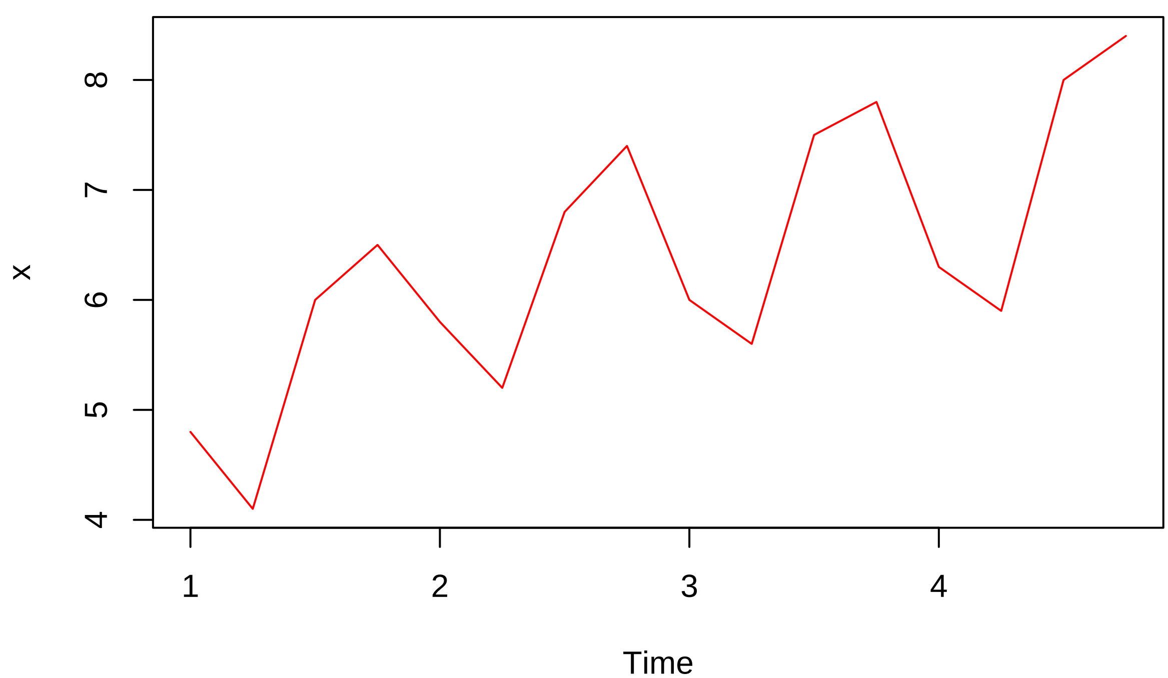
Act7:Series de tiempo

Elías Garza A01284041

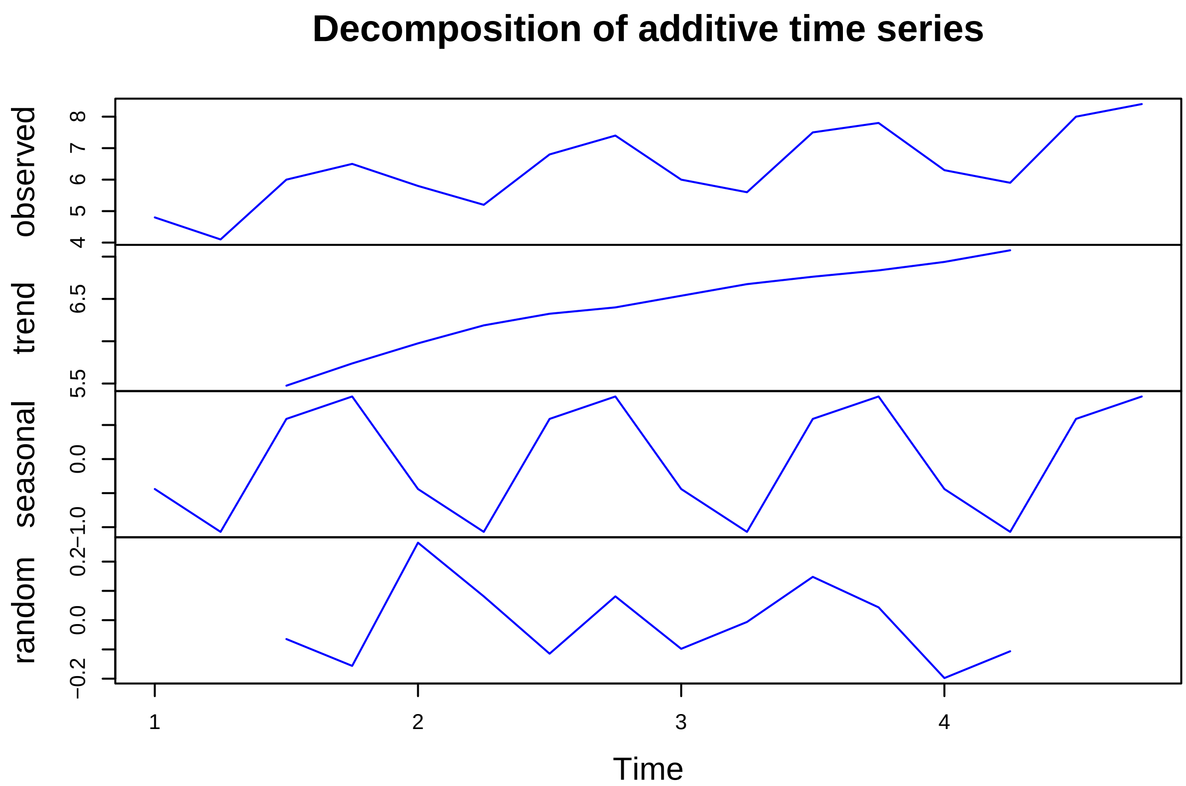
14/11/2023

Problema 1

```r
ser = c(4.8, 4.1, 6, 6.5, 5.8, 5.2, 6.8, 7.4, 6, 5.6, 7.5, 7.8, 6.3, 5.9, 8, 8.4)
x= ts(ser, frequency = 4, start(c(2016,1)))
plot.ts(x, col = "red")
```



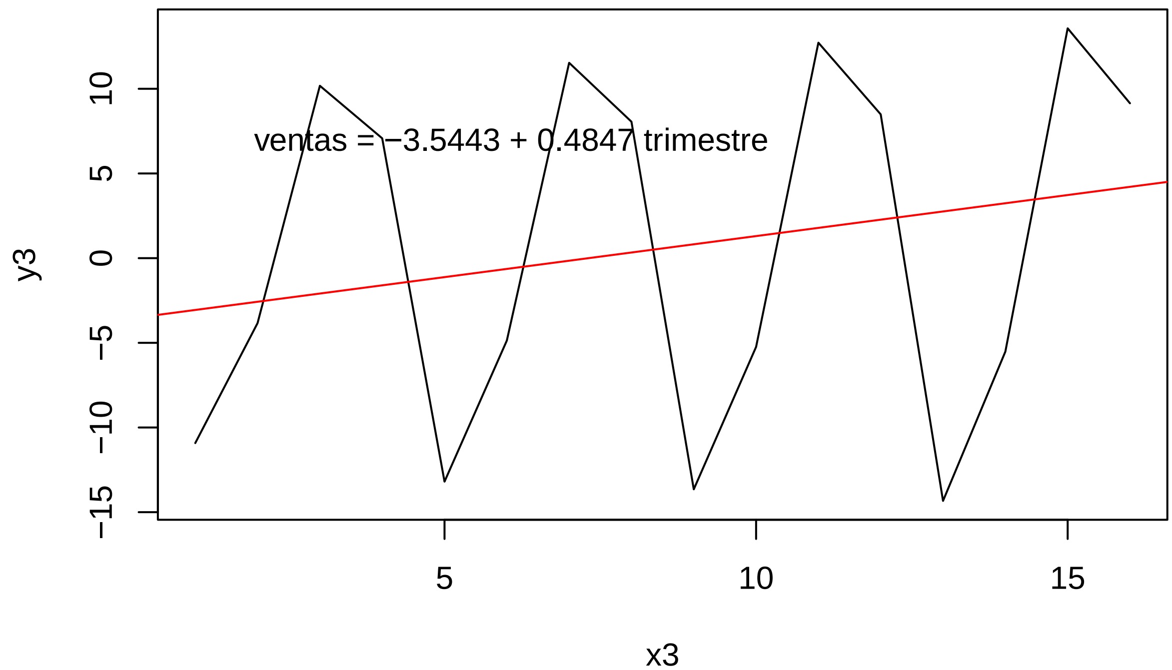

```r
T = decompose(x)
plot(T, col ="blue")
```




```
ventas_desestacionalizadas = (T$x)/(T$seasonal)
x3 = 1:16
y3 = ventas_desestacionalizadas
N3 = lm(y3~x3)
N3
```

```
##
## Call:
## lm(formula = y3 ~ x3)
##
## Coefficients:
## (Intercept)           x3
##     -3.5443       0.4847
```

```
plot(x3, y3, type = "l")
abline(N3, col = "red")
text(6, 7, " ventas = -3.5443 + 0.4847 trimestre")
```



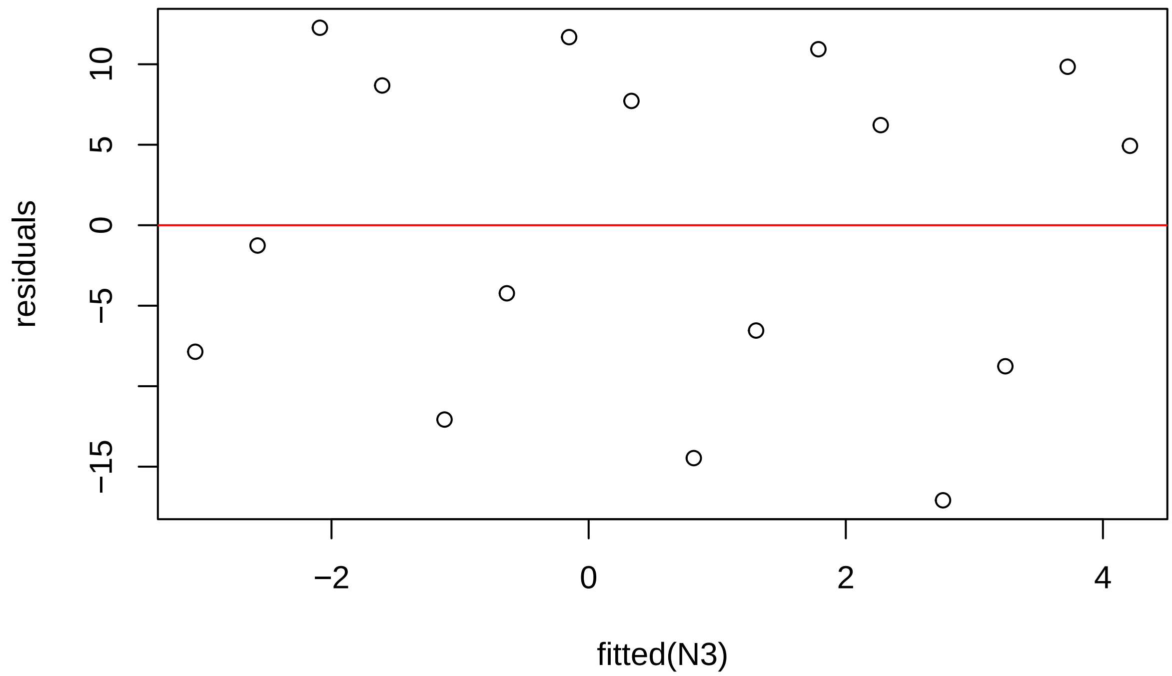

```r
residuals <- residuals(N3)
summary(residuals)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -17.088  -8.085    1.836    0.000    8.971   12.267
```

```r
summary(N3)
```

```
##
## Call:
## lm(formula = y3 ~ x3)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.088  -8.085   1.836   8.971  12.267
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -3.5443     5.5166  -0.642    0.531
## x3            0.4847     0.5705   0.850    0.410
##
## Residual standard error: 10.52 on 14 degrees of freedom
## Multiple R-squared:  0.04902,    Adjusted R-squared:  -0.0189
## F-statistic: 0.7217 on 1 and 14 DF,  p-value: 0.4099
```

```r
plot(fitted(N3), residuals)
abline(h = 0, col = "red")  # adds a horizontal line at 0
```




```r
predictions <- predict(N3, newdata = y3)

library(Metrics)
```

```
## Warning: package 'Metrics' was built under R version 4.1.3
```

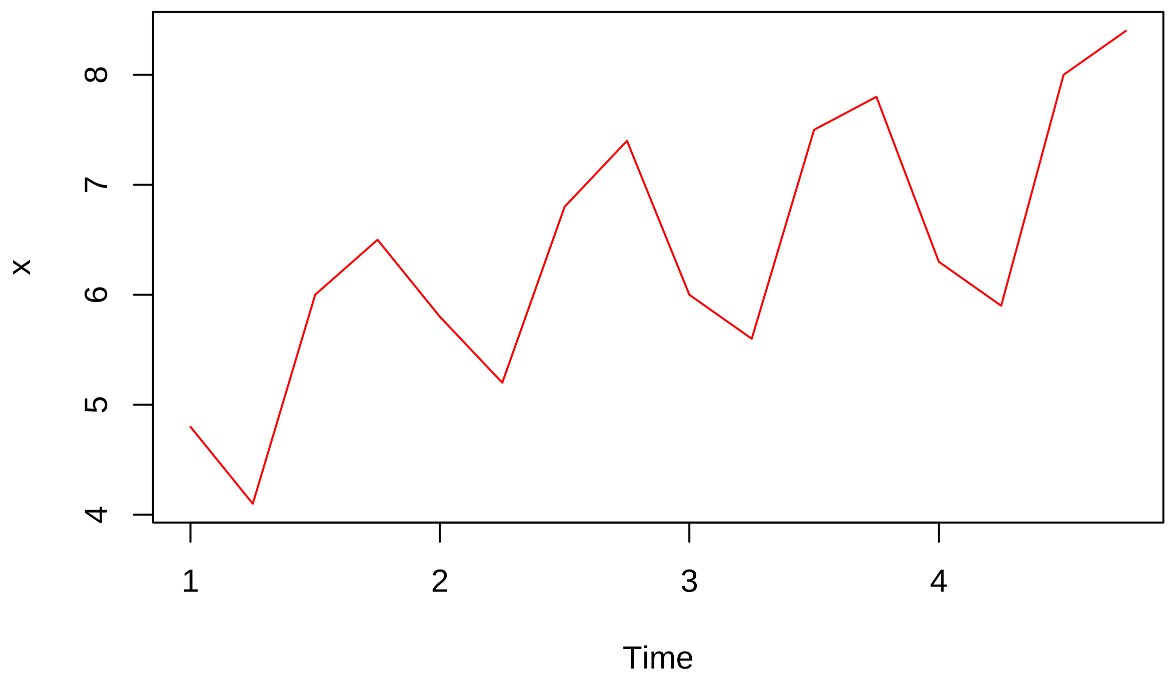
```r
mape<-mape(y3, predictions)
mape
```

```
## [1] 0.9488578
```

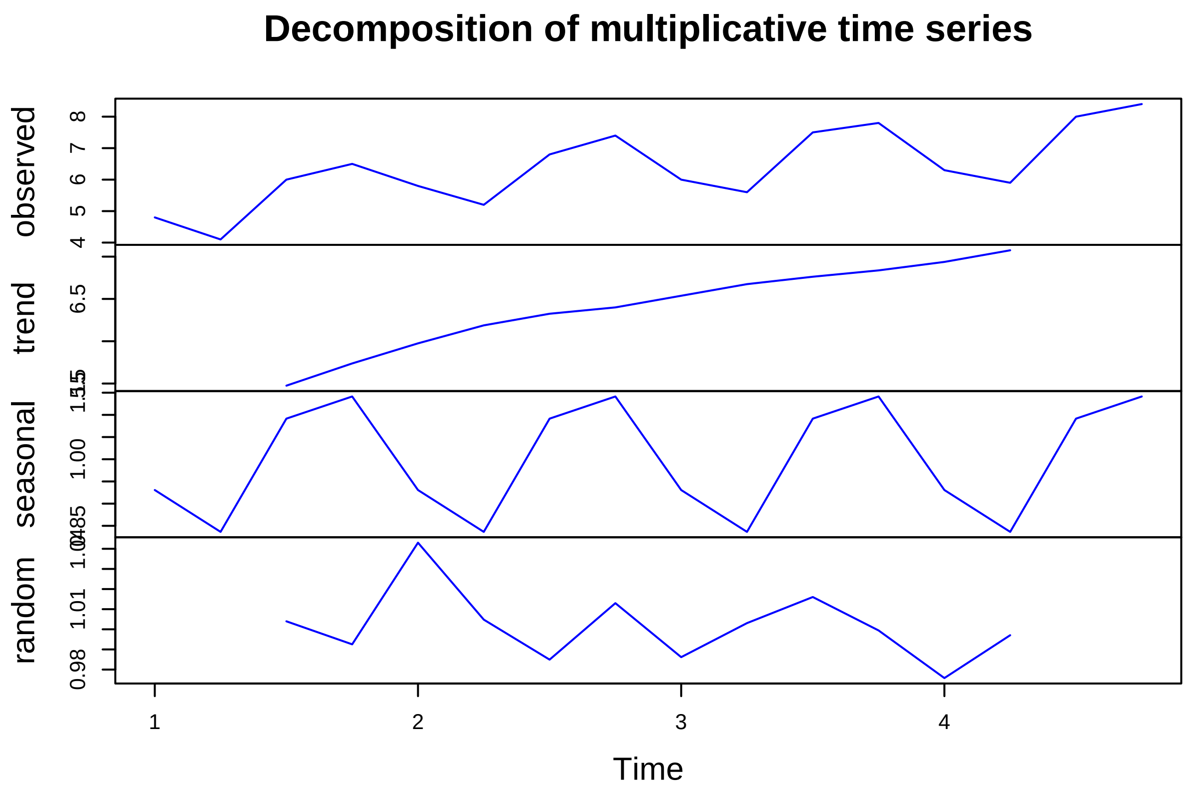
Oservamos que los residuos de nuestro modelo aditivo son bastante altos. Particularmente el modelo tiene un valor p de 0.4099 lo cual es bastante alto por lo que no es recomendado utilizar el modelo. Tambien se puede ver directamente del cálculo de la serie desestacionalizada. Tiene una clara estacionalidad cuando se supone que que deberia quitar la estacionalidad. Además, tenemos un error porcentual del 95% lo cual es simplemente exagerado por lo que intentaremos otro modelo.

Problema 1

```r
ser = c(4.8, 4.1, 6, 6.5, 5.8, 5.2, 6.8, 7.4, 6, 5.6, 7.5, 7.8, 6.3, 5.9, 8, 8.4)
x= ts(ser, frequency = 4, start(c(2016,1)))
plot.ts(x, col = "red")
```



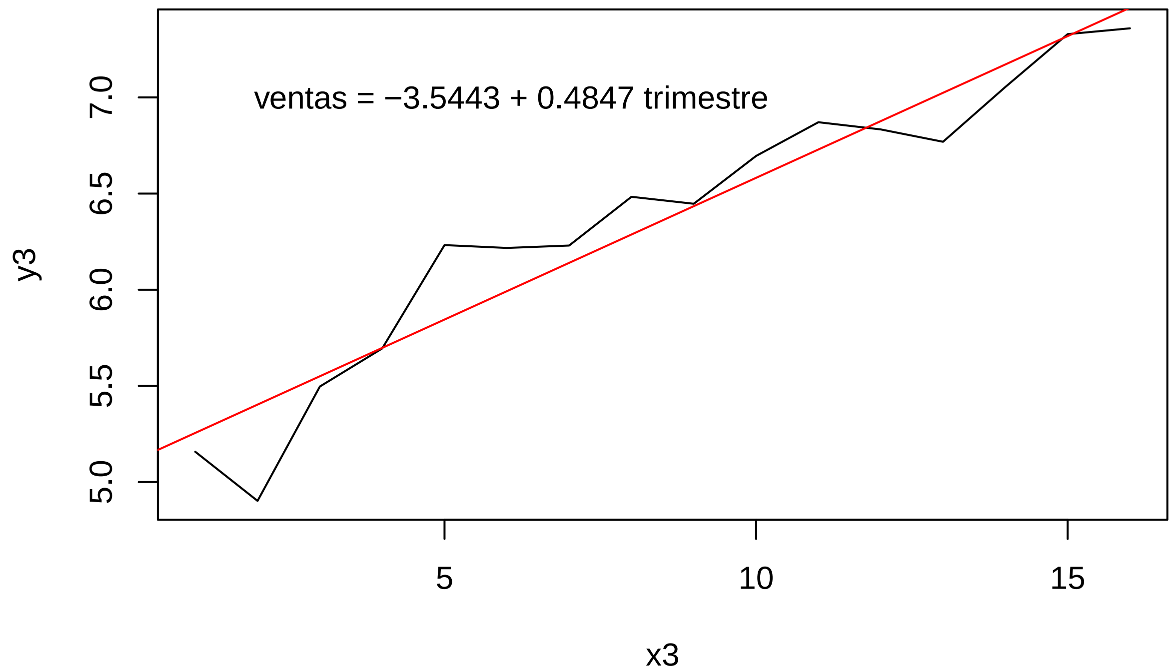

```r
T = decompose(x, type='m')
plot(T, col ="blue")
```




```
ventas_desestacionalizadas = (T$x)/(T$seasonal)
x3 = 1:16
y3 = ventas_desestacionalizadas
N3 = lm(y3~x3)
N3
```

```
##
## Call:
## lm(formula = y3 ~ x3)
##
## Coefficients:
## (Intercept)           x3
##      5.1080       0.1474
```

```
plot(x3, y3, type = "l")
abline(N3, col = "red")
text(6, 7, " ventas = -3.5443 + 0.4847 trimestre")
```




```r
residuals <- residuals(N3)
summary(residuals)
```

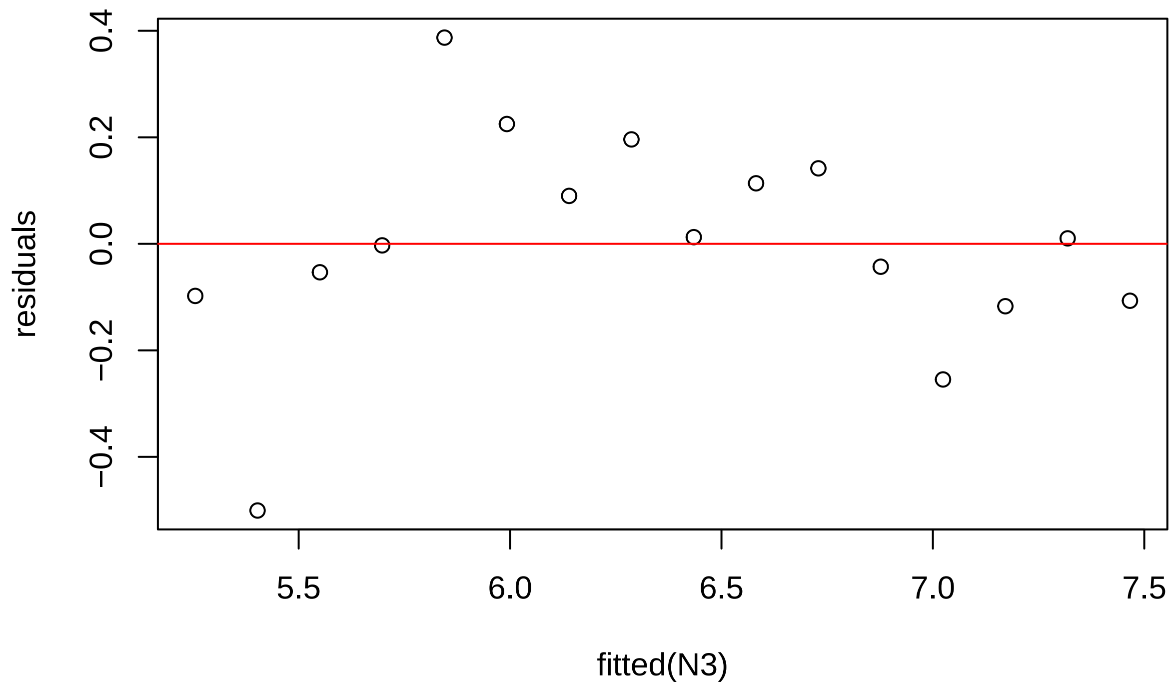
```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -0.500706 -0.100074  0.003699  0.000000  0.120706  0.387173
```

```r
summary(N3)
```

```
##
## Call:
## lm(formula = y3 ~ x3)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.5007 -0.1001  0.0037  0.1207  0.3872
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.10804    0.11171   45.73  < 2e-16 ***
## x3           0.14738    0.01155   12.76 4.25e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.213 on 14 degrees of freedom
```

```
## Multiple R-squared:  0.9208, Adjusted R-squared:  0.9151
## F-statistic: 162.7 on 1 and 14 DF,  p-value: 4.248e-09
```

```
plot(fitted(N3), residuals)
abline(h = 0, col = "red")  # adds a horizontal line at 0
```



Vemos que ahora el valor p es considerablemente más chico y la estacionalidad se ve mucho más reducida.

```
predictions <- predict(N3, newdata = y3)

library(Metrics)
mape<-mape(y3, predictions)
mape
```

```
## [1] 0.02439533
```

Aquí observamos que el error es considerablemente menor con tan solo 2% lo cual es mucho más aceptable. Sin embargo, verenmos algunos otras pruebas de módelos más complejos para estimar la serie de tiempo. Esto con un módelo clásico como lo es un auto-arima.

```
#Training and making forecast  using AUTO-Arima
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.1.3
```

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

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

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

```
library(readr)
```

```
## Warning: package 'readr' was built under R version 4.1.3
```

```
library(ggplot2)
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 4.1.3

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

##
## Attaching package: 'forecast'

## The following object is masked from 'package:Metrics':
##
##     accuracy
```

```
library(forecastHybrid)
```

```
## Loading required package: thief
```

```
library(gbm)
```

```
## Loaded gbm 2.1.8
```

```
library(nnfor)
```

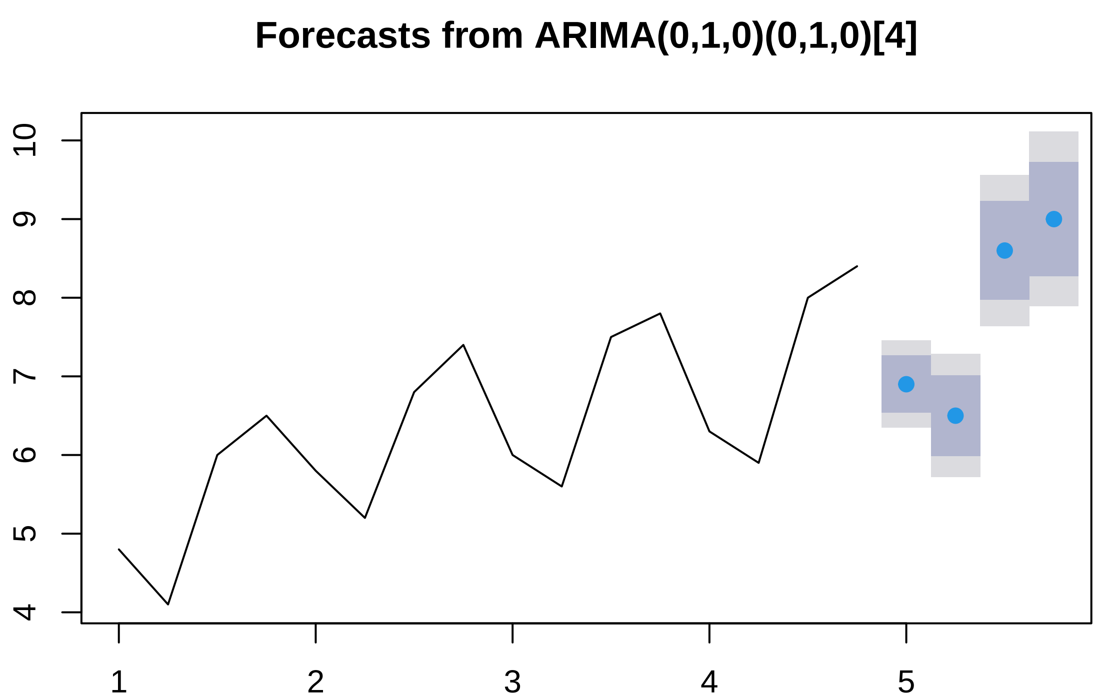
```
## Warning: package 'nnfor' was built under R version 4.1.3

## Loading required package: generics

## Warning: package 'generics' was built under R version 4.1.3

##
## Attaching package: 'generics'
```

```
## The following object is masked from 'package:dplyr':
##
##     explain

## The following object is masked from 'package:Metrics':
##
##     accuracy

## The following objects are masked from 'package:base':
##
##     as.difftime, as.factor, as.ordered, intersect, is.element, setdiff,
##     setequal, union
```

```
sarima_ts<-auto.arima(x)
sarima_ts
```

```
## Series: x
## ARIMA(0,1,0)(0,1,0)[4]
##
## sigma^2 = 0.08001:  log likelihood = -1.72
## AIC=5.43   AICc=5.88   BIC=5.83
```

```
arima_model<-forecast::forecast(sarima_ts,h=4)
plot(arima_model)
```

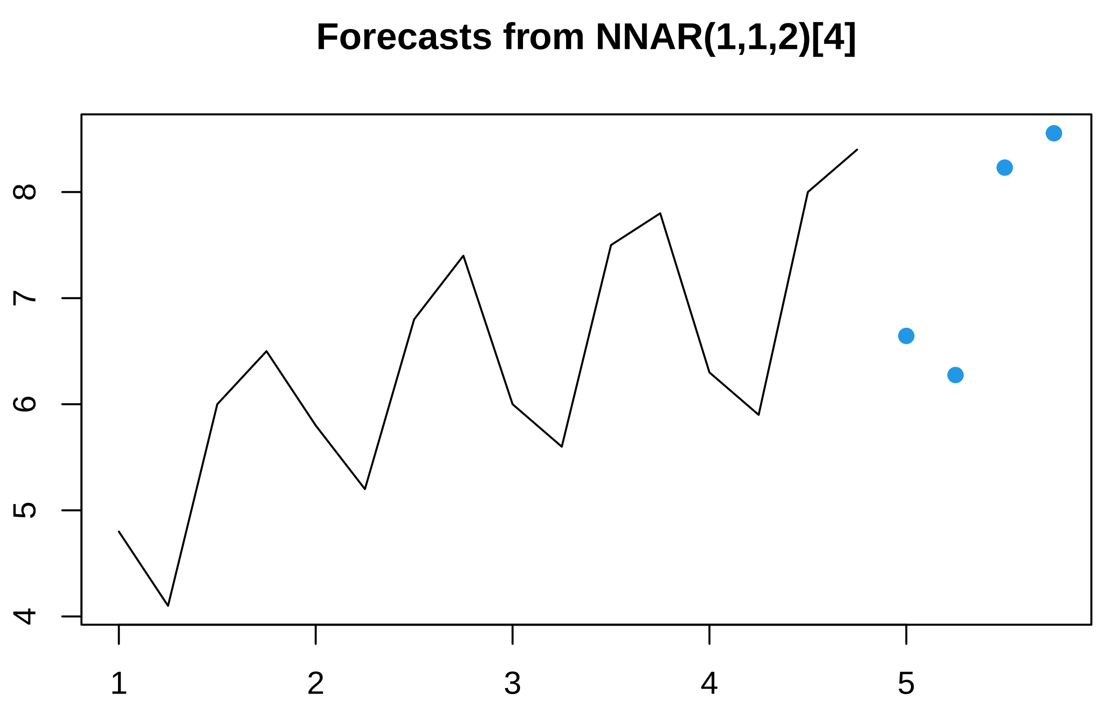


Ahora una pequeña red neuronal simple con consideraciones de estacionalidad.

```r
fit<-nnetar(x,repeats=40,lambda=NULL)
fit
```

```
## Series: x
## Model:  NNAR(1,1,2)[4]
## Call:   nnetar(y = x, repeats = 40, lambda = NULL)
##
## Average of 40 networks, each of which is
## a 2-2-1 network with 9 weights
## options were - linear output units
##
## sigma^2 estimated as 0.01144
```

```r
nn_model<-forecast::forecast(fit,h=4)


#Plotting prediction and testing data (red for testing data)
plot(nn_model)
```



Vemos que ambas predicciones siguen la linea de tendencia de manera considerable.

Un problemilla más

Vamos a calcular los promedios móviles centrados utilizando la librería zoo

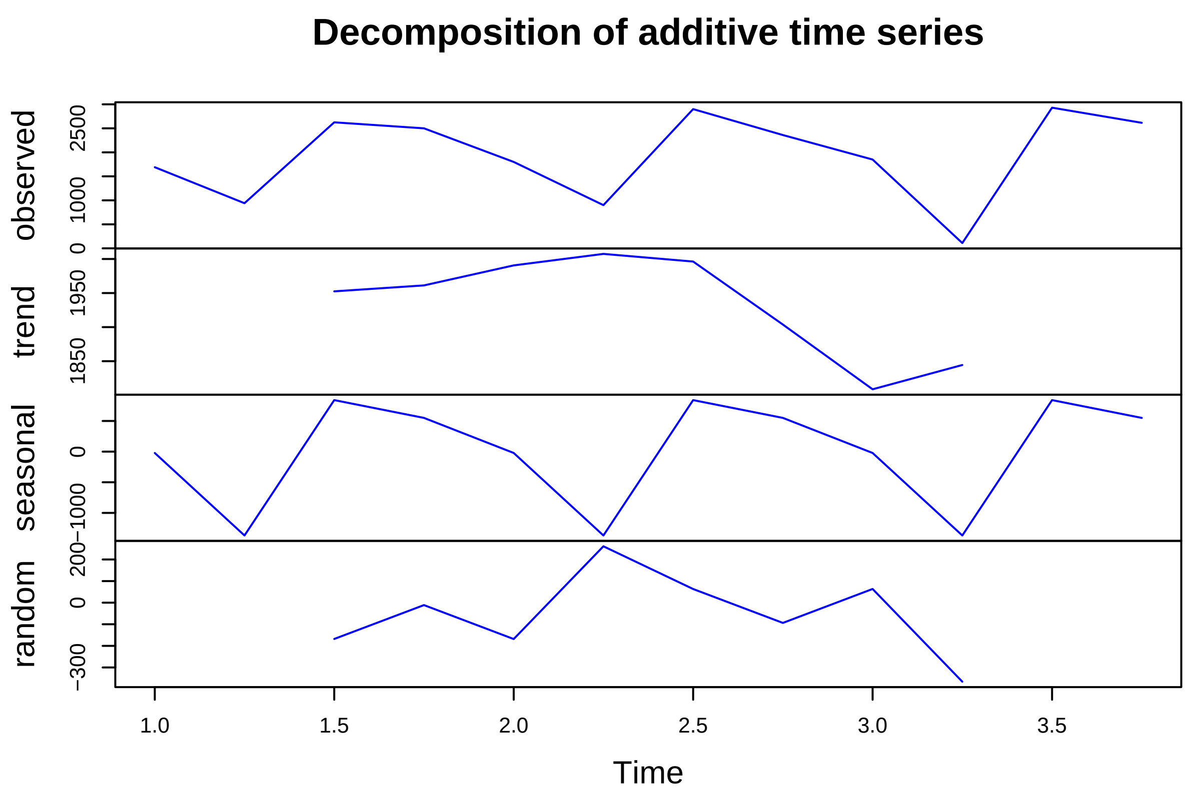
```
library(zoo)
```

```
## Warning: package 'zoo' was built under R version 4.1.3

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
ser = c(1690, 940, 2625, 2500, 1800, 900, 2900, 2360, 1850, 110, 2930, 2615)
# Calculate the 4-period moving average
rollmean(ser, 4, fill = NA, align = "center")
```

```
##  [1]      NA 1938.75 1966.25 1956.25 2025.00 1990.00 2002.50 1805.00 1812.50
## [10] 1876.25      NA      NA
```

```
x = ts(ser, frequency = 4)

T = decompose(x)
plot(T, col ="blue")
```



```
T$seasonal
```

```
##         Qtr1       Qtr2       Qtr3       Qtr4
## 1   -22.1875 -1368.4375   840.6250   550.0000
## 2   -22.1875 -1368.4375   840.6250   550.0000
## 3   -22.1875 -1368.4375   840.6250   550.0000
```

Observamos que el componente estacional más grande es por una cantidad considerable el del tercer trimestre siendo este de 840.62 lo cual tiene bastante sentido ya que este trimestre es en el cual ganan más para los 3 años de los cuales tenemos datos.