency=365)
```

```
ntrans <- ts(bitcoin_A[,3], start=c(2018,258), frequency=365)
```

En nuestro ejemplo **los datos están tomados por días, con la opción, *frequency=365***, y la fecha de inicio es 18/09/2018 (día 258 del año).



La opción **start** creará la variable fecha comenzando en la fecha indicada. El problema es que esta función está pensada con base un año y solo tiene dos argumentos: c(año, inicio).

En nuestro ejemplo **los datos están tomados por días, con la opción, *frequency=365***, y la fecha de inicio es 18/09/2018.

Si, por ejemplo, los datos son **meses y empieza en marzo sería *start=c(2018,3)*, *frequency=12***.

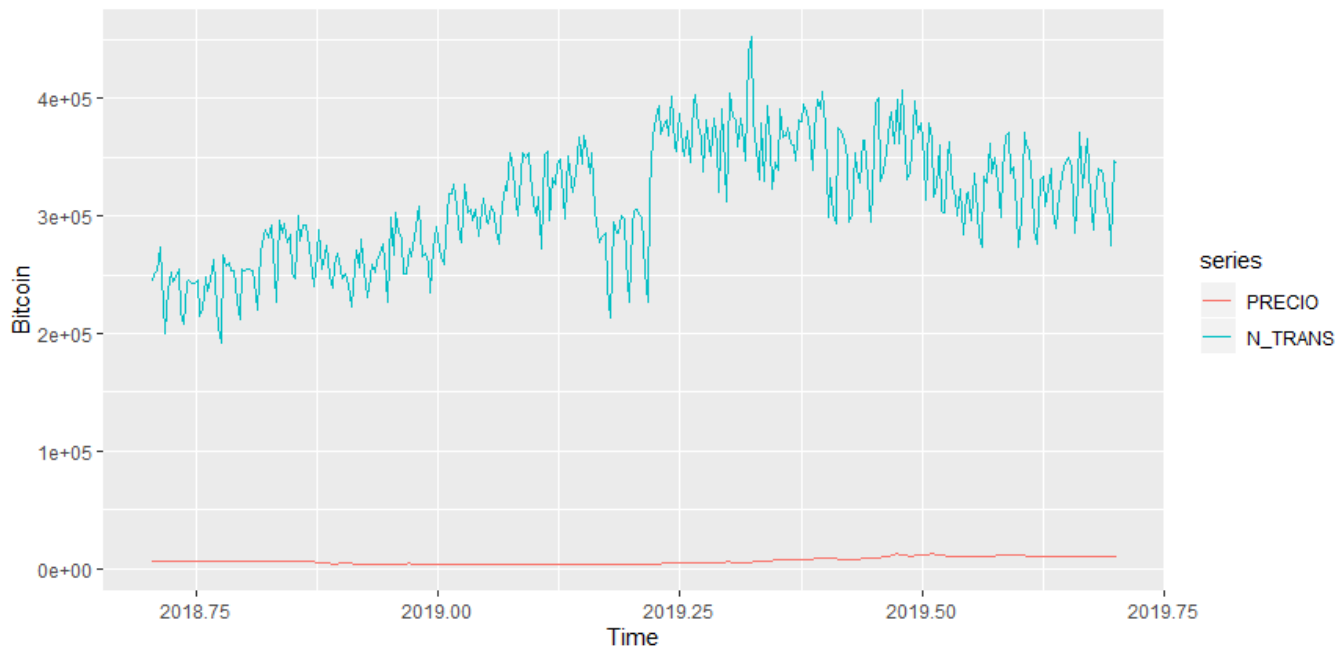
En general los valores de frecuencia temporal más utilizados son:

Data	frequency
Annual	1
Quarterly	4
Monthly	12
Weekly	52

Para representar las dos series juntas utilizamos la sintaxis:

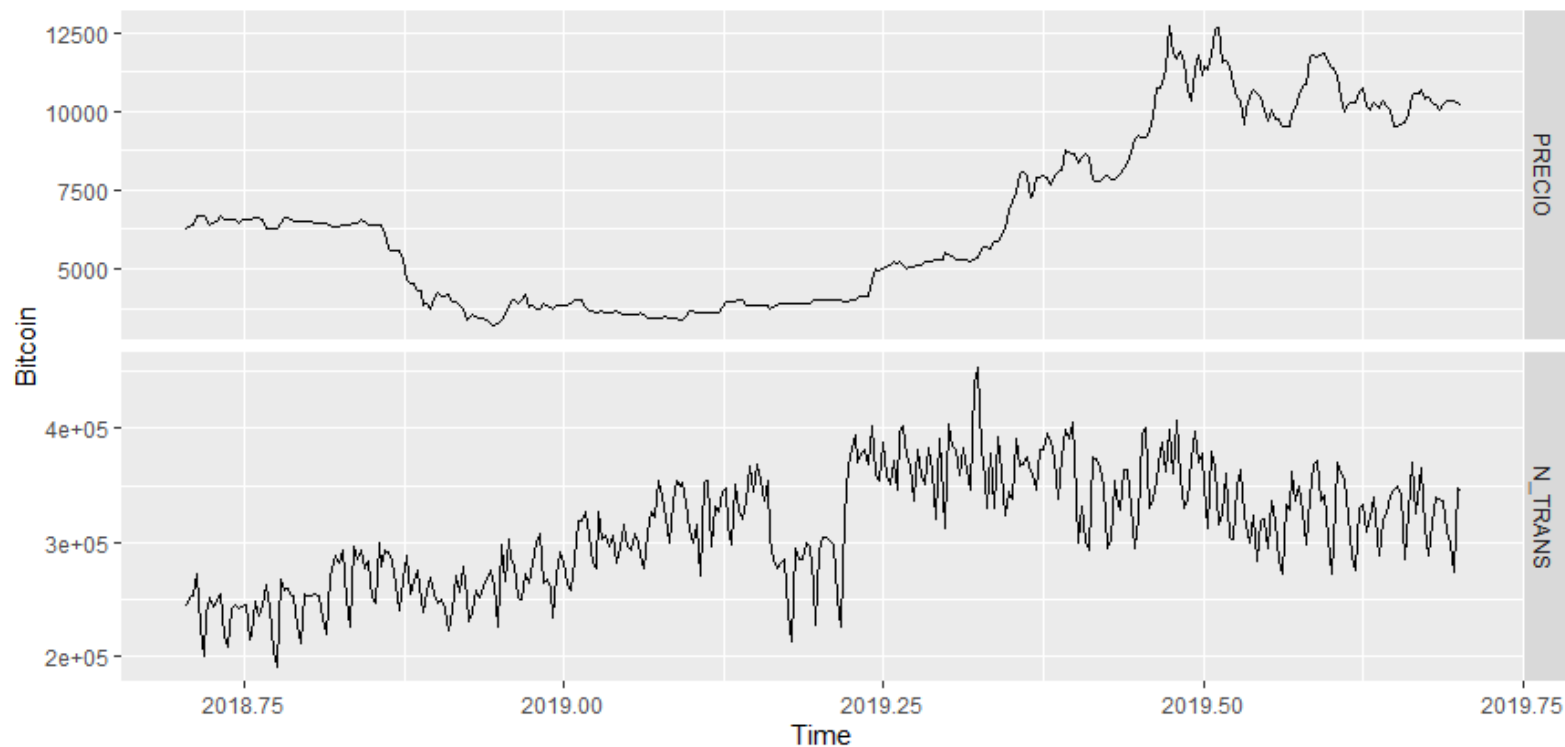
```
#Construcción de gráficas de series juntas
```

```
autoplot(Bitcoin))
```



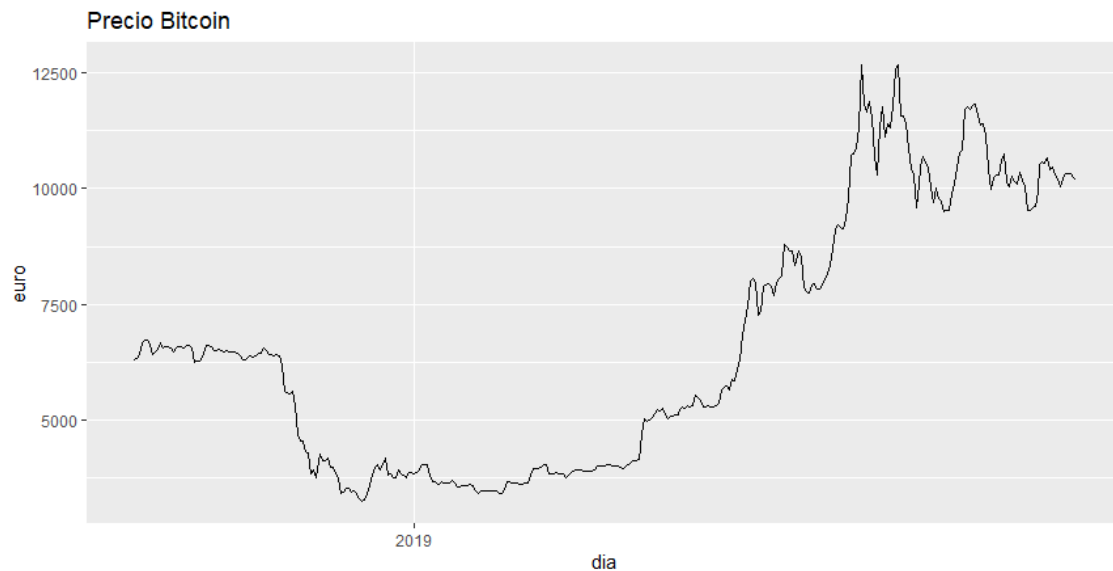
```
#Para representarlas juntas pero cada una con su escala
```

```
autoplot(Bitcoin, facets=TRUE)
```

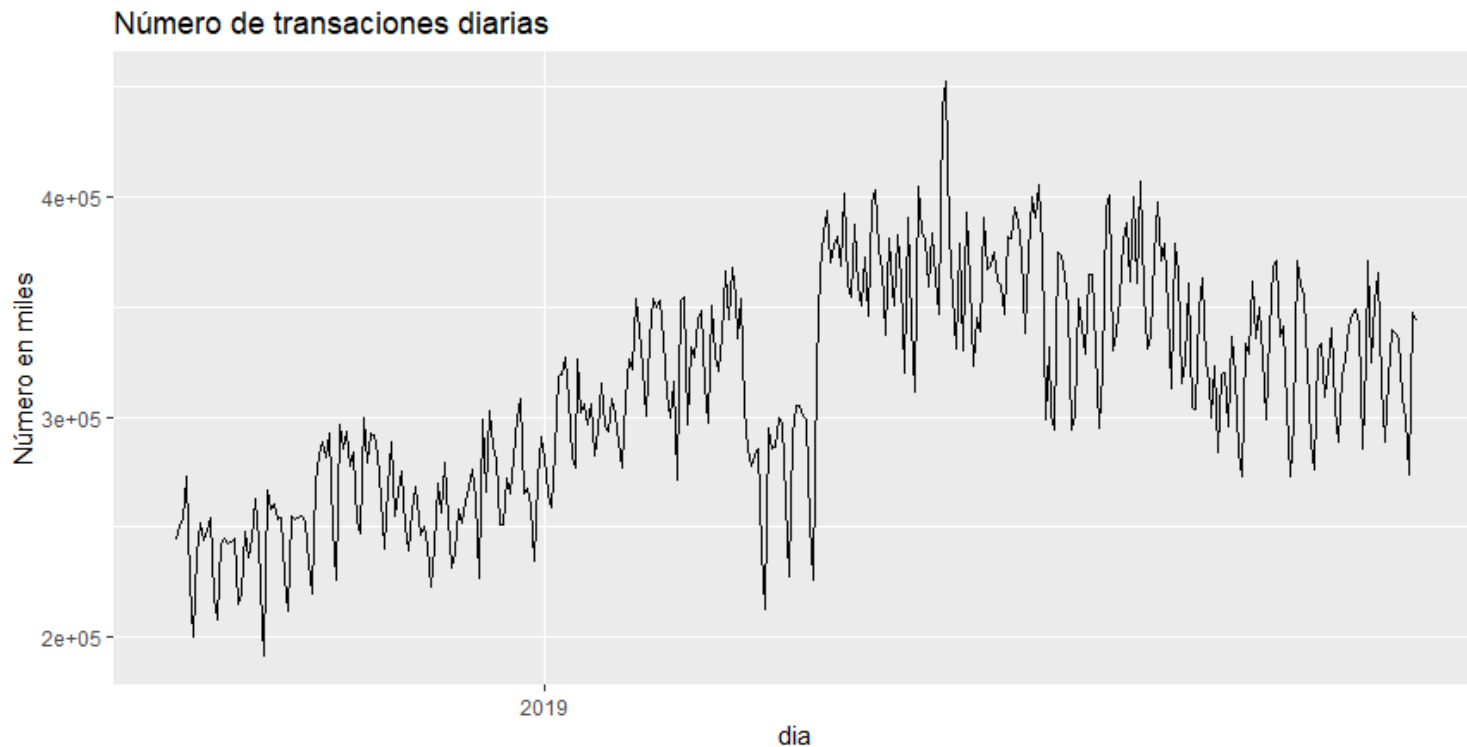


También podemos representarlas por separado añadiendo título a la gráfica y a los ejes.

```
autoplot(precio)+ ggtitle("Precio Bitcoin") + xlab("dia") + ylab("euro")
```



```
autoplot(ntrans)+ ggtitle("Número de transacciones diarias") +  
  xlab("dia") + ylab("Número en miles")
```



Si queremos que aparezca el día en el eje, podemos utilizar **la librería zoo para manejo de fechas diarias y tipo datetime.**

```
install.packages("zoo")  
library(zoo)
```



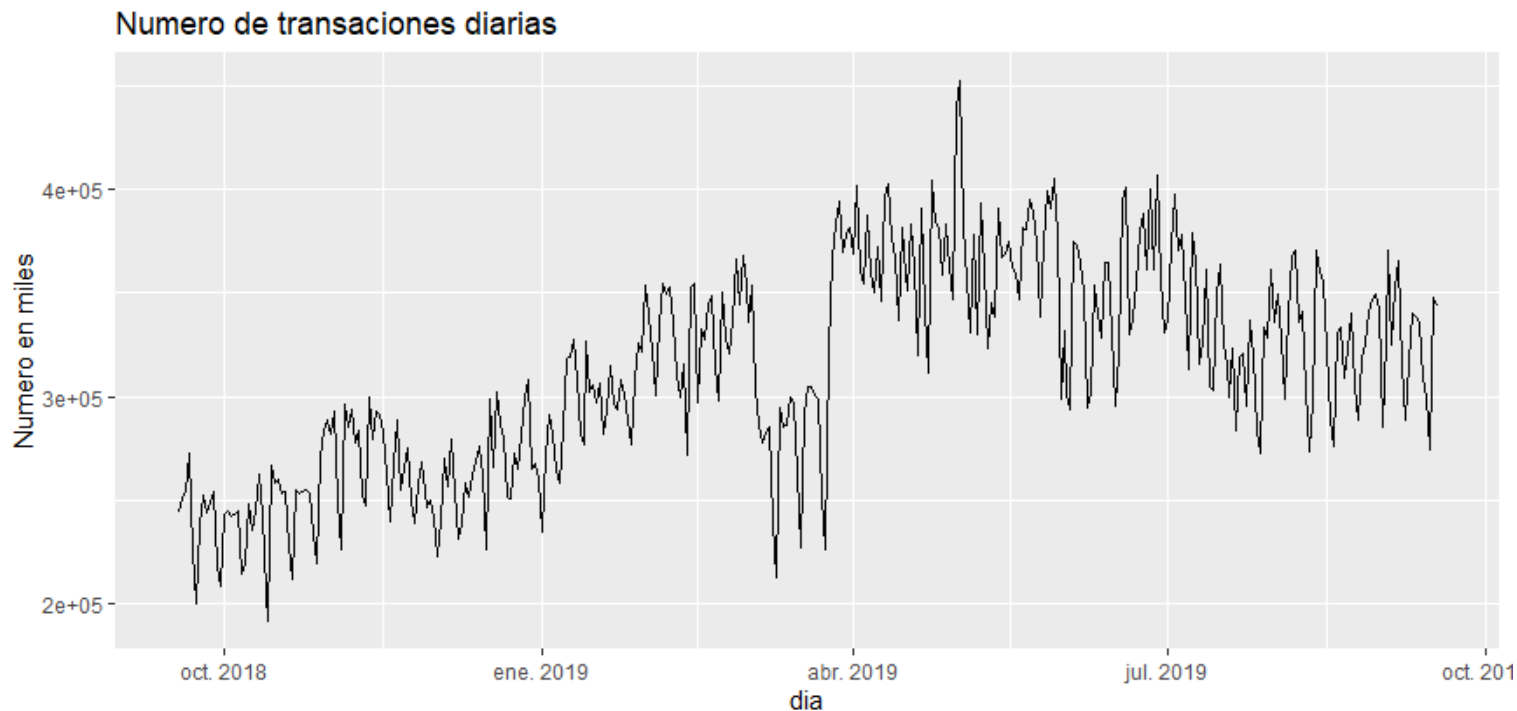
```
#Creamos la secuencia de días en los que hemos observado la serie  
dt= seq(from=as.Date("2018/09/18"), by="day", length.out=365)  
head(dt)
```

"2019-09-18" "2019-09-19" "2019-09-20" "2019-09-21" "2019-09-22" "2019-09-23"

```
#Creamos los objetos tipo zoo que son otra forma de trabajar con series de tiempo  
BITCOINz = zoo(x=bitcoin_A[,-1], order.by=dt)  
precioBz = zoo(x=bitcoin_A[,2], order.by=dt)  
ntransBz = zoo(x=bitcoin_A[,3], order.by=dt)
```

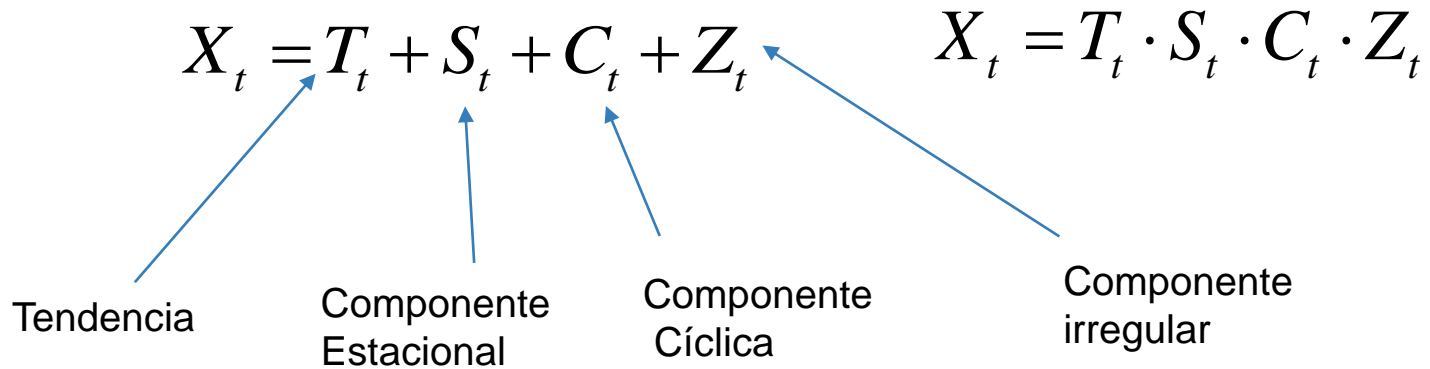


```
autoplot(ntransBz)+ ggtitle("Numero de transacciones diarias") +  
  xlab("dia") + ylab("Numero en miles")
```



3. DESCOMPOSICIÓN ESTACIONAL

Para estudiar más a fondo una serie se descompone en varios elementos: **tendencia, componente estacional, componente cíclica y componente aleatoria,**



Debido a que el comportamiento cíclico de una serie no suele tener periodos tan claros como el estacional e involucrar grandes periodos de tiempo, esta componente se suele incluir dentro de la tendencia.

$$X_t = T_t + S_t + Z_t$$

$$X_t = T_t * S_t * Z_t$$

Si tenemos una serie con periodo s , para estimar cada una de estas componentes se utilizan **medias móviles para estimar la tendencia** y **las medias de cada “mes” para estimar la componente estacional**. Este método sigue los siguientes pasos:

- Se ajusta la tendencia de la serie

$$\hat{T}_t = \sum_{k=-\lfloor s/2 \rfloor}^{k=\lfloor s/2 \rfloor} \frac{X_{t+k}}{s}$$

- Eliminamos la tendencia

$$\left(\hat{E}_t = X_t - \hat{T}_t \right) \text{ ó } \left(\hat{E}_t = X_t / \hat{T}_t \right)$$

- Los coeficientes estacionales se estiman en la serie sin tendencia como la diferencia, el cociente en el modelo multiplicativo, entre las medias de cada periodo y la media global

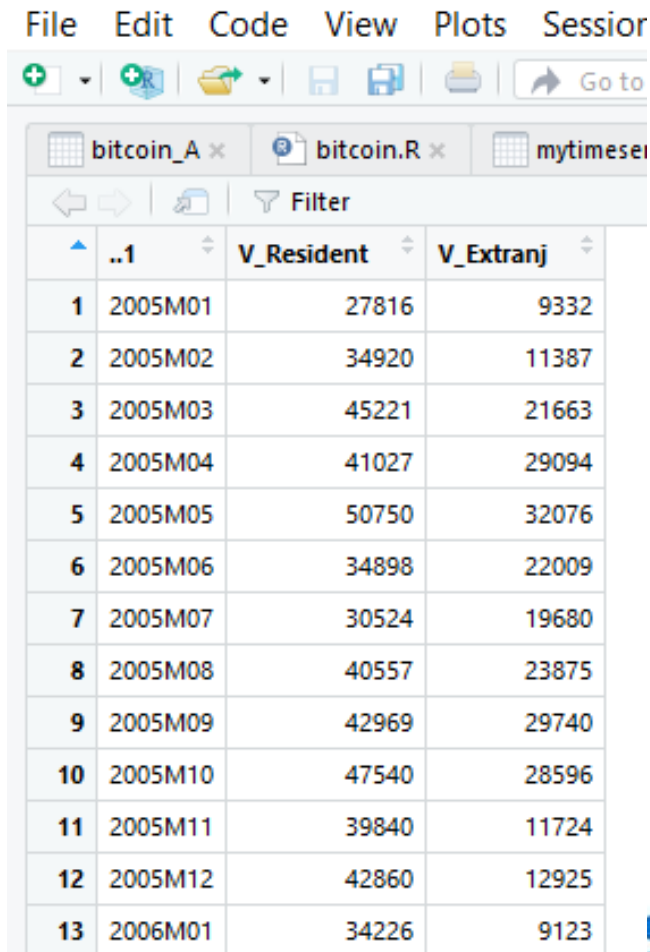
$$\left(\hat{S}_j = \bar{E}_j - \bar{E} \right) \text{ ó } \left(\hat{S}_j = \bar{E}_j / \bar{E} \right)$$

- Calculamos la serie de innovaciones estimada

$$\left(\hat{E}_t - \hat{S}_t \right) \text{ ó } \left(\hat{E}_t / \hat{S}_t \right)$$

Ejemplo: El fichero Córdoba contiene datos mensuales de viajeros alojados en hoteles en Córdoba.

```
#Dtos mensuales de viajeros alojados en hoteles de Córdoba  
v_cordoba <- ts(Cordoba[,-1], start=(2005,1), frequency=12)  
v_cordoba_R <- ts(Cordoba[,2], start=(2005,1), frequency=12)  
v_cordoba_E <- ts(Cordoba[,3], start=(2005,1), frequency=12)
```

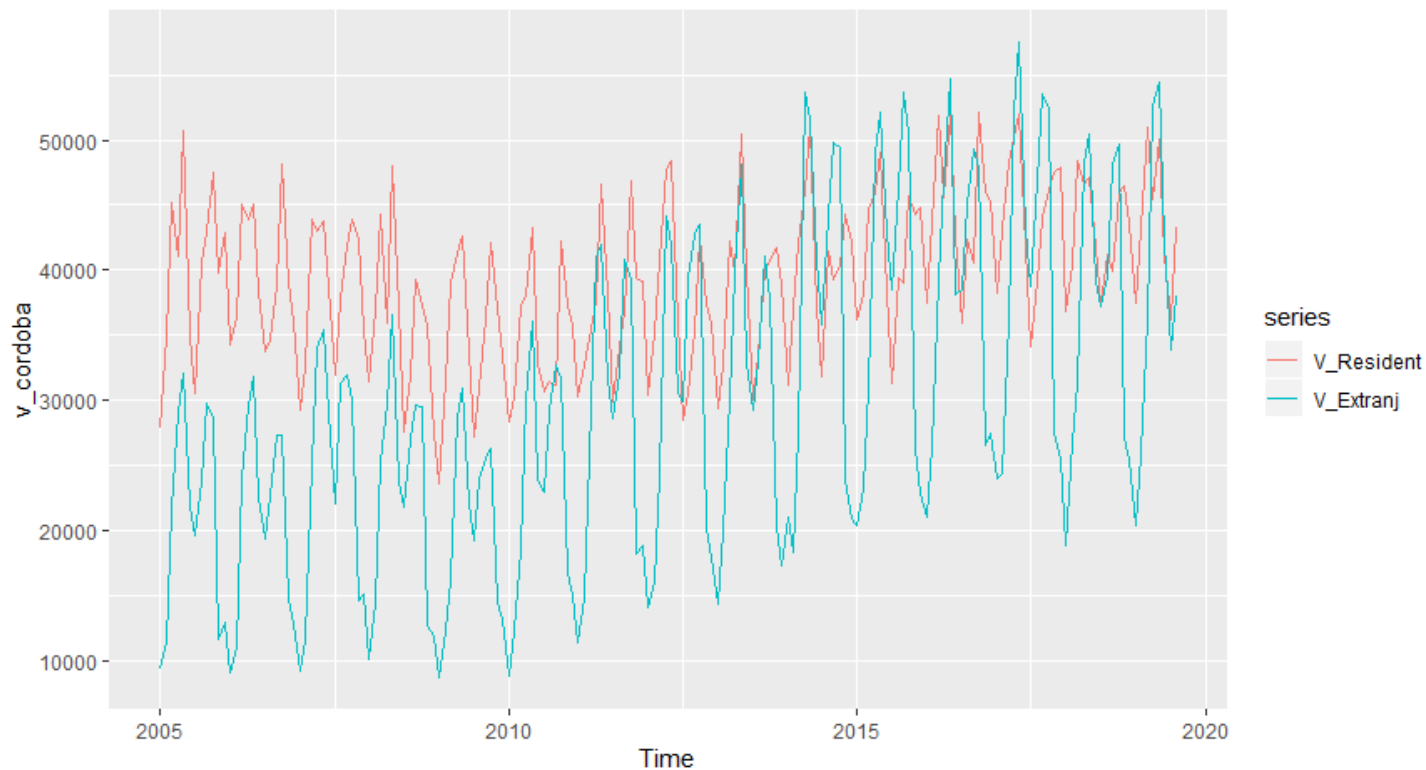


The screenshot shows the RStudio interface with three open files: bitcoin_A, bitcoin.R, and mytimeser. The bitcoin.R file is active, displaying a data table with three columns: an index column (1-13), a date column (2005M01 to 2006M01), and two numeric columns labeled V_Resident and V_Extranj. The table contains 13 rows of monthly data.

	..1	V_Resident	V_Extranj
1	2005M01	27816	9332
2	2005M02	34920	11387
3	2005M03	45221	21663
4	2005M04	41027	29094
5	2005M05	50750	32076
6	2005M06	34898	22009
7	2005M07	30524	19680
8	2005M08	40557	23875
9	2005M09	42969	29740
10	2005M10	47540	28596
11	2005M11	39840	11724
12	2005M12	42860	12925
13	2006M01	34226	9123

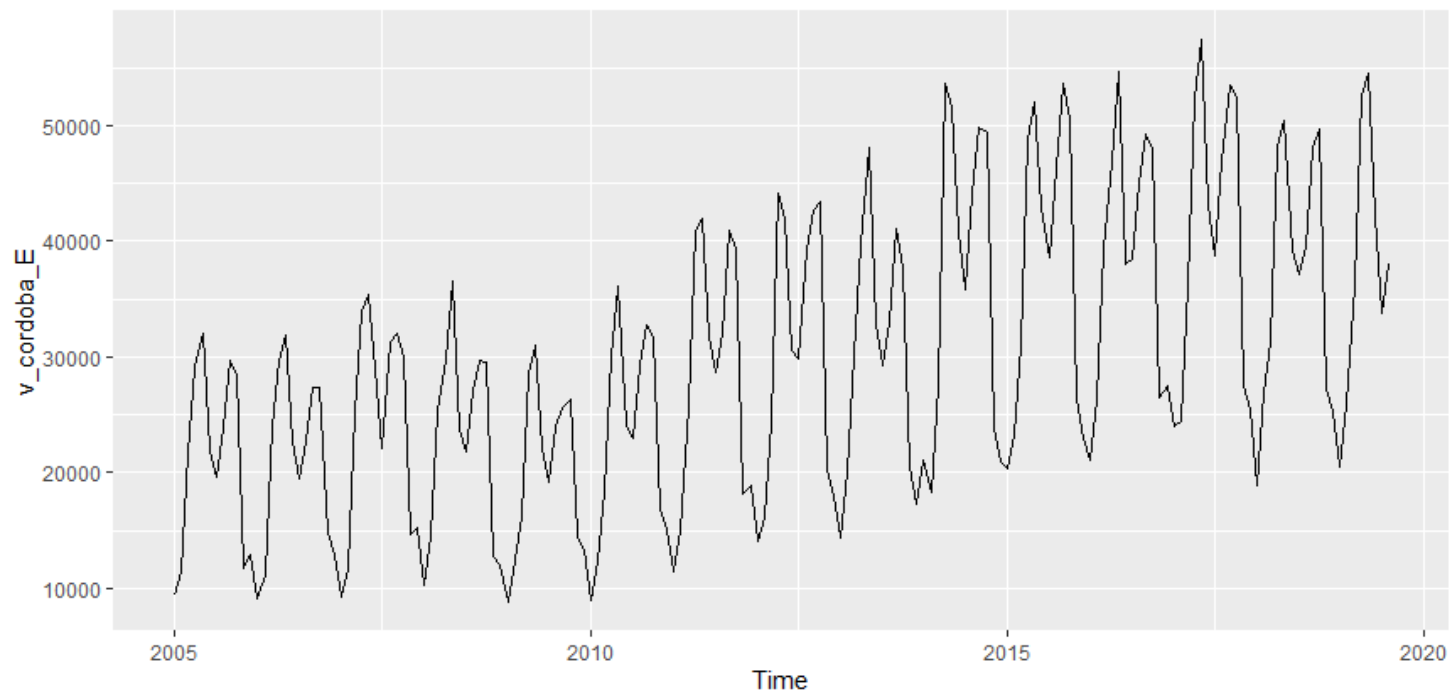
Comenzamos por representar las series de forma conjunta:

```
autoplot(v_cordoba)
```



Nos centramos en la serie viajeros extranjeros. La representamos

```
autoplot(v_cordoba_E)
```



Realizamos la descomposición estacional según el modelo multiplicativo, esto debemos indicarlo porque por defecto ajusta el modelo aditivo y representamos sus componentes

```
#Guardamos los componentes de la descomposición estacional en  
v_cordoba_E_Comp<- decompose(v_cordoba_E,type=c("multiplicative"))
```

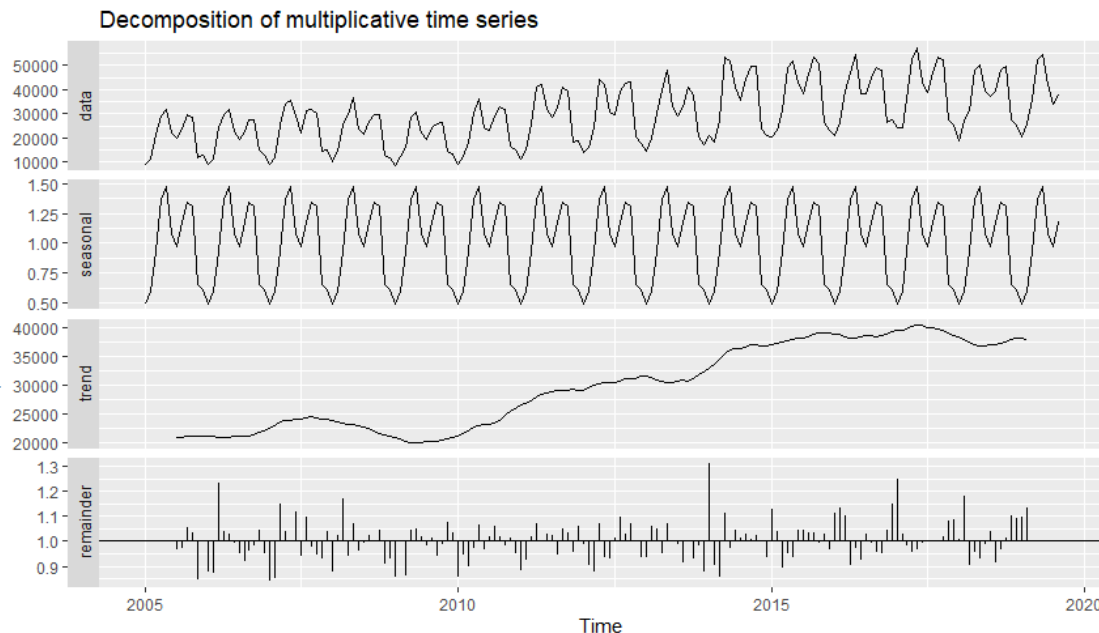
Representamos los componentes de la serie obtenidos.

```
autoplot(v_cordoba_E_Comp)
```

$$\left(\hat{S}_j = \bar{E}_j / \bar{E} \right) \longrightarrow$$

$$\hat{T}_t = \sum_{k=-\lfloor s/2 \rfloor}^{k=\lfloor s/2 \rfloor} \frac{X_{t+k}}{s} \longrightarrow$$

$$\left(\hat{E}_t / \hat{S}_t \right) \longrightarrow$$



En v_cordoba_E_Comp tenemos guardadas las tablas de los diferentes componentes que se obtienen de la descomposición estacional

```
print(v_cordoba_E_Comp)
```

Por ejemplo, los coeficientes de estacionalidad son los siguientes.

```
$seasonal
```

```
0.4896350 0.6019633 0.9276545 1.3666168 1.4767595 1.0800872 0.9695678 1.174  
0330 1.3413241 1.3078890 0.6538099 0.6106598
```

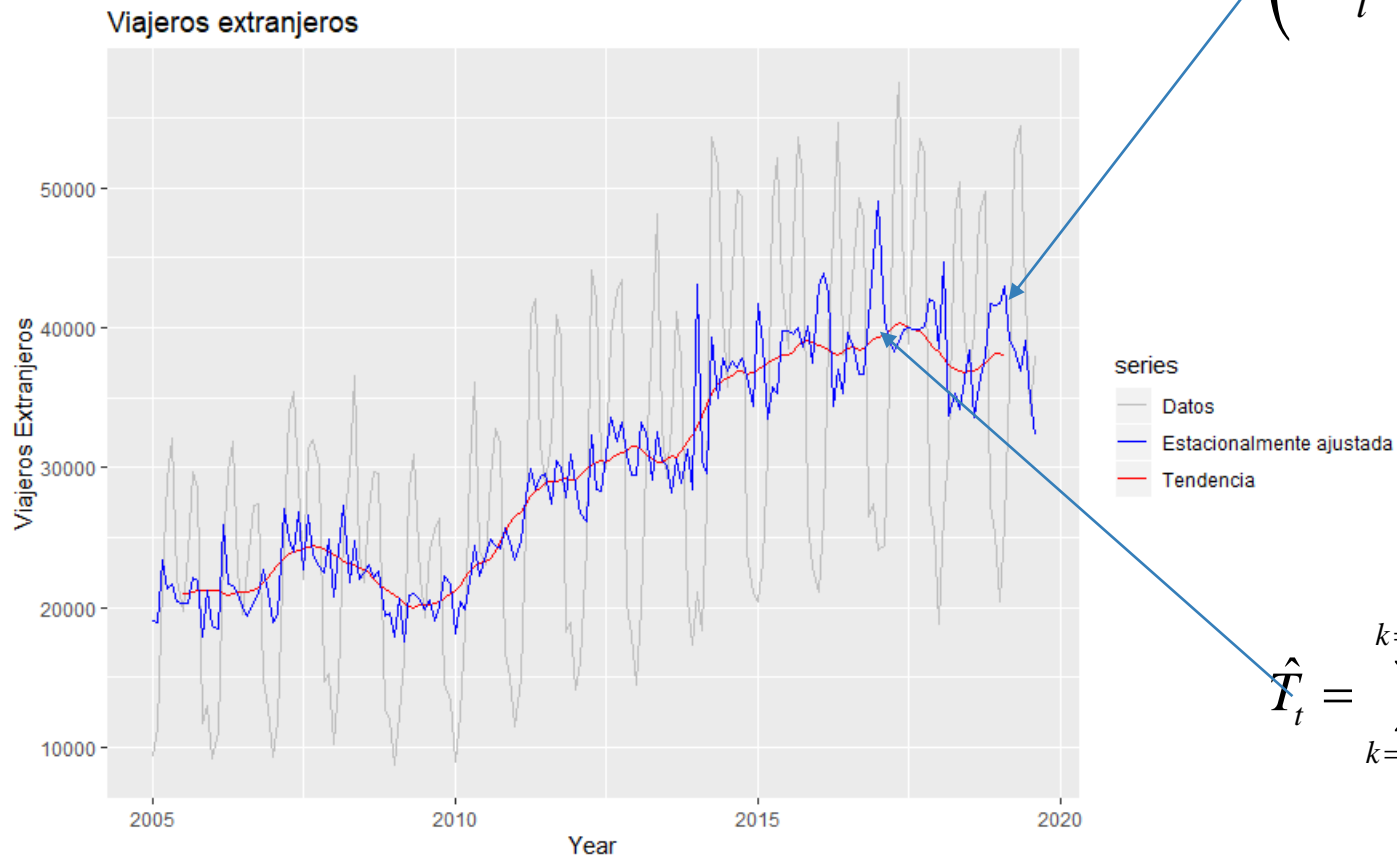
En el mes de mayo hay un 47% más de viajeros que la media del año

#Representamos la serie con la tendencia y la serie ajustada estacionalmente

```
autoplot(v_cordoba_E, series="Datos") +  
  autolayer(trendcycle(v_cordoba_E_Comp), series="Tendencia") +  
  autolayer(seasadj(v_cordoba_E_Comp), series="Estacionalmente ajustada") +  
  xlab("Year") + ylab("Viajeros Extranjeros") +  
  ggtitle("Viajeros extranjeros") +  
  scale_colour_manual(values=c("gray","blue","red"),  
    breaks=c("Datos","Estacionalmente ajustada","Tendencia"))
```

$$\hat{T}_t = \sum_{k=-\lceil s/2 \rceil}^{\lceil s/2 \rceil} \frac{X_{t+k}}{s}$$

$$(X_t - \hat{S}_t) \text{ ó } (X_t / \hat{S}_t)$$

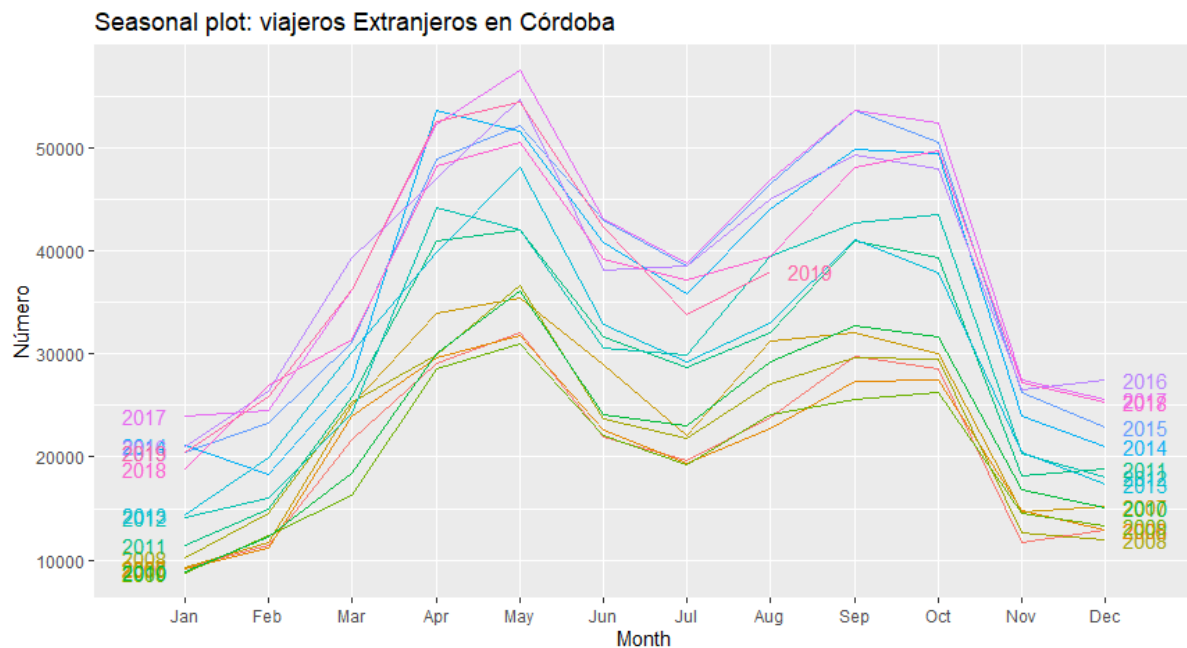


$$\left(X_t / \hat{S}_t \right)$$

$$\hat{T}_t = \sum_{k=-\lfloor s/2 \rfloor}^{k=\lfloor s/2 \rfloor} \frac{X_{t+k}}{s}$$

Para estudiar el comportamiento estacional resulta útil la representación gráfica de los valores de la serie, dibujando cada año en un color diferente.

```
ggseasonplot(v_cordoba_E, year.labels=TRUE, year.labels.left=TRUE) +  
  ylab("Número") +  
  ggtitle("Seasonal plot: viajeros Extranjeros en Córdoba")
```



4. MÉTODOS DE SUAVIZADO

El objetivo es hacer predicciones utilizando los datos de la serie eliminando las fluctuaciones aleatorias y quedándonos solo con la componente tendencia o tendencia- estacionalidad

4.1 El modelo de alisado simple.

Este método se utiliza cuando la serie no presenta tendencia creciente o decreciente, es decir podemos modelizarla como

$$X_t = L_t + z_t \quad \hat{x}_{t+1} = \alpha x_t + (1 - \alpha) \hat{x}_t \quad 0 \leq \alpha \leq 1$$

Repitiendo este proceso tenemos

$$\hat{x}_{t+1} = \alpha x_t + (1 - \alpha) \alpha x_{t-1} + \dots + \alpha (1 - \alpha)^{t-1} x_1 + (1 - \alpha)^t \hat{x}_1$$

$$\hat{x}_{t+1} = \alpha \left(x_t + (1 - \alpha) x_{t-1} + (1 - \alpha)^2 x_{t-2} + \dots \right)$$

Que es la media ponderada de las observaciones anteriores con pesos decrecientes que suman uno.

La ecuación de predicción para valores futuros

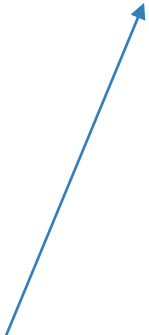
$$\hat{x}_{n+m} = \hat{x}_{n+1} \quad m \geq 2$$

Ejemplo. La librería `forecast` nos permite hacer los métodos de suavizado explicados anteriormente. Utilizaremos los datos del precio de Bitcoin que mostramos antes, pero **nos quedamos con los datos finales. Para ello utilizamos la función `window()`** que es muy útil para extraer parte de los datos de una serie.

```
#Seleccionamos los valores de precio desde agosto de 2019  
precio_R<-window(precio,start=c(2019,210))  
  
# Igual funciona para serie creada con zoo  
precio_Rz<-window(precioBz,start="2019/08/01")
```

```
# Hacemos un suavizado exponencial simple con la función ses y calculamos la  
#predicción para 7 días  
precio_s1=ses(precio_R, h=7)
```

```
#Para ver los parámetros del modelo ajustado  
precio_s1[["model"]]
```

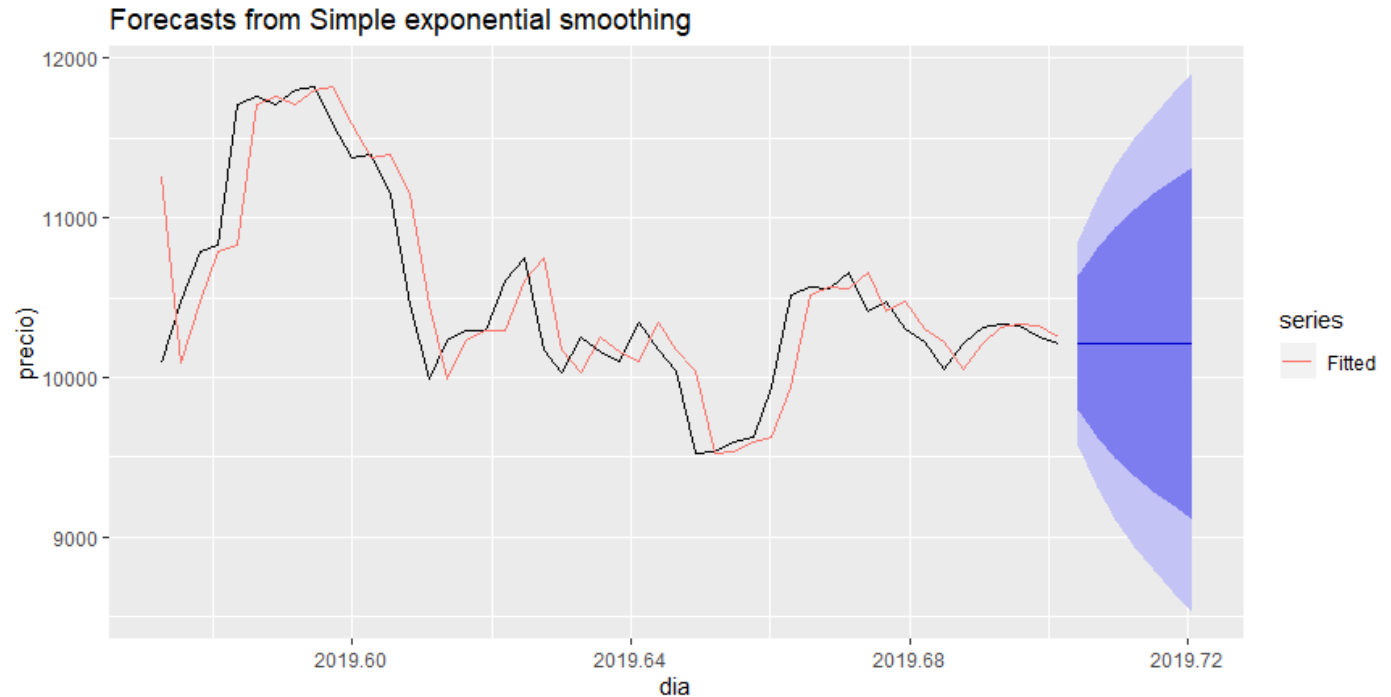
$$\hat{x}_{t+1} = 0.999x_t + 0.001\hat{x}_t$$


\$par		
alpha		1
0.9999	11253.8011	

```
#Representamos los valores observados y los suavizados con la predicción para 7  
#días
```

```
autoplot(precio_s1) +
```

```
  autolayer(fitted(precio_s1), series="Fitted") + ylab("precio") + xlab("dia")
```



Método de alisado doble de Holt.

El método de suavizado anterior no proporciona buenos resultados cuando los datos tienen una tendencia que varía más rápidamente. **Este método supone que la tendencia es lineal pero su pendiente varía con el tiempo**

$$x_t = L_t + b_t t + z_t$$

De la serie original obtendremos la serie suavizada mediante las ecuaciones:

$$L_t = \alpha x_t + (1 - \alpha)(L_{t-1} + b_{t-1})$$

$$L_0 = \bar{x}$$

$$b_t = \beta(L_t - L_{t-1}) + (1 - \beta)b_{t-1}$$

$$b_0 = x_2 - x_1$$

$$\hat{x}_{t+1} = L_t + b_t$$

La ecuación de predicción para valores futuros supuesto que hemos observado hasta el tiempo n es

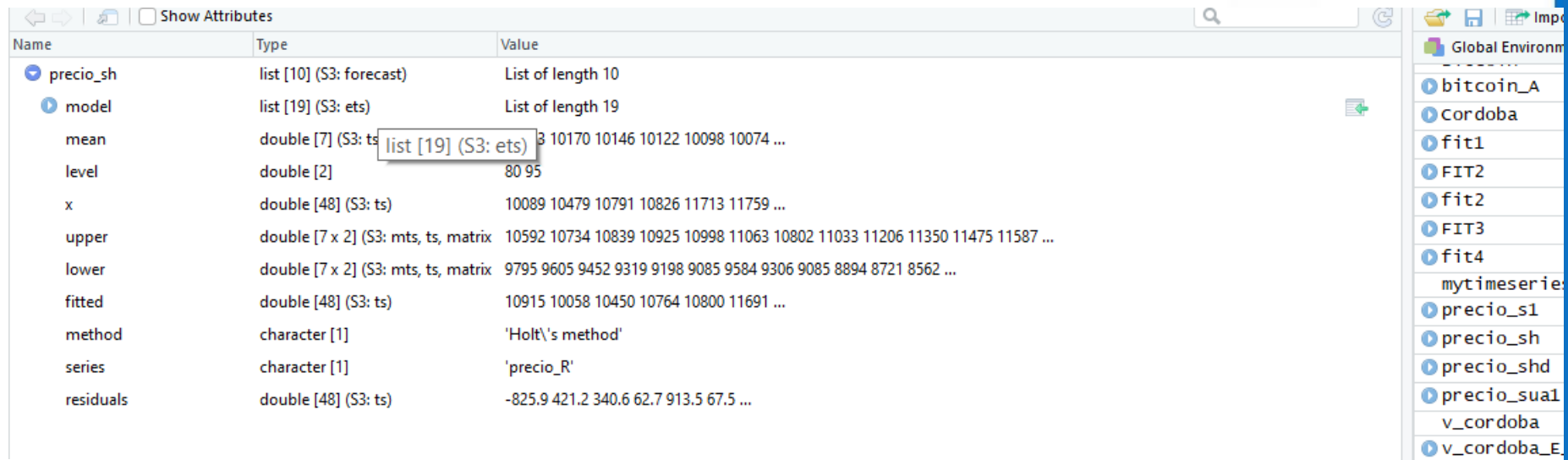
$$\hat{x}_{n+m} = L_n + b_n m \quad m \geq 1$$

Ejemplo: Continuamos con el ejemplo del precio de Bitcoin y ahora utilizamos la función holt:

```
precio_sh <- holt(precio_R, h=7)  
  
print(precio_sh)
```

Point	Forecast	Lo 80	Hi 80	Lo 95	Hi 95
2019.7041	10193.37	9795.118	10591.62	9584.298	10802.44
2019.7068	10169.53	9604.791	10734.27	9305.835	11033.23
2019.7096	10145.70	9452.132	10839.26	9084.982	11206.41
2019.7123	10121.86	9318.799	10924.92	8893.683	11350.04
2019.7151	10098.03	9197.706	10998.35	8721.106	11474.95
2019.7178	10074.19	9085.232	11063.15	8561.709	11586.67
2019.7205	10050.35	8979.229	11121.48	8412.209	11688.50

Para ver los parámetros del modelo hacemos click en precio_sh y en la model



Name	Type	Value
precio_sh	list [10] (S3: forecast)	List of length 10
model	list [19] (S3: ets)	List of length 19
mean	double [7] (S3: ts)	3 10170 10146 10122 10098 10074 ...
level	double [2]	80 95
x	double [48] (S3: ts)	10089 10479 10791 10826 11713 11759 ...
upper	double [7 x 2] (S3: mts, ts, matrix)	10592 10734 10839 10925 10998 11063 10802 11033 11206 11350 11475 11587 ...
lower	double [7 x 2] (S3: mts, ts, matrix)	9795 9605 9452 9319 9198 9085 9584 9306 9085 8894 8721 8562 ...
fitted	double [48] (S3: ts)	10915 10058 10450 10764 10800 11691 ...
method	character [1]	'Holt's method'
series	character [1]	'precio_R'
residuals	double [48] (S3: ts)	-825.9 421.2 340.6 62.7 913.5 67.5 ...

```
> precio_sh[["model"]]
```

Smoothing parameters:

alpha = 0.9999

beta = 0.0055

Initial states:

l = 10941.3217 $L_0 = \bar{x}$

b = -26.6411 $b_0 = x_2 - x_1$

$$L_t = 0.99x_t + 0.01(L_{t-1} + b_{t-1})$$

$$b_t = 0.055(L_t - L_{t-1}) + 0.955b_{t-1}$$

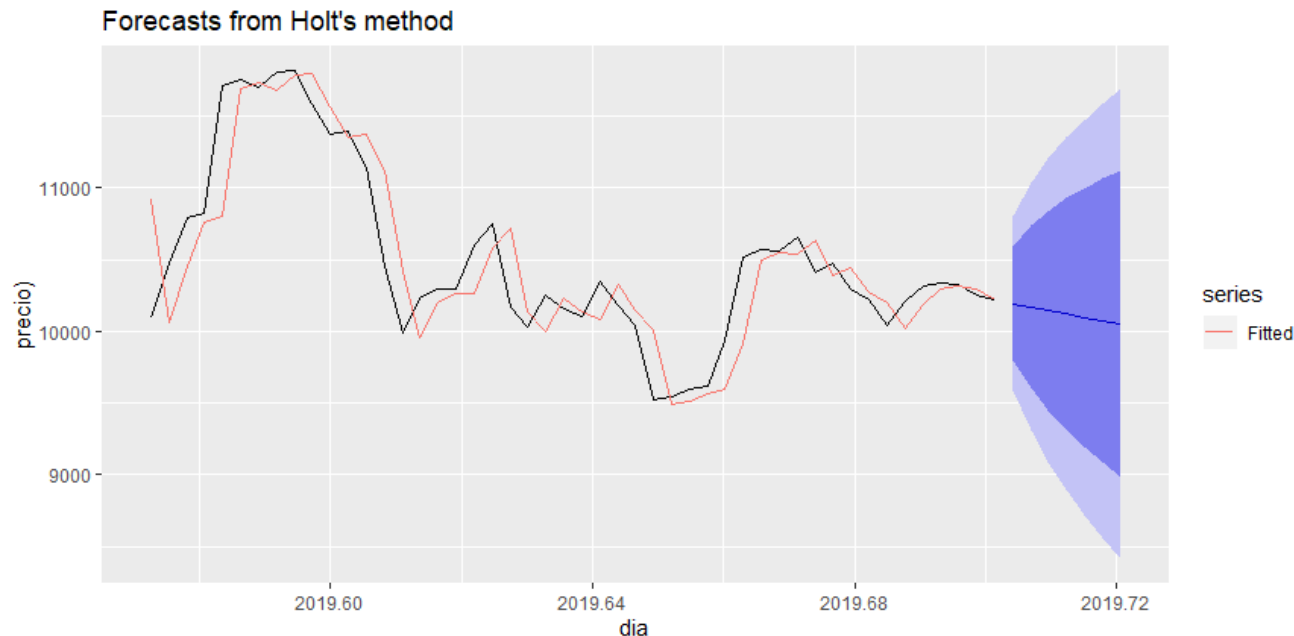

```
autoplot(precio_sh) +  
  autolayer(fitted(precio_sh), series="Fitted") +  
  ylab("precio") + xlab("dia")
```

$$L_t = 0.99x_t + 0.01(L_{t-1} + b_{t-1})$$

$$b_t = 0.055(L_t - L_{t-1}) + 0.955b_{t-1}$$

$$\hat{x}_{n+m} = L_n + b_n m \quad m \geq 1$$

```
> precio_sh[["model"]]  
Smoothing parameters:  
  alpha = 0.9999  
  beta  = 0.0055  
Initial states:  
  l = 10941.3217  
  b = -26.6411
```



Método de suavizado para series con estacionalidad: Holt-Winters.

Cuando existe estacionalidad el efecto debe modelarse con un modelo:

$$X_t = (L_t + b_t) + S_t + z_t \quad \text{si la incidencia de la estacionalidad no aumenta con el tiempo}$$

$$X_t = (L_t + b_t) \cdot S_t + z_t \quad \text{si las variaciones estacionales aumentan con el tiempo.}$$

La serie suavizada para los modelos multiplicativo y aditivo se obtiene aplicando las siguientes fórmulas:

Modelo Multiplicativo

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

$$b_t = \beta (L_t - L_{t-1}) + (1 - \beta)b_{t-1}$$

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

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

Modelo Aditivo

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

$$b_t = \beta (L_t - L_{t-1}) + (1 - \beta)b_{t-1}$$

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

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



En este modelo **los valores iniciales de la tendencia se estiman a partir de la media de los valores del primer ciclo**

$$L_0 = \frac{x_1 + \dots + x_s}{s}$$

Para estimar la pendiente necesitamos conocer al menos dos ciclos completos y se inicia con:

$$b_0 = \frac{1}{s} \left(\frac{(x_{s+1} - x_1)}{s} + \frac{(x_{s+2} - x_2)}{s} + \dots + \frac{(x_{s+s} - x_s)}{s} \right)$$

Finalmente **los índices estacionales se estiman con los valores del primer periodo:**

En el modelo multiplicativo:

$$S_1 = \frac{x_1}{L_0}, S_2 = \frac{x_2}{L_0}, \dots, S_s = \frac{x_s}{L_0},$$

En el modelo aditivo:

$$S_1 = x_1 - L_0, S_2 = x_2 - L_0, \dots, S_s = x_s - L_0$$

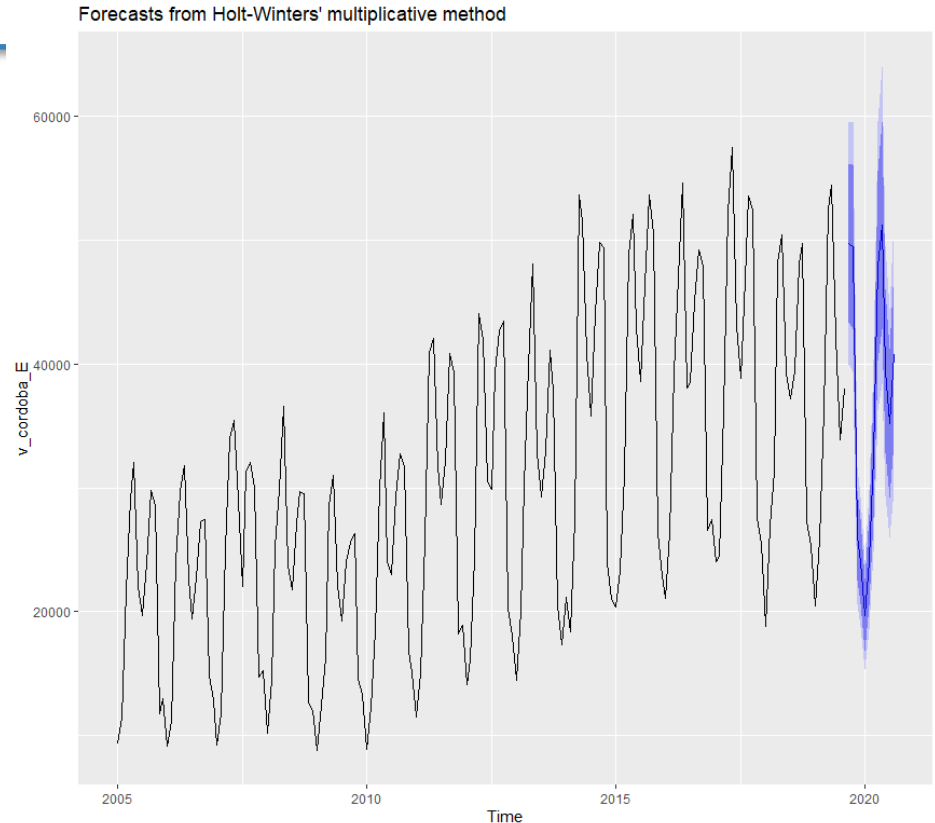
Las ecuaciones de predicción para valores futuros

$$\hat{x}_{n+m} = (L_n + b_n m) \cdot S_{n+m-s} \quad m \geq 1$$

$$\hat{x}_{n+m} = (L_n + b_n m) + S_{n+m-s} \quad m \geq 1$$

```
fit1 <- hw(v_cordoba_E,h=12, seasonal="multiplicative",level = c(80, 95))  
autoplot(fit1)
```

Por ejemplo, si queremos predecir el número de viajeros extranjeros que vendrán a Córdoba un año después del último observado, utilizaremos el modelo de Holt-winters multiplicativo porque la serie es estacional. Guardamos los resultados del modelo ajustado en fit1.



```
print(fit1)
```

		Point Forecast	Lo 80	Hi 80	Lo 95	Hi 95
Sep	2019	49749.65	43350.41	56148.89	39962.86	59536.44
Oct	2019	49481.59	42875.06	56088.12	39377.77	59585.41
Nov	2019	26136.11	22523.34	29748.87	20610.86	31661.35
Dec	2019	24048.22	20614.43	27482.00	18796.69	29299.74
Jan	2020	19712.70	16810.88	22614.52	15274.75	24150.65
Feb	2020	24148.23	20489.90	27806.57	18553.29	29743.18
Mar	2020	32601.97	27526.92	37677.01	24840.36	40363.57
Apr	2020	47931.60	40275.49	55587.71	36222.59	59640.61
May	2020	51233.81	42847.22	59620.40	38407.63	64060.00
Jun	2020	39472.75	32858.62	46086.87	29357.32	49588.18
Jul	2020	35212.55	29179.09	41246.01	25985.17	44439.93
Aug	2020	40809.86	33666.30	47953.43	29884.73	51735.00
Jun	2020	39472.75	32858.62	46086.87	29357.32	49588.18
Jul	2020	35212.55	29179.09	41246.01	25985.17	44439.93
Aug	2020	40809.86	33666.30	47953.43	29884.73	51735.00

Los parámetros del modelo ajustados son:

```
fit1[["model"]]
```

Smoothing

parameters:

alpha = 0.2769

beta = 1e-04

gamma = 0.3342

$$L_t = 0.2769 \frac{x_t}{S_{t-s}} + (1 - 0.2769)(L_{t-1} + b_{t-1})$$

$$b_t = 0.0001(L_t - L_{t-1}) + 0.9999b_{t-1}$$

$$S_t = 0.3344 \frac{x_t}{L_t} + (1 - 0.3344)S_{t-s}$$

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

Bibliografía:

<https://otexts.com/fpp2/>

A. Coghlan. A Little Book of R For Time Series.

Librería Forecast de R

R. J. Hyndman, Y. Khandakar. Automatic Time Series Forecasting: The forecast Package for R
Journal of Statistical Software. 2008



