Práctica 5: Modelos de series con exógenas

Francisco Javier Mercader Martínez

COMPLETAR PRÁCTICA 5: MODELOS DE SERIES TEMPORALES CON VARIABLES EXÓGENAS

PROCESOS ESTOCÁSTICOS Y SERIES TEMPORALES

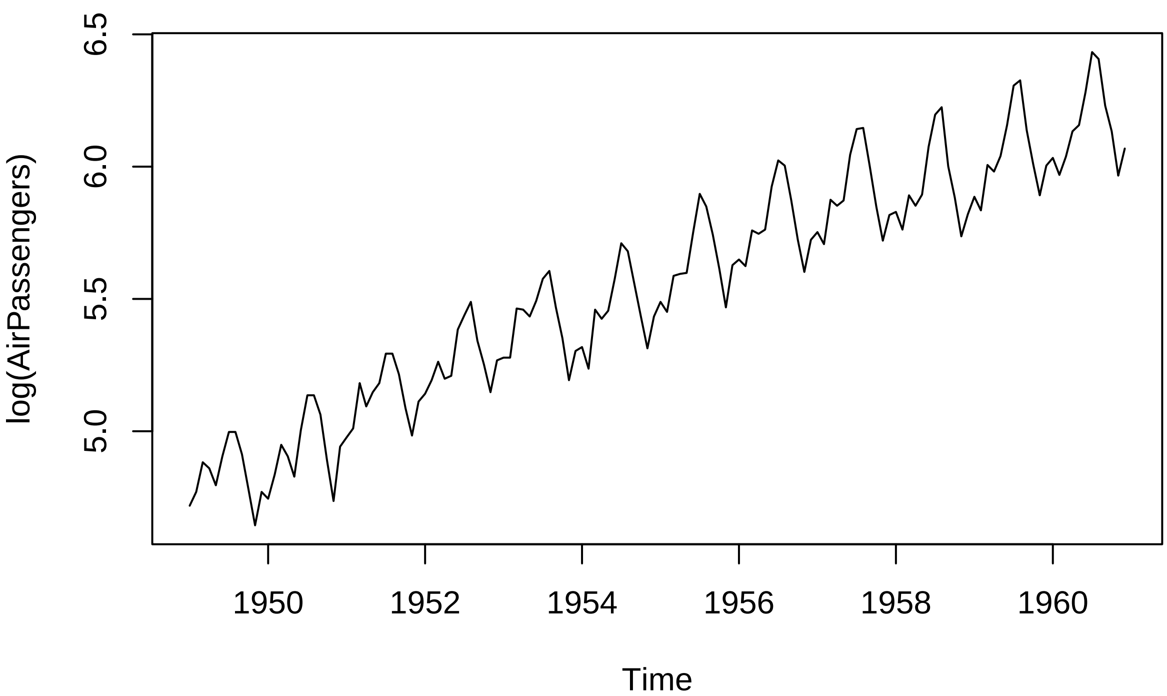
GRADO EN CIENCIA E INGENIERÍA DE DATOS

1. Modelos de Regresión para series temporales

1.1. Ajuste de un modelo de Regresión Lineal para la serie log(AirPassengers)

En esta sección trabajaremos con la serie transformada log(AirPassengers).

```
plot(log(AirPassengers))
```



Vamos a estimar un modelo de regresión lineal múltiple para explicar el comportamiento de la serie en función de los predictores (variables exógenas) "tiempo" y "dummy estacionales".

A) Forma 1: Ajuste RLM usando la función lm()

```
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
```

```
##   method            from
##   as.zoo.data.frame zoo
```

Definimos los predictores:

```
tiempo <- time(log(AirPassengers)) # predictor tiempo
estacional.dummy <- seasonaldummy(log(AirPassengers)) # predictores dummy estacionales
```

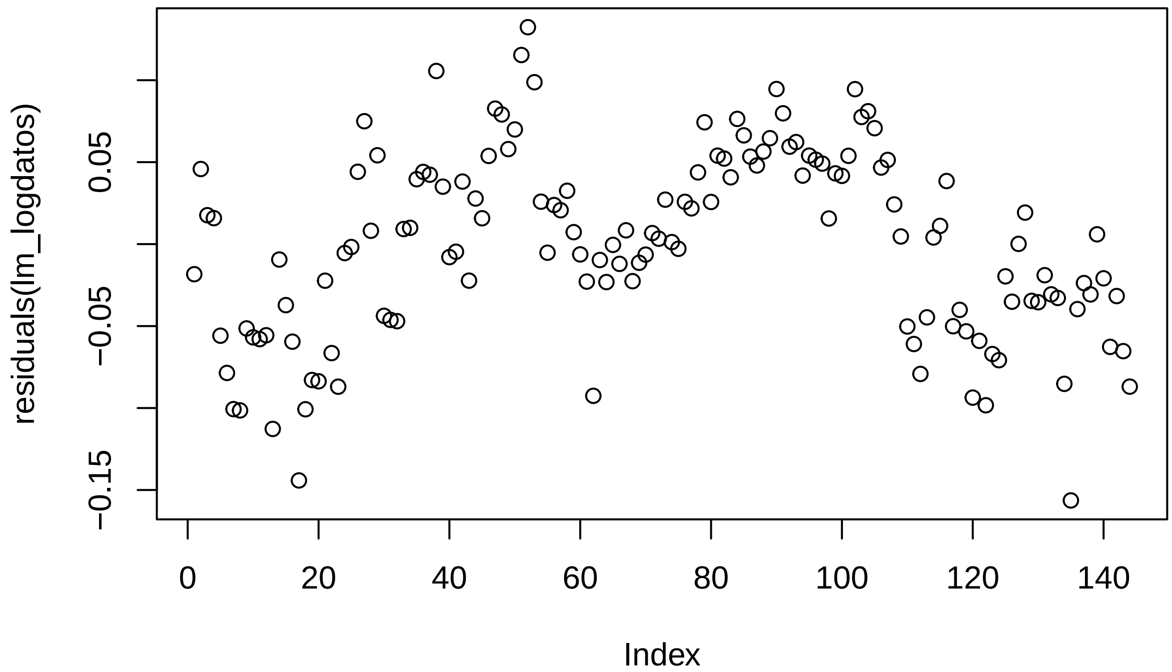
Ajustamos el modelo RLM:

```
lm_logdatos <- lm(log(AirPassengers) ~ tiempo + estacional.dummy)
summary(lm_logdatos)
```

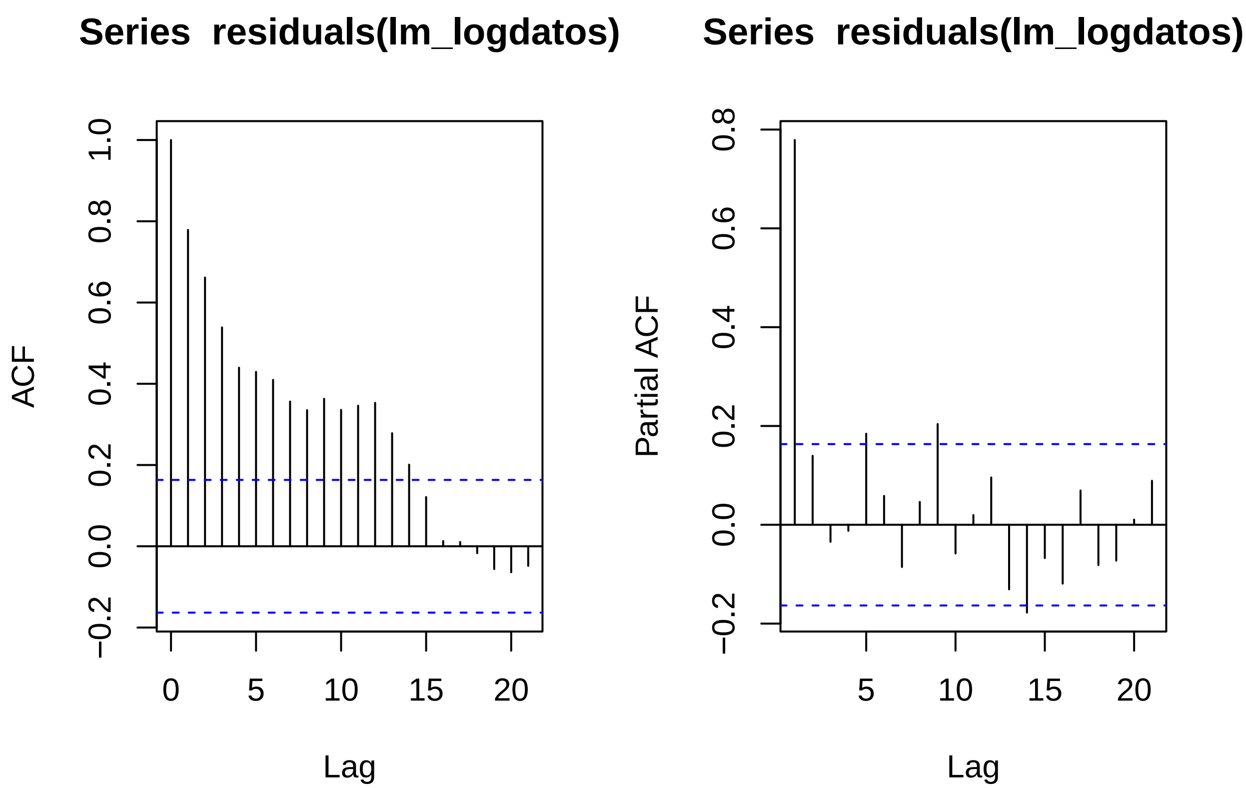
```
##
## Call:
## lm(formula = log(AirPassengers) ~ tiempo + estacional.dummy)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.156370 -0.041016  0.003677  0.044069  0.132324
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -2.308e+02  2.799e+00 -82.436  < 2e-16 ***
## tiempo               1.208e-01  1.432e-03  84.399  < 2e-16 ***
## estacional.dummyJan  2.132e-02  2.425e-02   0.879 0.380816
## estacional.dummyFeb -7.338e-04  2.424e-02  -0.030 0.975897
## estacional.dummyMar  1.295e-01  2.423e-02   5.343 3.92e-07 ***
## estacional.dummyApr  9.822e-02  2.423e-02   4.054 8.59e-05 ***
## estacional.dummyMay  9.585e-02  2.423e-02   3.957 0.000124 ***
## estacional.dummyJun  2.180e-01  2.422e-02   9.000 2.25e-15 ***
## estacional.dummyJul  3.219e-01  2.422e-02  13.293  < 2e-16 ***
## estacional.dummyAug  3.126e-01  2.422e-02  12.911  < 2e-16 ***
## estacional.dummySep  1.680e-01  2.421e-02   6.939 1.64e-10 ***
## estacional.dummyOct  2.985e-02  2.421e-02   1.233 0.219790
## estacional.dummyNov -1.139e-01  2.421e-02  -4.703 6.41e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0593 on 131 degrees of freedom
## Multiple R-squared:  0.9835, Adjusted R-squared:  0.982
## F-statistic: 649.4 on 12 and 131 DF,  p-value: < 2.2e-16
```

Comportamiento de los residuos del modelo:

```
plot(residuals(lm_logdatos)) # gráfico residuos
```




```r
par(mfrow = c(1, 2))
acf(residuals(lm_logdatos)) # correlograma simple residuos
pacf(residuals(lm_logdatos)) # correlogerama parcial resiudos
```



¿Y si usamos términos de Fourier en lugar de dummy estacionales?

Podemos crear manualmente los predictores correspondientes a las series de Fourier:

```r
# Escribir bloque de código
n <- length(log(AirPassengers))
t <- 1:n
s1 <- sin(2 * pi * t / 12)
c1 <- cos(2 + pi * t / 12)
s2 <- sin(4 * pi * t / 12)
c2 <- cos(4 * pi * t / 12)
```

Podemos crear automáticamente los predictores series de Fourier para distintos valores de K:

```r
# Para K = 6 = L/2
fourier_terms_k6 <- fourier(log(AirPassengers), K = 6)
head(fourier_terms_k6)
```

```
##          S1-12      C1-12      S2-12 C2-12 S3-12 C3-12      S4-12 C4-12
## [1,] 0.5000000  0.8660254  0.8660254   0.5     1     0  0.8660254  -0.5
## [2,] 0.8660254  0.5000000  0.8660254  -0.5     0    -1 -0.8660254  -0.5
## [3,] 1.0000000  0.0000000  0.0000000  -1.0    -1     0  0.0000000   1.0
## [4,] 0.8660254 -0.5000000 -0.8660254  -0.5     0     1  0.8660254  -0.5
## [5,] 0.5000000 -0.8660254 -0.8660254   0.5     1     0 -0.8660254  -0.5
## [6,] 0.0000000 -1.0000000  0.0000000   1.0     0    -1  0.0000000   1.0
##           S5-12      C5-12 C6-12
## [1,]  0.5000000 -0.8660254    -1
## [2,] -0.8660254  0.5000000     1
## [3,]  1.0000000  0.0000000    -1
## [4,] -0.8660254 -0.5000000     1
## [5,]  0.5000000  0.8660254    -1
## [6,]  0.0000000 -1.0000000     1
```

Ajustamos el modelo RLM:

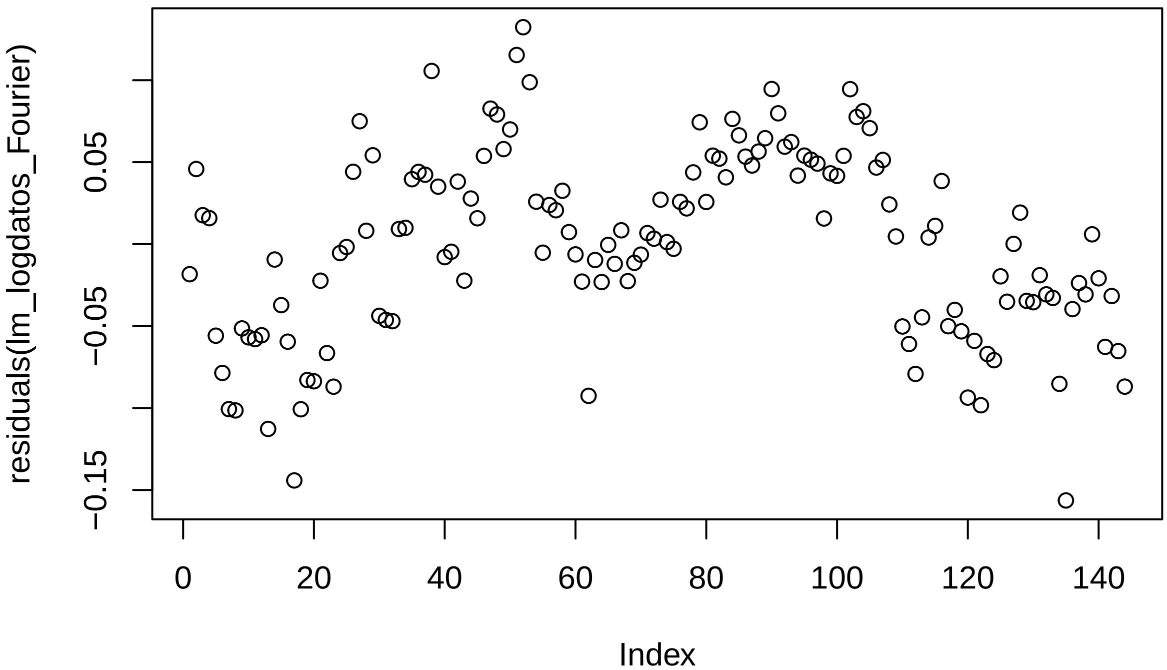
```r
lm_logdatos_Fourier <- lm(log(AirPassengers) ~ tiempo + fourier_terms_k6)
summary(lm_logdatos_Fourier)
```

```
##
## Call:
## lm(formula = log(AirPassengers) ~ tiempo + fourier_terms_k6)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.156370 -0.041016  0.003677  0.044069  0.132324
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)           -2.307e+02  2.799e+00 -82.419  < 2e-16 ***
## tiempo                 1.208e-01  1.432e-03  84.399  < 2e-16 ***
## fourier_terms_k6S1-12 -4.936e-02  7.003e-03  -7.048 9.31e-11 ***
## fourier_terms_k6C1-12 -1.418e-01  6.990e-03 -20.287  < 2e-16 ***
## fourier_terms_k6S2-12  7.868e-02  6.992e-03  11.253  < 2e-16 ***
## fourier_terms_k6C2-12 -2.281e-02  6.990e-03  -3.264 0.001403 **
## fourier_terms_k6S3-12 -8.731e-03  6.990e-03  -1.249 0.213877
## fourier_terms_k6C3-12  2.729e-02  6.990e-03   3.904 0.000150 ***
## fourier_terms_k6S4-12  2.561e-02  6.989e-03   3.664 0.000359 ***
## fourier_terms_k6C4-12  2.215e-02  6.990e-03   3.168 0.001908 **
## fourier_terms_k6S5-12  2.137e-02  6.989e-03   3.057 0.002706 **
## fourier_terms_k6C5-12  5.515e-03  6.990e-03   0.789 0.431541
```

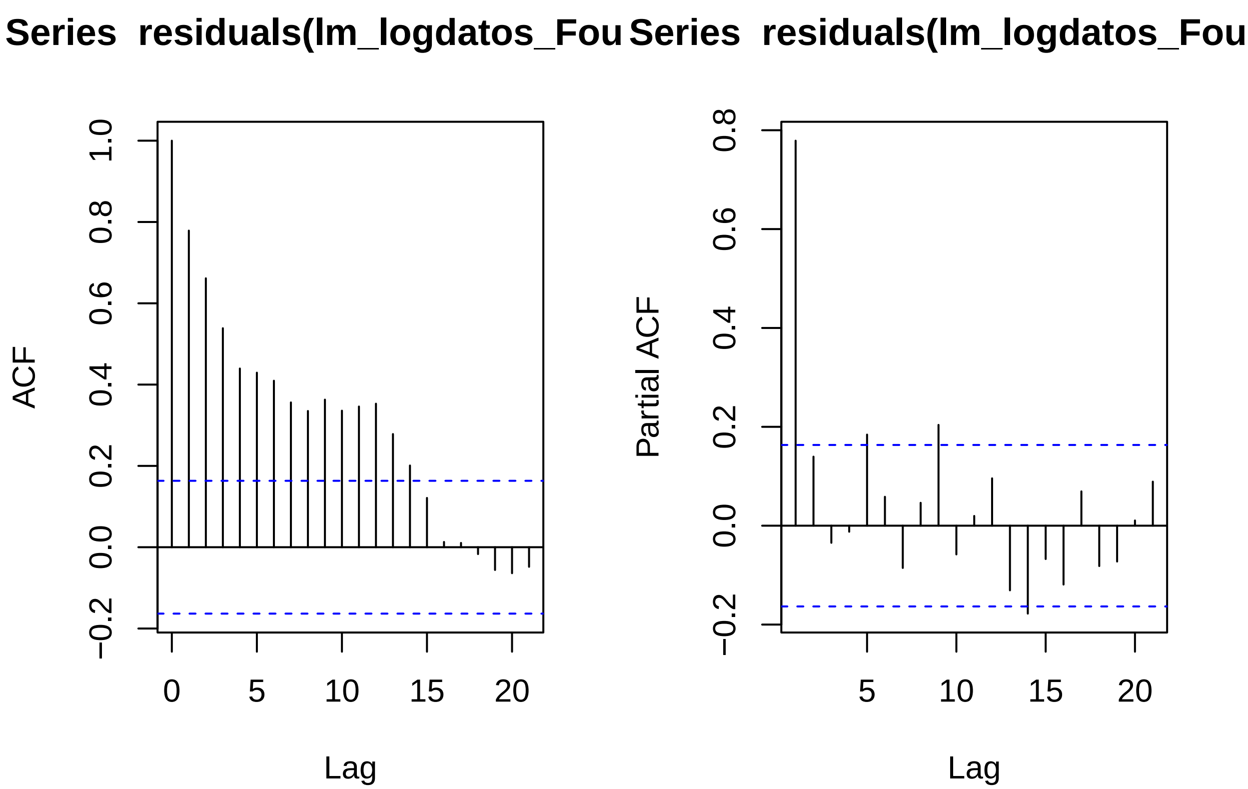
```
## fourier_terms_k6C6-12  2.936e-03  4.942e-03   0.594 0.553474
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0593 on 131 degrees of freedom
## Multiple R-squared:  0.9835, Adjusted R-squared:  0.982
## F-statistic: 649.4 on 12 and 131 DF,  p-value: < 2.2e-16
```

Comportamiento de los residuos del modelo:

```r
plot(residuals(lm_logdatos_Fourier)) # gráfico residuos
```




```r
par(mfrow = c(1, 2))
acf(residuals(lm_logdatos_Fourier)) # correlograma simple residuos
pacf(residuals(lm_logdatos_Fourier)) # correlogerama parcial resiudos
```



B) Ajuste RLM usando la función tslm()

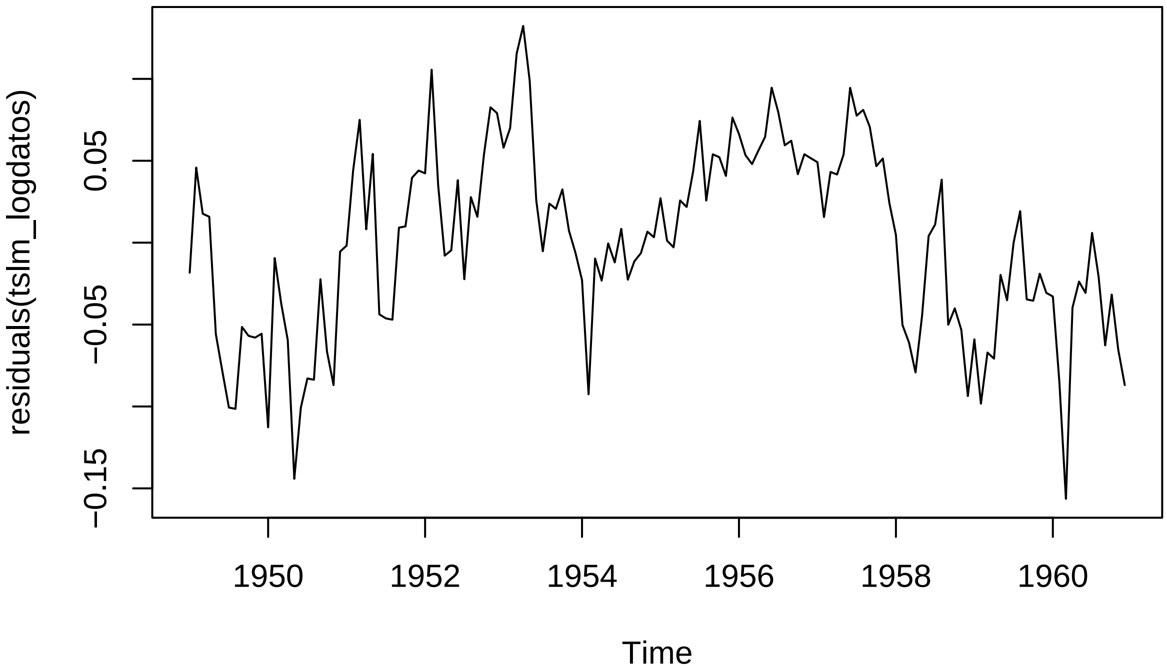
```
tslm_logdatos <- tslm(log(AirPassengers) ~ trend + season)
summary(tslm_logdatos)
```

```
##
## Call:
## tslm(formula = log(AirPassengers) ~ trend + season)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.156370 -0.041016  0.003677  0.044069  0.132324
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.7267804  0.0188935 250.180  < 2e-16 ***
## trend        0.0100688  0.0001193  84.399  < 2e-16 ***
## season2     -0.0220548  0.0242109  -0.911  0.36400
## season3      0.1081723  0.0242118   4.468 1.69e-05 ***
## season4      0.0769034  0.0242132   3.176  0.00186 **
## season5      0.0745308  0.0242153   3.078  0.00254 **
## season6      0.1966770  0.0242179   8.121 2.98e-13 ***
## season7      0.3006193  0.0242212  12.411  < 2e-16 ***
## season8      0.2913245  0.0242250  12.026  < 2e-16 ***
## season9      0.1466899  0.0242294   6.054 1.39e-08 ***
## season10     0.0085316  0.0242344   0.352  0.72537
## season11    -0.1351861  0.0242400  -5.577 1.34e-07 ***
## season12    -0.0213211  0.0242461  -0.879  0.38082
## ---
```

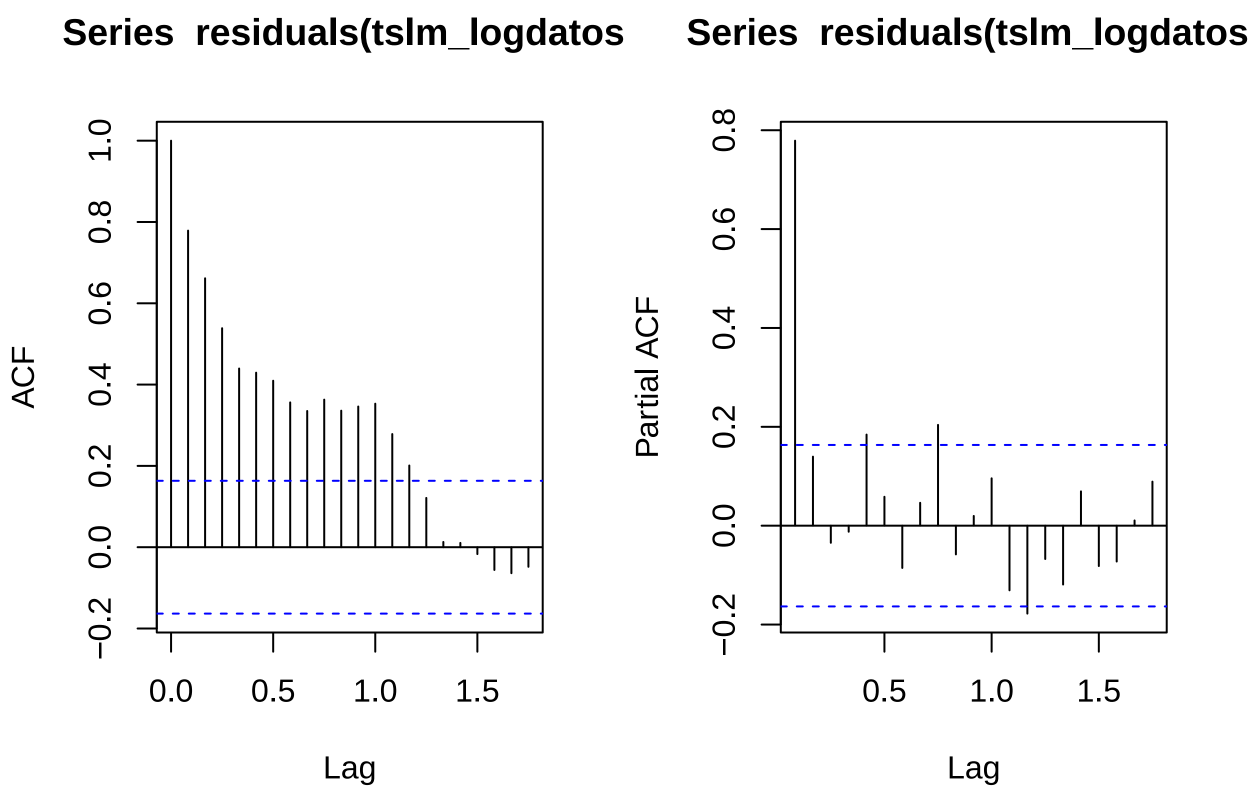
```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0593 on 131 degrees of freedom
## Multiple R-squared:  0.9835, Adjusted R-squared:  0.982
## F-statistic: 649.4 on 12 and 131 DF,  p-value: < 2.2e-16
```

Comportamiento de los residuos del modelo:

```r
plot(residuals(tslm_logdatos)) # gráfico residuos
```

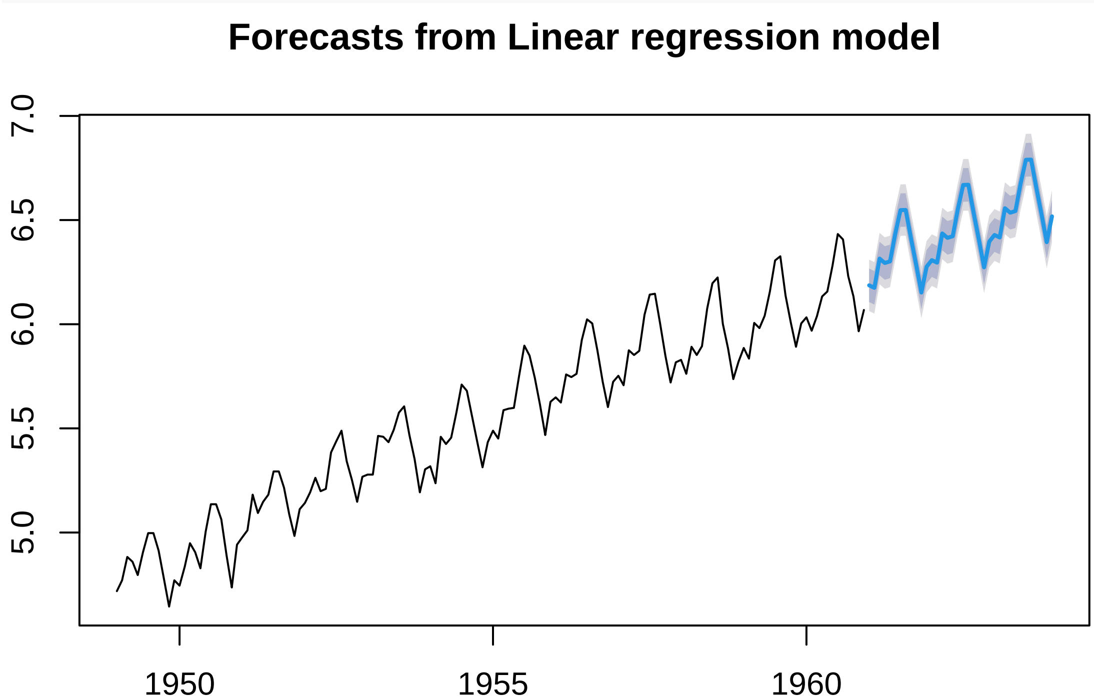



```r
par(mfrow = c(1, 2))
acf(residuals(tslm_logdatos)) # correlograma simple residuos
pacf(residuals(tslm_logdatos)) # correlogerama parcial resiudos
```



Representamos la serie original y las predicciones a 3 años vista:

```
plot(forecast(tslm_logdatos, h = 36))
```



¿Y si usamos términos de Fourier en lugar de dummy estacionales?

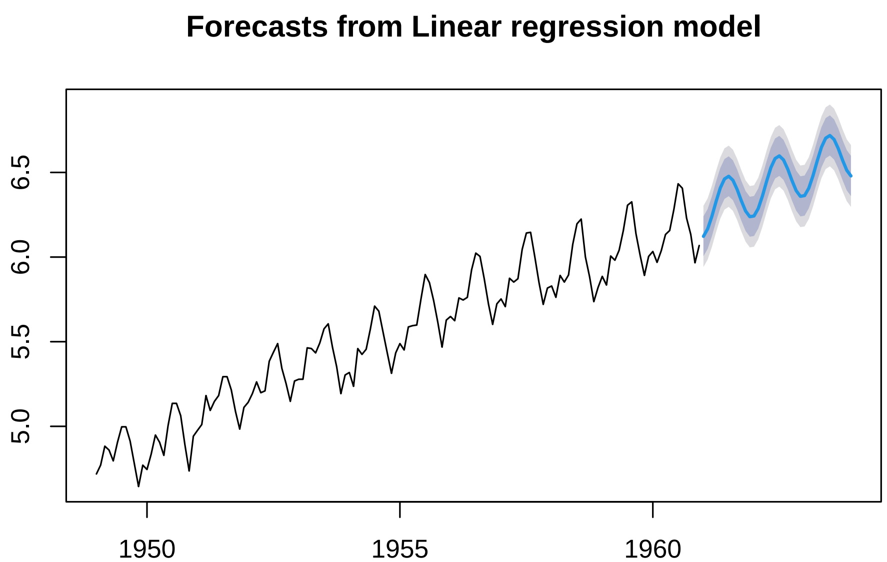
Para $K = 1$:

```r
tslm_logdatos_FourierK1 <- tslm(log(AirPassengers) ~ trend + fourier(log(AirPassengers), K = 1))

summary(tslm_logdatos_FourierK1)
```

```
##
## Call:
## tslm(formula = log(AirPassengers) ~ trend + fourier(log(AirPassengers),
##     K = 1))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.254905 -0.060940  0.004394  0.069431  0.186910
##
## Coefficients:
##                                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)                             4.8145681  0.0150203 320.537  < 2e-16
## trend                                   0.0100360  0.0001798  55.809  < 2e-16
## fourier(log(AirPassengers), K = 1)S1-12 -0.0494811  0.0105698  -4.681 6.66e-06
## fourier(log(AirPassengers), K = 1)C1-12 -0.1417735  0.0105500 -13.438  < 2e-16
##
## (Intercept)                             ***
## trend                                   ***
## fourier(log(AirPassengers), K = 1)S1-12 ***
## fourier(log(AirPassengers), K = 1)C1-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.08951 on 140 degrees of freedom
## Multiple R-squared:  0.9598, Adjusted R-squared:  0.9589
## F-statistic:  1113 on 3 and 140 DF,  p-value: < 2.2e-16
```

```r
plot(forecast(tslm_logdatos_FourierK1, newdata = data.frame(fourier(log(AirPassengers), K = 1, h = 36))
```



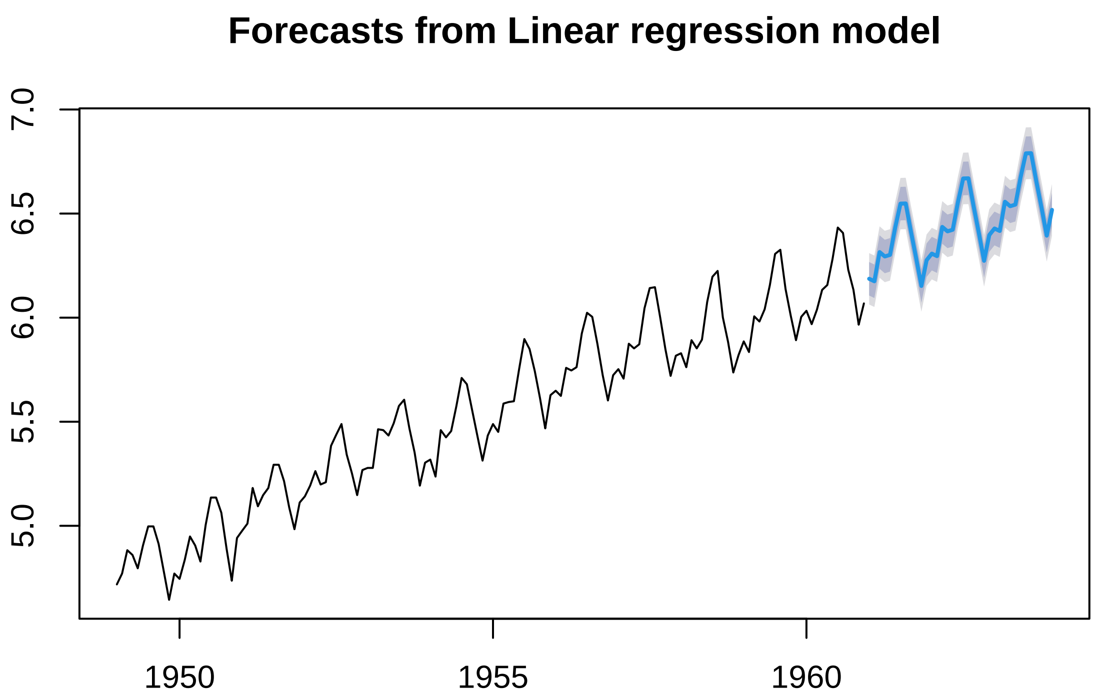
Para $K = 6$ obtenemos:

```r
# Escribir bloque de código
tslm_logdatos_FourierK6 <- tslm(log(AirPassengers) ~ trend + fourier(log(AirPassengers), K = 6))
```

```r
summary(tslm_logdatos_FourierK6)
```

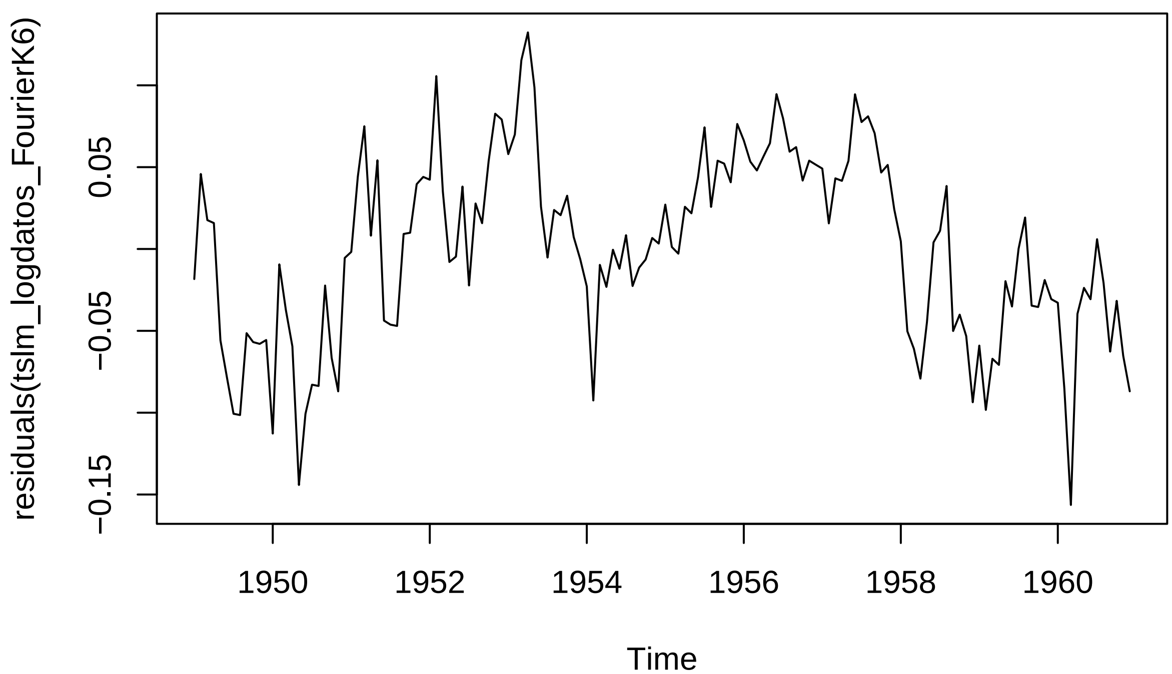
```
##
## Call:
## tslm(formula = log(AirPassengers) ~ trend + fourier(log(AirPassengers),
##     K = 6))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.156370 -0.041016  0.003677  0.044069  0.132324
##
## Coefficients:
##                                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)                             4.8121876  0.0099616 483.076  < 2e-16
## trend                                   0.0100688  0.0001193  84.399  < 2e-16
## fourier(log(AirPassengers), K = 6)S1-12 -0.0493586  0.0070032  -7.048 9.31e-11
## fourier(log(AirPassengers), K = 6)C1-12 -0.1418063  0.0069900 -20.287  < 2e-16
## fourier(log(AirPassengers), K = 6)S2-12  0.0786797  0.0069920  11.253  < 2e-16
## fourier(log(AirPassengers), K = 6)C2-12 -0.0228128  0.0069900  -3.264 0.001403
## fourier(log(AirPassengers), K = 6)S3-12 -0.0087308  0.0069900  -1.249 0.213877
## fourier(log(AirPassengers), K = 6)C3-12  0.0272922  0.0069900   3.904 0.000150
## fourier(log(AirPassengers), K = 6)S4-12  0.0256113  0.0069893   3.664 0.000359
## fourier(log(AirPassengers), K = 6)C4-12  0.0221473  0.0069900   3.168 0.001908
## fourier(log(AirPassengers), K = 6)S5-12  0.0213690  0.0069891   3.057 0.002706
## fourier(log(AirPassengers), K = 6)C5-12  0.0055151  0.0069900   0.789 0.431541
## fourier(log(AirPassengers), K = 6)C6-12  0.0029362  0.0049423   0.594 0.553474
##
## (Intercept)                             ***
## trend                                   ***
## fourier(log(AirPassengers), K = 6)S1-12 ***
## fourier(log(AirPassengers), K = 6)C1-12 ***
## fourier(log(AirPassengers), K = 6)S2-12 ***
## fourier(log(AirPassengers), K = 6)C2-12 **
## fourier(log(AirPassengers), K = 6)S3-12
## fourier(log(AirPassengers), K = 6)C3-12 ***
## fourier(log(AirPassengers), K = 6)S4-12 ***
## fourier(log(AirPassengers), K = 6)C4-12 **
## fourier(log(AirPassengers), K = 6)S5-12 **
## fourier(log(AirPassengers), K = 6)C5-12
## fourier(log(AirPassengers), K = 6)C6-12
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0593 on 131 degrees of freedom
## Multiple R-squared:  0.9835, Adjusted R-squared:  0.982
## F-statistic: 649.4 on 12 and 131 DF,  p-value: < 2.2e-16
```

```r
plot(forecast(tslm_logdatos_FourierK6, newdata = data.frame(fourier(log(AirPassengers), K = 6, h = 36))
```

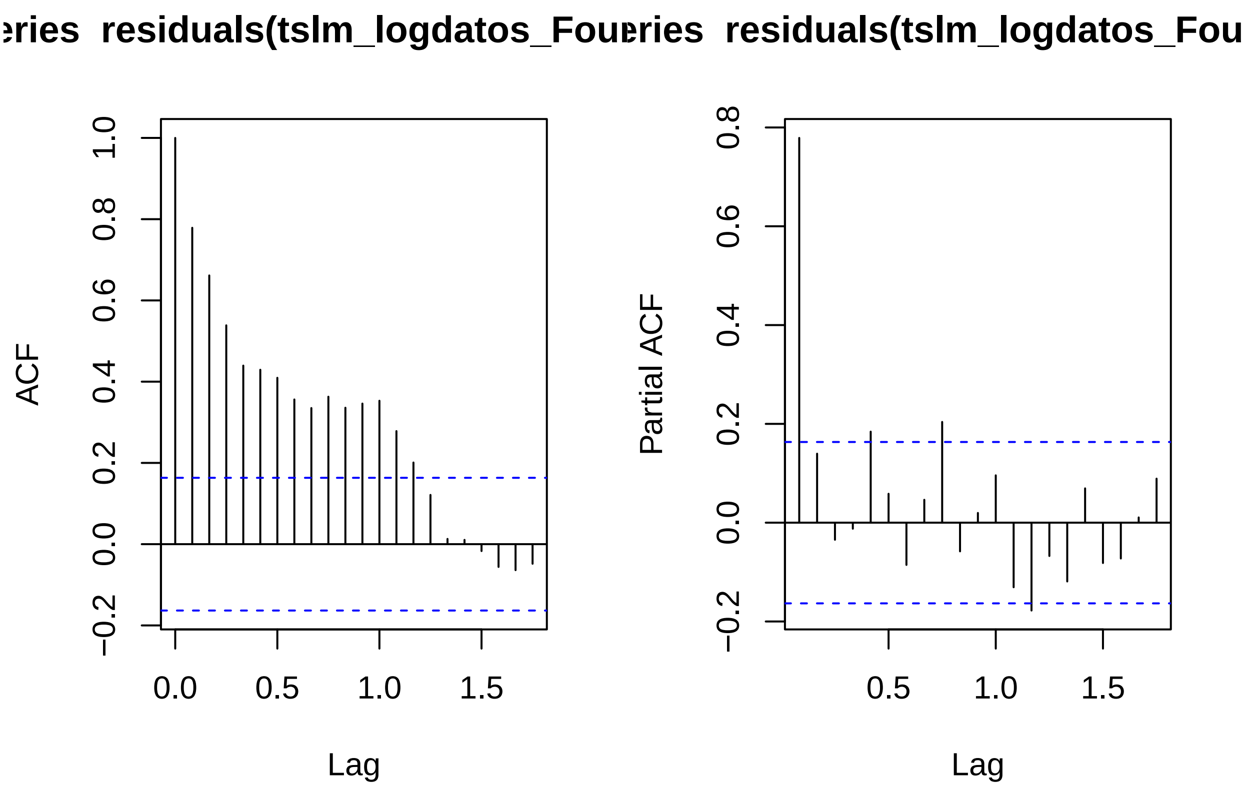


Comportamiento de los residuos del modelo:

```r
# Comportamiento de los residuos del modelo:
plot(residuals(tslm_logdatos_FourierK6)) # gráfico residuos
```




```r
par(mfrow = c(1, 2))
acf(residuals(tslm_logdatos_FourierK6)) # correlograma simple residuos
pacf(residuals(tslm_logdatos_FourierK6)) # correlograma parcial residuos
```

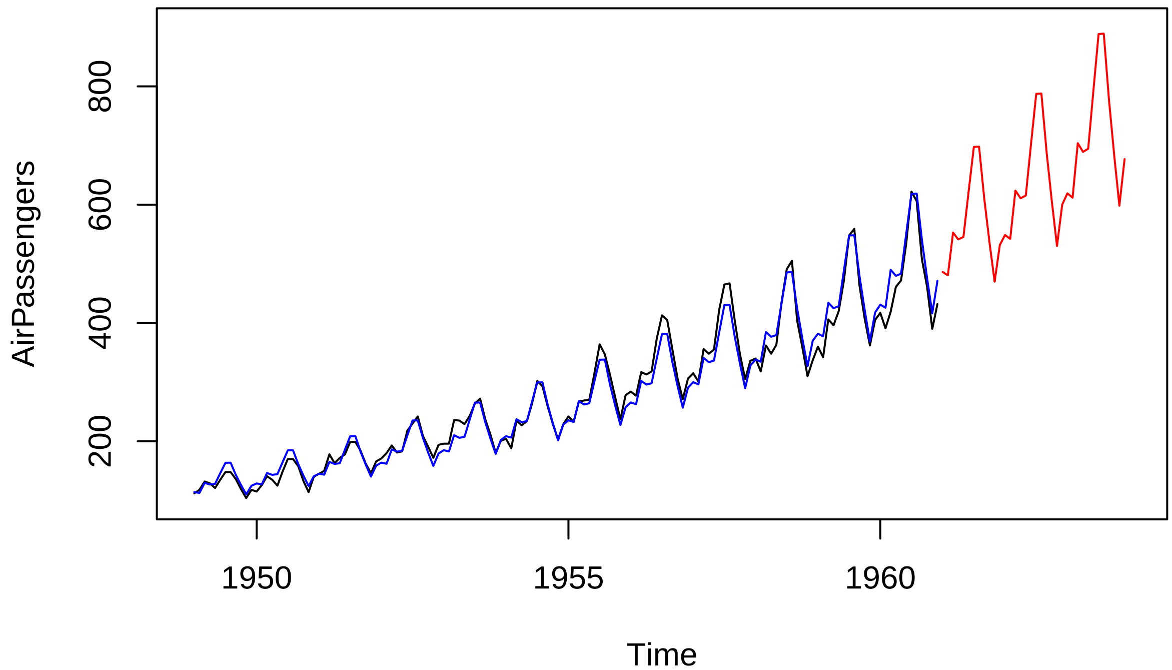


1.2. Ajuste de un modelo de Regresión (no-lineal) para la serie AirPassengers

Ahora queremos modelizar la serie AirPassengers y no su logaritmo.

ESCENARIO 1: Tomamos exponenciales sobre los resultados que obtuvimos al analizar log(AirPassengers)

```
plot(AirPassengers, xlim = c(1949, 1964), ylim = c(100, 900))
lines(exp(forecast(tslm_logdatos, h = 36)$fitted), col = "blue")
lines(exp(forecast(tslm_logdatos, h = 36)$mean), col = "red")
```



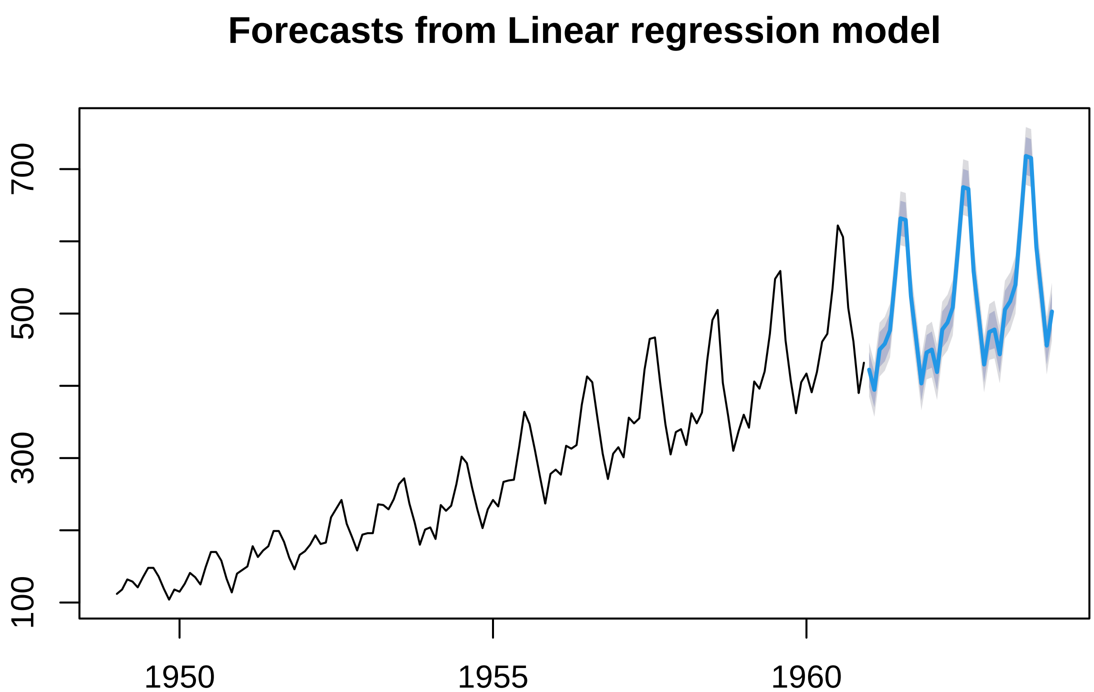
ESCENARIO 2: Ajuste de un modelo no-lineal para AirPassengers, que incluye interacciones entre tendencia y estacionalidad.

```
tslm_datos <- tslm(AirPassengers ~ trend * season)
summary(tslm_datos)
```

```
##
## Call:
## tslm(formula = AirPassengers ~ trend * season)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -36.652  -9.904   0.737   7.761  34.051
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     86.607517   8.794167   9.848  < 2e-16 ***
## trend            2.315559   0.111641  20.741  < 2e-16 ***
## season2          9.380828  12.504294   0.750   0.4546
## season3         24.328380  12.572386   1.935   0.0553 .
## season4          9.004371  12.641097   0.712   0.4777
## season5         -1.293998  12.710418  -0.102   0.9191
## season6          3.310897  12.780338   0.259   0.7960
## season7          2.164044  12.850847   0.168   0.8666
## season8          0.150058  12.921937   0.012   0.9908
## season9          2.544289  12.993596   0.196   0.8451
## season10       -13.057984  13.065816  -0.999   0.3196
## season11       -21.998543  13.138589  -1.674   0.0967 .
## season12        -9.092366  13.211903  -0.688   0.4927
## trend:season2   -0.271270   0.157884  -1.718   0.0883 .
## trend:season3   -0.007867   0.157884  -0.050   0.9603
## trend:season4    0.134033   0.157884   0.849   0.3976
## trend:season5    0.311480   0.157884   1.973   0.0508 .
```

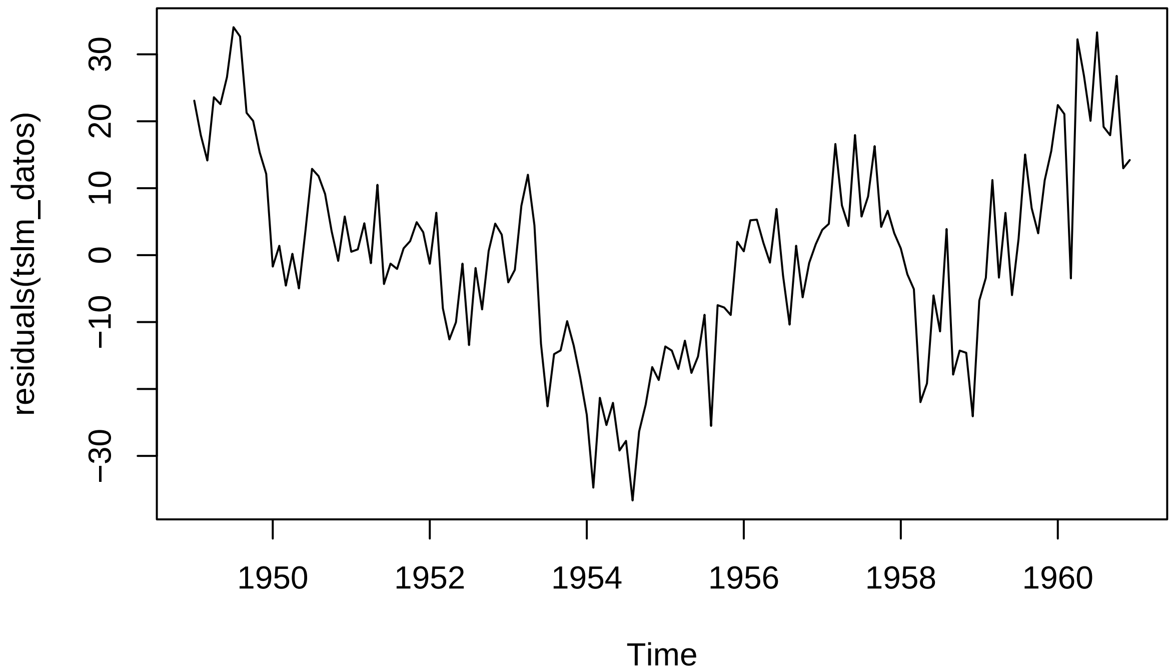
```
## trend:season6    0.764277   0.157884   4.841 3.89e-06 ***
## trend:season7    1.281177   0.157884   8.115 4.81e-13 ***
## trend:season8    1.256410   0.157884   7.958 1.10e-12 ***
## trend:season9    0.527972   0.157884   3.344   0.0011 **
## trend:season10   0.224359   0.157884   1.421   0.1579
## trend:season11  -0.130828   0.157884  -0.829   0.4090
## trend:season12   0.047494   0.157884   0.301   0.7641
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 16.02 on 120 degrees of freedom
## Multiple R-squared:  0.985,  Adjusted R-squared:  0.9822
## F-statistic: 343.4 on 23 and 120 DF,  p-value: < 2.2e-16
```

```r
plot(forecast(tslm_datos, h = 36))
```

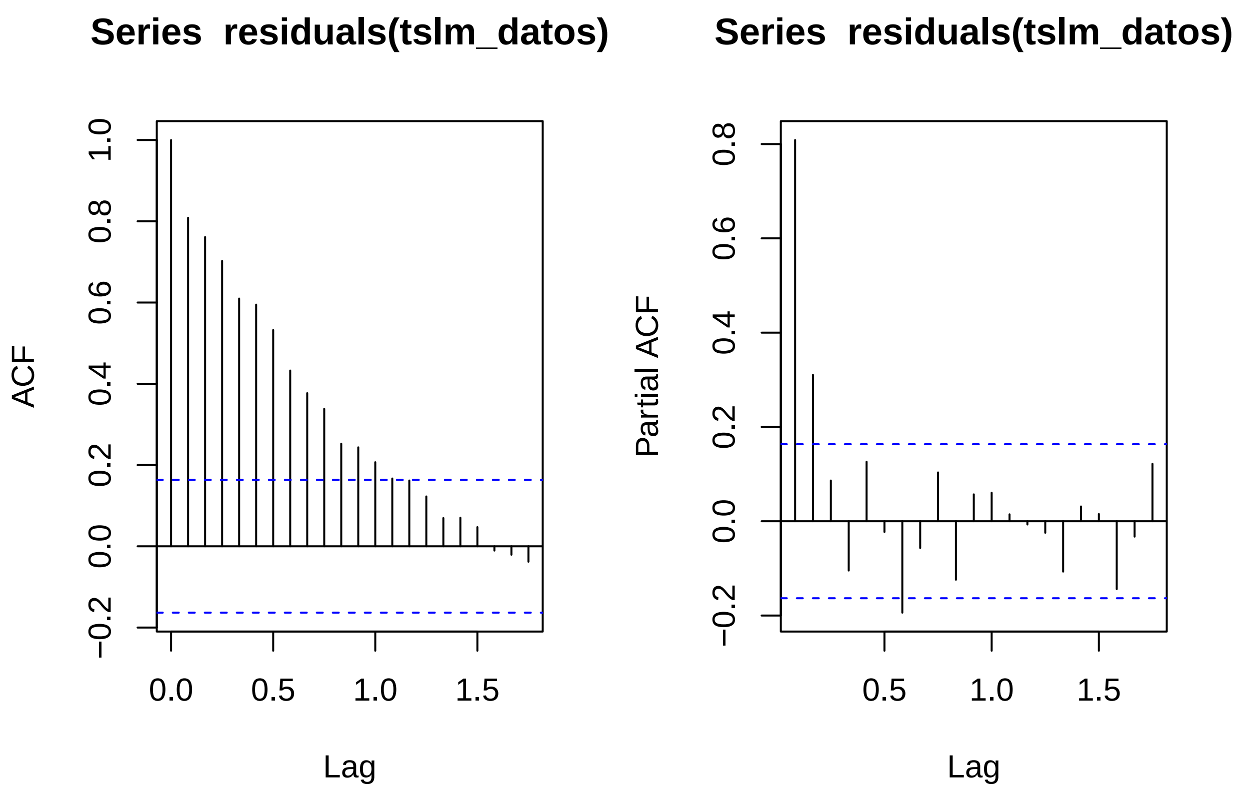


Comportamiento de los residuos del modelo:

```r
plot(residuals(tslm_datos)) # gráfico residuos
```



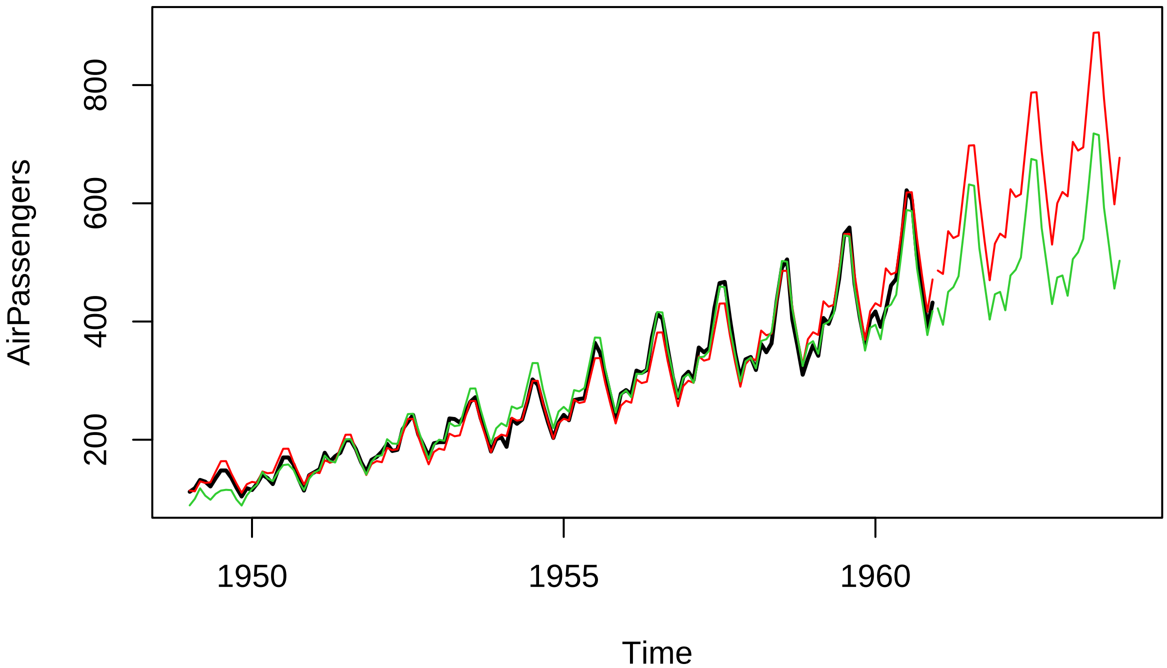

```
par(mfrow = c(1, 2))
acf(residuals(tslm_datos)) # correlograma simple residuos
pacf(residuals(tslm_datos)) # correlograma parcial residuos
```



Comparación de predicciones en los dos escenarios:

```
plot(AirPassengers, xlim = c(1949, 1964), ylim = c(100, 900), lwd = 2)
lines(exp(forecast(tslm_logdatos, h = 36)$fitted), col = "red")
```

```r
lines(exp(forecast(tslm_logdatos, h = 36)$mean), col = "red")
lines(forecast(tslm_datos, h = 36)$fitted, col = "limegreen")
lines(forecast(tslm_datos, h = 36)$mean, col = "limegreen")
```



Intervalos de predicción:

```r
pred_log <- forecast(tslm_logdatos, h = 36)
pred_datos <- forecast(tslm_datos, h = 36)
```

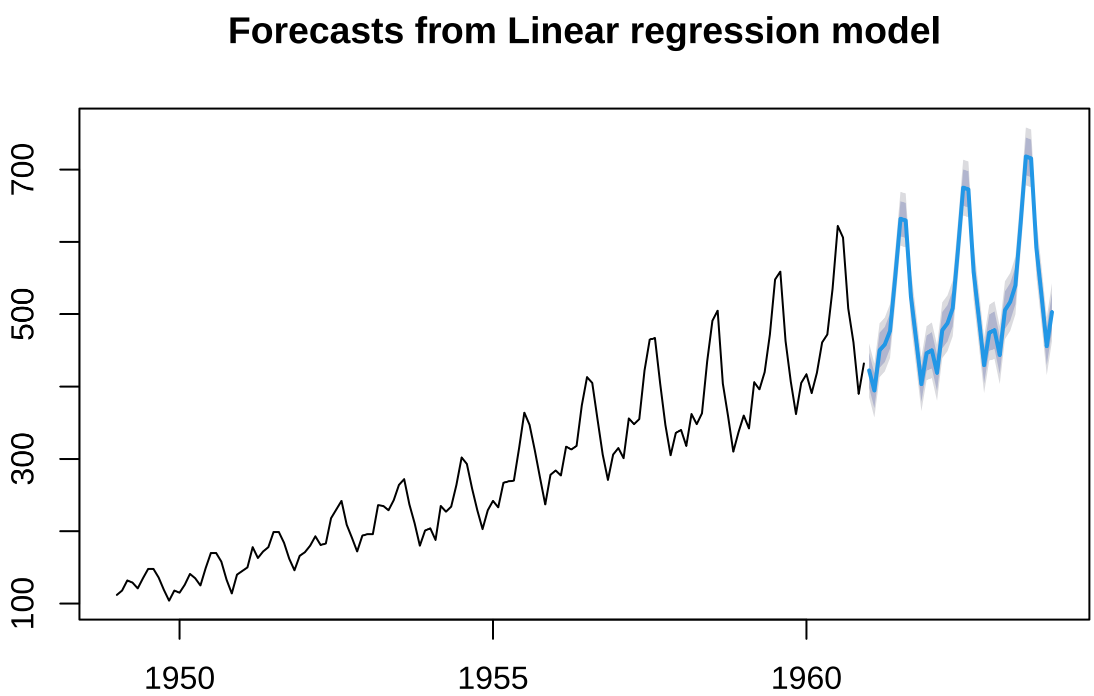
¿Y si usamos términos de Fourier en lugar de dummy estacionales?

```r
# Modelo con Fourier K=6 para AirPassengers (sin logaritmo)
tslm_datos_Fourier <- tslm(AirPassengers ~ trend * fourier(AirPassengers, K = 6))
summary(tslm_datos_Fourier)
```

```
##
## Call:
## tslm(formula = AirPassengers ~ trend * fourier(AirPassengers,
##     K = 6))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -36.652  -9.904   0.737   7.761  34.051
##
## Coefficients:
##                                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)                               87.06085    2.69334  32.325  < 2e-16
## trend                                      2.66033    0.03223  82.547  < 2e-16
## fourier(AirPassengers, K = 6)S1-12         9.69250    3.78872   2.558  0.01176
## fourier(AirPassengers, K = 6)C1-12        -6.43731    3.82908  -1.681  0.09533
## fourier(AirPassengers, K = 6)S2-12         5.63509    3.78909   1.487  0.13959
## fourier(AirPassengers, K = 6)C2-12        -7.65950    3.82872  -2.001  0.04770
## fourier(AirPassengers, K = 6)S3-12        -0.54060    3.78913  -0.143  0.88679
## fourier(AirPassengers, K = 6)C3-12         0.07139    3.82867   0.019  0.98515
```

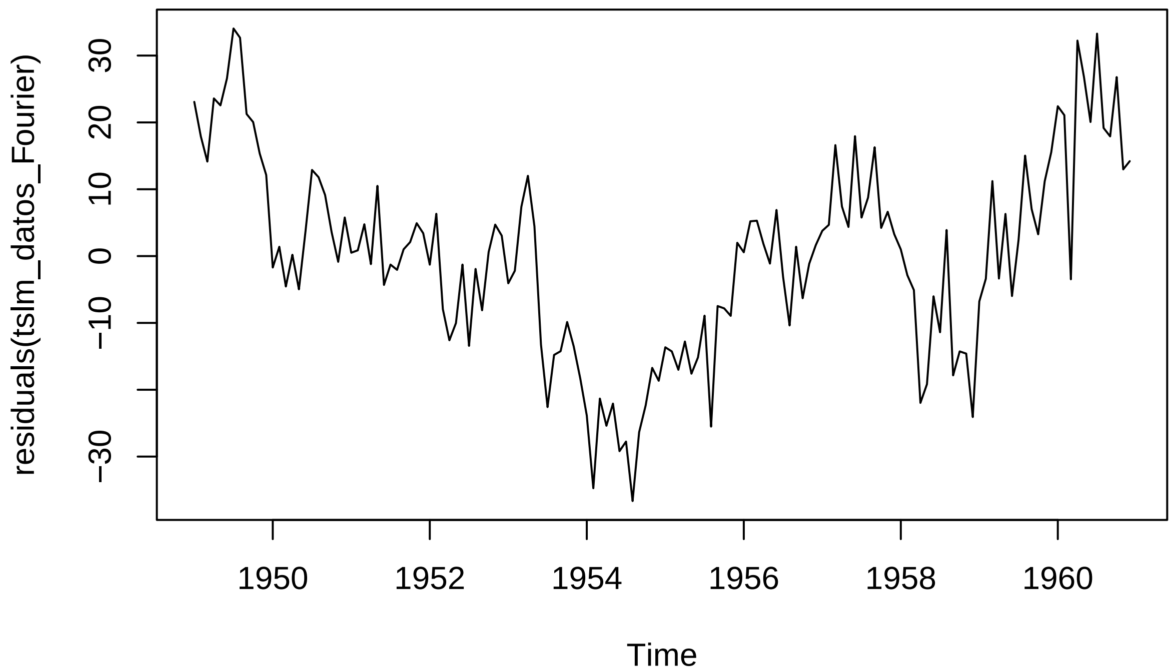
```
## fourier(AirPassengers, K = 6)S4-12        1.71359    3.78909   0.452  0.65191
## fourier(AirPassengers, K = 6)C4-12        4.81947    3.82872   1.259  0.21056
## fourier(AirPassengers, K = 6)S5-12        0.65895    3.78872   0.174  0.86222
## fourier(AirPassengers, K = 6)C5-12        0.16429    3.82908   0.043  0.96585
## fourier(AirPassengers, K = 6)C6-12       -0.50403    2.69334  -0.187  0.85187
## trend:fourier(AirPassengers, K = 6)S1-12 -0.39275    0.04558  -8.617 3.26e-14
## trend:fourier(AirPassengers, K = 6)C1-12 -0.48801    0.04558 -10.707  < 2e-16
## trend:fourier(AirPassengers, K = 6)S2-12  0.24931    0.04558   5.470 2.50e-07
## trend:fourier(AirPassengers, K = 6)C2-12  0.05847    0.04558   1.283  0.20201
## trend:fourier(AirPassengers, K = 6)S3-12 -0.05051    0.04558  -1.108  0.27003
## trend:fourier(AirPassengers, K = 6)C3-12  0.12009    0.04558   2.635  0.00952
## trend:fourier(AirPassengers, K = 6)S4-12  0.06838    0.04558   1.500  0.13614
## trend:fourier(AirPassengers, K = 6)C4-12 -0.01180    0.04558  -0.259  0.79614
## trend:fourier(AirPassengers, K = 6)S5-12  0.07433    0.04558   1.631  0.10556
## trend:fourier(AirPassengers, K = 6)C5-12  0.00952    0.04558   0.209  0.83490
## trend:fourier(AirPassengers, K = 6)C6-12  0.01445    0.03223   0.448  0.65476
##
## (Intercept)                              ***
## trend                                    ***
## fourier(AirPassengers, K = 6)S1-12       *
## fourier(AirPassengers, K = 6)C1-12       .
## fourier(AirPassengers, K = 6)S2-12
## fourier(AirPassengers, K = 6)C2-12       *
## fourier(AirPassengers, K = 6)S3-12
## fourier(AirPassengers, K = 6)C3-12
## fourier(AirPassengers, K = 6)S4-12
## fourier(AirPassengers, K = 6)C4-12
## fourier(AirPassengers, K = 6)S5-12
## fourier(AirPassengers, K = 6)C5-12
## fourier(AirPassengers, K = 6)C6-12
## trend:fourier(AirPassengers, K = 6)S1-12 ***
## trend:fourier(AirPassengers, K = 6)C1-12 ***
## trend:fourier(AirPassengers, K = 6)S2-12 ***
## trend:fourier(AirPassengers, K = 6)C2-12
## trend:fourier(AirPassengers, K = 6)S3-12
## trend:fourier(AirPassengers, K = 6)C3-12 **
## trend:fourier(AirPassengers, K = 6)S4-12
## trend:fourier(AirPassengers, K = 6)C4-12
## trend:fourier(AirPassengers, K = 6)S5-12
## trend:fourier(AirPassengers, K = 6)C5-12
## trend:fourier(AirPassengers, K = 6)C6-12
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 16.02 on 120 degrees of freedom
## Multiple R-squared:  0.985,  Adjusted R-squared:  0.9822
## F-statistic: 343.4 on 23 and 120 DF,  p-value: < 2.2e-16
```

```r
# Predicción
fourier_futuro <- fourier(AirPassengers, K = 6, h = 36)
plot(forecast(tslm_datos_Fourier, newdata = data.frame(fourier_futuro)))
```

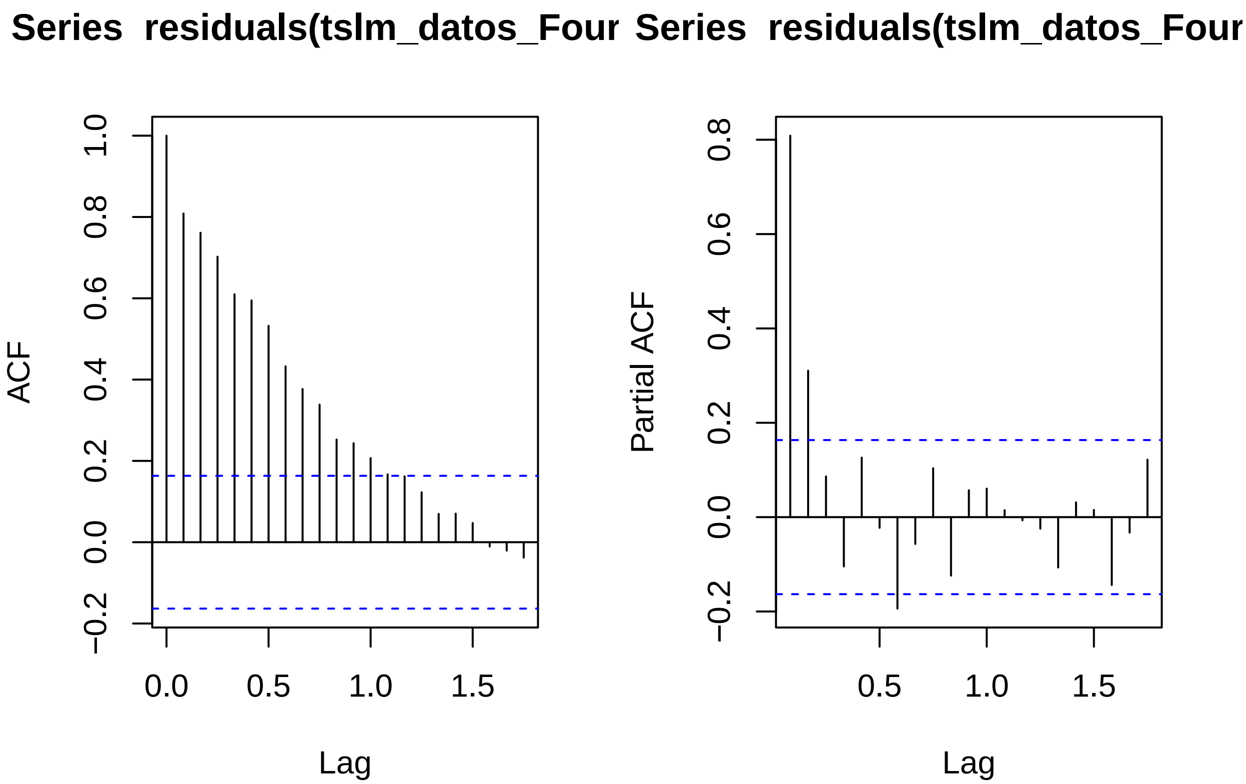


Comportamiento de los residuos

```r
plot(residuals(tslm_datos_Fourier)) # gráfico residuos
```




```r
par(mfrow = c(1, 2))
acf(residuals(tslm_datos_Fourier)) # correlograma simple residuos
pacf(residuals(tslm_datos_Fourier)) # correlograma parcial residuos
```



2. Modelos de Regresión Dinámica para series temporales

Forma 1: Ajuste de regresión y ARIMA conjunto.

Forma 2: Ajuste de regresión y ajuste ARIMA de los residuos por separado.

2.1. Ajuste de regresión y ARIMA conjunto.

COMPLETAR

```
# Crear variables exógenas: tendencia y términos de Fourier
xreg_train <- cbind(
  trend = 1:length(log(AirPassengers)),
  fourier(log(AirPassengers), K = 6)
)

# Ajuste conjunto regresión + ARIMA
modelo_dinamico <- auto.arima(log(AirPassengers), xreg = xreg_train)
summary(modelo_dinamico)
```

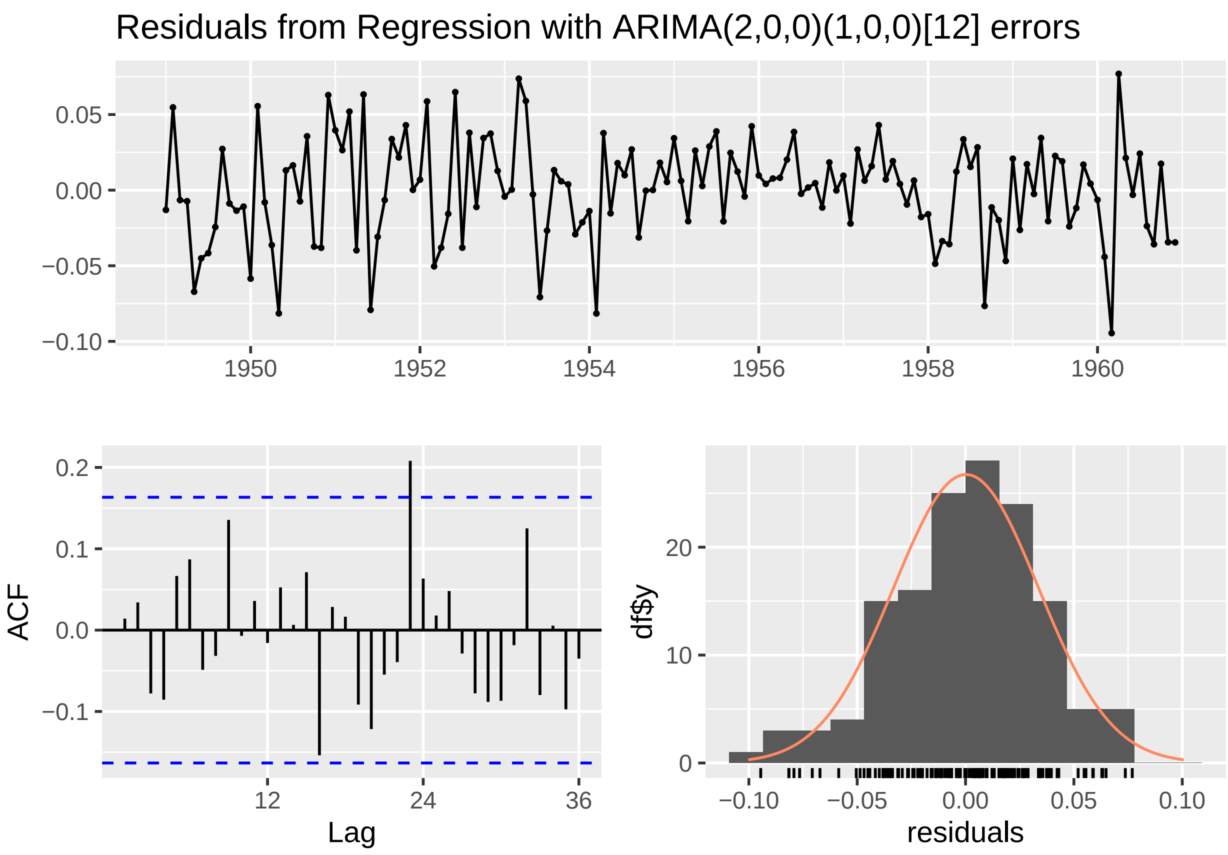
```
## Series: log(AirPassengers)
## Regression with ARIMA(2,0,0)(1,0,0)[12] errors
##
## Coefficients:
##          ar1     ar2    sar1  intercept   trend    S1-12    C1-12   S2-12
##       0.6281  0.1781  0.2768     4.8197  0.0099  -0.0515  -0.1416  0.0773
## s.e.  0.0831  0.0830  0.0876     0.0344  0.0004   0.0089   0.0088  0.0051
##         C2-12   S3-12   C3-12   S4-12   C4-12   S5-12   C5-12   C6-12
##       -0.0223  -0.009  0.0278  0.0256  0.0220  0.0206  0.0062  0.0035
## s.e.   0.0051   0.004  0.0040  0.0036  0.0036  0.0036  0.0036  0.0026
```

```
##
## sigma^2 = 0.001253:  log likelihood = 284.27
## AIC=-534.54   AICc=-529.69   BIC=-484.06
##
## Training set error measures:
##                        ME       RMSE        MAE        MPE      MAPE      MASE
## Training set 0.0002146134 0.03337733 0.02621857 0.00022698 0.4816838 0.2166072
##                    ACF1
## Training set 0.01412906
```

¿El modelo es válido? Análisis de los residuos

COMPLETAR

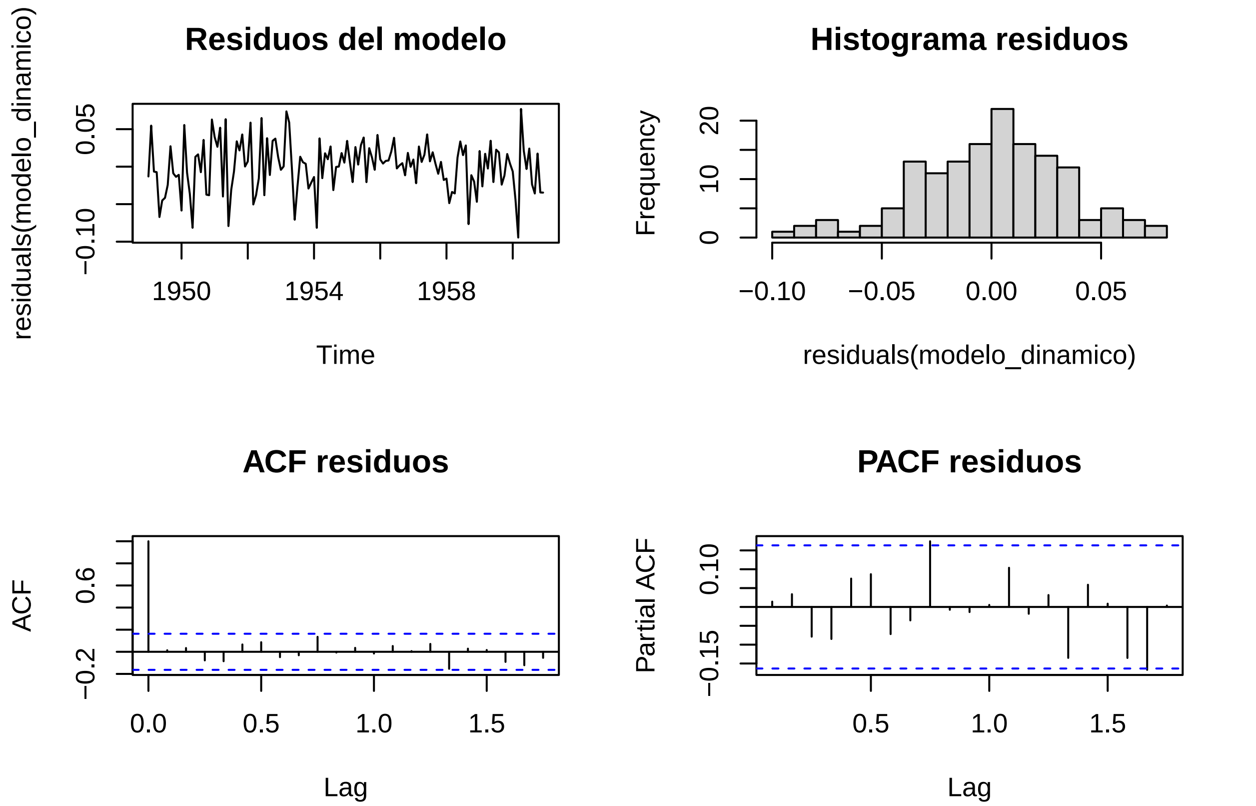
```
# Análisis completo de residuos
checkresiduals(modelo_dinamico)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Regression with ARIMA(2,0,0)(1,0,0)[12] errors
## Q* = 25.922, df = 21, p-value = 0.2094
##
## Model df: 3.   Total lags used: 24
```

```
# Graficos adicionales
par(mfrow = c(2, 2))
plot(residuals(modelo_dinamico), main = "Residuos del modelo")
hist(residuals(modelo_dinamico), main = "Histograma residuos", breaks = 20)
```

```r
acf(residuals(modelo_dinamico), main = "ACF residuos")
pacf(residuals(modelo_dinamico), main = "PACF residuos")
```




```r
# Test de Ljung-Box
Box.test(residuals(modelo_dinamico), lag = 24, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  residuals(modelo_dinamico)
## X-squared = 25.922, df = 24, p-value = 0.3571
```

Predicciones de la serie estacionaria y de la serie original

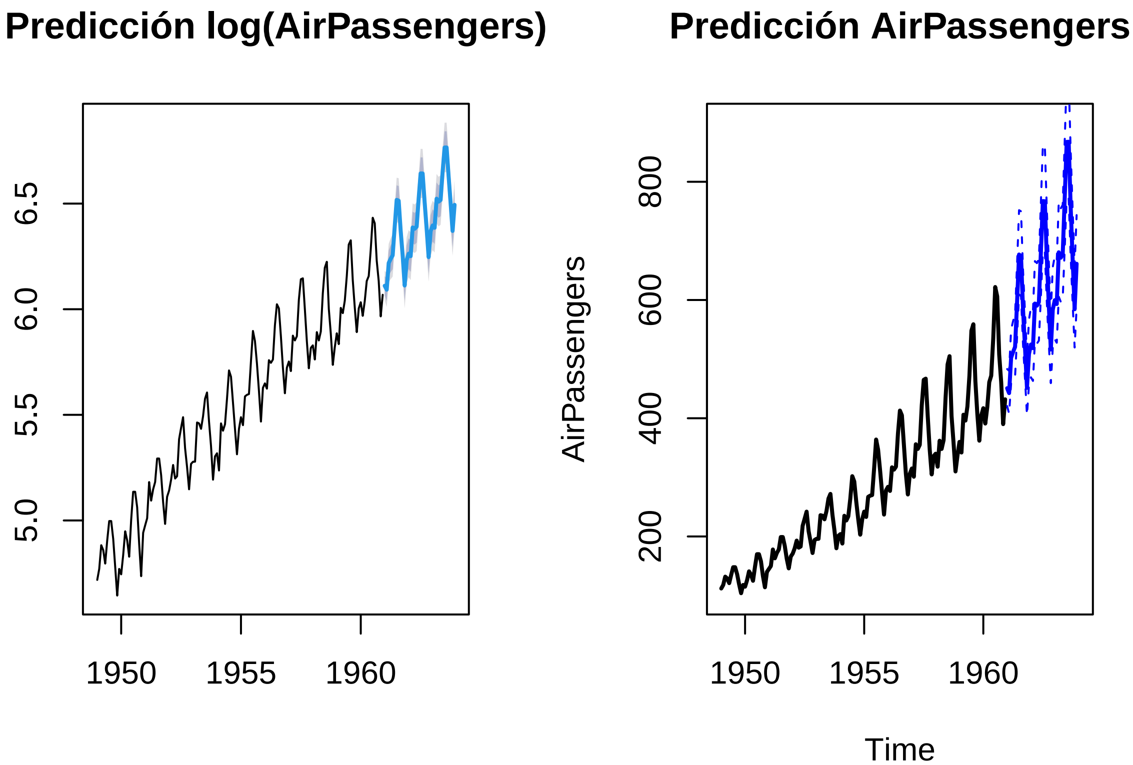
COMPLETAR

```r
# Crear variables exógenas para predicción
n <- length(log(AirPassengers))
xreg_futuro <- cbind(
  trend = (n + 1):(n + 36),
  fourier(log(AirPassengers), K = 6, h = 36)
)

# Predicciones en escala logarítmica
pred_dinamico <- forecast(modelo_dinamico, xreg = xreg_futuro, h = 36)

par(mfrow = c(1, 2))
# Serie log(AirPassengers)
plot(pred_dinamico, main = "Predicción log(AirPassengers)")
```

```r
# Serie original AirPassengers
plot(AirPassengers,
  xlim = c(1949, 1964), ylim = c(100, 900),
  main = "Predicción AirPassengers", lwd = 2
)
lines(exp(pred_dinamico$mean), col = "blue", lwd = 2)
lines(exp(pred_dinamico$lower[, 2]), col = "blue", lty = 2)
lines(exp(pred_dinamico$upper[, 2]), col = "blue", lty = 2)
```



2.2. Ajuste de regresión y ajuste ARIMA por separado

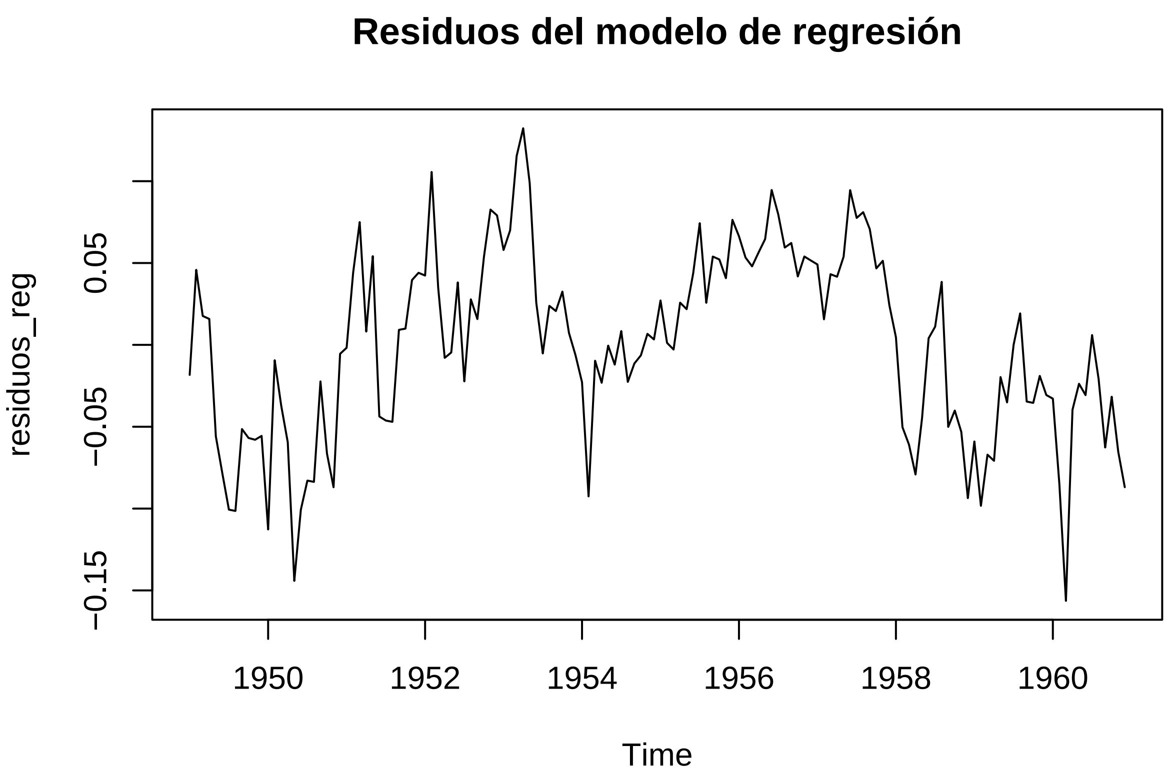
COMPLETAR

```r
# Paso 1: Ajustar modelo de regresión
reg_modelo <- tslm(log(AirPassengers) ~ trend + season)
summary(reg_modelo)
```

```
##
## Call:
## tslm(formula = log(AirPassengers) ~ trend + season)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.156370 -0.041016  0.003677  0.044069  0.132324
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.7267804  0.0188935 250.180  < 2e-16 ***
## trend        0.0100688  0.0001193  84.399  < 2e-16 ***
## season2     -0.0220548  0.0242109  -0.911  0.36400
```

```
## season3       0.1081723  0.0242118   4.468 1.69e-05 ***
## season4       0.0769034  0.0242132   3.176  0.00186 **
## season5       0.0745308  0.0242153   3.078  0.00254 **
## season6       0.1966770  0.0242179   8.121 2.98e-13 ***
## season7       0.3006193  0.0242212  12.411  < 2e-16 ***
## season8       0.2913245  0.0242250  12.026  < 2e-16 ***
## season9       0.1466899  0.0242294   6.054 1.39e-08 ***
## season10      0.0085316  0.0242344   0.352  0.72537
## season11     -0.1351861  0.0242400  -5.577 1.34e-07 ***
## season12     -0.0213211  0.0242461  -0.879  0.38082
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0593 on 131 degrees of freedom
## Multiple R-squared:  0.9835, Adjusted R-squared:  0.982
## F-statistic: 649.4 on 12 and 131 DF,  p-value: < 2.2e-16
```

```r
# Paso 2: Obtener residuos de la regresión
residuos_reg <- residuals(reg_modelo)
plot(residuos_reg, main = "Residuos del modelo de regresión")
```

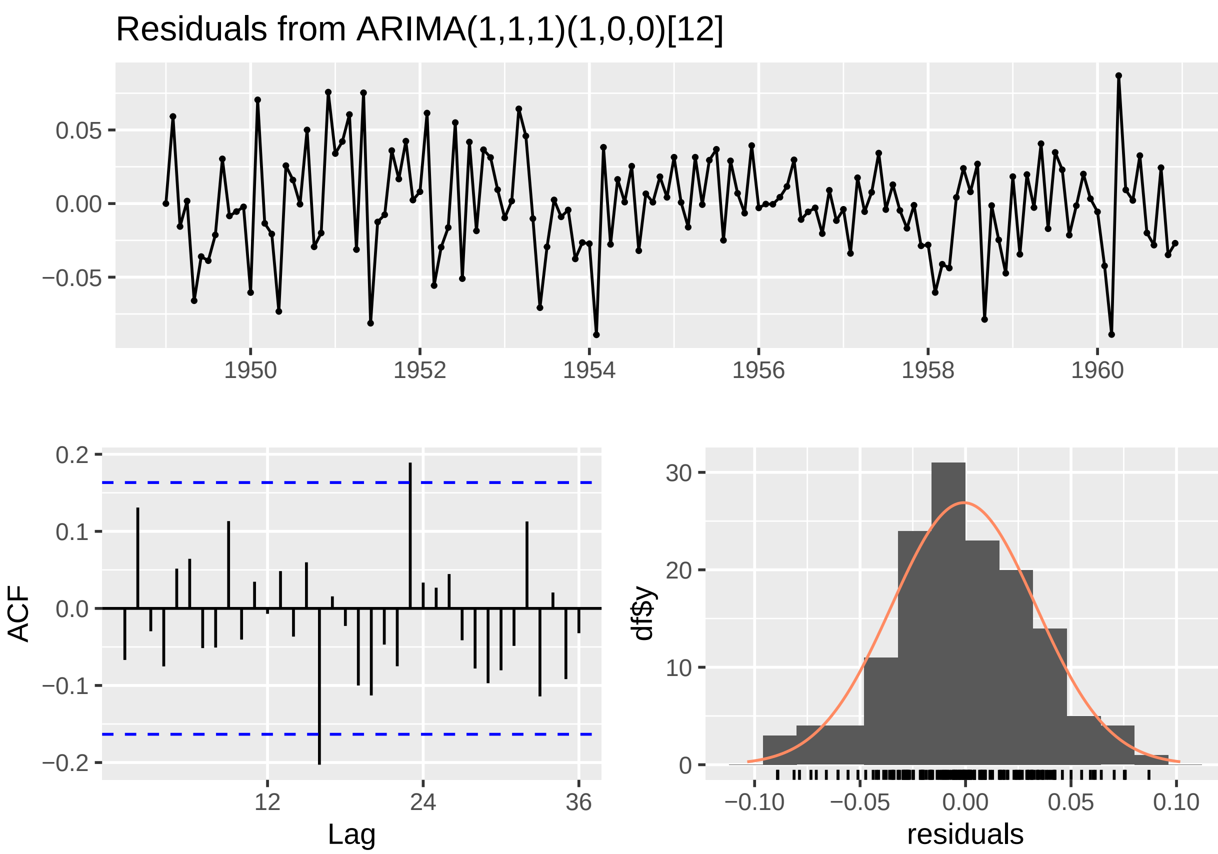



```r
# Paso 3: Ajustar modelo ARIMA a los residuos
arima_residuos <- auto.arima(residuos_reg)
summary(arima_residuos)
```

```
## Series: residuos_reg
## ARIMA(1,1,1)(1,0,0)[12]
##
## Coefficients:
##          ar1     ma1    sar1
##       0.6063  -0.877  0.2266
```

```
## s.e.  0.2054   0.146  0.0871
##
## sigma^2 = 0.001196:  log likelihood = 279.22
## AIC=-550.44   AICc=-550.15   BIC=-538.59
##
## Training set error measures:
##                         ME       RMSE        MAE      MPE    MAPE   MASE
## Training set -0.0008472376 0.03410461 0.02610686 156.5361 323.236 0.5456
##                     ACF1
## Training set -0.06686943
```

```r
# Verificar residuos del ARIMA
checkresiduals(arima_residuos)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,1,1)(1,0,0)[12]
## Q* = 28.133, df = 21, p-value = 0.1364
##
## Model df: 3.   Total lags used: 24
```

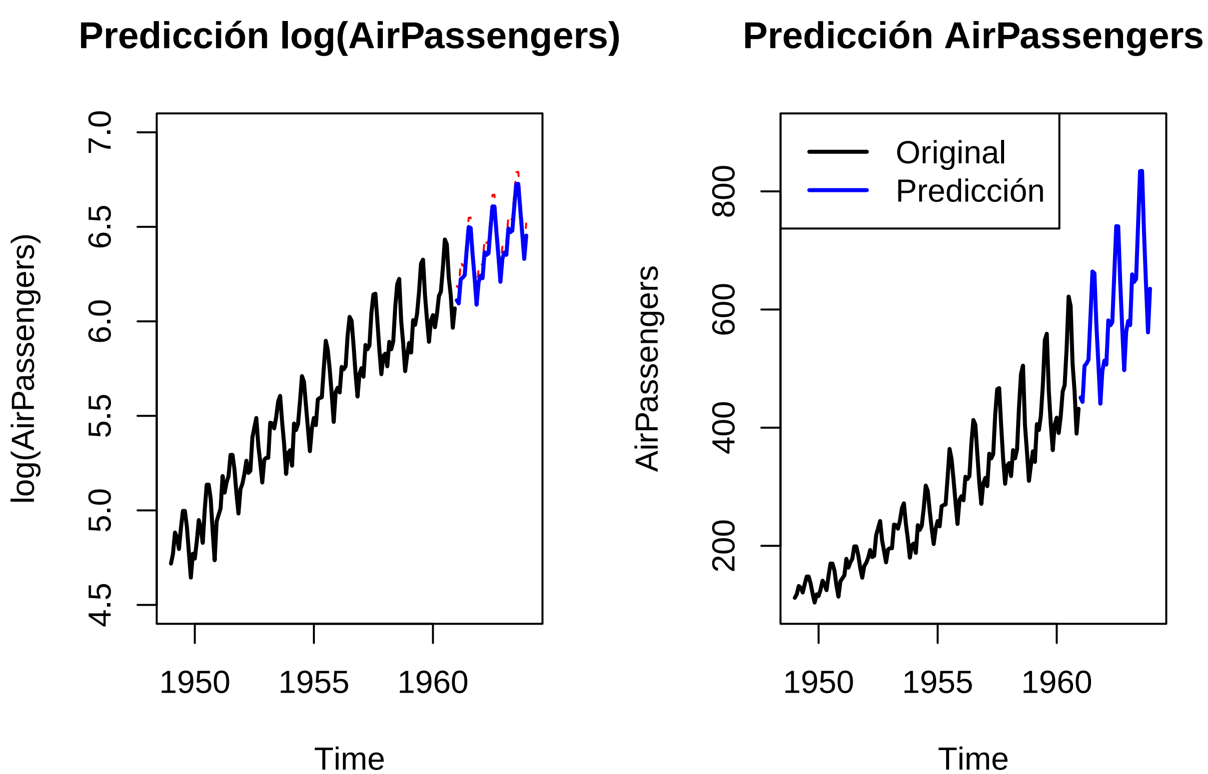
```r
# Paso 4: Predicciones combinadas
# Predicción de la parte de regresión
pred_reg <- forecast(reg_modelo, h = 36)

# Predicción de la parte ARIMA (residuos)
pred_arima_res <- forecast(arima_residuos, h = 36)
```

```r
# Predicción combinada = regresión + ARIMA residuos
pred_combinada <- pred_reg$mean + pred_arima_res$mean

# Visualización
par(mfrow = c(1, 2))
plot(log(AirPassengers),
  xlim = c(1949, 1964), ylim = c(4.5, 7),
  main = "Predicción log(AirPassengers)", lwd = 2
)
lines(pred_reg$mean, col = "red", lty = 2) # Solo regresión
lines(pred_combinada, col = "blue", lwd = 2) # Regresión + ARIMA

plot(AirPassengers,
  xlim = c(1949, 1964), ylim = c(100, 900),
  main = "Predicción AirPassengers", lwd = 2
)
lines(exp(pred_combinada), col = "blue", lwd = 2)
legend("topleft",
  legend = c("Original", "Predicción"),
  col = c("black", "blue"), lwd = 2
)
```




```r
# Comparación de ambos métodos
par(mfrow = c(1, 1))
plot(AirPassengers,
  xlim = c(1949, 1964), ylim = c(100, 900),
  main = "Comparación de métodos de Regresión Dinámica", lwd = 2
)
lines(exp(pred_dinamico$mean), col = "red", lwd = 2) # Método conjunto
```

```r
lines(exp(pred_combinada), col = "blue", lwd = 2) # Método separado
legend("topleft",
  legend = c("Original", "Conjunto (auto.arima+xreg)", "Separado (tslm+ARIMA)"),
  col = c("black", "red", "blue"), lwd = 2
)
```