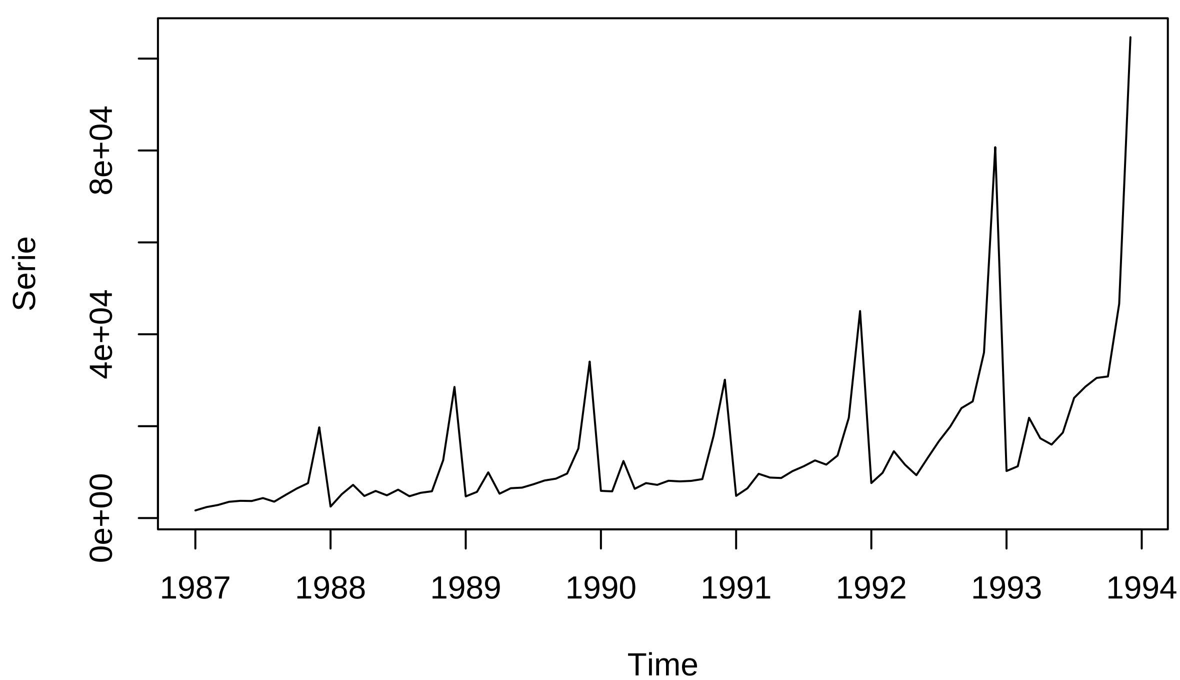
Tarea 1

Cuéllar, E. Tapia, J. Maciel, J. Saldaña, R. Miranda, G

15/Oct/2021

1.- Grafique los datos. Describa lo que observe de su información (varianza contante o no constante, tendencia, ciclos estacionales, periodicidad de los ciclos).

Gráfica de la serie de tiempo



Varianza

Podemos observar una varianza creciente conforme pasa el tiempo, teniendo primero un ligero aumento de 1987 a 1990 de manera lineal, después parece que se mantene constante de 1990 a 1991 para posteriormente crecer demasiado de 1991 a 1994 Es decir, no es constante la varianza. Lo que hace que se dispare es la temporada alta vacacional (Al parecer, en noviembre y diciembre, ya que analizamos una playa en Australia, donde el verano comienza en diciembre). Podemos comprobarlo con un bp test:

```r
# Ho: (Homocedasticidad)  vs  H1: La varianza no es constate (Heterocedasticidad)
t1 = seq(1987+0/12, 1993+11/12, by = 1 / 12)
bptest(Serie ~t1)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  Serie ~ t1
## BP = 5.9528, df = 1, p-value = 0.01469
```

Inclusive podemos ver de una vez que no es estacionaria tanto con el test de Dickey-Fuller como con el de Phillips:

```r
# Ho: La serie no es estacionaria vs. H1: La serie es estacionaria
adf.test(Serie)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  Serie
## Dickey-Fuller = -2.0809, Lag order = 4, p-value = 0.5427
## alternative hypothesis: stationary
```

```r
#Por lo tanto:

# Ho: La serie es estacionaria vs. H1: La serie no es estacionaria
kpss.test(Serie)
```

```
## Warning in kpss.test(Serie): p-value smaller than printed p-value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  Serie
## KPSS Level = 1.3089, Truncation lag parameter = 3, p-value = 0.01
```

Tendencia

La tendencia parece ser en general creciente, repitiendo casi el mismo patrón que la varianza: Es creciente de manera ligera (lineal, con pendiente pequeña) de 1987 a 1990, para después decrecer un poco de 1990 a 1991, sin embargo crece de manera cuadrática, al parecer, de 1991 a 1994.

Ciclos estacionales

Dado que estamos analizando una base de datos de ventas mensuales de una tienda de souvenirs en una playa en Asutralia, hace todo el sentido del mundo que tenga un ciclo ya que tanto en las ventas como visitas a sitios vacacionales hay una fuerte dependencia en los meses del año. Esto lo confirmamos con la gráfica, donde se observa un ciclo estacional bastante claro.

Periodicidad de los ciclos

Complementando el comentario del punto anterior, en la gráfica observamos que el ciclo es anual. De enero a febrero (o los primeros meses del año)parece crecer ligeramente, depués baja un poco para crecer de manera ligera nuevamente, pero al llegar lo que parece ser noviembre y diciembre (o los últimos meses del año)crece exageradamente; posteriormente baja de diciembre a enero y se repite el ciclo. Esto se confirma en la pregunta 3 con más detalle

2.-Si la base presenta datos faltantes NA. Use algún método de imputación de la paquetería imputeTS.

No hay ningún NA, como podemos observar

```
## [1] 0
```

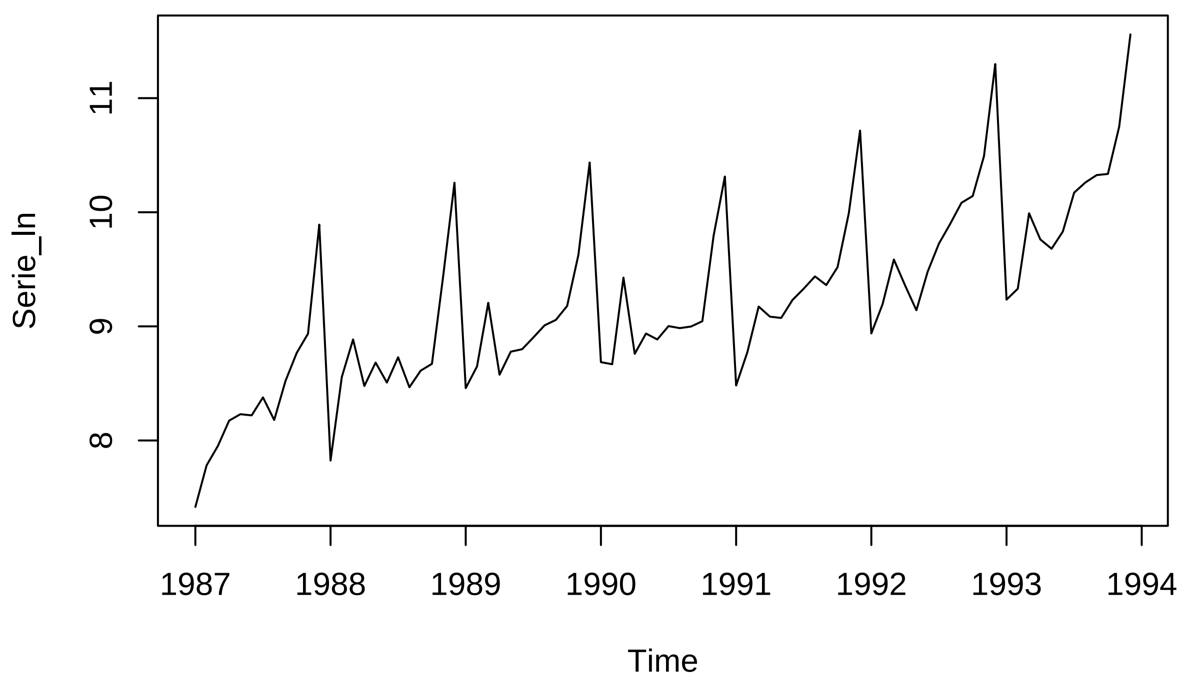
Por lo que no es necesario aplicar ningún método de imputación

3.- Use distintos métodos de descomposición de las series para obtener sus componentes (tendencia y ciclos estacionales), en específico use los siguientes:

(a) Ajuste de curvas (modelos deterministas o de regresión). Realice un pronóstico de 3 años futuros.

Primero realizareos una transformación con el logarítmo para estabilizar la varianza

```
Serie_ln<-(log(Serie))
ts.plot(Serie_ln)
```



Al parecer, la varianza ya se estabilizó considerablemente (De manera gráfica). ¡Comprobémoslo con un bp test!

```
bptest(Serie_ln~t1)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  Serie_ln ~ t1
## BP = 0.3213, df = 1, p-value = 0.5708
```

```
# Ho: La serie no es estacionaria vs. H1: La serie es estacionaria
adf.test(diff(Serie_ln))
```

```
## Warning in adf.test(diff(Serie_ln)): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff(Serie_ln)
## Dickey-Fuller = -4.935, Lag order = 4, p-value = 0.01
## alternative hypothesis: stationary
```

```r
#Por lo tanto:

# Ho: La serie es estacionaria vs. H1: La serie no es estacionaria
kpss.test(diff(Serie_ln))
```

```
## Warning in kpss.test(diff(Serie_ln)): p-value greater than printed p-value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  diff(Serie_ln)
## KPSS Level = 0.062948, Truncation lag parameter = 3, p-value = 0.1
```

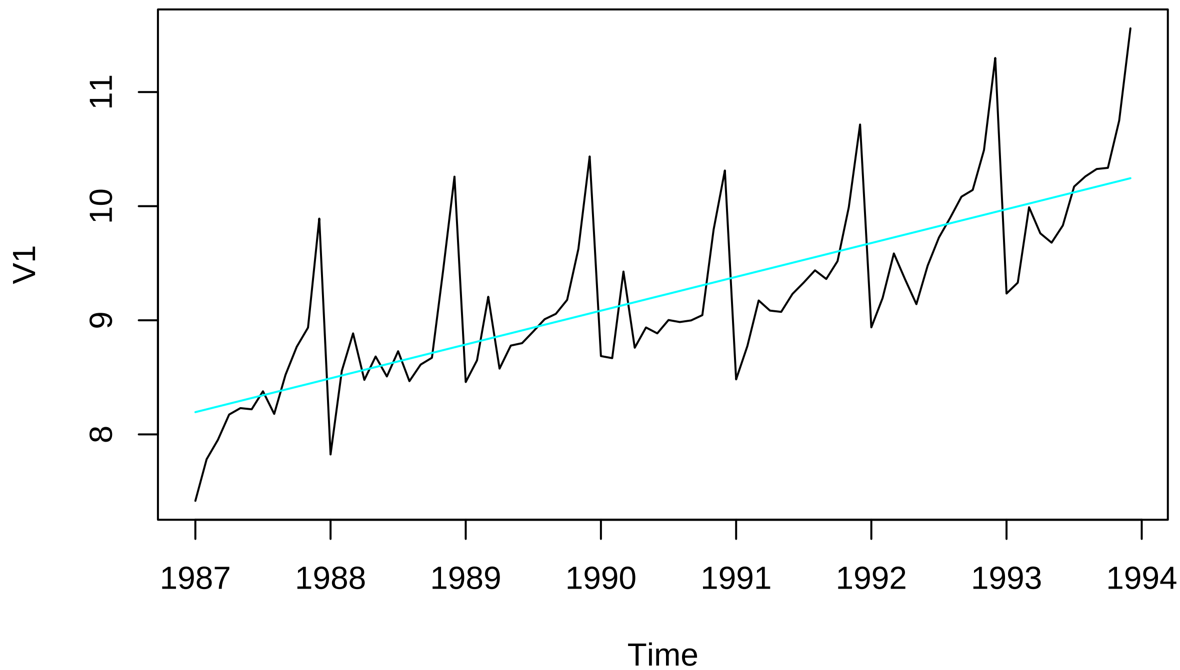
¡Se pudo estabilizar! Vemos que pasa la prueba de kpss y de df.

Proponemos una tendencia lineal

```r
M <- factor(cycle(Serie_ln))
t = time(Serie_ln)
regresion_1= lm(Serie_ln ~t,  na.action=NULL)
summary(regresion_1)
```

```
##
## Call:
## lm(formula = Serie_ln ~ t, na.action = NULL)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.8986 -0.2874 -0.0992  0.1582  1.4961
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -580.76432   55.27122  -10.51   <2e-16 ***
## t              0.29641    0.02777   10.67   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5142 on 82 degrees of freedom
## Multiple R-squared:  0.5815, Adjusted R-squared:  0.5764
## F-statistic: 113.9 on 1 and 82 DF,  p-value: < 2.2e-16
```

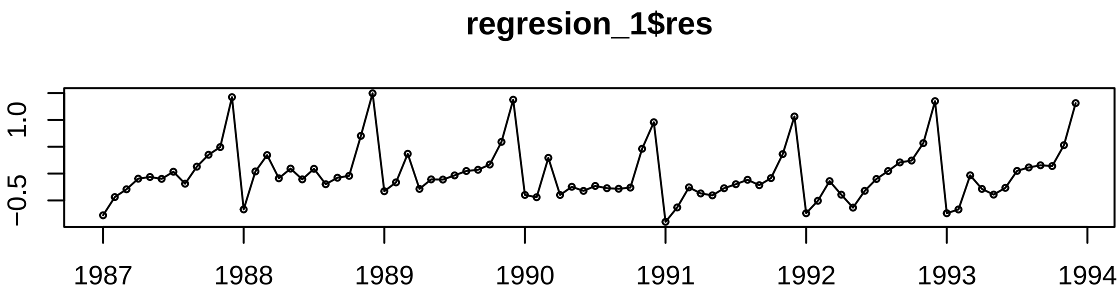
```r
par(mfrow=c(1,1))
plot(Serie_ln, type="l",col='black')
lines(fitted(regresion_1), col='cyan')
```

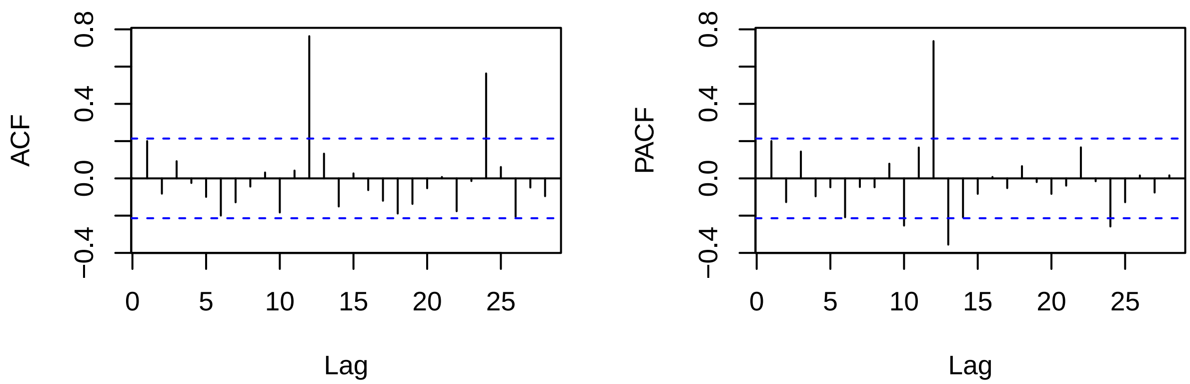


Observamos que todos los valores,tienen un p-value menor a 0.05.

Comrpobamos los supuestos y quitamos la tendencia:

```
tsdisplay(regresion_1$res)
```

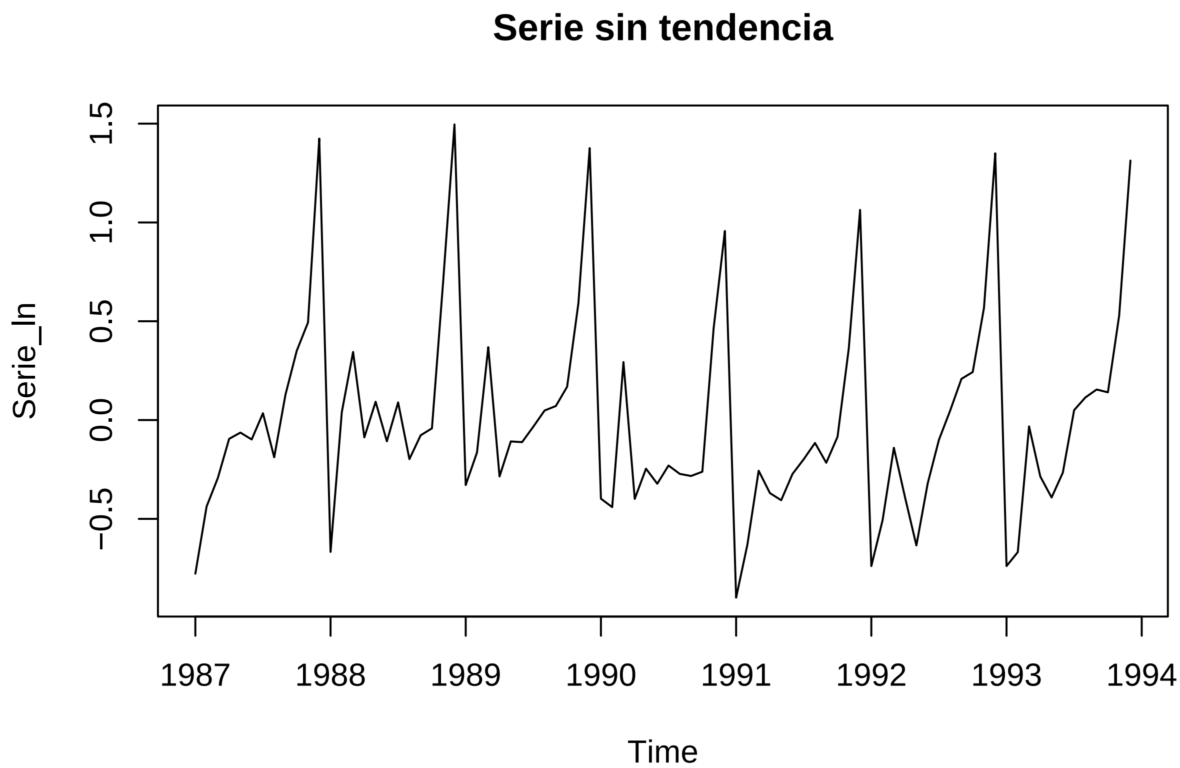





```
bptest(regresion_1$res~t)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  regresion_1$res ~ t
## BP = 0.3213, df = 1, p-value = 0.5708
```

```
sin_tendencia=Serie_ln-regresion_1$fitted.values
sin_tendencia<-ts(sin_tendencia,start=start(Serie_ln),end=end(Serie_ln),frequency = 12)
plot(sin_tendencia,main='Serie sin tendencia')
```



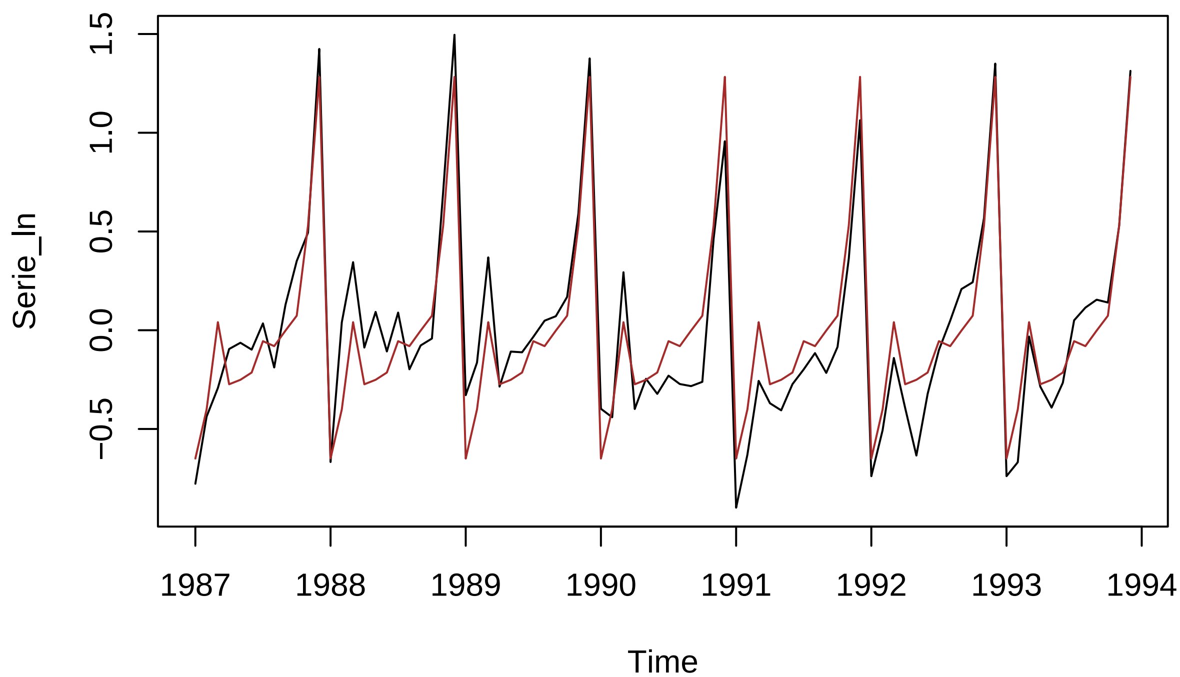
Ahora los ciclos, usando cycle, ya que en el inciso b se muestra más a detalle que el periodo es el mismo que la frecuencia de la serie.

```
regresion_2<-lm(sin_tendencia~ M ,na.action = NULL)
summary(regresion_2)
```

```
##
## Call:
## lm(formula = sin_tendencia ~ M, na.action = NULL)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.38351 -0.12104 -0.01479  0.13355  0.44105
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.64975    0.07332  -8.862 3.77e-13 ***
## M2           0.24874    0.10369   2.399 0.019033 *
## M3           0.69059    0.10369   6.660 4.64e-09 ***
## M4           0.37601    0.10369   3.626 0.000533 ***
## M5           0.39876    0.10369   3.846 0.000257 ***
## M6           0.43542    0.10369   4.199 7.55e-05 ***
## M7           0.59437    0.10369   5.732 2.16e-07 ***
## M8           0.56920    0.10369   5.490 5.72e-07 ***
## M9           0.64788    0.10369   6.249 2.60e-08 ***
## M10          0.72327    0.10369   6.976 1.23e-09 ***
## M11          1.17994    0.10369  11.380  < 2e-16 ***
```

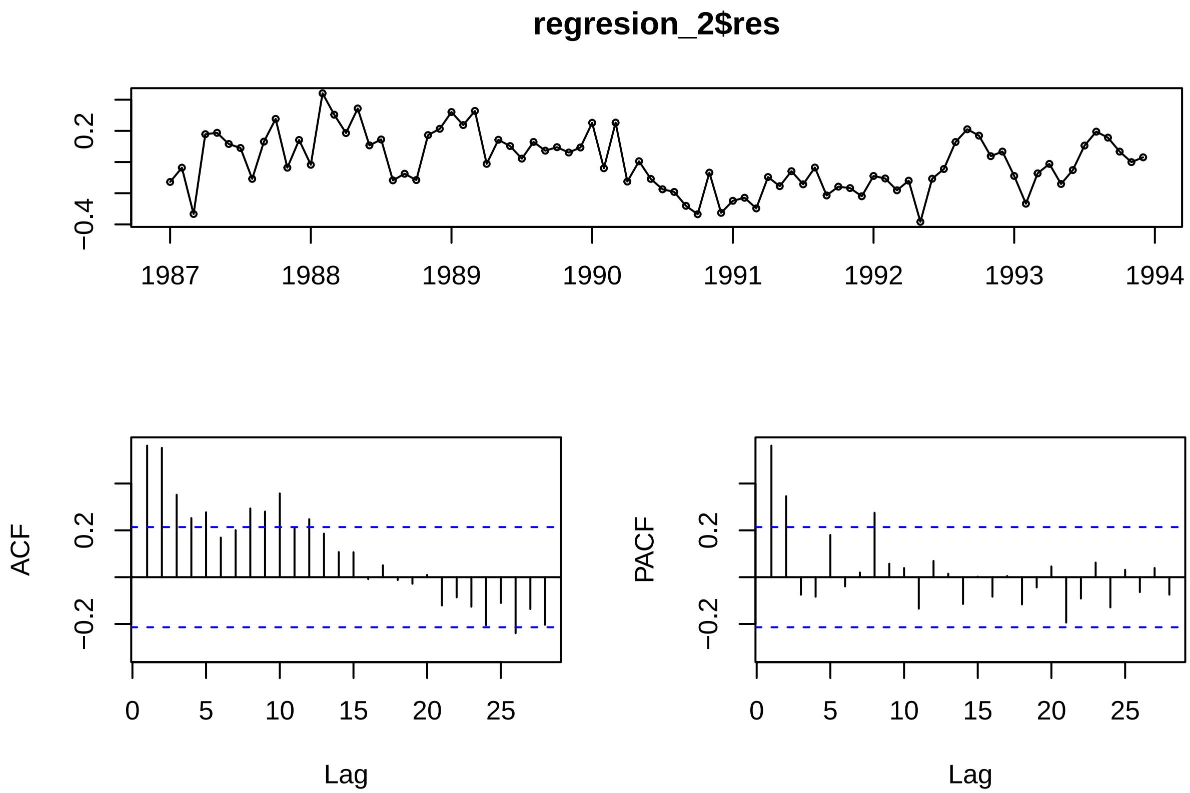
```
## M12           1.93275    0.10369  18.641  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.194 on 72 degrees of freedom
## Multiple R-squared:  0.8751, Adjusted R-squared:  0.856
## F-statistic: 45.84 on 11 and 72 DF,  p-value: < 2.2e-16
```

```
par(mfrow=c(1,1))
plot(sin_tendencia, type="l",col='black')
lines(fitted(regresion_2), col='brown')
```



Comprobamos supuestos y quitamos ciclos:

```
tsdisplay(regresion_2$res)
```

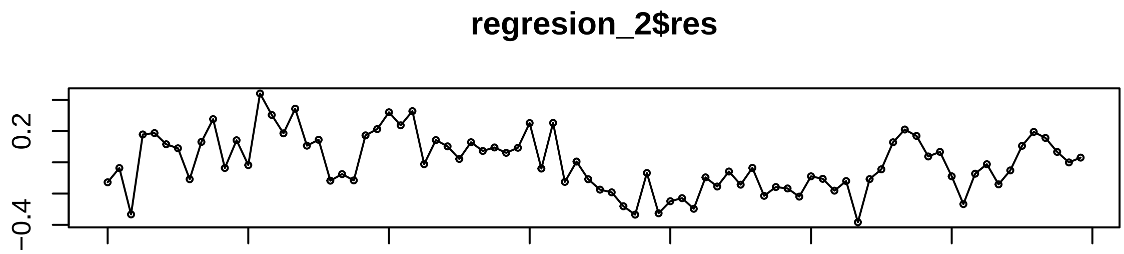


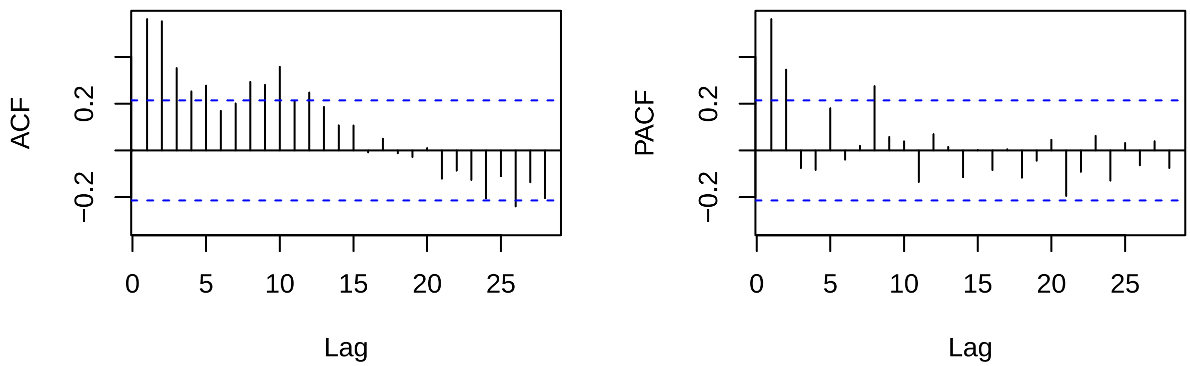

```
bptest(regresion_2$res~t)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  regresion_2$res ~ t
## BP = 0.68569, df = 1, p-value = 0.4076
```

Nos quedamos con la parte aleatoria:

```
aleatoria<-sin_tendencia-regresion_2$fitted.values
aleatoria=ts(aleatoria,start = start(Serie_ln),end=end(Serie_ln),frequency=12)
tsdisplay(regresion_2$res)
```





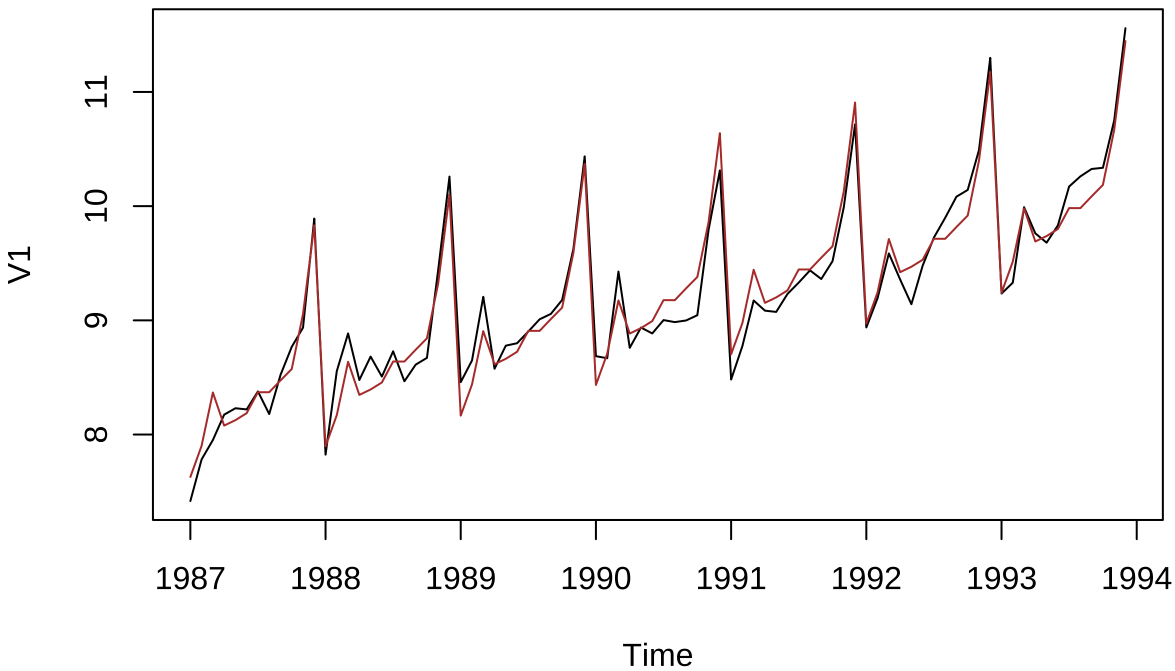
Loggramos quitar de manera sgnificativa la tendencia y ciclos. Si los juntamos en una sola regresión el resultado mejora:

```
regresion_3<-lm(Serie_ln~t+M,na.action = NULL)
summary(regresion_3)
```

```
##
## Call:
## lm(formula = Serie_ln ~ t + M, na.action = NULL)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.41644 -0.12619  0.00608  0.11389  0.38567
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept) -526.30995   20.17321 -26.090  < 2e-16 ***
## t              0.26872    0.01014  26.508  < 2e-16 ***
## M2             0.25104    0.09933   2.527 0.013718 *
## M3             0.69521    0.09934   6.998 1.18e-09 ***
## M4             0.38293    0.09936   3.854 0.000252 ***
## M5             0.40799    0.09938   4.105 0.000106 ***
## M6             0.44696    0.09941   4.496 2.63e-05 ***
## M7             0.60822    0.09945   6.116 4.69e-08 ***
## M8             0.58535    0.09950   5.883 1.21e-07 ***
## M9             0.66634    0.09955   6.693 4.27e-09 ***
## M10            0.74403    0.09961   7.469 1.61e-10 ***
```

```
## M11            1.20302    0.09968   12.068  < 2e-16 ***
## M12            1.95814    0.09976   19.629  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1858 on 71 degrees of freedom
## Multiple R-squared:  0.9527, Adjusted R-squared:  0.9447
## F-statistic: 119.1 on 12 and 71 DF,  p-value: < 2.2e-16
```

```
par(mfrow=c(1,1))
plot(Serie_ln, type="l",col='black')
lines(fitted(regresion_3), col='brown')
```



Comprobando supuestos:

```
ad.test(regresion_3$residuals)
```

```
##
##  Anderson-Darling normality test
##
## data:  regresion_3$residuals
## A = 0.23786, p-value = 0.7765
```

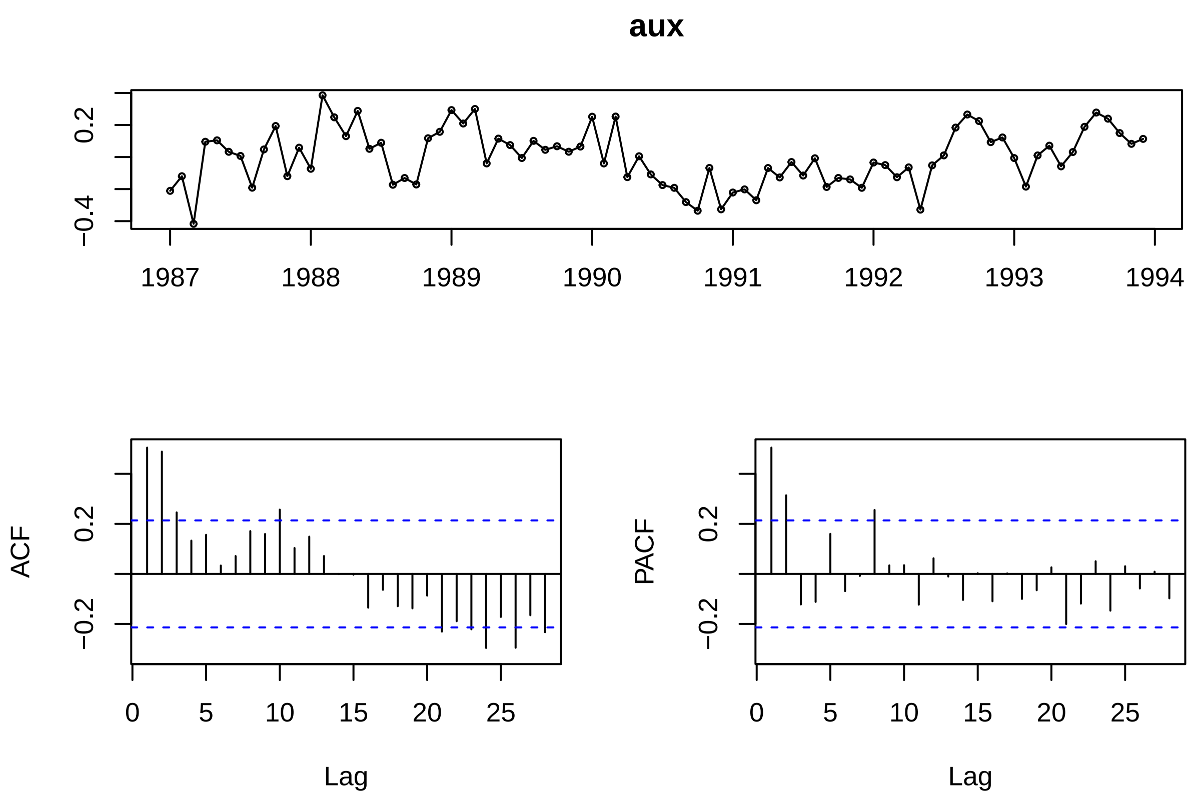
```
bptest(regresion_3)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  regresion_3
```

```
## BP = 20.975, df = 12, p-value = 0.05075
```

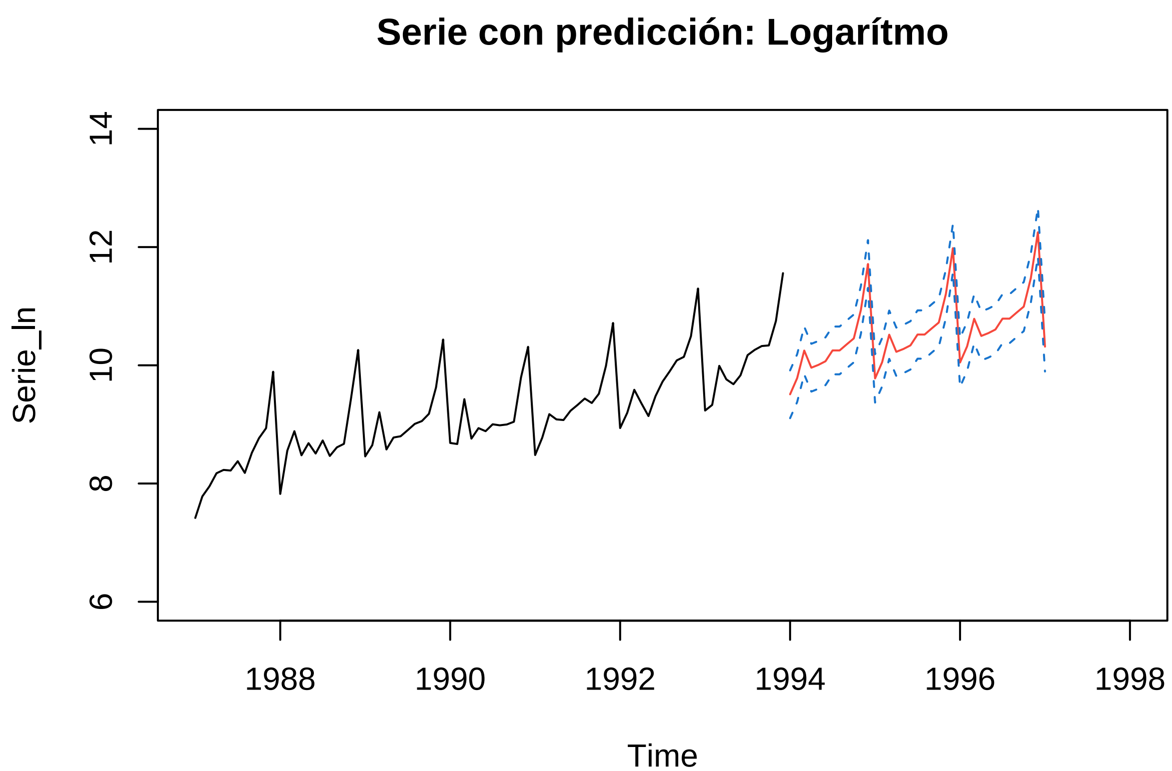
¡Pasamos los supuestos! Visualizamos la parte aleatoria:

```
aux<-Serie_ln-regresion_3$fitted.values
aux=ts(aux,start = start(Serie_ln),end=end(Serie_ln),frequency=12)
tsdisplay(aux)
```

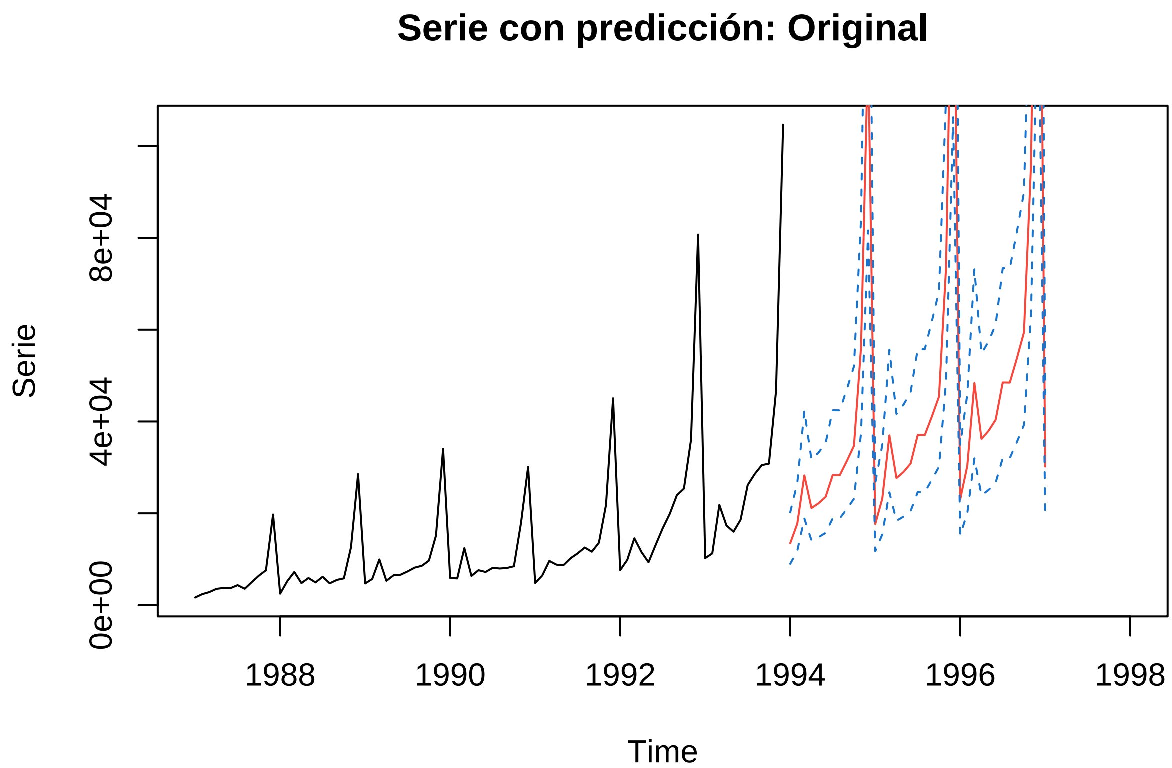


Hagamos las predicciones:

```
tnew = 1994 + seq(0,3,length.out=37)
Mnew = factor(c((1:12),(1:12),(1:12),1))
pred1<- predict(regresion_3, newdata=list(t=tnew, M=Mnew), interval="prediction")
par(mfrow=c(1,1))
ts.plot(Serie_ln, xlim=c(1987,1998), ylim=c(6,14),main='Serie con predicción: Logarítmo')
lines(tnew,(pred1[,1]), lty=1,col=2)
lines(tnew,(pred1[,2]), lty=2,col=4)
lines(tnew,(pred1[,3]), lty=2,col=4)
```




```
ts.plot(Serie, xlim=c(1987,1998),main='Serie con predicción: Original')
lines(tnew,exp(pred1[,1]), lty=1,col=2)
lines(tnew,exp(pred1[,2]), lty=2,col=4)
lines(tnew,exp(pred1[,3]), lty=2,col=4)
```



Sin regresar a los valores normales, las predicciones son:

```
ts(data=pred1,start=c(1994,1),end=c(1997,1),frequency=12)
```

```
##                fit       lwr       upr
## Jan 1994  9.509264  9.105002  9.913525
## Feb 1994  9.782700  9.378439 10.186962
## Mar 1994 10.249256  9.844995 10.653518
## Apr 1994  9.959377  9.555115 10.363638
## May 1994 10.006830  9.602569 10.411091
## Jun 1994 10.068191  9.663930 10.472452
## Jul 1994 10.251837  9.847576 10.656099
## Aug 1994 10.251367  9.847106 10.655628
## Sep 1994 10.354752  9.950491 10.759014
## Oct 1994 10.454834 10.050573 10.859095
## Nov 1994 10.936210 10.531949 11.340471
## Dec 1994 11.713723 11.309462 12.117984
## Jan 1995  9.777979  9.369195 10.186763
## Feb 1995 10.051416  9.642632 10.460200
## Mar 1995 10.517972 10.109188 10.926756
## Apr 1995 10.228092  9.819308 10.636876
## May 1995 10.275546  9.866762 10.684330
## Jun 1995 10.336907  9.928123 10.745691
## Jul 1995 10.520553 10.111769 10.929337
## Aug 1995 10.520083 10.111299 10.928867
## Sep 1995 10.623468 10.214684 11.032252
## Oct 1995 10.723550 10.314766 11.132334
```

```
## Nov 1995 11.204926 10.796142 11.613710
## Dec 1995 11.982439 11.573655 12.391223
## Jan 1996 10.046695  9.632451 10.460940
## Feb 1996 10.320132  9.905887 10.734376
## Mar 1996 10.786688 10.372443 11.200932
## Apr 1996 10.496808 10.082564 10.911053
## May 1996 10.544261 10.130017 10.958506
## Jun 1996 10.605623 10.191378 11.019867
## Jul 1996 10.789269 10.375024 11.203513
## Aug 1996 10.788798 10.374554 11.203043
## Sep 1996 10.892184 10.477939 11.306428
## Oct 1996 10.992266 10.578021 11.406510
## Nov 1996 11.473641 11.059397 11.887886
## Dec 1996 12.251155 11.836910 12.665399
## Jan 1997 10.315411  9.894804 10.736018
```

Entonces las predicciones son, regresándolas a sus valores normales::

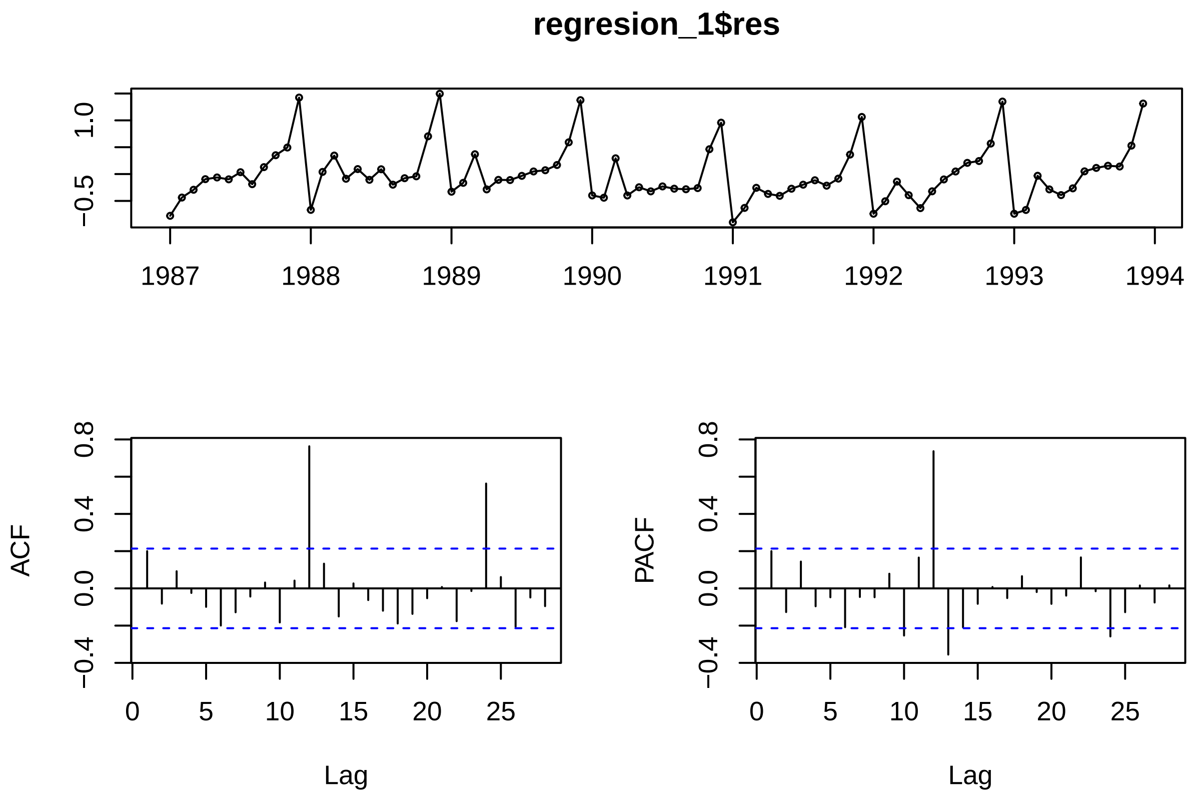
```r
pred2<-ts(data=exp(pred1),start=c(1994,1),end=c(1997,1),frequency=12)
pred2
```

```
##                fit        lwr       upr
## Jan 1994  13484.06   9000.202  20201.76
## Feb 1994  17724.45  11830.533  26554.69
## Mar 1994  28261.51  18863.703  42341.27
## Apr 1994  21149.61  14116.722  31686.25
## May 1994  22177.42  14802.755  33226.11
## Jun 1994  23580.87  15739.516  35328.75
## Jul 1994  28334.55  18912.452  42450.69
## Aug 1994  28321.23  18903.560  42430.74
## Sep 1994  31405.93  20962.508  47052.23
## Oct 1994  34711.77  23169.053  52005.01
## Nov 1994  56174.03  37494.462  84159.68
## Dec 1994 122237.73  81589.975 183136.01
## Jan 1995  17640.97  11721.681  26549.43
## Feb 1995  23188.60  15407.845  34898.53
## Mar 1995  36974.06  24567.703  55645.47
## Apr 1995  27669.68  18385.332  41642.49
## May 1995  29014.35  19278.808  43666.20
## Jun 1995  30850.46  20498.826  46429.53
## Jul 1995  37069.62  24631.193  55789.28
## Aug 1995  37052.19  24619.613  55763.05
## Sep 1995  41087.86  27301.145  61836.67
## Oct 1995  45412.83  30174.903  68345.69
## Nov 1995  73491.54  48832.025 110603.79
## Dec 1995 159921.57 106261.125 240679.82
## Jan 1996  23079.39  15251.766  34924.36
## Feb 1996  30337.26  20048.051  45907.16
## Mar 1996  48372.55  31966.480  73198.66
## Apr 1996  36199.78  23922.233  54778.49
## May 1996  37958.98  25084.787  57440.57
## Jun 1996  40361.14  26672.224  61075.57
## Jul 1996  48497.56  32049.090  73387.82
## Aug 1996  48474.75  32034.022  73353.32
## Sep 1996  53754.55  35523.121  81342.86
```

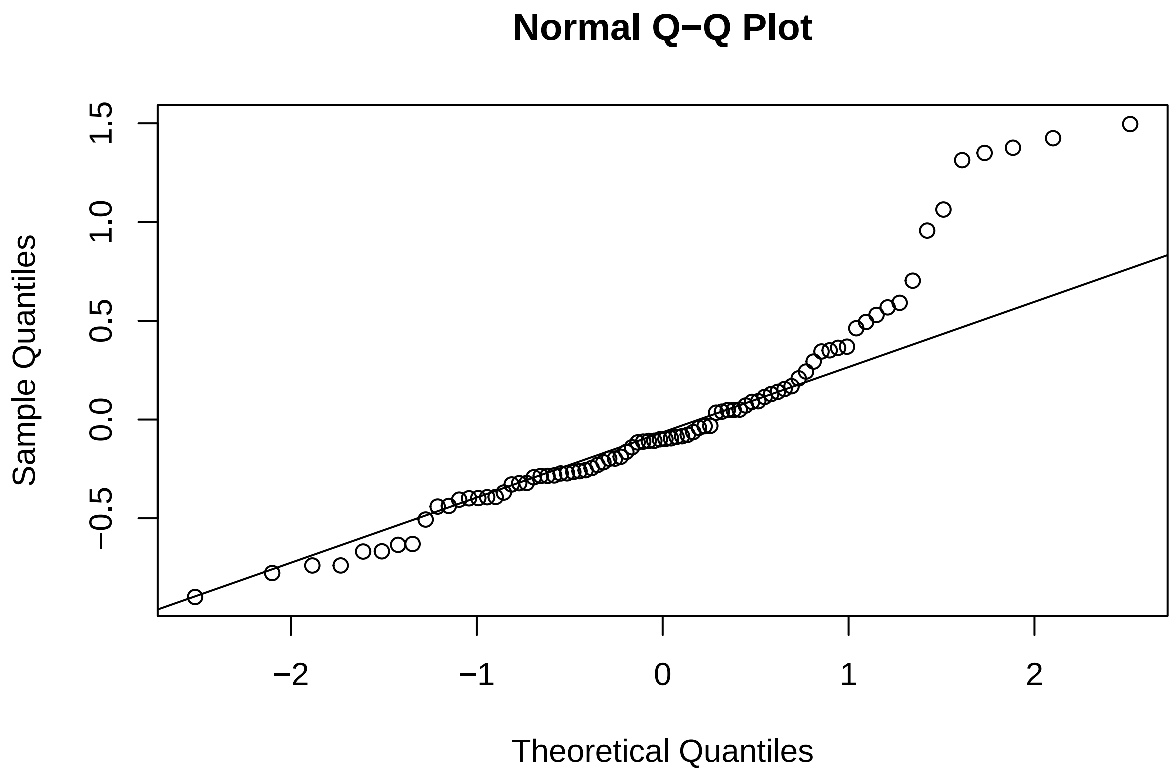
```
## Oct 1996  59412.84  39262.336  89905.12
## Nov 1996  96147.75  63538.212 145493.40
## Dec 1996 209222.70 138262.581 316601.49
## Jan 1997  30194.38  19827.086  45982.57
```

Ahora comprobemos los supuestos de regresión

```
tsdisplay(regresion_1$res)
```




```
qqnorm(regresion_1$res)
qqline(regresion_1$res)
```




```
ad.test(regresion_1$res)
```

```
##
##  Anderson-Darling normality test
##
## data:  regresion_1$res
## A = 2.5603, p-value = 1.619e-06
```

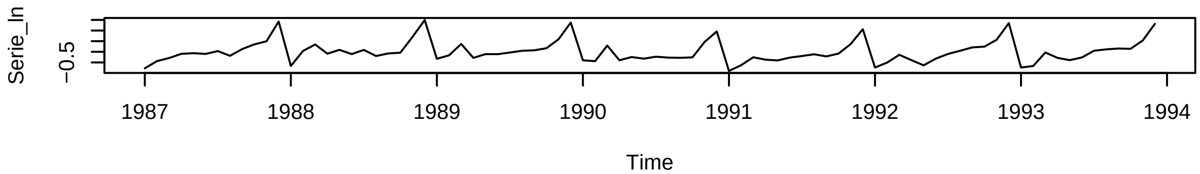
```
bptest(regresion_1)
```

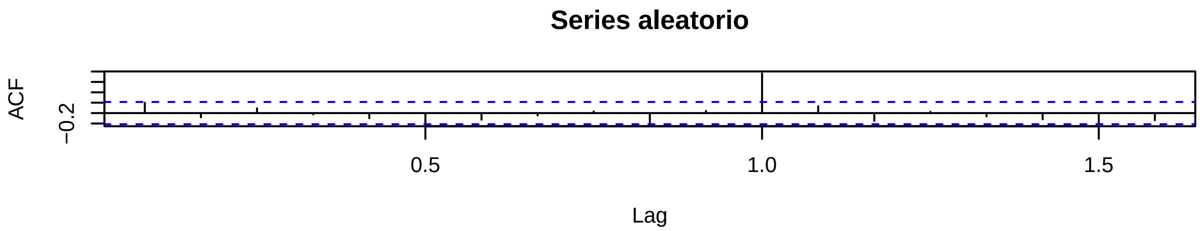
```
##
##  studentized Breusch-Pagan test
##
## data:  regresion_1
## BP = 0.3213, df = 1, p-value = 0.5708
```

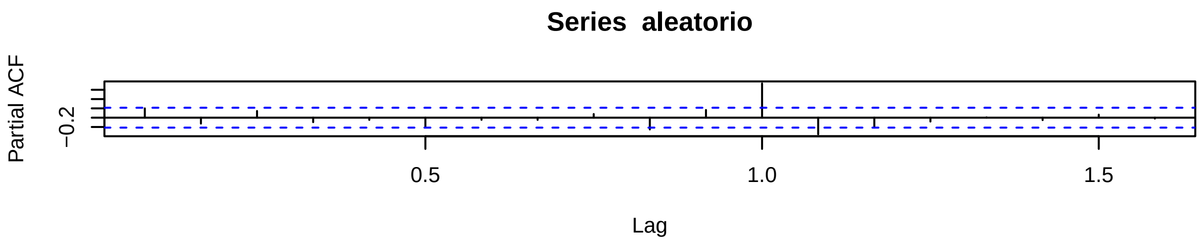
Pasamos los tests de Normalidad y homocedasticidad.

¿Desapareció la tendencia? ¿Desaparecieron los ciclos?

```
aleatorio=ts(Serie_ln-regresion_1$fit,start=start(Serie_ln),end=end(Serie_ln),frequency = 12)
par(mfrow=c(3,1))
plot(aleatorio)
acf(aleatorio)
pacf(aleatorio)
```



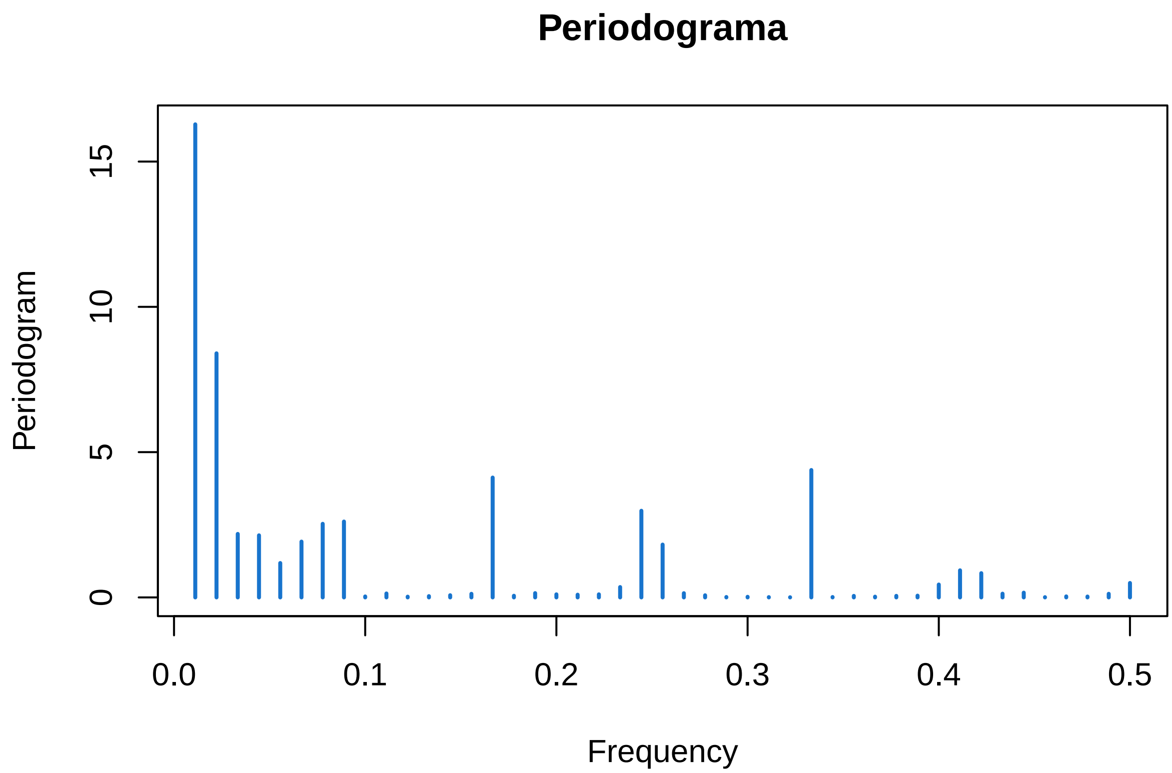




¡Parece que pudimos deshacernos de los ciclos considerablemente! Ya no son tan notorios. Aunque como es un ajuste deterministico, no es tan perfecto. (b) Filtros lineales o suavizamientos exponenciales. Realice un pronóstico de 3 años futuros.

Vamos a descomponer usando filtros lineales. Con Holt Winters trabajaremos el suavizamiento exponencial y el pronóstico.

```
#Realizamos mediante filtros lineales

#Veamos la tendencia y los ciclos
Xt = Serie_ln
p = periodogram(Xt, main="Periodograma", col=4) # Obtenemos el periodograma
```




```
names(p)
```

```
##  [1] "freq"      "spec"      "coh"       "phase"     "kernel"    "df"
##  [7] "bandwidth" "n.used"    "orig.n"    "series"    "snames"    "method"
## [13] "taper"     "pad"       "detrend"   "demean"
```

```
# Ordenamos de mayor a menor las estimaciones del periodograma.
spec = sort(p$spec, decreasing = TRUE)
(spec = spec[1:7]) # Nos quedamos con los coeficientes de mayor frecuencia.
```

```
## [1] 16.280651  8.399828  4.383422  4.124501  2.981310  2.610981  2.532479
```

```
i = match(spec, p$spec) # Buscamos sus indices en el periodograma.
d = p$freq # Vemos las frecuencias del periodograma.
d = d[i] # Nos quedamos con las frecuencias que nos interesan.

cbind(spec,d,i)#
```

```
##           spec          d  i
## [1,] 16.280651 0.01111111  1
## [2,]  8.399828 0.02222222  2
## [3,]  4.383422 0.33333333 30
## [4,]  4.124501 0.16666667 15
## [5,]  2.981310 0.24444444 22
## [6,]  2.610981 0.08888889  8
## [7,]  2.532479 0.07777778  7
```

```r
d = 1 / d # Obtenemos los parametros para utilizar en promedios moviles.
d = floor(d) #
(d = sort(d))
```

```
## [1]  2  4  6 11 12 45 90
```

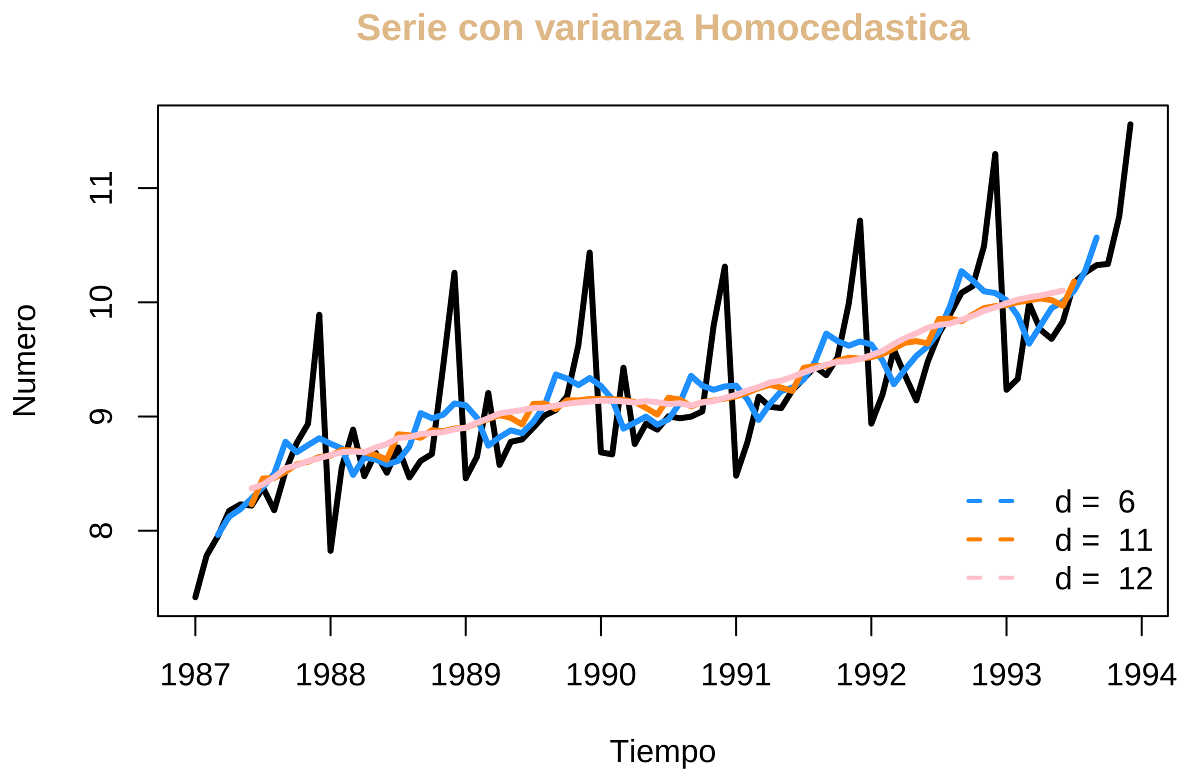
```r
# Quitamos los periodos mas grandes
d = d[-length(d)]
d = d[-length(d)]
# Quitamos los periodos mas chicos
d = d[-1]
d = d[-1]
d #Posibles periodos del ciclo
```

```
## [1]  6 11 12
```

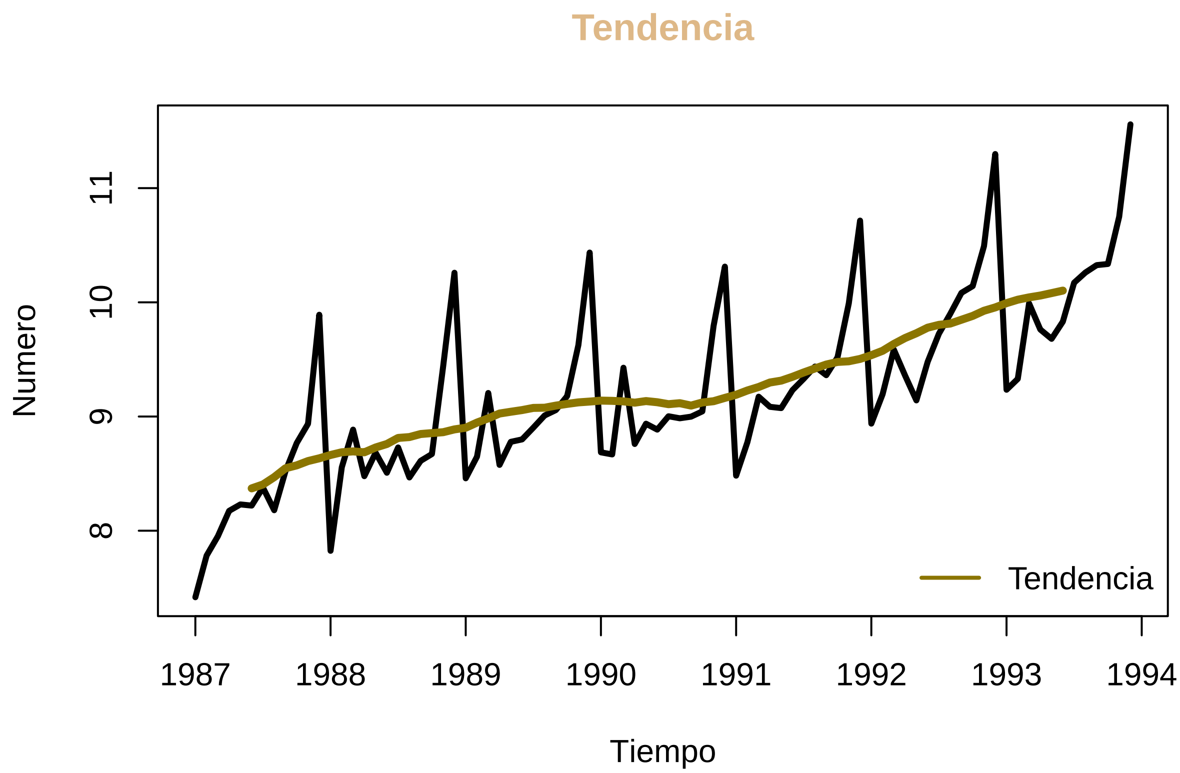
```r
#Realizamos la grafica:
col = c("dodgerblue1", "darkorange1", "pink")
plot(Serie_ln, lwd = 3, xlab = "Tiempo", col = "gray0",
     main = "Serie con varianza Homocedastica",
     ylab = "Numero", col.main = "burlywood")
library(dplyr)
```

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

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

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

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

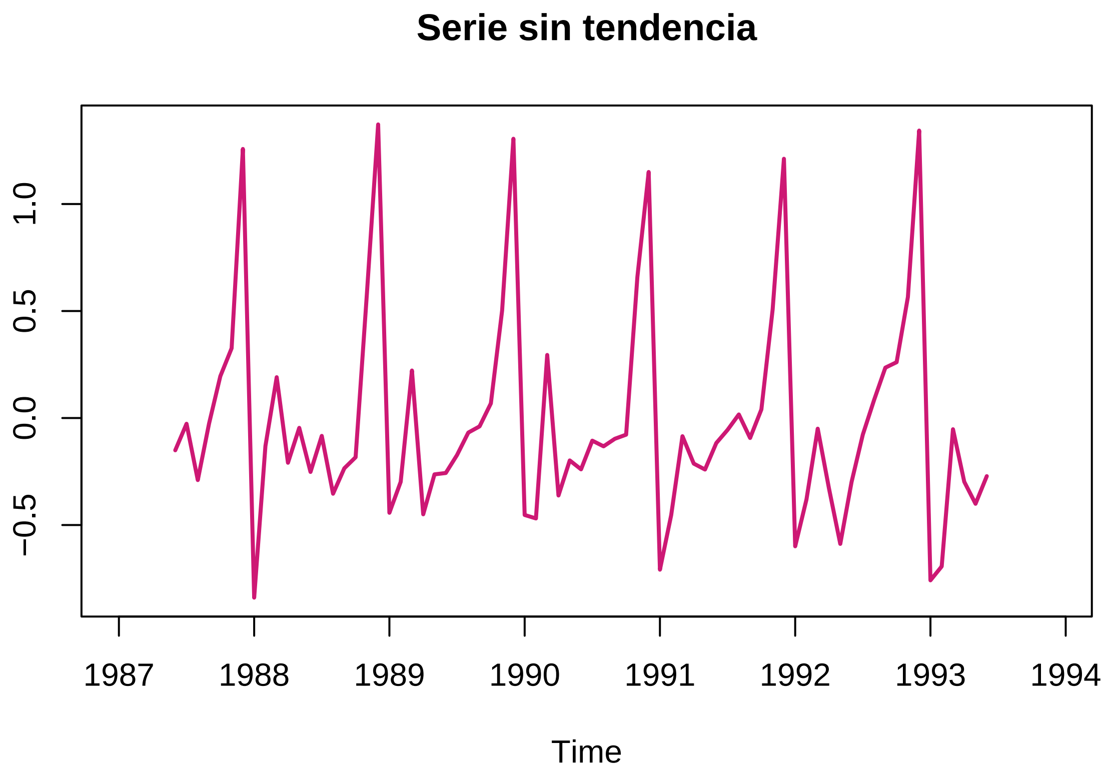
```r
t1 = seq(1987+0/12, 1993+11/12, by = 1 / 12)
for (i in 1:3) {
  lines(t1, stats::filter(Serie_ln, rep(1 / d[i], d[i])), col = col[i],
        lwd = 3)
}
legend("bottomright", col = col, lty = 2, lwd = 2, bty = "n",
       legend = c(paste("d = ", d[1]), paste("d = ", d[2]),
                  paste("d = ", d[3])), cex = 1)
```




```
# Notemos que podemos aproximar la tendencia con d = 12, ya que esta nos
# muestra un mayor ajuste.

tendencia = stats::filter(Serie_ln, rep(1 / 12, 12))
plot(Serie_ln, lwd = 3, xlab = "Tiempo", col = "black",
     main = "Tendencia",
     ylab = "Numero", col.main = "burlywood")
lines(tendencia, col = "gold4", lwd = 4)
legend("bottomright", col = "gold4", lty = 1, lwd = 2, bty = "n",
       legend = "Tendencia", cex = 1)
```




```
# Quitamos la tendencia
# Solo trabajamos con la serie cuya varianza es cte.

datosSinTendencia = Serie_ln - tendencia # Serie sin tendencia
plot(datosSinTendencia, main="Serie sin tendencia", lwd=2, ylab="", col=14)
```



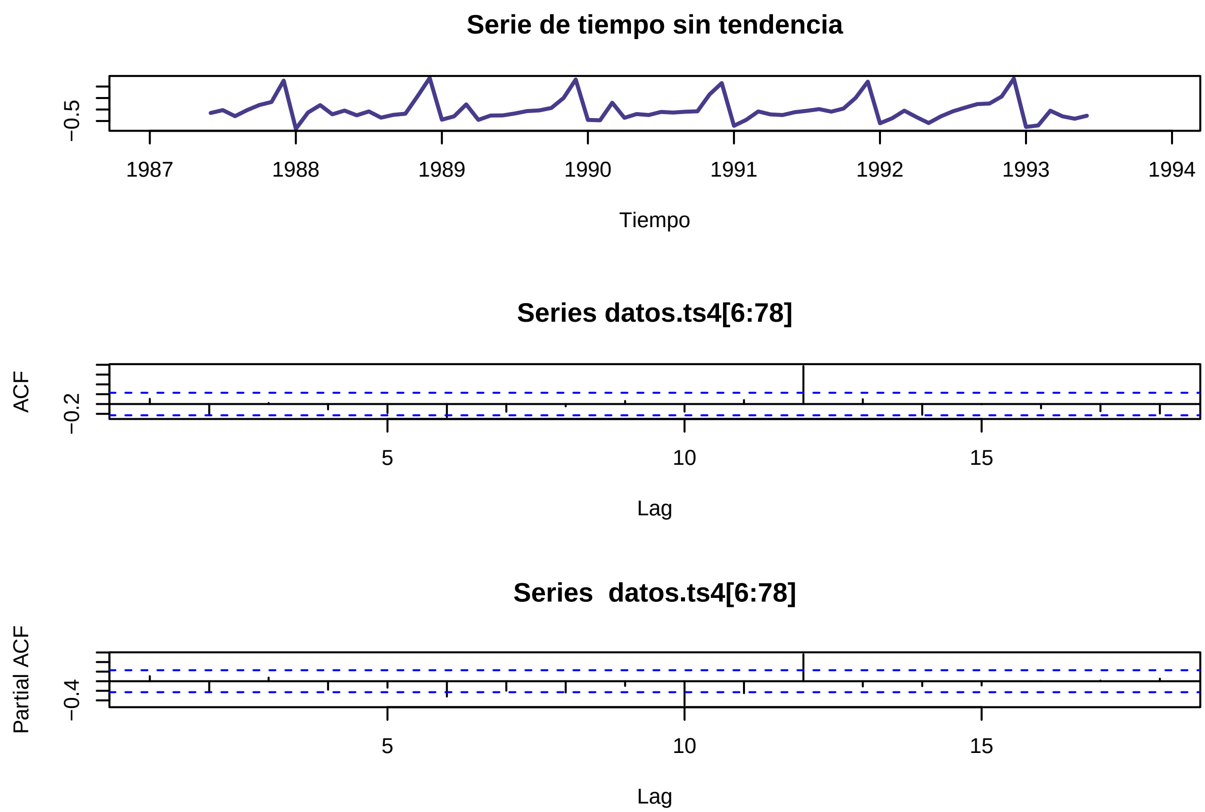

```
# Convertimos datosSinTendencia en objeto TS, dado que hicimos promedios moviles

start(Serie_ln)
```

```
## [1] 1987    1
```

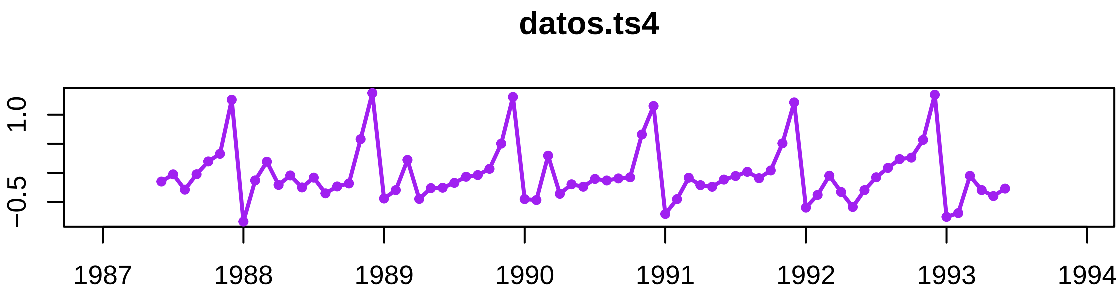
```
end(Serie_ln)
```

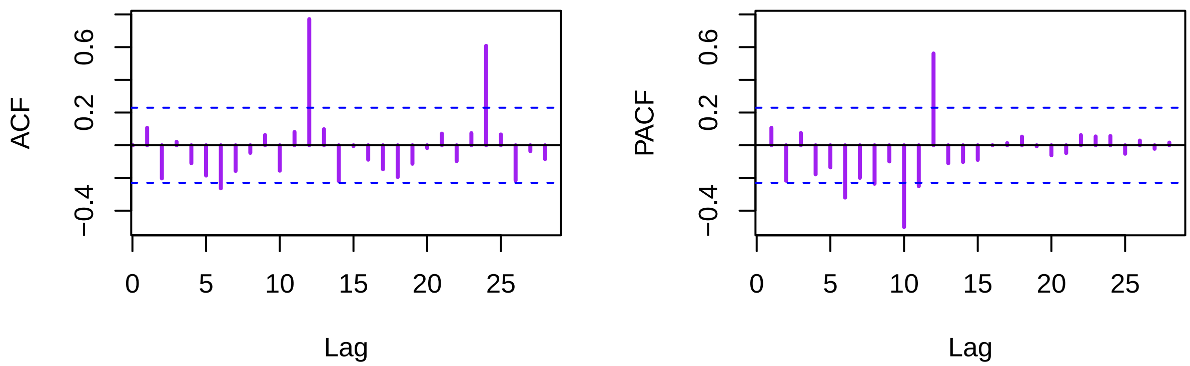
```
## [1] 1993   12
```

```
datos.ts4=ts(datosSinTendencia, frequency = 12, start=c(1987,01),end=c(1993,12))

View(datos.ts4)
par(mfrow = c(3,1))
plot(datos.ts4, col = "slateblue4", lwd = 2, ylab = " ", type = "l",
     main = "Serie de tiempo sin tendencia", xlab = "Tiempo")

acf(datos.ts4[6:78])
pacf(datos.ts4[6:78])
```



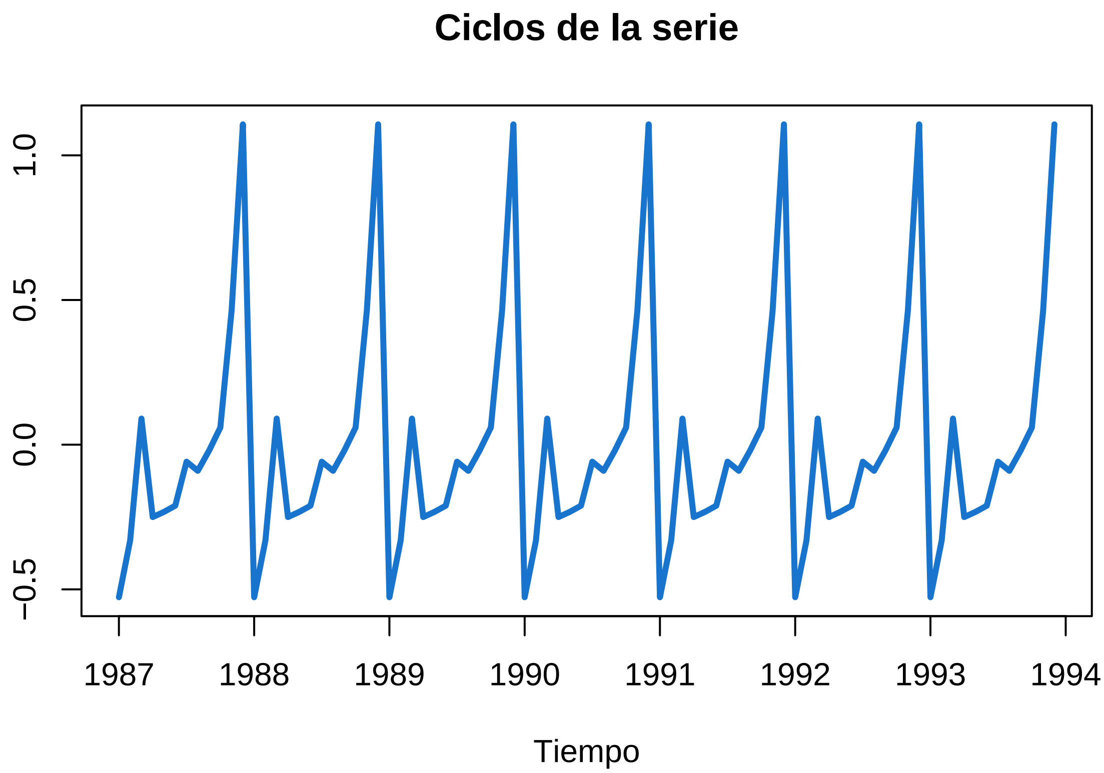

```
par(mfrow = c(1,1))
tsdisplay(datos.ts4, col="purple", lwd=2)
```



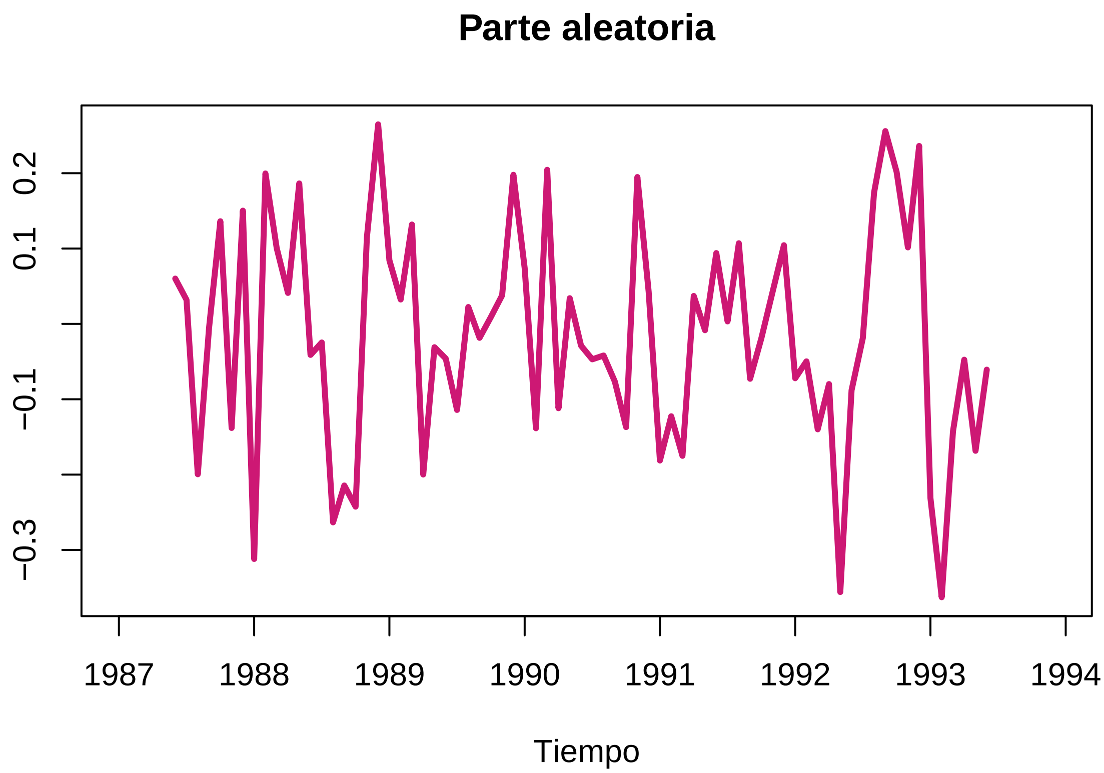



```
# Tenemos problemas de ciclos y muy marcados
```

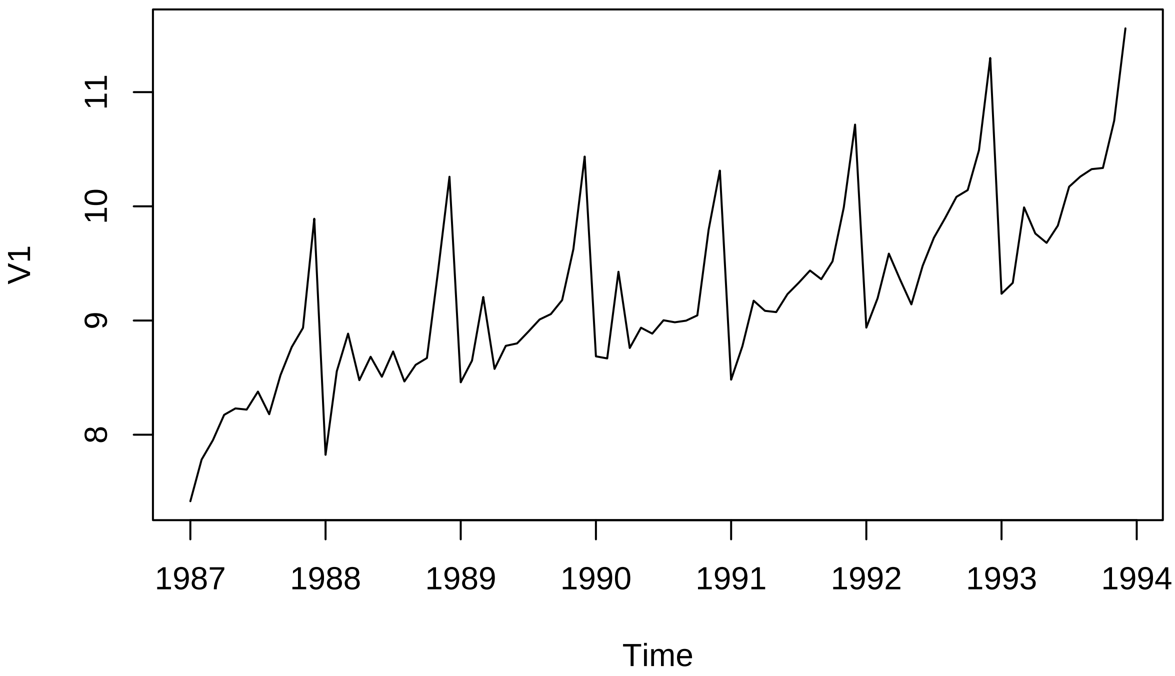
```
# Ahora, estimaremos la parte estacional. Tenemos que d = 12.
# n = length(Serie_ln) = 84, tenemos 72 (por los NA), then 72 / 12 = 6 ciclos.
# Creamos un ciclo promedio que estime la parte estacional,
# usando la serie sin tendencia.
d = 12
k = length(datos.ts4) / d # Numero de ciclos de la serie sin tendencia
w = rep(0, 12)
# Para el resto de los meses
for (i in 1:12)
  w[i] = sum(datos.ts4[d * (0:(k-1)) + i], na.rm = TRUE) / k

# Ahora, ajustamos el ciclo obtenido
ciclo  = w - mean(w)
ciclo = ts(rep(ciclo, times = k), start = start(Serie_ln),
           frequency = frequency(Serie_ln))
par(mfrow = c(1, 1))
plot(ciclo, col =20, lwd = 3, ylab = " ", xlab = "Tiempo",
     main = "Ciclos de la serie")# Es el ciclo de la serie
```



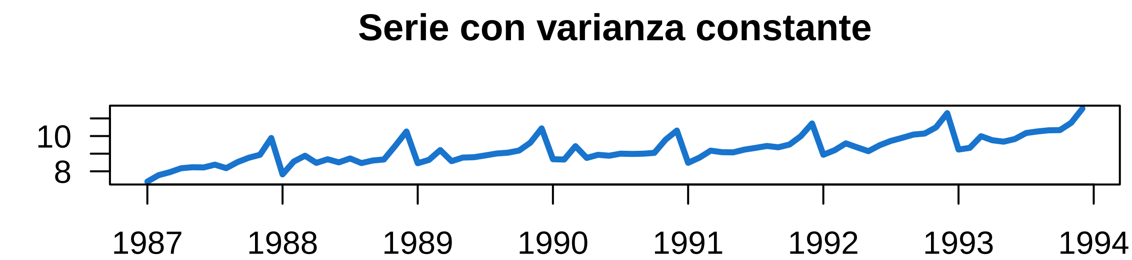

```
# Ciclos anuales

# Calculamos la parte aleatoria
parte_aleatoria = datos.ts4 - ciclo
plot(parte_aleatoria, main = "Parte aleatoria",
     col =30, lwd = 3, xlab = "Tiempo", ylab = "")
```

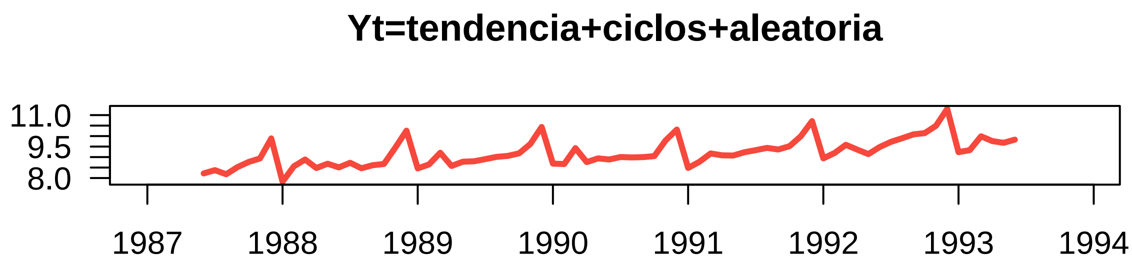



```
plot(Serie_ln)
```

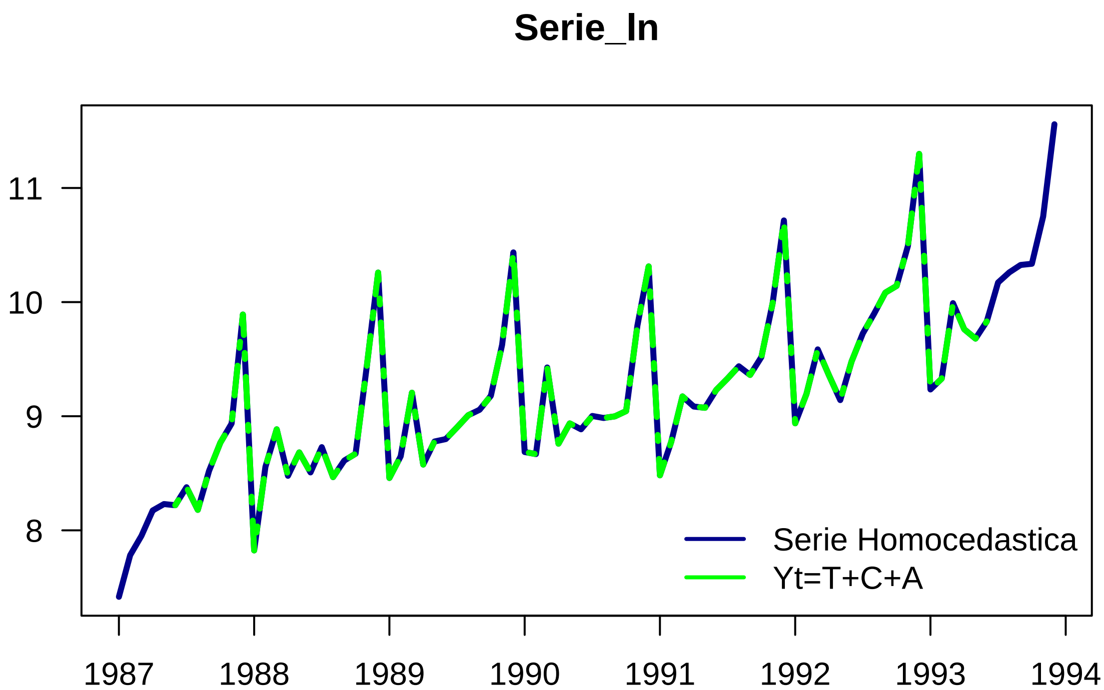



```
# Con esto, ya tenemos nuestras series

componentes = tendencia + ciclo+parte_aleatoria
componentes = ts(componentes, start = start(Serie_ln), frequency = 12)
par(mfrow = c(2,1))
plot(Serie_ln, col=28, las=1, main="Serie con varianza constante", lwd=3, xlab="",ylab="")
plot(componentes, col = 18, lwd = 3, las=1, main="Yt=tendencia+ciclos+aleatoria", xlab="",ylab="")
```






```
par(mfrow = c(1,1))
plot(Serie_ln,col="darkblue", las=1, lwd=3,main="Serie_ln", ylab="",xlab="")
invisible(lines(componentes, type="l", lwd=3, col="green",lty=6))
legend("bottomright", col = c("darkblue","green"), lty = 1, lwd = 2, bty = "n",
       legend = c("Serie Homocedastica","Yt=T+C+A"), cex = 1)
```

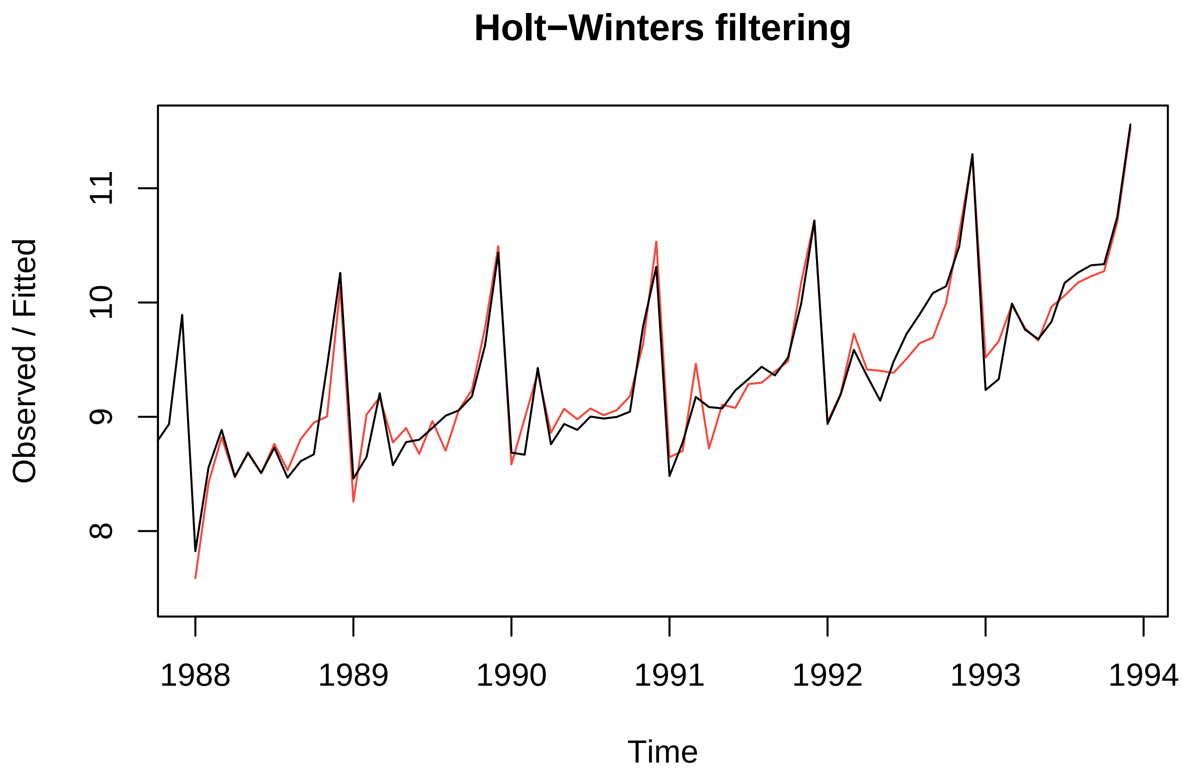


Veamos que ya la logramos descomponer.

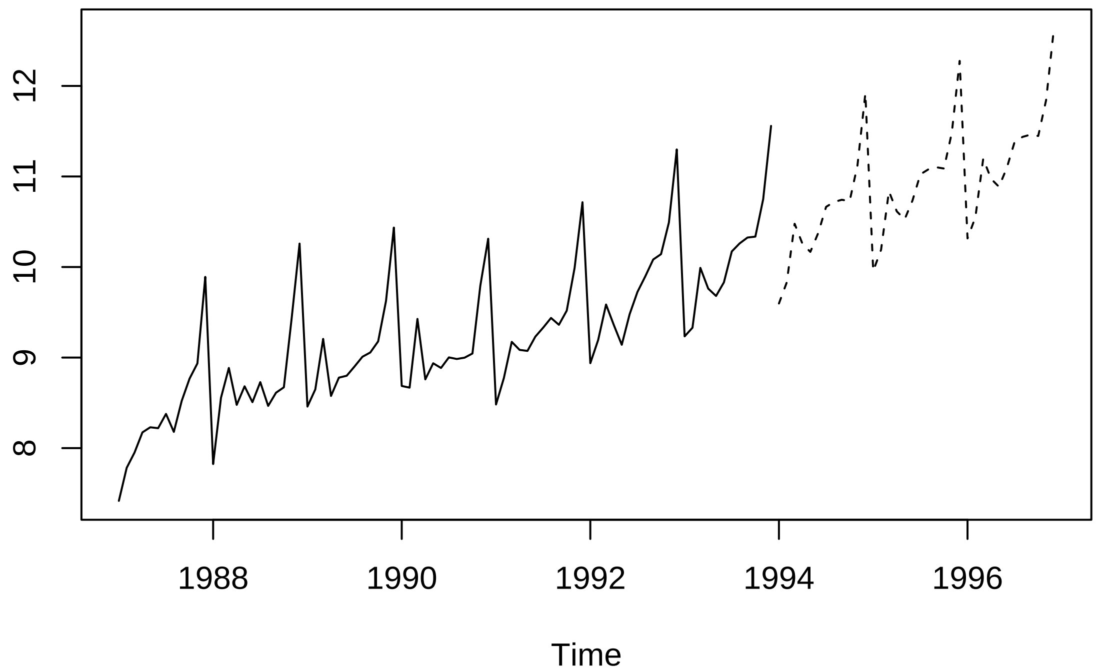
Suavizamiento exponencial

Para esta parte usaremos el método de Holt-Winter que pertenece a los métodos de suavizamiento exponencial. Usamos aditivo, debido que usamos el logaritmo.

```
xt.hw = HoltWinters(Serie_ln, seasonal="additive")
plot(xt.hw)
```




```
xt.predict = predict(xt.hw, n.ahead=3*12)
ts.plot(Serie_ln, xt.predict, lty=1:3)
```




```r
xt.predict
```

```
##            Jan       Feb       Mar       Apr       May       Jun       Jul
## 1994  9.597062  9.830781 10.477542 10.254867 10.167100 10.375632 10.664248
## 1995  9.956620 10.190339 10.837100 10.614425 10.526659 10.735190 11.023806
## 1996 10.316179 10.549898 11.196658 10.973983 10.886217 11.094748 11.383364
##            Aug       Sep       Oct       Nov       Dec
## 1994 10.717796 10.742782 10.728230 11.124151 11.917062
## 1995 11.077354 11.102340 11.087788 11.483709 12.276620
## 1996 11.436912 11.461898 11.447346 11.843268 12.636179
```

Explícitamente los valores de predicción son:

```r
xt.predict
```


```
##            Jan       Feb       Mar       Apr       May       Jun       Jul
## 1994  9.597062  9.830781 10.477542 10.254867 10.167100 10.375632 10.664248
## 1995  9.956620 10.190339 10.837100 10.614425 10.526659 10.735190 11.023806
## 1996 10.316179 10.549898 11.196658 10.973983 10.886217 11.094748 11.383364
##            Aug       Sep       Oct       Nov       Dec
## 1994 10.717796 10.742782 10.728230 11.124151 11.917062
## 1995 11.077354 11.102340 11.087788 11.483709 12.276620
## 1996 11.436912 11.461898 11.447346 11.843268 12.636179
```


Pero al hacer la transformación inversa:

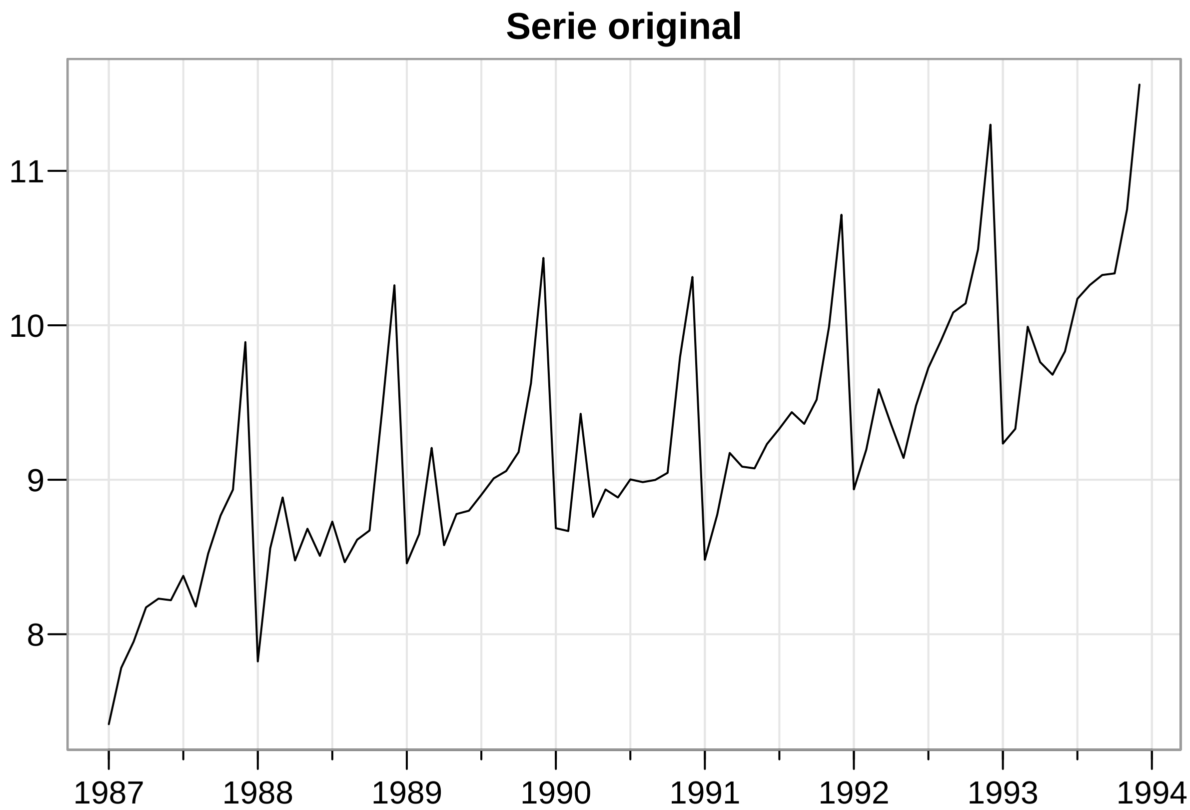
```r
exp(xt.predict)
```

```
##            Jan       Feb       Mar       Apr       May       Jun       Jul
```

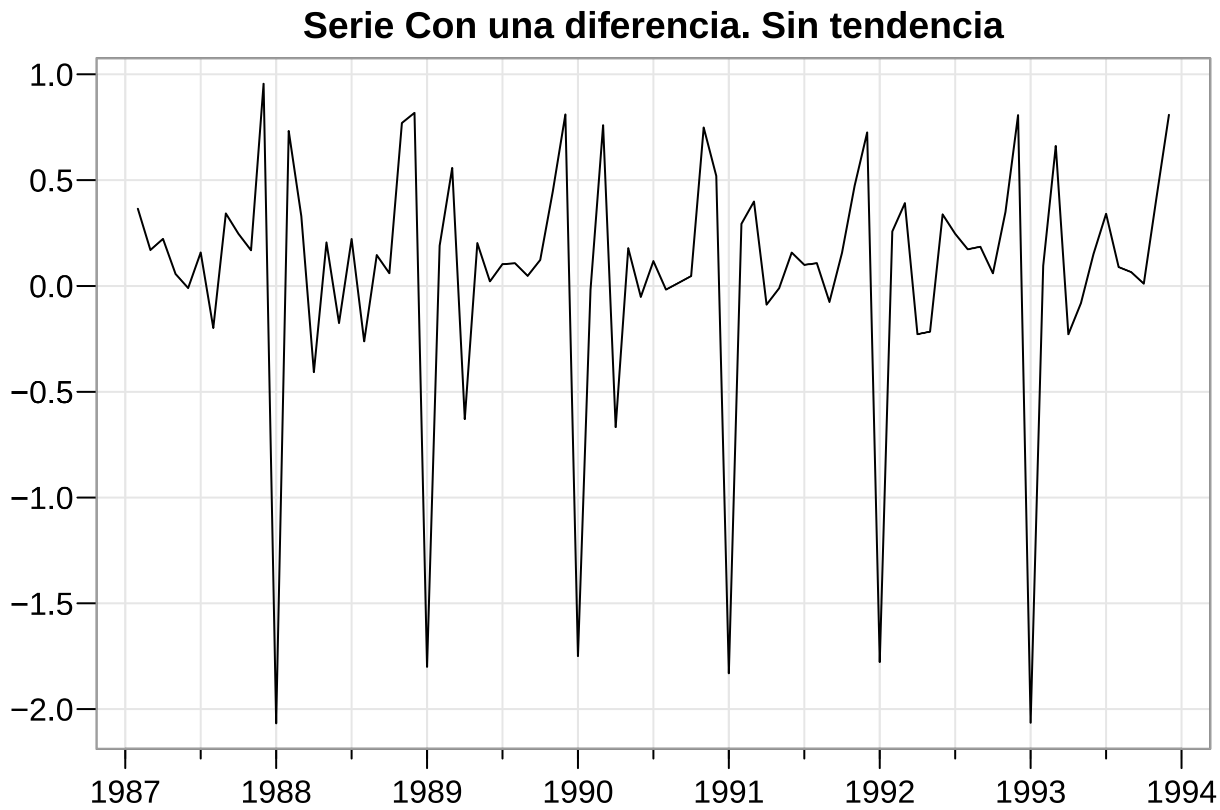
```
## 1994   14721.47   18597.48   35509.00   28420.52   26032.48   32068.58   42798.06
## 1995   21091.39   26644.54   50873.62   40717.98   37296.64   45944.53   61316.62
## 1996   30217.57   38173.53   72886.45   58336.51   53434.77   65824.57   87848.10
##              Aug        Sep        Oct        Nov        Dec
## 1994   45152.26   46294.66   45625.85   67788.72  149800.86
## 1995   64689.48   66326.19   65367.99   97120.65  214619.15
## 1996   92680.39   95025.29   93652.48  139144.40  307484.08
```

(c) Diferencias.

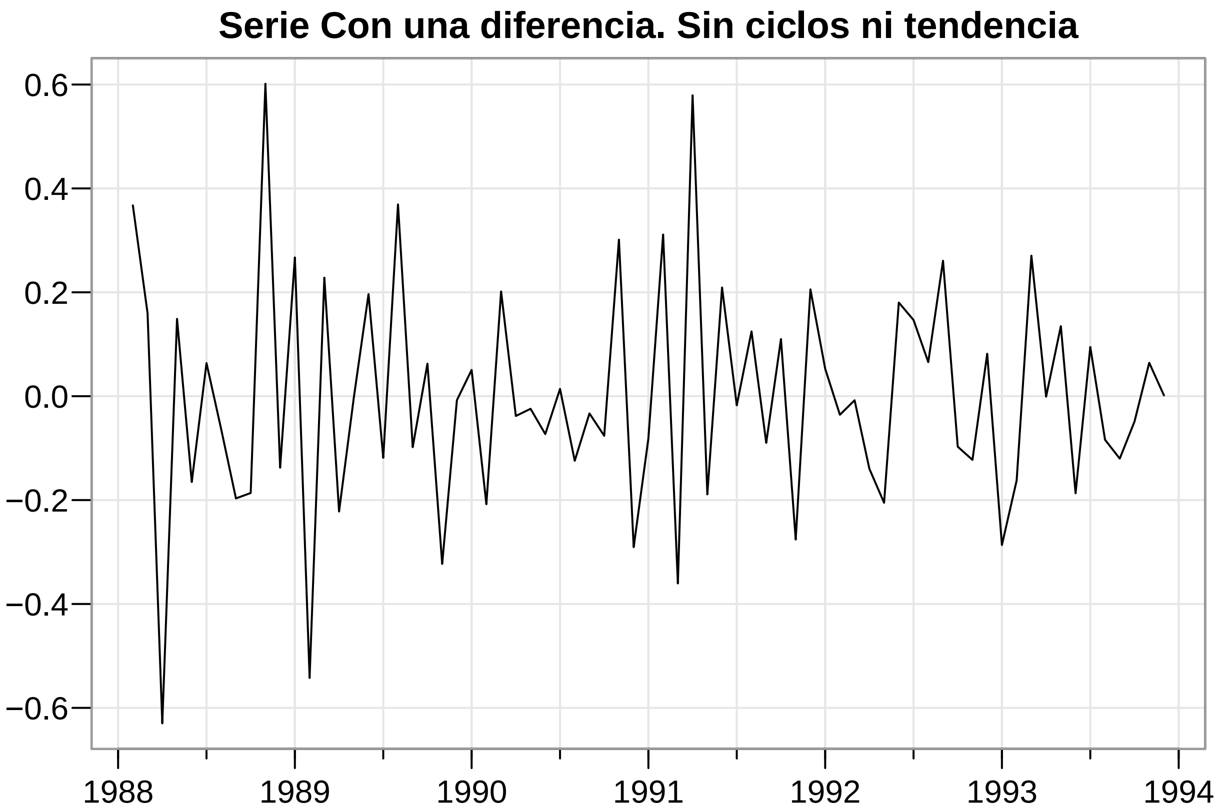
```
yt= Serie_ln
tsplot(Serie_ln, main="Serie original", ylab="", xlab="", las=1)
```



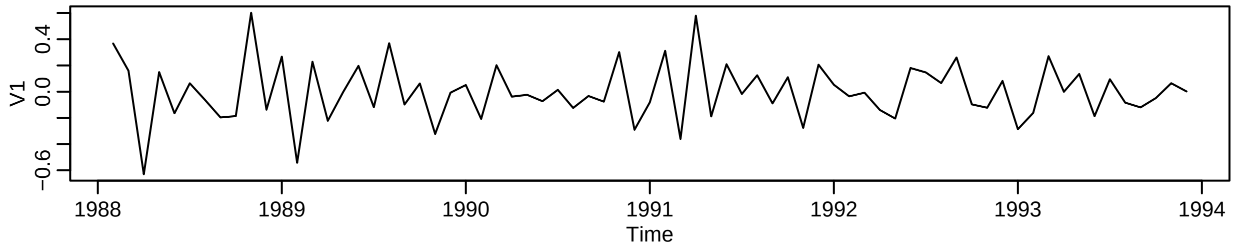

```
wt= diff(yt)
tsplot(wt, main="Serie Con una diferencia. Sin tendencia", ylab="", xlab="", las=1)
```

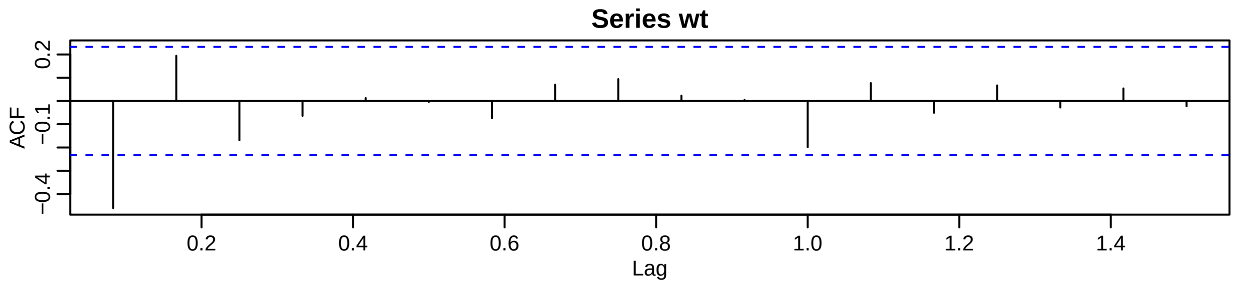


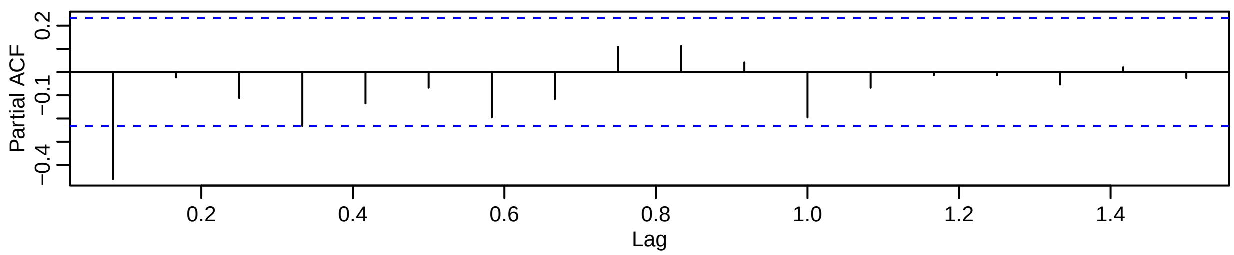

```r
#Como el periodo es de 12, hacemos diiferencia con lag=12 para quitarla
wt=diff(wt,12)
tsplot(wt, main="Serie Con una diferencia. Sin ciclos ni tendencia", ylab="", xlab="", las=1)
```




```
par(mfrow=c(3,1))
plot(wt)
acf(wt)
pacf(wt)
```







Hacemos la diferencia con lag igual a 12, ya que en el inciso b) el periodo era de 12. 4.- Describa brevemente en qué consisten los métodos de suavizado exponencial (exponential smoothing) para las series de tiempo y el método de Holt Winters.

En las notas del curso se nos describe de manera breve y concisa cómo es que funcionan:

La selección del método se basa generalmente en el reconocimiento de la tendencia y estacionalidad, así como en la forma en que estos entran en el método de suavizamiento, como aditiva o multiplicativa. Generalmente se usa el promedio para pronosticar si todos los pronósticos futuros son iguales a un promedio simple de los datos obervados, puede ser sensato asignar mayor peso a las obser vaciones más recientes que a las del pasado más distante. En palabras más simples podemos definir lo de la siguiente manera: "Son básicamente promedios ponderados de observaciones pasadas, con los pesos decayendo exponencialmente a medida que las observaciones"envejecen"(...)".

Por otro lado, el método de Holt Winters habla de la forma del componente para el método aditivo y el método multiplicativo.

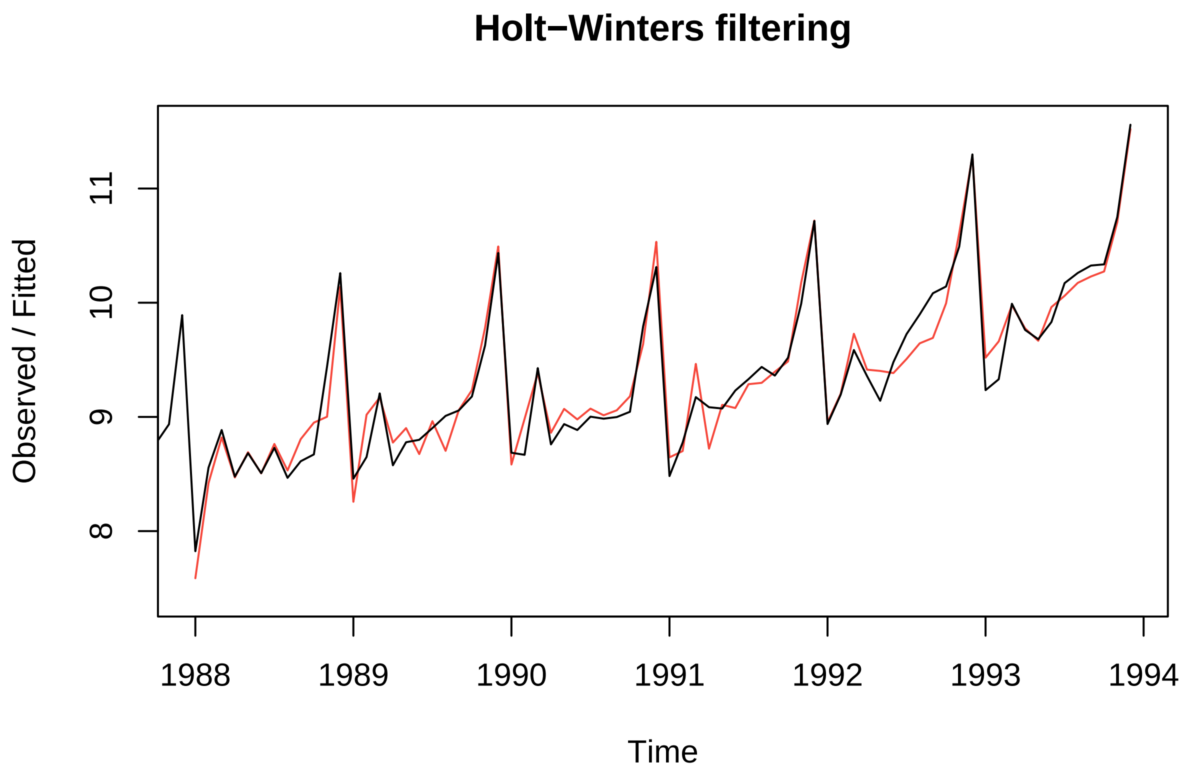
Más especificamente, el método de Holt Winters amplía el suavizado simple exponencial para permitir además el pronóstico de datos con tendencia y capturar la estacionalidad, además la ecuación estacional muestra un promedio ponderado entre el índice estacional actual y el índice estacional pero un año atrás.

El método multiplicativo es similar al aditivo. El método de multiplicativo de Holt-Winters también calcula valores suavizados simple exponencialmente para el nivel, tendencia y ajuste estacional para la previsión. Este método multiplica la previsión con tendencia por la estacionalidad, lo que produce la previsión de multiplicativo de Holt-Winters.

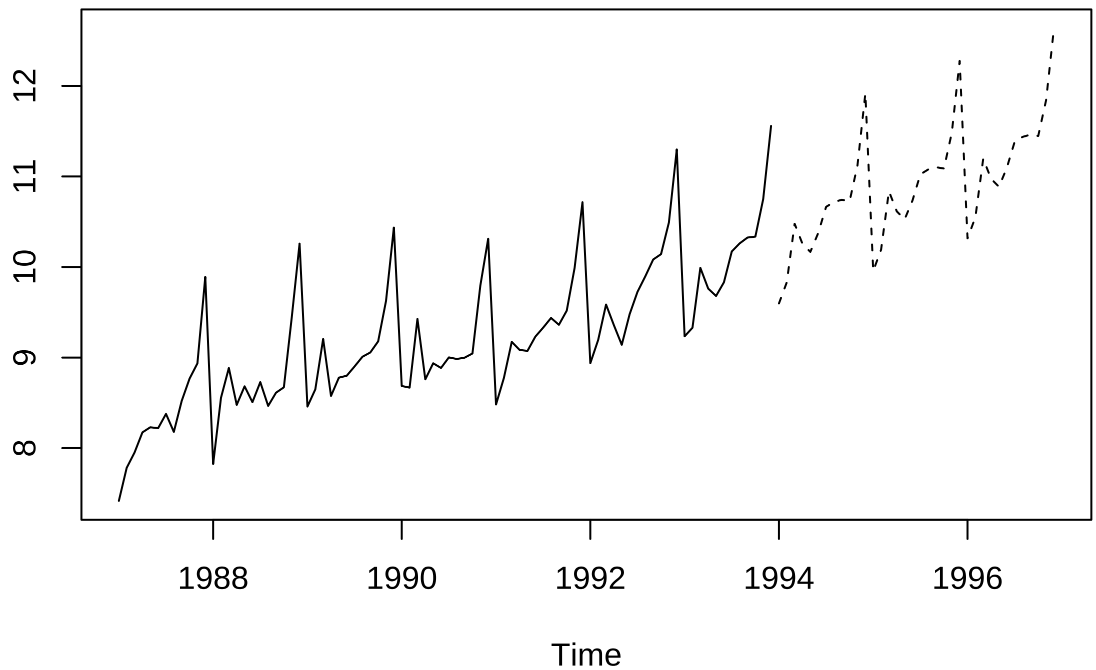
5.- Use el método de Holt Winters para el ajuste de la curva y predicción de los datos de 3 años futuros.

Esto se hizo en el 3b), con el siguiente código:

```
xt.hw = HoltWinters(Serie_ln, seasonal="additive")
plot(xt.hw)
```




```
xt.predict = predict(xt.hw, n.ahead=3*12)
ts.plot(Serie_ln, xt.predict, lty=1:3)
```



Explícitamente los valores de predicción son:

```
xt.predict
```

```
##            Jan       Feb       Mar       Apr       May       Jun       Jul
## 1994  9.597062  9.830781 10.477542 10.254867 10.167100 10.375632 10.664248
## 1995  9.956620 10.190339 10.837100 10.614425 10.526659 10.735190 11.023806
## 1996 10.316179 10.549898 11.196658 10.973983 10.886217 11.094748 11.383364
##            Aug       Sep       Oct       Nov       Dec
## 1994 10.717796 10.742782 10.728230 11.124151 11.917062
## 1995 11.077354 11.102340 11.087788 11.483709 12.276620
## 1996 11.436912 11.461898 11.447346 11.843268 12.636179
```

Pero al hacer la transformación inversa:

```
exp(xt.predict)
```

```
##            Jan       Feb       Mar       Apr       May       Jun       Jul
## 1994  14721.47  18597.48  35509.00  28420.52  26032.48  32068.58  42798.06
## 1995  21091.39  26644.54  50873.62  40717.98  37296.64  45944.53  61316.62
## 1996  30217.57  38173.53  72886.45  58336.51  53434.77  65824.57  87848.10
##            Aug       Sep       Oct       Nov       Dec
## 1994  45152.26  46294.66  45625.85  67788.72 149800.86
## 1995  64689.48  66326.19  65367.99  97120.65 214619.15
## 1996  92680.39  95025.29  93652.48 139144.40 307484.08
```