Análisis de los indices del mercado financiero europeo

By: Ael-Dev

Descripción:

Contains the daily closing prices of major European stock indices: Germany DAX (Ibis), Switzerland SMI, France CAC, and UK FTSE. The data are sampled in business time, i.e., weekends and holidays are omitted

Leer dataset

```
data(EuStockMarkets)
head(EuStockMarkets)
```

```
##          DAX    SMI    CAC   FTSE
## [1,] 1628.75 1678.1 1772.8 2443.6
## [2,] 1613.63 1688.5 1750.5 2460.2
## [3,] 1606.51 1678.6 1718.0 2448.2
## [4,] 1621.04 1684.1 1708.1 2470.4
## [5,] 1618.16 1686.6 1723.1 2484.7
## [6,] 1610.61 1671.6 1714.3 2466.8
```

```
class(EuStockMarkets)
```

```
## [1] "mts"    "ts"     "matrix" "array"
```

```
frequency(EuStockMarkets)
```

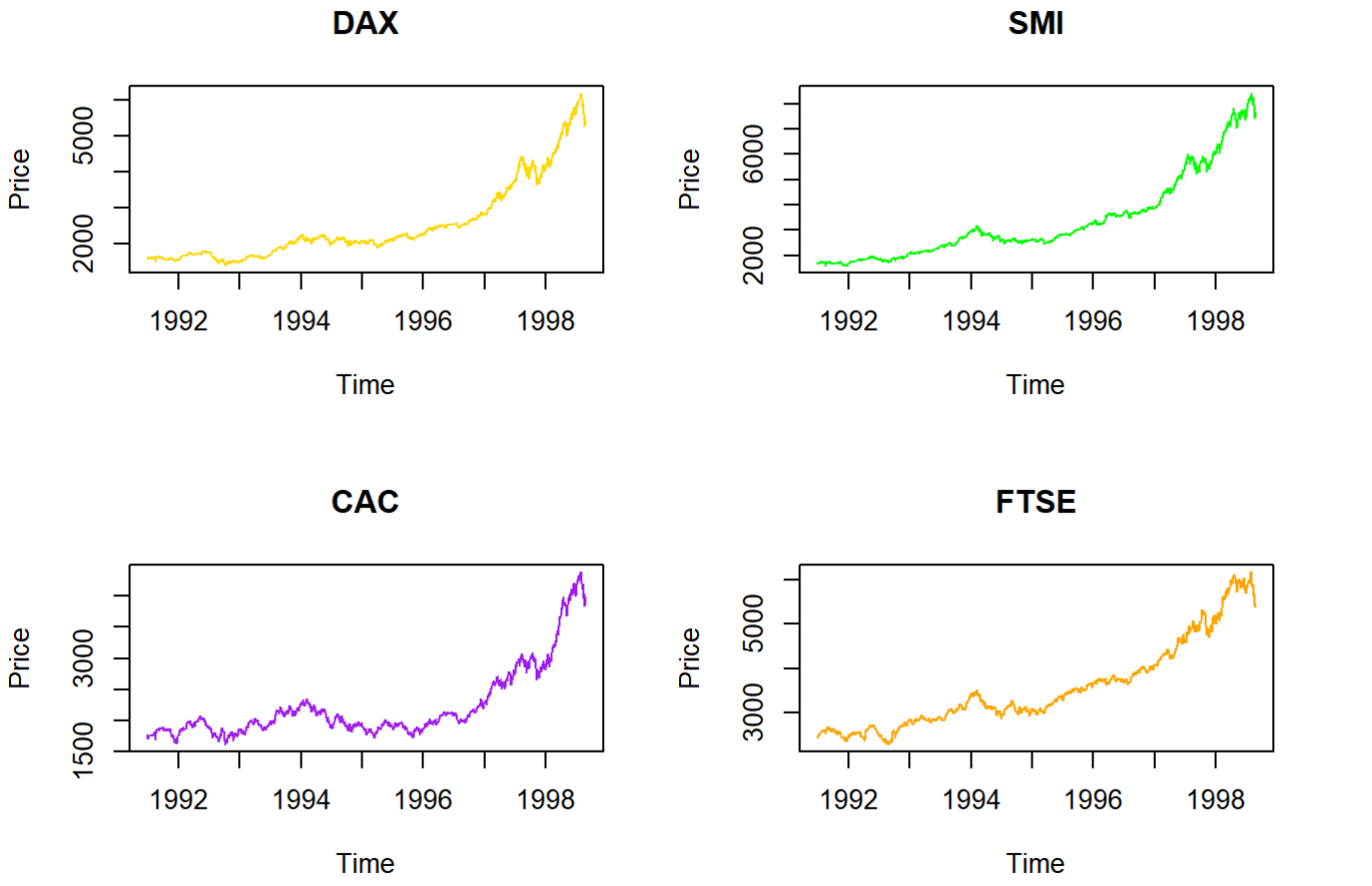
```
## [1] 260
```

```
start(EuStockMarkets)
```

```
## [1] 1991  130
```

Análisis exploratorio

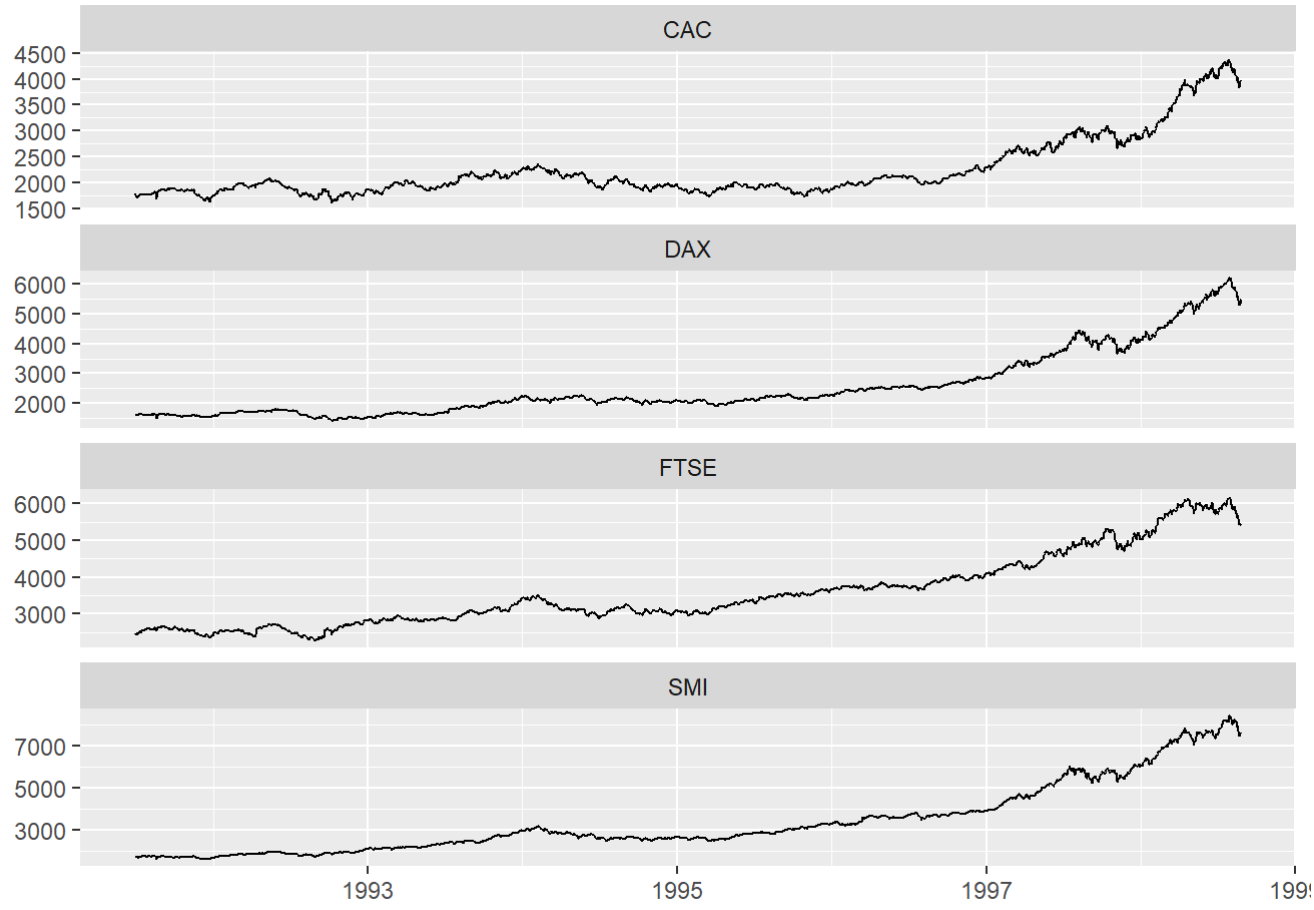
```
# Crear un lienzo para los 4 gráficos
par(mfrow = c(2, 2))

# Gráfico 1: Gold
plot(EuStockMarkets[, "DAX"], main = "DAX", ylab = "Price", type = "l", col = "gold")

# Gráfico 2: Silver
plot(EuStockMarkets[, "SMI"], main = "SMI", ylab = "Price", type = "l", col = "green")

# Gráfico 3: Platinum
plot(EuStockMarkets[, "CAC"], main = "CAC", ylab = "Price", type = "l", col = "purple")

# Gráfico 4: Palladium
plot(EuStockMarkets[, "FTSE"], main = "FTSE", ylab = "Price", type = "l", col = "orange")
```




```
# Con autoplot:
library(ggplot2)
library(ggfortify)
```

```
## Warning: package 'ggfortify' was built under R version 4.3.3
```

```
autoplot(EuStockMarkets)
```



Dividir la serie en conjunto de entrenamiento y de prueba

```
library(dplyr)
```

```
##
## 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
```

```
n_obs=30
end=dim(EuStockMarkets)[1]
X_train = EuStockMarkets[1:(end-n_obs),]
X_test = EuStockMarkets[(end-n_obs+1):end,]
dim(X_test)
```

```
## [1] 30  4
```

Prueba de estacionariedad

H0 (Hipótesis Nula): La serie temporal es no estacionaria, lo que implica que tiene una raíz unitaria.

H1 (Hipótesis Alternativa): La serie temporal es estacionaria, lo que significa que no tiene una raíz unitaria.

```
library(tseries)
```

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

```
apply(X_train, 2, adf.test) #2 para especificar que lo queremos aplicar por columnas
```

```
## Warning in FUN(newX[, i], ...): p-value greater than printed p-value

## Warning in FUN(newX[, i], ...): p-value greater than printed p-value

## Warning in FUN(newX[, i], ...): p-value greater than printed p-value
```

```
## $DAX
##
##  Augmented Dickey-Fuller Test
##
## data:  newX[, i]
## Dickey-Fuller = 1.565, Lag order = 12, p-value = 0.99
## alternative hypothesis: stationary
##
##
## $SMI
##
##  Augmented Dickey-Fuller Test
##
## data:  newX[, i]
## Dickey-Fuller = 1.0746, Lag order = 12, p-value = 0.99
## alternative hypothesis: stationary
##
##
## $CAC
##
##  Augmented Dickey-Fuller Test
##
## data:  newX[, i]
## Dickey-Fuller = 1.444, Lag order = 12, p-value = 0.99
## alternative hypothesis: stationary
##
##
## $FTSE
##
##  Augmented Dickey-Fuller Test
##
## data:  newX[, i]
## Dickey-Fuller = -0.85073, Lag order = 12, p-value = 0.9571
## alternative hypothesis: stationary
```

Dado que todos los p-values son > 0.05, se rechaza h1, por lo que se sugiere diferenciar las series

Diferenciando todo la mts

```
library(MTS)
```

```
## Warning: package 'MTS' was built under R version 4.3.3
```

```
stnry = diffM(X_train)
```

```
## Aplicando el test:
apply(stnry, 2, adf.test)
```

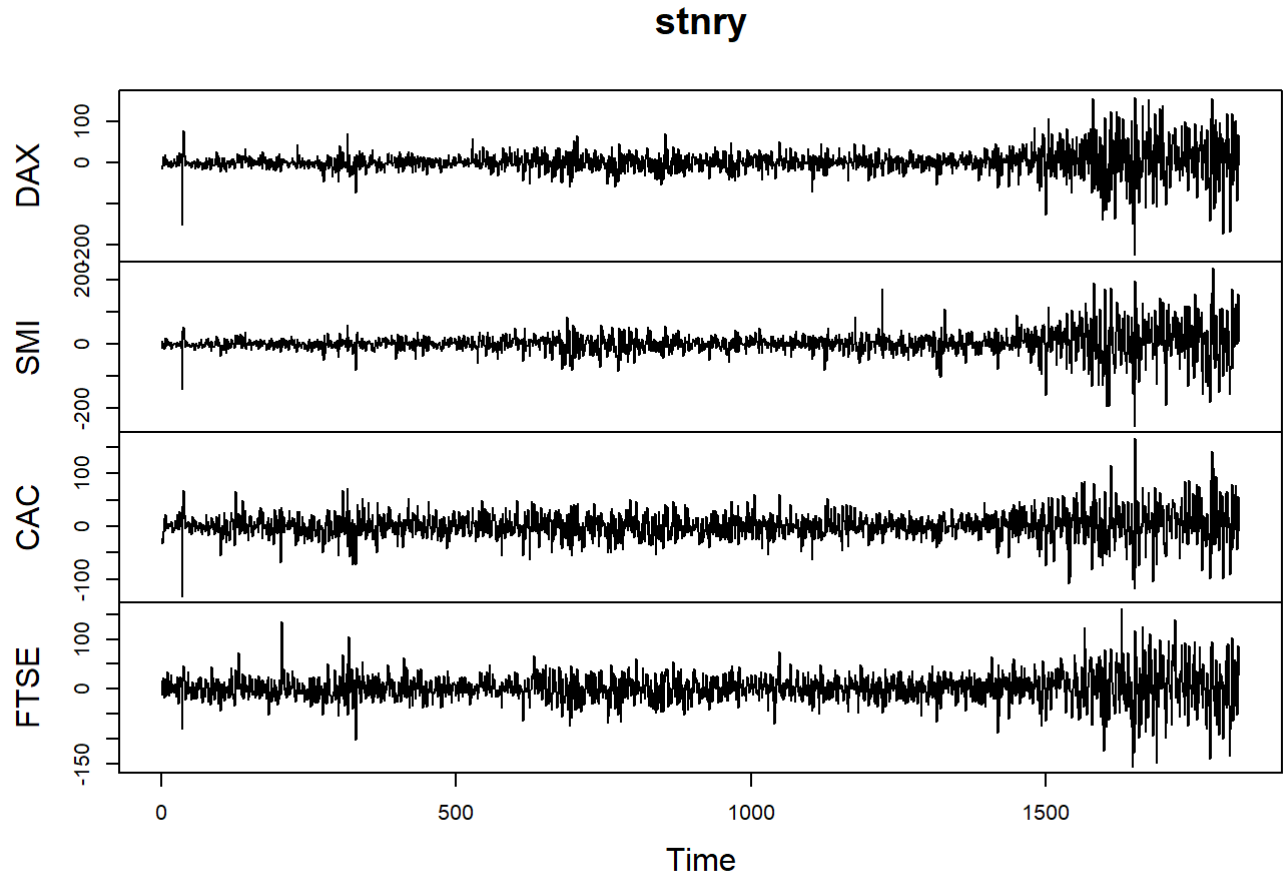
```
## Warning in FUN(newX[, i], ...): p-value smaller than printed p-value

## Warning in FUN(newX[, i], ...): p-value smaller than printed p-value

## Warning in FUN(newX[, i], ...): p-value smaller than printed p-value

## Warning in FUN(newX[, i], ...): p-value smaller than printed p-value
```

```
## $DAX
##
##  Augmented Dickey-Fuller Test
##
## data:  newX[, i]
## Dickey-Fuller = -10.833, Lag order = 12, p-value = 0.01
## alternative hypothesis: stationary
##
##
## $SMI
##
##  Augmented Dickey-Fuller Test
##
## data:  newX[, i]
## Dickey-Fuller = -11.266, Lag order = 12, p-value = 0.01
## alternative hypothesis: stationary
##
##
## $CAC
##
##  Augmented Dickey-Fuller Test
##
## data:  newX[, i]
## Dickey-Fuller = -11.426, Lag order = 12, p-value = 0.01
## alternative hypothesis: stationary
##
##
## $FTSE
##
##  Augmented Dickey-Fuller Test
##
## data:  newX[, i]
## Dickey-Fuller = -11.525, Lag order = 12, p-value = 0.01
## alternative hypothesis: stationary
```

Dado que todos los p-values son < 0.05, se acepta h1, esto quiere decir que todas las series son estacionarias

VAR modeling

```
plot.ts(stnry) # plot de las series diferenciadas
```



Se observa que las series lucen como una serie estacionaria.

Identificación del orden del modelo

```
library(vars)
```

```
## Warning: package 'vars' was built under R version 4.3.3
```

```
## Loading required package: MASS
```

```
##
## Attaching package: 'MASS'
```

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

```
## Loading required package: strucchange
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

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

```
## Loading required package: sandwich
```

```
## Loading required package: urca
```

```
## Loading required package: lmtest
```

```
##
## Attaching package: 'vars'
```

```
## The following object is masked from 'package:MTS':
##
##     VAR
```

```
VARselect(stnry, type = "none", lag.max = 10)
```

```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##      9      1      1      9
##
## $criteria
##                   1            2            3            4            5
## AIC(n) 2.501576e+01 2.502058e+01 2.501810e+01 2.501300e+01 2.501035e+01
## HQ(n)  2.503362e+01 2.505632e+01 2.507171e+01 2.508447e+01 2.509969e+01
## SC(n)  2.506419e+01 2.511745e+01 2.516340e+01 2.520673e+01 2.525251e+01
## FPE(n) 7.314841e+10 7.350246e+10 7.332023e+10 7.294723e+10 7.275442e+10
##                   6            7            8            9           10
## AIC(n) 2.501013e+01 2.500973e+01 2.501344e+01 2.500671e+01 2.501239e+01
## HQ(n)  2.511734e+01 2.513481e+01 2.515639e+01 2.516753e+01 2.519108e+01
## SC(n)  2.530072e+01 2.534875e+01 2.540090e+01 2.544260e+01 2.549671e+01
## FPE(n) 7.273862e+10 7.270949e+10 7.298058e+10 7.249142e+10 7.290485e+10
```

Según el criterio AIC sugiere un modelo de orden 9

Creando el modelo

```
var.a <- vars::VAR(stnry,

                   lag.max = 10,

                   ic = "AIC",

                   type = "none")

summary(var.a)
```

```
##
## VAR Estimation Results:
## ========================
## Endogenous variables: DAX, SMI, CAC, FTSE
## Deterministic variables: none
## Sample size: 1820
## Log Likelihood: -32941.436
## Roots of the characteristic polynomial:
## 0.8205 0.8205 0.8086 0.8086 0.8033 0.8033 0.789 0.7836 0.7836 0.7682 0.7682 0.7669 0.7669
0.7485 0.7485 0.7455 0.7455 0.7414 0.7414 0.7229 0.7229 0.7223 0.7223 0.7212 0.7009 0.7009 0.
6889 0.6889 0.6714 0.6714 0.6242 0.5214 0.5214 0.4495 0.4067 0.4067
## Call:
## vars::VAR(y = stnry, type = "none", lag.max = 10, ic = "AIC")
##
##
## Estimation results for equation DAX:
## ====================================
## DAX = DAX.l1 + SMI.l1 + CAC.l1 + FTSE.l1 + DAX.l2 + SMI.l2 + CAC.l2 + FTSE.l2 + DAX.l3 + S
MI.l3 + CAC.l3 + FTSE.l3 + DAX.l4 + SMI.l4 + CAC.l4 + FTSE.l4 + DAX.l5 + SMI.l5 + CAC.l5 + FT
SE.l5 + DAX.l6 + SMI.l6 + CAC.l6 + FTSE.l6 + DAX.l7 + SMI.l7 + CAC.l7 + FTSE.l7 + DAX.l8 + SM
I.l8 + CAC.l8 + FTSE.l8 + DAX.l9 + SMI.l9 + CAC.l9 + FTSE.l9
##
##          Estimate Std. Error t value Pr(>|t|)
## DAX.l1   0.024697   0.041810   0.591  0.55480
## SMI.l1  -0.087076   0.030185  -2.885  0.00396 **
## CAC.l1   0.038452   0.046335   0.830  0.40672
## FTSE.l1  0.054659   0.036319   1.505  0.13251
## DAX.l2  -0.012175   0.041722  -0.292  0.77047
## SMI.l2  -0.019378   0.030323  -0.639  0.52287
## CAC.l2   0.070666   0.046255   1.528  0.12675
## FTSE.l2 -0.050218   0.036603  -1.372  0.17025
## DAX.l3  -0.089221   0.041728  -2.138  0.03264 *
## SMI.l3   0.035712   0.030449   1.173  0.24101
## CAC.l3   0.041188   0.046188   0.892  0.37265
## FTSE.l3  0.019960   0.036560   0.546  0.58517
## DAX.l4  -0.066196   0.041838  -1.582  0.11378
## SMI.l4   0.040886   0.030483   1.341  0.18001
## CAC.l4   0.114717   0.046365   2.474  0.01345 *
## FTSE.l4 -0.080836   0.036507  -2.214  0.02694 *
## DAX.l5  -0.024973   0.042070  -0.594  0.55285
## SMI.l5  -0.080351   0.030537  -2.631  0.00858 **
## CAC.l5   0.072917   0.046341   1.573  0.11578
## FTSE.l5 -0.033139   0.036560  -0.906  0.36483
## DAX.l6  -0.017055   0.042043  -0.406  0.68504
## SMI.l6   0.032370   0.030565   1.059  0.28971
## CAC.l6   0.052174   0.046296   1.127  0.25991
## FTSE.l6  0.029614   0.036581   0.810  0.41831
## DAX.l7   0.058050   0.042067   1.380  0.16778
## SMI.l7   0.019055   0.030484   0.625  0.53200
## CAC.l7  -0.053602   0.046099  -1.163  0.24508
## FTSE.l7 -0.066428   0.036645  -1.813  0.07004 .
## DAX.l8  -0.038252   0.042095  -0.909  0.36364
## SMI.l8   0.066159   0.030565   2.165  0.03055 *
## CAC.l8  -0.080179   0.046074  -1.740  0.08200 .
## FTSE.l8  0.053379   0.036693   1.455  0.14591
```

```
## DAX.l9   0.003293   0.041893   0.079  0.93737
## SMI.l9   0.010637   0.030320   0.351  0.72575
## CAC.l9   0.106768   0.046154   2.313  0.02082 *
## FTSE.l9 -0.020306   0.036484  -0.557  0.57788
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 30.91 on 1784 degrees of freedom
## Multiple R-Squared: 0.04756, Adjusted R-squared: 0.02834
## F-statistic: 2.475 on 36 and 1784 DF,  p-value: 3.27e-06
##
##
## Estimation results for equation SMI:
## ====================================
## SMI = DAX.l1 + SMI.l1 + CAC.l1 + FTSE.l1 + DAX.l2 + SMI.l2 + CAC.l2 + FTSE.l2 + DAX.l3 + S
MI.l3 + CAC.l3 + FTSE.l3 + DAX.l4 + SMI.l4 + CAC.l4 + FTSE.l4 + DAX.l5 + SMI.l5 + CAC.l5 + FT
SE.l5 + DAX.l6 + SMI.l6 + CAC.l6 + FTSE.l6 + DAX.l7 + SMI.l7 + CAC.l7 + FTSE.l7 + DAX.l8 + SM
I.l8 + CAC.l8 + FTSE.l8 + DAX.l9 + SMI.l9 + CAC.l9 + FTSE.l9
##
##          Estimate Std. Error t value Pr(>|t|)
## DAX.l1   0.064478   0.050163   1.285 0.198828
## SMI.l1  -0.009789   0.036215  -0.270 0.786955
## CAC.l1  -0.014208   0.055592  -0.256 0.798307
## FTSE.l1  0.101650   0.043575   2.333 0.019770 *
## DAX.l2  -0.016300   0.050058  -0.326 0.744739
## SMI.l2   0.021741   0.036381   0.598 0.550188
## CAC.l2   0.078407   0.055496   1.413 0.157878
## FTSE.l2 -0.043919   0.043916  -1.000 0.317408
## DAX.l3  -0.165465   0.050064  -3.305 0.000968 ***
## SMI.l3   0.012468   0.036532   0.341 0.732934
## CAC.l3   0.091621   0.055416   1.653 0.098436 .
## FTSE.l3  0.081139   0.043864   1.850 0.064512 .
## DAX.l4  -0.167636   0.050196  -3.340 0.000856 ***
## SMI.l4   0.071381   0.036573   1.952 0.051126 .
## CAC.l4   0.128507   0.055628   2.310 0.020996 *
## FTSE.l4 -0.037297   0.043800  -0.852 0.394595
## DAX.l5  -0.072508   0.050475  -1.437 0.151032
## SMI.l5  -0.061432   0.036637  -1.677 0.093764 .
## CAC.l5   0.090843   0.055600   1.634 0.102461
## FTSE.l5 -0.027523   0.043864  -0.627 0.530427
## DAX.l6  -0.078770   0.050443  -1.562 0.118565
## SMI.l6   0.021149   0.036671   0.577 0.564209
## CAC.l6   0.080483   0.055546   1.449 0.147527
## FTSE.l6  0.020769   0.043889   0.473 0.636122
## DAX.l7   0.104714   0.050472   2.075 0.038157 *
## SMI.l7  -0.007977   0.036574  -0.218 0.827363
## CAC.l7  -0.083889   0.055309  -1.517 0.129514
## FTSE.l7 -0.052147   0.043967  -1.186 0.235758
## DAX.l8   0.002721   0.050505   0.054 0.957046
## SMI.l8   0.058545   0.036671   1.596 0.110558
## CAC.l8  -0.061109   0.055279  -1.105 0.269109
## FTSE.l8 -0.011009   0.044023  -0.250 0.802560
## DAX.l9   0.108148   0.050263   2.152 0.031559 *
## SMI.l9  -0.089295   0.036378  -2.455 0.014198 *
## CAC.l9   0.044129   0.055374   0.797 0.425599
```

```
## FTSE.l9  0.058559   0.043773   1.338 0.181138
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 37.09 on 1784 degrees of freedom
## Multiple R-Squared: 0.05362, Adjusted R-squared: 0.03452
## F-statistic: 2.808 on 36 and 1784 DF,  p-value: 7.953e-08
##
##
## Estimation results for equation CAC:
## ====================================
## CAC = DAX.l1 + SMI.l1 + CAC.l1 + FTSE.l1 + DAX.l2 + SMI.l2 + CAC.l2 + FTSE.l2 + DAX.l3 + S
MI.l3 + CAC.l3 + FTSE.l3 + DAX.l4 + SMI.l4 + CAC.l4 + FTSE.l4 + DAX.l5 + SMI.l5 + CAC.l5 + FT
SE.l5 + DAX.l6 + SMI.l6 + CAC.l6 + FTSE.l6 + DAX.l7 + SMI.l7 + CAC.l7 + FTSE.l7 + DAX.l8 + SM
I.l8 + CAC.l8 + FTSE.l8 + DAX.l9 + SMI.l9 + CAC.l9 + FTSE.l9
##
##           Estimate Std. Error t value Pr(>|t|)
## DAX.l1   5.495e-03  3.418e-02   0.161 0.872299
## SMI.l1  -5.721e-02  2.468e-02  -2.318 0.020535 *
## CAC.l1   2.303e-02  3.788e-02   0.608 0.543243
## FTSE.l1  7.187e-02  2.969e-02   2.421 0.015595 *
## DAX.l2  -4.537e-03  3.411e-02  -0.133 0.894191
## SMI.l2  -7.630e-06  2.479e-02   0.000 0.999754
## CAC.l2   8.279e-02  3.781e-02   2.189 0.028690 *
## FTSE.l2 -6.682e-02  2.992e-02  -2.233 0.025673 *
## DAX.l3  -4.704e-02  3.411e-02  -1.379 0.168061
## SMI.l3   3.223e-02  2.489e-02   1.295 0.195584
## CAC.l3  -3.821e-02  3.776e-02  -1.012 0.311748
## FTSE.l3  8.899e-03  2.989e-02   0.298 0.765937
## DAX.l4  -1.220e-01  3.420e-02  -3.567 0.000371 ***
## SMI.l4   6.556e-02  2.492e-02   2.631 0.008590 **
## CAC.l4   7.674e-02  3.790e-02   2.025 0.043052 *
## FTSE.l4 -4.055e-02  2.985e-02  -1.359 0.174450
## DAX.l5  -6.411e-02  3.439e-02  -1.864 0.062473 .
## SMI.l5  -3.124e-02  2.496e-02  -1.251 0.211004
## CAC.l5   3.877e-02  3.789e-02   1.023 0.306228
## FTSE.l5  5.082e-03  2.989e-02   0.170 0.864992
## DAX.l6  -1.290e-02  3.437e-02  -0.375 0.707563
## SMI.l6   2.044e-02  2.499e-02   0.818 0.413348
## CAC.l6   1.941e-02  3.785e-02   0.513 0.608188
## FTSE.l6  1.820e-02  2.991e-02   0.609 0.542884
## DAX.l7   9.270e-02  3.439e-02   2.695 0.007097 **
## SMI.l7  -1.125e-02  2.492e-02  -0.451 0.651803
## CAC.l7  -3.482e-02  3.769e-02  -0.924 0.355681
## FTSE.l7 -7.676e-02  2.996e-02  -2.562 0.010481 *
## DAX.l8   1.913e-02  3.441e-02   0.556 0.578444
## SMI.l8   1.069e-02  2.499e-02   0.428 0.668911
## CAC.l8  -9.993e-02  3.767e-02  -2.653 0.008050 **
## FTSE.l8  4.787e-02  3.000e-02   1.596 0.110739
## DAX.l9   3.677e-02  3.425e-02   1.074 0.283101
## SMI.l9  -1.766e-02  2.479e-02  -0.713 0.476217
## CAC.l9   3.237e-02  3.773e-02   0.858 0.391039
## FTSE.l9 -1.692e-02  2.983e-02  -0.567 0.570694
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
##
## Residual standard error: 25.27 on 1784 degrees of freedom
## Multiple R-Squared: 0.04492, Adjusted R-squared: 0.02565
## F-statistic: 2.331 on 36 and 1784 DF,  p-value: 1.515e-05
##
##
## Estimation results for equation FTSE:
## ====================================
## FTSE = DAX.l1 + SMI.l1 + CAC.l1 + FTSE.l1 + DAX.l2 + SMI.l2 + CAC.l2 + FTSE.l2 + DAX.l3 +
SMI.l3 + CAC.l3 + FTSE.l3 + DAX.l4 + SMI.l4 + CAC.l4 + FTSE.l4 + DAX.l5 + SMI.l5 + CAC.l5 + F
TSE.l5 + DAX.l6 + SMI.l6 + CAC.l6 + FTSE.l6 + DAX.l7 + SMI.l7 + CAC.l7 + FTSE.l7 + DAX.l8 + S
MI.l8 + CAC.l8 + FTSE.l8 + DAX.l9 + SMI.l9 + CAC.l9 + FTSE.l9
##
##          Estimate Std. Error t value Pr(>|t|)
## DAX.l1   0.025378   0.039361   0.645  0.51917
## SMI.l1  -0.084206   0.028416  -2.963  0.00308 **
## CAC.l1  -0.011552   0.043621  -0.265  0.79118
## FTSE.l1  0.158643   0.034191   4.640 3.74e-06 ***
## DAX.l2  -0.007697   0.039278  -0.196  0.84466
## SMI.l2  -0.002444   0.028546  -0.086  0.93177
## CAC.l2   0.017046   0.043545   0.391  0.69550
## FTSE.l2 -0.018463   0.034459  -0.536  0.59217
## DAX.l3  -0.066749   0.039283  -1.699  0.08946 .
## SMI.l3   0.018867   0.028665   0.658  0.51051
## CAC.l3   0.064868   0.043482   1.492  0.13592
## FTSE.l3 -0.001620   0.034418  -0.047  0.96246
## DAX.l4  -0.084911   0.039387  -2.156  0.03123 *
## SMI.l4   0.058068   0.028697   2.023  0.04318 *
## CAC.l4   0.110360   0.043649   2.528  0.01155 *
## FTSE.l4 -0.107840   0.034368  -3.138  0.00173 **
## DAX.l5  -0.029721   0.039605  -0.750  0.45309
## SMI.l5  -0.023582   0.028748  -0.820  0.41215
## CAC.l5   0.104449   0.043627   2.394  0.01676 *
## FTSE.l5 -0.069614   0.034418  -2.023  0.04326 *
## DAX.l6  -0.028954   0.039580  -0.732  0.46455
## SMI.l6   0.046586   0.028774   1.619  0.10562
## CAC.l6   0.064943   0.043584   1.490  0.13639
## FTSE.l6 -0.057706   0.034438  -1.676  0.09398 .
## DAX.l7   0.032686   0.039603   0.825  0.40929
## SMI.l7   0.070058   0.028698   2.441  0.01473 *
## CAC.l7  -0.054059   0.043399  -1.246  0.21306
## FTSE.l7 -0.074192   0.034499  -2.151  0.03164 *
## DAX.l8   0.004442   0.039629   0.112  0.91076
## SMI.l8   0.040585   0.028774   1.410  0.15857
## CAC.l8  -0.076945   0.043375  -1.774  0.07624 .
## FTSE.l8  0.018622   0.034543   0.539  0.58990
## DAX.l9   0.026218   0.039439   0.665  0.50629
## SMI.l9  -0.014494   0.028544  -0.508  0.61166
## CAC.l9   0.067197   0.043450   1.547  0.12215
## FTSE.l9 -0.015438   0.034347  -0.449  0.65314
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 29.1 on 1784 degrees of freedom
```

```
## Multiple R-Squared: 0.05837, Adjusted R-squared: 0.03937
## F-statistic: 3.072 on 36 and 1784 DF,  p-value: 3.635e-09
##
##
##
## Covariance matrix of residuals:
##        DAX    SMI   CAC  FTSE
## DAX  949.8  838.2 575.3 590.5
## SMI  838.2 1364.3 594.2 650.7
## CAC  575.3  594.2 636.2 485.3
## FTSE 590.5  650.7 485.3 843.6
##
## Correlation matrix of residuals:
##         DAX    SMI    CAC   FTSE
## DAX  1.0000 0.7363 0.7400 0.6597
## SMI  0.7363 1.0000 0.6378 0.6065
## CAC  0.7400 0.6378 1.0000 0.6625
## FTSE 0.6597 0.6065 0.6625 1.0000
```

Causalidad de Granger

```
causality(var.a, cause = c("DAX"))
```

```
## $Granger
##
##  Granger causality H0: DAX do not Granger-cause SMI CAC FTSE
##
## data:  VAR object var.a
## F-Test = 1.8679, df1 = 27, df2 = 7136, p-value = 0.004151
##
##
## $Instant
##
##  H0: No instantaneous causality between: DAX and SMI CAC FTSE
##
## data:  VAR object var.a
## Chi-squared = 738.28, df = 3, p-value < 2.2e-16
```

```
causality(var.a, cause = c("SMI"))
```

```
## $Granger
##
##  Granger causality H0: SMI do not Granger-cause DAX CAC FTSE
##
## data:  VAR object var.a
## F-Test = 2.2926, df1 = 27, df2 = 7136, p-value = 0.0001553
##
##
## $Instant
##
##  H0: No instantaneous causality between: SMI and DAX CAC FTSE
##
## data:  VAR object var.a
## Chi-squared = 666.86, df = 3, p-value < 2.2e-16
```

```
causality(var.a, cause = c("CAC"))
```

```
## $Granger
##
##  Granger causality H0: CAC do not Granger-cause DAX SMI FTSE
##
## data:  VAR object var.a
## F-Test = 1.539, df1 = 27, df2 = 7136, p-value = 0.03681
##
##
## $Instant
##
##  H0: No instantaneous causality between: CAC and DAX SMI FTSE
##
## data:  VAR object var.a
## Chi-squared = 689.61, df = 3, p-value < 2.2e-16
```

```
causality(var.a, cause = c("FTSE"))
```

```
## $Granger
##
##  Granger causality H0: FTSE do not Granger-cause DAX SMI CAC
##
## data:  VAR object var.a
## F-Test = 1.8144, df1 = 27, df2 = 7136, p-value = 0.006072
##
##
## $Instant
##
##  H0: No instantaneous causality between: FTSE and DAX SMI CAC
##
## data:  VAR object var.a
## Chi-squared = 623.12, df = 3, p-value < 2.2e-16
```

Se observa que los p-values son < 0.05, por lo que rechazamos H0: No instantaneous causality between: DAX and SMI CAC FTSE (la variable estudiada no causa a las demás), segun el resultado observamos que cada serie causa a las demás.

Diagnosis del modelo (Portmanteau test para objetos var)

verificando si existe autocorrelacion entre los residuos

- H0 (Hipótesis Nula): Los residuos son independientes (ruido blanco).
- H1 (Hipótesis Alternativa): Los residuos no son independientes y presentan autocorrelación.

Nota: si P-value < 0.05 => rechazar h0, Esto significa que los residuos del modelo no son completamente aleatorios, sugiriendo que el modelo podría no estar capturando toda la información predictiva en los datos, y podrían existir autocorrelaciones en los residuos.

```
serial.test(var.a)
```

```
##
##  Portmanteau Test (asymptotic)
##
## data:  Residuals of VAR object var.a
## Chi-squared = 186.87, df = 112, p-value = 1.169e-05
```

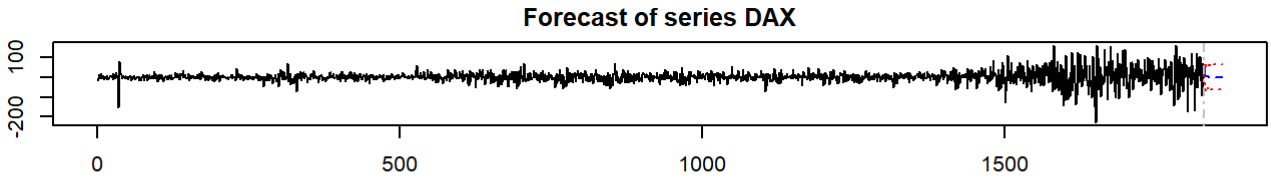
El p-value es < 0.05, se desea obtener un p-valor > 0.05, por lo que se sugiere lo siguiente:

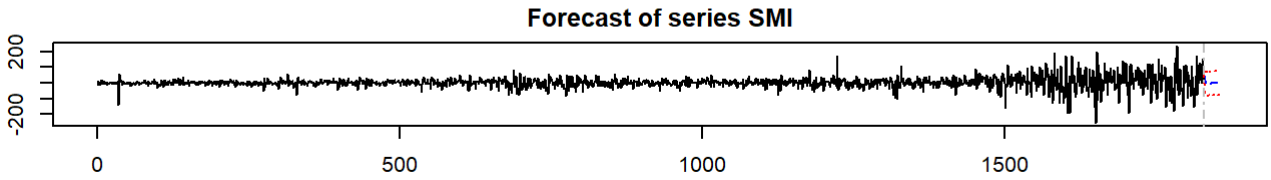
Posibles soluciones: a) Cambiar el orden del modelo. b) Cambiar el tipo de modelo. c) Añadir otro paso de diferenciación o transformar con logaritmos.

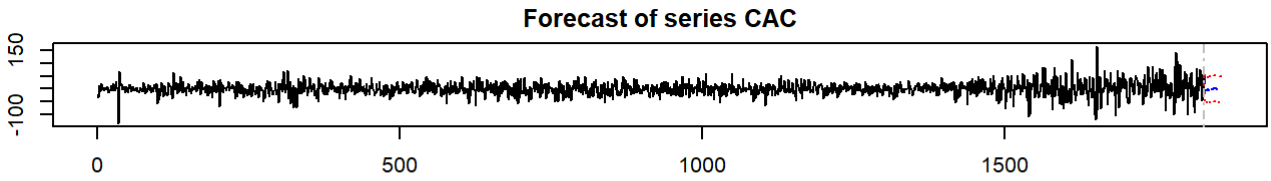
Estos modelos son a menudo tan complejos que no se puede alcanzar un resultado completamente satisfactorio sin cambiar mucho los datos con logaritmos o varios pasos de diferencias.

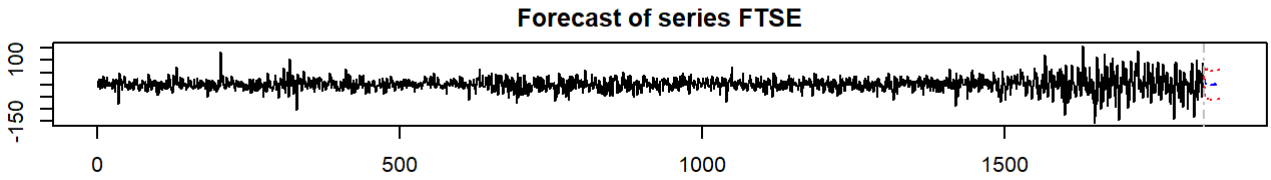
Forecasting usando el modelo VAR (Hallando los pronósticos)

```
# Adjust the margins to reduce the size
par(mar = c(4, 2, 2, 3) + 0.1)

# Generate the forecast
fcast = predict(var.a, n.ahead = 30)

# Plot the forecast
plot(fcast)
```










```
# Solo para DAX
DAX = fcast$fcst[1]; DAX
```

```
## $DAX
##                fcst     lower    upper       CI
##  [1,]  8.1127694961 -52.47420 68.69974 60.58697
##  [2,] 11.5668729141 -49.18913 72.32288 60.75600
##  [3,] -7.1425046516 -67.98445 53.69944 60.84195
##  [4,] 10.1749045776 -50.74798 71.09778 60.92288
##  [5,]  4.7651921576 -56.33852 65.86890 61.10371
##  [6,]  4.1305507676 -57.25472 65.51583 61.38528
##  [7,]  7.9971190585 -53.56645 69.56068 61.56357
##  [8,] -3.7805588693 -65.44605 57.88493 61.66549
##  [9,]  4.1046116383 -57.74505 65.95427 61.84966
## [10,] -1.8904051829 -63.93267 60.15186 62.04226
## [11,]  1.6996361048 -60.34865 63.74792 62.04829
## [12,] -1.7632833455 -63.82125 60.29468 62.05796
## [13,]  0.5895265094 -61.47231 62.65136 62.06184
## [14,] -2.0539225182 -64.11823 60.01039 62.06431
## [15,] -0.3721838750 -62.45010 61.70574 62.07792
## [16,]  0.4585620349 -61.62243 62.53955 62.08099
## [17,]  0.2242724964 -61.85953 62.30808 62.08381
## [18,]  0.2856580800 -61.80149 62.37280 62.08714
## [19,] -0.1634292597 -62.25117 61.92431 62.08774
## [20,] -0.0619888128 -62.14988 62.02590 62.08789
## [21,] -0.2389747627 -62.32723 61.84928 62.08825
## [22,]  0.3263661355 -61.76197 62.41470 62.08833
## [23,]  0.0430359565 -62.04547 62.13154 62.08850
## [24,]  0.1991054097 -61.88950 62.28771 62.08861
## [25,] -0.0083420367 -62.09700 62.08032 62.08866
## [26,]  0.0546800519 -62.03404 62.14340 62.08872
## [27,] -0.0039392220 -62.09270 62.08482 62.08876
## [28,]  0.0432783667 -62.04548 62.13204 62.08876
## [29,]  0.0005118017 -62.08825 62.08928 62.08877
## [30,] -0.0389748992 -62.12775 62.04980 62.08878
```

```
# Extrayendo la columna de pronósticos
x = DAX$DAX[,1]; x
```

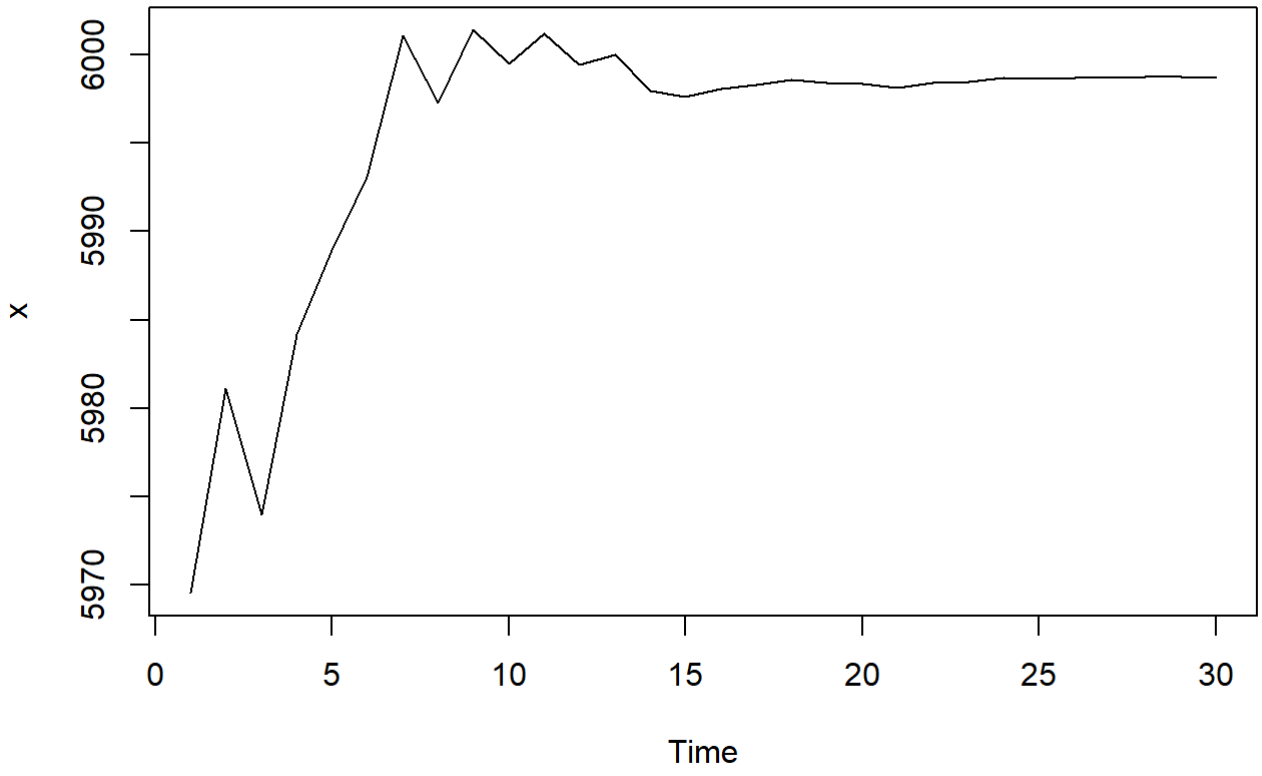
```
##  [1]  8.1127694961 11.5668729141 -7.1425046516 10.1749045776  4.7651921576
##  [6]  4.1305507676  7.9971190585 -3.7805588693  4.1046116383 -1.8904051829
## [11]  1.6996361048 -1.7632833455  0.5895265094 -2.0539225182 -0.3721838750
## [16]  0.4585620349  0.2242724964  0.2856580800 -0.1634292597 -0.0619888128
## [21] -0.2389747627  0.3263661355  0.0430359565  0.1991054097 -0.0083420367
## [26]  0.0546800519 -0.0039392220  0.0432783667  0.0005118017 -0.0389748992
```

Invirtiendo la diferenciación

```
tail(X_train)
```

```
##             DAX    SMI    CAC   FTSE
## [1825,] 5870.49 7816.9 4215.7 5877.4
## [1826,] 5933.73 7881.9 4248.2 5884.5
## [1827,] 5841.83 7882.0 4203.5 5832.5
## [1828,] 5910.51 8038.2 4260.7 5919.9
## [1829,] 5905.15 8047.3 4252.1 5960.2
## [1830,] 5961.45 8099.0 4304.4 5988.4
```

```
x = cumsum(x) + 5961.45

plot.ts(x)
```



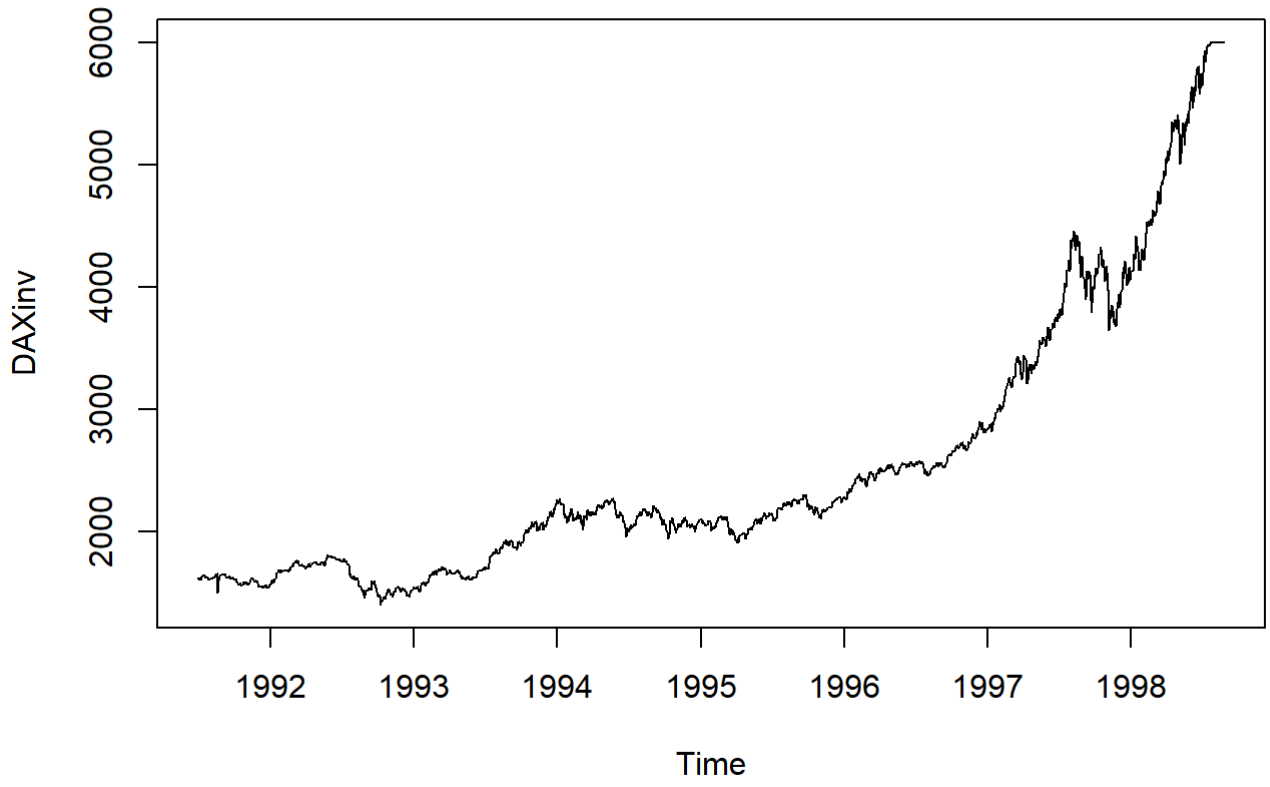

```
# Combinando los datos reales y la predicción en una sola serie de tiempo
start(EuStockMarkets)
```

```
## [1] 1991  130
```

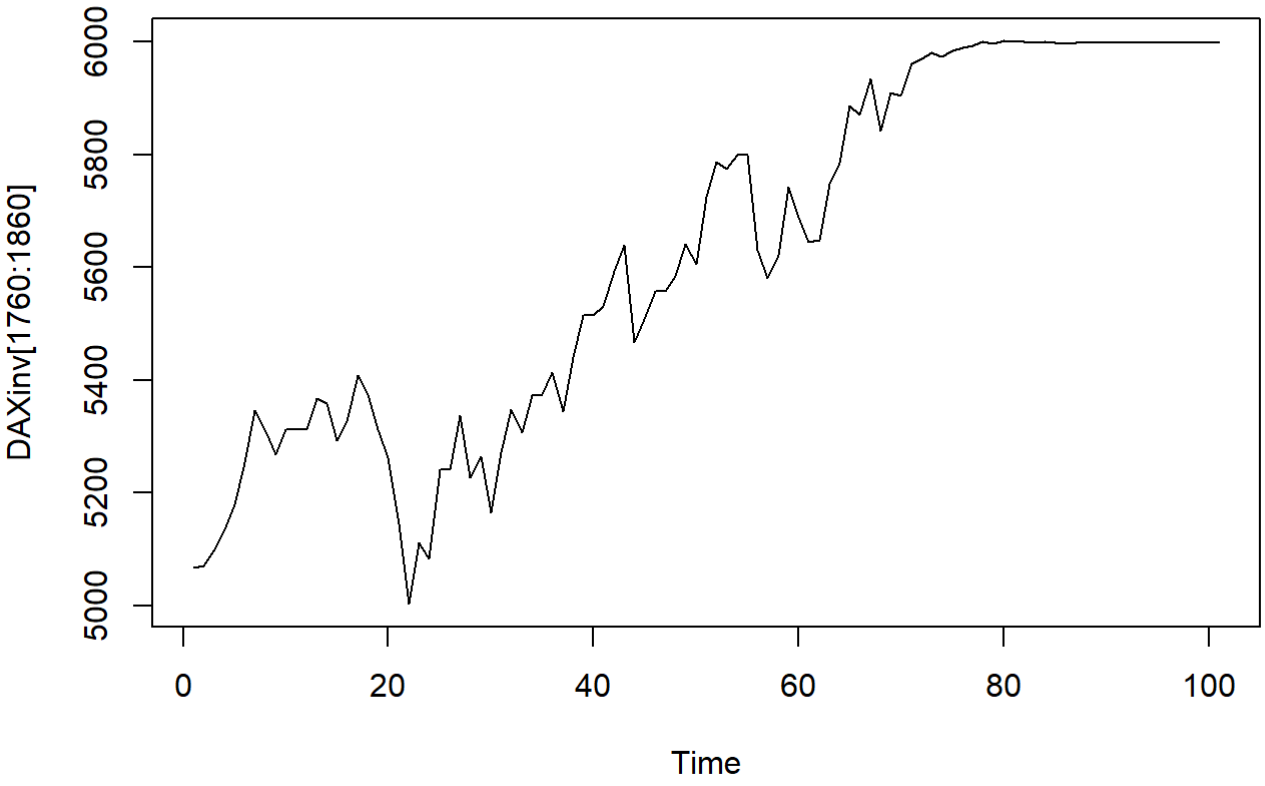
```
frequency(EuStockMarkets)
```

```
## [1] 260
```

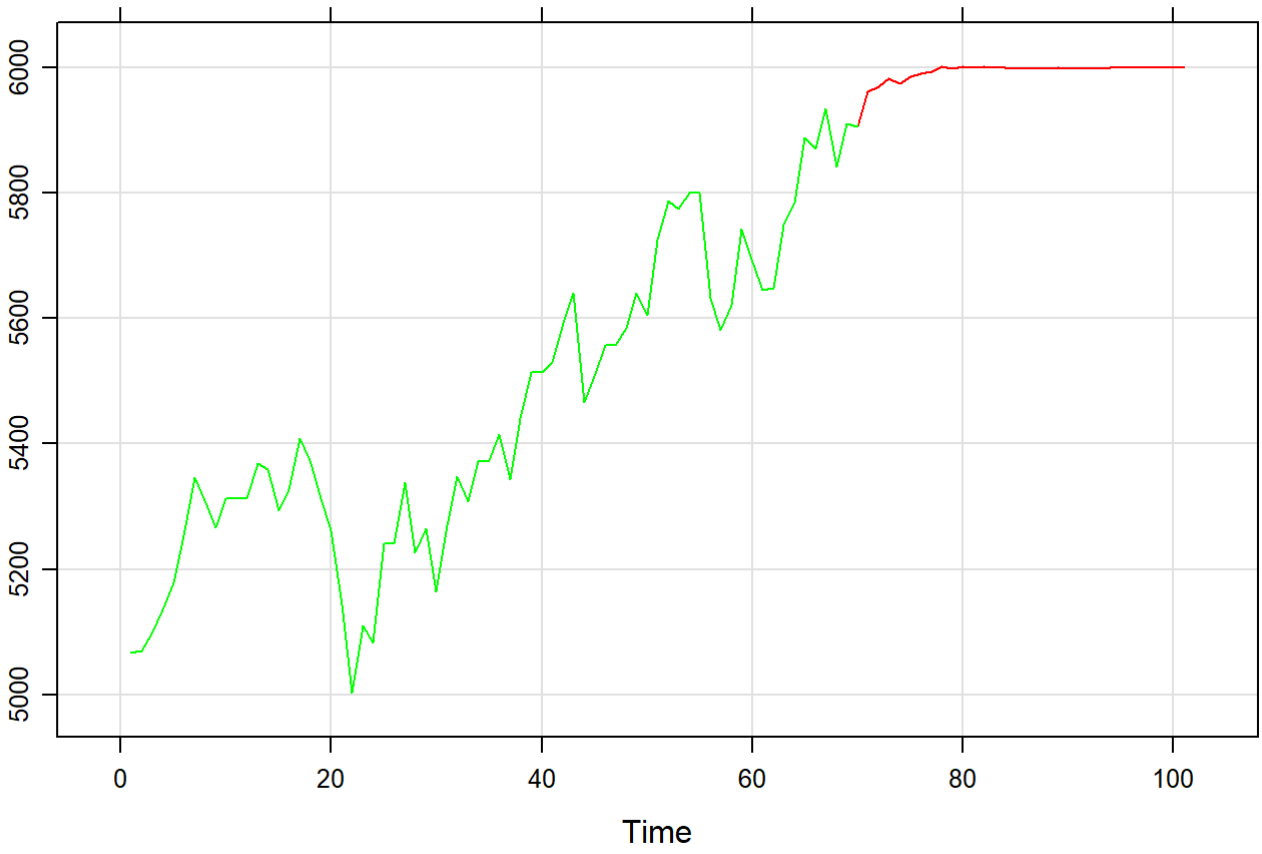
```
DAXinv =ts(c(X_train[,1], x),
           start = c(1991,130), frequency = 260)

# Dibujando todo
plot(DAXinv)
```




```
plot.ts(DAXinv[1760:1860]) # zoom
```

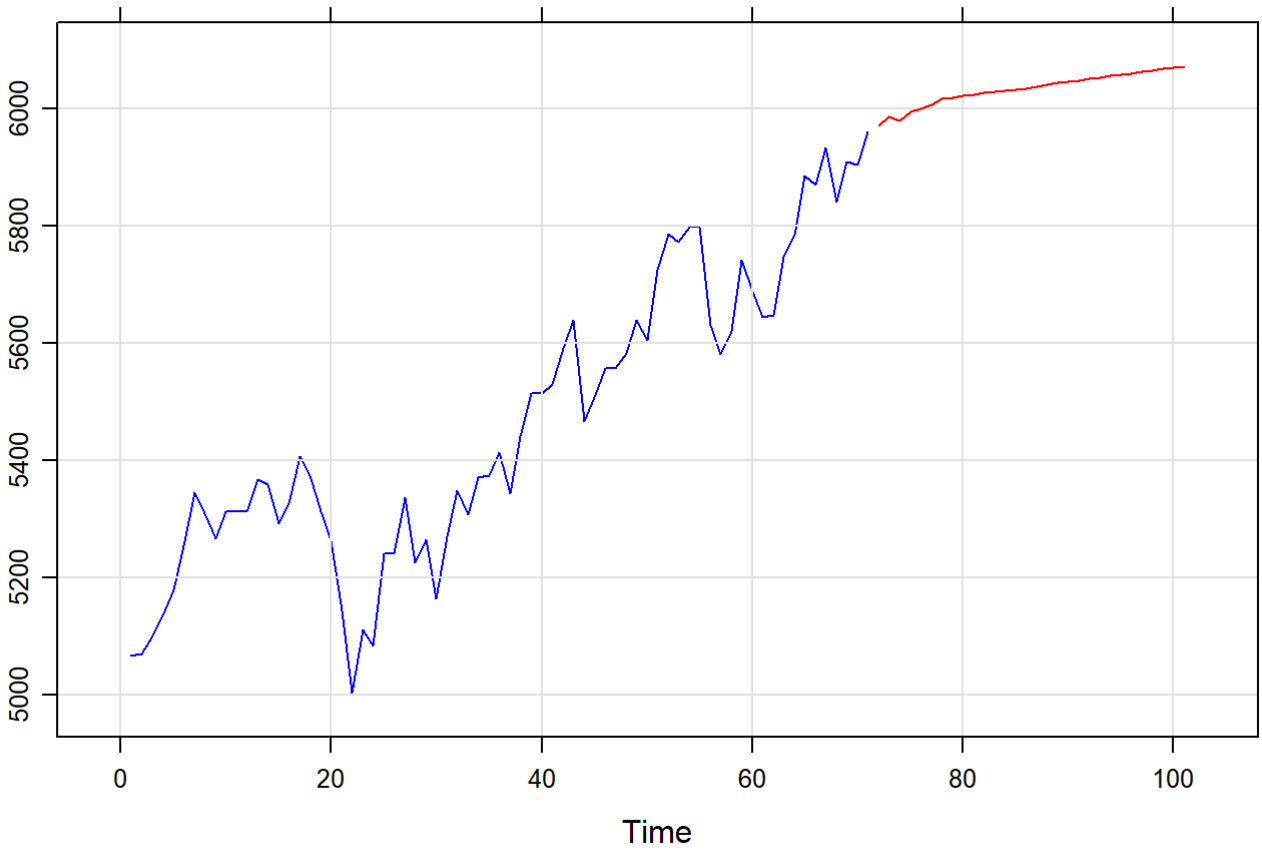



```
# Plot avanzado con separación visual entre lo real y lo pronosticado
library(lattice)
library(grid)
library(zoo)

# Objeto zoo
xx = zoo(DAXinv[1760:1860])

# En el parámetro grid.clip ponemos la cantidad de observaciones que son reales dentro de las
# que hemos elegido. Hemos cogido 101 de las que 30 son pronósticos, así que grid.clip sería
71-1

xyplot(xx, grid=TRUE, panel = function(xx, y, ...){

  panel.xyplot(xx, y, col="red", ...)

  grid.clip(unit(70, "native"), just=c("right"))

  panel.xyplot(xx, y, col="green", ...) })
```




```
# Como vemos si nos vamos demasiado lejos en el futuro se aplana la predicción
```

```r
library(MTS)

# Diferenciando la serie de tiempo multivariante
stnry = diffM(X_train)

# Ajustar un modelo VAR a la serie diferenciada
var_model <- VAR(stnry, p = 10) # Ajusta 'p' según los criterios de selección de modelo

# Generar pronósticos usando el modelo VAR
fcast <- predict(var_model, n.ahead = n_obs)

# Extraer los valores pronosticados para el índice DAX y revertir la diferenciación
DAX_forecasted <- cumsum(fcast$fcst$DAX[, "fcst"]) + tail(X_train[, "DAX"], 1)

# Combinar los valores reales del DAX y los pronosticados en un objeto de serie de tiempo
DAX_combined <- ts(c(X_train[, "DAX"], DAX_forecasted), start = start(EuStockMarkets), freque
ncy = frequency(EuStockMarkets))

# Crear un objeto zoo para la serie combinada
xx <- zoo(DAX_combined[1760:1860])

# Graficar los valores reales y pronosticados con una separación visual
xyplot(xx, grid = TRUE, panel = function(x, y, ...) {
    # Graficar los valores reales en azul
    panel.xyplot(x[1:(length(x) - n_obs)], y[1:(length(y) - n_obs)], col = "blue", ...)

    # Graficar los valores pronosticados en rojo
    panel.xyplot(x[(length(x) - n_obs + 1):length(x)], y[(length(y) - n_obs + 1):length(y)],
col = "red", ...)
})
```


6000
5800
5600
5400
5200
5000
0
20
40
60
80
100
Time