Procesos Estocásticos y Series Temporales

Práctica 5: Series con variables exógenas

Francisco Javier Mercader Martínez

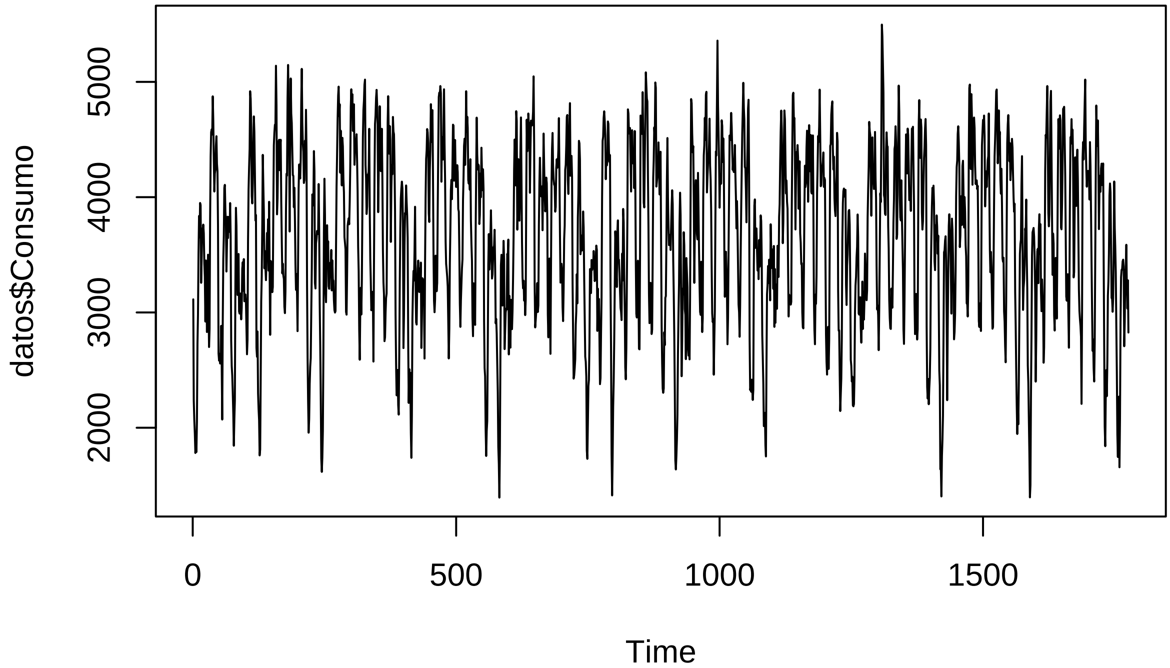
Problema 1

En el fichero **Consumo_Electrico_Sim1.csv**, se encuentran los datos horarios del consumo eléctrico (en kW) de los habitantes de una población (datos simulados), entre el 01/01/2009 y el 15/03/2009. También se muestran los registros de Temperatura ambiente (en ºKelvin), Humedad relativa, variables dummy del día de la semana (WH2 a WH7) y la variable indicadora de festividad FH1.

El objetivo es modelizar la serie de consumo eléctrico horario usando como predictores las variables climáticas, las dummy calendario proporcionadas, así como predictores que representen la tendencia y estacionalidad de la serie (si fueran necesarios). Responder a las siguientes cuestiones:

1) Representa la serie de Consumo en un gráfico temporal. ¿Cómo es la tendencia? ¿La serie presenta estacionalidad? ¿Cuál sería el periodo?

```
datos <- read.csv2("./Consumo_Electrico_Sim1.csv")
ts.plot(datos$Consumo)
```



- La serie **no** presenta estacionalidad (no hay patrón periódico).
- La serie oscila alrededor de la media.
- La varianza parece constante

2) Dividir los datos en entrenamiento y test, dejando para test los datos de marzo. Pasar la variable Consumo a serie temporal teniendo en cuenta la periodicidad diaria (periodo = 24 horas). Llámala **consumo.ts** y sepárala también en entrenamiento (**consumo.ts_train**) y test (**consumo.ts_test**). Usaremos **consumo.ts_train** para ajustar los modelos y usaremos **consumo.ts_test** para testeo.

```
inicio_test <- which(datos$Fecha == "01/03/2009 0:00")
indices_train <- 1:(inicio_test - 1)
indices_test <- inicio_test:nrow(datos)
datos_train <- datos[indices_train, ]
datos_test <- datos[indices_test, ]
```

```r
consumo.ts <- ts(datos$Consumo, start = c(1, 1), frequency = 24)
consumo.ts_train <- window(consumo.ts, end = c(59, 24))
consumo.ts_test <- window(consumo.ts, start = c(60, 1))
```

3) Usando la función **tslm()** del paquete **forecast**, ajusta un modelo de regresión lineal múltiple a la serie **consumo.ts_train**. Usa los predictores indicados en el enunciado y variables dummy para la estacionalidad diaria. ¿Debemos contemplar una recta para la tendencia?

```r
library(forecast)
library(tidyverse)
```

```r
tslm(
  consumo.ts_train ~ season + datos_train$Temperatura +
    datos_train$Humedad_rel + datos_train$WH2 + datos_train$WH4 +
    datos_train$WH5 + datos_train$WH6 + datos_train$WH7 +
    datos_train$FH1
)
```

```
##
## Call:
## tslm(formula = consumo.ts_train ~ season + datos_train$Temperatura +
##     datos_train$Humedad_rel + datos_train$WH2 + datos_train$WH4 +
##     datos_train$WH5 + datos_train$WH6 + datos_train$WH7 + datos_train$FH1)
##
## Coefficients:
##             (Intercept)                  season2                  season3
##                 7608.85                  -373.77                  -557.38
##                 season4                  season5                  season6
##                 -643.93                  -679.43                  -690.05
##                 season7                  season8                  season9
##                 -721.41                  -500.69                   156.39
##                season10                 season11                 season12
##                  583.37                   779.53                   882.50
##                season13                 season14                 season15
##                  960.97                  1082.44                   961.87
##                season16                 season17                 season18
##                  421.00                   558.28                   687.53
##                season19                 season20                 season21
##                  629.69                   724.80                   722.53
##                season22                 season23                 season24
##                  617.89                   708.46                   365.69
## datos_train$Temperatura  datos_train$Humedad_rel          datos_train$WH2
##                  -15.22                   304.24                   166.11
##         datos_train$WH4          datos_train$WH5          datos_train$WH6
##                   98.15                    49.34                  -615.44
##         datos_train$WH7          datos_train$FH1
##                 -977.95                 -1015.84
```

```r
p1 <- datos |>
  select(-Fecha, -Consumo)

p1_train <- p1[indices_train, ]
p1_test <- p1[indices_test, ]

p1_train_matrix <- as.matrix(p1_train)

modelo_season <- tslm(consumo.ts_train ~ season + p1_train_matrix)

modelo_season
```

```
##
## Call:
## tslm(formula = consumo.ts_train ~ season + p1_train_matrix)
```

```
##
## Coefficients:
##               (Intercept)                      season2                      season3
##                   7597.33                      -373.77                      -557.38
##                   season4                      season5                      season6
##                   -643.94                      -679.44                      -690.06
##                   season7                      season8                      season9
##                   -721.42                      -500.71                       156.37
##                  season10                     season11                     season12
##                    583.35                       779.52                       882.51
##                  season13                     season14                     season15
##                    961.00                      1082.48                       961.93
##                  season16                     season17                     season18
##                    421.06                       558.34                       687.58
##                  season19                     season20                     season21
##                    629.73                       724.83                       722.55
##                  season22                     season23                     season24
##                    617.90                       708.46                       365.69
## p1_train_matrixTemperatura  p1_train_matrixHumedad_rel          p1_train_matrixWH2
##                    -15.24                       304.31                       181.38
##        p1_train_matrixWH3           p1_train_matrixWH4           p1_train_matrixWH5
##                     30.55                       113.43                        64.62
##        p1_train_matrixWH6           p1_train_matrixWH7           p1_train_matrixFH1
##                   -600.17                      -962.69                     -1015.85
```

4) Repite el apartado anterior pero modelizando la estacionalidad con series de Fourier.

```r
tslm(
  consumo.ts_train ~ fourier(consumo.ts_train, K = 12) +
    datos_train$Temperatura +
    datos_train$Humedad_rel + datos_train$WH2 + datos_train$WH4 +
    datos_train$WH5 + datos_train$WH6 + datos_train$WH7 +
    datos_train$FH1
)
```

```
##
## Call:
## tslm(formula = consumo.ts_train ~ fourier(consumo.ts_train, K = 12) +
##     datos_train$Temperatura + datos_train$Humedad_rel + datos_train$WH2 +
##     datos_train$WH4 + datos_train$WH5 + datos_train$WH6 + datos_train$WH7 +
##     datos_train$FH1)
##
## Coefficients:
##                          (Intercept)   fourier(consumo.ts_train, K = 12)S1-24
##                             7887.024                                 -692.669
##  fourier(consumo.ts_train, K = 12)C1-24   fourier(consumo.ts_train, K = 12)S2-24
##                             -275.769                                 -130.533
##  fourier(consumo.ts_train, K = 12)C2-24   fourier(consumo.ts_train, K = 12)S3-24
##                              375.857                                  -32.203
##  fourier(consumo.ts_train, K = 12)C3-24   fourier(consumo.ts_train, K = 12)S4-24
##                              -67.781                                  -39.196
##  fourier(consumo.ts_train, K = 12)C4-24   fourier(consumo.ts_train, K = 12)S5-24
##                               10.395                                  -16.199
##  fourier(consumo.ts_train, K = 12)C5-24   fourier(consumo.ts_train, K = 12)S6-24
##                              109.256                                   -6.835
##  fourier(consumo.ts_train, K = 12)C6-24   fourier(consumo.ts_train, K = 12)S7-24
##                              -54.837                                    7.030
##  fourier(consumo.ts_train, K = 12)C7-24   fourier(consumo.ts_train, K = 12)S8-24
##                               23.194                                  -32.623
##  fourier(consumo.ts_train, K = 12)C8-24   fourier(consumo.ts_train, K = 12)S9-24
##                               37.957                                  -13.345
##  fourier(consumo.ts_train, K = 12)C9-24  fourier(consumo.ts_train, K = 12)S10-24
##                              -45.114                                    5.090
## fourier(consumo.ts_train, K = 12)C10-24  fourier(consumo.ts_train, K = 12)S11-24
```

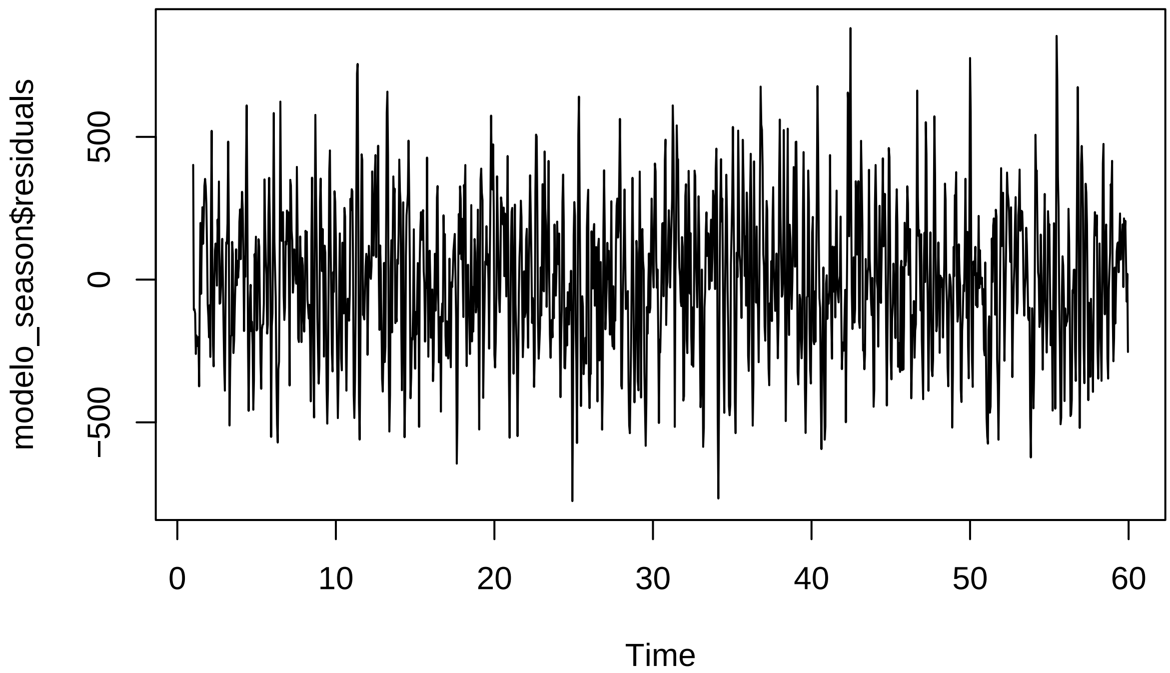
```
##                                  -8.344                                   -8.250
## fourier(consumo.ts_train, K = 12)C11-24  fourier(consumo.ts_train, K = 12)C12-24
##                                  -2.189                                  -15.114
##                 datos_train$Temperatura                  datos_train$Humedad_rel
##                                 -15.223                                  304.239
##                         datos_train$WH2                          datos_train$WH4
##                                 166.106                                   98.149
##                         datos_train$WH5                          datos_train$WH6
##                                  49.338                                 -615.437
##                         datos_train$WH7                          datos_train$FH1
##                                -977.946                                -1015.837
```

```
modelo_fourier <- tslm(
  consumo.ts_train ~ fourier(consumo.ts_train, K = 12) +
    p1_train_matrix
)
modelo_fourier
```

```
##
## Call:
## tslm(formula = consumo.ts_train ~ fourier(consumo.ts_train, K = 12) +
##     p1_train_matrix)
##
## Coefficients:
##                             (Intercept)   fourier(consumo.ts_train, K = 12)S1-24
##                                7875.524                                 -692.702
##  fourier(consumo.ts_train, K = 12)C1-24   fourier(consumo.ts_train, K = 12)S2-24
##                                -275.775                                 -130.519
##  fourier(consumo.ts_train, K = 12)C2-24   fourier(consumo.ts_train, K = 12)S3-24
##                                 375.850                                  -32.204
##  fourier(consumo.ts_train, K = 12)C3-24   fourier(consumo.ts_train, K = 12)S4-24
##                                 -67.778                                  -39.196
##  fourier(consumo.ts_train, K = 12)C4-24   fourier(consumo.ts_train, K = 12)S5-24
##                                  10.396                                  -16.199
##  fourier(consumo.ts_train, K = 12)C5-24   fourier(consumo.ts_train, K = 12)S6-24
##                                 109.256                                   -6.835
##  fourier(consumo.ts_train, K = 12)C6-24   fourier(consumo.ts_train, K = 12)S7-24
##                                 -54.837                                    7.030
##  fourier(consumo.ts_train, K = 12)C7-24   fourier(consumo.ts_train, K = 12)S8-24
##                                  23.194                                  -32.622
##  fourier(consumo.ts_train, K = 12)C8-24   fourier(consumo.ts_train, K = 12)S9-24
##                                  37.957                                  -13.345
##  fourier(consumo.ts_train, K = 12)C9-24  fourier(consumo.ts_train, K = 12)S10-24
##                                 -45.115                                    5.090
## fourier(consumo.ts_train, K = 12)C10-24  fourier(consumo.ts_train, K = 12)S11-24
##                                  -8.344                                   -8.250
## fourier(consumo.ts_train, K = 12)C11-24  fourier(consumo.ts_train, K = 12)C12-24
##                                  -2.189                                  -15.114
##              p1_train_matrixTemperatura               p1_train_matrixHumedad_rel
##                                 -15.237                                  304.311
##                      p1_train_matrixWH2                       p1_train_matrixWH3
##                                 181.379                                   30.546
##                      p1_train_matrixWH4                       p1_train_matrixWH5
##                                 113.425                                   64.622
##                      p1_train_matrixWH6                       p1_train_matrixWH7
##                                -600.166                                 -962.686
##                      p1_train_matrixFH1
##                               -1015.855
```

5) ¿Los residuos de los modelos de los apartados (3) y (4) se comportan como un ruido blanco?

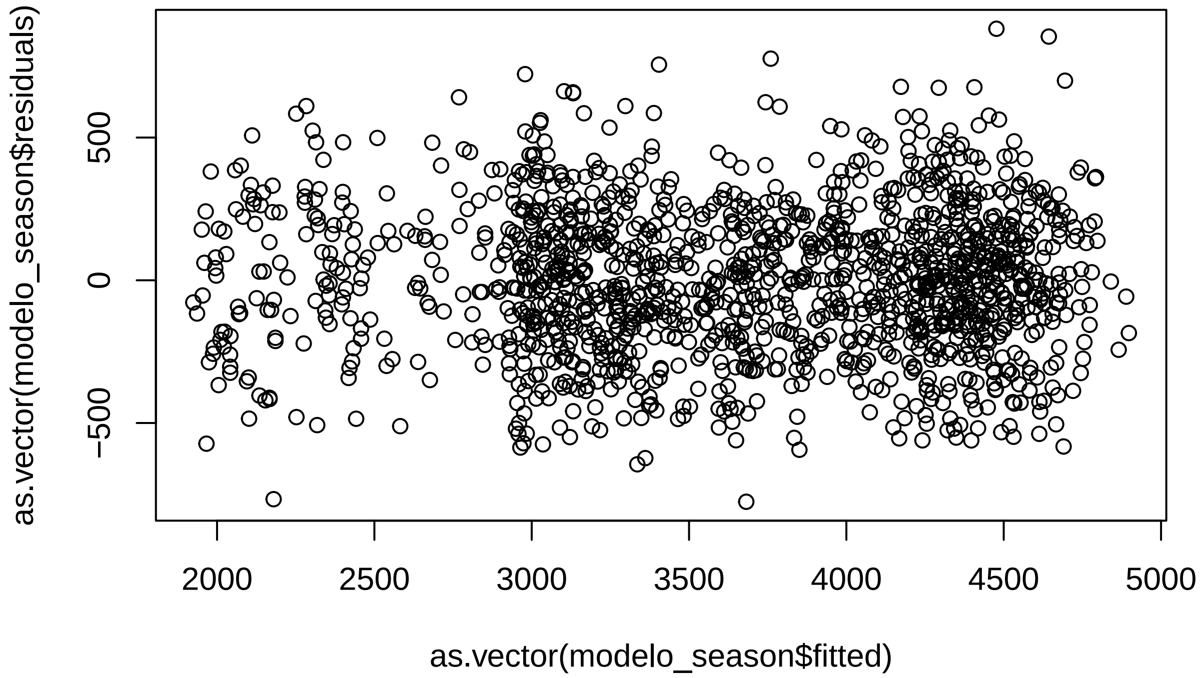
```
plot(modelo_season$residuals)
```



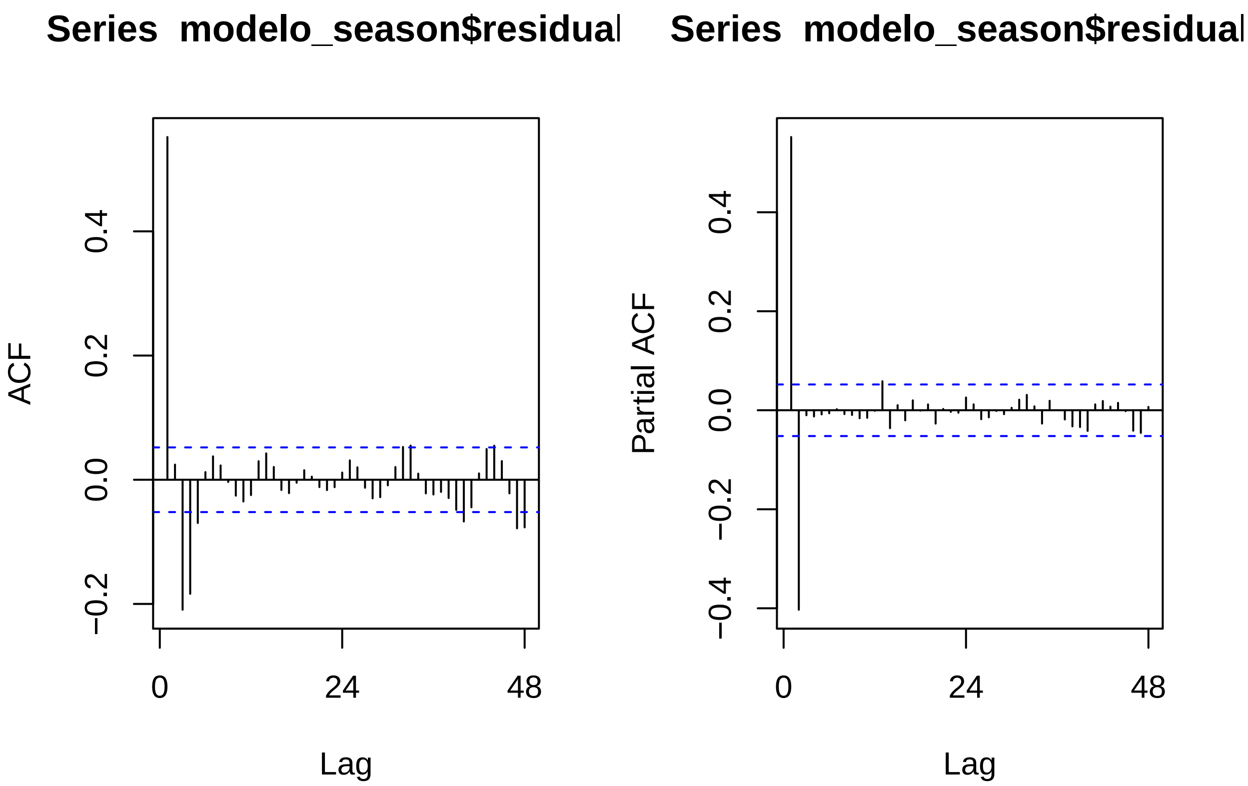

```r
shapiro.test(modelo_season$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  modelo_season$residuals
## W = 0.99837, p-value = 0.1925
```

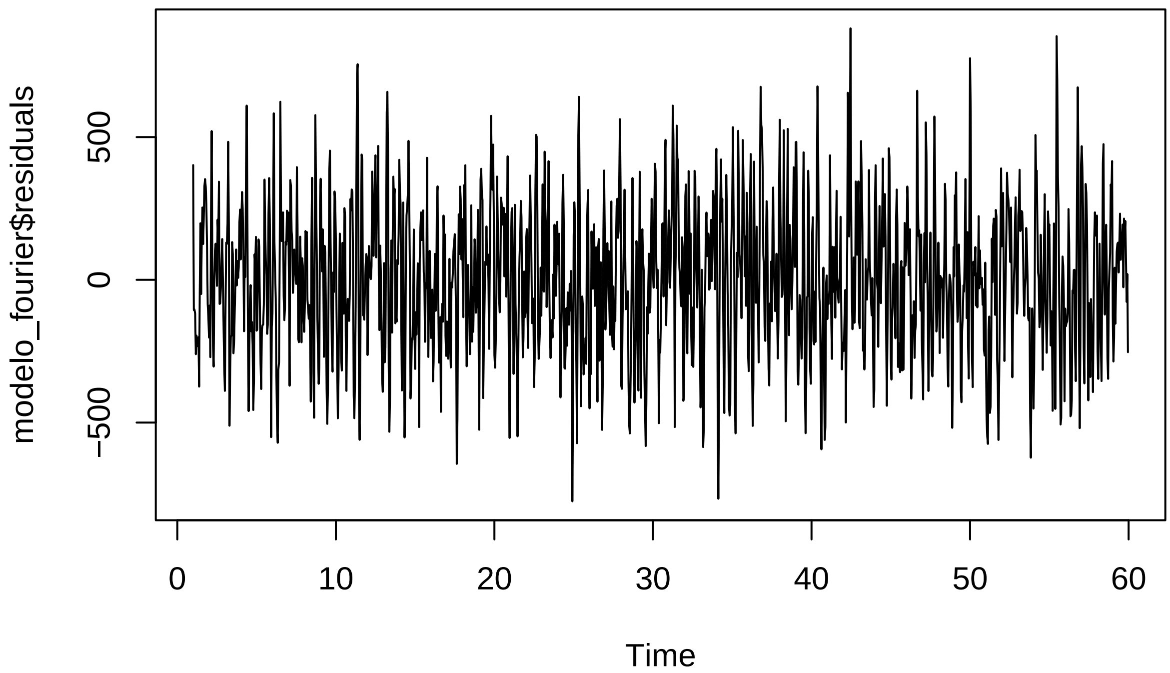
```r
plot(as.vector(modelo_season$fitted), as.vector(modelo_season$residuals))
```




```r
par(mfrow = c(1, 2))
Acf(modelo_season$residuals)
Pacf(modelo_season$residuals)
```



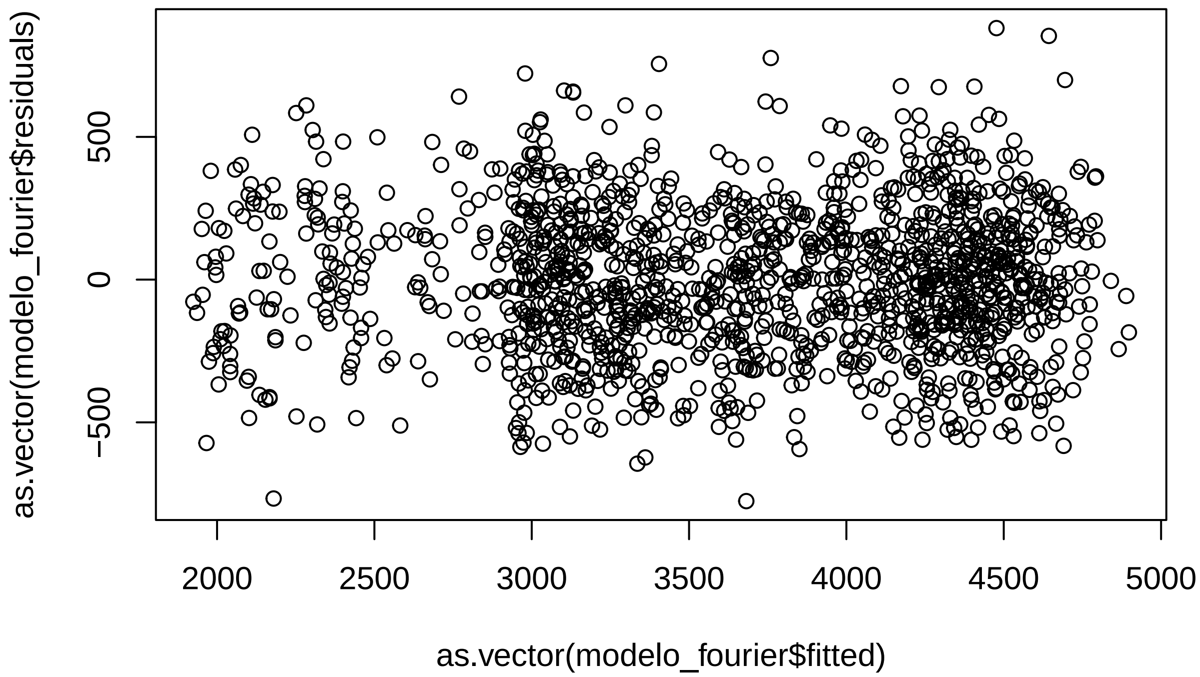

```r
plot(modelo_fourier$residuals)
```



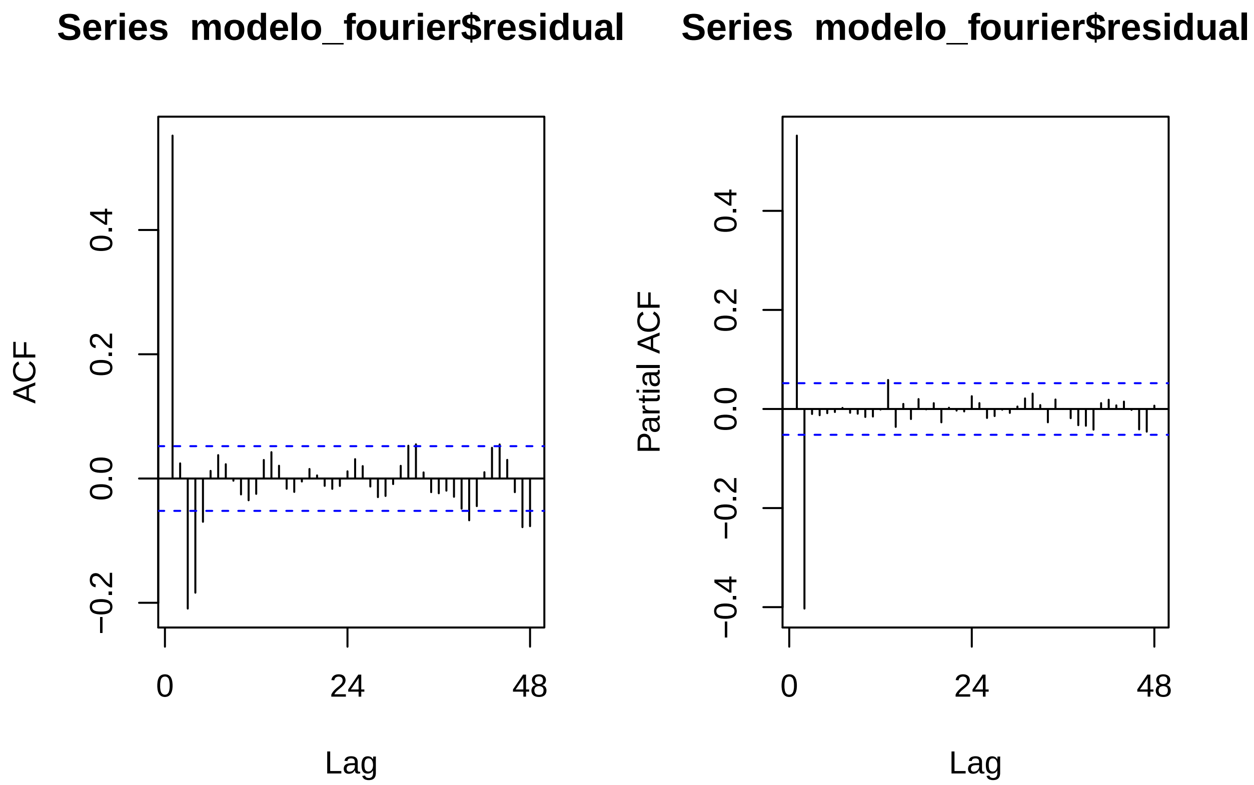

```r
shapiro.test(modelo_fourier$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  modelo_fourier$residuals
## W = 0.99837, p-value = 0.1925
```

```r
plot(as.vector(modelo_fourier$fitted), as.vector(modelo_fourier$residuals))
```



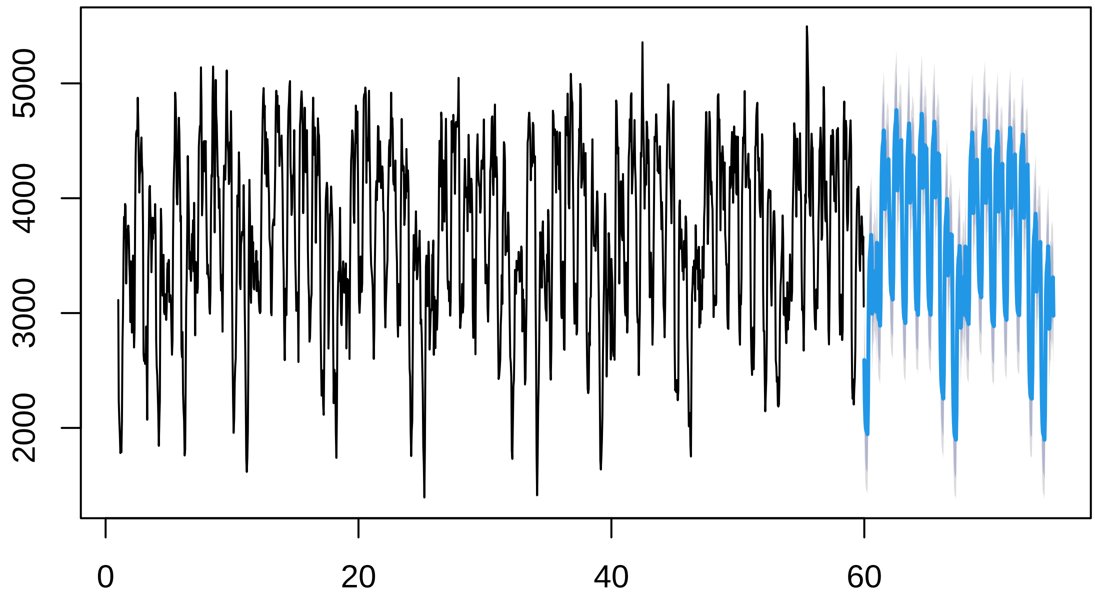

```
par(mfrow = c(1, 2))
Acf(modelo_fourier$residuals)
Pacf(modelo_fourier$residuals)
```



6) Con los modelos resultantes en los apartados (3) y (4), realiza las predicciones de la zona test (primera quincena de marzo).

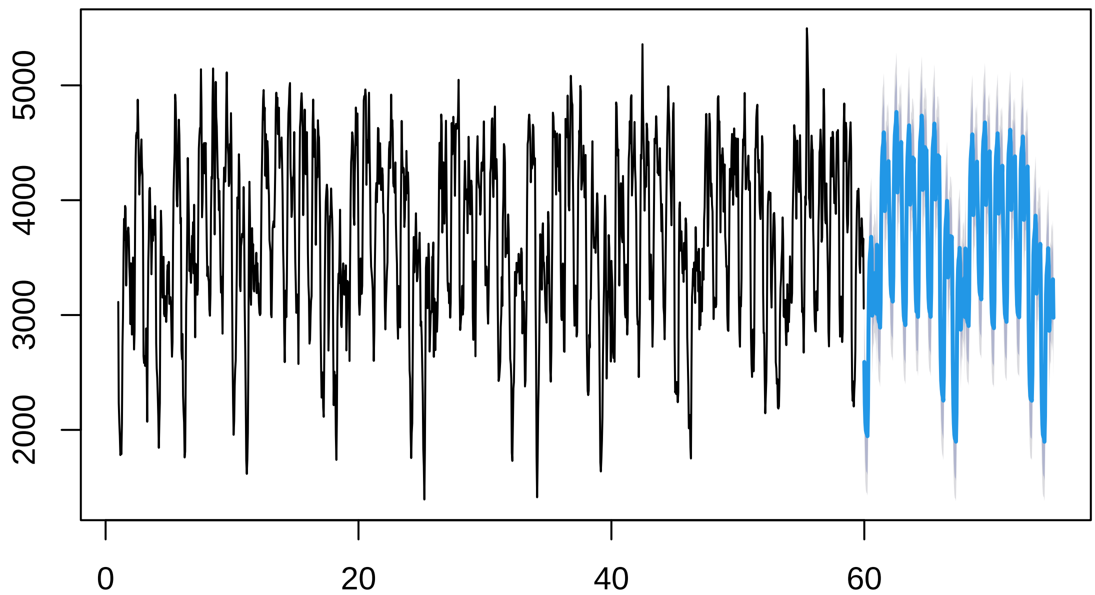
```
pred_season <- forecast(modelo_season, newdata = p1_test) # Datos futuros
plot(pred_season, main = "pred_season")
```

pred_season



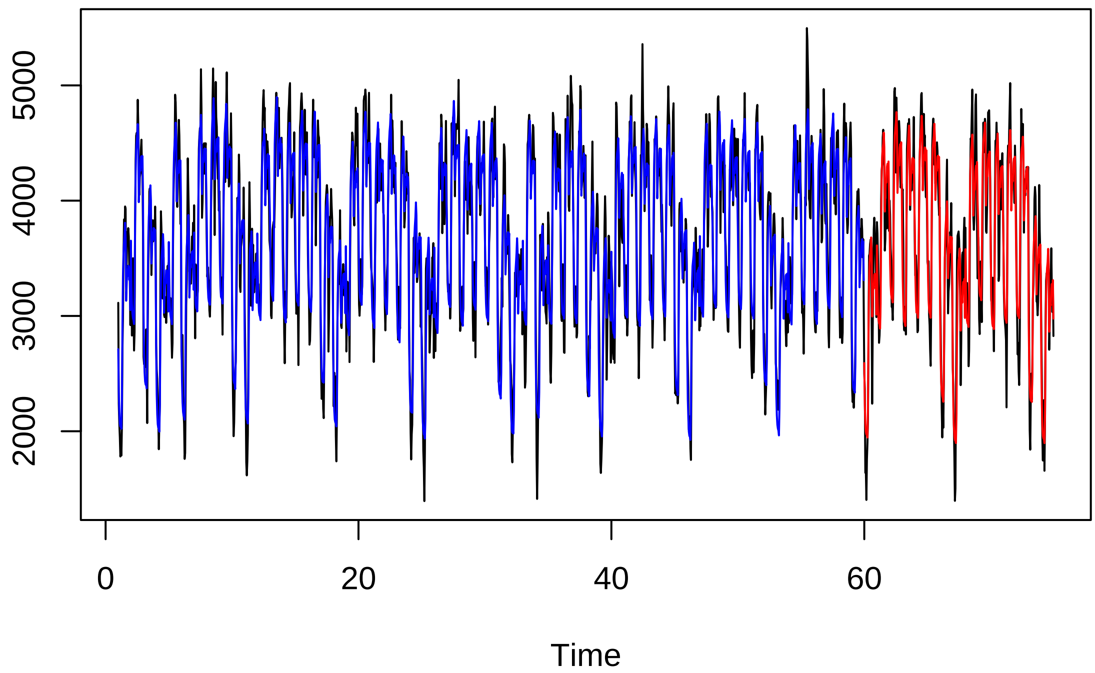

```r
p1_test_fourier_k_12 <- data.frame(p1_test, fourier(consumo.ts_train, K = 12, h = 24 * 15))
pred_fourier <- forecast(modelo_fourier, newdata = p1_test_fourier_k_12)
plot(pred_fourier, main = "pred_fourier")
```

pred_fourier



7) Representa los datos reales (negro) junto a valores ajustados (azul) y predicciones puntuales (rojo).

```r
ts.plot(consumo.ts,
  modelo_fourier$fitted.values,
  pred_fourier$mean,
  col = c("black", "blue", "red")
)
```



Ahora vamos a contruir un modelo de regresión dinámica para la serie `consumo.ts_train`.

8) Crea una matriz con los predictores del conjunto de datos, incluyendo la estacionalidad diaria con dummies.

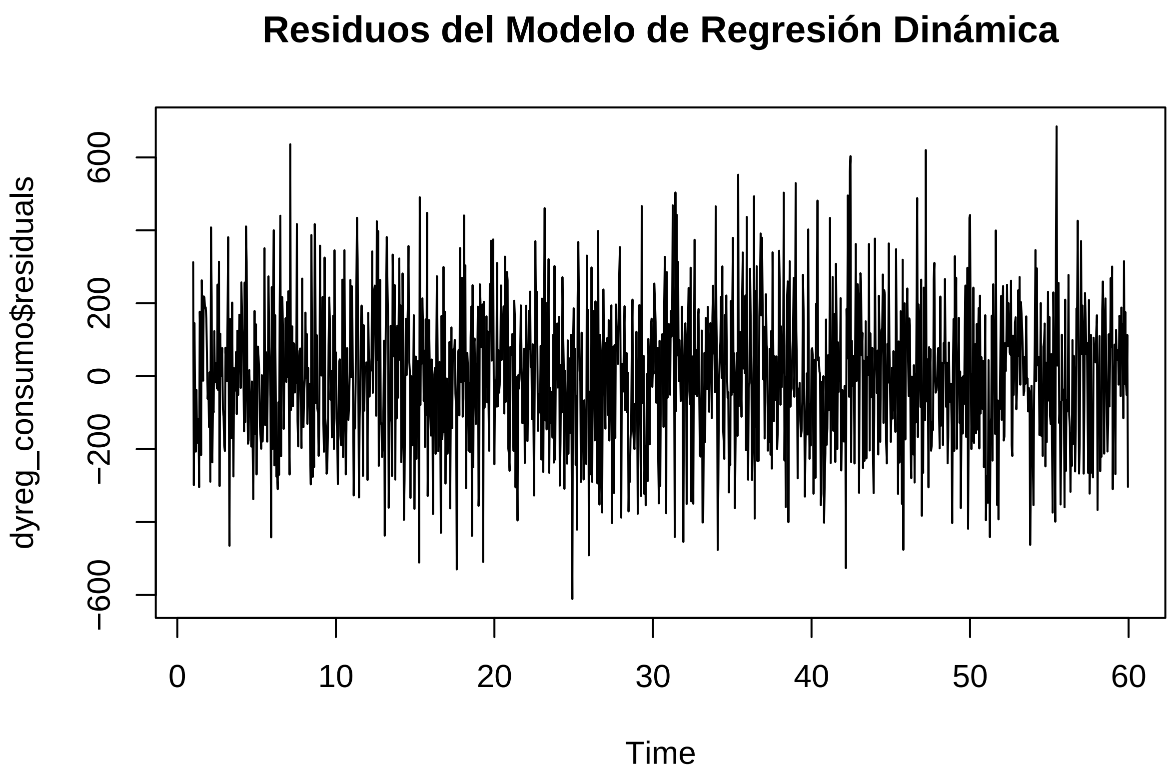
```
xreg_train <- cbind(seasonaldummy(consumo.ts_train), as.matrix(p1_train))
```

9) Obtén el modelo de regresión dinámica usando la función `auto.arima()`. ¿Qué modelo ARIMA se obtiene para los residuos del modelo de regresión?

```
dyreg_consumo <- auto.arima(
  consumo.ts_train,
  xreg = xreg_train
)
```

10) Comprueba si ahora los residuos son ruido blanco gaussiano.

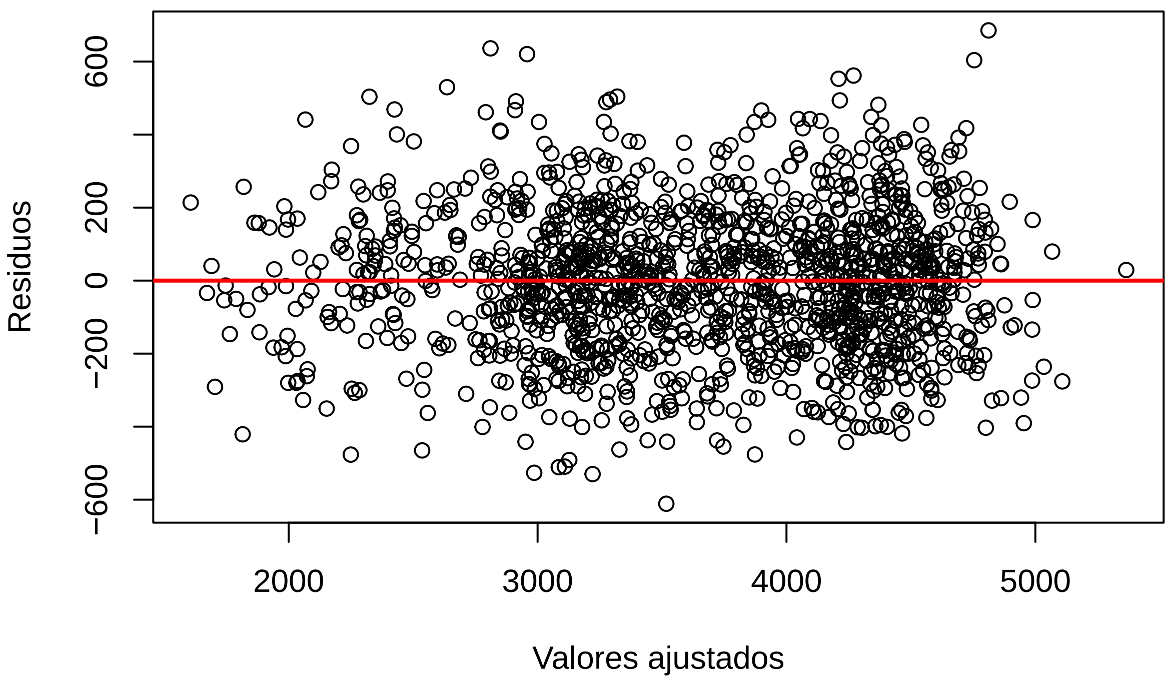
```
# Gráfico de los residuos
plot(dyreg_consumo$residuals,
  main = "Residuos del Modelo de Regresión Dinámica"
)
```




```r
# Test de normalidad
shapiro.test(dyreg_consumo$residuals)
```

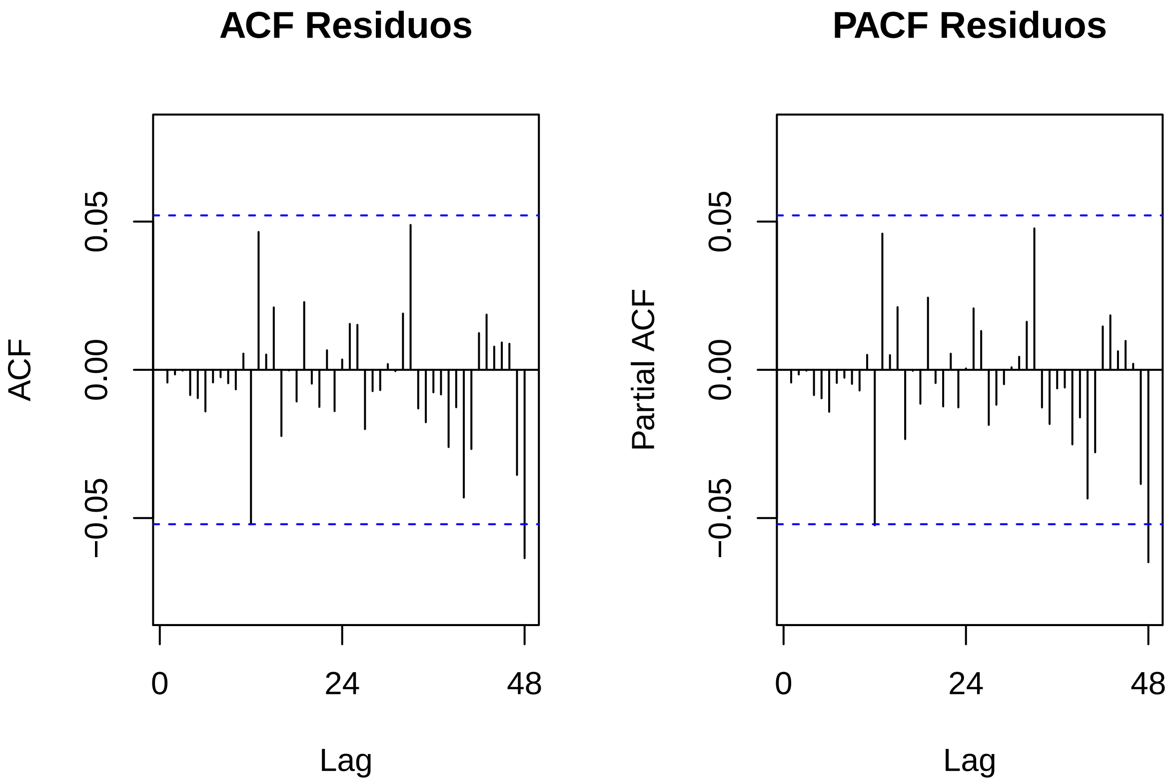
```
##
##  Shapiro-Wilk normality test
##
## data:  dyreg_consumo$residuals
## W = 0.99868, p-value = 0.3701
```

```r
# Gráfico de residuos vs ajustados
plot(as.vector(fitted(dyreg_consumo)), as.vector(dyreg_consumo$residuals),
  xlab = "Valores ajustados", ylab = "Residuos"
)
abline(h = 0, col = "red", lwd = 2)
```

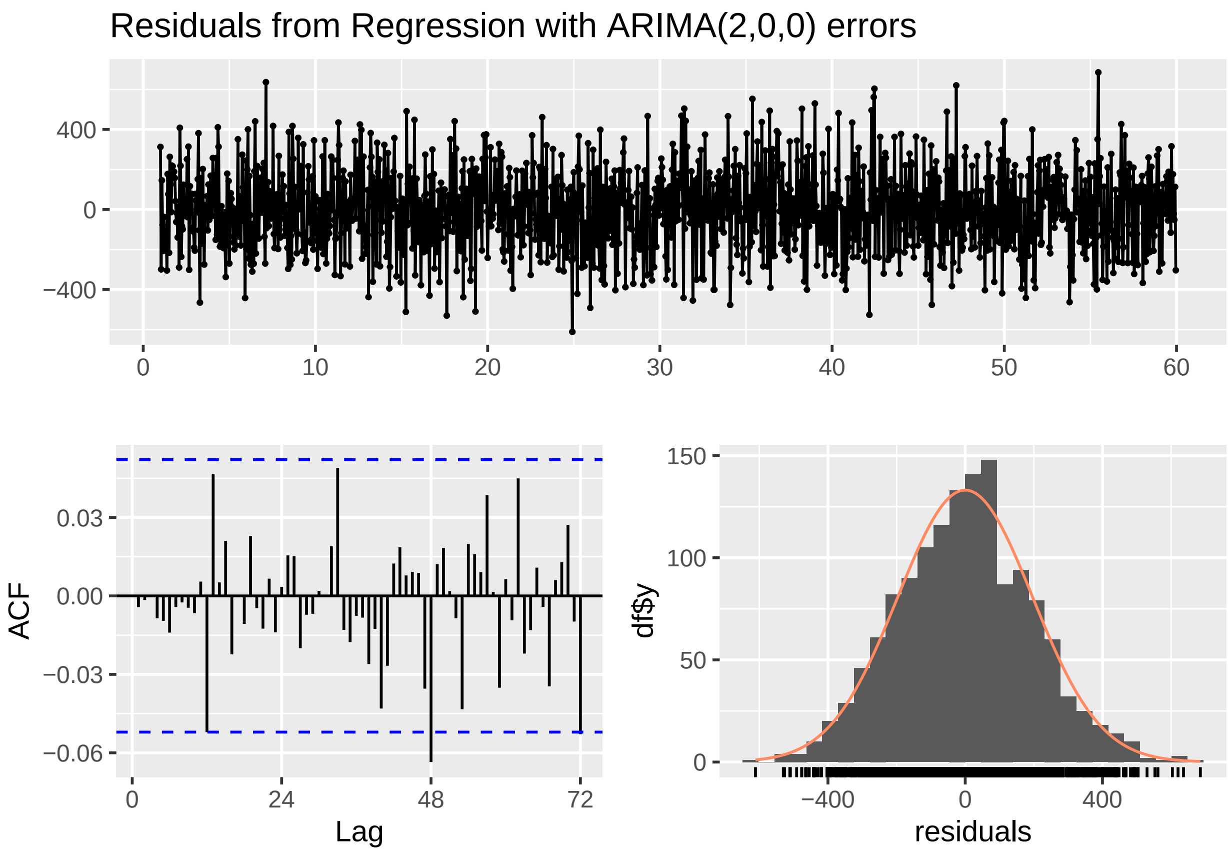



```r
# ACF y PACF de los residuos
par(mfrow = c(1, 2))
```

```
Acf(dyreg_consumo$residuals, main = "ACF Residuos")
Pacf(dyreg_consumo$residuals, main = "PACF Residuos")
```




```
# Test de Ljung-Box para autocorrelación
checkresiduals(dyreg_consumo)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Regression with ARIMA(2,0,0) errors
## Q* = 30.689, df = 46, p-value = 0.9597
```

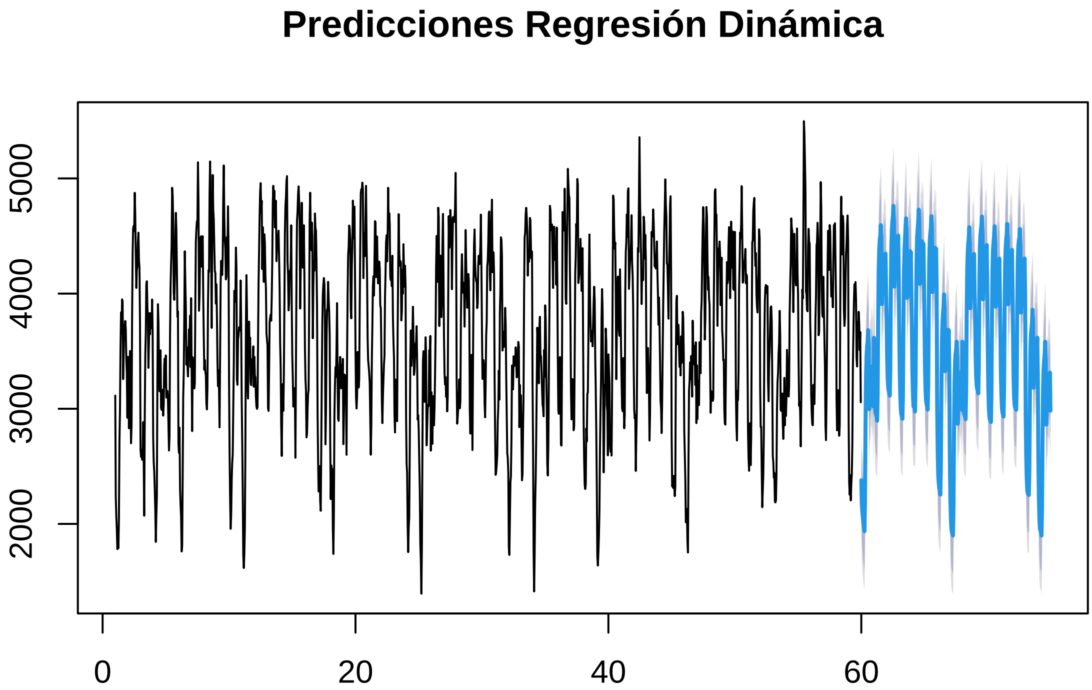
```
##
## Model df: 2.   Total lags used: 48
```

- **Shapiro Test:** $p - value = 0.3701 \gg 0.05 \implies$ los residuos son normales.
- **Ljung-Box:** $p - value = 0.9597 \gg 0.05 \implies$ los residuos son ruido blanco.

Conclusión: Los residuos del modelo de regresión dinámica se comportan como **ruido blanco gaussiano**, lo cual indica que el modelo ARIMA captura correctamente la estructura de dependendecia temporal que quedaba en los residuos del modelo de regresión lineal simple.

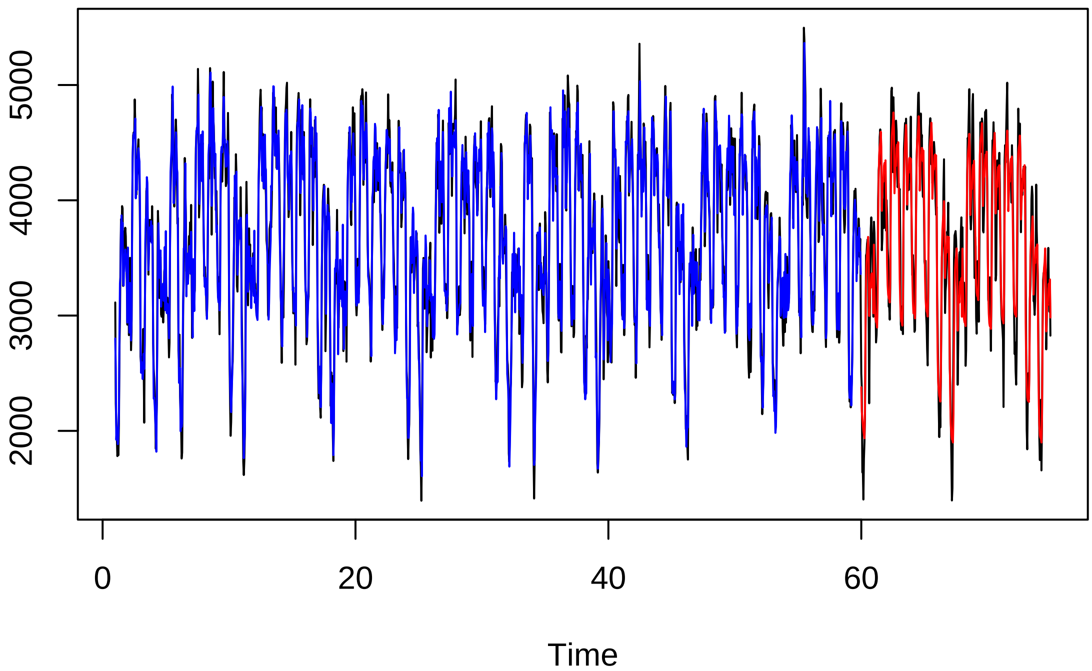
11) Realiza las predicciones de la zona test (primera quincena de marzo). ¿Se trata de predicciones ex-ante o predicciones ex-post?

```r
xreg_test <- cbind(seasonaldummy(consumo.ts_test), as.matrix(p1_test))
pred_dyreg <- forecast(dyreg_consumo, xreg = xreg_test)
plot(pred_dyreg, main = "Predicciones Regresión Dinámica")
```



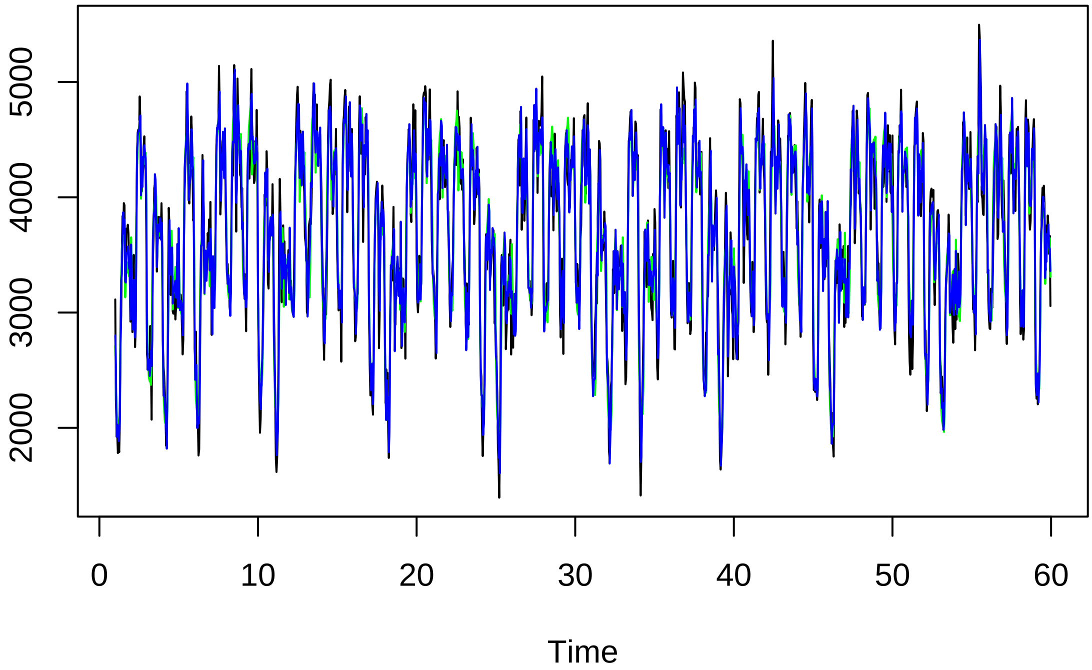
12) Representamos los datos reales (negro) junto a valores (azul) y predicciones puntuales (rojo).

```r
ts.plot(consumo.ts, fitted(dyreg_consumo), pred_dyreg$mean,
  col = c("black", "blue", "red")
)
```



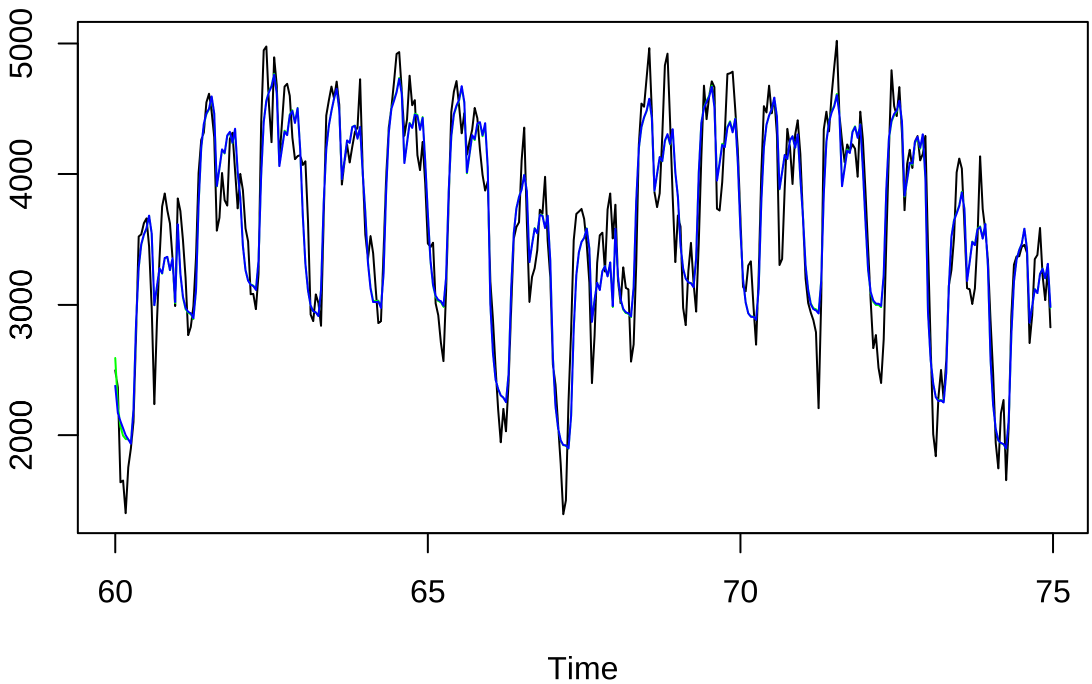
13) Compara los valores reales (negro) en la zona entrenamiento, con los ajustados usando sólo el ajuste RLM (verde) y los ajustados usando Regresión Dinámica (azul).

```
ts.plot(consumo.ts_train, modelo_season$fitted.values, fitted(dyreg_consumo),
  col = c("black", "green", "blue")
)
```



14) ¿Qué sucede con las predicciones para marzo usando modelo RLM frente a Regresión Dinámica?

```
ts.plot(consumo.ts_test, pred_season$mean, pred_dyreg$mean,
  col = c("black", "green", "blue")
)
```




```
accuracy(pred_season$mean, consumo.ts_test)
```

```
##                  ME     RMSE      MAE        MPE     MAPE      ACF1 Theil's U
## Test set 9.819265 269.8456 214.8079 -0.6140048 6.512016 0.5563487 0.7100989
```

```
accuracy(pred_dyreg$mean, consumo.ts_test)
```

```
##                  ME     RMSE      MAE        MPE     MAPE     ACF1 Theil's U
```

```
## Test set 10.17714 271.1167 215.8092 -0.6083953 6.554934 0.557954  0.716412
```

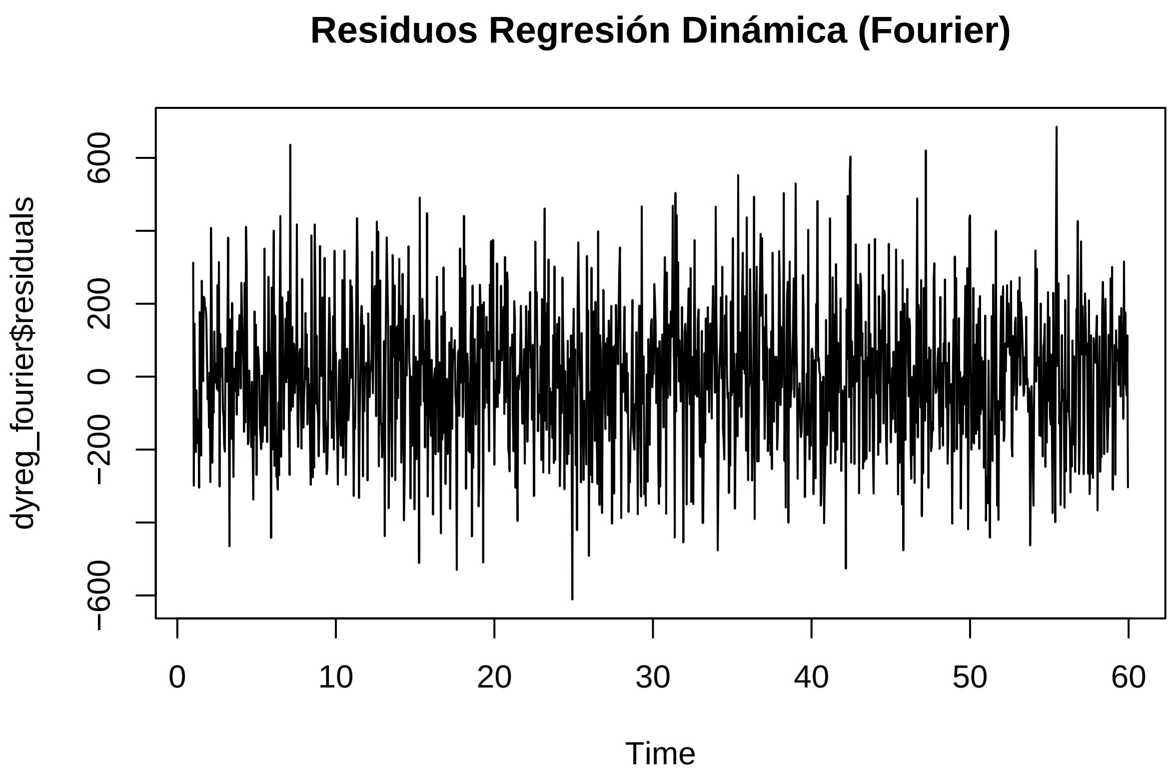
15) Repite los apartados (8) a (14) usando series de Fourier para modelar la estacionalidad diaria.

```r
fourier_train <- fourier(consumo.ts_train, K = 12)
x_reg_train_fourier <- cbind(fourier_train, as.matrix(p1_train))
```

```r
dyreg_fourier <- auto.arima(consumo.ts_train, xreg = x_reg_train_fourier)
dyreg_fourier
```

```
## Series: consumo.ts_train
## Regression with ARIMA(2,0,0) errors
##
## Coefficients:
##          ar1      ar2  intercept      S1-24      C1-24      S2-24     C2-24     S3-24
##       0.7786  -0.4062  7810.1825  -693.5395  -276.4328  -129.7608  375.8855  -32.3754
## s.e.  0.0244   0.0244   778.3813    14.1023    12.7932    14.3303   13.9896   15.6284
##          C3-24     S4-24    C4-24      S5-24      C5-24     S6-24      C6-24   S7-24     C7-24
##       -67.3262  -39.8660  11.0274   -16.9409  109.9706   -7.4138   -54.2121  6.6044   23.8017
## s.e.   15.6461   14.1939  14.1901    10.4329   10.4275    7.5400     7.5394  5.7750    5.7762
##          S8-24     C8-24     S9-24      C9-24  S10-24    C10-24    S11-24    C11-24     C12-24
##       -32.9342   38.5447  -13.5738   -44.5656  4.9416   -7.7851   -8.3165   -1.6335   -14.8391
## s.e.    4.7080    4.7092    4.0520     4.0556  3.6575    3.6605    3.4455    3.4490     2.3902
##       Temperatura  Humedad_rel        WH2      WH3       WH4      WH5        WH6        WH7
##          -15.0630     330.3363   173.8178  27.6999  103.7787  68.0800  -607.6050  -967.5398
## s.e.       2.7195     100.4100    32.4624  31.0387   30.8331  30.4291    30.5795    30.8175
##             FH1
##       -1019.2685
## s.e.     40.6924
##
## sigma^2 = 39550:  log likelihood = -9486.27
## AIC=19044.54   AICc=19046.47   BIC=19233.74
```

```r
# Grafico de residuos
plot(dyreg_fourier$residuals, main = "Residuos Regresión Dinámica (Fourier)")
```

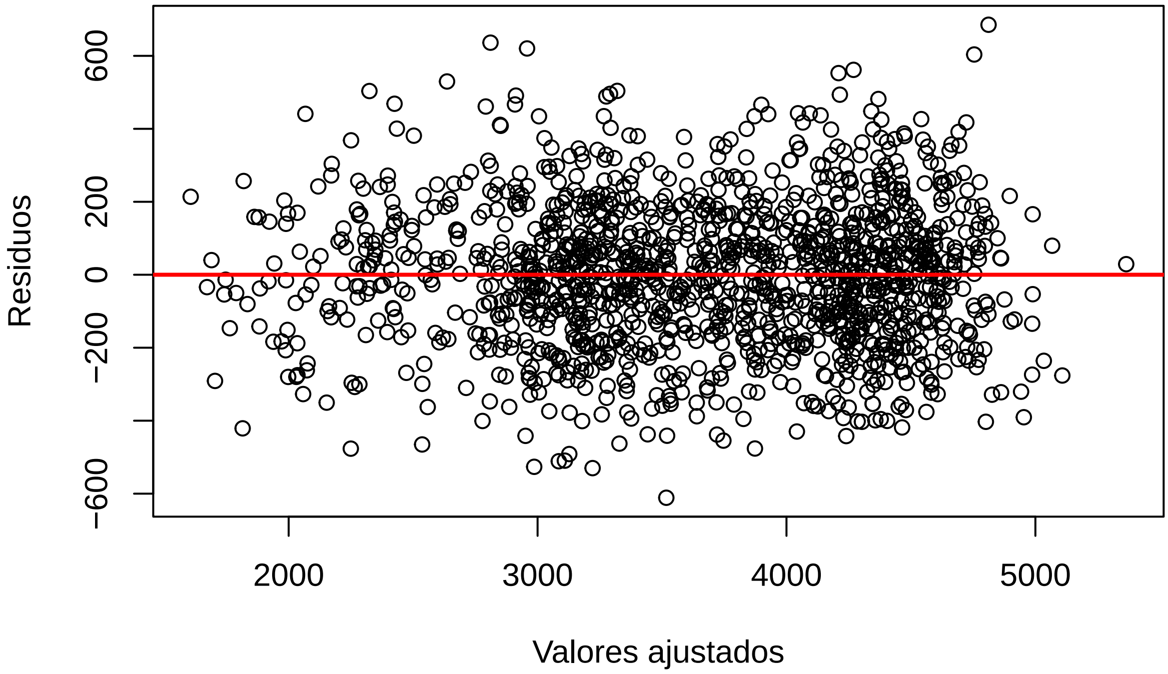



```r
# Test de normalidad
shapiro.test(dyreg_fourier$residuals)
```

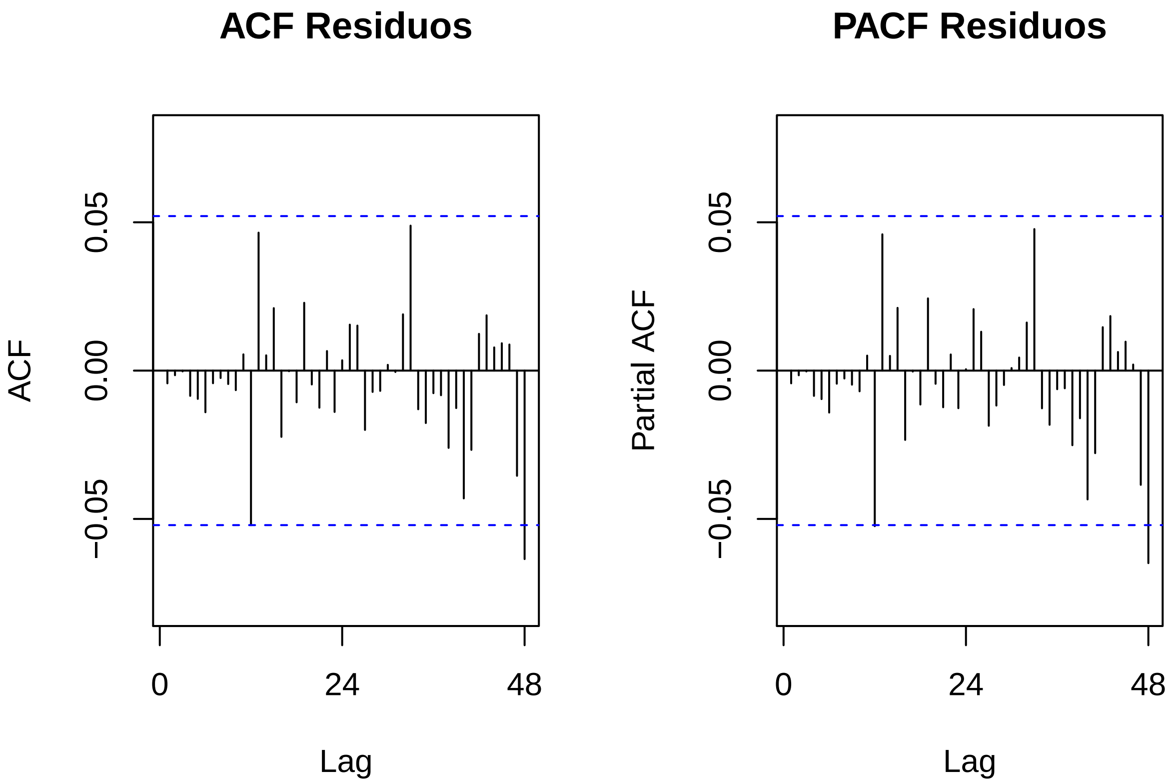
```
##
```

```
##  Shapiro-Wilk normality test
##
## data:  dyreg_fourier$residuals
## W = 0.99868, p-value = 0.3701
```

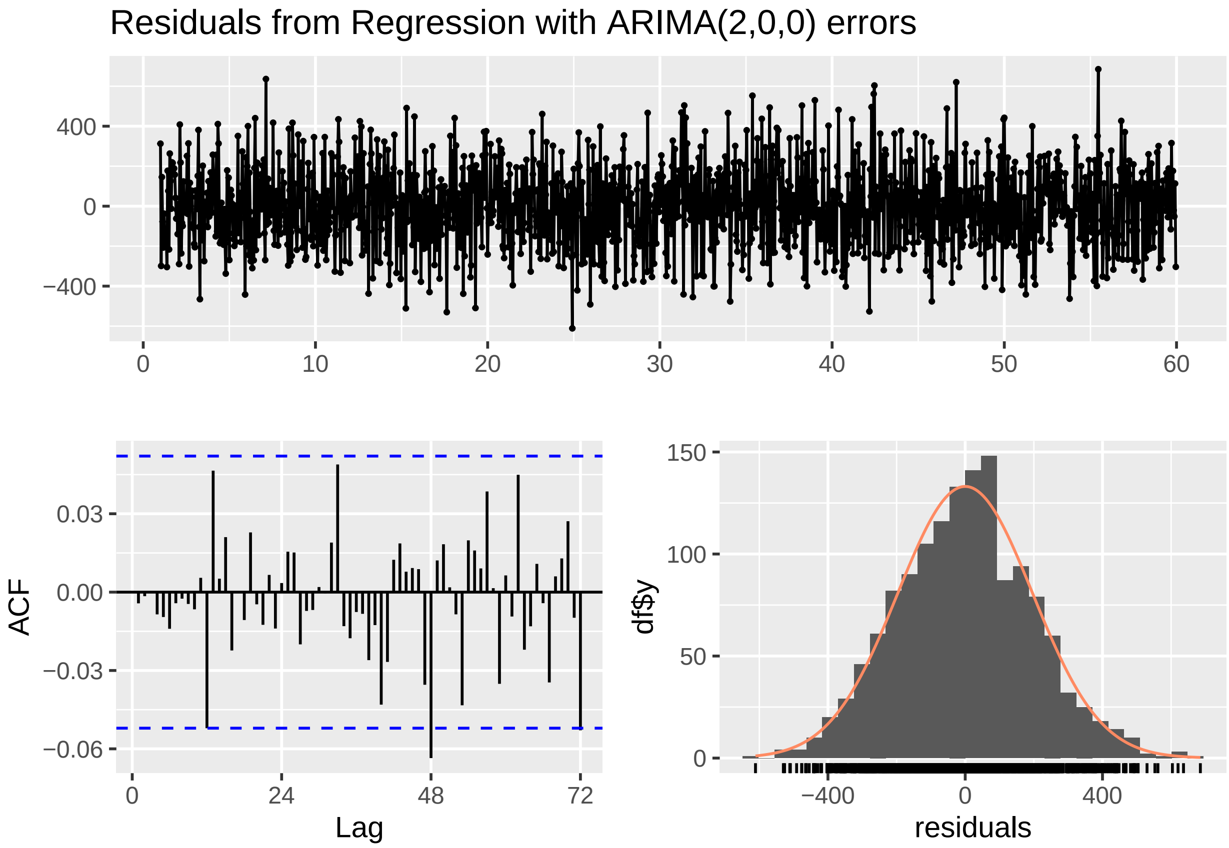
```r
# Residuos vs ajustados
plot(as.vector(fitted(dyreg_fourier)), as.vector(dyreg_fourier$residuals),
  xlab = "Valores ajustados", ylab = "Residuos"
)
abline(h = 0, col = "red", lwd = 2)
```




```r
# ACF y PACF de los residuos
par(mfrow = c(1, 2))
Acf(dyreg_fourier$residuals, main = "ACF Residuos")
Pacf(dyreg_fourier$residuals, main = "PACF Residuos")
```



```r
# Test de Ljung-Box para autocorrelación
checkresiduals(dyreg_fourier)
```

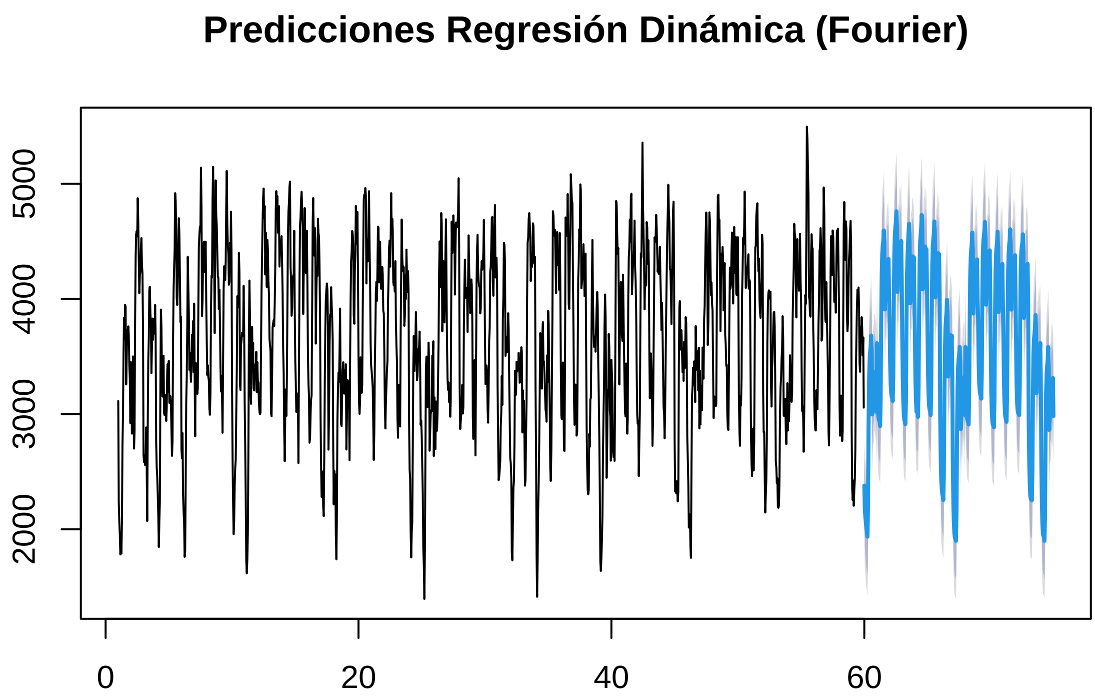



```
##
##  Ljung-Box test
##
## data:  Residuals from Regression with ARIMA(2,0,0) errors
## Q* = 30.689, df = 46, p-value = 0.9597
##
## Model df: 2.   Total lags used: 48
```

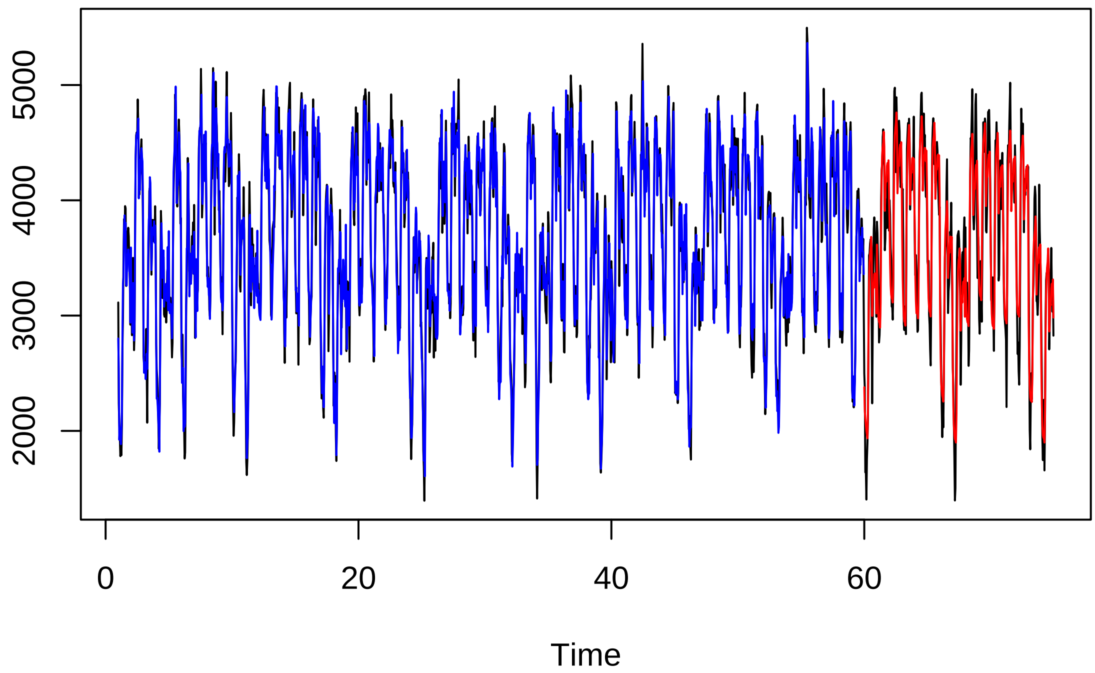
- **Shapiro Test:** $p - value = 0.3701 \gg 0.05 \implies$ los residuos son normales.
- **Ljung-Box:** $p - value = 0.9597 \gg 0.05 \implies$ los residuos son ruido blanco.

Conclusión: Los residuos del modelo de regresión dinámica se comportan como **ruido blanco gaussiano**, lo cual indica que el modelo ARIMA captura correctamente la estructura de dependendecia temporal que quedaba en los residuos del modelo de regresión lineal simple.

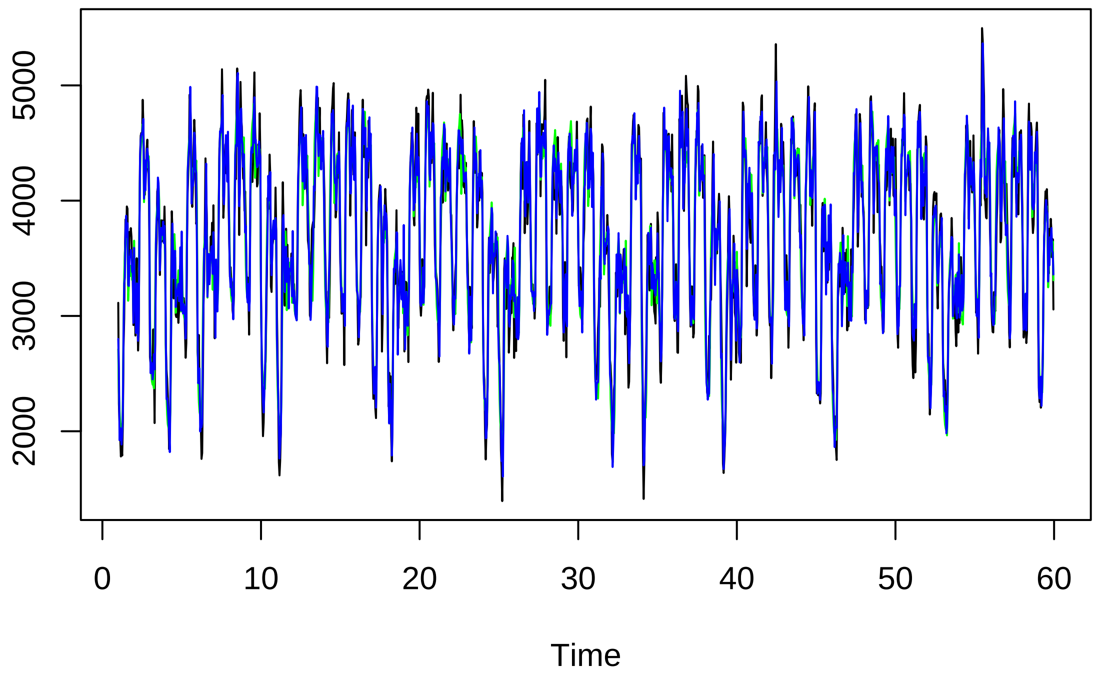
```r
fourier_test <- fourier(consumo.ts_train, K = 12, h = length(consumo.ts_test))
xreg_test_fourier <- cbind(fourier_test, as.matrix(p1_test))
pred_dyreg_fourier <- forecast(dyreg_fourier, xreg = xreg_test_fourier)
plot(pred_dyreg_fourier, main = "Predicciones Regresión Dinámica (Fourier)")
```




```r
ts.plot(consumo.ts, fitted(dyreg_fourier), pred_dyreg_fourier$mean,
  col = c("black", "blue", "red")
)
```




```r
ts.plot(consumo.ts_train, modelo_fourier$fitted.values, fitted(dyreg_fourier),
  col = c("black", "green", "blue")
)
```




```r
ts.plot(consumo.ts_test, pred_fourier$mean, pred_dyreg_fourier$mean,
  col = c("black", "green", "blue")
)
```