

Procesos Estocásticos y Series Temporales

Práctica 3: Métodos de Alisado Exponencial

Francisco Javier Mercader Martínez

PRÁCTICA 3 PARA COMPLETAR

PROCESOS ESTOCÁSTICOS Y SERIES TEMPORALES

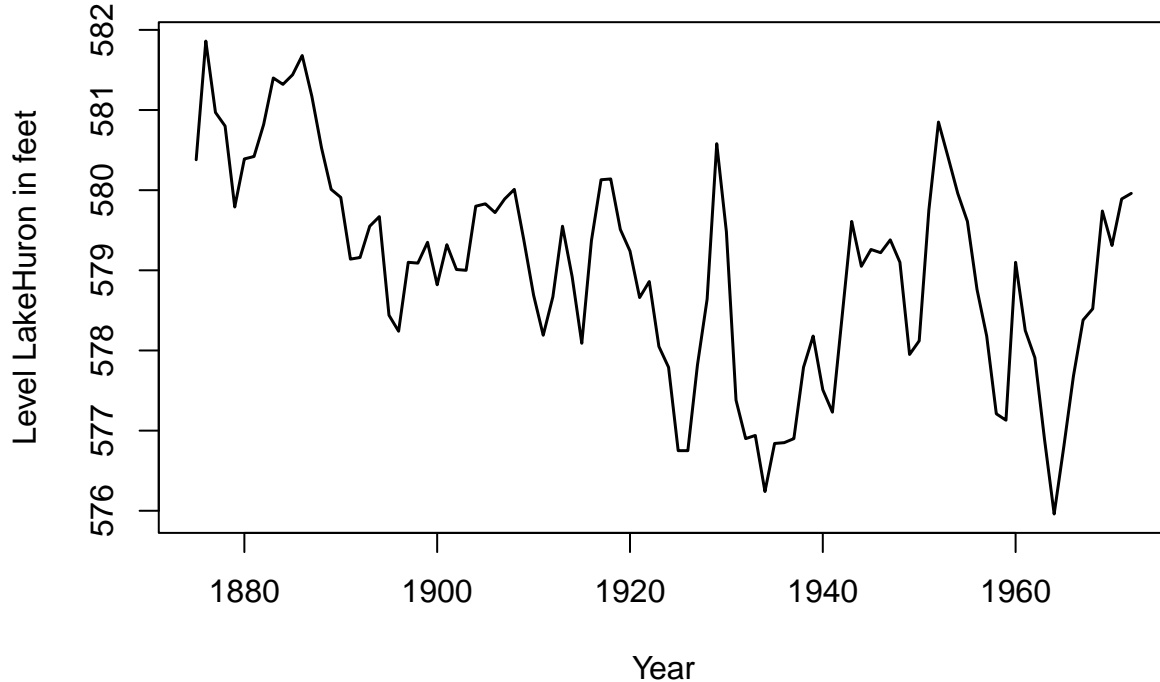
GRADO EN CIENCIA E INGENIERÍA DE DATOS

Ejemplos de los métodos de alisado exponencial con R

1. Alisado exponencial simple

Consideremos la serie de datos “LakeHuron” de R:

```
ts.plot(LakeHuron,  
        gpars = list(xlab = "Year",  
                      ylab = "Level LakeHuron in feet",  
                      lwd = 1.5))
```



Método de alisado exponencial simple:

```
ejemplo_aes <- HoltWinters(LakeHuron, beta = FALSE, gamma = FALSE)
```

```
ejemplo_aes
```

```
## Holt-Winters exponential smoothing without trend and without seasonal component.
##
## Call:
## HoltWinters(x = LakeHuron, beta = FALSE, gamma = FALSE)
##
## Smoothing parameters:
##  alpha: 0.9999339
##  beta : FALSE
##  gamma: FALSE
##
## Coefficients:
##      [,1]
## a 579.96
```

Suma de cuadrados residual para el alpha óptimo:

```
ejemplo_aes$SSE
```

```
## [1] 53.86594
```

Predicciones para los siguientes 5 años:

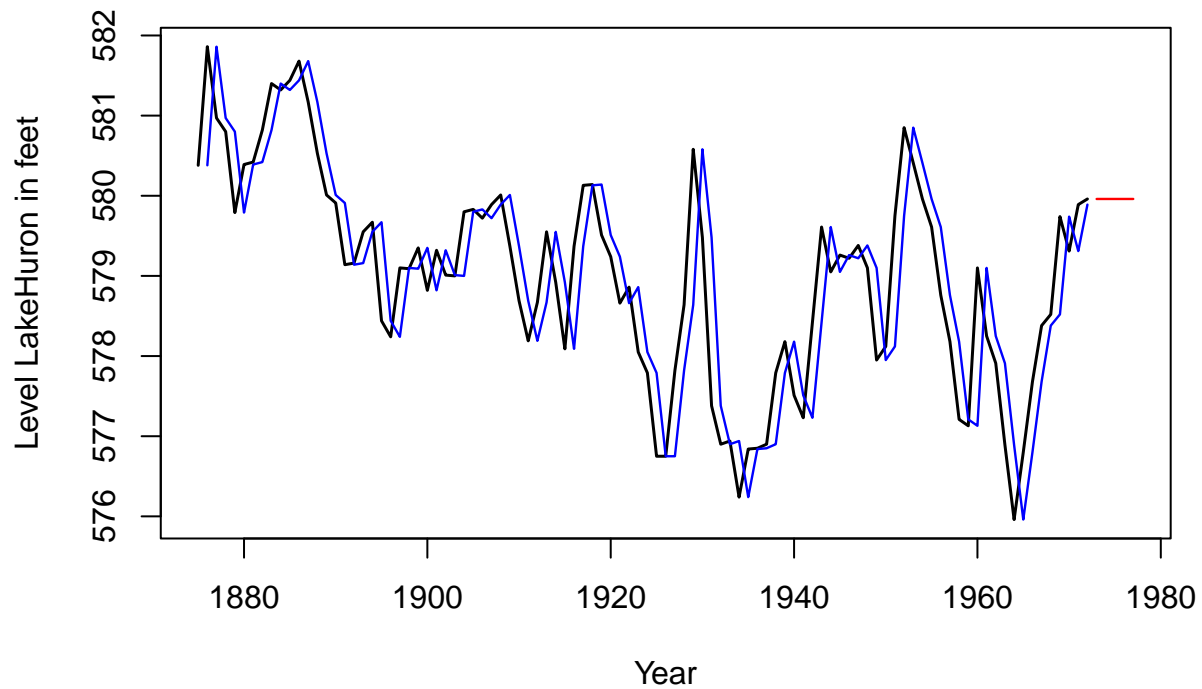
```
predict(ejemplo_aes, n.ahead = 5)
```

```
## Time Series:
## Start = 1973
## End = 1977
## Frequency = 1
##      fit
## [1,] 579.96
## [2,] 579.96
## [3,] 579.96
## [4,] 579.96
## [5,] 579.96
```

Representación gráfica de valores reales y predicciones:

```
ts.plot(LakeHuron, xlim = c(1875, 1977),
        gpars = list(xlab = "Year",
                     ylab = "Level LakeHuron in feet",
                     lwd = 1.5))

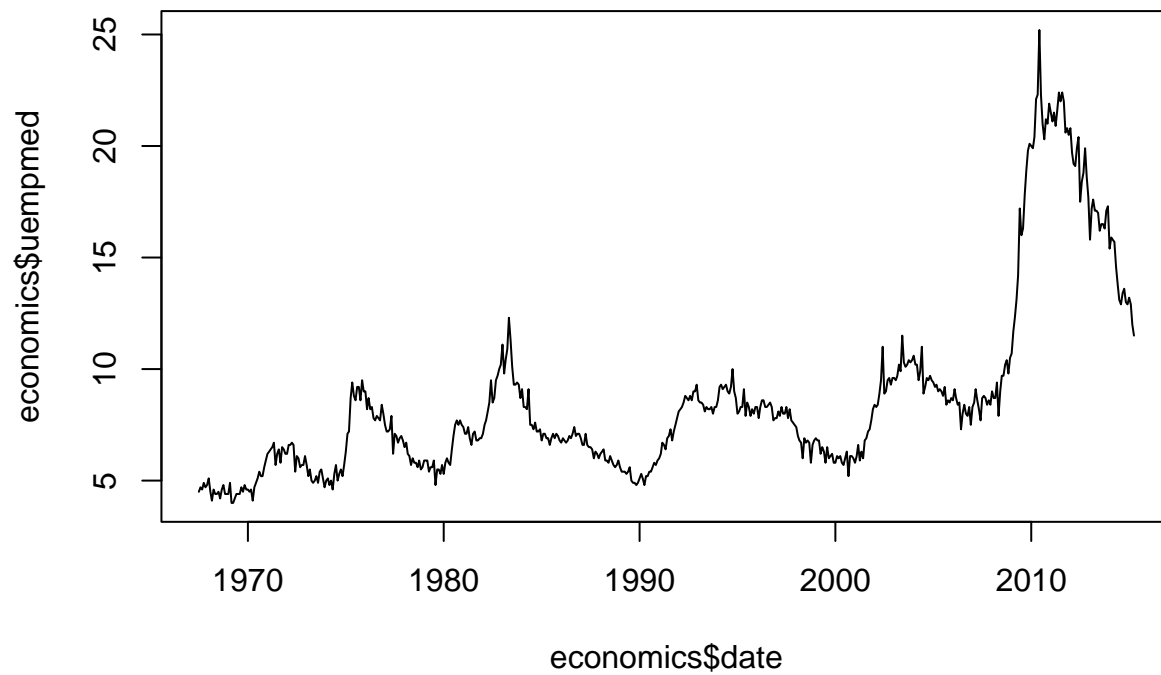
lines(ejemplo_aes$fitted[ , 1], col = "blue", lwd = 1.2)
lines(predict(ejemplo_aes, n.ahead = 5), col = "red", lwd = 1.2)
```



2. Método de Holt

Consideraremos la serie:

```
library(ggplot2)
# ts.plot(economics$uempmed)
plot(economics$date, economics$uempmed, type = "l")
```



Método de Holt:

```
ejemplo_Holt <- HoltWinters(economics$uempmed, gamma = FALSE)
ejemplo_Holt
```

```
## Holt-Winters exponential smoothing with trend and without seasonal component.
##
## Call:
## HoltWinters(x = economics$uempmed, gamma = FALSE)
##
## Smoothing parameters:
##  alpha: 0.7050079
##  beta : 0.0925097
##  gamma: FALSE
##
## Coefficients:
##           [,1]
## a 11.6412203
## b -0.2600279
```

Suma de cuadrados residual para los parámetros de alisado óptimos:

```
ejemplo_Holt$SSE
```

```
## [1] 174.6968
```

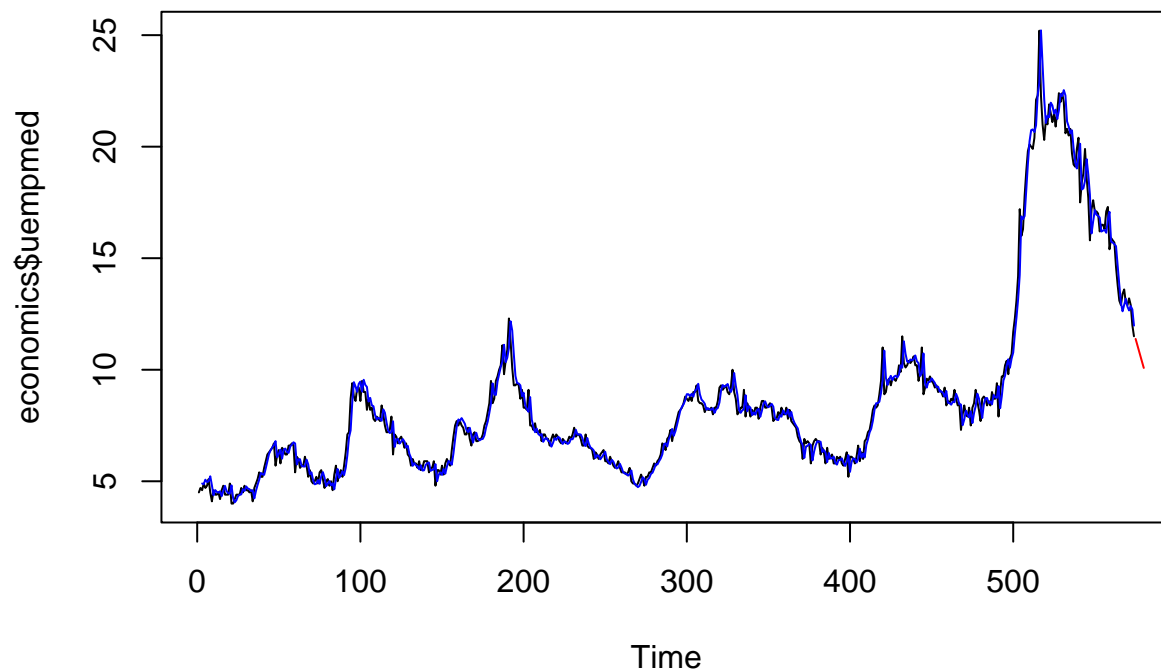
Predicciones para los siguientes 6 meses.

```
predict(ejemplo_Holt, n.ahead = 6)
```

```
## Time Series:
## Start = 575
## End = 580
## Frequency = 1
##           fit
## [1,] 11.38119
## [2,] 11.12116
## [3,] 10.86114
## [4,] 10.60111
## [5,] 10.34108
## [6,] 10.08105
```

Gráfico de serie original y predicciones:

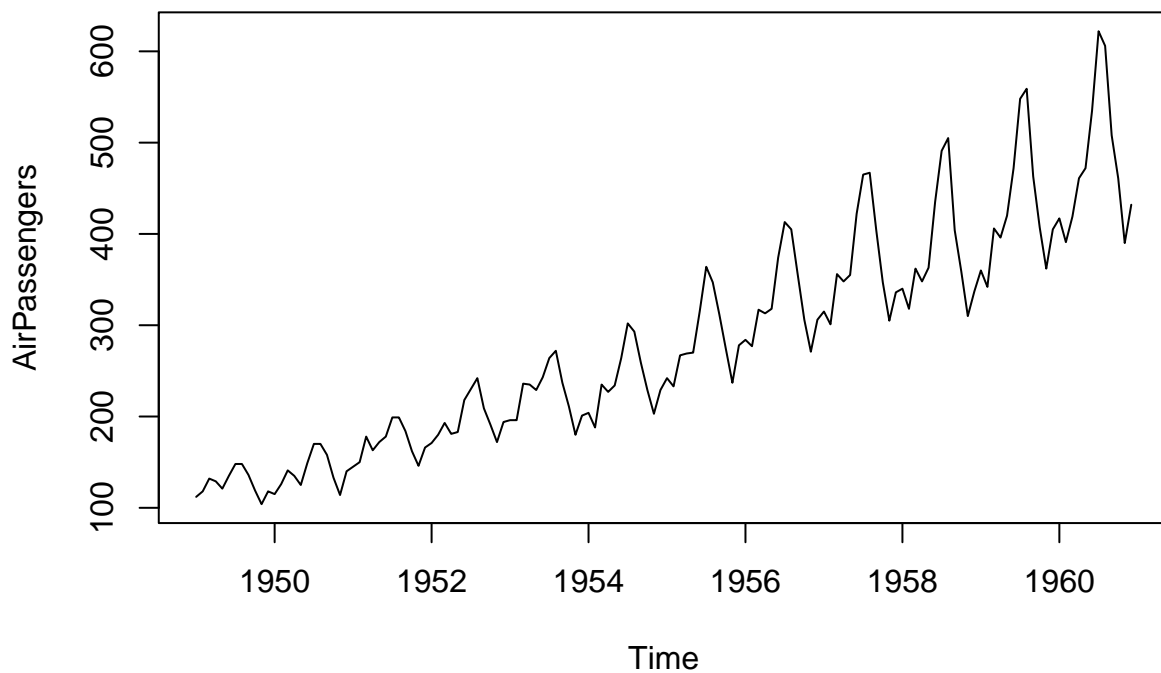
```
ts.plot(economics$uempmed)
lines(ejemplo_Holt$fitted[ , 1], col = "blue")
lines(predict(ejemplo_Holt, n.ahead = 6), col = "red")
```



3. Método de Holt-Winters multiplicativo

Serie temporal “AirPassengers” incluida en R:

```
ts.plot(AirPassengers)
```



Método de Holt-Winters multiplicativo:

```
class(AirPassengers) #comprobamos que sí es de clase ts
```

```
## [1] "ts"
```

```
ejemplo_HW_mult <- HoltWinters(AirPassengers, seasonal = "multiplicative")
ejemplo_HW_mult
```

```
## Holt-Winters exponential smoothing with trend and multiplicative seasonal component.
##
## Call:
## HoltWinters(x = AirPassengers, seasonal = "multiplicative")
##
## Smoothing parameters:
##  alpha: 0.2755925
##  beta : 0.03269295
##  gamma: 0.8707292
##
## Coefficients:
##           [,1]
## a  469.3232206
## b    3.0215391
## s1   0.9464611
## s2   0.8829239
## s3   0.9717369
## s4   1.0304825
## s5   1.0476884
## s6   1.1805272
## s7   1.3590778
## s8   1.3331706
## s9   1.1083381
## s10  0.9868813
## s11  0.8361333
## s12  0.9209877
```

Suma de cuadrados residual para los parámetros de alisado óptimos:

```
ejemplo_HW_mult$SSE
```

```
## [1] 16570.78
```

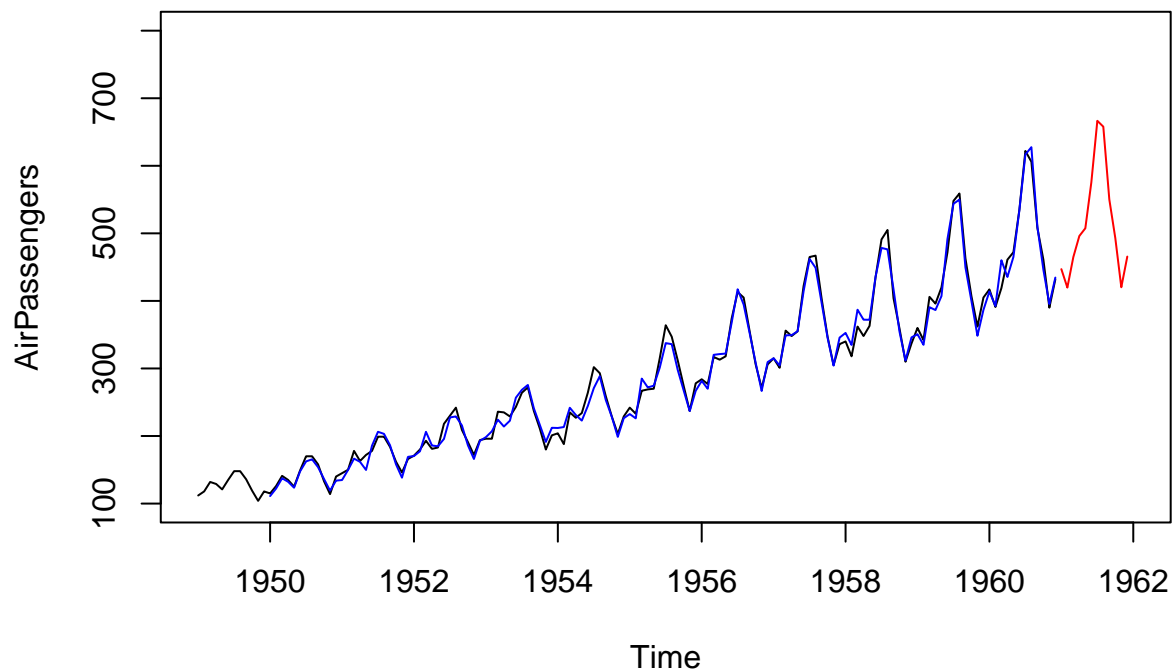
Predicciones para los siguientes 12 meses:

```
predict(ejemplo_HW_mult, n.ahead = 12)
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 1961 447.0559 419.7123 464.8671 496.0839 507.5326 575.4509 666.5923 657.9137
##           Sep      Oct      Nov      Dec
## 1961 550.3088 492.9853 420.2073 465.6345
```

Gráfico de serie original y predicciones:

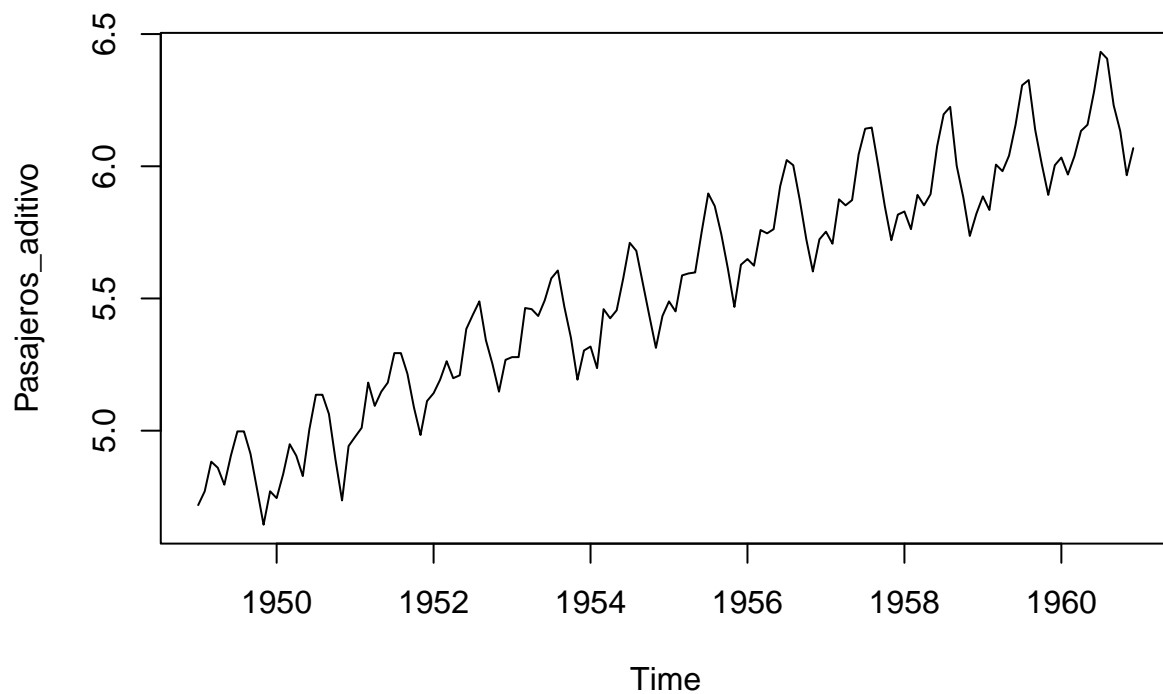
```
ts.plot(AirPassengers, xlim = c(1949, 1962), ylim = c(100, 800))
lines(ejemplo_HW_mult$fitted[, 1], col = "blue")
lines(predict(ejemplo_HW_mult, n.ahead = 12), col = "red")
```



4. Método de Holt-Winters aditivo

Serie “AirPassengers”, pero transformados tomando logaritmos neperianos.

```
Pasajeros_aditivo <- log(AirPassengers)
ts.plot(Pasajeros_aditivo)
```



Método Holt-Winters aditivo:

```
ejemplo_HW_aditivo <- HoltWinters(Pasajeros_aditivo, seasonal = 'additive')
ejemplo_HW_aditivo
```

```
## Holt-Winters exponential smoothing with trend and additive seasonal component.
##
## Call:
## HoltWinters(x = Pasajeros_aditivo, seasonal = "additive")
##
## Smoothing parameters:
##   alpha: 0.3266015
##   beta : 0.005744138
##   gamma: 0.8206654
##
## Coefficients:
##           [,1]
## a      6.172308435
## b      0.008981893
## s1 -0.073201087
## s2 -0.140973564
## s3 -0.036703294
## s4  0.014522733
## s5  0.032554237
## s6  0.154873570
## s7  0.294317062
## s8  0.276063997
## s9  0.088237657
## s10 -0.032657089
## s11 -0.198012716
## s12 -0.102863837
```

Y las predicciones del próximo año serán:

```
predict(ejemplo_HW_aditivo, n.ahead = 12)
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 1961 6.108089 6.049299 6.162551 6.222759 6.249772 6.381073 6.529499 6.520228
##           Sep      Oct      Nov      Dec
## 1961 6.341383 6.229470 6.073097 6.177227
```

Modelos ETS (Error, Trend, Seasonal)

Análisis de la serie “Pasajeros_aditivo” que vimos en una sección anterior.

Usaremos el modelo ETS(A,A,A):

```
library(forecast)
```

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

ejemplo_ETS_AAA <- ets(Pasajeros_aditivo, model = "AAA")
ejemplo_ETS_AAA

## ETS(A,A,A)
##
## Call:
## ets(y = Pasajeros_aditivo, model = "AAA")
##
## Smoothing parameters:
```



```
##      alpha = 0.6975
##      beta  = 0.0031
##      gamma = 1e-04
##
## Initial states:
##      l = 4.7925
##      b = 0.0111
##      s = -0.1045 -0.2206 -0.0787 0.0562 0.2049 0.2149
##           0.1146 -0.0081 -0.0059 0.0225 -0.1113 -0.0841
##
##      sigma: 0.0383
##
##      AIC      AICc      BIC
## -207.1694 -202.3123 -156.6826
```

Intervalos de predicción para los próximos 12 meses con el modelo ETS:

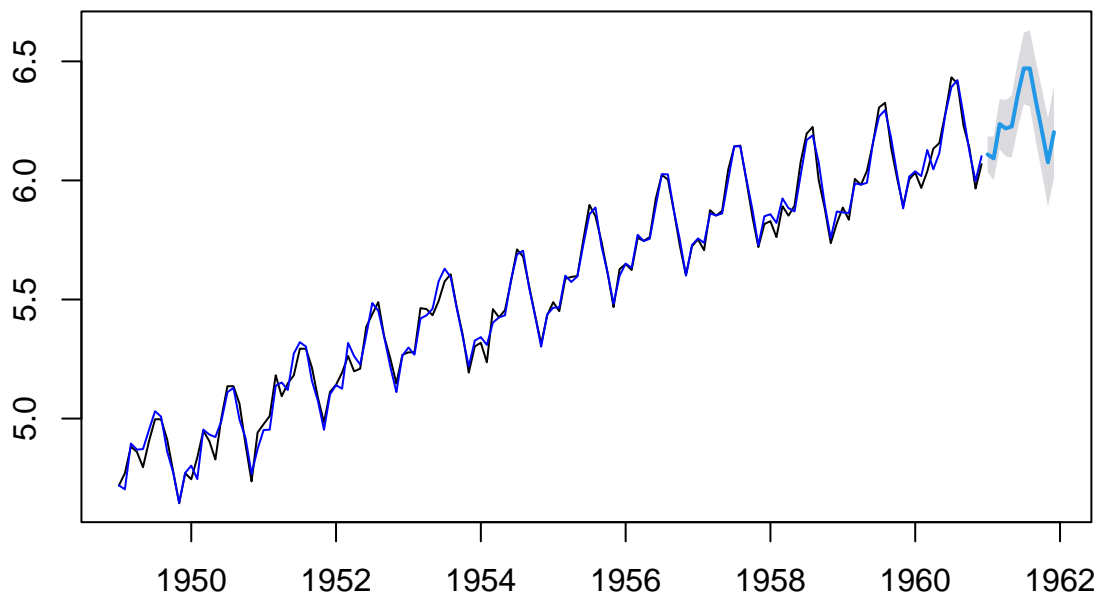
```
forecast(ejemplo_ETS_AAA, h = 12, level = 0.95)
```

```
##      Point Forecast    Lo 95    Hi 95
## Jan 1961      6.109335 6.034351 6.184319
## Feb 1961      6.092542 6.000989 6.184094
## Mar 1961      6.236626 6.130960 6.342292
## Apr 1961      6.218531 6.100323 6.336738
## May 1961      6.226734 6.097100 6.356369
## Jun 1961      6.359734 6.219514 6.499954
## Jul 1961      6.470379 6.320237 6.620521
## Aug 1961      6.470714 6.311189 6.630240
## Sep 1961      6.332402 6.163942 6.500863
## Oct 1961      6.207747 6.030732 6.384762
## Nov 1961      6.076240 5.890997 6.261482
## Dec 1961      6.202710 6.009525 6.395894
```

Representamos la serie original y predicciones:

```
plot(forecast(ejemplo_ETS_AAA, h = 12, level = 0.95))
lines(ejemplo_ETS_AAA$fitted, col = "blue")
```

Forecasts from ETS(A,A,A)



Veamos para la serie “Pasajeros_aditivo” la búsqueda automática del modelo:

```
busqueda_ETS <- ets(Pasajeros_aditivo)
busqueda_ETS
```

```
## ETS(M,A,M)
##
## Call:
## ets(y = Pasajeros_aditivo)
##
## Smoothing parameters:
##   alpha = 0.5871
##   beta  = 1e-04
##   gamma = 1e-04
##
## Initial states:
##   l = 4.8055
##   b = 0.0091
##   s = 0.9814 0.96 0.9866 1.0116 1.0376 1.0379
##       1.02 0.9985 0.9986 1.004 0.9804 0.9833
##
## sigma: 0.0069
##
##      AIC      AICc      BIC
## -208.3403 -203.4832 -157.8535
```

Comparación de errores:

```
sum((Pasajeros_aditivo - busqueda_ETS$fitted)^2)
```

```
## [1] 0.178084
```

```
sum(ejemplo_ETS_AAA$residuals^2)
```

```
## [1] 0.1873479
```