CONTENIDO

1. EXPLORACIÓN DE PATRONES DE DATOS E INTRODUCCIÓN A LAS TÉCNICAS DE PRONÓSTICOS.

1.1. Estudio de patrones de datos en la serie de tiempo

- El modelo de pronóstico más elaborado fallará si se aplica a datos pocos confiables.
- Cualquier variable integrada con datos recopilados, registrados u observados durante incrementos de tiempos sucesivos se llama *series de tiempo.*

Uno de los pasos más importantes en la selección de un método para pronosticar adecuado con datos de una serie de tiempo es considerar os diferentes tipos de patrones de datos. Existen 4 tipos generales:

1. Horizontal: Cuando los datos reco-pilados en el transcurso del tiempo fluctúan alrededor de un nivel o una media constantes, hay un patrón *horizontal.* Se dice que este tipo de series es *estacionario* en su media.
2. Tendencias: Cuando los datos crecen y descienden en varios períodos, existe un patrón de *tendencia.* La *tendencia* es el componente de largo plazo que representa el crecimiento o el descenso en a serie de tiempo, durante un período extenso.
3. Estacionales: Cuando las observaciones se ven influidas por factores temporales, existe un patrón estaciona.El *componente estacional* se refiere a un patrón de cambio que se repite año trass año.
4. Cíclicos: Cuando las observaciones indican aumentos y caídas que no tienen un período fijo, existe un patrón cíclico. El *componente cíclico* es la fluctuación con forma de onda alrededor de la tendencia y, por lo común, se ve afectada por las condiciones económicas generales. Un componente cíclico, si existe, típicamente presenta un ciclo durante varios años. Las fluctuaciones cíclicas a menudo están influidas por cambios en las expansiones y contracciones económicas.
   El *compo0nente cíclico* es la oscilación alrededor de la *tendencia.*

1.2. Exploración de patrones de datos con análisis de autocorrelación.

Cuando se mide una variable a lo largo del tiempo, las observaciones en fdiferentes períodos a menudo están relacionas o correlacionadas. Esta correlación se mide usando el coeficiente de autocorrelación.
Autocorrelación es la correlación que existe entre una variable retrasada uno o más períodos consigo misma.

1.3. Series temporales con R-Studio

Las series temporales se trata de una serie de modelos donde se observan repeticiones a lo largo del tiempo. Los análisis de estas nos ayudarán en cosas como:

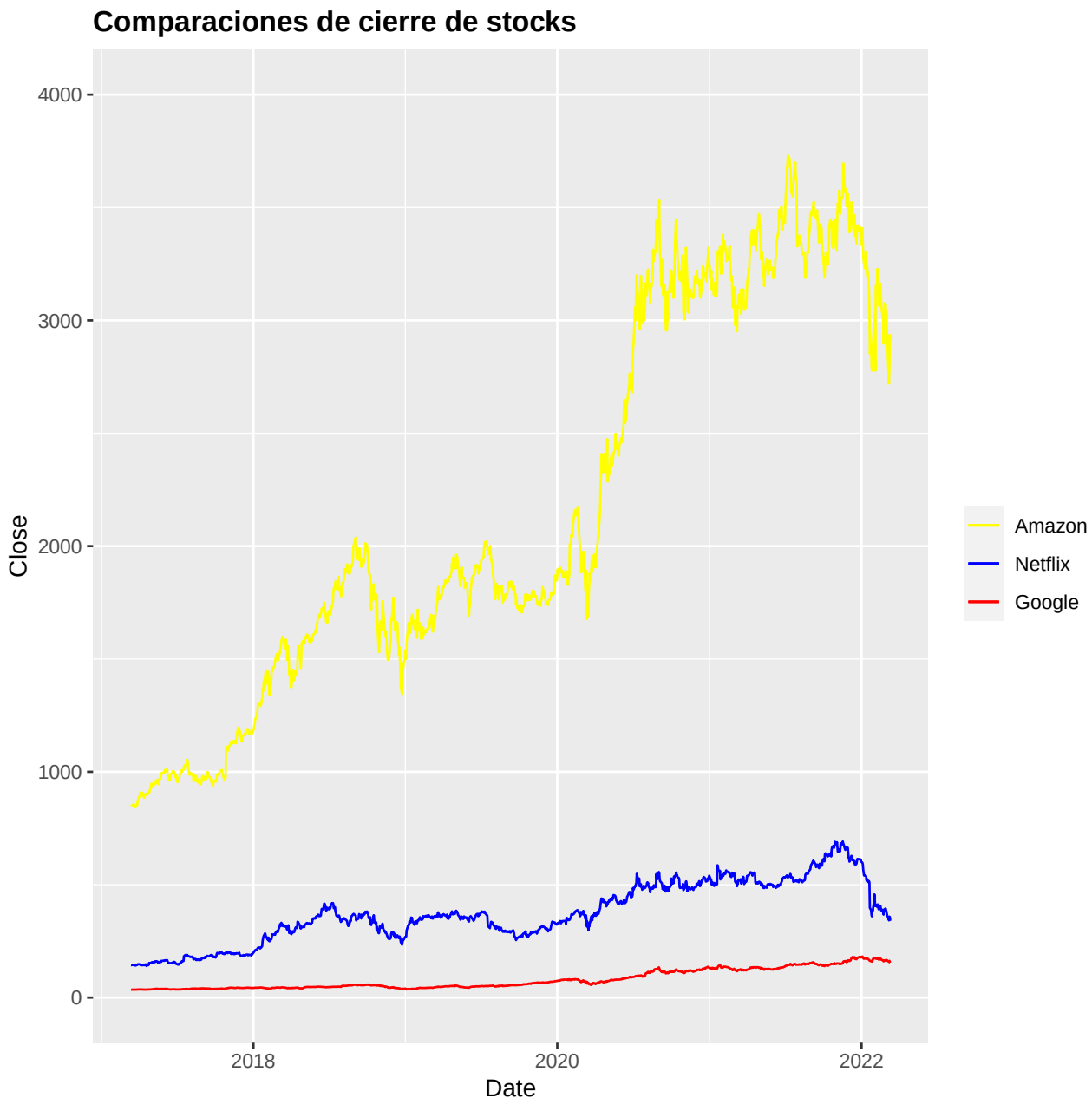
- Analizar y prever el tráfico en una página web.
- Hacer pronósticos de ventas.
- Estudiar el inventario si hay que pedir más de un cierto *ítem* en una determinada época del año o producir menos.

```
setwd("C:\\Users\\81799\\OneDrive\\Documentos\\ESFM_CLASES\\Servicio Social ARTF\\Machine Learni
AMZN = read.csv("AMZN_actual.csv", stringsAsFactors = F)
NX = read.csv("NFLX_actual.csv", stringsAsFactors = F)
GOOG = read.csv("AAPL_actual.csv", stringsAsFactors = F)
#Conversión de Columna Date en formato fecha
```

```
AMZN$Date = as.Date(AMZN$Date)
NX$Date = as.Date(NX$Date)
GOOG$Date = as.Date(GOOG$Date)
```

Gráficos

```
library(ggplot2)
ggplot(AMZN, aes(Date, Close))  +
  geom_line(data=AMZN, aes(color ="Amazon"))+
  geom_line(data=NX, aes(color="Netflix"))+
  geom_line(data=GOOG, aes(color="Google"))+
  labs(color="Legend")+
  scale_color_manual("",
                     breaks = c("Amazon", "Netflix", "Google" ),
                     values = c("yellow", "blue", "red"))+
  ggtitle("Comparaciones de cierre de stocks")+
  scale_y_continuous(limits = c(0,4000))+
  theme(plot.title = element_text(lineheight = 1, face = "bold"))
```



1.4. El formato fecha en R

```
Sys.Date() #Nos da la fecha de hoy

## [1] "2022-04-28"

as.Date("1/1/80", format="%m/%d/%y") #Años en dígitos

## [1] "1980-01-01"

as.Date("1/1/1980", format="%m/%d/%Y") #Ano en cuatro dígitos

## [1] "1980-01-01"

as.Date("2018-01-06") #Formato yyyy-mm-dd o yyyy/mm/dd

## [1] "2018-01-06"

nac <- as.Date("99/11/8")
as.numeric(as.Date("1988/05/19")) #Días que han pasado hasta la fecha

## [1] 6713

#Nombre de los meses
as.Date("Ene 6, 2018", format="%b %d, %Y") #nota en la b e Y

## [1] NA

as.Date("Enero 6, 18", format="%B %d, %y") #Nota en la B e y

## [1] "2018-01-06"

#Fechas desde días de EPOCH
#EPOCH : 1 de Enero de 1970
dt <- 2018
class(dt) <- "Date"
dt

## [1] "1975-07-12"

dt <- -2018
class(dt) <- "Date"
dt

## [1] "1964-06-23"

#Fechas desde días de un punto dado
dt <- as.Date(2018, origin = as.Date("1999-08-11"))
as.Date(-2018, origin = as.Date("1999-08-11"))

## [1] "1994-01-31"

#Componentes de las fechas
dt

## [1] "2005-02-18"

format(dt, "%Y")#Año en 4 dígitos
```

```
## [1] "2005"

as.numeric(format(dt, "%Y"))#Año como número en lugar de String

## [1] 2005

format(dt, "%y")#Año en 2 dígitos

## [1] "05"

#Año como número en lugar de String
as.numeric(format(dt, "%y")) #Año como número en lugar de String

## [1] 5

#Mes como String
format(dt, "%b") #Abreviado

## [1] "feb."

format(dt, "%B") #Nombre completo del mes

## [1] "febrero"

months(dt) #Nos da el mismo resultado de format()

## [1] "febrero"

weekdays(dt) #Nos da el dia

## [1] "viernes"

quarters(dt) #Nos da el 4to trimestre

## [1] "Q1"

julian(dt) #Calendario Juliano

## [1] 12832
## attr(,"origin")
## [1] "1970-01-01"
```

Operaciones de fechas.

```
dt <- as.Date("1/1/2001", format = "%d/%m/%Y")
dt+100 #Sumar 100 días

## [1] "2001-04-11"

dt-100 #Restar 100 días

## [1] "2000-09-23"

dt2 <- as.Date("2001/01/02") #Formto anglosajon
dt2-dt #Diferencia de fechas
```

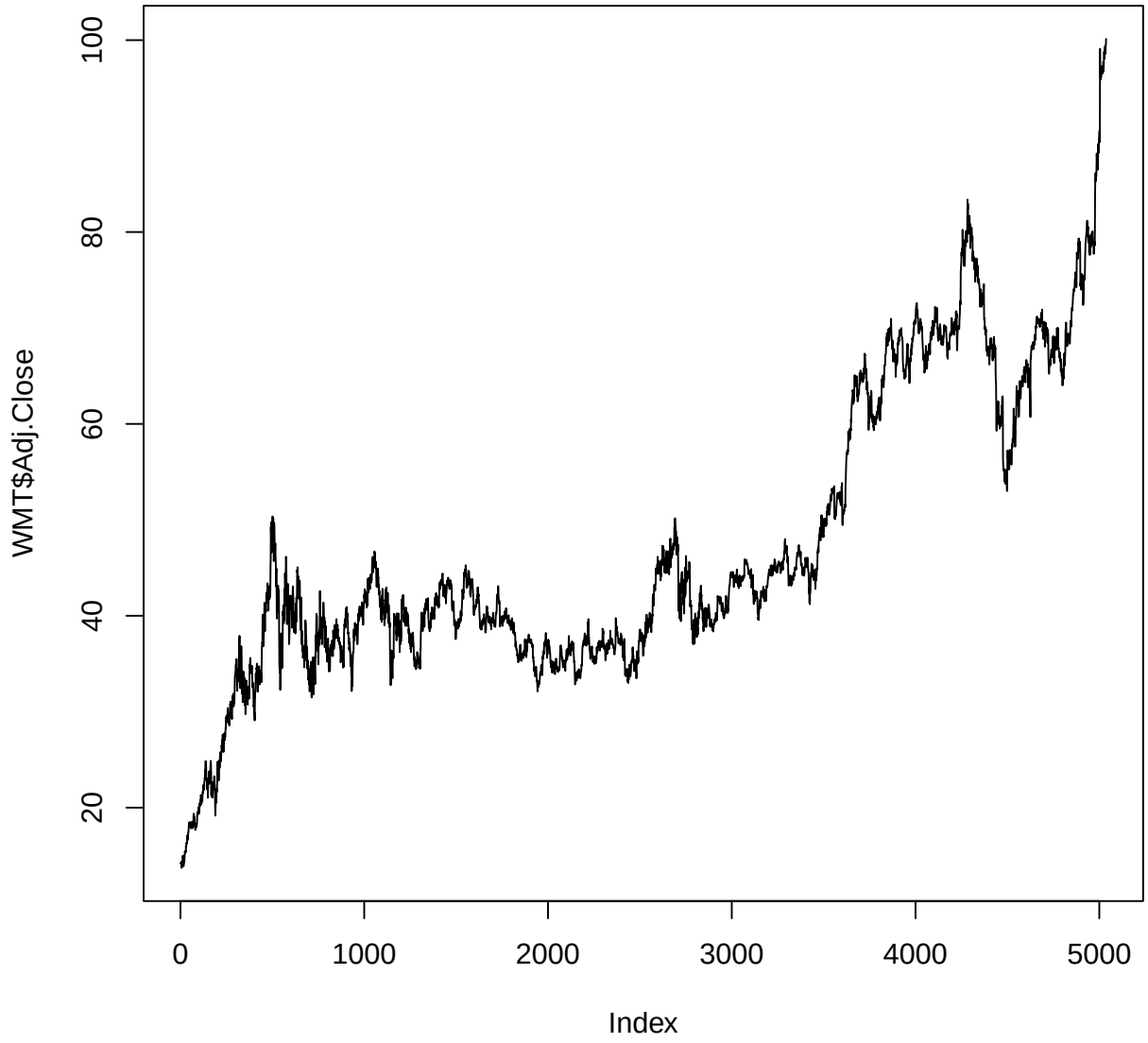
```
## Time difference of 1 days

dt-dt2 #Diferencia de fechas

## Time difference of -1 days

as.numeric(dt2-dt)  #No. excto de la diferencia en dias

## [1] 1

dt<dt2 #Devuelve un Booleano

## [1] TRUE

dt==dt2 #Devuelve un Booleano

## [1] FALSE

dt2<dt #Devuelve un Booleano

## [1] FALSE
```

Secuencias de fechas.

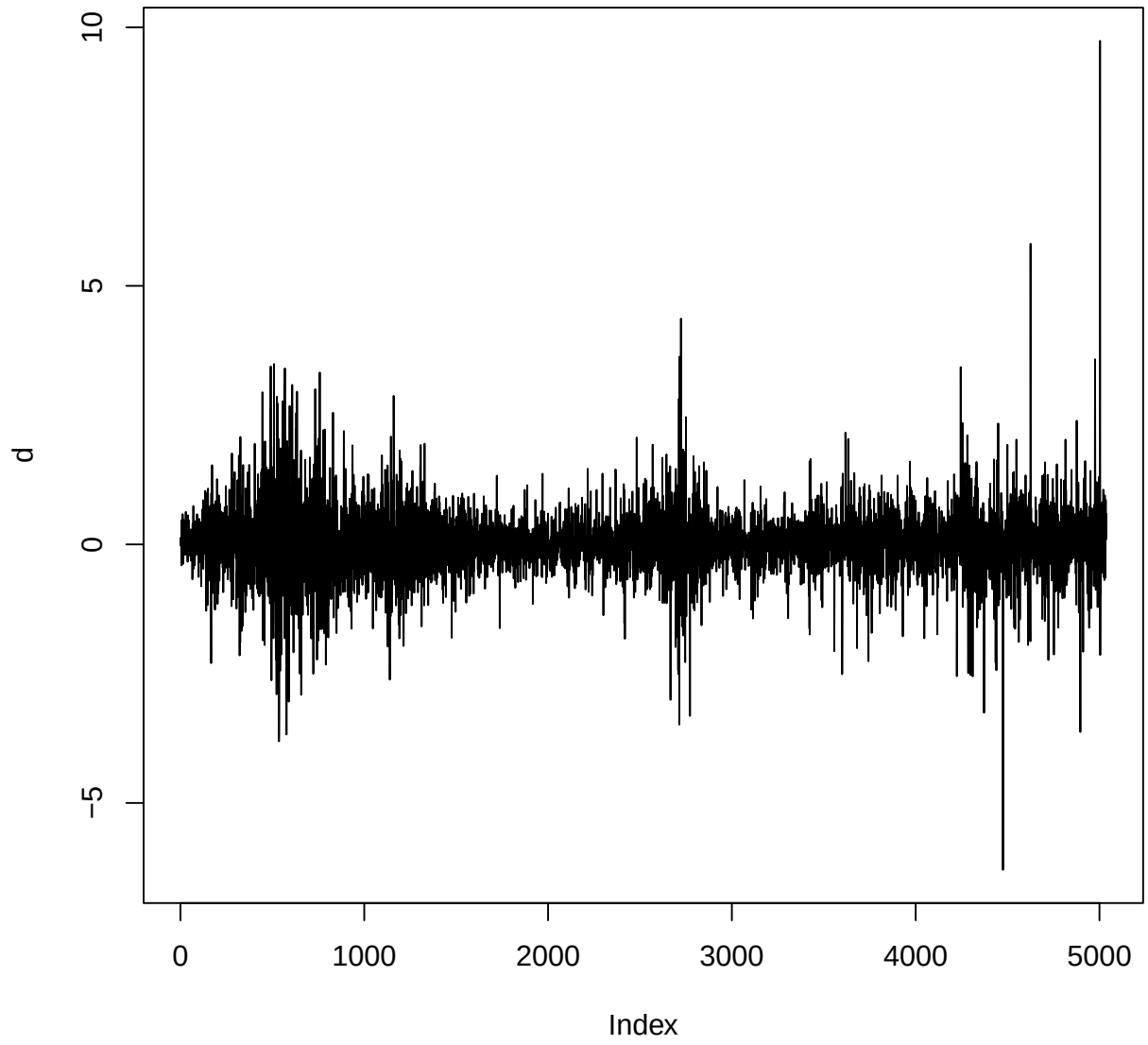
```
seq(dt, dt+180, "month") #Es una secuencia mensual

## [1] "2001-01-01" "2001-02-01" "2001-03-01" "2001-04-01" "2001-05-01"
## [6] "2001-06-01"

seq(dt, as.Date("2001/01/10"), "day") #Es una secuencia diaria.

##  [1] "2001-01-01" "2001-01-02" "2001-01-03" "2001-01-04" "2001-01-05"
##  [6] "2001-01-06" "2001-01-07" "2001-01-08" "2001-01-09" "2001-01-10"

seq(dt, dt+180, "2 months") #Secuencia bimensual.

## [1] "2001-01-01" "2001-03-01" "2001-05-01"

seq(from = dt, by = "4 months", length.out = 6) #Me da los 6 meses con secuencia de 4 meses

## [1] "2001-01-01" "2001-05-01" "2001-09-01" "2002-01-01" "2002-05-01"
## [6] "2002-09-01"

seq(from = dt, by = "4 months", length.out = 6)[3] #Me da los 6 meses con secuencia de 4 meses g

## [1] "2001-09-01"
```

1.5. Análisis preliminar de un serie temporal

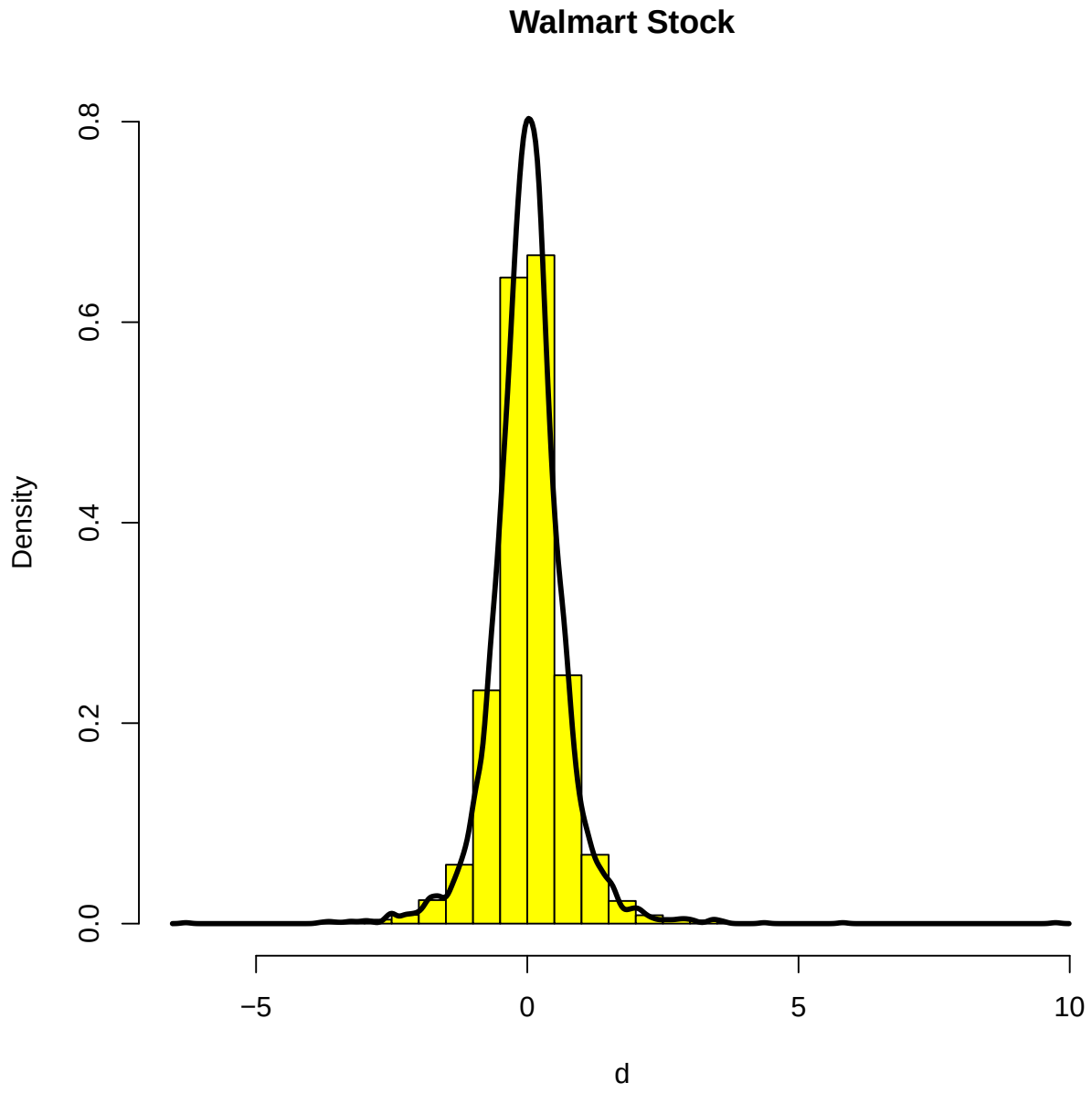
```
setwd("C:\\Users\\81799\\OneDrive\\Documentos\\ESFM_CLASES\\Servicio Social ARTF\\Machine Learni
WMT = read.csv("WMT.csv", stringsAsFactors = F)#Cargar datos
plot(WMT$Adj.Close, type = "l") #Gráfico
```




```r
d <- diff(WMT$Adj.Close) #Diferencias de un día al siguiente. X_{n+1}-X_{n}
plot(d, type = "l") #Gráfico de la diferencias
```




```
hist(d, probability = T, ylim = c(0,0.8), main = "Walmart Stock", breaks = 40, col = "yellow") #
lines(density(d), lwd =3) #Gráfico de densidad
```




```
setwd("C:\\Users\\81799\\OneDrive\\Documentos\\ESFM_CLASES\\Servicio Social ARTF\\Machine Learni
wmt.m <- read.csv("WMT-monthly.csv", stringsAsFactors = F)
wmt.m <- wmt.m[-1,]
wmt.m$Adj.Close <-  as.numeric(wmt.m$Adj.Close)
wmt.m$Date <- as.Date(wmt.m$Date) #Convierte la fecha en tipo Date
wmt.m.ts <- ts(wmt.m$Adj.Close) #Time series, nos da una serie temproal.


wmt.m.ts

## Time Series:
## Start = 1
## End = 242
## Frequency = 1
##   [1]  14.30459  16.64004  18.25688  18.19465  19.83644  21.86056  22.74612
##   [8]  21.25974  19.68328  24.91670  27.17159  29.38141  31.05937  31.10452
##  [15]  33.29402  33.26094  30.82062  34.88784  30.58544  32.07852  34.43124
```

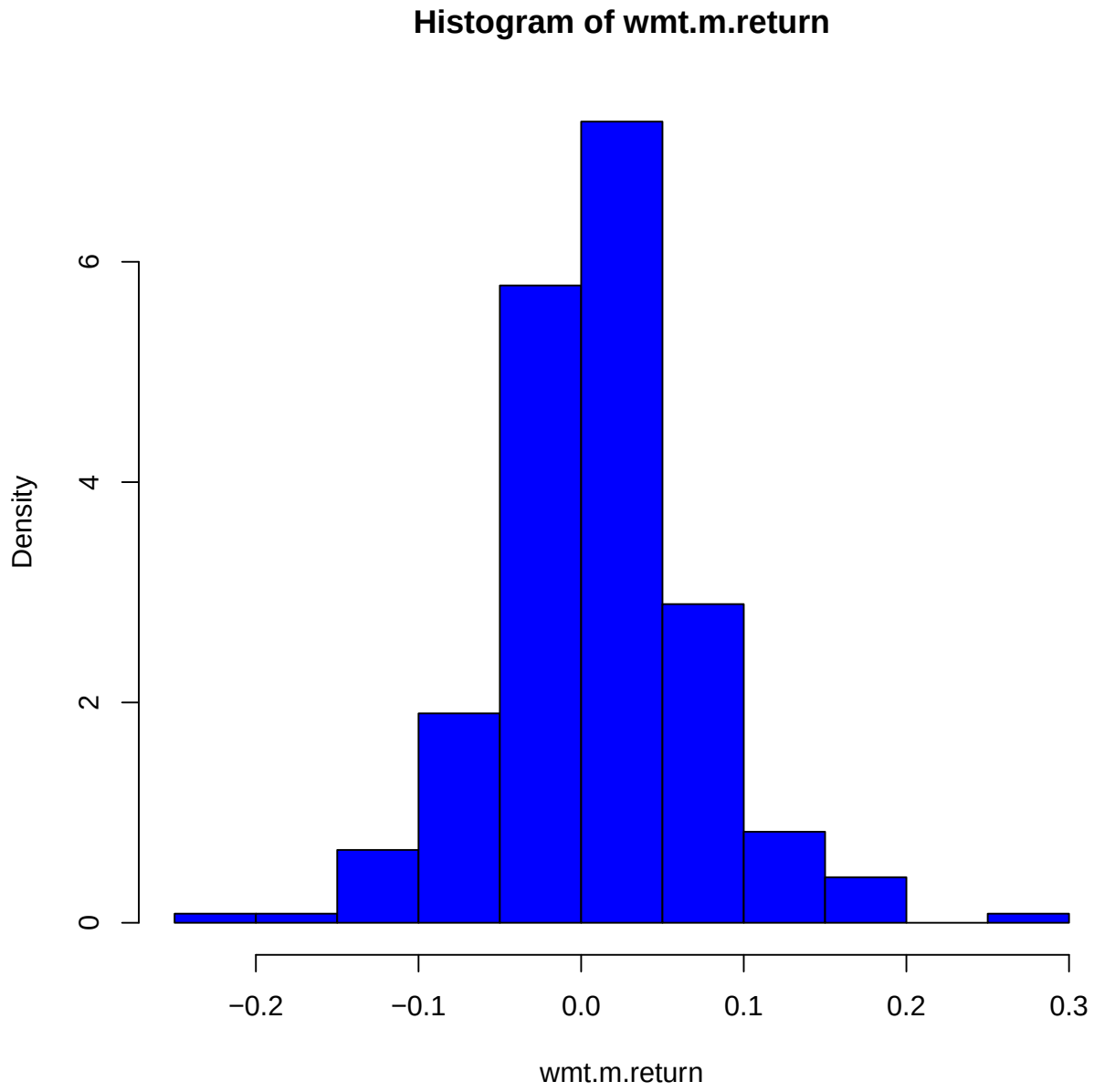
```
##  [22]  40.80898  41.76013  50.09405  39.70625  35.35489  40.97542  40.21006
##  [29]  41.84386  41.84386  40.16309  34.62022  34.98370  33.02132  37.97906
##  [36]  38.66132  41.38750  36.49824  36.79701  37.75573  37.76303  35.61035
##  [43]  40.84988  35.11336  36.17297  37.61950  40.36412  42.12067  43.95433
##  [50]  45.44194  44.92164  40.98355  39.69227  40.35993  36.12884  39.28772
##  [57]  36.17291  39.39360  39.65107  37.15723  35.21550  35.40707  38.33185
##  [64]  41.56399  38.82604  39.60831  41.32898  43.73880  41.28463  43.57617
##  [71]  41.19437  39.27680  39.93905  44.17400  44.27042  42.37032  41.42630
##  [78]  39.11742  39.49740  39.24408  39.73295  40.27069  38.88152  39.44915
##  [85]  39.23077  38.63932  37.51630  35.39676  35.46435  36.30858  37.17486
##  [92]  33.86792  33.11359  35.75088  36.69548  35.36549  34.95005  34.38157
##  [99]  35.80657  34.25782  36.85968  36.77519  33.97333  34.14130  37.79568
## [106]  37.76503  35.32808  35.38940  36.68156  37.15844  36.11237  37.03489
## [113]  36.78759  37.35430  35.67720  33.87588  34.06242  35.27978  37.37891
## [120]  37.09019  39.77358  38.87214  41.29430  45.66510  45.47607  44.44991
## [127]  46.36393  46.71986  47.55946  44.31947  44.37506  44.51799  37.58051
## [134]  39.27132  41.55231  40.42322  39.89387  39.06070  40.22189  41.02018
## [141]  39.80200  40.28036  44.22893  43.33705  43.53930  44.06083  45.30760
## [148]  43.95680  41.43281  39.62116  42.19278  41.32732  44.37070  44.90959
## [155]  44.84327  44.71062  46.74187  43.33231  43.39065  46.15445  46.35593
## [162]  44.90497  44.54161  44.94723  44.17441  48.27693  50.13244  50.86441
## [169]  52.55257  50.59983  52.41554  50.79194  56.74972  60.52019  64.60867
## [176]  63.02015  64.40826  65.47302  62.85480  59.54711  61.38697  62.11537
## [183]  65.66960  68.64586  66.10209  66.18758  69.25307  64.84590  66.11571
## [190]  68.60980  72.41798  70.34404  67.14797  67.16595  68.72147  72.13303
## [197]  69.47252  68.35471  66.99796  68.74623  70.08817  69.90487  80.23433
## [204]  78.71286  78.32311  77.35537  75.80697  72.36805  68.86323  66.18144
## [211]  67.16114  60.39652  60.91224  53.77263  55.27571  57.58669  62.86234
## [218]  62.84340  64.88007  63.81441  67.54574  70.19359  70.14553  68.67476
## [225]  69.80302  67.77048  68.16732  66.89940  65.05818  69.14259  70.26362
## [232]  73.82436  77.18269  74.81267  79.07327  77.17527  77.73035  86.85226
## [239]  96.72027  98.23229 100.13000 100.13000
```

```
#Nos da la diferencia de un día
d <- diff(as.numeric(wmt.m.ts)) # X_{n+1}-X_{n}
#Nos da la diferencia de 2 días
d.2 <- diff(as.numeric(wmt.m.ts), lag = 2) # X_{n+2}-X_{n}
r <- lag(as.numeric(wmt.m.ts),k=-1)
```

Para sacar el incremento porcentual, tenemos que:

$$\sum_{k=0}^{n} \frac{X_{k+2} - X_{k+1}}{X_k}$$

```
#(X_{n+2}-X_{n+1})/X_{n} para todo n>=1
wmt.m.return <- d / lag(as.numeric(wmt.m.ts), k = -1) #k=-1 nos da un dia hacia atrás, esto nos

## Warning in d/lag(as.numeric(wmt.m.ts), k = -1): longitud de objeto mayor no es múltiplo
de la longitud de uno menor

hist(wmt.m.return, prob = T, col = "blue")
```

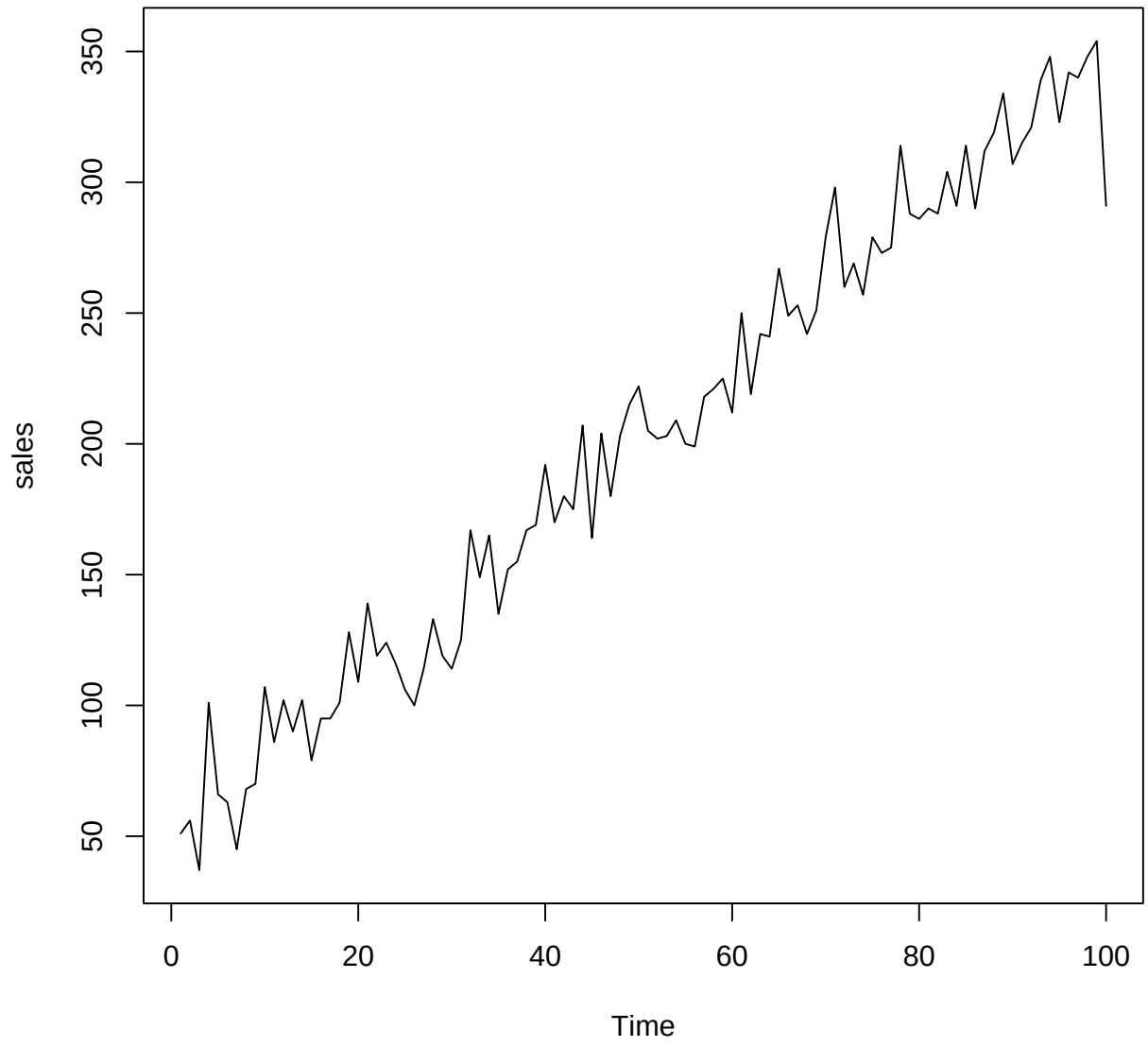


1.6. El objeto serie temporal de R

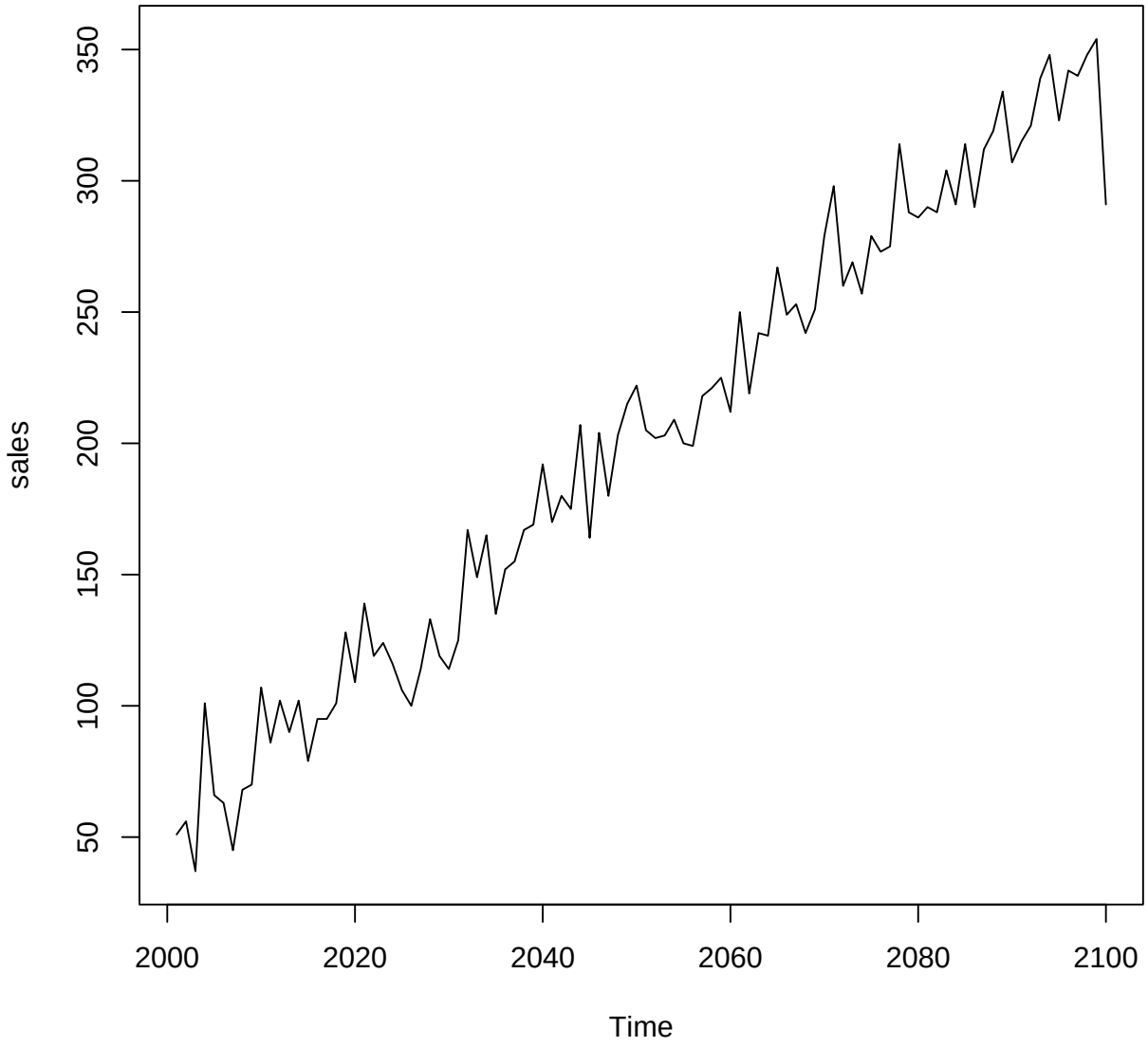
1.6.1. Serie temporal con una columna de datos

```
setwd("C:\\Users\\81799\\OneDrive\\Documentos\\ESFM_CLASES\\Servicio Social ARTF\\Machine Learni
s <- read.csv("ts-example.csv")
head(s,1)

##   sales
## 1    51

s.ts <- ts(s) #Serie temporal
class(s.ts)#Nos da que s.ts es de clase Serie temporal

## [1] "ts"

head(s.ts,1)
```

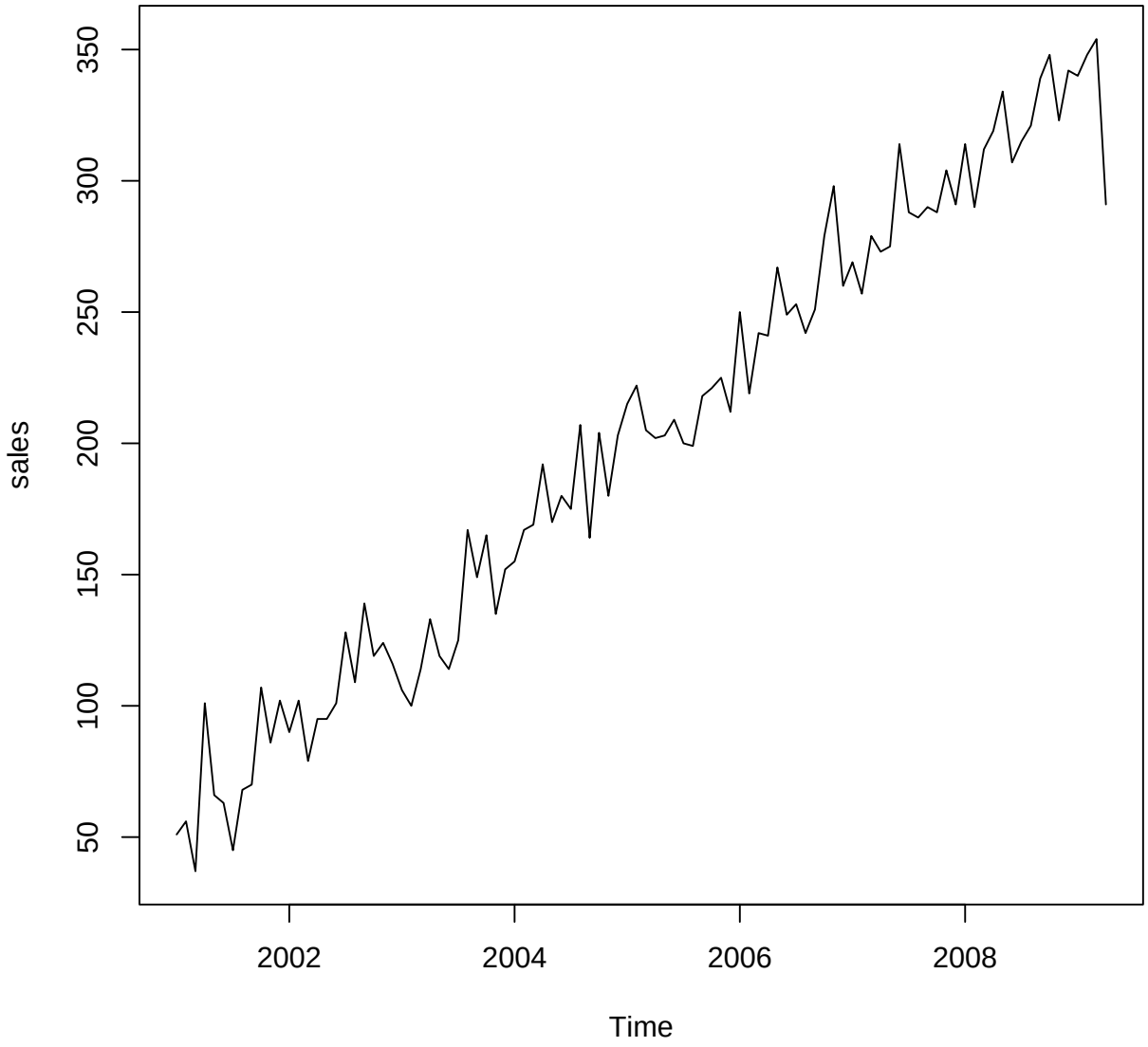
```
##      sales
## [1,]    51

plot(s.ts)
```




```
s.ts.a <- ts(s, start = 2001) #Serie temporal en donde inicia la serie (2001), frecuencia por a
plot(s.ts.a)
```



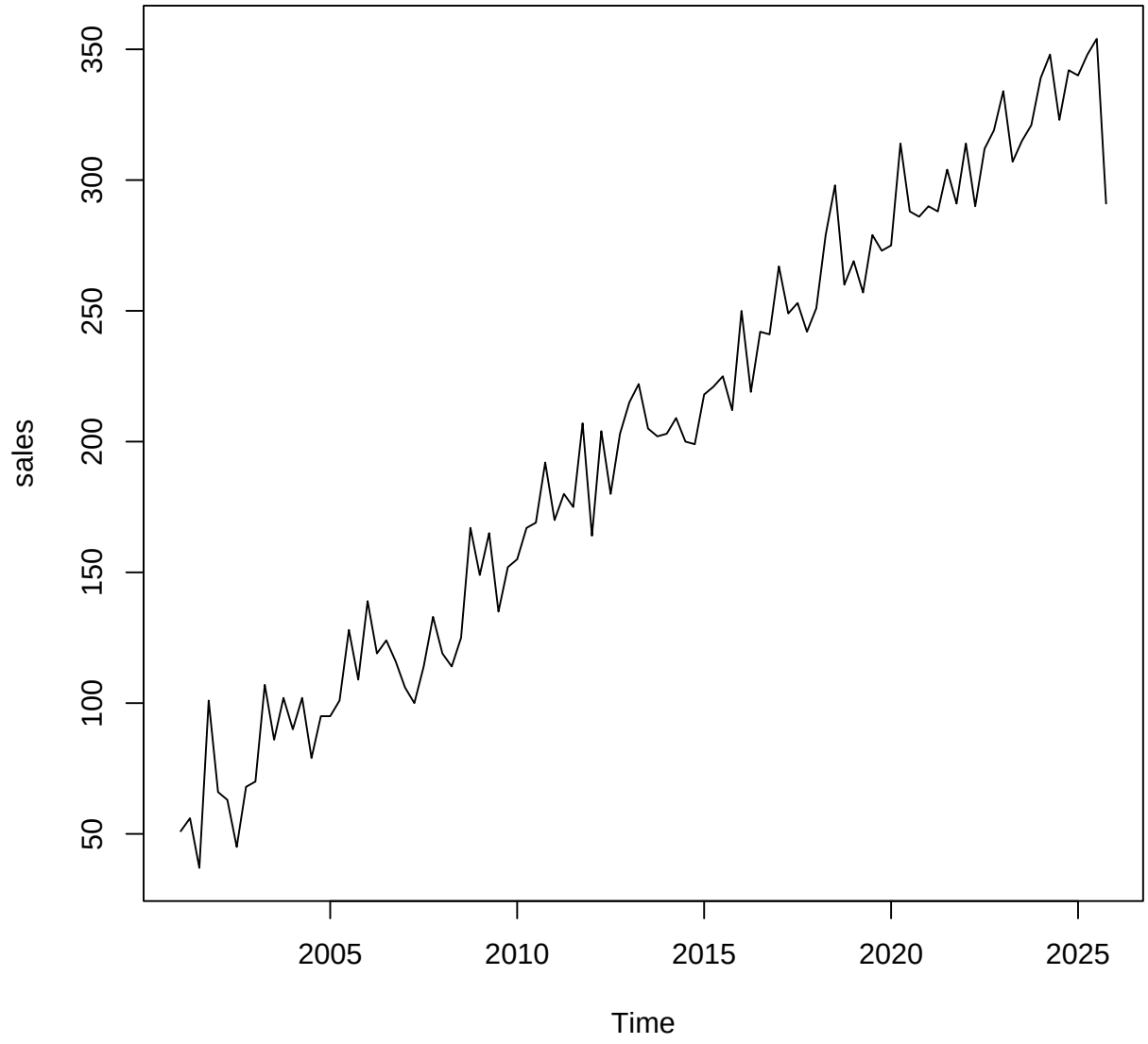

```
 #c(2001,1) indica enero-2001, frecuency=12 indica que tomo 12 muestras anuales
s.ts.m <- ts(s, start = c(2001,1), frequency = 12)
s.ts.m #Nos da la tabla de valores mensuales

##      Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
## 2001  51  56  37 101  66  63  45  68  70 107  86 102
## 2002  90 102  79  95  95 101 128 109 139 119 124 116
## 2003 106 100 114 133 119 114 125 167 149 165 135 152
## 2004 155 167 169 192 170 180 175 207 164 204 180 203
## 2005 215 222 205 202 203 209 200 199 218 221 225 212
## 2006 250 219 242 241 267 249 253 242 251 279 298 260
## 2007 269 257 279 273 275 314 288 286 290 288 304 291
## 2008 314 290 312 319 334 307 315 321 339 348 323 342
## 2009 340 348 354 291

plot(s.ts.m)
```




```
s.ts.q <- ts(s, start = 2001, frequency = 4) #4 tomas al año
s.ts.q #Nos da la tabla de valores trimestrales.

##      Qtr1 Qtr2 Qtr3 Qtr4
## 2001   51   56   37  101
## 2002   66   63   45   68
## 2003   70  107   86  102
## 2004   90  102   79   95
## 2005   95  101  128  109
## 2006  139  119  124  116
## 2007  106  100  114  133
## 2008  119  114  125  167
## 2009  149  165  135  152
## 2010  155  167  169  192
## 2011  170  180  175  207
## 2012  164  204  180  203
## 2013  215  222  205  202
## 2014  203  209  200  199
```

```
## 2015  218  221  225  212
## 2016  250  219  242  241
## 2017  267  249  253  242
## 2018  251  279  298  260
## 2019  269  257  279  273
## 2020  275  314  288  286
## 2021  290  288  304  291
## 2022  314  290  312  319
## 2023  334  307  315  321
## 2024  339  348  323  342
## 2025  340  348  354  291
```

```
plot(s.ts.q)
```




```
start(s.ts.q) #Nos indica cuando empieza la serie
```

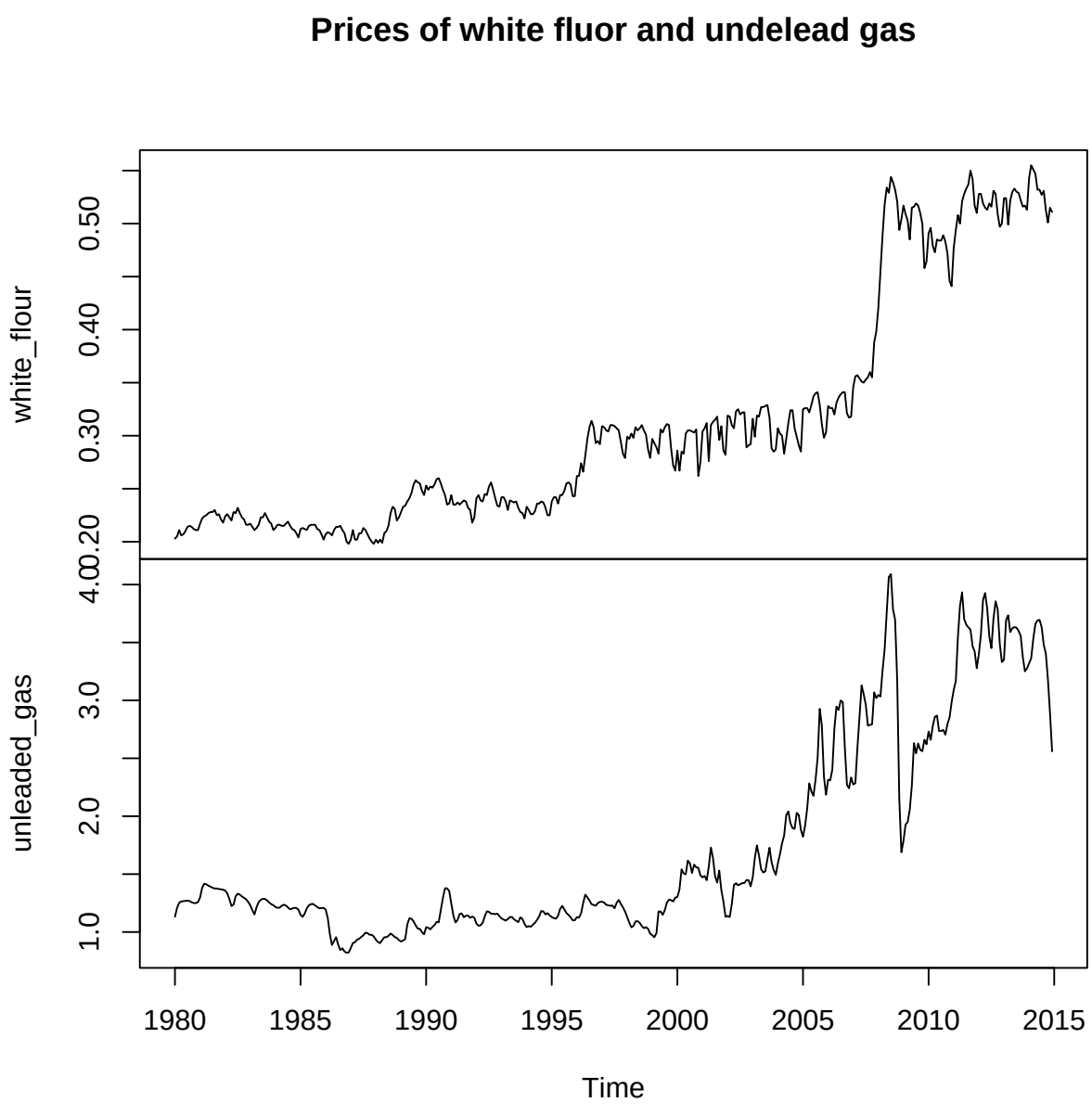
```
## [1] 2001    1
```

```
end(s.ts.q) #Nos indica cuando acaba la serie
```

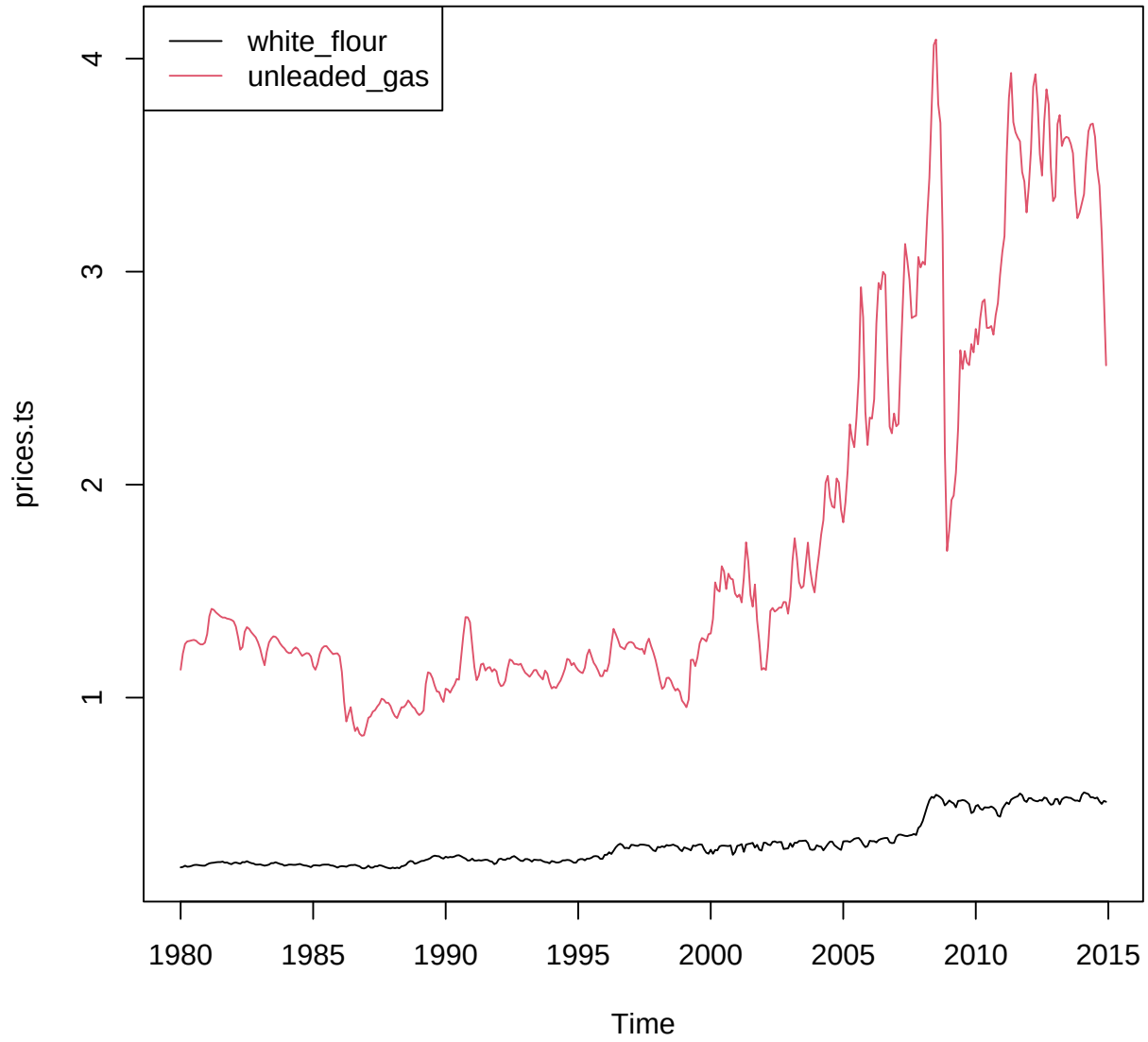
```
## [1] 2025    4

frequency(s.ts.q) #Nos indica el período de la serie

## [1] 4
```

1.6.2. Serie temporal con 2 columnas de datos.

```
setwd("C:\\Users\\81799\\OneDrive\\Documentos\\ESFM_CLASES\\Servicio Social ARTF\\Machine Learni
prices <- read.csv("prices.csv")
head(prices,2)

##   white_flour unleaded_gas
## 1       0.203        1.131
## 2       0.205        1.207

prices.ts <- ts(prices, start = c(1980,1), frequency = 12) #Fijamos la fecha de inicio, anual
plot(prices.ts, main = "Prices of white fluor and undelead gas")#Nos da dos gráficos
```



```
plot(prices.ts, plot.type = "single", col = 1:2) #Nos da 1 gráfico con las 2 columnas.
#plot.type = "single", nos junta los gráficos
legend("topleft", colnames(prices.ts), col = 1:2, lty = 1) #Nos dice una leyenda
```



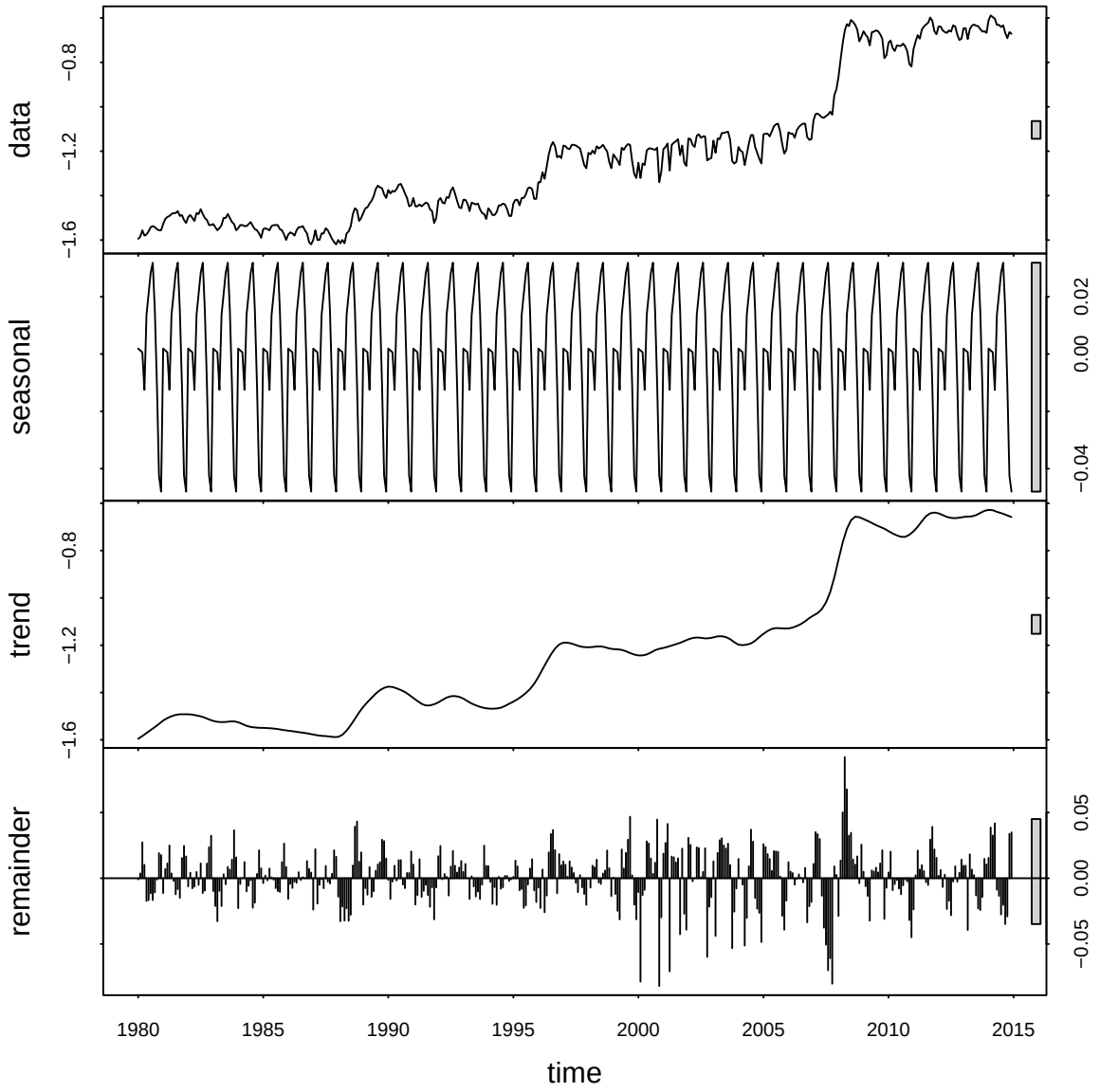

```
#topleft:Arriba a la izquierda, colnames:Con los nombres de la columna. lty=1 : Es el tamaño
```

1.6.3. Descomposición de una serie temporal.

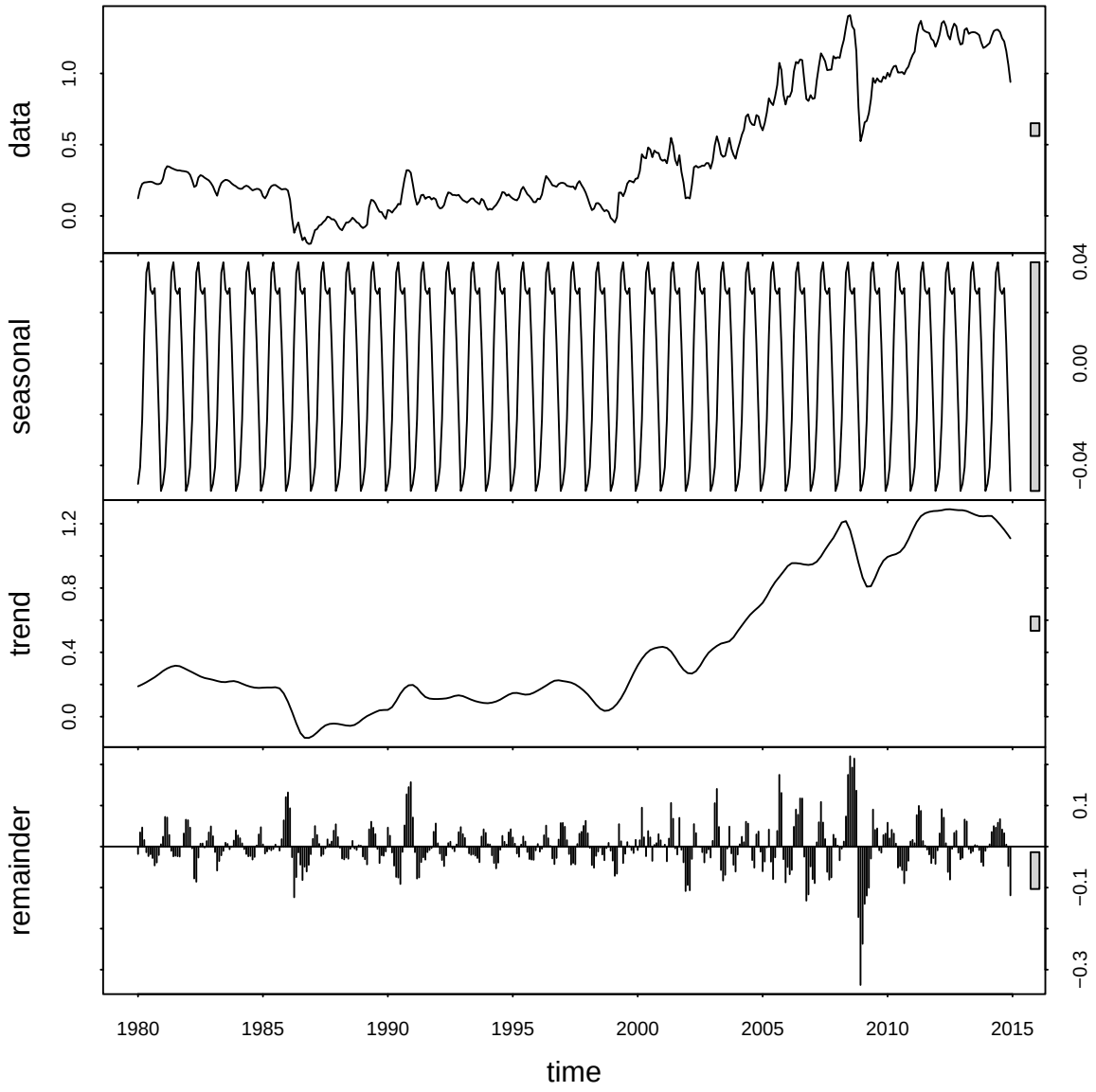
- Una serie temporal se basa en el hecho de que en la mayoría de procesos que existen en el mundo real, tienden a tener lo que llamamos una *estacionalidad* o una *tendencia*
- Es muy útil para poder extraer datos que se van repitiendo a lo largo de diferentes estaciones
- Hay factores que no tienen nada que ver con al *temporalidad*.
- Una **serie multiplicativa** se da cuando la serie va creciendo constantemente, de modo que la amplitud de las fluctuaciones tiene un incremento con el tiempo.
- Aplicar logaritmo cuando una *serie temporal* tenemos la sospecha de que es una *serie multiplicativa* de que cada factor es el anterior multiplicado por un número.

Seasonal Decomposition of Time Series by Loess

```
flour.l <- log(prices.ts[,1]) #Perder el factor multiplicativo
flour.stl<- stl(flour.l, s.window = "period")#Descomposición en estaciones,
#s.window ="period" toma el período original
#data:Valores del dato origina, seasonal:Función períodica ,trend : Tendencia, remainder: Ruido
#data=seasonal+trend+remainder
#flour.stl -> Descompone los valores: seasonal, trend y remainder
plot(flour.stl)
```

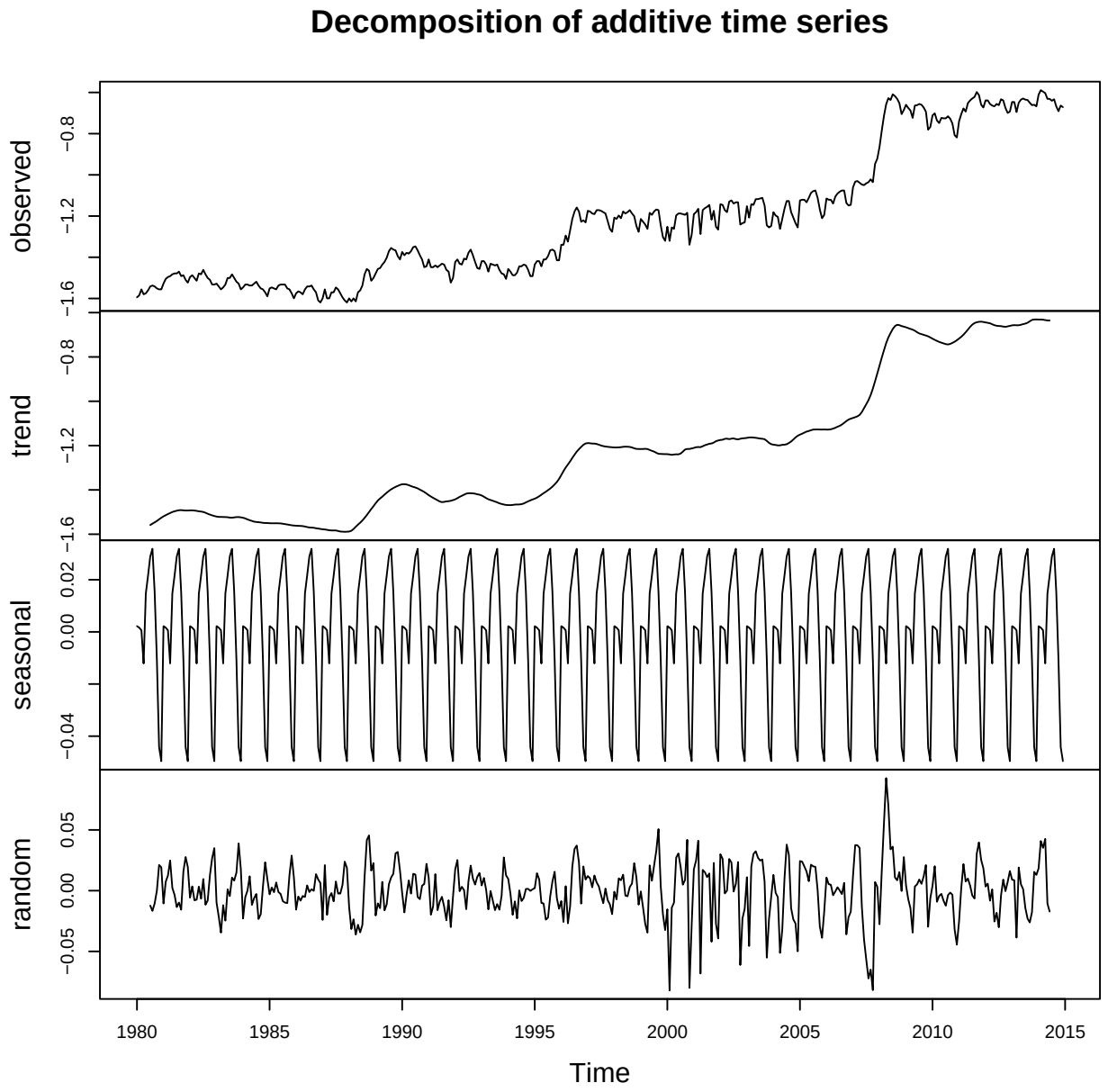



```
gas.l <- log(prices.ts[,2]) #Aplicamos lo mismo para la 2da columna.
gas.stl <- stl(gas.l, s.window = "period")#Aplica el mismo período
plot(gas.stl) #Hace la gráfica
```

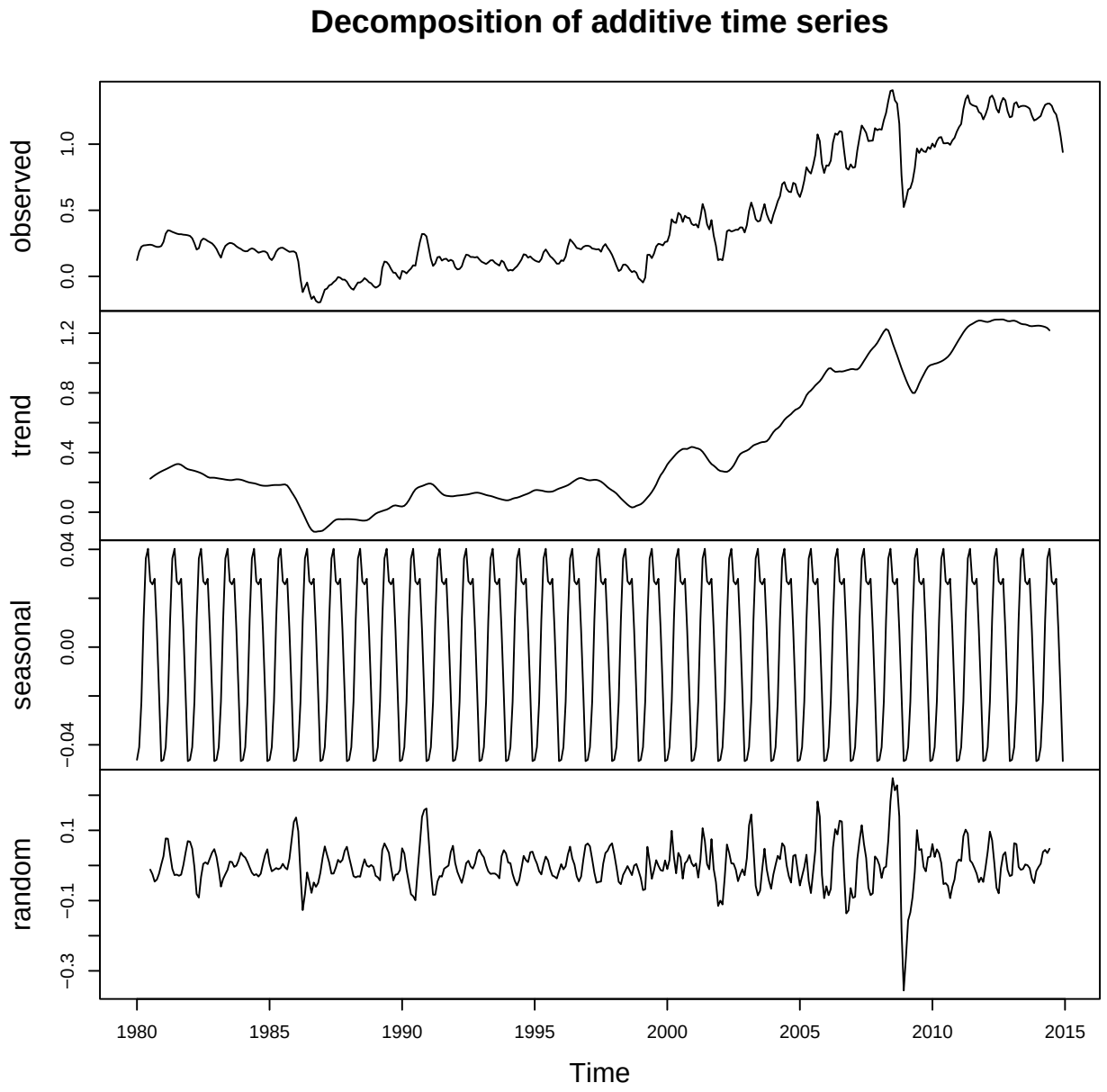


Classical Seasonal Decomposition by Moving Averages

```
flour.dec <- decompose(flour.l) #decompose(): Esta función que nos sirve para descomoponer la se
plot(flour.dec)
```



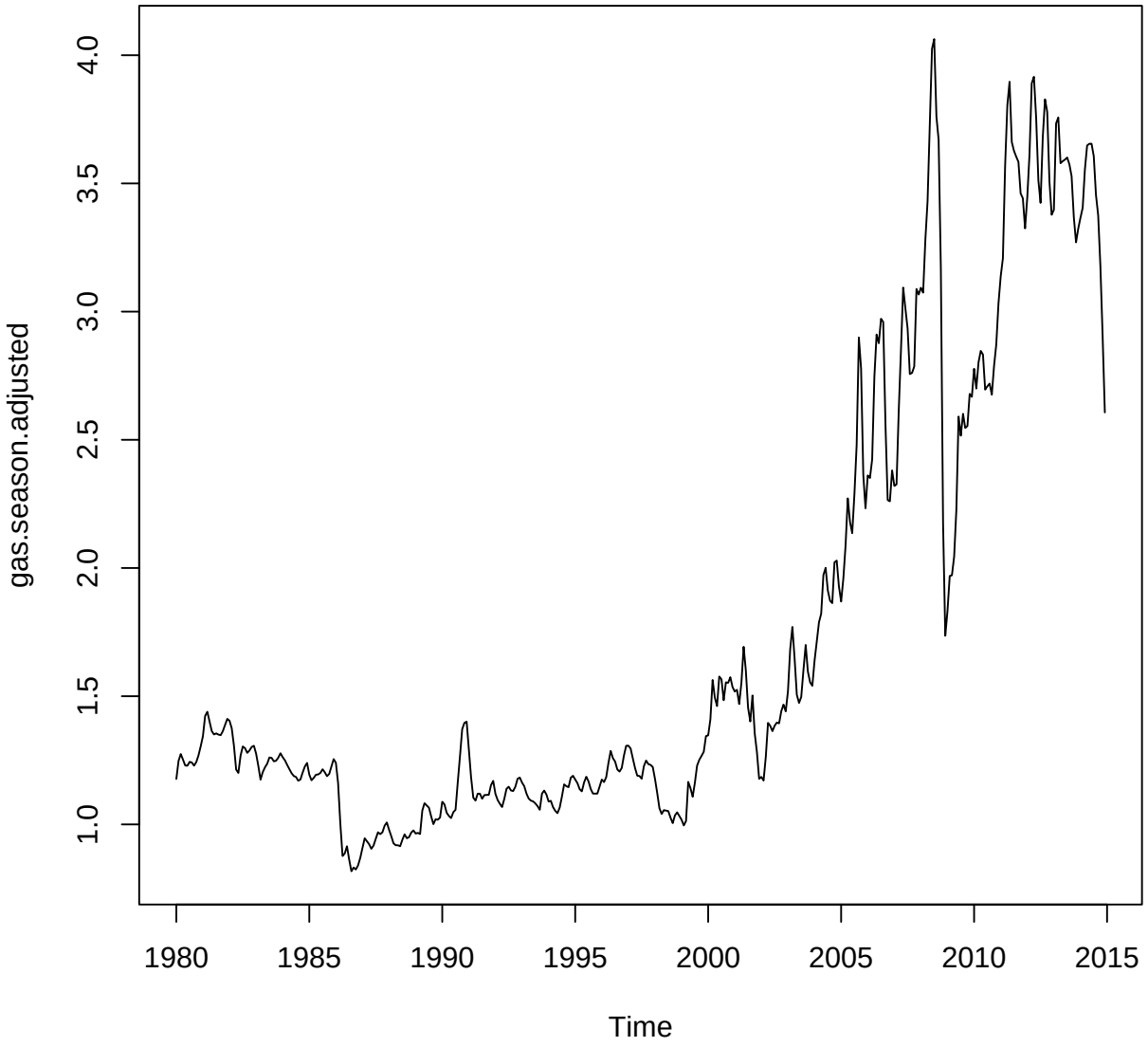

```
gas.dec <- decompose(gas.l)
plot(gas.dec)
```



Ajustar datos originales

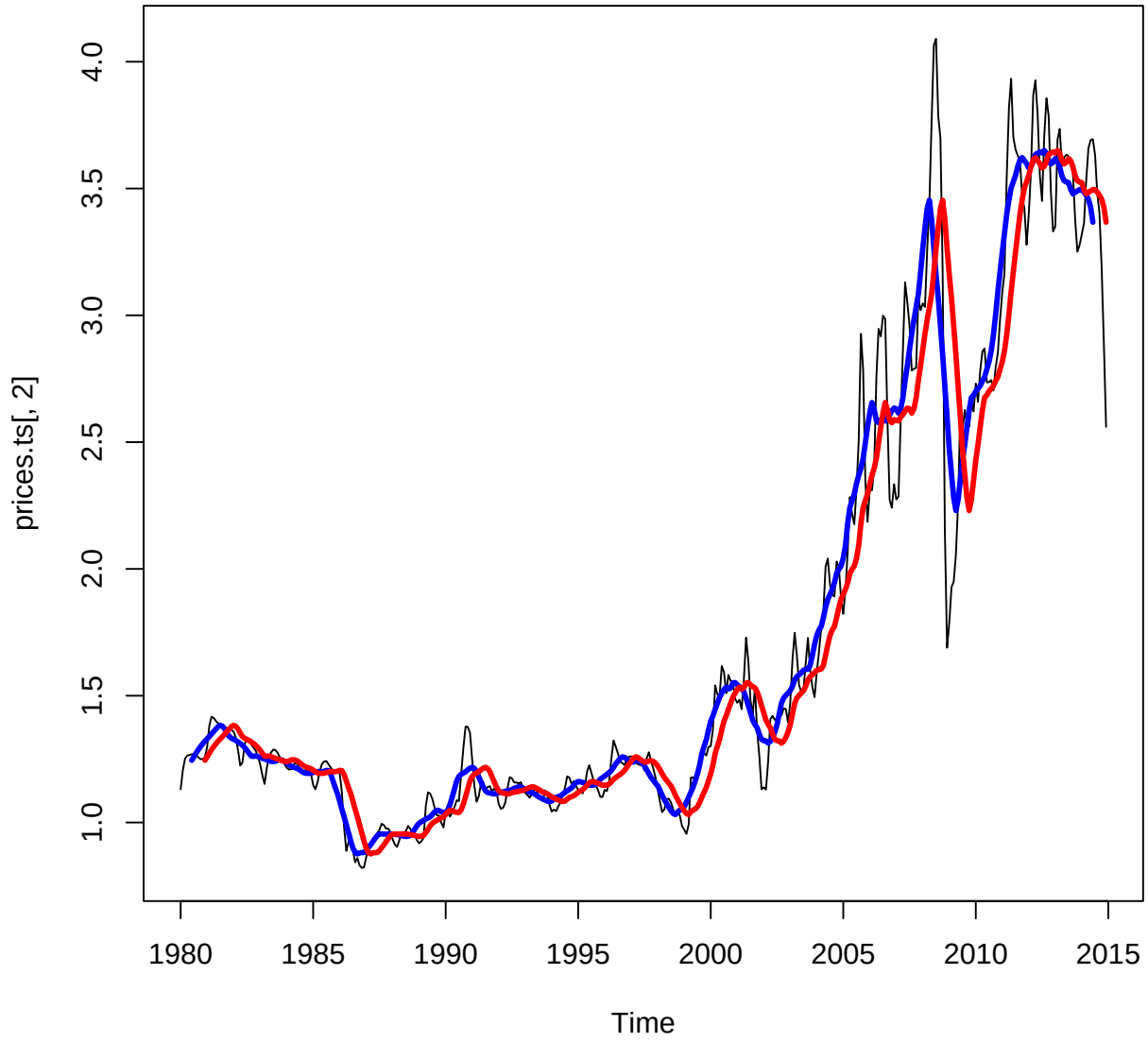
- Podemos tomar los datos originales y ajustarlos en el sentido de eliminar la información que se repite estación tras estación, con la finalidad de obtener cual es la tendencia del mercado.

```
gas.season.adjusted <- prices.ts[,2] - (gas.dec$seasonal)
plot(gas.season.adjusted) #Nos da información con menor ruido
```



De la información original se elimina la infomación que la serie temporal determina que estacional, es decir, que se va repitiendo estación con estación para así tener una visión más clara y completa de cual es la **tendencia global** del mercado.

```
n <- 12 #Perído (En este caso sería anualmente)
gas.f.1 <- filter(prices.ts[,2], filter = rep(1/n, n), sides = 2)
gas.f.2 <- filter(prices.ts[,2], filter = rep(1/n,n), sides = 1)
plot(prices.ts[,2])
lines(gas.f.1, col = "blue", lwd = 3)
lines(gas.f.2, col = "red", lwd = 3)
```



1.7. Suavizado y predicción

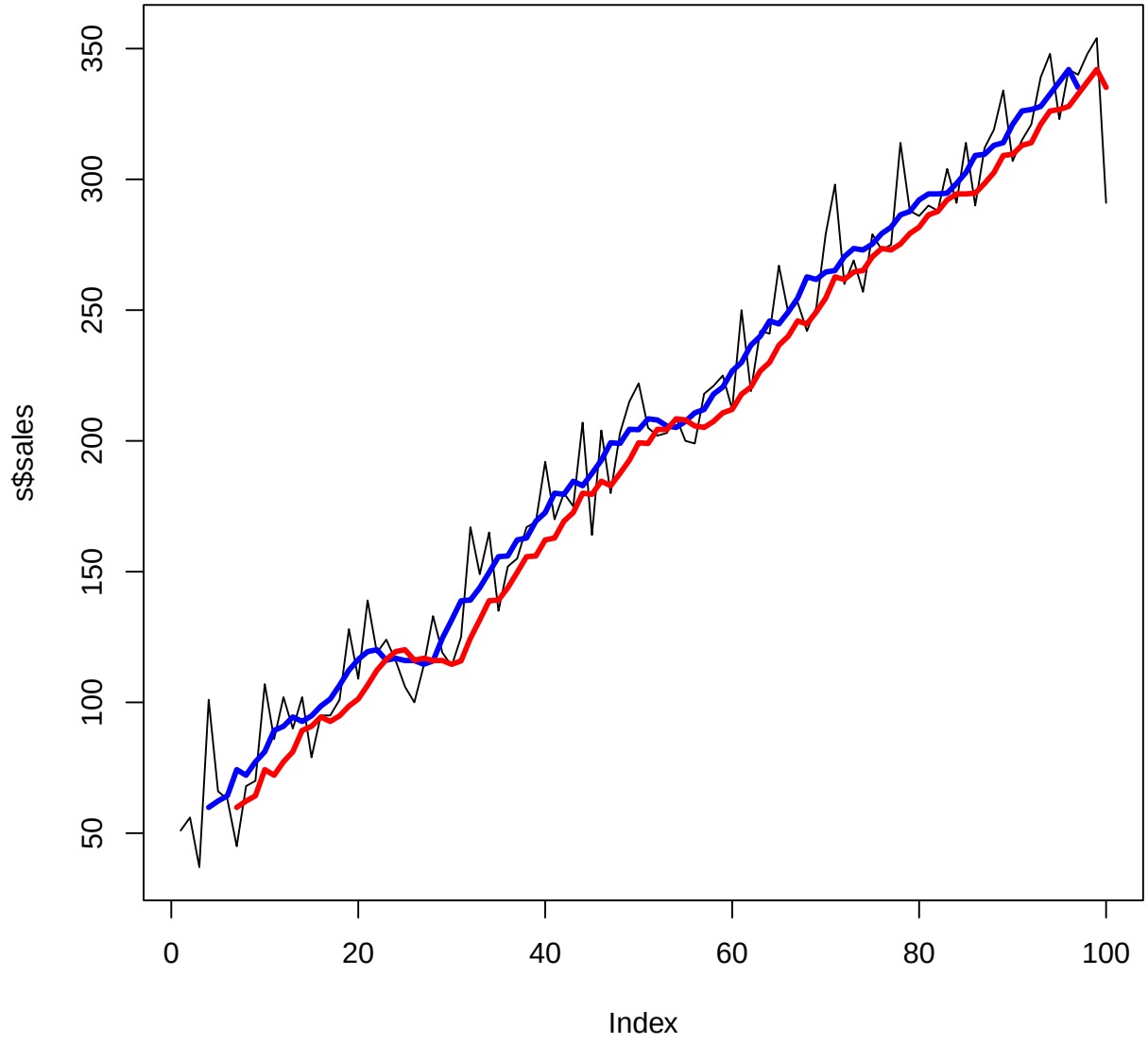
1.7.1. Promedios móviles ponderados exponenacialmente(EWMA)

```
setwd("C:\\Users\\81799\\OneDrive\\Documentos\\ESFM_CLASES\\Servicio Social ARTF\\Machine Learni
s <- read.csv("ts-example.csv")
s$sales
```

```
##   [1]  51  56  37 101  66  63  45  68  70 107  86 102  90 102  79  95  95 101
##  [19] 128 109 139 119 124 116 106 100 114 133 119 114 125 167 149 165 135 152
##  [37] 155 167 169 192 170 180 175 207 164 204 180 203 215 222 205 202 203 209
##  [55] 200 199 218 221 225 212 250 219 242 241 267 249 253 242 251 279 298 260
##  [73] 269 257 279 273 275 314 288 286 290 288 304 291 314 290 312 319 334 307
##  [91] 315 321 339 348 323 342 340 348 354 291
```

```
plot(s$sales, type = "l")
```

```
n <- 7 #Período (Filtro semanamente)
weights <- rep(1/n, n) #Pesos
weights

## [1] 0.1428571 0.1428571 0.1428571 0.1428571 0.1428571 0.1428571 0.1428571

#Suavizado de los 3 anteriores con los 3 posteriores del actual
s.fil.1 <- filter(s$sales, filter = weights, sides = 2) #Bi-lateral
lines(s.fil.1, col = "blue", lwd = 3) #Los primeros no se pueden predecir
#Suaviza los 6 anteriores y el actual
s.fil.2 <- filter(s$sales, filter = weights, sides = 1)#Uni-lateral
lines(s.fil.2, col = "red", lwd = 3)
```



Con la técnica de **Moving Avergane**, las tendencias se localizan mucho más fáciles si se promedian con los valores que tenemos cerca de ellos, antes y después con algunas de estas dos técnicas. Lo único que se hace es:

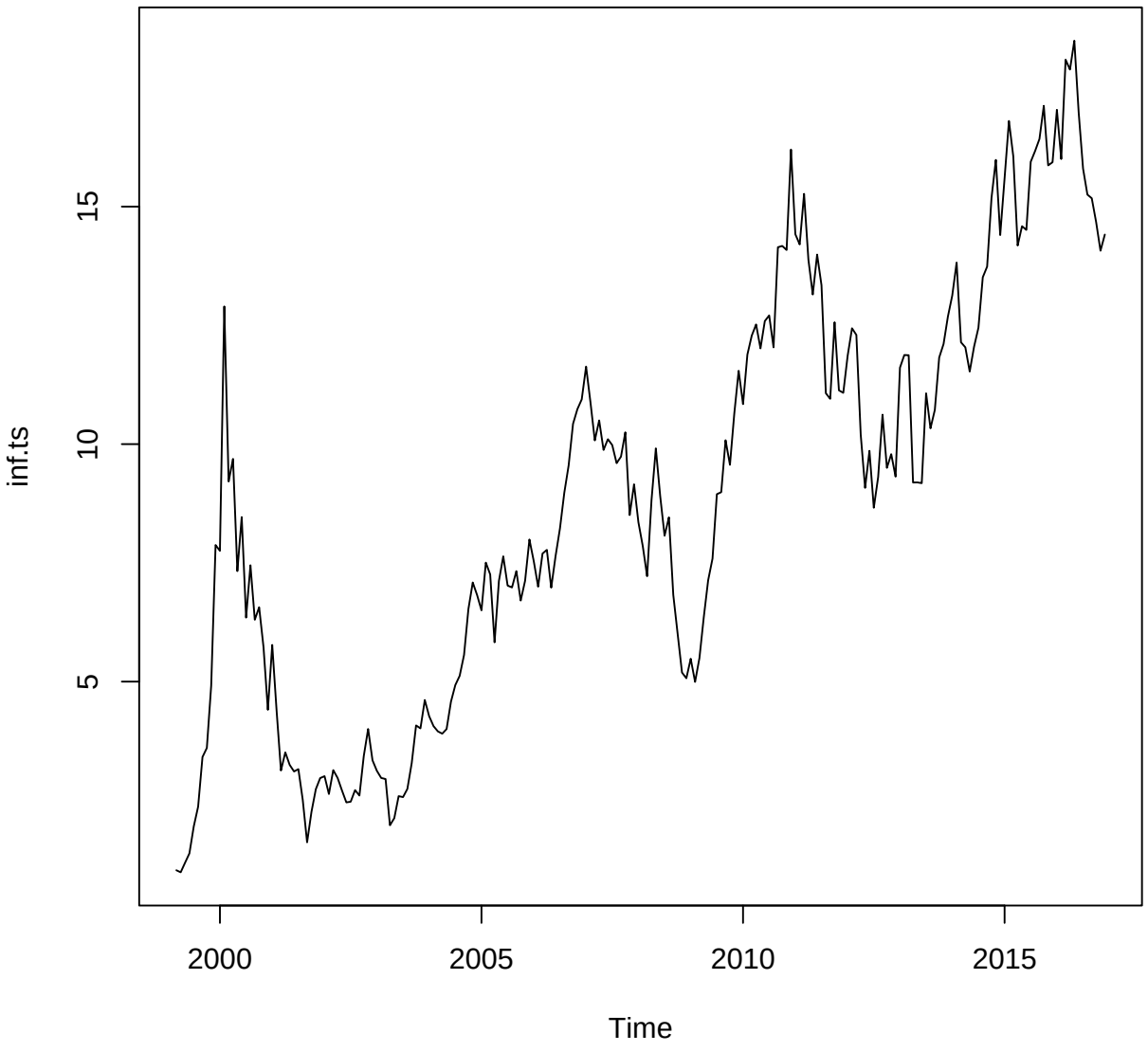
- Crear un filtro de pesos, (tienen que tener la particularidad que sumen 1)

1.7.2. Suavizado exponencial doble (Método de Holt-Winters).

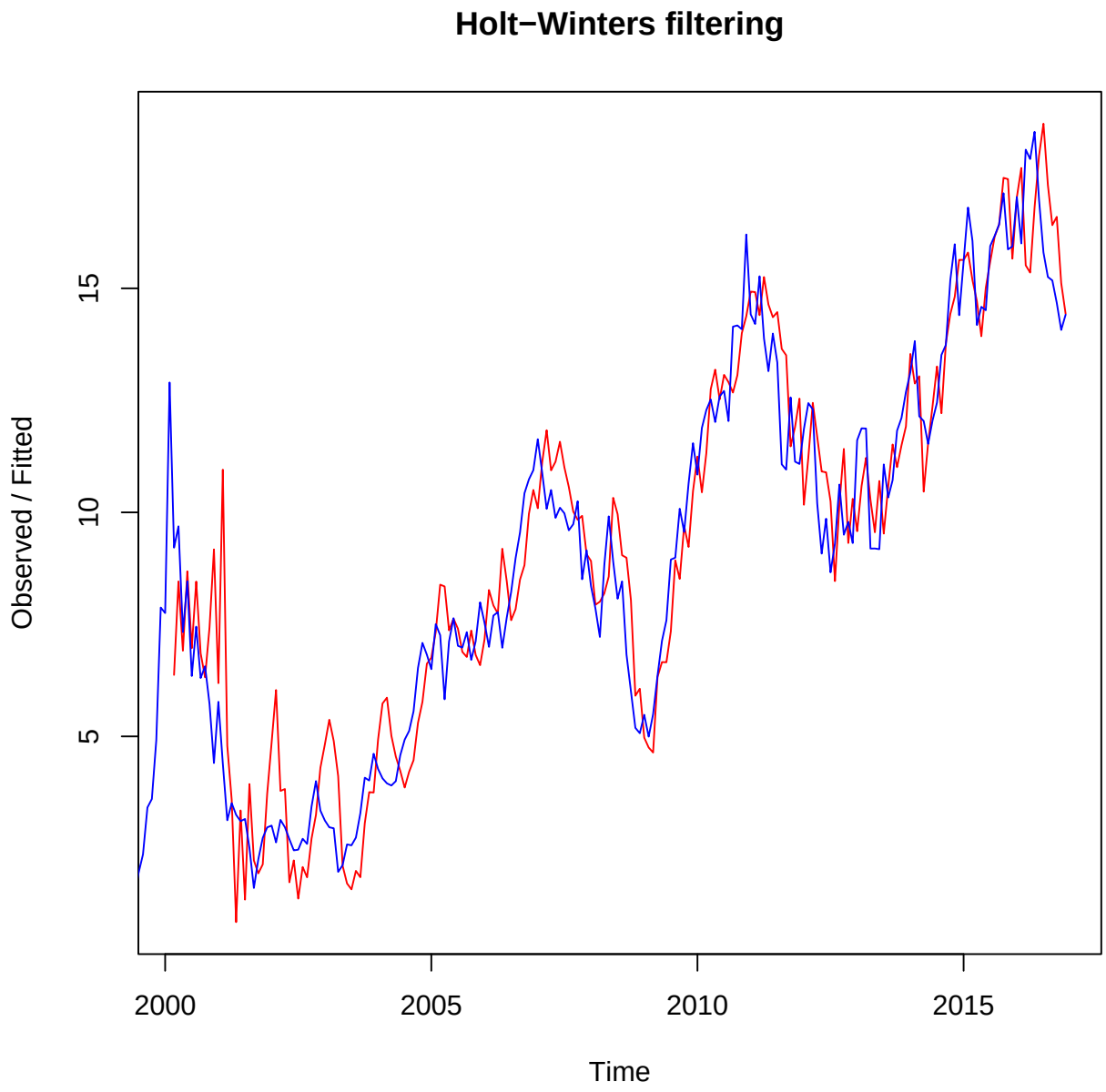
En los anteriores suavizados, no tuvieron en cuenta otros factores que contribuyen, como la *tendencia* y la *estacionalidad*, es decir, este método(Holt-Winters) lleva a cabo un suavizado exponencial en la presencia de tendencias y la estacionalidad

1. Suavizamos las curvas, es decir, eliminamos el ruido de las mismas.
2. Utilizamos el paquete *forecast*, para llevar a cabo una predicción de valores en el futuro.

```
setwd("C:\\Users\\81799\\OneDrive\\Documentos\\ESFM_CLASES\\Servicio Social ARTF\\Machine Learni
inf <- read.csv("INFY-monthly.csv")
#head(inf, 3): Quiero ver los primeros 3 datos de la tabla.
#tail(inf): Me da los últimos valores de la tabla
inf.ts <- ts(inf$Adj.Close, start=c(1999,3), frequency = 12)
#inf.ts
plot(inf.ts)
```



```
inf.hw <- HoltWinters(inf.ts)
#head(inf.hw)
plot(inf.hw, col = "blue", col.predicted = "red")
```




```
library(forecast)

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

inf.hw$SSE #Nos va el valor sse

## [1] 347.1345

inf.hw$alpha #Me da el valor de aalpha

##     alpha
## 0.5519187
```

```
inf.hw$beta #Me da el valor de beta

##       beta
## 0.01042698

inf.hw$gamma #Me da el valor de gamma

## gamma
##     1

head(inf.hw$fitted) #Nos da los primeros valores ajustados

##               xhat    level     trend      season
## Mar 2000 6.371099 5.038675 0.2702327  1.06219146
## Apr 2000 8.459056 6.876835 0.2865814  1.29563975
## May 2000 6.911948 7.842393 0.2936611 -1.22410629
## Jun 2000 8.684473 8.364917 0.2960474  0.02350837
## Jul 2000 6.968464 8.538907 0.2947748 -1.86521779
## Aug 2000 8.455786 8.490957 0.2912012 -0.32637179
```

Predicciones

```
infy.fore <- forecast(inf.hw, h=24)#Predicción a 2 años, (24 por que estabamos en meses)

#intervalos de confianza del 85% en tono oscuro y 95% en tono claro
infy.fore$lower #Nos da los valores de predicción

##                 80%       95%
## Jan 2017 13.820929 12.936811
## Feb 2017 13.442656 12.430351
## Mar 2017 13.872663 12.744433
## Apr 2017 12.028455 10.793086
## May 2017 11.468344 10.132484
## Jun 2017 10.255859  8.824740
## Jul 2017 10.437364  8.915217
## Aug 2017 10.819623  9.209944
## Sep 2017 11.239823  9.545552
## Oct 2017 11.626875  9.850518
## Nov 2017 11.442917  9.586633
## Dec 2017 11.645096  9.710761
## Jan 2018 12.310981 10.158075
## Feb 2018 12.042318  9.819248
## Mar 2018 12.560493 10.268172
## Apr 2018 10.789409  8.428657
## May 2018 10.291257  7.862816
## Jun 2018  9.132117  6.636656
## Jul 2018  9.360123  6.798248
## Aug 2018  9.783316  7.155581
## Sep 2018 10.239846  7.546750
## Oct 2018 10.659357  7.901357
## Nov 2018 10.504566  7.682078
## Dec 2018 10.733080  7.846482

infy.fore$upper #Zona superior, valores de predicción
```

```
##                80%      95%
## Jan 2017 17.16121 18.04533
## Feb 2017 17.26724 18.27954
## Mar 2017 18.13522 19.26345
## Apr 2017 16.69579 17.93116
## May 2017 16.51534 17.85120
## Jun 2017 15.66275 17.09387
## Jul 2017 16.18817 17.71032
## Aug 2017 16.90113 18.51081
## Sep 2017 17.64093 19.33520
## Oct 2017 18.33811 20.11447
## Nov 2017 18.45613 20.31241
## Dec 2017 18.95319 20.88752
## Jan 2018 20.44485 22.59776
## Feb 2018 20.44127 22.66434
## Mar 2018 21.22109 23.51341
## Apr 2018 19.70854 22.06929
## May 2018 19.46613 21.89457
## Jun 2018 18.56019 21.05566
## Jul 2018 19.03911 21.60099
## Aug 2018 19.71114 22.33887
## Sep 2018 20.41460 23.10770
## Oct 2018 21.07933 23.83733
## Nov 2018 21.16818 23.99067
## Dec 2018 21.63890 24.52550
```

1.8. ARIMA

El método ARIMA (autorregresivo de medias móviles) es una clase de modelo que explican una serie de tiemp dada basado en su pasado (tanto en valores como en errores cometidos), de modo que su ecuación puede ser usada para predecir el futuro.
Estos modelos se caracterizan con 3 parámetros: p, d y q donde:

- p es el orden de la parte autorregresiva (Ar(p))
- d es el número de diferenciaciones requeridas para tener una serie estacionaria (l)
- q es el orden de la parte de medias móviles (Ma(q))

¿ Cómo conseguimos una serie estacionaria?

La manera más usual es diferenciandola. Esto significa restar el valor previo del valor actual. A veces, dependiendo de la complejidad de la serie, se requiere una o más diferenciaciones.
El valor d es, por lo tanto, el mínimo número de diferenciaciones que se requieren para obtener una serie estacionaria. Si la serie original ya era estacionaria, entonces $d = 0$.
Un modelo Arima es aquel donde la serie original ha sido diferenciada al menos una vez para volverla estacionaria, y luego se le combinan las partes autorregresivas y de medias móviles, de modo que se satisface:

$$x_t = AR(p) + MA(q) + c$$

donde:

$$Ar(p) = \phi_1 x_{t-1} + \phi_2 x_{t-2} + \cdots + \phi_p x_{t-p}$$
$$MA(p) = \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \cdots + \theta_q \epsilon_{t-q}$$

Modelo de medias móviles MA(q)

Los modelos de medias móviles predicen el valor actual de nuestra serie temporal en función de los residuos pasados. Un modelo de medias móviles de orden simple, de orden uno, sólo consideraría el valor del residuo en el período anterior, y el modelo se expresaría de la siguiente manera:

$$x_t = \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \cdots + \theta_q \epsilon_{t-q} + \epsilon_t$$

Donde x_t es el valor de interés, del período actual de nuestra serie de tiempo, θ_k son coeficientes que debemos estimar, ϵ_k son los residuos.
No funciona bien si los datos no son estacionarios.

Modelo Autorregresivo

Los modelos autoregresivos predicen el valor actual de nuestra serie temporal en función de los valores pasados. Un modelo autoregresivo de orden simple, de orden uno, sólo consideraría el valor del período anterior, y el modelo se expresaría de la siguiente manera:

$$x_t = c + \phi x_{t-1} + \epsilon_t$$

Donde x_t es el valor de interés, del período actual de nuestra serie de tiempo, c es una constante, ϕ es el coeficiente que debemos estimar, ϵ_t es el residuo en el período actual y x_{t-1} es el valor de la serie en el período anterior.
Observación: No funcionan bien si los datos no son estacionarios
Sabemos que la ACF captura los eectos directos e indirectos del valor anterior sobre el valor presente. Como queremos un modelo eficiente solo queremos considerar aquellos retrasos que tengan un efecto directo y significativo sobre el período presente. Por lo tanto, debemos examinar la PACF antes de construir un modelo con demasiados coeficientes de retraso.
Dada la situación anterior, el modelo se puede hacer más complejo si así se requiere, aumentando el número de retraso:

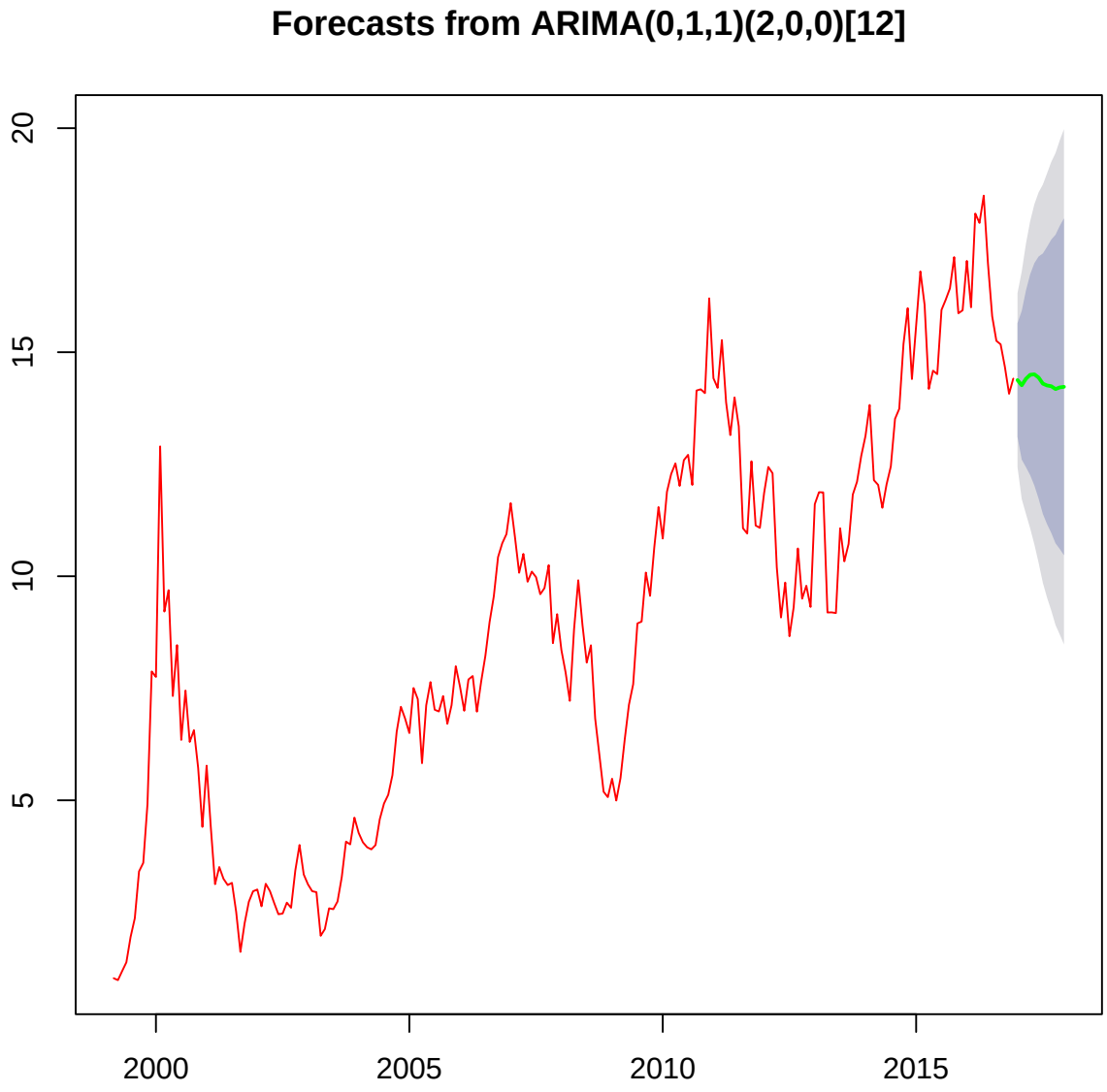
$$\begin{aligned} AR(1) &\qquad x_t = c + \phi 1 x_{t-1} + \epsilon_t \\ AR(2) &\qquad x_t c + \phi_1 x_{t-1} + \phi_1 t_{t-2} + \epsilon_t \\ AR(p) &\qquad x_t = c + \phi_1 x_{t-1} + \phi_2 x_{t-2} + \phi_p x_{t-p} + \epsilon_t \end{aligned}$$

Cómo encontrar d

El orden correcto de diferenciación es la diferenciación mínima requerida para obtener una serie casi estacionaria que oscila alrededor de una media bien definida y la gráfica de la función de correlación llega a 0 con bastante rápidez.
Si las autocorrelaciones son positivas para muchos retraso (10 o más), entonces la serie necesita una mayor diferenciación. Por otro lado, si la autocorrelación del retardo 1 en sí es demasiado negativa, entonces la serie probablemente esté sobrediferenciada.
Si es el caso en que no puede realmente decidir entre dos órdenes de diferenciación, elija el orden dé la menor desviación estándar en la serie diferenciada.

```
library(forecast)
setwd("C:\\Users\\81799\\OneDrive\\Documentos\\ESFM_CLASES\\Servicio Social ARTF\\Machine Learni
inf <- read.csv("INFY-monthly.csv")
inf.ts <- ts(inf$Adj.Close, start = c(1999,3), frequency = 12)
inf.arima <- auto.arima(inf.ts)
summary(inf.arima)

## Series: inf.ts
## ARIMA(0,1,1)(2,0,0)[12]
##
## Coefficients:
```

```
##           ma1    sar1     sar2
##       -0.1598  0.0505  -0.0533
## s.e.   0.0607  0.0799   0.0844
##
## sigma^2 = 0.982:  log likelihood = -298.84
## AIC=605.69   AICc=605.88   BIC=619.13
##
## Training set error measures:
##                       ME      RMSE       MAE       MPE    MAPE      MASE
## Training set 0.07413207 0.9816299 0.7329892 0.4829683 9.84262 0.3027089
##                      ACF1
## Training set -0.02620284

inf.fore <- forecast(inf.arima, h = 12)
plot(inf.fore, col = "red",
     fcol = "green")
```



```
inf.fore

##          Point Forecast    Lo 80    Hi 80     Lo 95    Hi 95
## Jan 2017       14.37850 13.10857 15.64844 12.436308 16.32070
## Feb 2017       14.26203 12.60336 15.92070 11.725308 16.79875
## Mar 2017       14.40743 12.43521 16.37965 11.391179 17.42368
## Apr 2017       14.49684 12.25449 16.73918 11.067466 17.92621
## May 2017       14.50587 12.02261 16.98913 10.708056 18.30369
## Jun 2017       14.43366 11.73087 17.13644 10.300108 18.56720
## Jul 2017       14.29770 11.39193 17.20347  9.853710 18.74169
## Aug 2017       14.25797 11.16250 17.35344  9.523854 18.99209
## Sep 2017       14.24052 10.96632 17.51473  9.233063 19.24799
## Oct 2017       14.17811 10.73444 17.62178  8.911471 19.44475
## Nov 2017       14.21441 10.60923 17.81959  8.700767 19.72806
## Dec 2017       14.22804 10.46828 17.98780  8.477983 19.97809
```