20. Series de tiempo

Jacobo Hirsch Rodriguez

2024-11-13

#Creación de la serie de tiempo

vamos a crear un vector con la variable objetivo (las ventas)

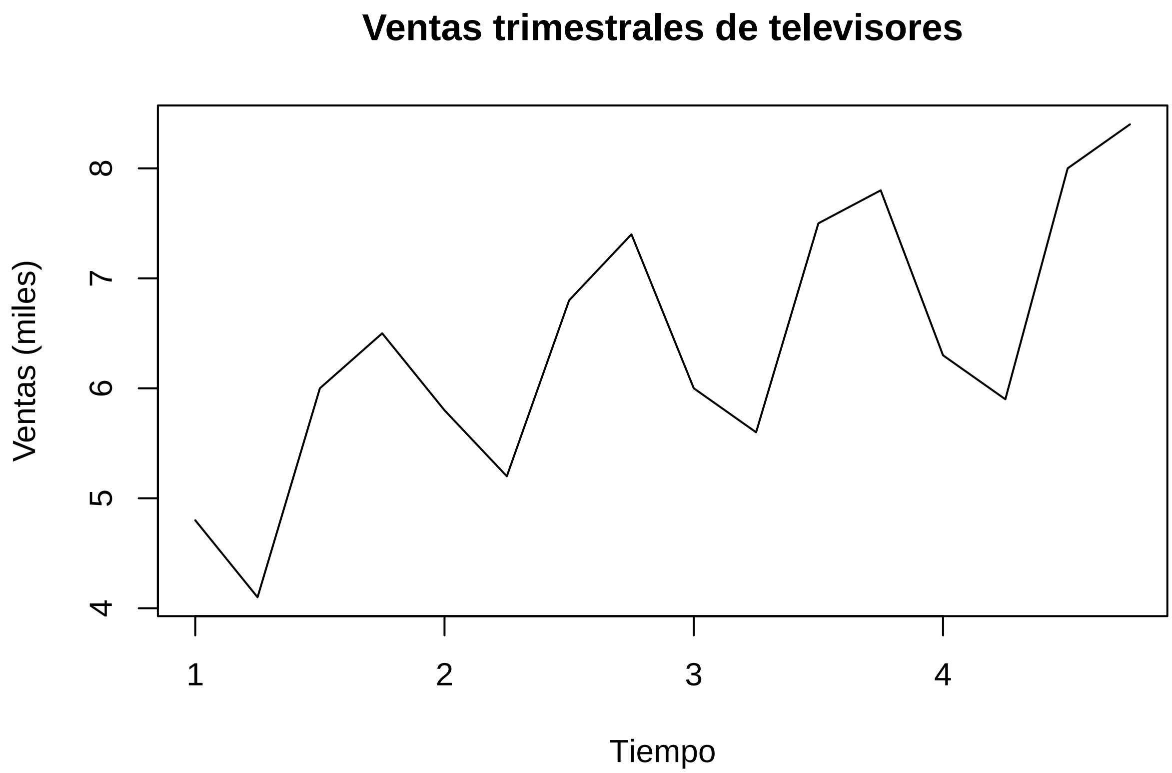
```r
# Paso 1: Crear un vector con los datos de ventas
ventas <- c(4.8, 4.1, 6.0, 6.5, 5.8, 5.2, 6.8, 7.4, 6.0, 5.6, 7.5, 7.8, 6.3, 5.9, 8.0, 8.4)
```

usamos la función ts() para crear una serie de tiempo trimestral. especificamos que empiezan en el año 1 y en el trimestre 1.

```r
# Crear la serie de tiempo
ventas_ts <- ts(ventas, start = c(1, 1), frequency = 4)
```

#Realiza el análisis de tendencia y estacionalidad graficamos la serie para observar de forma visual la tendencia y la estacionalidad

```r
# Graficar la serie de tiempo
plot(ventas_ts, main = "Ventas trimestrales de televisores", ylab = "Ventas (miles)", xlab = "Tiempo")
```



La serie parece que si presenta estacionalidad, ya que presenta patrones que se repiten a lo largo del tiempo y tambien parece que tiene una ligera tendencia ascendente. vamos a hacer una prueba de estacionalidad, en concreto la prueba de Dickey-Fuller para verificar si la serie es o no estacionaria.

```
library(tseries)
```

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

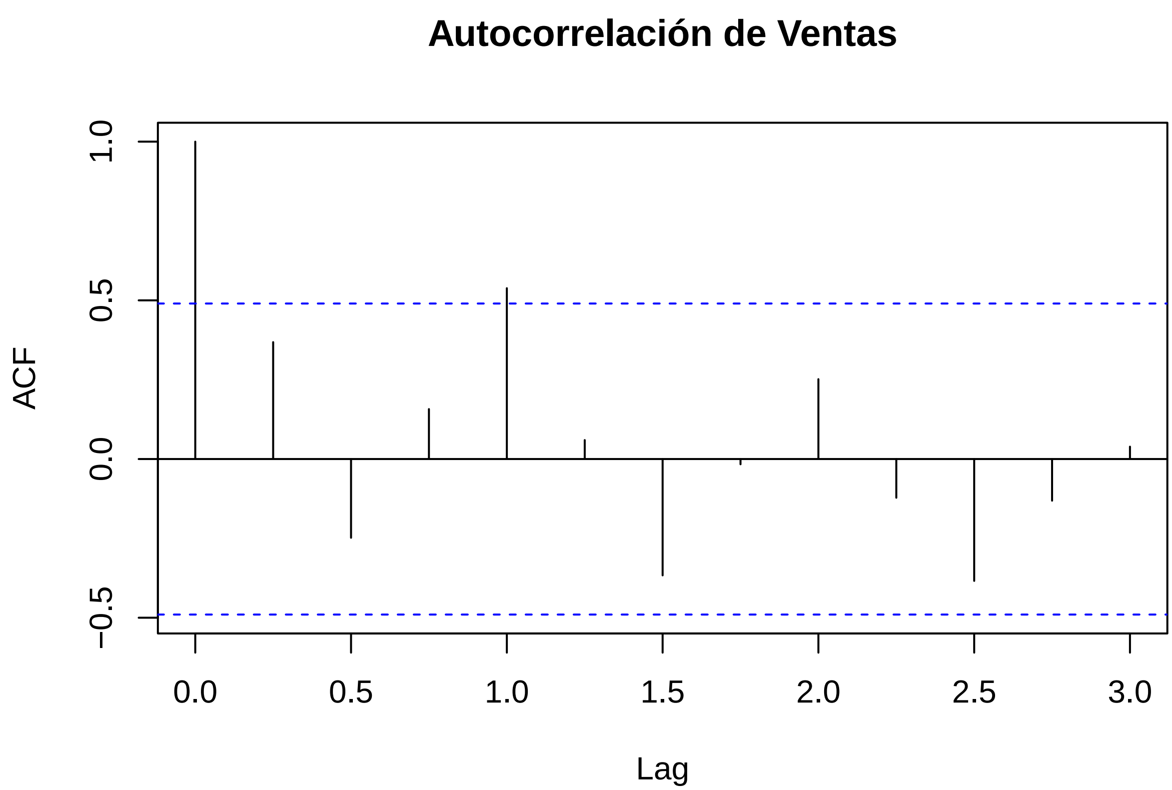
```
# Prueba de Dickey-Fuller aumentada
adf.test(ventas_ts)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  ventas_ts
## Dickey-Fuller = -2.7111, Lag order = 2, p-value = 0.3015
## alternative hypothesis: stationary
```

con un umbral de 0.05 para el p-value no podemos rechazar la hipotesis nula de no estacionariedad por lo que tendremos que eliminarla o diferenciarla.

hacemo un gráfico de autocorrelación

```r
acf(ventas_ts, main = "Autocorrelación de Ventas")
```

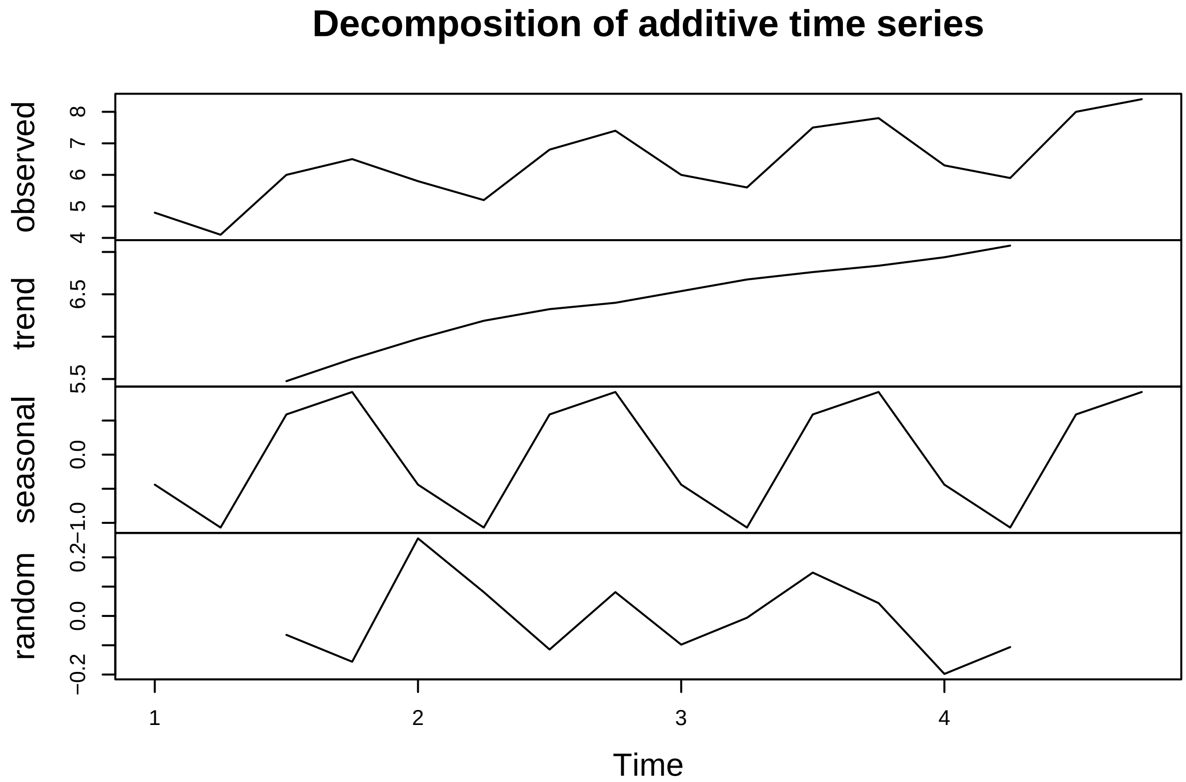


El grafico sugiere que la serie temporal tiene una dependencia fuerte en el primer lag, pero esta dependencia se reduce rápidamente. La presencia de autocorrelación significativa en el primer lag y la caída en los siguientes indican que la serie probablemente tiene una tendencia, aunque claramente por la prueba anterior sabemos que no es una serie estacionaria.

Ahora vamos a probar si el modelo es sumativo o multiplicativo, probando con ambos modelos

primero probamos con la descomposicion sumativa

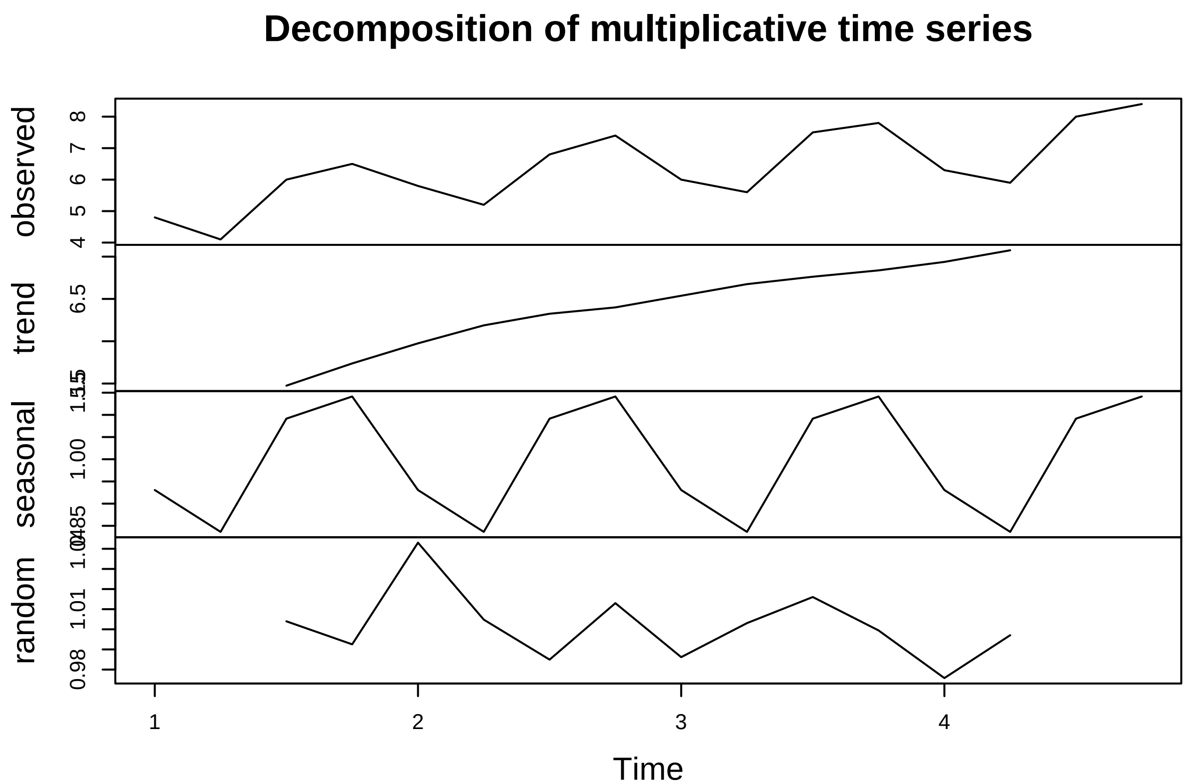
```r
descomposicion_add <- decompose(ventas_ts, type = "additive")
plot(descomposicion_add)
```



análisis de la desomposición sumativa: La descomposición muestra que la serie tiene una tendencia positiva, es decir, las ventas están aumentando a lo largo del tiempo. Además, la serie tiene un componente estacional consistente que se repite periódicamente. El componente de ruido es relativamente bajo, lo que indica que la mayoría de la variabilidad en la serie se explica por la tendencia y la estacionalidad.

luego con la descomposicion multiplicativa

```
descomposicion_mult <- decompose(ventas_ts, type = "multiplicative")
plot(descomposicion_mult)
```



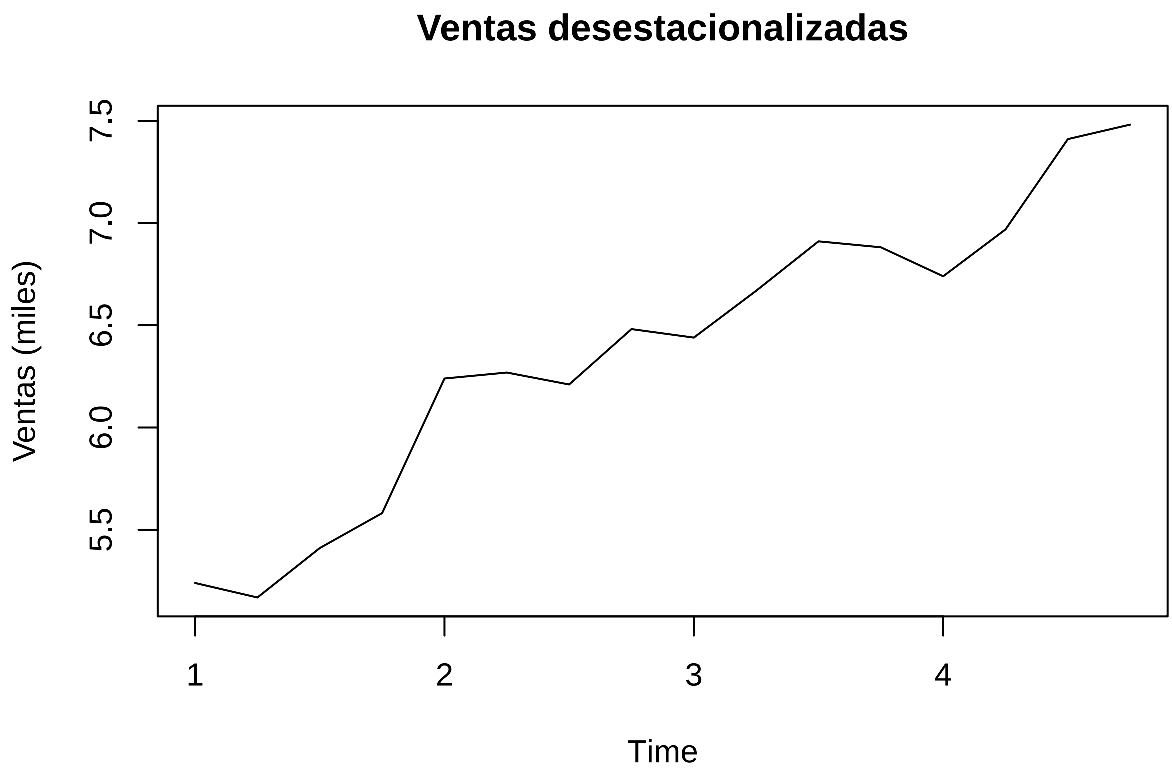
análisis de la desomposición multiplicativa: La descomposición muestra que la serie tiene una tendencia creciente y una estacionalidad regular. Sin embargo, el impacto de la estacionalidad no parece depender significativamente del nivel de la tendencia, ya que el efecto estacional se mantiene constante en amplitud y patrón.

Dado que ambas descomposiciones producen gráficos casi idénticos, podemos concluir que la serie se comporta de manera casi lineal en su estacionalidad y tendencia, y el efecto proporcional (multiplicativo) no es tan pronunciado.

#Calcula los índices estacionales y grafica la serie desestacionalizada

```r
# Índices estacionales y serie desestacionalizada
indices_estacionales <- descomposicion_add$seasonal
ventas_desestacionalizadas <- ventas_ts - indices_estacionales

# Graficar la serie desestacionalizada
plot(ventas_desestacionalizadas, main = "Ventas desestacionalizadas", ylab = "Ventas (miles)")
```



la serie desestacionalizada refuerza la idea de que la tendencia general en la gráfica es ascendente, lo cual indica un crecimiento constante en las ventas a lo largo de cuatro años. sugiriendo que, sin importar la estacionalidad, la empresa está experimentando un aumento en el volumen de ventas.

#Análisi edl modelo lineal de la tendencia

```
# Crear variable de tiempo
tiempo <- 1:length(ventas_desestacionalizadas)

# Regresión lineal de la tendencia
modelo_lineal_ventas_television <- lm(ventas_desestacionalizadas ~ tiempo)
summary(modelo_lineal_ventas_television)
```

```
##
## Call:
## lm(formula = ventas_desestacionalizadas ~ tiempo)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.2992 -0.1486 -0.0037  0.1005  0.3698
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.13917    0.10172   50.52  < 2e-16 ***
## tiempo       0.14613    0.01052   13.89  1.4e-09 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.194 on 14 degrees of freedom
## Multiple R-squared:  0.9324, Adjusted R-squared:  0.9275
## F-statistic:   193 on 1 and 14 DF,  p-value: 1.399e-09
```

el p-value del modelo es de 1.399e-09, lo cual es extremadamente bajo, indicando que el modelo es globalmente significativo. En el análisis de los coeficientes se muestra que ambos son altamente significativos: el intercepto es de 5.13917 con un p-value menor que 2e-16, indicando su importancia en el modelo, y el coeficiente de la variable tiempo es 0.14613 con un p-value de 1.4e-09, lo que sugiere que, en promedio, las ventas desestacionalizadas aumentan en 0.14613 por cada unidad de tiempo, implicando una tendencia de crecimiento positiva a lo largo del tiempo. El coeficiente de determinación es 0.9324, lo cual indica que el modelo explica aproximadamente el 93.24% de la variabilidad en las ventas desestacionalizadas. Ahora haremos un análisis de los residuos para comprobar la linealidad del modelo.

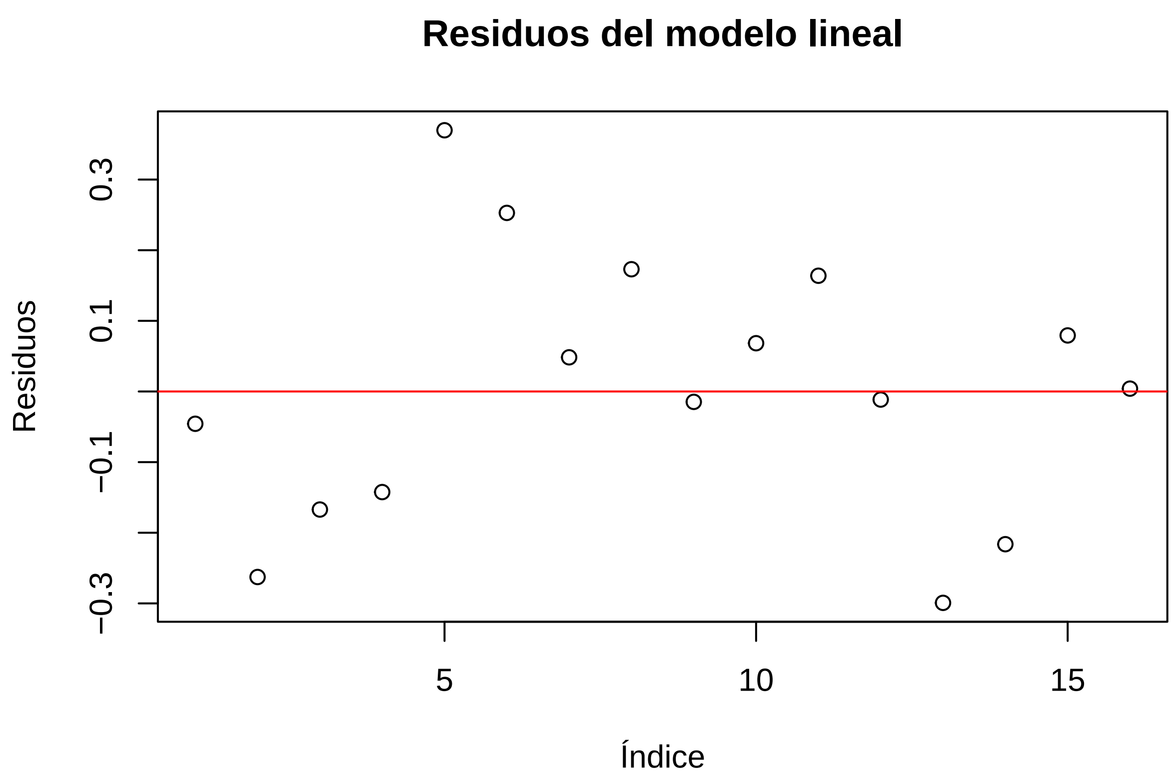
#Análisis de los residuos

primero sustraemos los residuos del modelo

```r
# Obtener los residuos del modelo
residuos_television <- residuals(modelo_lineal_ventas_television)
```

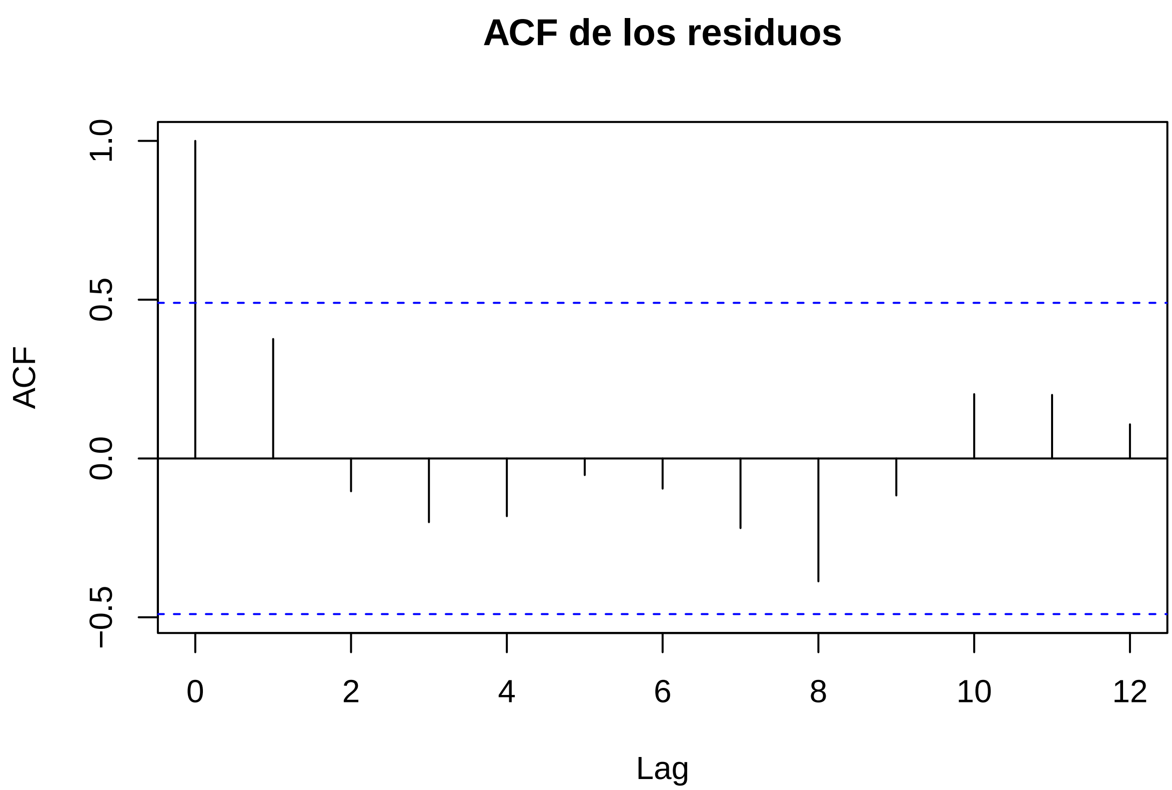
ahora graficamos los residuos para ver tendencias

```r
# Graficar los residuos
plot(residuos_television, main = "Residuos del modelo lineal", ylab = "Residuos", xlab = "Índice")
abline(h = 0, col = "red")
```



podemos ver que los residuos parecen estar distribuidos de forma aleatoria, pero esto no es suficiente, vamos a hacer una prueba de correlación

```r
# Gráfico de autocorrelación de los residuos
acf(residuos_television, main = "ACF de los residuos")
```



podemos observar una autocorrelación significativa en el lag 1. Esto indica que los residuos de un período están correlacionados con los residuos del período anterior, lo cual sugiere una dependencia temporal entre ellos, no es buena señal, pero recordando que el modelo no es perfecto (aunque el 93% explicado sigue siendo un modelo bueno) es esperado. Haremos una ultima prueba, la de normalidad

```r
shapiro.test(residuos_television)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuos_television
## W = 0.97816, p-value = 0.9473
```

el p-value es significativamente mayor que el umbral que generalmente se usa de 0.05, por lo que no hay evidencia suficiente para rechazar la hipótesis nula de normalidad. dicho de otra forma, podemos asumir que los residuos siguen una distribución normal, y por ende el modelo es adecuado según esta prueba ya que recordemos que los modelos de regresión lineal suponen que los residuos siguen una distribución normal. No olvidamos el resultado de la prueba anterior por lo que podría haber un modelo que explique mejor las ventas a lo largo del tiempo.

#Cálculo del CME y el EPAM

```r
# Inicialización de los vectores de pronóstico y errores
p <- rep(NA, length(ventas))
e <- rep(NA, length(ventas))

# Calcular pronósticos de promedios móviles simples de 3 períodos
for (i in 1:(length(ventas) - 3)) {
  p[i + 3] <- (ventas[i] + ventas[i + 1] + ventas[i + 2]) / 3
  e[i + 3] <- p[i + 3] - ventas[i + 3]
}

# Calcular CME y EPAM usando promedios móviles simples
CME_promedios_simples <- mean(e^2, na.rm = TRUE)
EPAM_promedios_simples <- mean(abs(e), na.rm = TRUE)

CME_promedios_simples
```

```
## [1] 1.378889
```

```r
EPAM_promedios_simples
```

```
## [1] 1.1
```

#Exploración de un mejor modelo

vamos a probar con un modelo cuadratico pero no creo que funcione

```r
modelo_cuadratico_ventas_television <- lm(ventas_desestacionalizadas ~ tiempo + I(tiempo^2))
summary(modelo_cuadratico_ventas_television)
```

```
##
## Call:
## lm(formula = ventas_desestacionalizadas ~ tiempo + I(tiempo^2))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.30333 -0.13440 -0.01928  0.11368  0.33301
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.930833   0.155679  31.673 1.08e-13 ***
## tiempo       0.215572   0.042149   5.115 0.000199 ***
## I(tiempo^2) -0.004085   0.002410  -1.695 0.113918
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1822 on 13 degrees of freedom
## Multiple R-squared:  0.9446, Adjusted R-squared:  0.9361
## F-statistic: 110.8 on 2 and 13 DF,  p-value: 6.805e-09
```

de manera global el modelo explica el 94.46% de la variabilidad en las ventas desestacionalizadas,sin embargo el termino cuadratico no es estadísticamente significativo por lo que simplemente podriamos utilizar un modelo de regresión lineal

#Pronostico para el año 5

```r
#Definimos los trimestres del próximo año para el pronóstico
nueva_data <- data.frame(tiempo = seq(length(ventas_desestacionalizadas) + 1, length(ventas_desestaciona

#Calculamos el pronóstico con el modelo lineal
pronostico_lineal <- predict(modelo_lineal_ventas_television, nueva_data)

#Graficamos los datos originales
plot(ventas_ts, xlim = c(1, 5), ylim = range(c(ventas_ts, pronostico_lineal)),
     main = "Pronóstico de Ventas (Lineal)", ylab = "Ventas (miles)", xlab = "Años")

#Graficamos el pronóstico lineal
trimestres_futuros <- 5 + (1:4) / 4  # Puntos para el siguiente año en trimestres
lines(c(time(ventas_ts), trimestres_futuros), c(ventas_ts, pronostico_lineal),
      col = "blue", lty = 2, type = "b", pch = 16)

#Agregamos una leyenda
legend("topleft", legend = c("Datos originales", "Pronóstico lineal"),
       col = c("black", "blue"), lty = c(1, 2), pch = 16)
```