A8-Series de tiempo no estacionarias. Tendencia

A01275465 Carol Arrieta Moreno

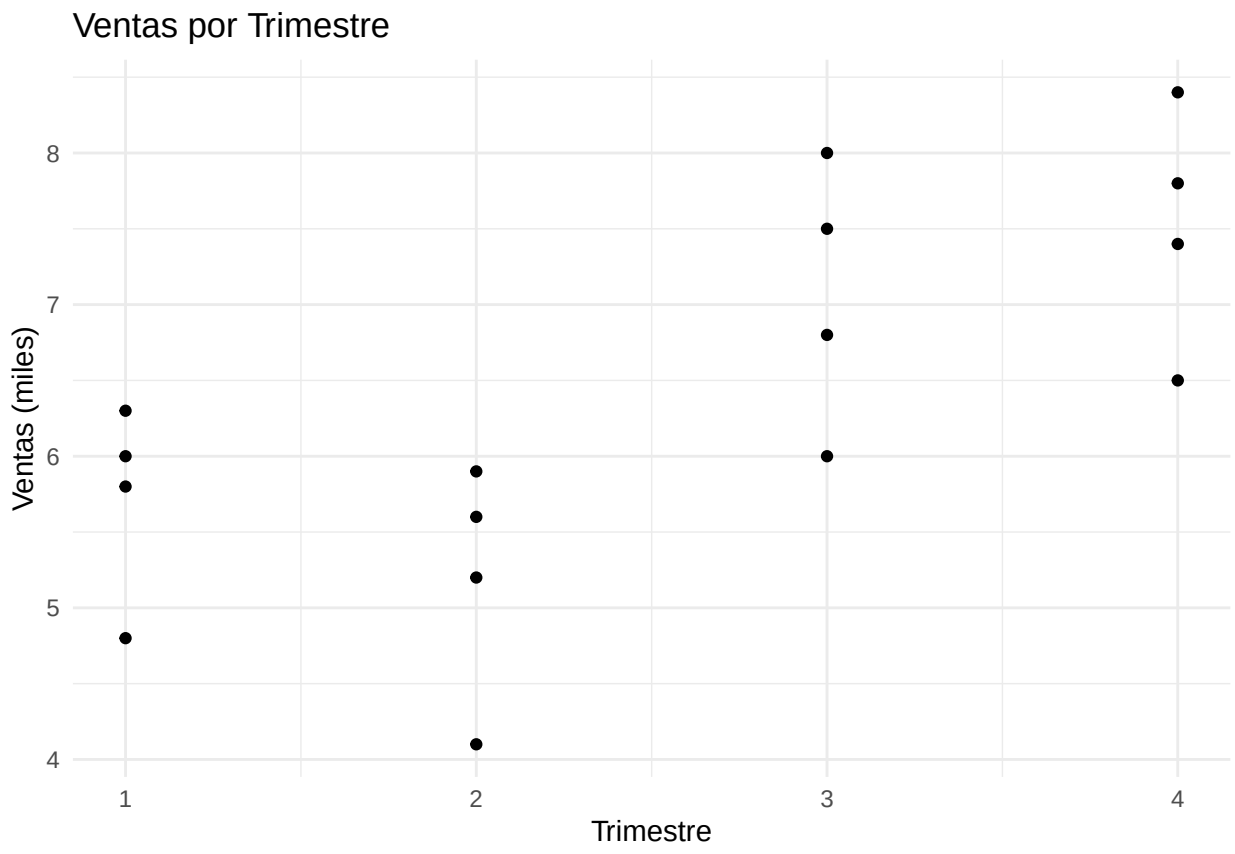
2023-11-17

```
library(ggplot2)
```

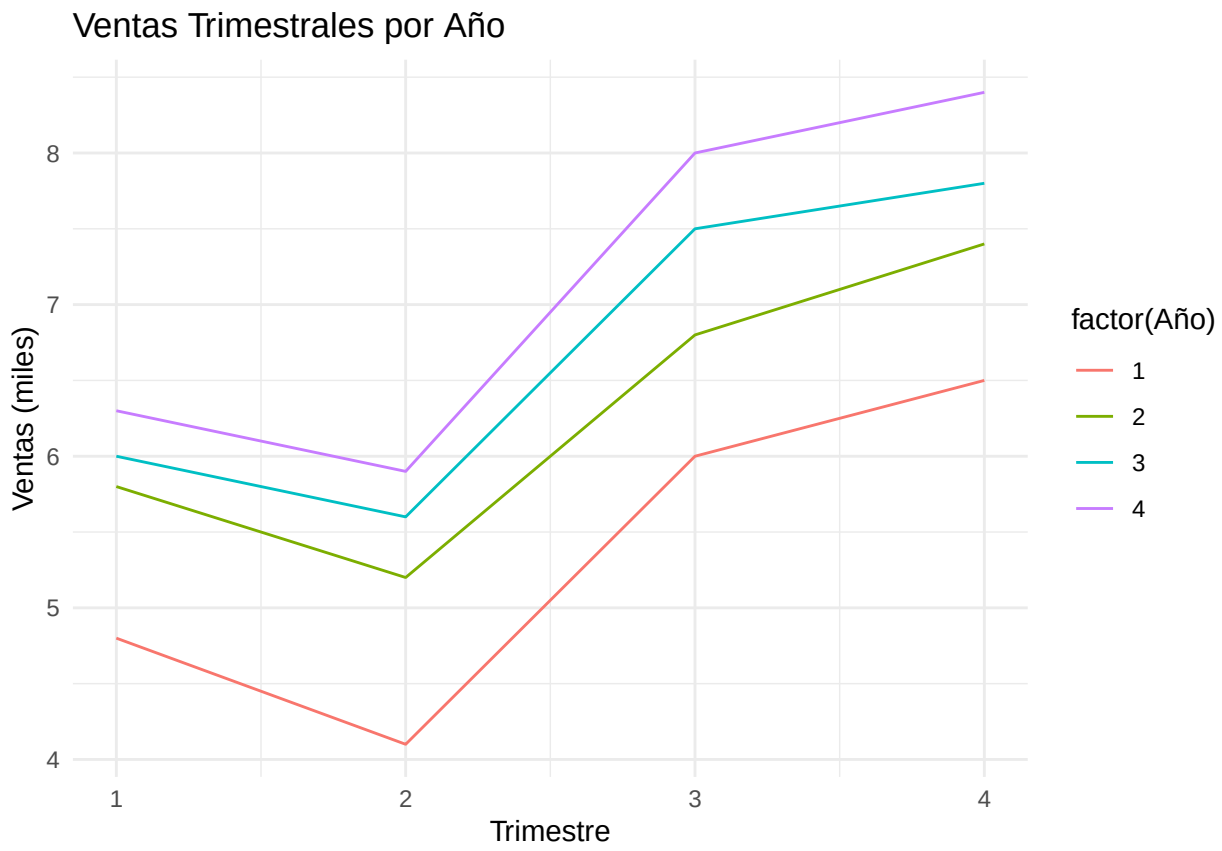
```
## Warning: package 'ggplot2' was built under R version 4.3.2
```

```
# Crear el dataframe con los datos proporcionados
data <- data.frame(
  Año = rep(1:4, each = 4), # Valores de los años
  Trimestre = rep(1:4, 4),  # Valores de los trimestres
  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) # Ventas
)
```

```
# Gráfico de dispersión
ggplot(data, aes(x = Trimestre, y = Ventas)) +
  geom_point() +
  labs(title = "Ventas por Trimestre", x = "Trimestre", y = "Ventas (miles)") +
  theme_minimal()
```



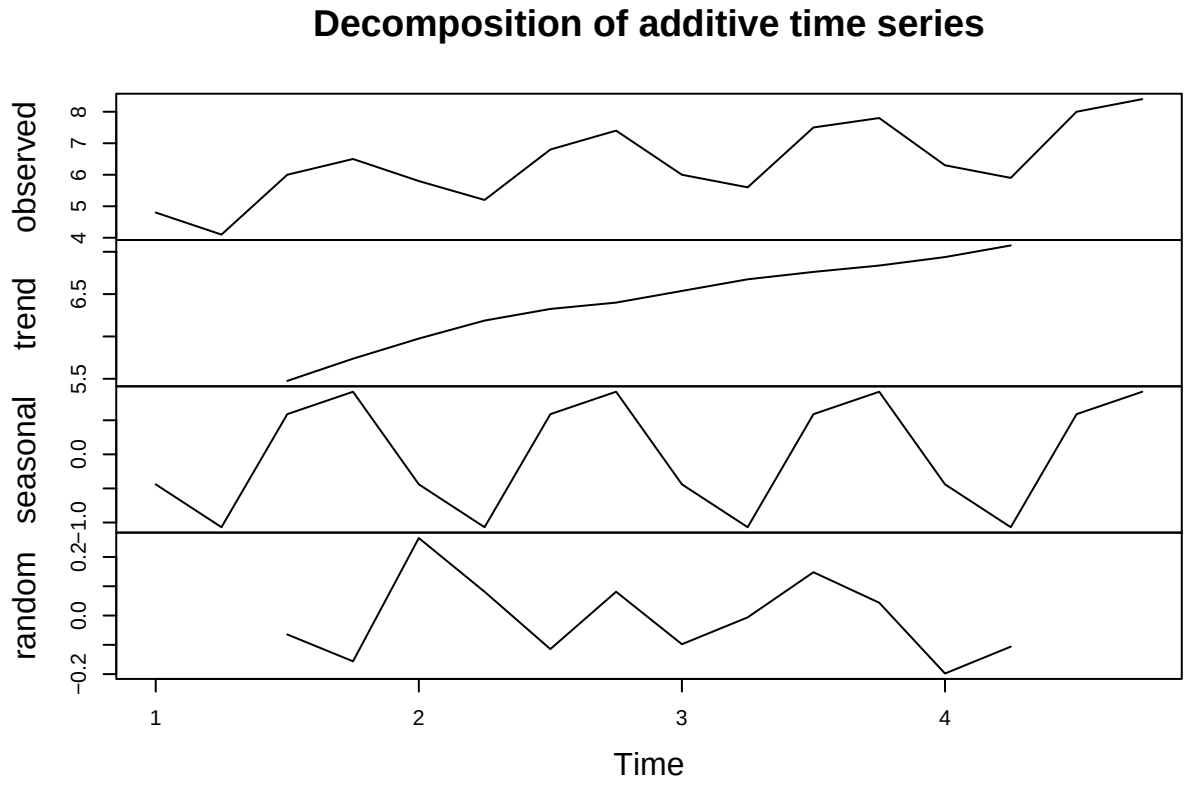

```
# Gráfico de línea para visualizar la serie temporal de ventas
ggplot(data, aes(x = Trimestre, y = Ventas, group = Año, color = factor(Año))) +
  geom_line() +
  labs(title = "Ventas Trimestrales por Año", x = "Trimestre", y = "Ventas (miles)") +
  theme_minimal()
```




```r
# Convertir los datos a una serie temporal
ts_data <- ts(data$Ventas, frequency = 4)  # 4 trimestres por año

# Realizar la descomposición de la serie temporal
decomposed_ts <- decompose(ts_data)

# Gráfico de la descomposición
plot(decomposed_ts)
```




```r
tiempo <- 1:length(data$Ventas)

# Crear un dataframe con los datos
datos <- data.frame(tiempo <- tiempo, data$Ventas <- data$Ventas)

# Realizar la regresión lineal de las ventas desestacionalizadas vs tiempo
modelo <- lm(data$Ventas ~ tiempo, data = datos)

# Resumen del modelo
summary(modelo)
```

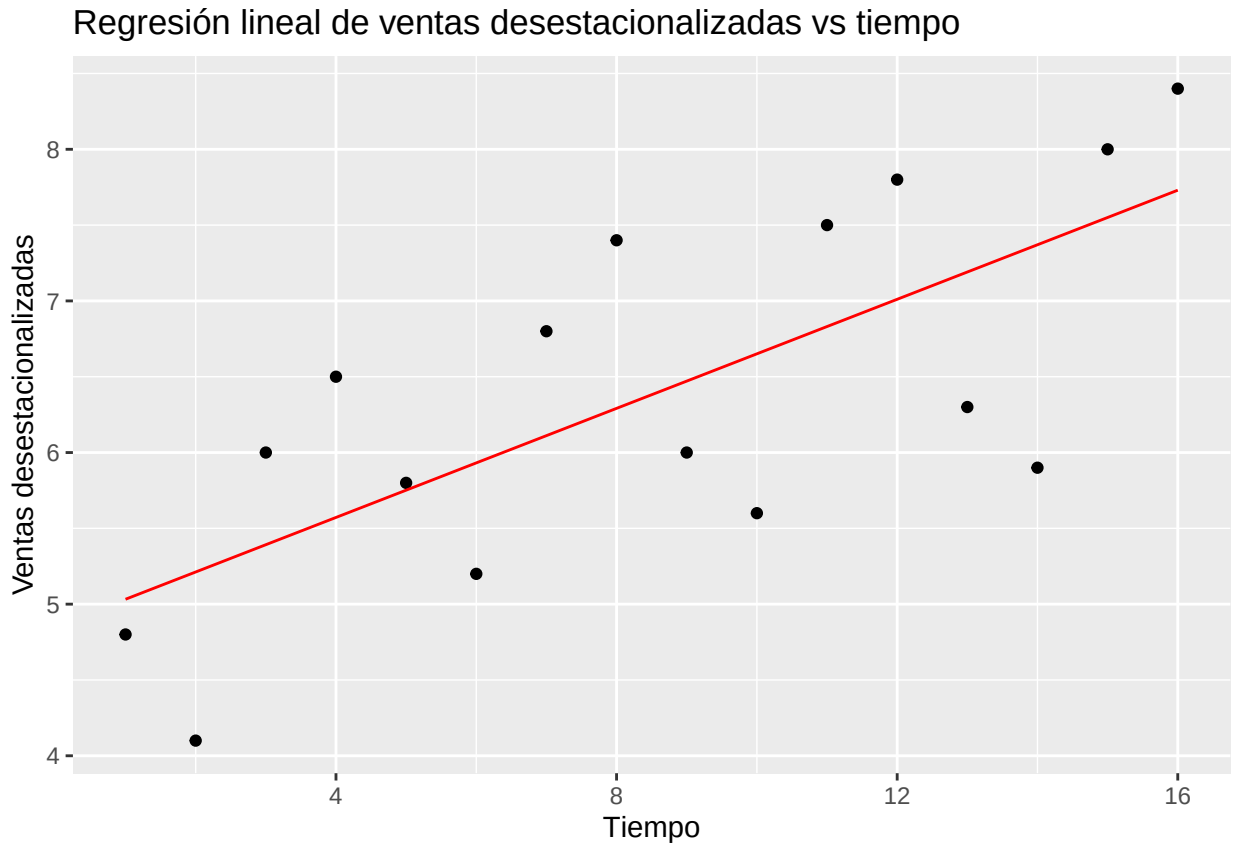
```
##
## Call:
## lm(formula = data$Ventas ~ tiempo, data = datos)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.4704 -0.7714  0.2490  0.6745  1.1087
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.85250    0.45987  10.552 4.78e-08 ***
## tiempo       0.17985    0.04756   3.782  0.00202 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.8769 on 14 degrees of freedom
## Multiple R-squared:  0.5053, Adjusted R-squared:   0.47
## F-statistic:  14.3 on 1 and 14 DF,  p-value: 0.002023
```

```r
# Obtener la línea de regresión
linea_regresion <- data.frame(tiempo = tiempo, ventas_pred = predict(modelo))

# Gráfico de las ventas desestacionalizadas y la línea de regresión
library(ggplot2)

ggplot(datos, aes(x = tiempo, y = data$Ventas)) +
  geom_point() +  # Puntos de las ventas desestacionalizadas
  geom_line(data = linea_regresion, aes(x = tiempo, y = ventas_pred), color = "red") +  # Línea de regr
  labs(title = "Regresión lineal de ventas desestacionalizadas vs tiempo", x = "Tiempo", y = "Ventas des
```




```r
# Resumen del modelo
summary(modelo)
```

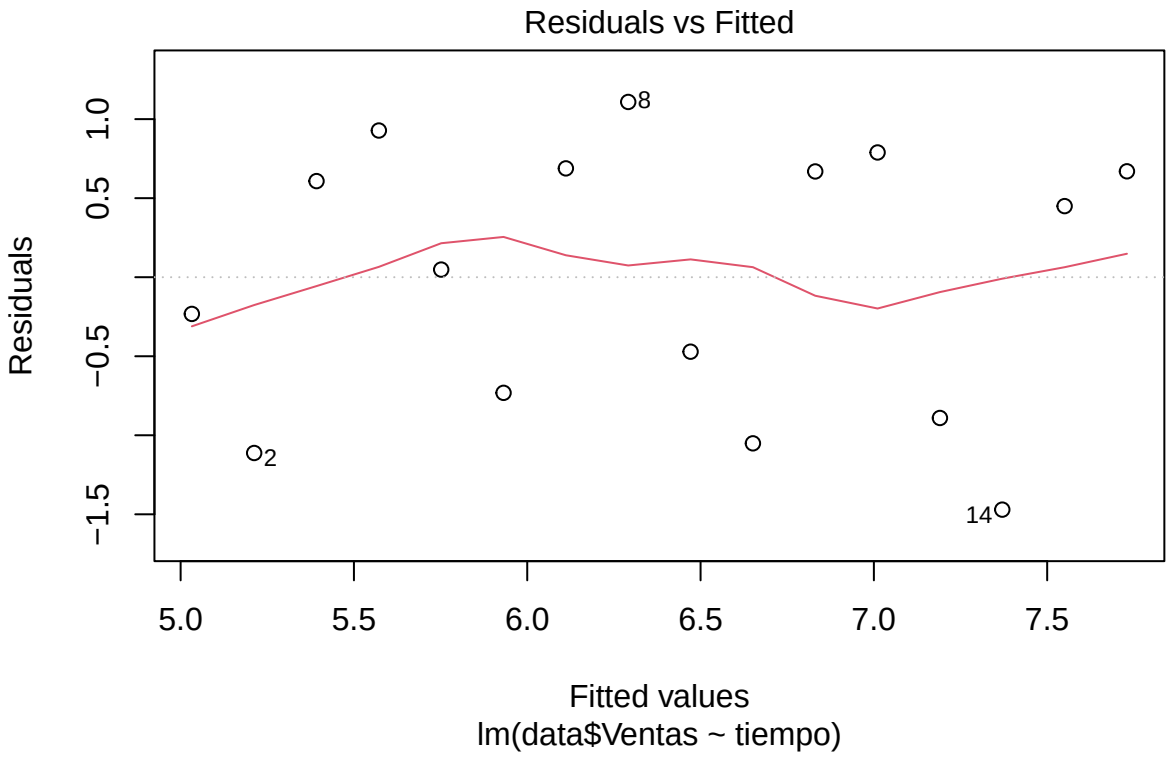
```
##
## Call:
## lm(formula = data$Ventas ~ tiempo, data = datos)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

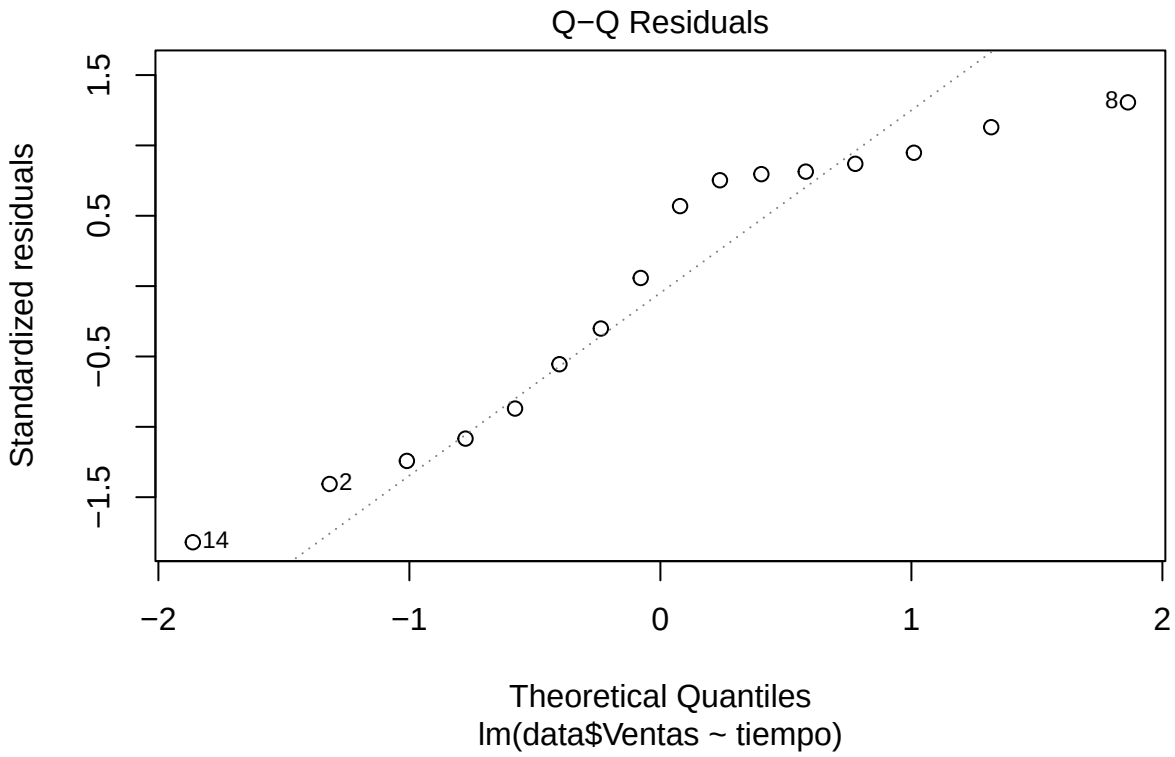
```
## -1.4704 -0.7714  0.2490  0.6745  1.1087
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.85250    0.45987  10.552 4.78e-08 ***
## tiempo       0.17985    0.04756   3.782  0.00202 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8769 on 14 degrees of freedom
## Multiple R-squared:  0.5053, Adjusted R-squared:   0.47
## F-statistic:  14.3 on 1 and 14 DF,  p-value: 0.002023
```

```
# Análisis de la varianza (ANOVA)
anova(modelo)
```

```
## Analysis of Variance Table
##
## Response: data$Ventas
##           Df Sum Sq Mean Sq F value   Pr(>F)
## tiempo     1 10.998  10.998  14.301 0.002023 **
## Residuals 14 10.766   0.769
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Gráficos de los residuos
plot(modelo, which = c(1, 2))  # 1 para residuos vs tiempo, 2 para residuos vs valores ajustados
```


Residuals vs Fitted
Residuals
1.0
0.5
−0.5
−1.5
8
2
14
5.0
5.5
6.0
6.5
7.0
7.5
Fitted values
lm(data$Ventas ~ tiempo)




```
# Prueba de normalidad de los residuos
shapiro.test(residuals(modelo))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(modelo)
## W = 0.90452, p-value = 0.09498
```

```
# Suponiendo que ya tienes el modelo lineal ajustado y los datos desestacionalizados
ventas_desest_pred <- predict(modelo)  # Obtener las predicciones del modelo

# Calcular el Error Cuadrático Medio (CME)
CME <- sum((data$Ventas - ventas_desest_pred)^2) / length(data$Ventas)

# Calcular el Error Porcentual Absoluto Medio (EPAM)
EPAM <- mean(abs((data$Ventas - ventas_desest_pred) / data$Ventas)) * 100

# Mostrar los resultados
print(paste("CME:", CME))
```

```
## [1] "CME: 0.672897977941177"
```

```r
print(paste("EPAM:", EPAM))
```

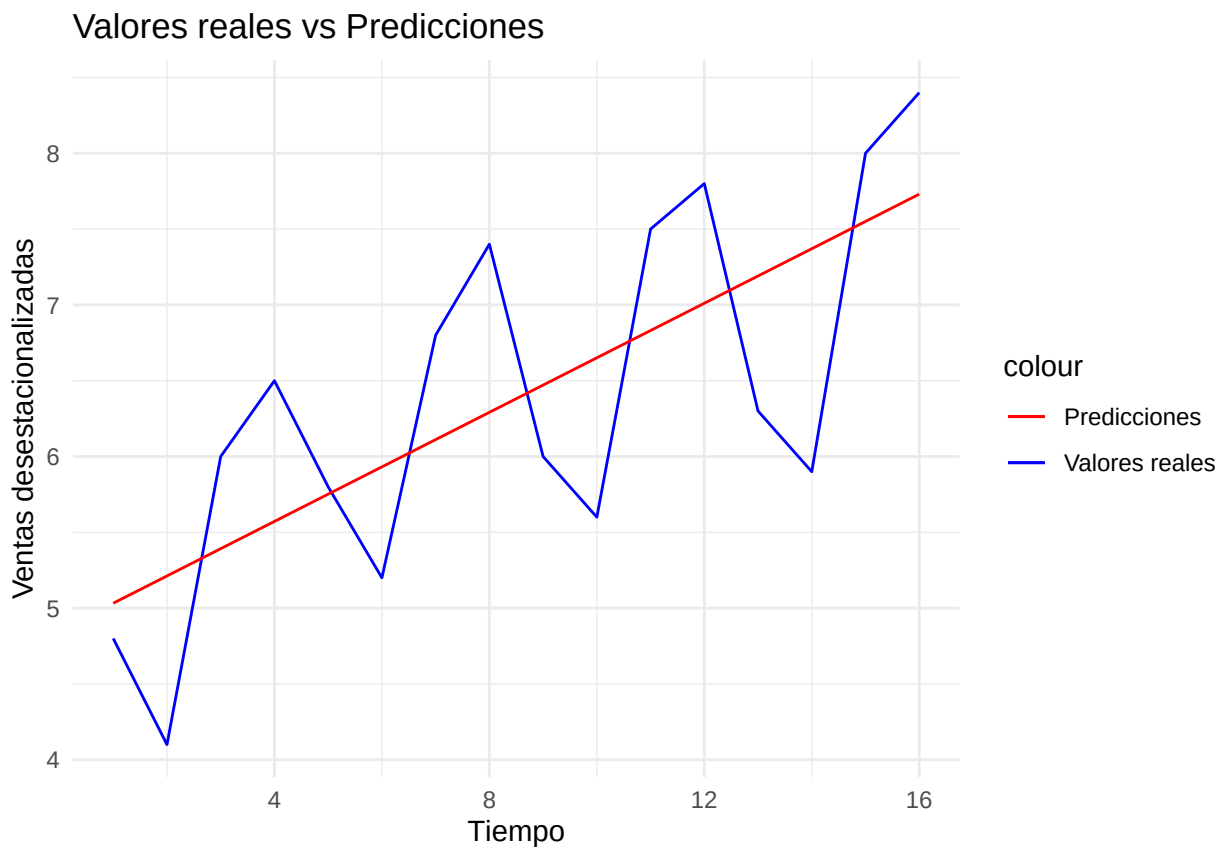
```
## [1] "EPAM: 12.1689712107864"
```

```r
# Suponiendo que ya tienes los datos originales y las predicciones del modelo
# datos contiene la variable de tiempo 'tiempo' y las ventas desestacionalizadas 'ventas_desest'

# Crear un dataframe con las ventas reales y las predicciones
df_predicciones <- data.frame(tiempo = datos$tiempo, Ventas = data$Ventas, Predicciones = ventas_desest

# Gráfico de los valores reales de las ventas y las predicciones vs el tiempo
library(ggplot2)

ggplot(df_predicciones, aes(x = tiempo)) +
  geom_line(aes(y = Ventas, color = "Valores reales")) +
  geom_line(aes(y = Predicciones, color = "Predicciones")) +
  labs(title = "Valores reales vs Predicciones", x = "Tiempo", y = "Ventas desestacionalizadas") +
  scale_color_manual(values = c("Valores reales" = "blue", "Predicciones" = "red")) +
  theme_minimal()
```




```r
# Suponiendo que ya tienes el modelo lineal ajustado y datos previos

# Último trimestre del último año en tus datos
ultimo_trimestre <- tail(datos$tiempo, 1)
```

```r
# Crear una secuencia de tiempo para el próximo año (suponiendo que continúas con la secuencia)
nuevo_tiempo <- seq(ultimo_trimestre + 1, ultimo_trimestre + 4, by = 1)  # Suponiendo 4 trimestres por

# Realizar las predicciones para el próximo año
predicciones_siguiente_anio <- predict(modelo, newdata = data.frame(tiempo = nuevo_tiempo))

# Mostrar las predicciones
print(predicciones_siguiente_anio)
```

```
##        1        2        3        4
## 7.910000 8.089853 8.269706 8.449559
```

Comclusión

Los valores dados van variando, aumentan y disminuyen sin perder el progreso de manera lineal, por lo que el modelo generado nos da buenos resultados en cuanto a los valores en los que se encontraran las ventas en cada año, aunque es complicado que nos brinde el número exacto con una precisión muy alta, el modelo es capaz de entender que la tendencia aumenta, sin embargo los valores en ocasiones disminuyen y en otras ocasiones aumentan, por lo que la precisión no es tan exacta pero el modelo es adecuado y yo diria que si se puede obtener un mejor modelo con usos de diferentes tecnicas, pero en cuanto a regresión lineal creo que es el mejor modelo que se pudo obtener