Memoria de Series Temporales

Enrique Sayas Bailach y Carlos Gila Blanco

2023-11-17

Introducción

Los dos datasets utilizados han sido descargados desde Kaggle.

El utilizado para la serie temporal con tendencia es annual_gold_rate, que muestra el valor anual del oro desde 1980 hasta 2022 en Dirham de los Emiratos Árabes Unidos.

Por otra parte, el dataset utilizado para la serie temporal con tendencia y estacionalidad es HospitalityEmployees, que muestra el número de empleados en la hostelería en California desde enero de 1990 hasta diciembre de 2018.

Carga de librerías

```
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
library(readr)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
```

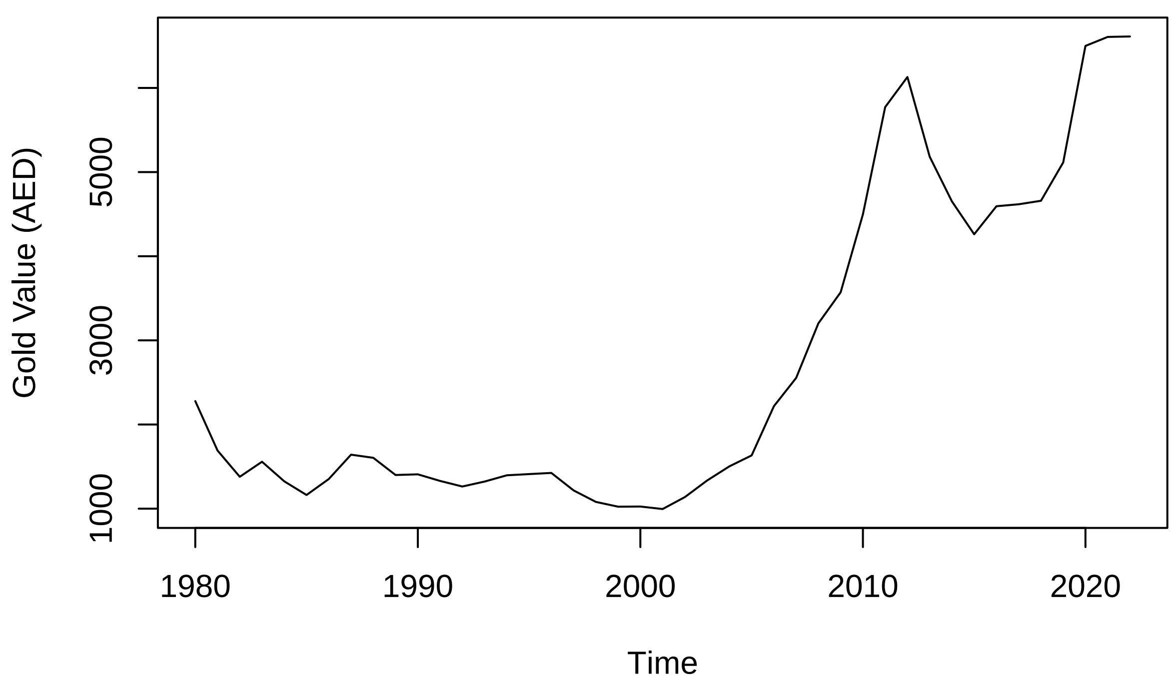
Serie temporal con tendencia

Importación de los datos

```
gold <- read_csv("Datos/annual_gold_rate.csv",
    col_types = cols(Date = col_date(format = "%Y-%m-%d")))
```

Creación de la serie temporal

```
attach(gold)
gold_ts <- ts(AED,start=c(1980),frequency=1)
plot(gold_ts, ylab = "Gold Value (AED)")
```



En base a la evolución temporal del valor del oro, se puede observar que existe una tendencia pero no una estacionalidad.

Modelo de suavizado exponencial

Por tanto, el mejor modelo de suavizado exponencial será el modelo de Holt.

Las ecuaciones de observación y actualización del modelo son las siguientes:

$$\hat{x}_t = L_{t-1} + T_{t-1}$$
$$L_t = \alpha \cdot x_t + (1-\alpha) \cdot (L_{t-1} + T_{t-1})$$
$$T_t = \beta \cdot (L_t - L_{t-1}) + (1-\beta) \cdot T_{t-1}$$

Ecuación de predicción:

$$\hat{x}_{n+k} = L_n + k \cdot T_n$$

Creación del modelo

```
gold_holt <- HoltWinters(gold_ts, gamma = FALSE)
```

Visualización de los coeficientes y los parámetros

```
gold_holt$coefficients
```

```
##         a         b 
## 6611.7100  141.3085
```

```
gold_holt$alpha
```

```
## alpha 
##     1
```

```
gold_holt$beta
```

```
##      beta 
## 0.6672362
```

Al ser $\alpha = 1$, sólo se tendrá en cuenta el valor anterior para calcular el nivel del valor siguiente.

A partir de los coeficientes obtenidos calculamos las ecuaciones de actualización:

$$L_t = x_t$$
$$T_t = 0.6672362 \cdot (L_t - L_{t-1}) + (1 - 0.6672362) \cdot T_{t-1}$$

Y la ecuación de predicción:

$$\hat{x}_{n+k} = 6611.71 + k \cdot 141.3085$$

Cálculo de la bondad del ajuste

```
fitval_gold <- fitted(gold_holt)
tail(fitval_gold, 10)
```

```
## Time Series:
## Start = 2013
## End = 2022
## Frequency = 1
##          xhat   level      trend
## 2013 6736.377 6130.18  606.19713
## 2014 4753.495 5183.46 -429.96532
## 2015 4153.513 4651.52 -498.00651
## 2016 3834.546 4260.90 -426.35435
## 2017 4674.664 4594.17   80.49448
## 2018 4659.736 4617.43   42.30557
## 2019 4701.048 4659.14   41.90818
## 2020 5433.078 5114.98  318.09846
## 2021 7529.487 6499.70 1029.78691
## 2022 7020.103 6606.30  413.80323
```

```
rmse <- sqrt(mean((gold_ts - fitval_gold[,1])^2))
rmse
```

```
## [1] 431.5184
```

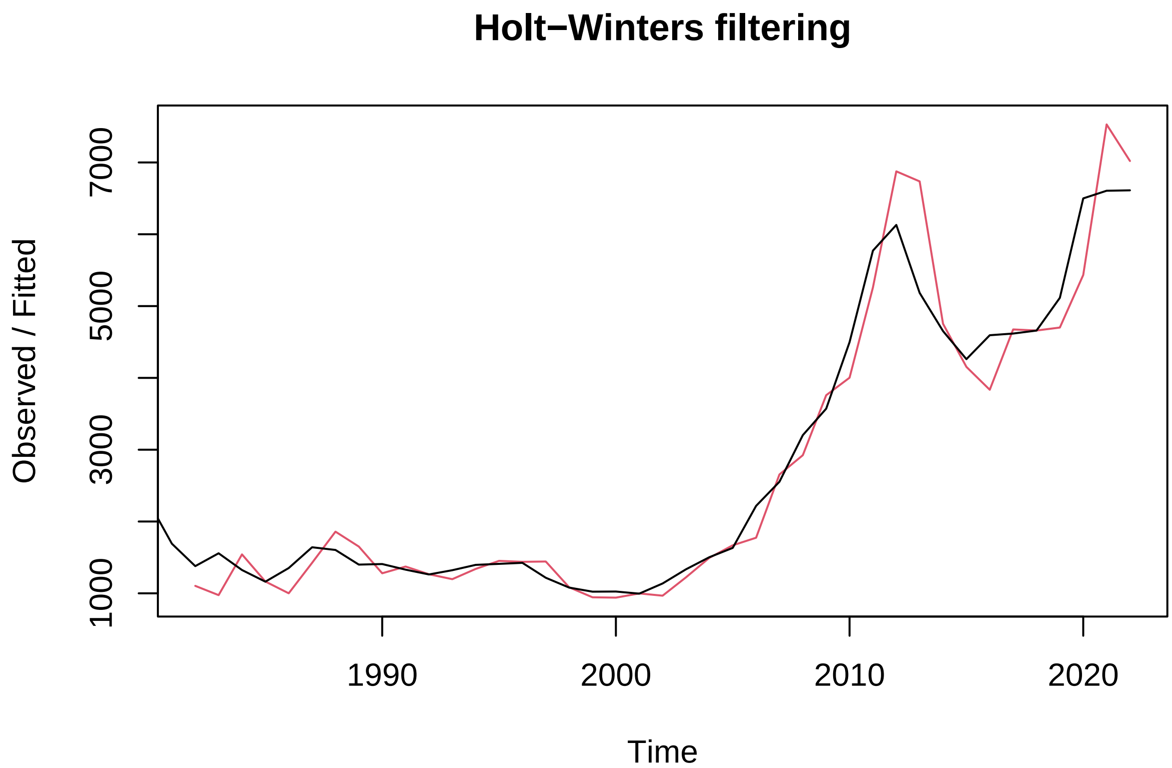
```
mape <- 100*mean(abs(gold_ts-fitval_gold[,1])/gold_ts)
mape
```

```
## [1] 9.892781
```

A partir del MAPE se puede concluir que se tiene un error medio del 9.89%.

Representación de la serie real frente a la serie ajustada

```
plot(gold_holt)
```



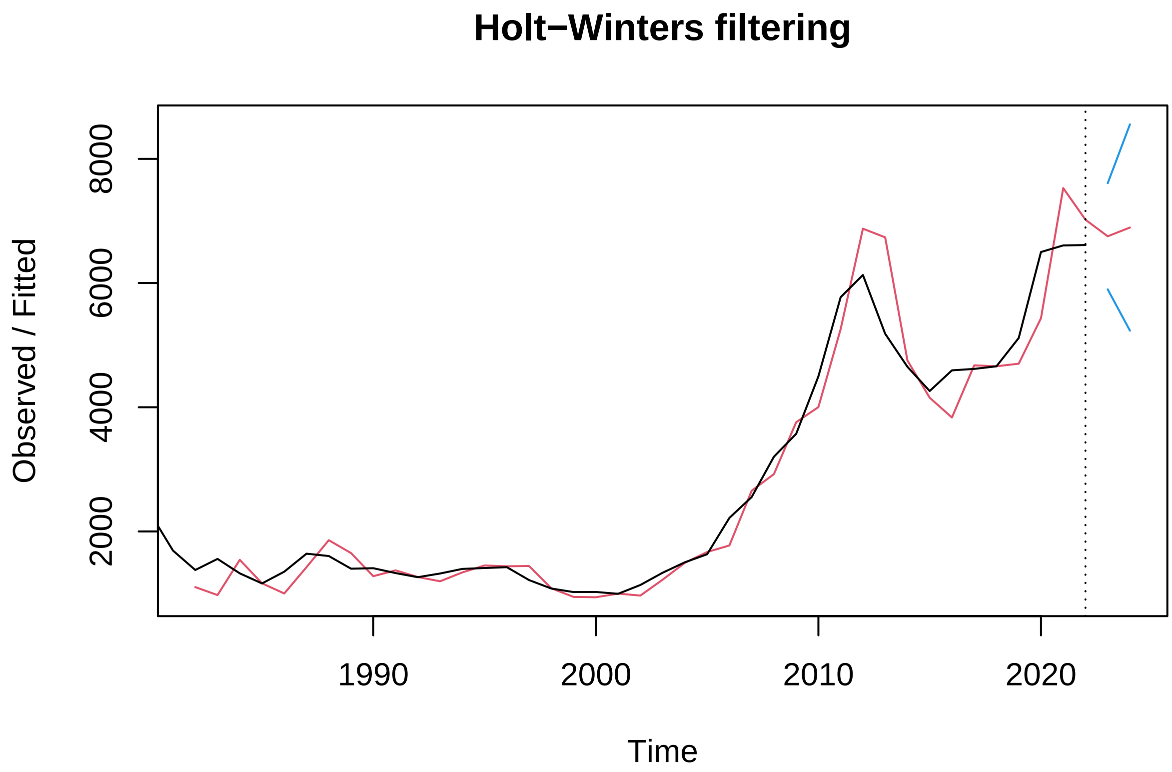
En el gráfico se puede observar el error medio del 9.89% respecto a la serie original.

Predicción para h=2

```
pred_gold <- predict(gold_holt,n.ahead=2,prediction.interval=TRUE,level=0.95)
```

Representación de la predicción junto a la serie

```
plot(gold_holt, pred_gold)
```



Modelo ARIMA

Transformación de la serie (si es necesario) a un proceso estacionario

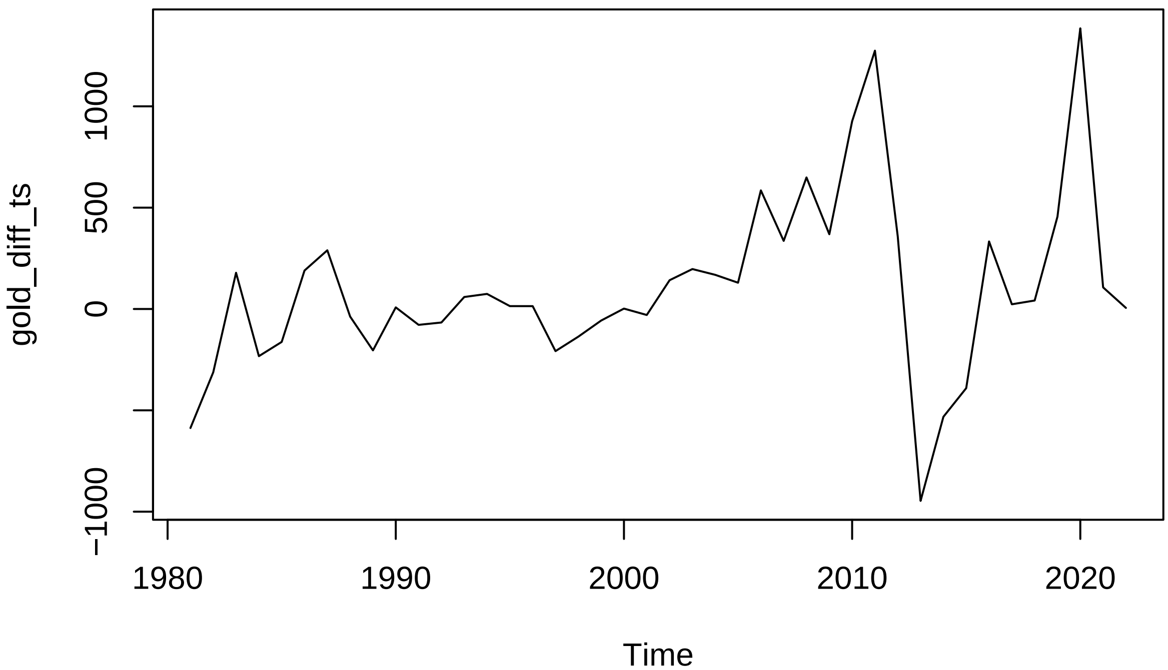
Como se ha visto previamente, la serie temporal tiene una tendencia creciente por lo que será necesario transformarla a estacionaria. Para saber el número de diferencias necesarias se hará uso de la función ndiffs.

```
ndiffs(gold_ts)
```

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

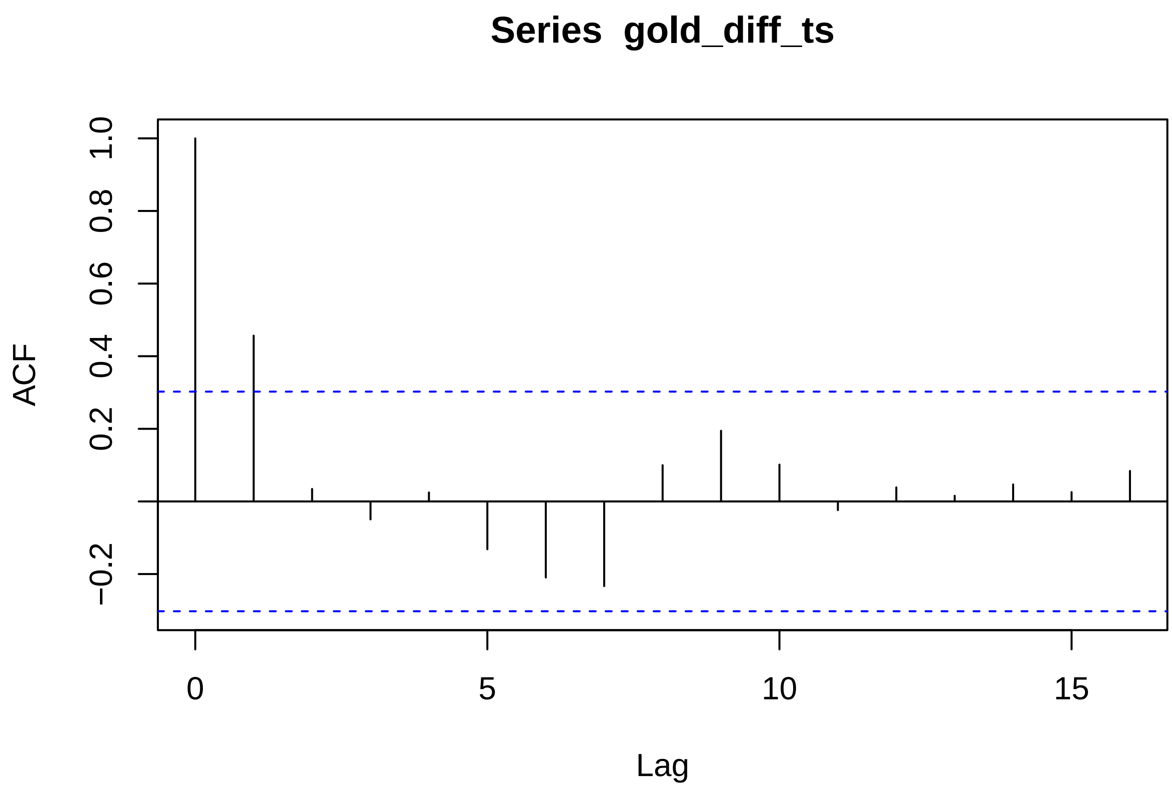
ndiffs devuelve que para que sea estacionaria se deberá realizar una diferencia.

```
gold_diff_ts <- diff(gold_ts)
plot(gold_diff_ts)
```



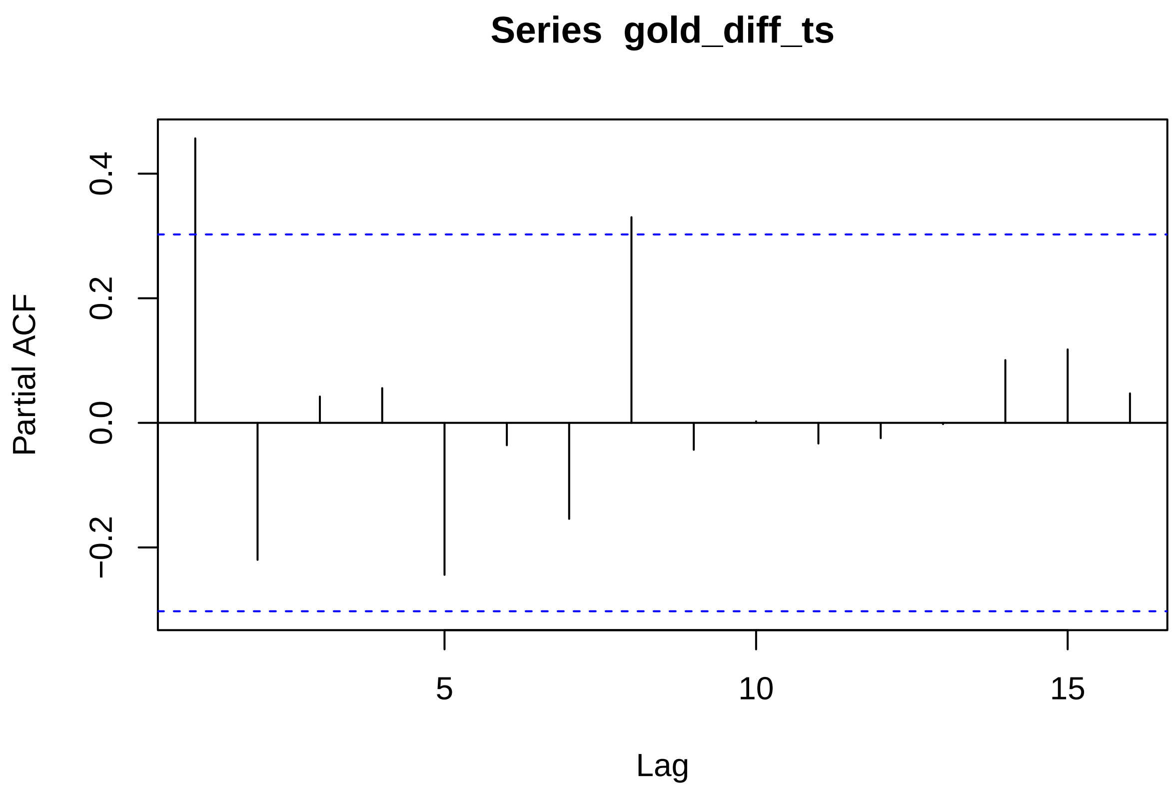
Representación gráfica e interpretación de la función de autocorrelación y de autocorrelación parcial

```
acf(gold_diff_ts)
```



A partir de la función de autocorrelación, se observa que hay únicamente un valor relevente. De este modo, se puede suponer un MA(1). Sin embargo, se puede considerar que existe un decrecimiento por lo que usaremos también un AR().

```
pacf(gold_diff_ts)
```



Se observa un decrecimiento en la función de autocorrelación parcial, por lo que se puede confirmar el MA(1). Asimismo, se puede considerar que existe únicamente un valor relevante y el resto nulos, entonces se estaría frente a un AR(1).

Encuentra el modelo ARIMA(p,d,q) que mejor describe la serie y escribe su ecuación

Inicialmente se utilizará el modelo MA(1) con una diferencia:

```
ma.fitted <- arima(gold_ts,order = c(0,1,1))
ma.fitted
```

```
##
## Call:
## arima(x = gold_ts, order = c(0, 1, 1))
##
## Coefficients:
##          ma1
##       0.4798
## s.e.  0.1059
##
## sigma^2 estimated as 144232:  log likelihood = -309.19,  aic = 622.38
```

Modelo AR(1) con una diferencia:

```r
ar.fitted <- arima(gold_ts, order = c(1,1,0))
ar.fitted
```

```
##
## Call:
## arima(x = gold_ts, order = c(1, 1, 0))
##
## Coefficients:
##          ar1
##       0.5048
## s.e.  0.1332
##
## sigma^2 estimated as 143690:  log likelihood = -309.13,  aic = 622.25
```

Comparamos los resultados con la función auto.arima

```r
auto.arima(gold_ts)
```

```
## Series: gold_ts
## ARIMA(0,1,2)
##
## Coefficients:
##          ma1     ma2
##       0.6688  0.2619
## s.e.  0.1700  0.1605
##
## sigma^2 = 141983:  log likelihood = -307.94
## AIC=621.88   AICc=622.51   BIC=627.09
```

Debido a la poca diferencia entre los modelos detectados y el devuelto por la función auto.arima, se empleará el modelo AR(1).

Cálculo de la bondad del ajuste

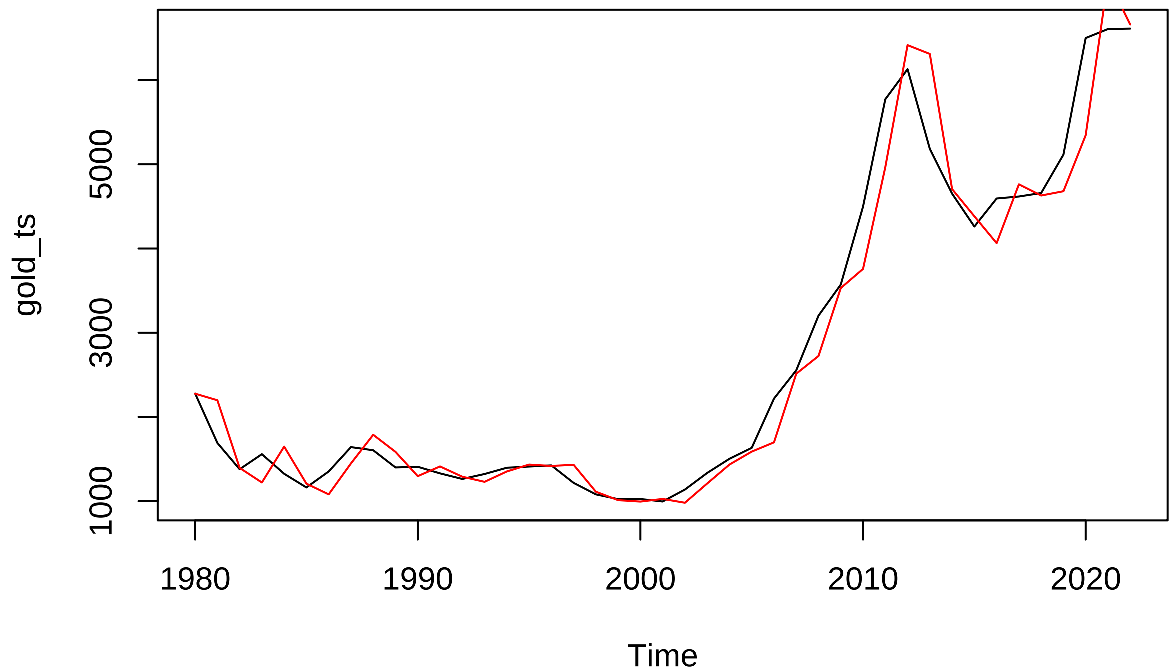
```r
accuracy(ar.fitted)
```

```
##                    ME     RMSE      MAE      MPE     MAPE      MASE       ACF1
## Training set 51.88582 374.6314 240.0059 1.299667 8.788439 0.8197346 0.06977422
```

En comparación con el modelo de suavizado exponencial, el método ARIMA ofrece un mejor ajuste.

Representación de la serie real frente a la serie ajustada

```r
plot(gold_ts)
lines(fitted(ar.fitted), col = "red")
```

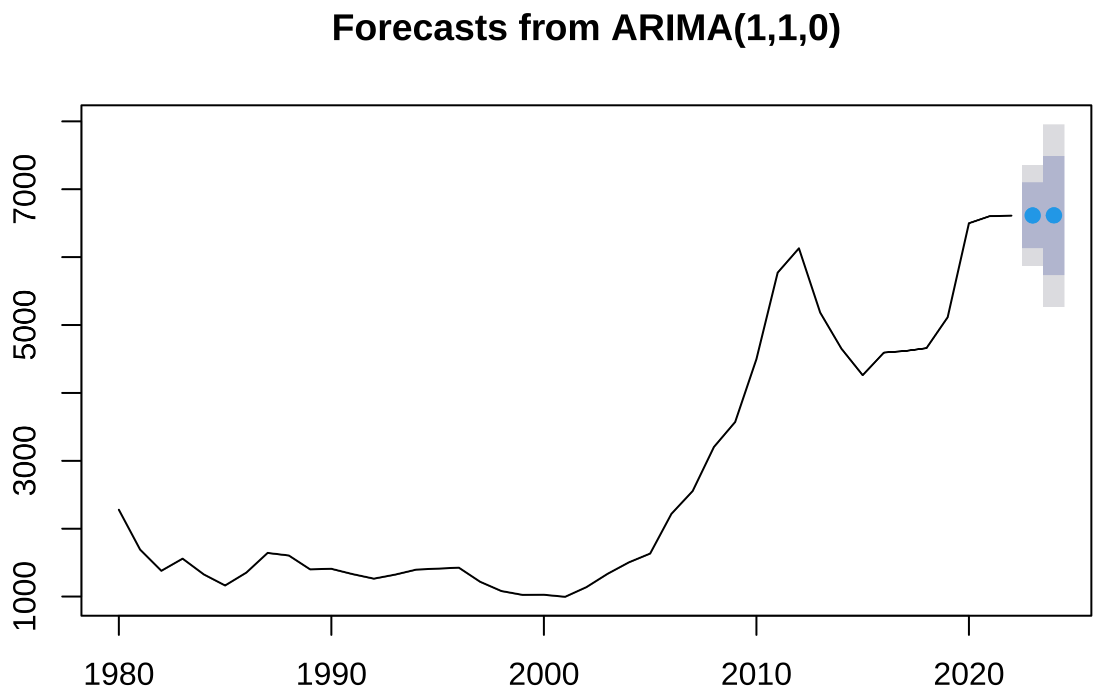


Cálculo de la predicción para h=2 instantes temporales futuros

```
ar.pred <- predict(ar.fitted,n.ahead=2,prediction.interval=TRUE,level=0.95)
```

Representación gráfica de la serie junto a la predicción obtenida

```
plot(forecast(ar.fitted,h=2))
```



Serie temporal con tendencia + estacionalidad

Importación y Adecuación de la serie

```r
HospitalityEmployees <- read_csv("Datos/HospitalityEmployees.csv")
```

```
## Rows: 348 Columns: 2
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Date
## dbl (1): Employees
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

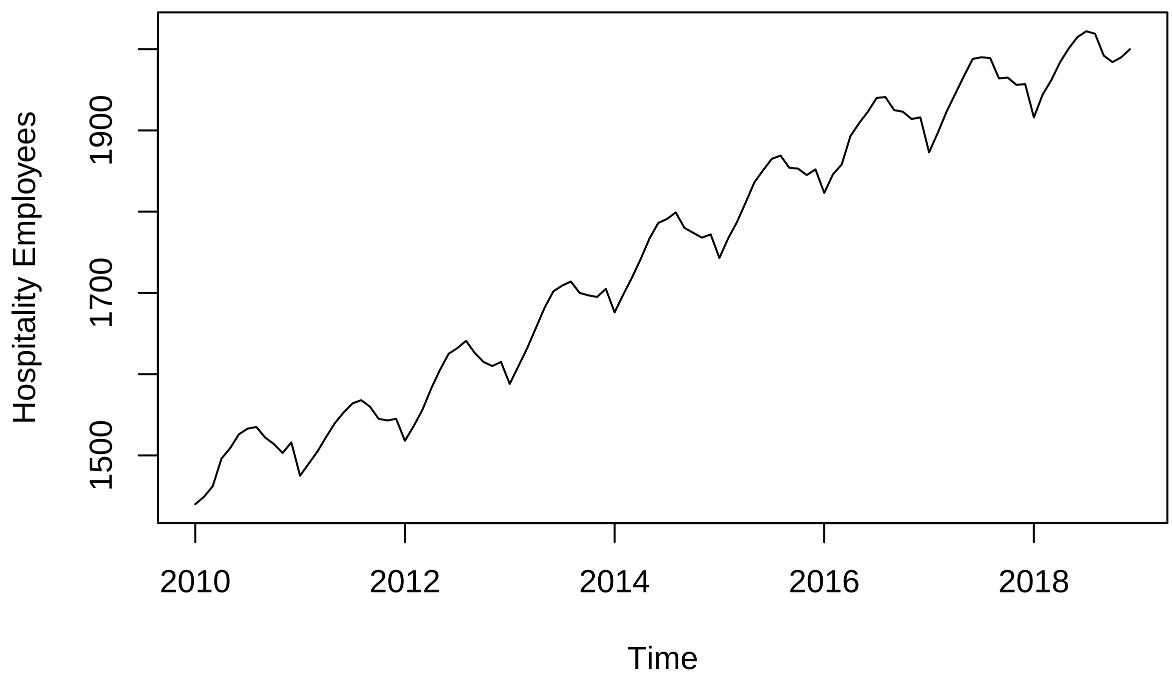
```r
HospitalityEmployees$Date <- as.Date(HospitalityEmployees$Date, format = "%m/%d/%Y")
HospitalityEmployees$Employees <- floor(HospitalityEmployees$Employees)
```

Representación de la serie temporal

```
attach(HospitalityEmployees)
```

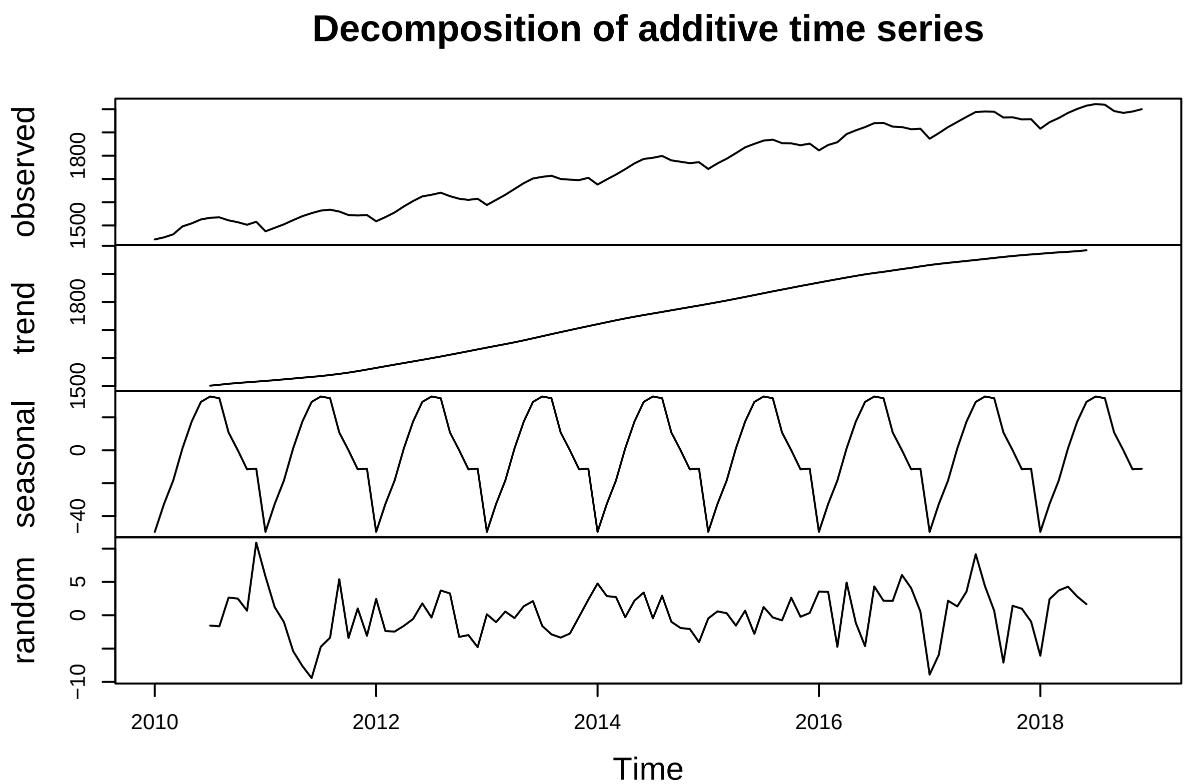
```
## The following object is masked from gold:
##
##     Date
```

```
HospitalityEmployees_ts <- ts(Employees,start=c(1990,1),end=c(2018,12),frequency=12)
HospEmp2010_ts <- window(HospitalityEmployees_ts, start = c(2010,1))
plot(HospEmp2010_ts, ylab = "Hospitality Employees")
```



Descripción de la serie

```
plot(decompose(HospEmp2010_ts, type="additive"))
```



En base a la evolución temporal del número de empleados en la hostelería, se puede observar que existe una tendencia y una estacionalidad.

Comparación de los modelos

Modelo Holt-Winters con estacionalidad aditiva

```
emp_ad <- HoltWinters(HospEmp2010_ts, seasonal = "additive")
fit_emp_ad <- fitted(emp_ad)

#RMSE
rmse_ad <- sqrt(mean((HospEmp2010_ts - fit_emp_ad[,1])^2))
rmse_ad
```

```
## [1] 6.192158
```

```
#MAPE
mape_ad <- 100*mean(abs(HospEmp2010_ts-fit_emp_ad[,1])/HospEmp2010_ts)
mape_ad
```

```
## [1] 0.2809599
```

Modelo Holt-Winters con estacionalidad multiplicativa

```r
emp_mult <- HoltWinters(HospEmp2010_ts, seasonal = "multiplicative")
fit_emp_mult <- fitted(emp_mult)

#RMSE
rmse_mult <- sqrt(mean((HospEmp2010_ts - fit_emp_mult[,1])^2))
rmse_mult
```

```
## [1] 5.668459
```

```r
#MAPE
mape_mult <- 100*mean(abs(HospEmp2010_ts-fit_emp_mult[,1])/HospEmp2010_ts)
mape_mult
```

```
## [1] 0.2562788
```

Comparación entre los modelos Holt-Winters aditivo y multiplicativo en base a los errores RMSE y MAPE

Holt-Winters	RMSE	MAPE
Additive	6.192158	0.2809599
Multiplicative	5.668459	0.2562788

El mejor modelo de suavizado exponencial será el modelo de Host-Winters con estacionalidad aditiva pues la diferencia en el RMSE entre ambos modelos es muy pequeña, siendo más fácil la implementación del modelo con estacionalidad aditiva.

Las ecuaciones de observación y actualización del modelo son las siguientes:

$$\hat{x}_t = L_{t-1} + T_{t-1} + S_{t-c}$$

$$L_t = \alpha \cdot (x_t - S_{t-c}) + (1-\alpha) \cdot (L_{t-1} + T_{t-1})$$
$$T_t = \beta \cdot (L_t - L_{t-1}) + (1-\beta) \cdot T_{t-1}$$

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

Y la ecuación de predicción:

$$\hat{x}_{n+k} = L_n + k \cdot T_n + S_{n+k-c}$$

Creación del modelo

```r
emp_fit <- HoltWinters(HospEmp2010_ts, seasonal = "additive")
```

Visualización de los coeficientes y los parámetros

```r
emp_fit$coefficients
```

```
##           a           b          s1          s2          s3          s4 
## 2009.266255    4.174587  -50.815982  -25.284286   -9.900944    8.480798 
##          s5          s6          s7          s8          s9         s10 
##   21.666781   31.274928   32.127005   26.301510    1.009574   -3.544818 
##         s11         s12 
##   -8.699546   -9.266255
```

```r
emp_fit$alpha
```

```
##     alpha 
## 0.6583277
```

```r
emp_fit$beta
```

```
##       beta 
## 0.05505944
```

```r
emp_fit$gamma
```

```
## gamma 
##     1
```

Al ser $\beta = 0.055$ la tendencia es constante y como $\gamma = 1$, el efecto estacional varía de año en año, y por tanto su actualización depende sólo del efecto estacionado en dicho instante, sin tener en cuenta el efecto estacional del año anterior.

A partir de los coeficientes obtenidos calculamos las ecuaciones de actualización:

$$L_t = 0.6583277 \cdot (x_t - S_{t-12}) + (1 - 0.6583277) \cdot (L_{t-1} + T_{t-1})$$
$$T_t = 0.05505944 \cdot (L_t - L_{t-1}) + (1 - 0.05505944) \cdot T_{t-1}$$
$$S_t = x_t - L_t$$

Y la ecuación de predicción

$$\hat{x}_{n+k} = 2009.266255 + k \cdot 4.174587 + S_{n+k-12}$$

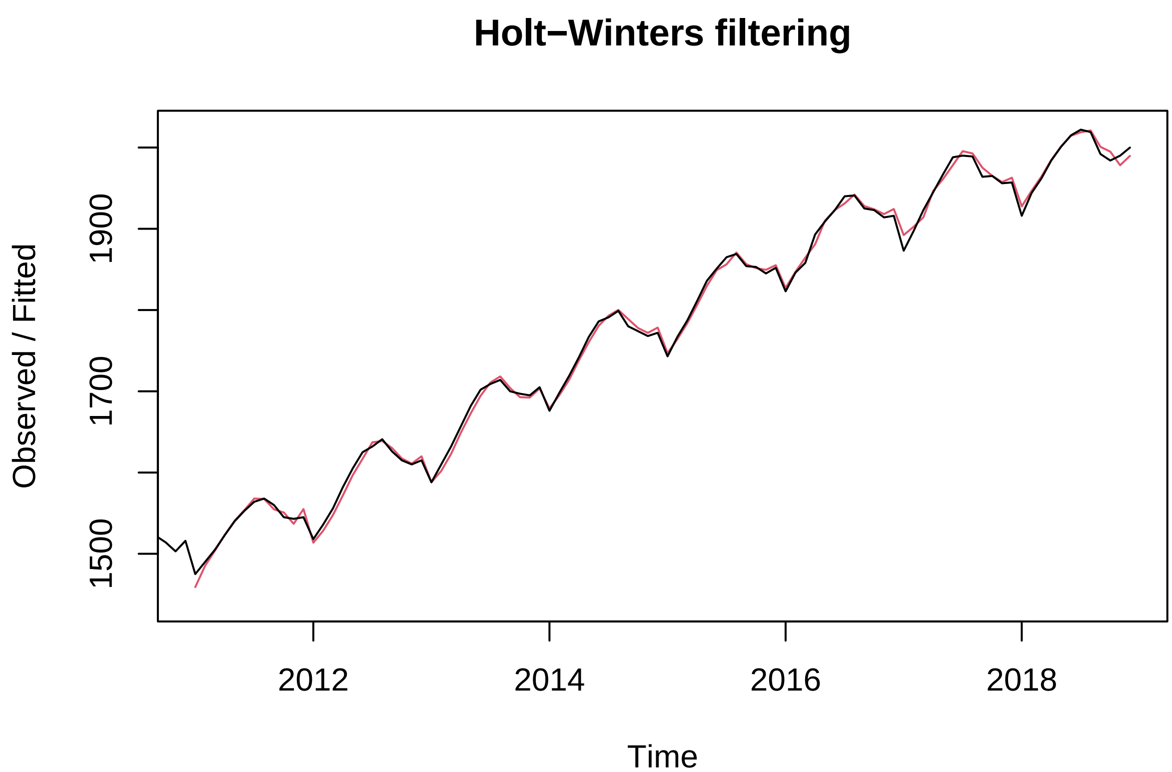
Cálculo de la bondad del ajuste

```r
fitval_emp <- fitted(emp_fit)
tail(fitval_emp, 10)
```

```
##                 xhat    level    trend      season
## Mar 2018 1964.329 1969.284 4.149945  -9.1051677
## Apr 2018 1984.679 1971.901 4.065523   8.7129286
## May 2018 2001.345 1975.519 4.040897  21.7845324
## Jun 2018 2014.448 1979.333 4.028405  31.0862977
## Jul 2018 2018.811 1983.725 4.048416  31.0373608
## Aug 2018 2021.033 1989.873 4.164014  26.9962024
## Sep 2018 2000.808 1992.698 4.090316   4.0189341
## Oct 2018 1994.962 1990.990 3.771060   0.2006361
## Nov 2018 1978.181 1987.545 3.373715 -12.7378949
## Dec 2018 1989.725 1998.700 3.802133 -12.7770725
```

Representación de la serie real frente a la serie ajustada

```
plot(emp_fit)
```



Predicción para h=c

```
pred_hosp <- predict(emp_fit,12)
```

Representación de la predicción junto a la serie

```
pred <- predict(emp_fit,n.ahead=12,prediction.interval=TRUE,level=0.95)

plot(emp_fit, pred)
```