Assignment 2

Ignacio Amaya

19 de diciembre de 2015

```r
library(TSA)
```

```
## Warning: package 'TSA' was built under R version 3.2.2

## Loading required package: leaps

## Warning: package 'leaps' was built under R version 3.2.2

## Loading required package: locfit

## Warning: package 'locfit' was built under R version 3.2.2

## locfit 1.5-9.1    2013-03-22
## Loading required package: mgcv
## Loading required package: nlme
## This is mgcv 1.8-6. For overview type 'help("mgcv-package")'.
## Loading required package: tseries

## Warning: package 'tseries' was built under R version 3.2.2

##
## Attaching package: 'TSA'
##
## The following objects are masked from 'package:stats':
##
##     acf, arima
##
## The following object is masked from 'package:utils':
##
##     tar
```

```r
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 3.2.2

## Loading required package: zoo

## Warning: package 'zoo' was built under R version 3.2.2

##
## Attaching package: 'zoo'
##
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
##
## Loading required package: timeDate
```

```
## Warning: package 'timeDate' was built under R version 3.2.2
```

```
##
## Attaching package: 'timeDate'
##
## The following objects are masked from 'package:TSA':
##
##     kurtosis, skewness
##
## This is forecast 6.1
##
##
## Attaching package: 'forecast'
##
## The following objects are masked from 'package:TSA':
##
##     fitted.Arima, plot.Arima
##
## The following object is masked from 'package:nlme':
##
##     getResponse
```

```r
library(astsa)
```

```
## Warning: package 'astsa' was built under R version 3.2.2
```

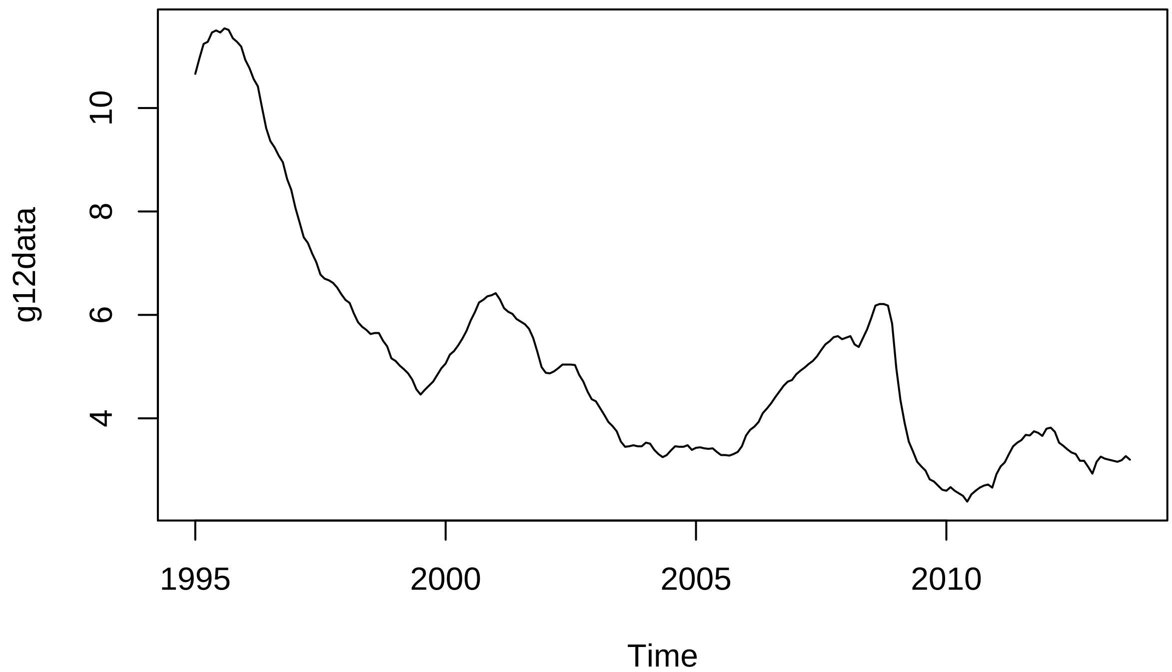
```
##
## Attaching package: 'astsa'
##
## The following object is masked from 'package:forecast':
##
##     gas
```

```r
dataAsigment2 <- read.csv("~/MIS DOCUMENTOS/DATA SCIENCE MASTER (EIT DIGITAL)/Intelligent Data Analysis
```

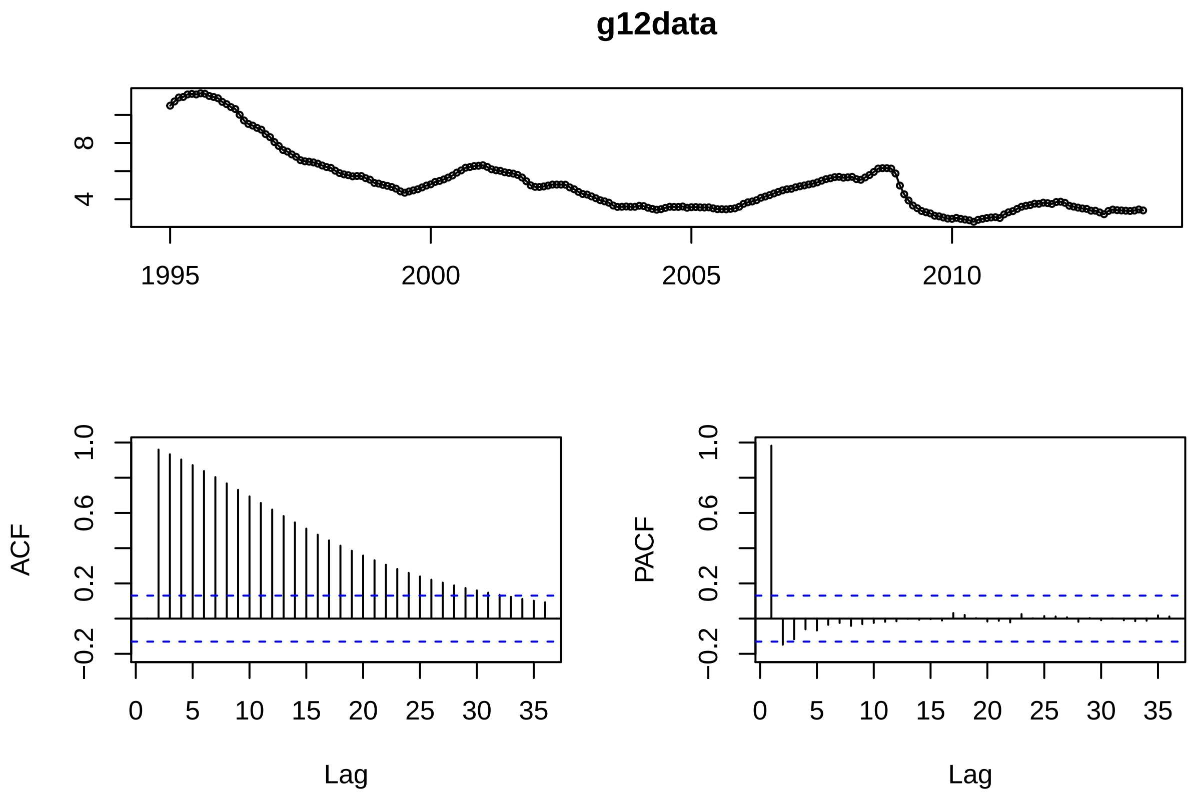
```r
g12data=ts(dataAsigment2$Tipo,start=c(1995,1), end=c(2013,9),frequency=12)
str(g12data)
```

```
##  Time-Series [1:225] from 1995 to 2014: 10.7 11 11.2 11.3 11.5 ...
```

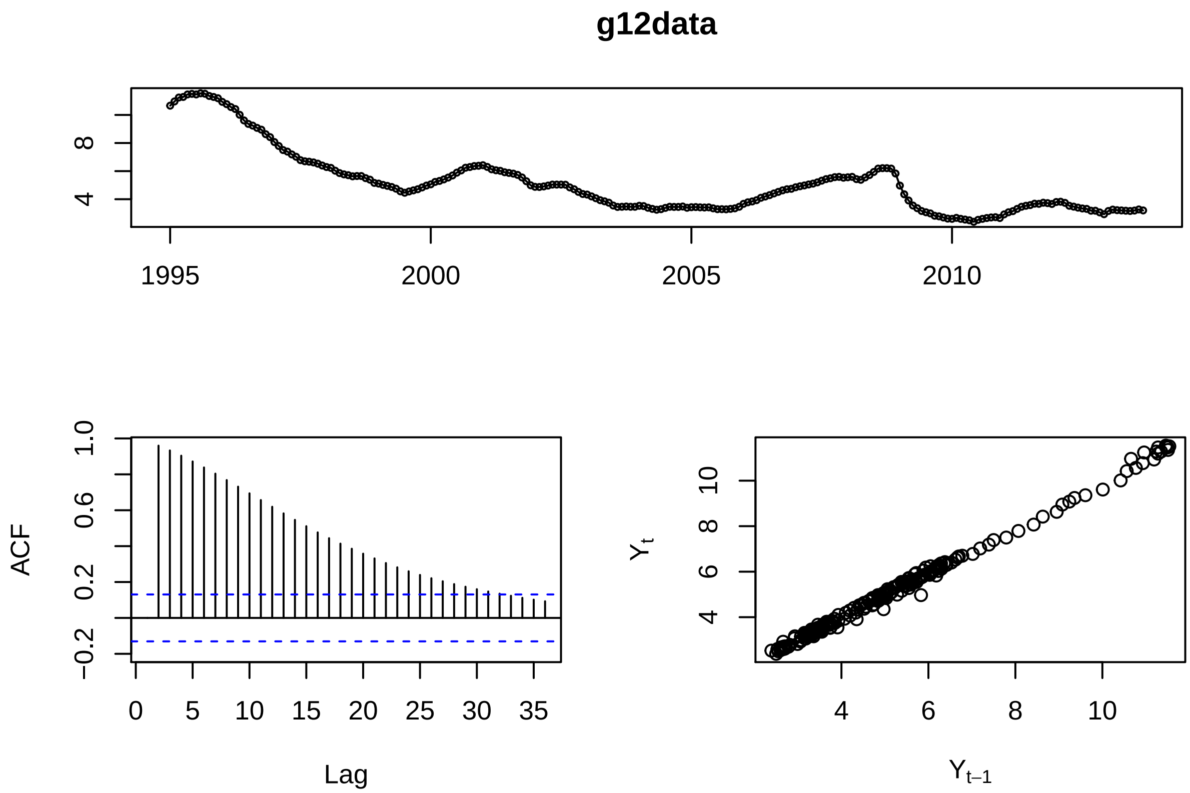
```r
plot(g12data)
```




```r
tsdisplay(g12data, plot.type="partial")
```




```r
tsdisplay(g12data, plot.type="scatter") #The one Arminda likes most
```

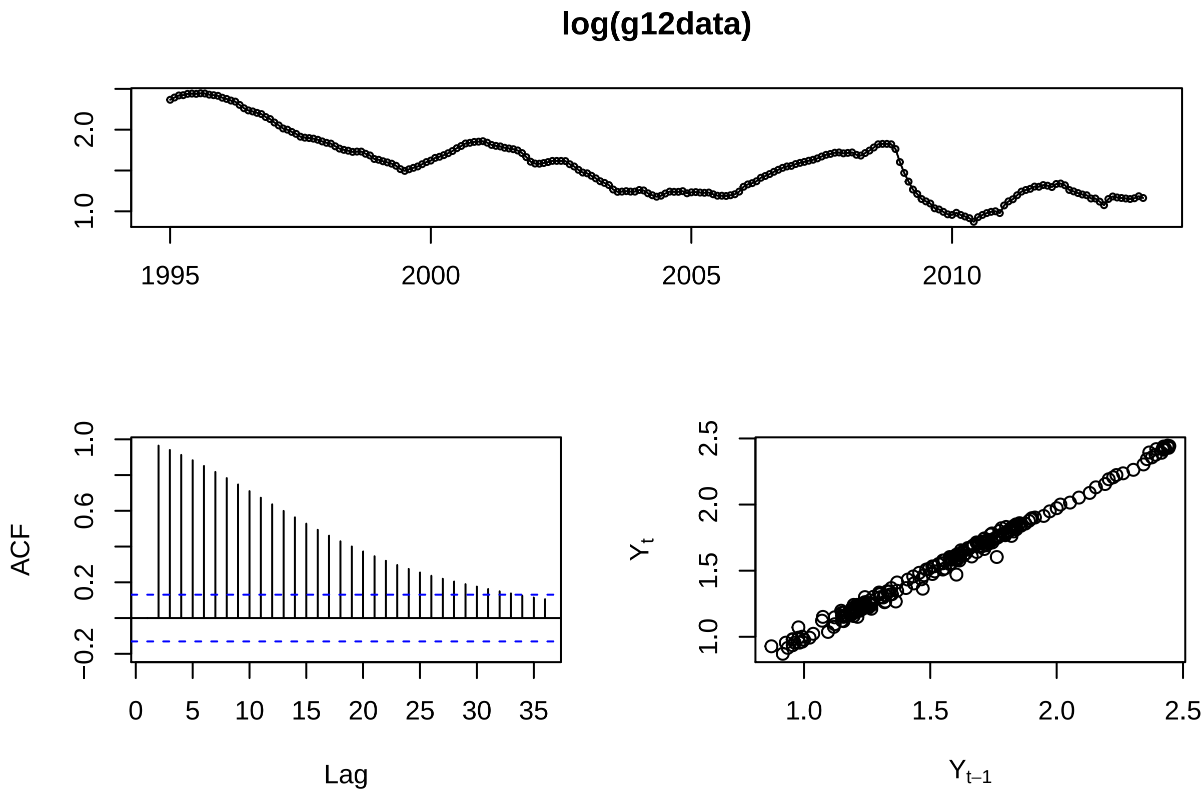


Let's apply BoxCox transformations to check if it is possible to apply a logaritmic transformation to our time series.

```
BoxCox.lambda(g12data,lower=0, upper=2)
```

```
## [1] 0.6487093
```

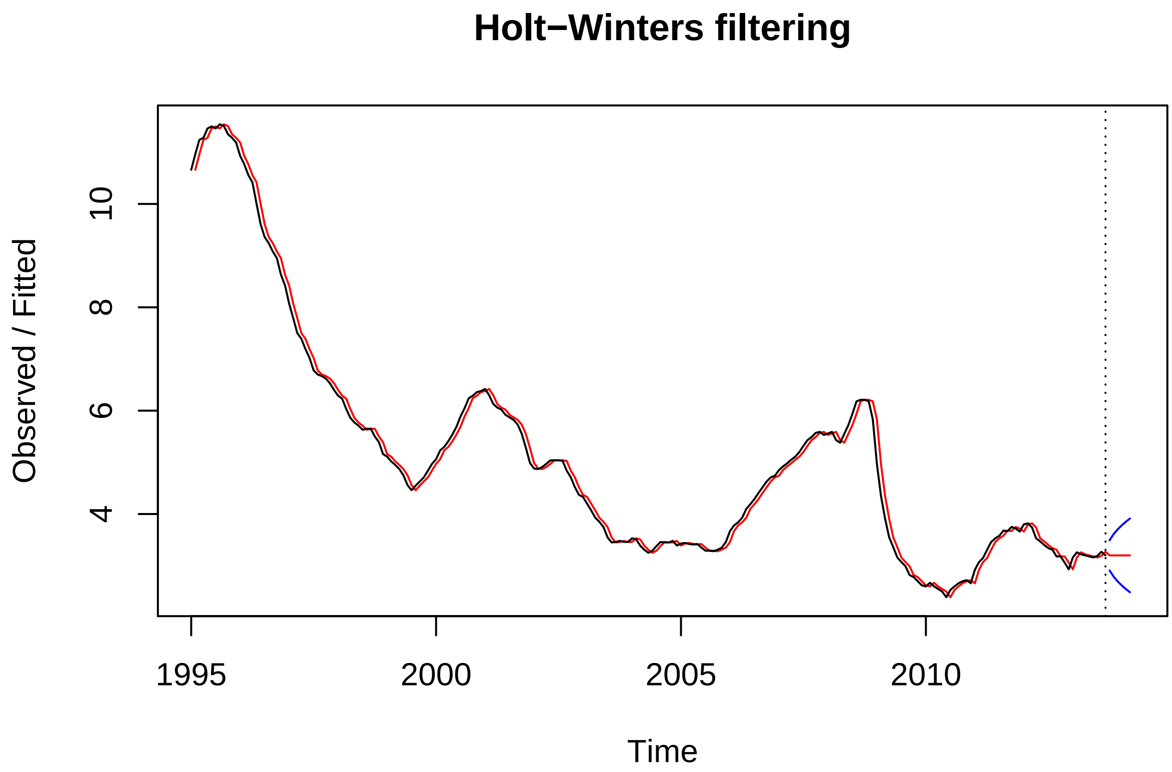
```
tsdisplay(log(g12data), plot.type="scatter")
```



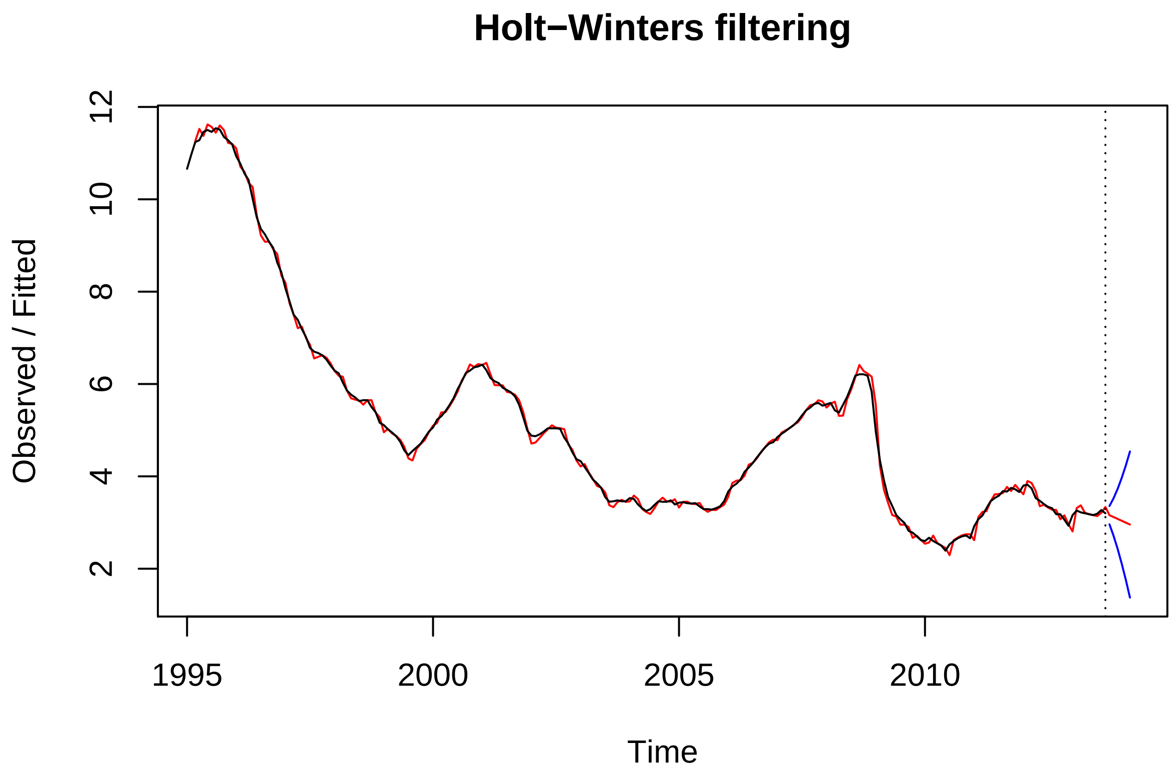
As it is not close to zero we can't apply this transformation. There is a slight improvement because we can see in the third plot that the values are more spread across the plot.

We can observe that we have a linear trend (decreasing values over time) and a seasonal component as after some years decreasing the time series goes up again (and that happens several times).

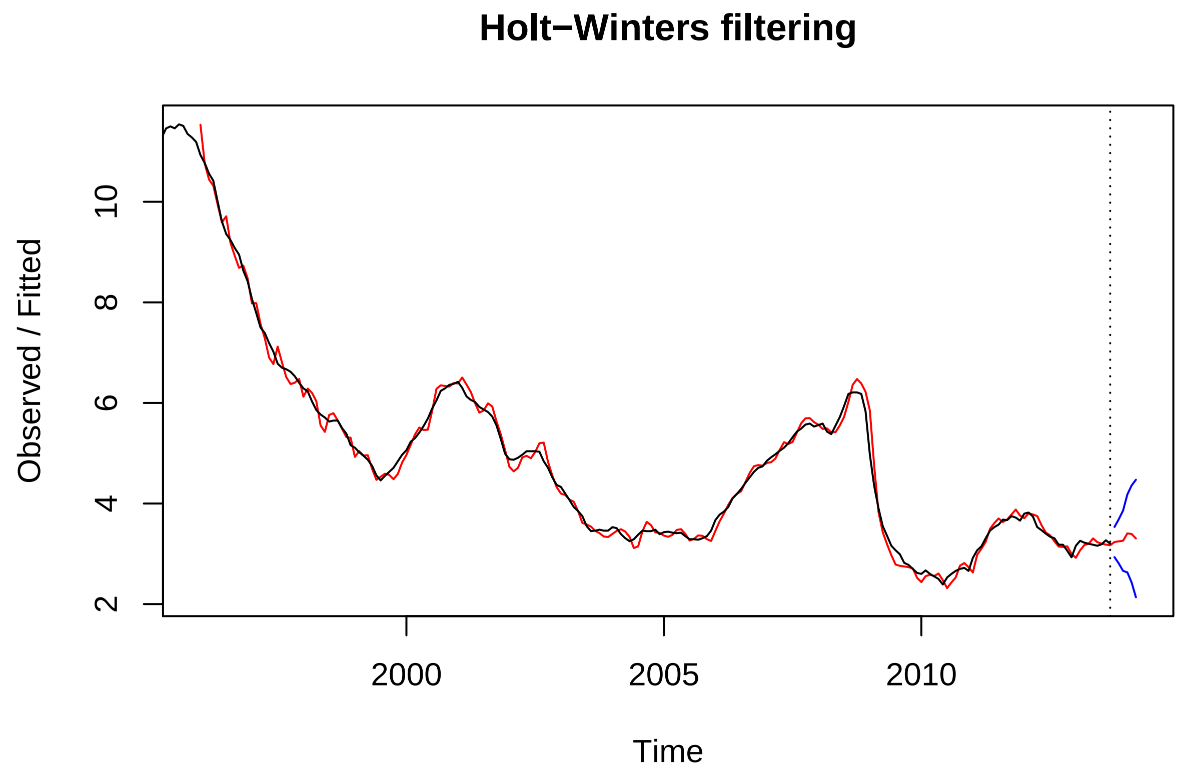
```
## Holt-Winters smoothing with series with linear trend and seasonal variation

smoothing1=HoltWinters(g12data,gamma=FALSE,beta=FALSE)
smoothing2=HoltWinters(g12data,gamma=FALSE)
smoothing3=HoltWinters(g12data)
smoothing4=HoltWinters(g12data,seasonal = "mult")

forecast1=predict(smoothing1, n.ahead=6, prediction.interval=TRUE, level=0.95)
forecast2=predict(smoothing2, n.ahead=6, prediction.interval=TRUE, level=0.95)
forecast3=predict(smoothing3, n.ahead=6, prediction.interval=TRUE, level=0.95)
forecast4=predict(smoothing4, n.ahead=6, prediction.interval=TRUE,level=0.95)

plot(smoothing1, forecast1)
```



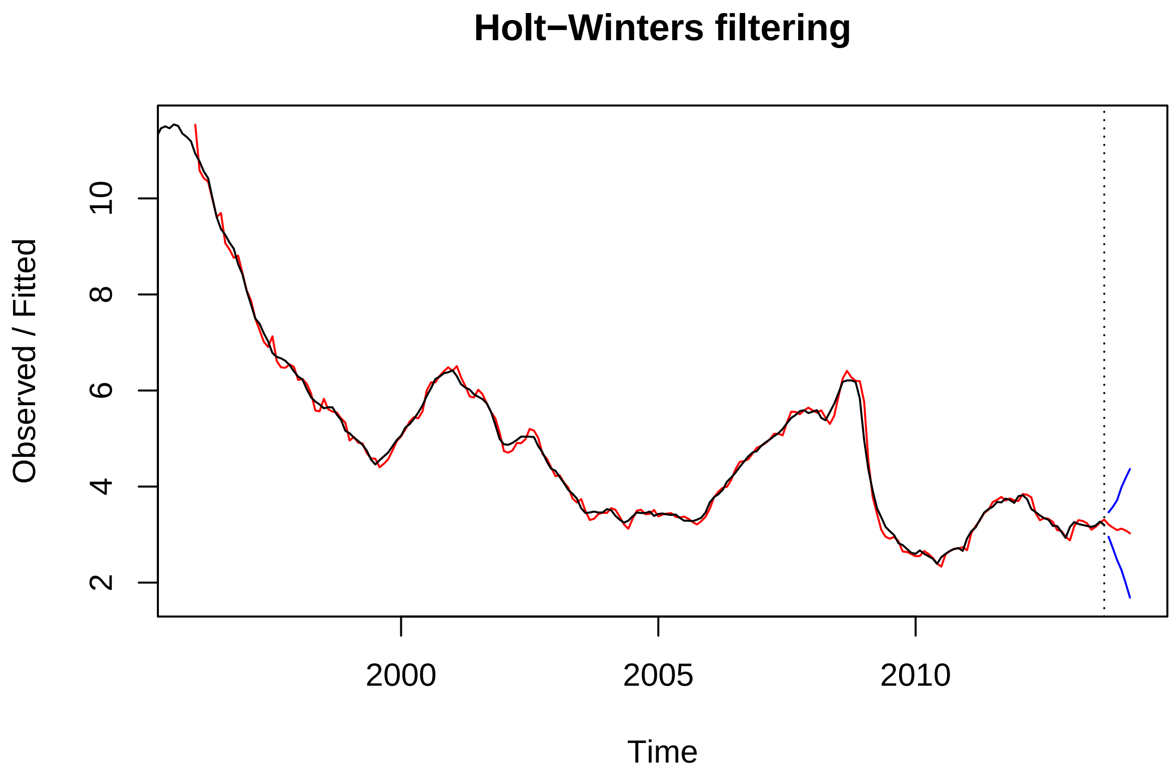

```r
plot(smoothing2, forecast2)
```




```
plot(smoothing3, forecast3)
```

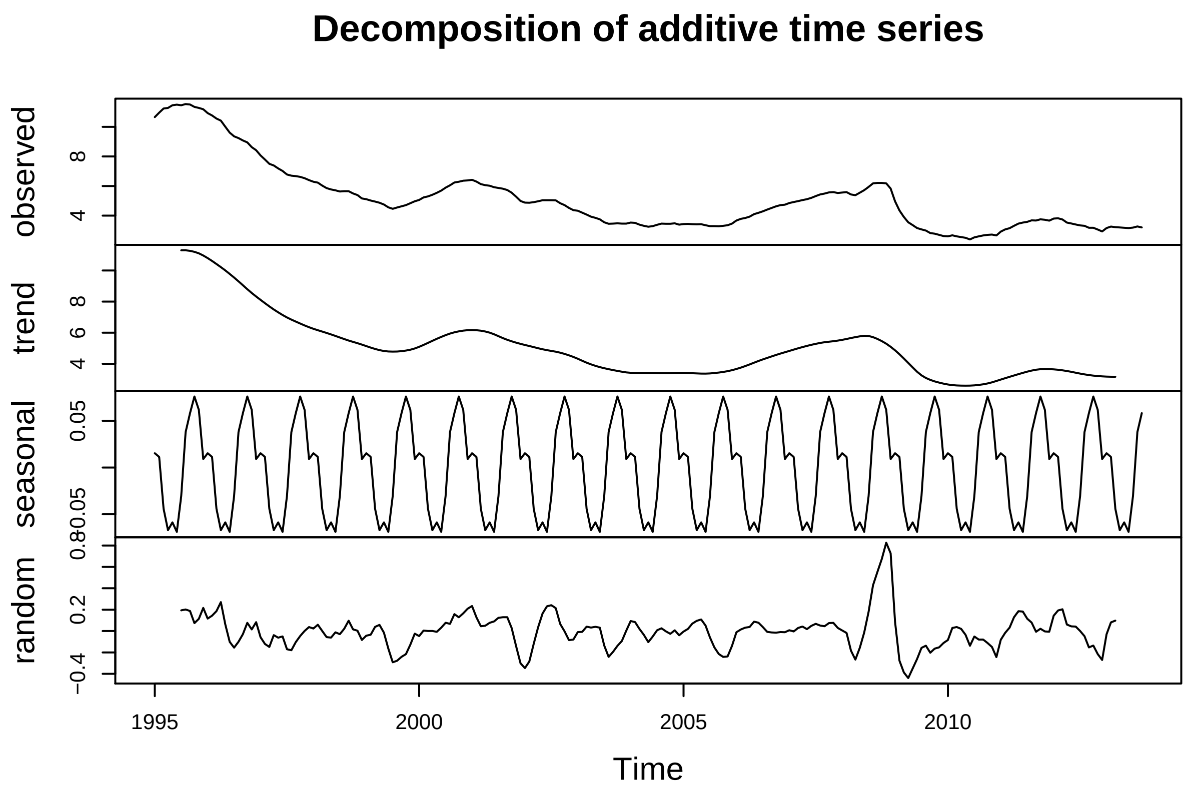



```r
plot(smoothing4, forecast4)
```

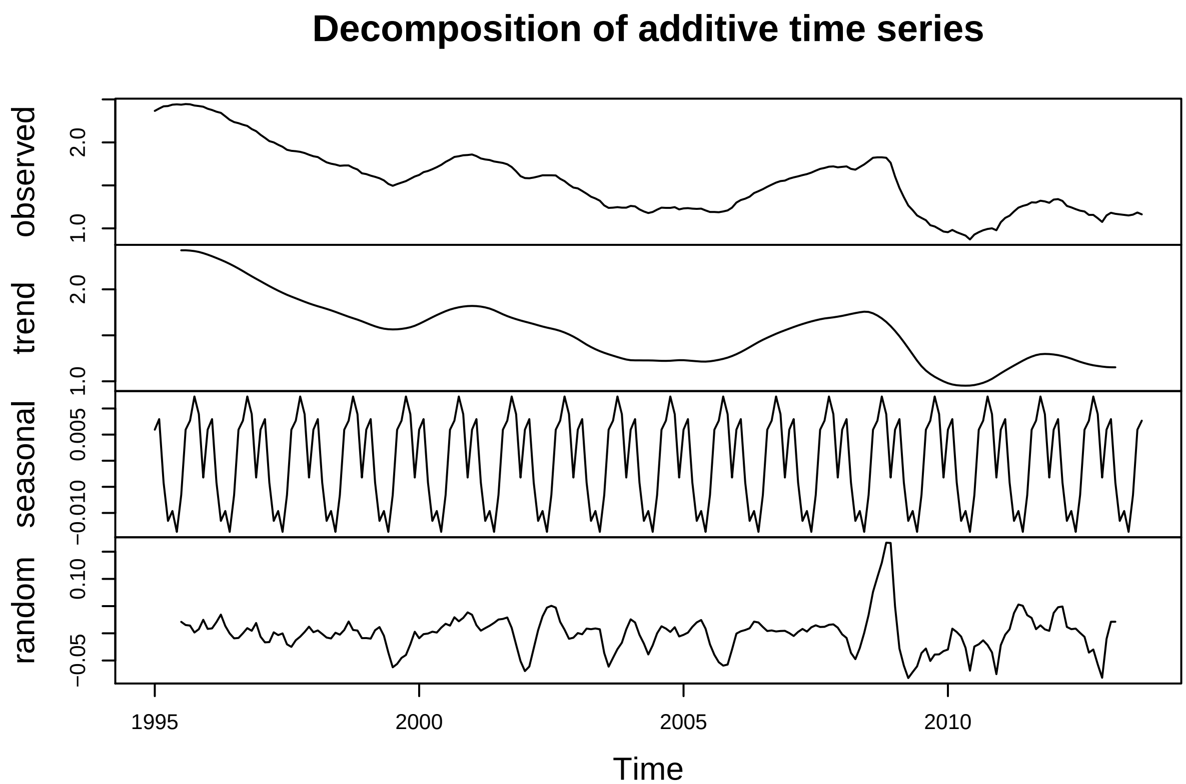


Let's try to decompose this time series.

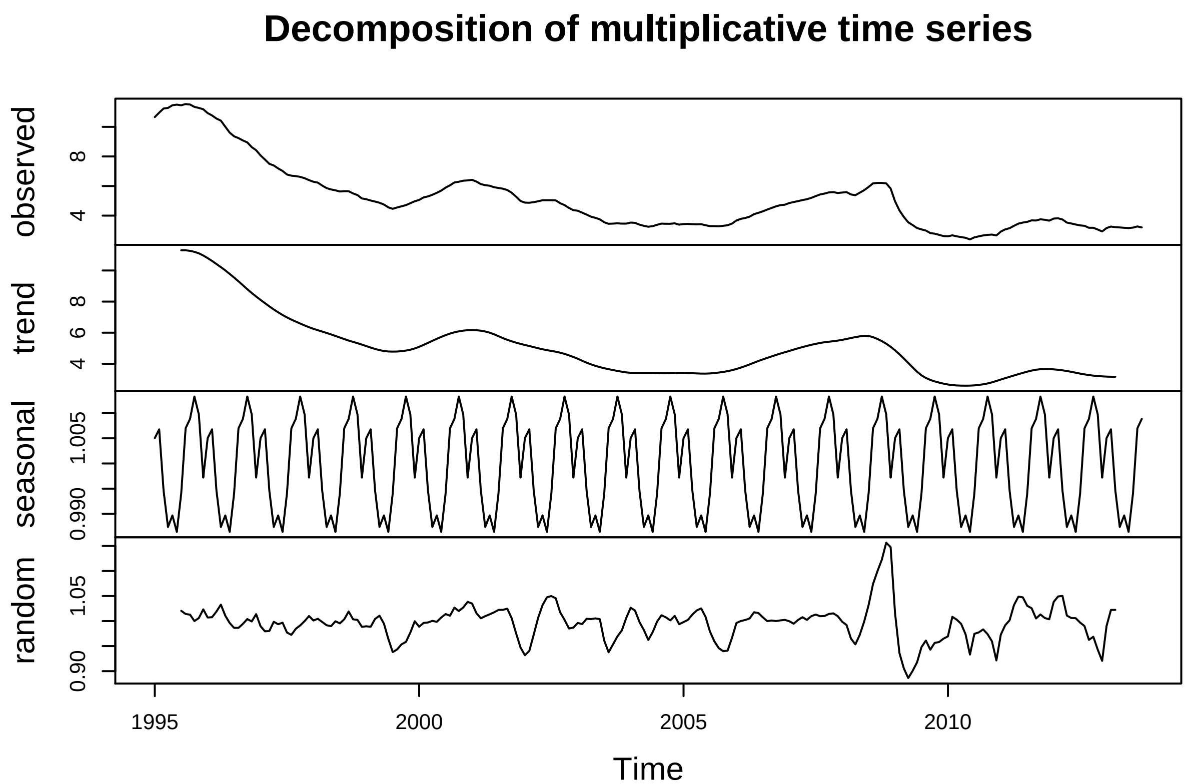
```
decomp=decompose(g12data)
plot(decomp)
```



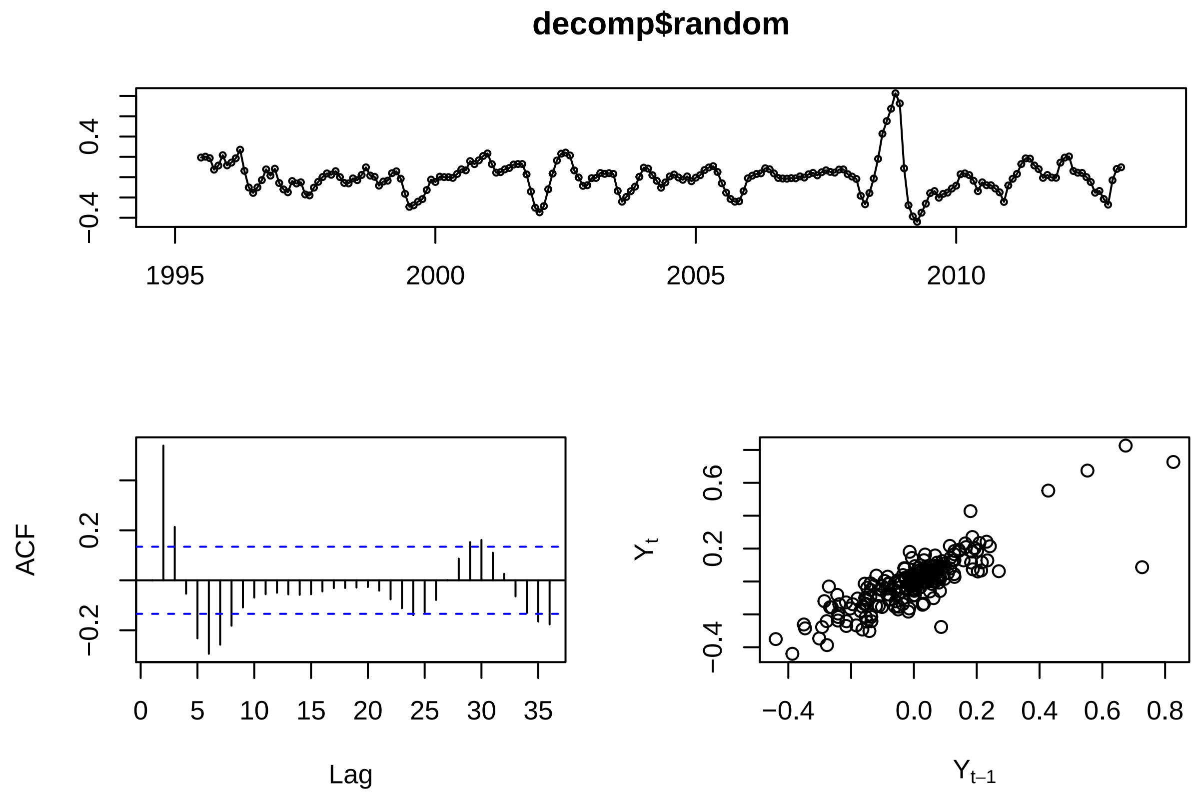

```
decompLog=decompose(log(g12data))
plot(decompLog)
```



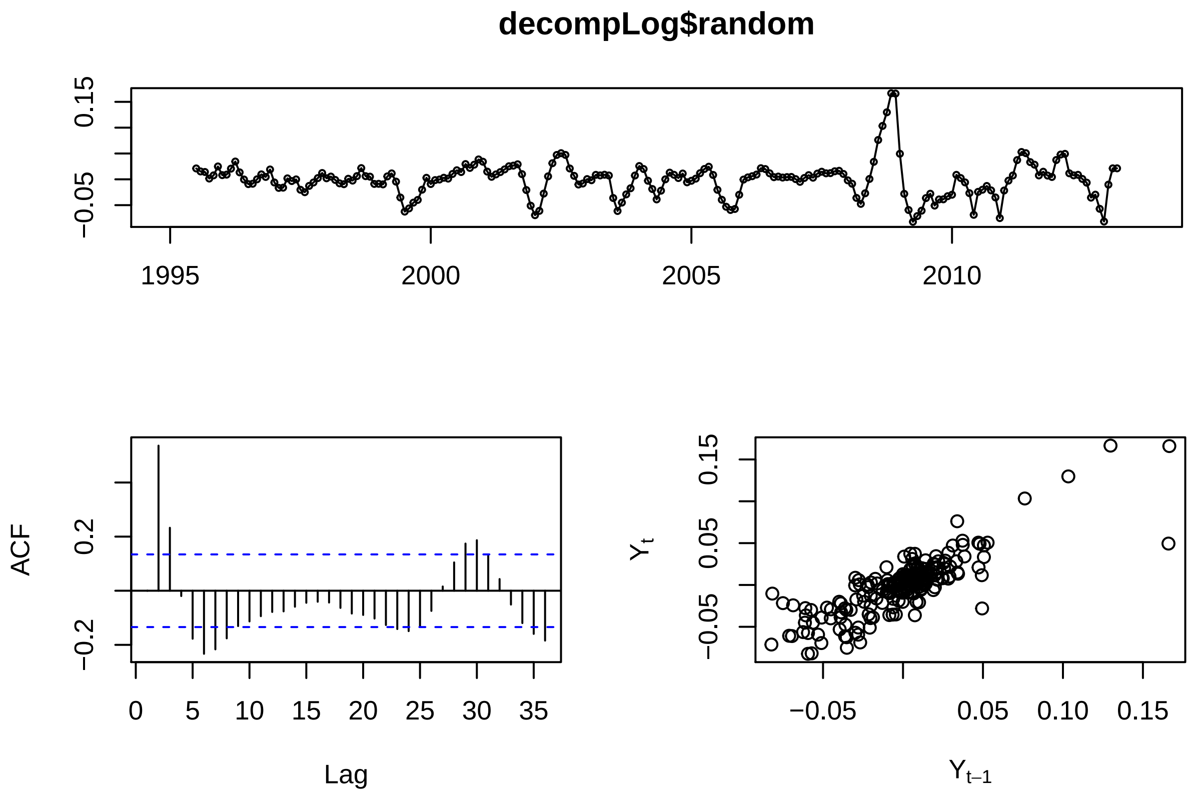

```
### Decomposition of time series, multiplicative decomposition
decompMult=decompose(g12data,type="multiplicative")
plot(decompMult)
```



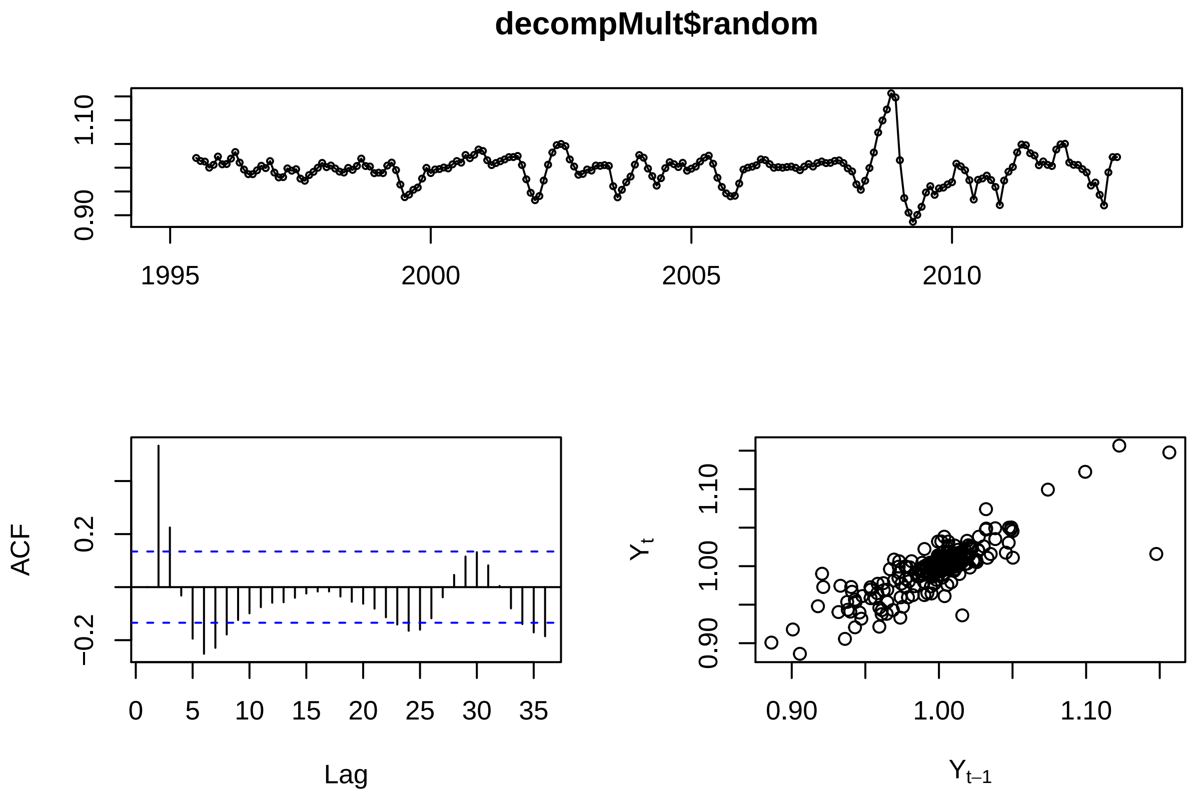

```r
tsdisplay(decomp$random,plot.type="scatter")
```



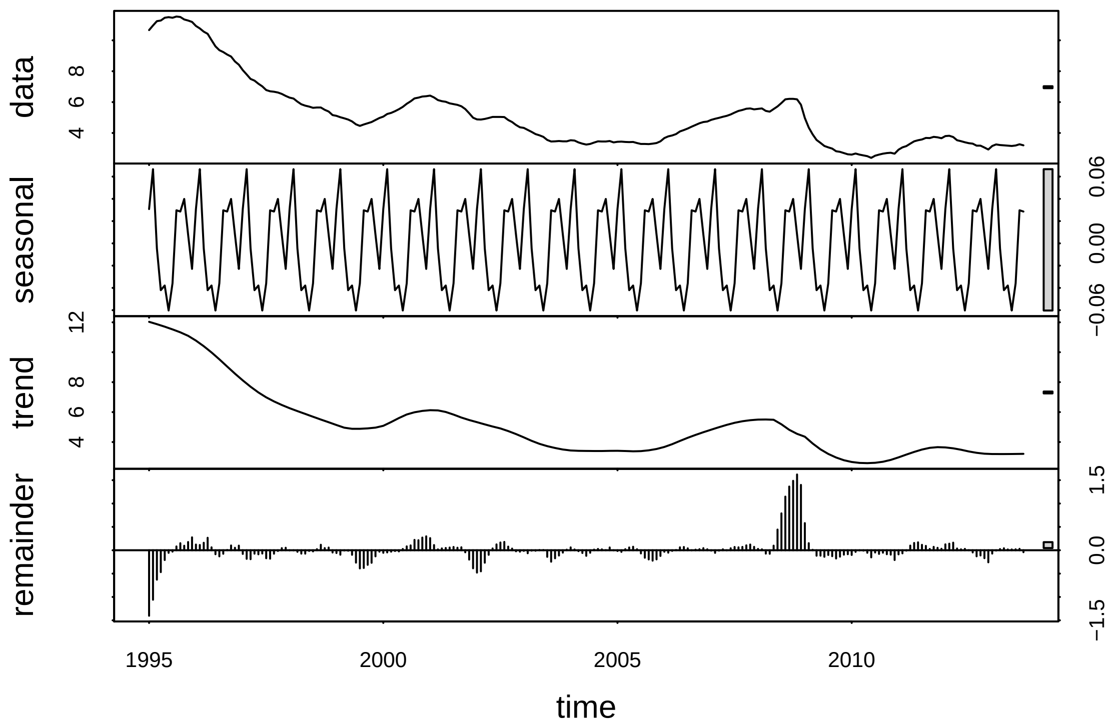

```r
tsdisplay(decompLog$random,plot.type="scatter")
```



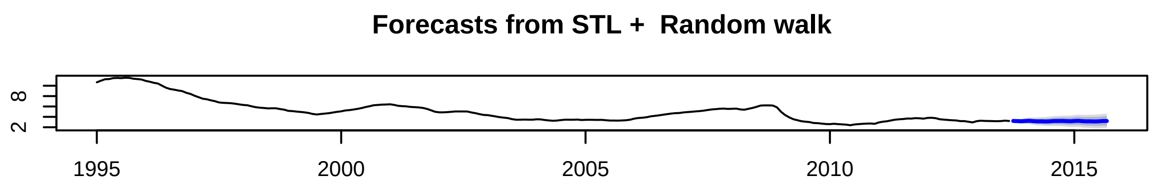

```
tsdisplay(decompMult$random,plot.type="scatter")
```

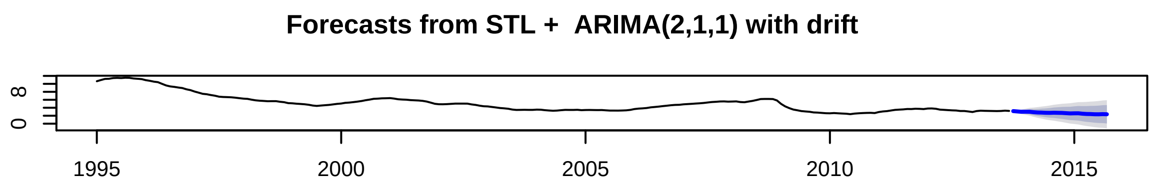


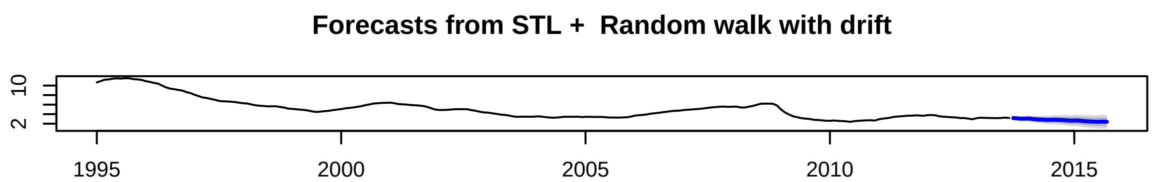

```
stl=stl(g12data, s.window="periodic", robust=TRUE)
plot(stl)
```




```
par(mfrow=c(3,1))
fcst1=forecast(stl, method="naive")
plot(fcst1)
fcst2=forecast(stl, method="arima")
plot(fcst2)
fcst3=forecast(stl, method="rwdrift")
plot(fcst3)
```



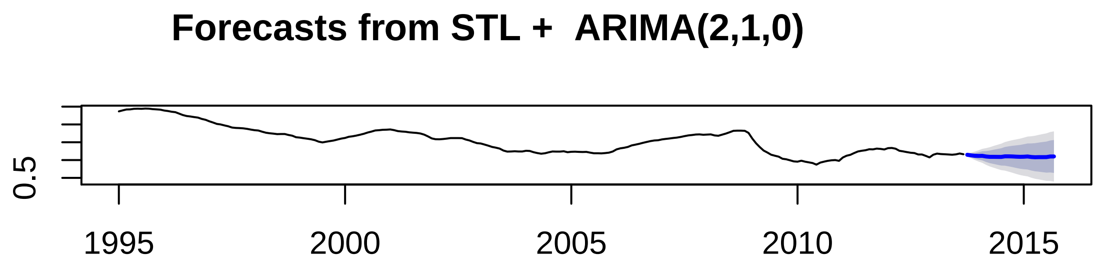


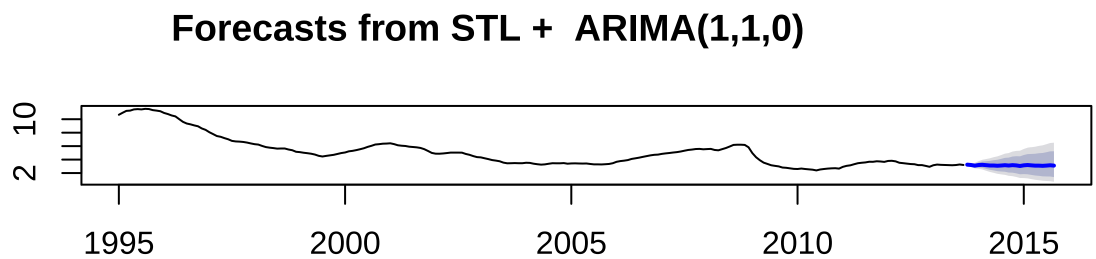



```
fcst3$mean
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul
## 2013
## 2014 3.069032 3.071608 2.967108 2.896285 2.867203 2.811453 2.802564
## 2015 2.669507 2.672084 2.567583 2.496760 2.467678 2.411929 2.403039
##           Aug      Sep      Oct      Nov      Dec
## 2013                   3.178048 3.112710 3.048582
## 2014 2.834928 2.800475 2.778523 2.713185 2.649057
## 2015 2.435404 2.400951
```

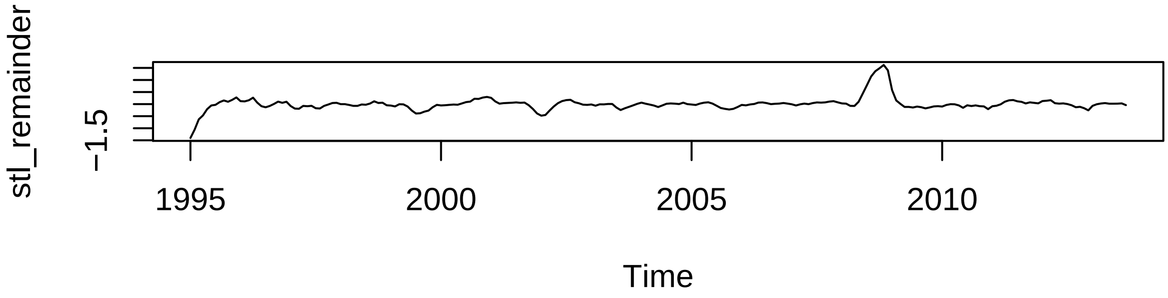
```
par(mfrow=c(2,1))
## in the logarithmic scale (log(a.ts))
lstl=stl(log(g12data), s.window="periodic", robust=TRUE)
fcst4=forecast(lstl, method="arima")
plot(fcst4)
## in the original scale
fcst5=stlf(g12data,method="arima",lambda=BoxCox.lambda(g12data))
plot(fcst5)
```

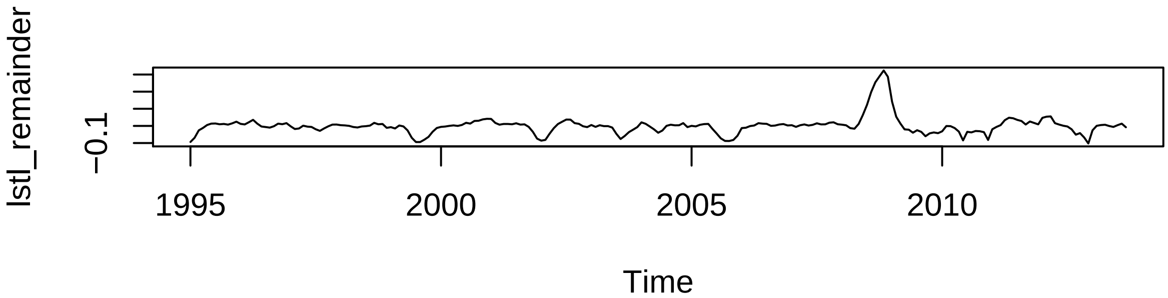




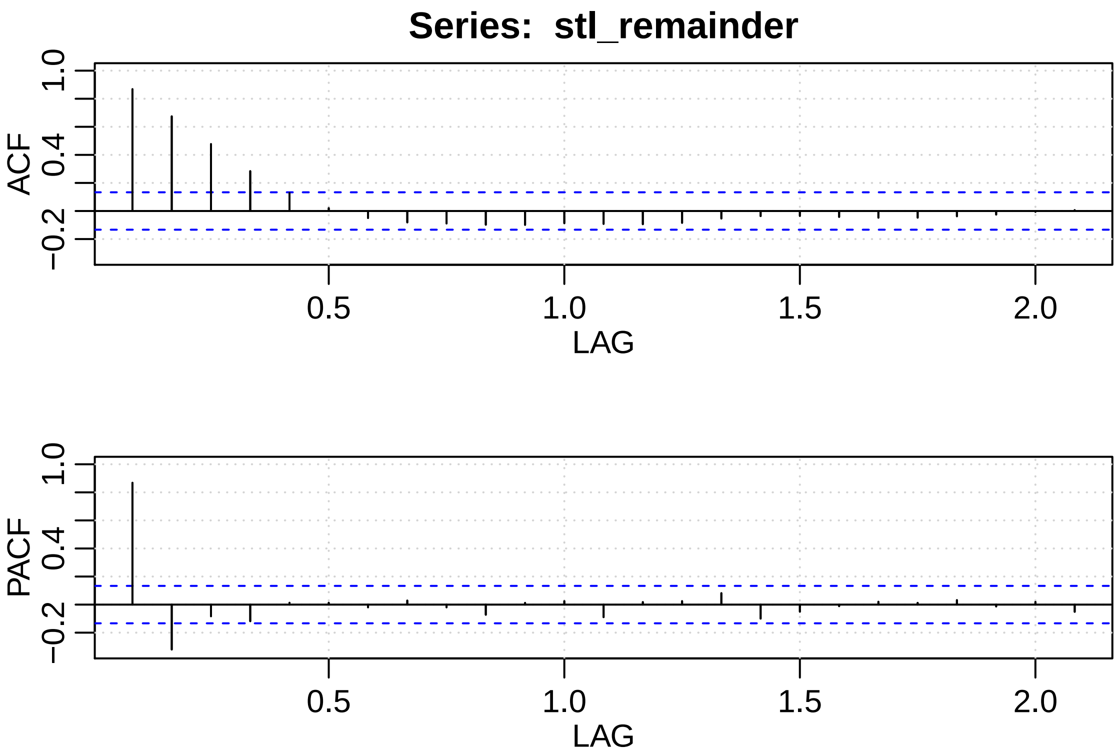
No parece que haya una mejora apreciable al realizar la transformación logarítmica.

```
par(mfrow=c(2,1))
stl_remainder=stl$time.series[,3]
lstl_remainder=lstl$time.series[,3]
plot(stl_remainder)
plot(lstl_remainder)
```



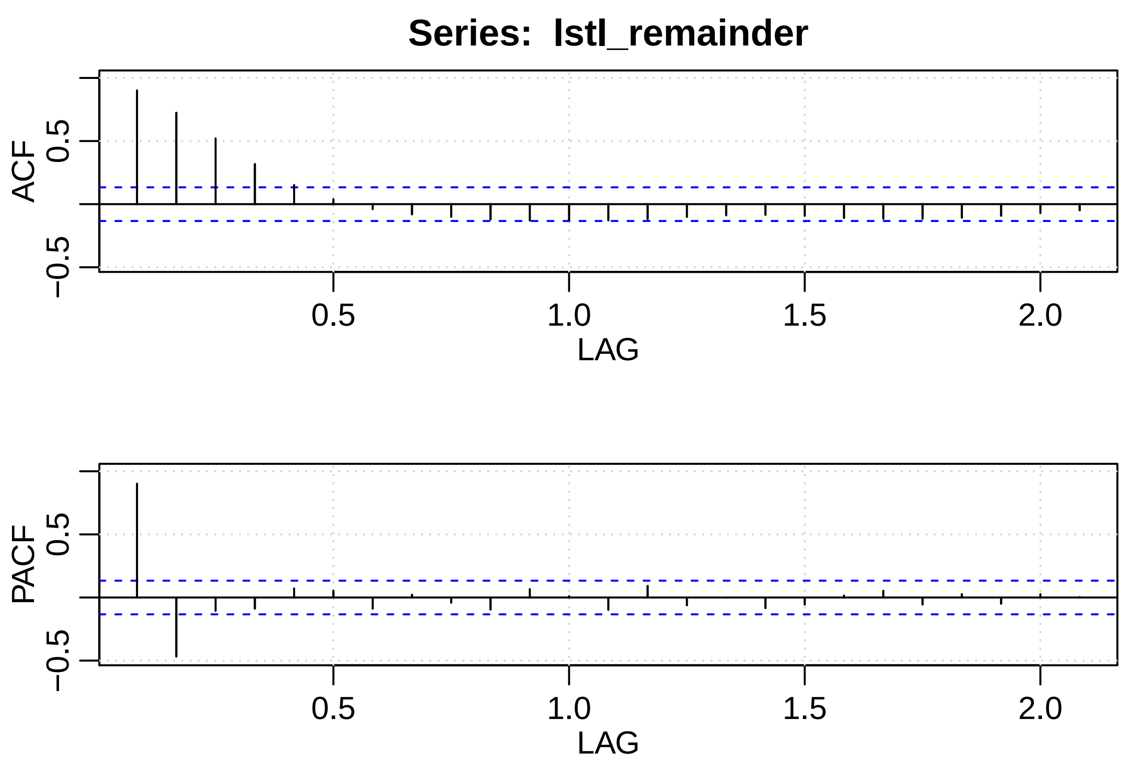



```
acf2(stl_remainder)
```




```
##        ACF  PACF
##  [1,]  0.87  0.87
##  [2,]  0.67 -0.32
##  [3,]  0.48 -0.08
##  [4,]  0.28 -0.12
##  [5,]  0.13  0.01
##  [6,]  0.02  0.01
##  [7,] -0.05 -0.02
##  [8,] -0.08  0.03
##  [9,] -0.09 -0.02
## [10,] -0.10 -0.07
## [11,] -0.10  0.01
## [12,] -0.09  0.02
## [13,] -0.09 -0.09
## [14,] -0.09  0.02
## [15,] -0.08  0.02
## [16,] -0.05  0.08
## [17,] -0.04 -0.10
## [18,] -0.04 -0.05
## [19,] -0.04 -0.01
## [20,] -0.05  0.02
## [21,] -0.05  0.01
## [22,] -0.04  0.03
## [23,] -0.03 -0.01
## [24,] -0.01  0.02
## [25,]  0.01 -0.05
```

```r
acf2(lstl_remainder)
```

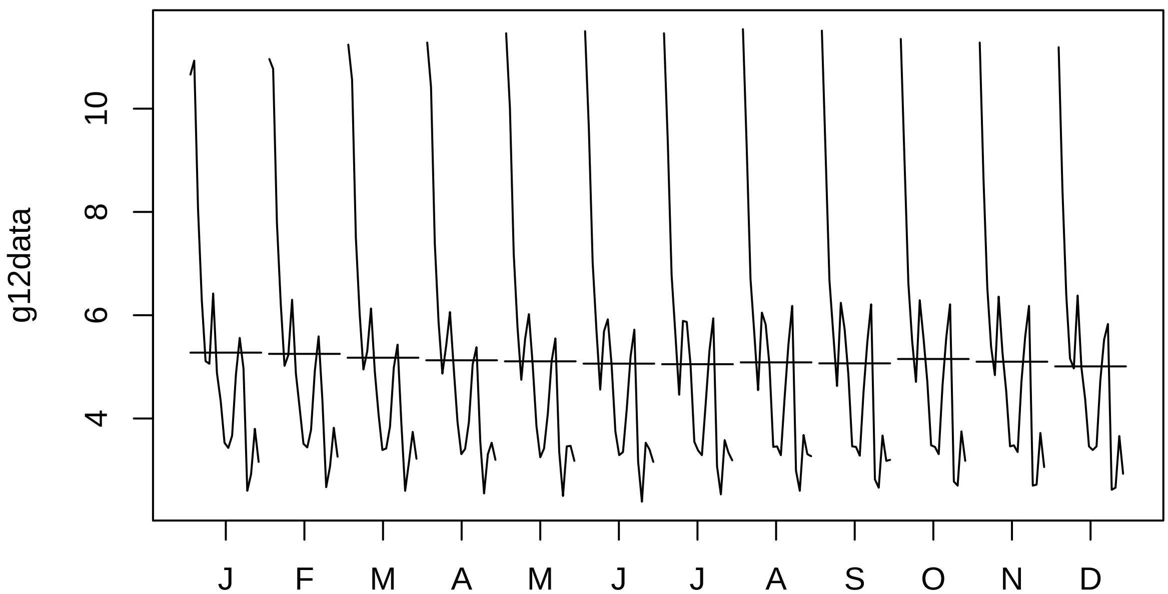



```
##         ACF  PACF
##  [1,]  0.90  0.90
##  [2,]  0.72 -0.47
##  [3,]  0.52 -0.11
##  [4,]  0.32 -0.09
##  [5,]  0.15  0.07
##  [6,]  0.04  0.05
##  [7,] -0.04 -0.09
##  [8,] -0.08  0.02
##  [9,] -0.10 -0.04
## [10,] -0.12 -0.10
## [11,] -0.13  0.06
## [12,] -0.13  0.01
## [13,] -0.13 -0.10
## [14,] -0.12  0.09
## [15,] -0.10 -0.06
## [16,] -0.09  0.00
## [17,] -0.08 -0.08
## [18,] -0.09 -0.06
## [19,] -0.11  0.01
## [20,] -0.11  0.05
## [21,] -0.12 -0.06
## [22,] -0.11  0.03
```

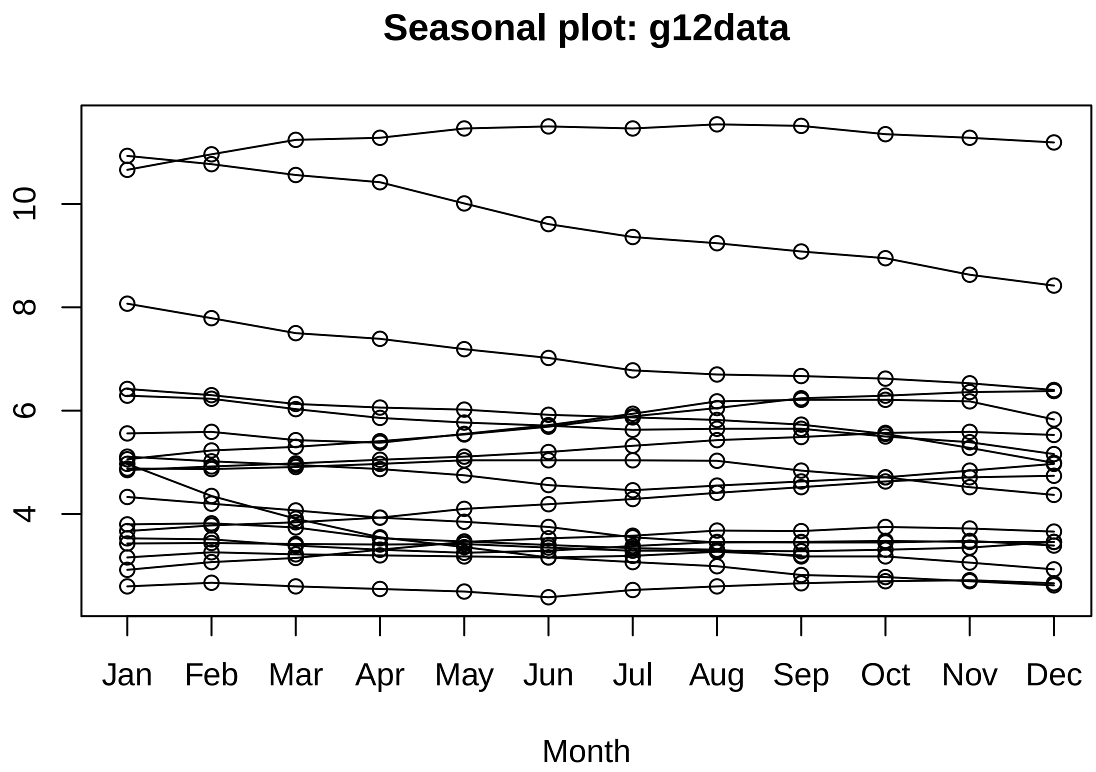
```
## [23,] -0.09 -0.05
## [24,] -0.07  0.03
## [25,] -0.05  0.00
```

There's not a clear seasonality pattern in the season plot. But in each season trough the years (monthplot) some patterns can be found

```r
par(mfrow=c(1,1))
monthplot(g12data)
```

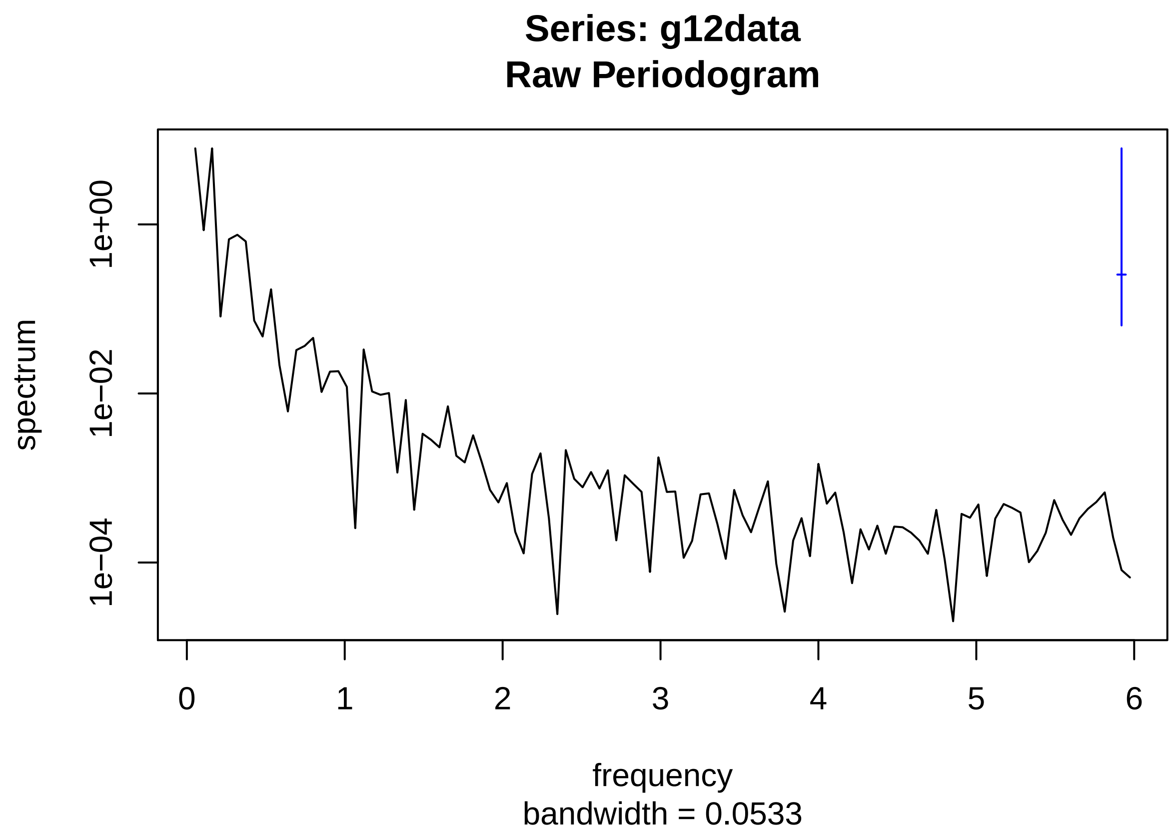



```r
seasonplot(g12data)
```



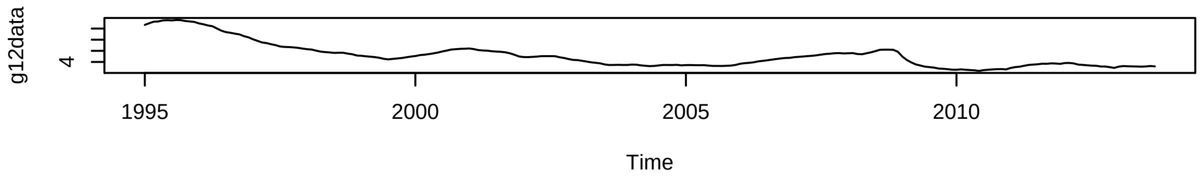
It may help us in determining the period of the series, if any. No idea!!!

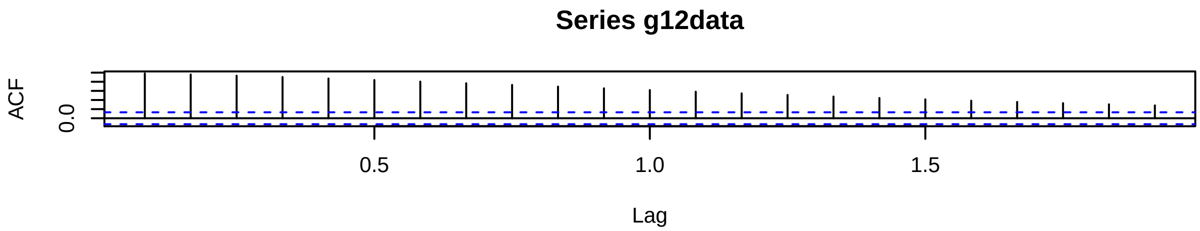
```
mvspec(g12data)
```

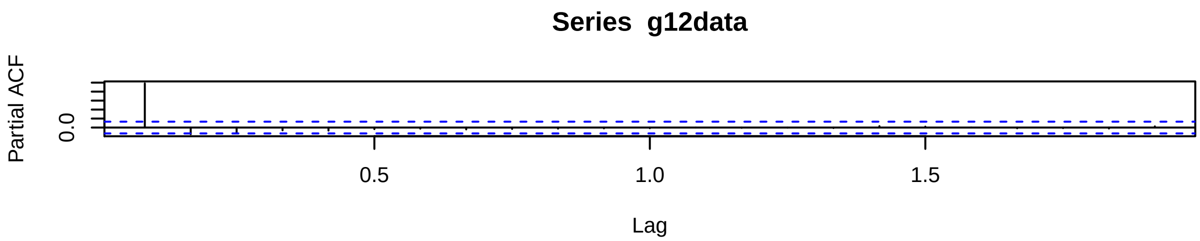


Exploring the autocorrelation structure of time series

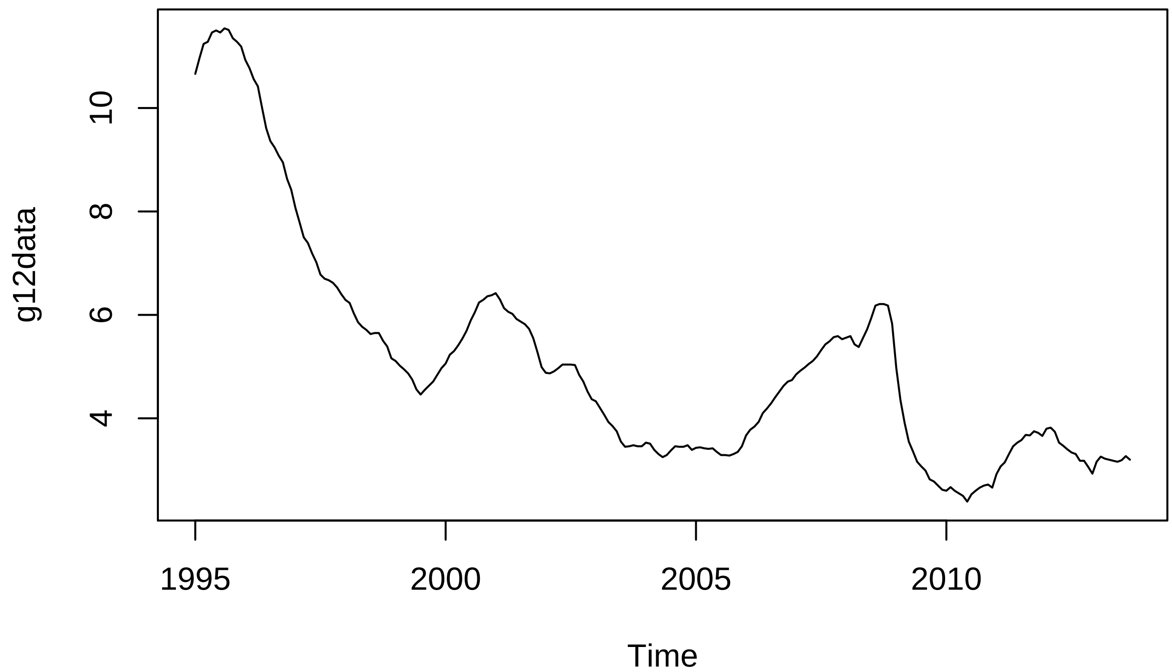
```
par(mfrow=c(3,1))
plot(g12data)
acf(g12data)
pacf(g12data)
```



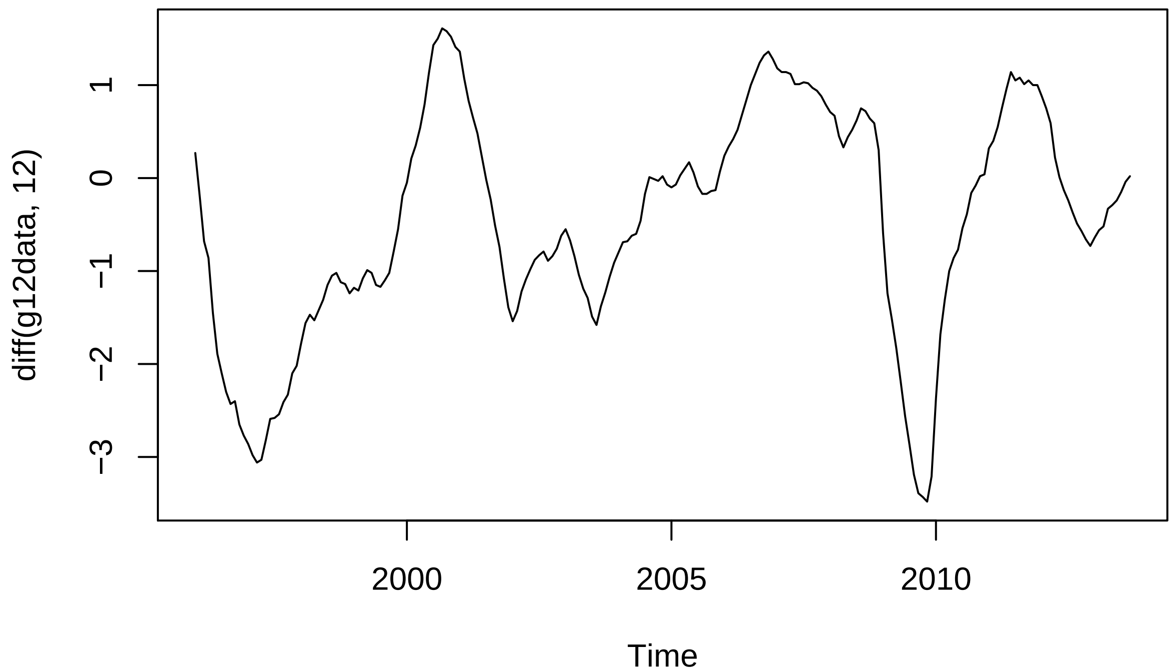




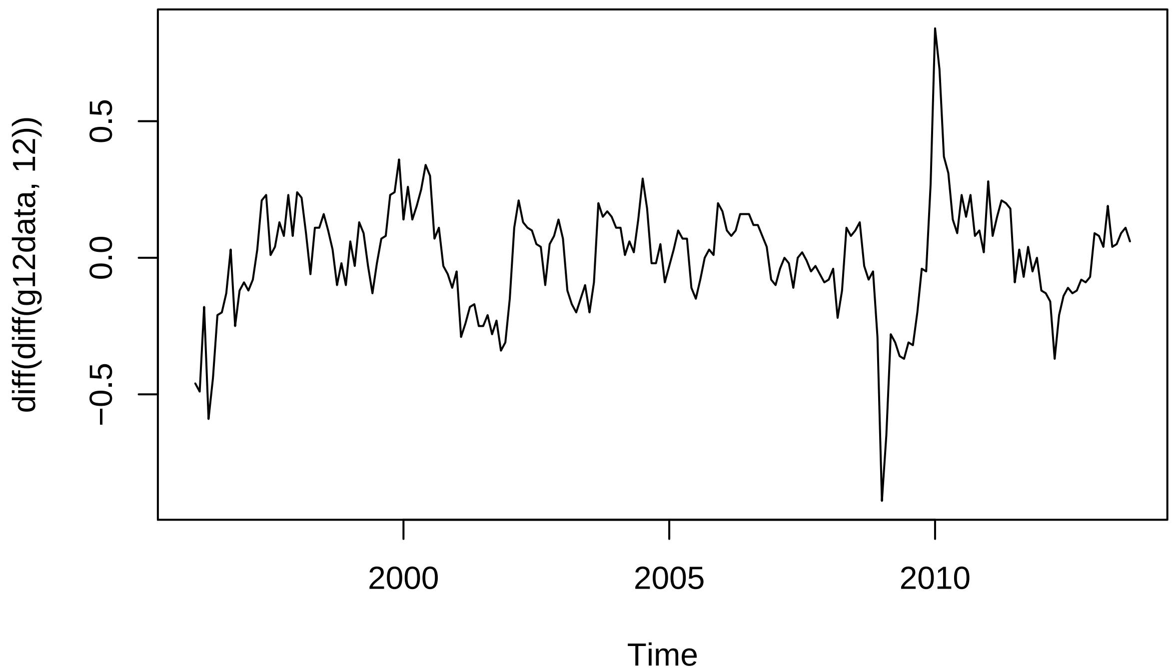

```r
plot(g12data)
```



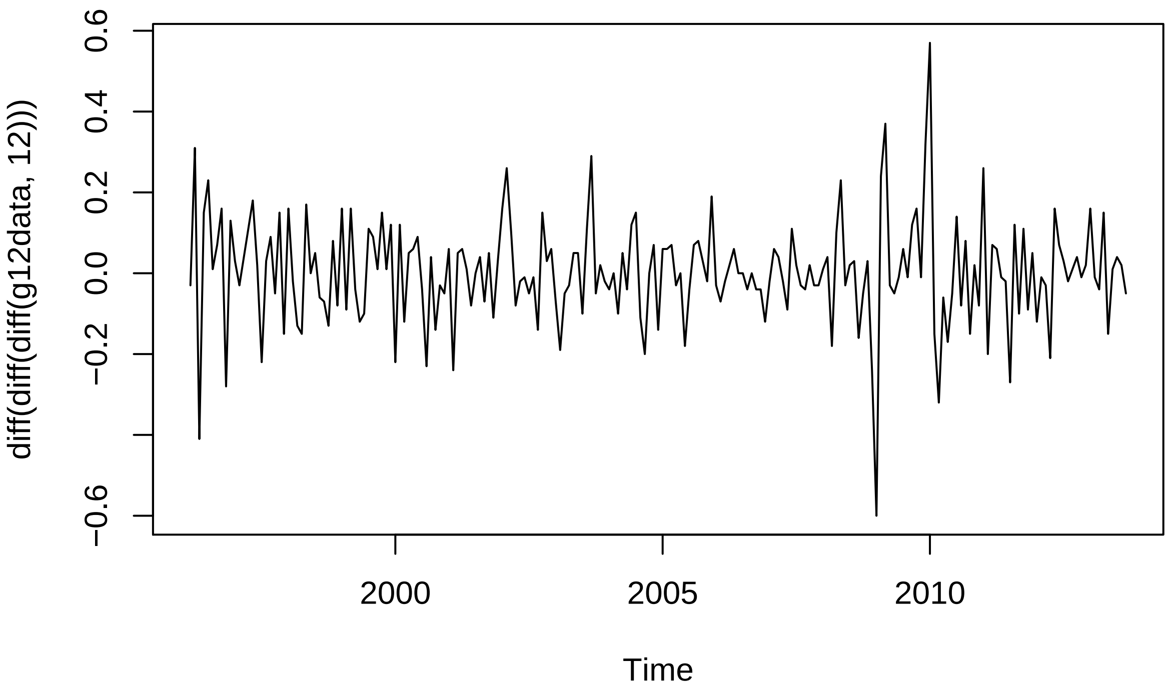

```r
plot(diff(g12data,12))
```



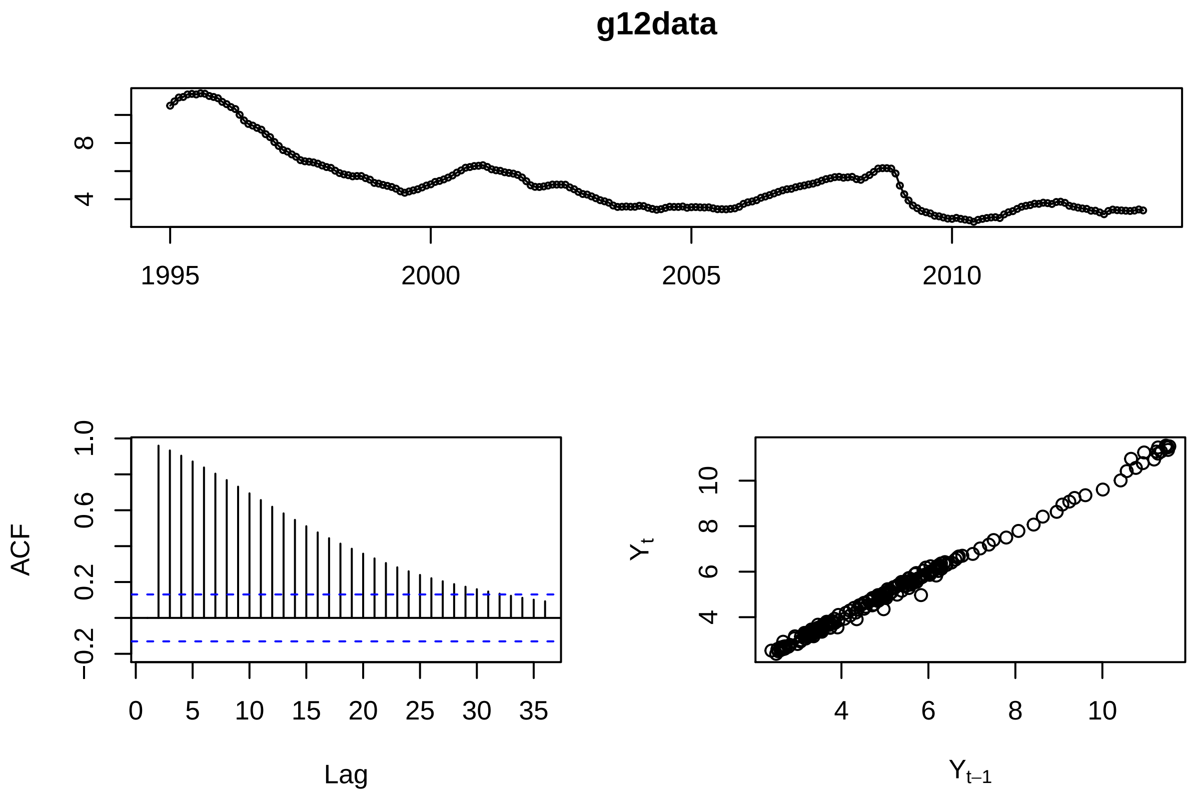

```r
plot(diff(diff(g12data,12))) #it looks more random
```



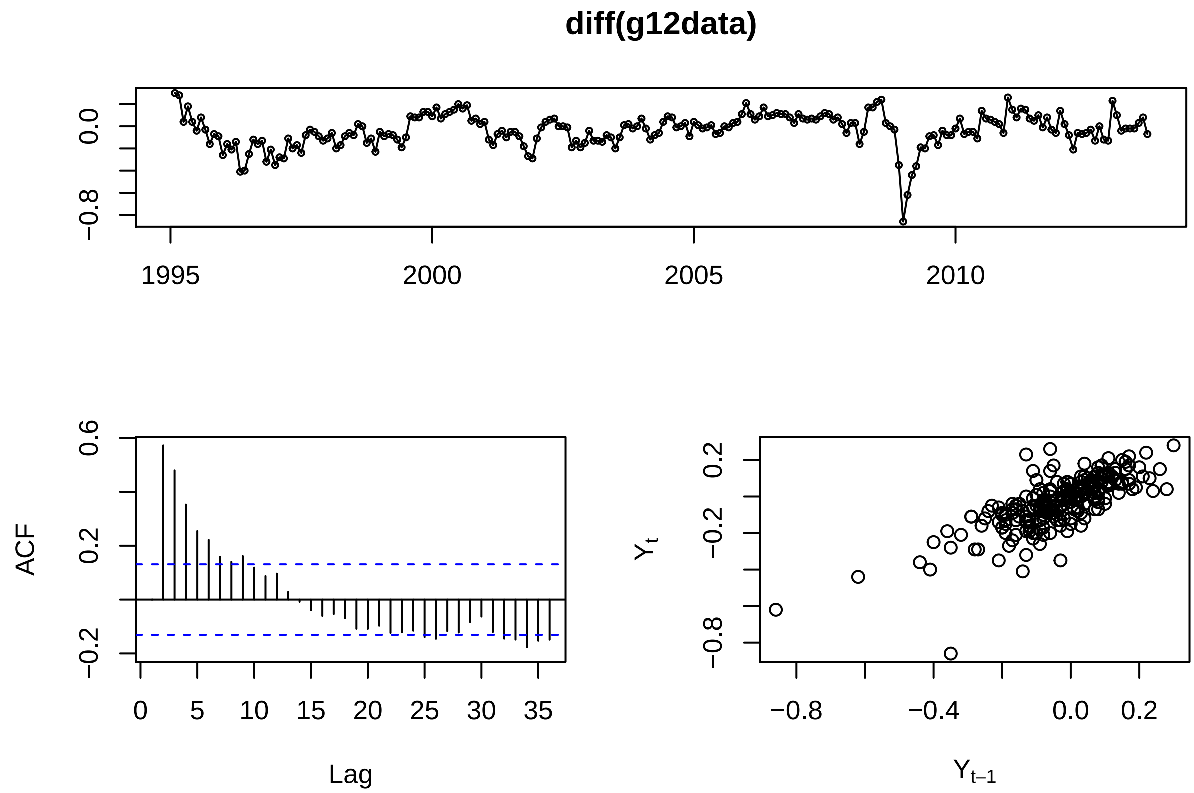

```
plot(diff(diff(diff(g12data,12)))) #it looks more random (seems the best one)
```



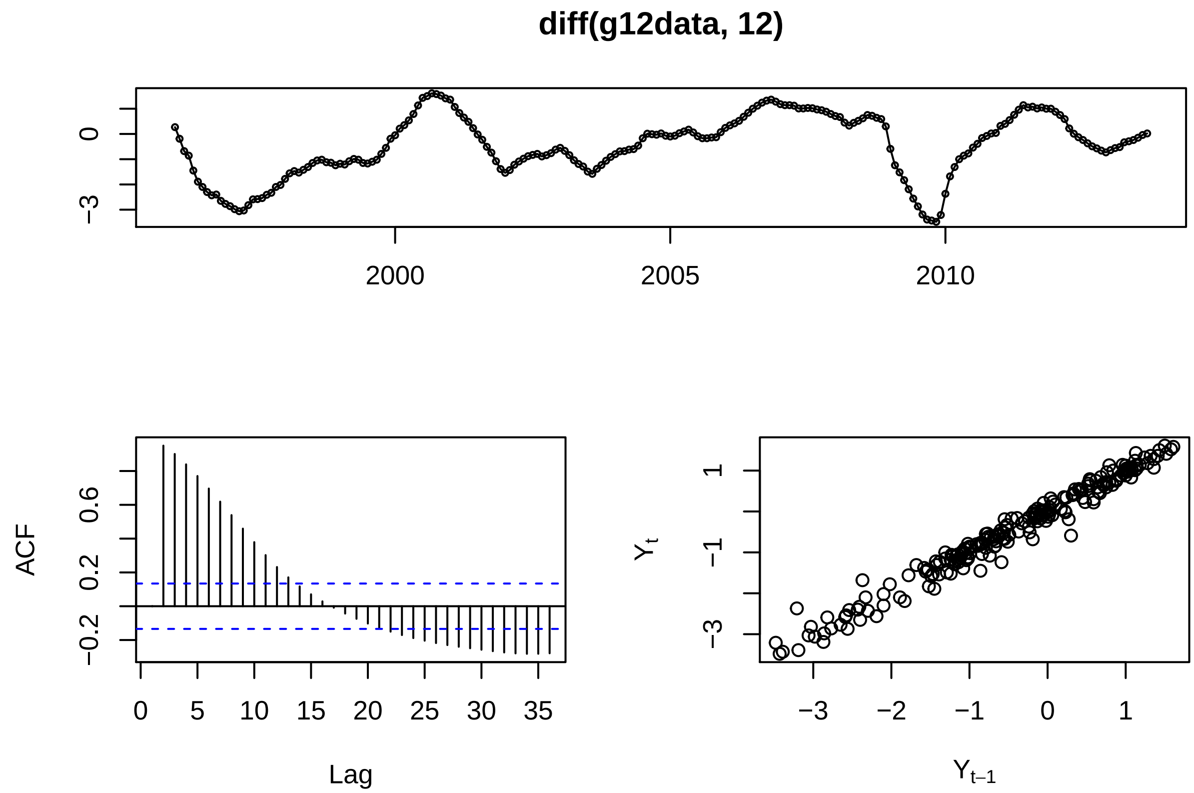

```r
# exploring ggb
tsdisplay(g12data, plot.type="scatter")
```



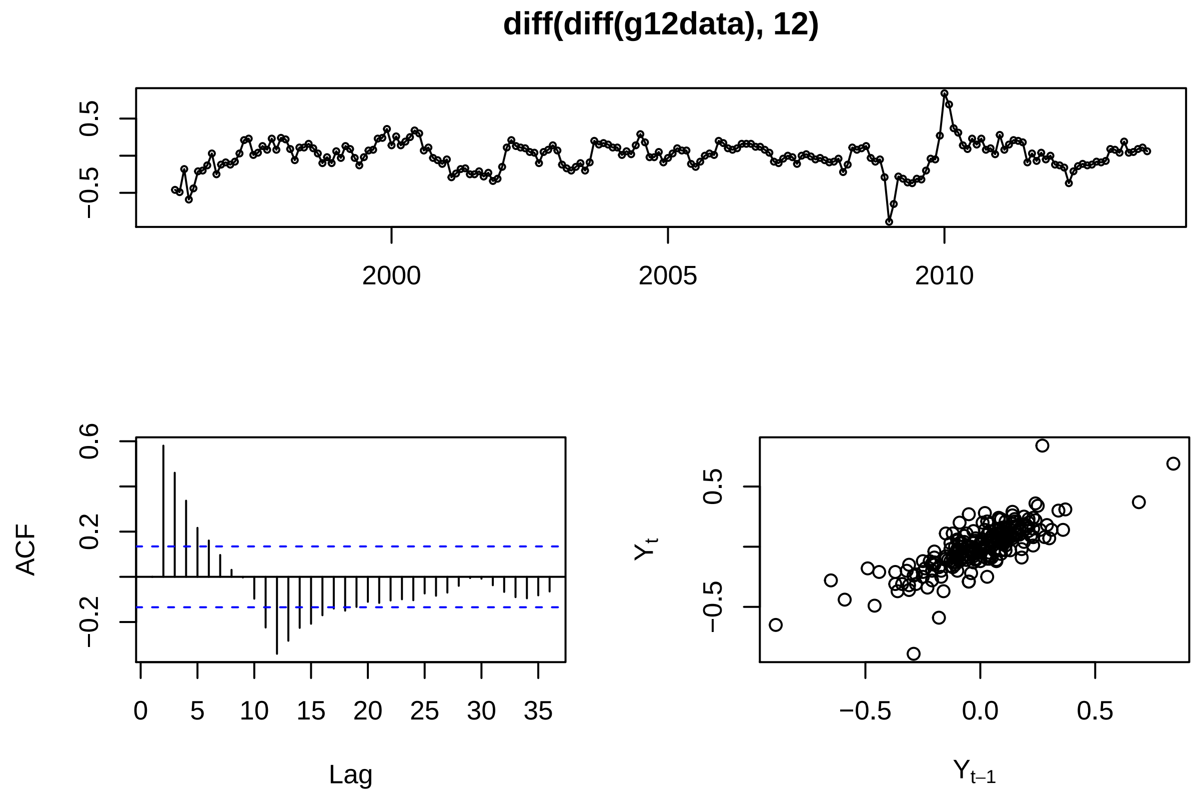

```
tsdisplay(diff(g12data), plot.type="scatter")
```



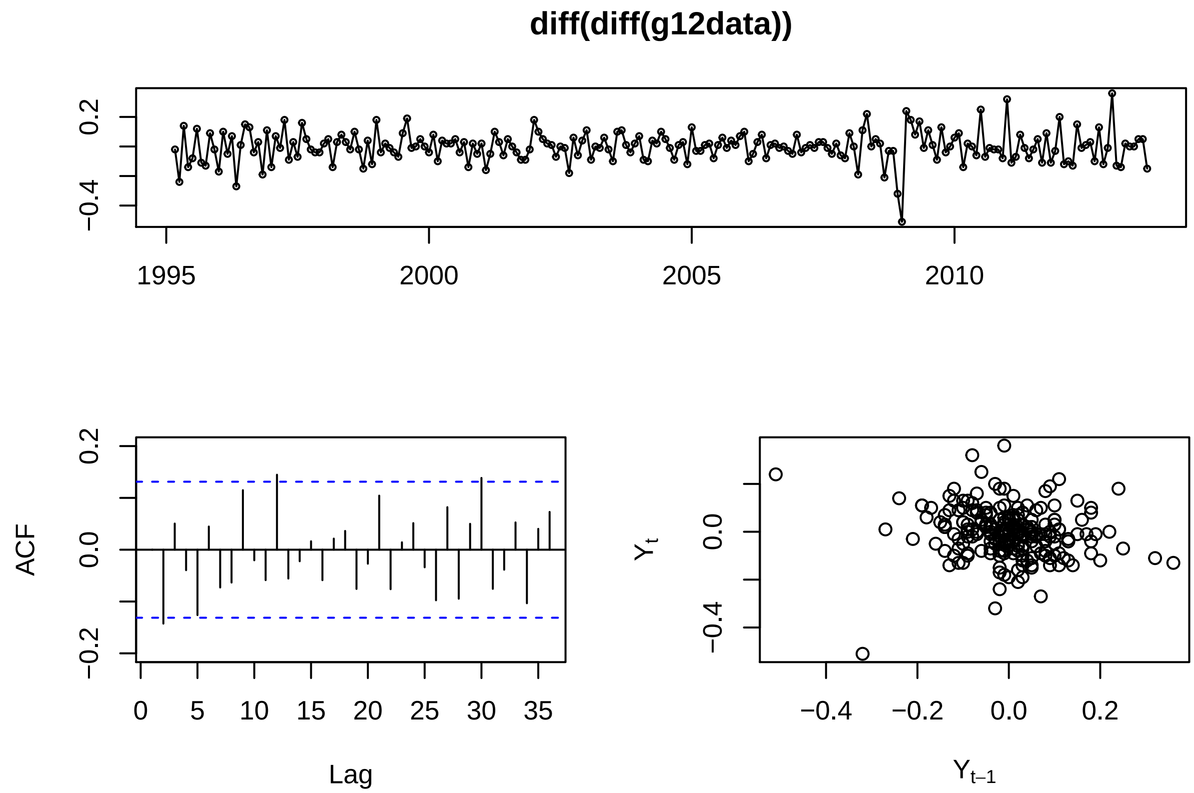

```r
tsdisplay(diff(g12data,12), plot.type="scatter")
```



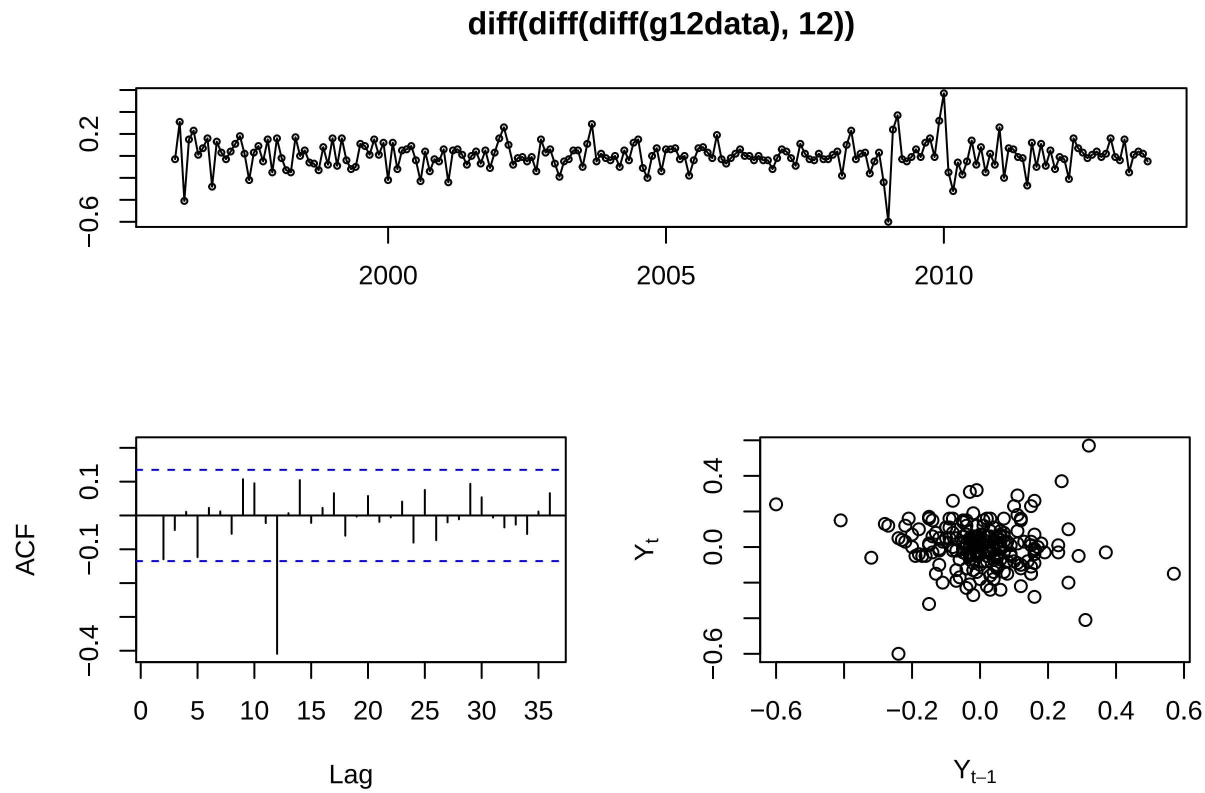

```
tsdisplay(diff(diff(g12data),12), plot.type="scatter")
```



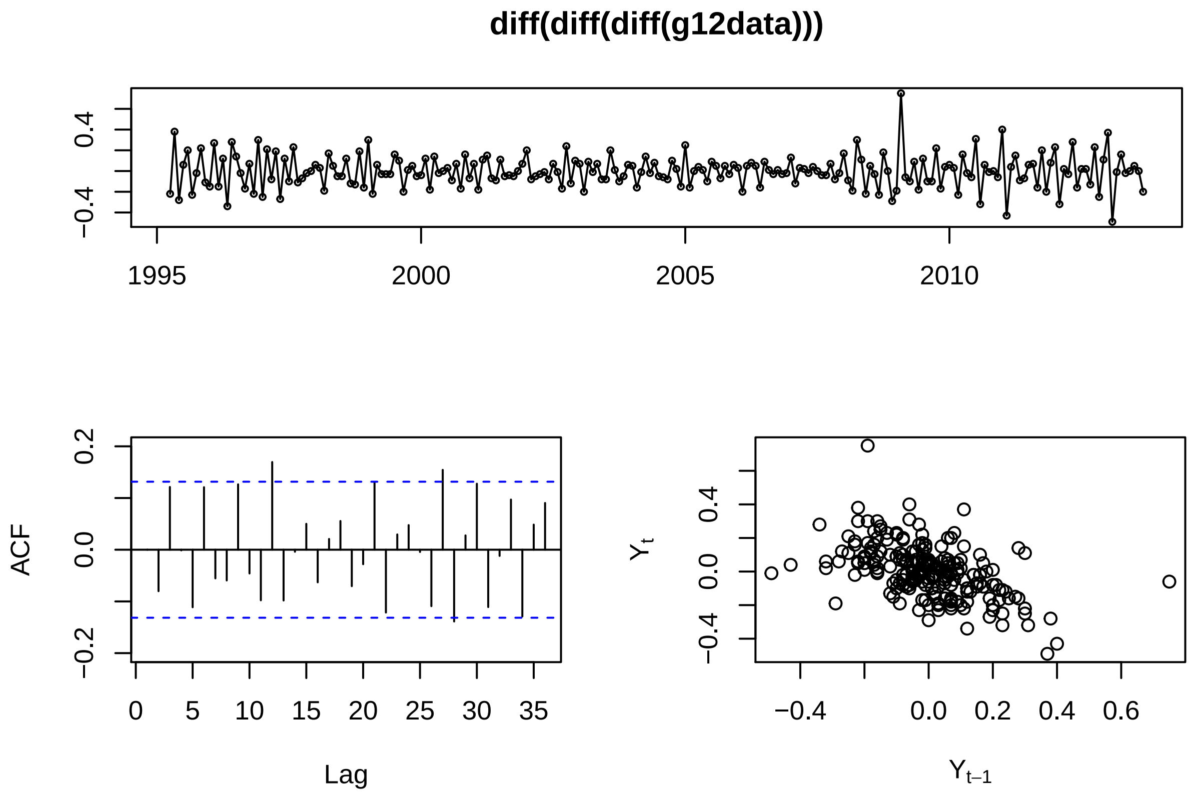

```
tsdisplay(diff(diff(g12data)), plot.type="scatter")
```



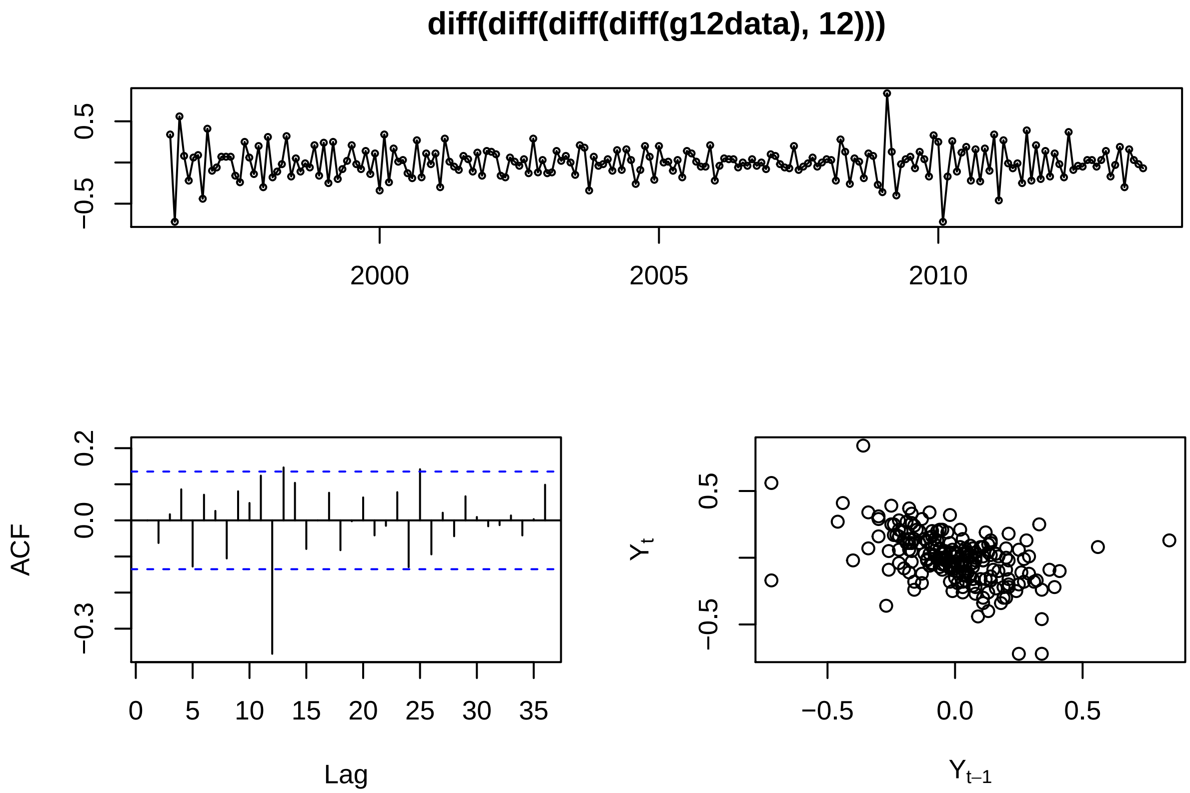

```r
tsdisplay(diff(diff(diff(g12data),12)), plot.type="scatter")
```



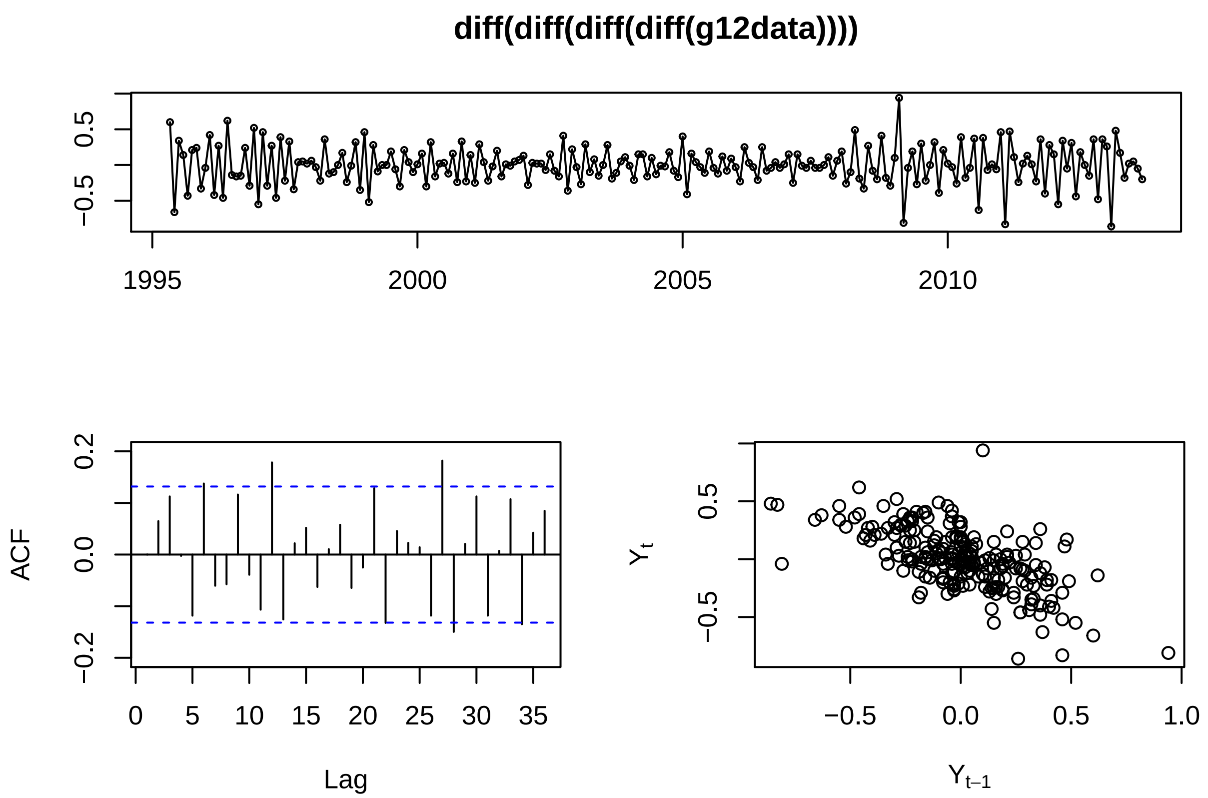

```r
tsdisplay(diff(diff(diff(g12data))), plot.type="scatter")
```




```r
tsdisplay(diff(diff(diff(diff(g12data),12))), plot.type="scatter") #not improved
```




```r
tsdisplay(diff(diff(diff(diff(g12data)))), plot.type="scatter") #not improved (bad)
```



Check if it's stationary to see when we can stop differentiating Can only be applied to seasonality adjusted data (in our case we didn't find a seasonality pattern so it's ok)

```r
adf.test(diff(g12data,12))
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff(g12data, 12)
## Dickey-Fuller = -3.534, Lag order = 5, p-value = 0.04067
## alternative hypothesis: stationary
```

```r
adf.test(diff(g12data))
```

```
## Warning in adf.test(diff(g12data)): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff(g12data)
## Dickey-Fuller = -4.3715, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

```r
adf.test(diff(diff(g12data,12)))
```

```
## Warning in adf.test(diff(diff(g12data, 12))): p-value smaller than printed
## p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff(diff(g12data, 12))
## Dickey-Fuller = -4.2591, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

```r
adf.test(diff(diff(g12data)))
```

```
## Warning in adf.test(diff(diff(g12data))): p-value smaller than printed p-
## value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff(diff(g12data))
## Dickey-Fuller = -7.6347, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

```r
#Check when to stop using standard deviation
sd(g12data)
```

```
## [1] 2.239662
```

```r
sd(diff(g12data)) #BETTER
```

```
## [1] 0.1486072
```

```r
sd(diff(g12data,12))
```

```
## [1] 1.205867
```

```r
sd(diff(diff(g12data,12)))
```

```
## [1] 0.199808
```

```r
sd(diff(diff(g12data))) #Smallest!!! Not seasonality applied
```

```
## [1] 0.1030107
```

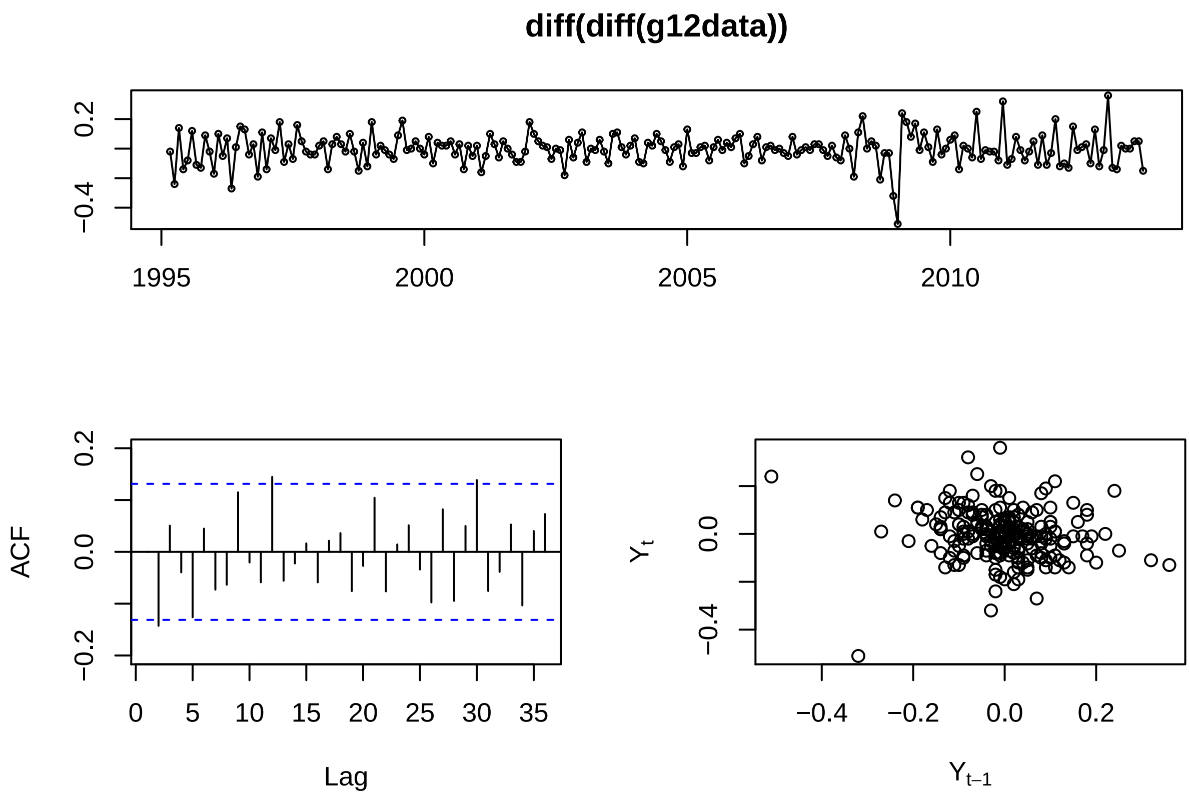
```r
sd(diff(diff(diff(g12data,12))))
```

```
## [1] 0.1311037
```

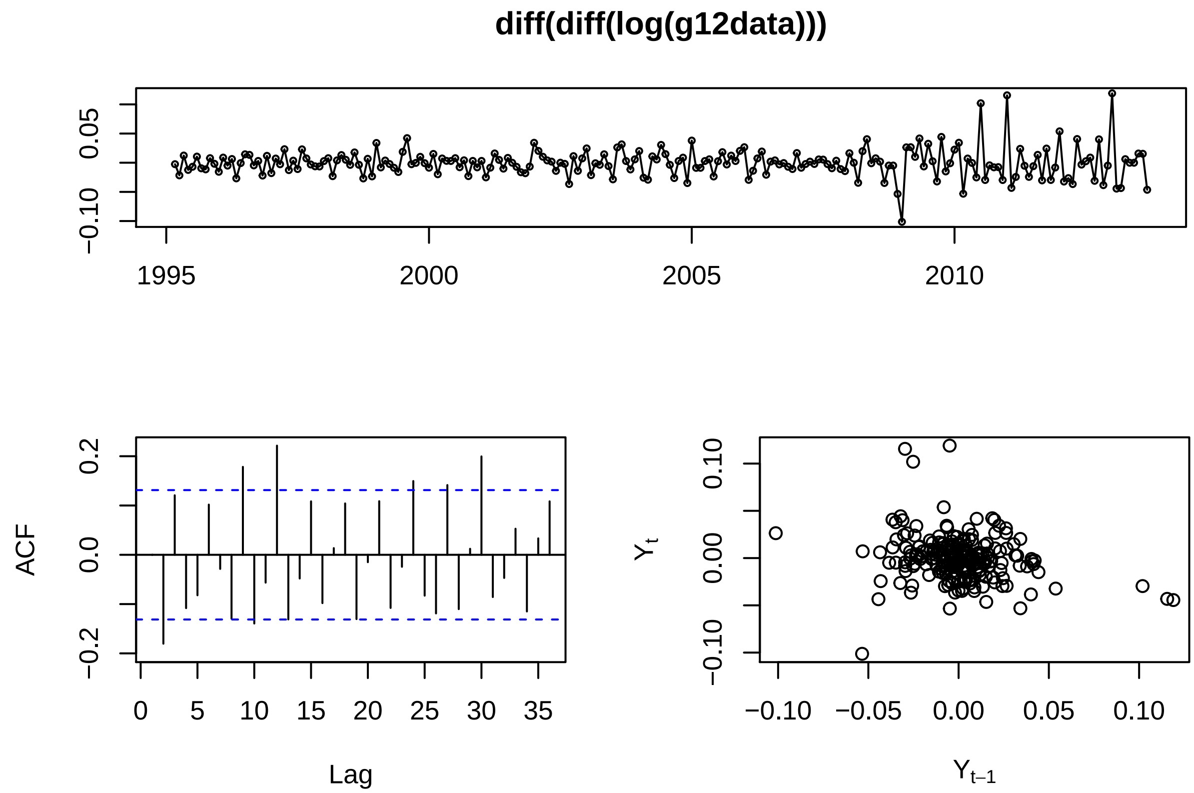
```r
sd(diff(diff(diff(g12data))))
```

```
## [1] 0.1562024
```

```r
tsdisplay(diff(diff(g12data)), plot.type="scatter")
```




```r
tsdisplay(diff(diff(log(g12data))), plot.type="scatter")  #Improves slightly (but model is more complic
```



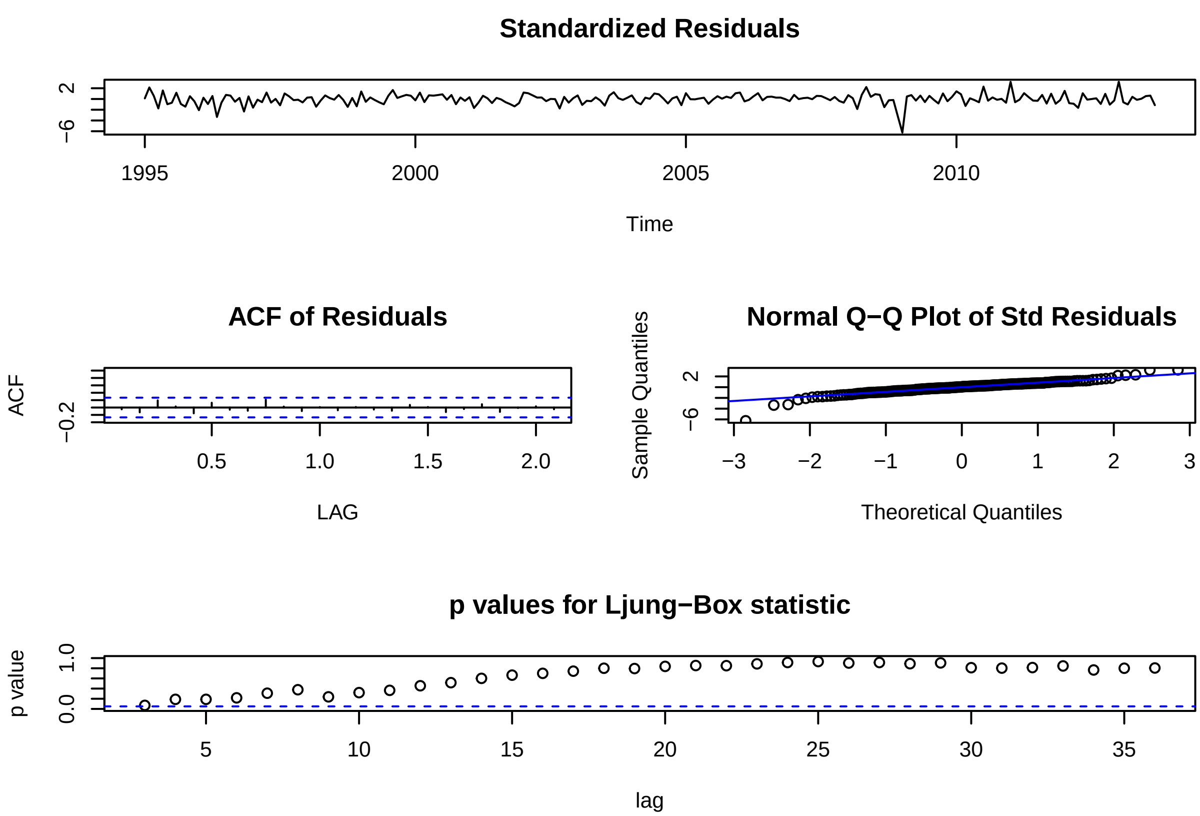
ARIMA(p,d,q)(P,D,Q) p->past observations q->past errors d->diferentiation order

D-> seasonal differentiation order (D=0)

```
#We start from 2 normal diffs and none in the seasonal part because
#any diff with period was done before to improve the sd
g12arima.1=Arima(g12data,order=c(0,2,0),include.mean=1,seasonal=list(order=c(0,0,0)))

#The best one is Arima3
g12arima.2=Arima(g12data,order=c(1,2,0),include.mean=1,seasonal=list(order=c(0,0,0)))
g12arima.3=Arima(g12data,order=c(1,2,1),include.mean=1,seasonal=list(order=c(0,0,0)))
g12arima.4=Arima(g12data,order=c(0,2,1),include.mean=1,seasonal=list(order=c(0,0,0)))

g12arima.4=Arima(g12data,order=c(1,0,1),include.mean=1,seasonal=list(order=c(0,1,0)))

#The best one now is Arima7
g12arima.5=Arima(g12data,order=c(1,2,1),include.mean=1,seasonal=list(order=c(1,0,1)))
g12arima.6=Arima(g12data,order=c(2,1,1),include.mean=1,seasonal=list(order=c(1,0,1)))
g12arima.7=Arima(g12data,order=c(2,1,0),include.mean=1,seasonal=list(order=c(1,0,1)))

#my-automatic (d=1 D=0): 1 0 1 0 (sarima)
sarima(g12data,1,1,0,1,0,0,12)
```

```
## initial  value -1.917453
## iter   2 value -2.365786
## iter   3 value -2.366825
```

```
## iter   4 value -2.366826
## iter   5 value -2.366828
## iter   6 value -2.366828
## iter   7 value -2.366831
## iter   8 value -2.366832
## iter   9 value -2.366833
## iter  10 value -2.366833
## iter  10 value -2.366833
## iter  10 value -2.366833
## final  value -2.366833
## converged
## initial  value -2.344369
## iter   2 value -2.344408
## iter   3 value -2.344421
## iter   4 value -2.344436
## iter   5 value -2.344453
## iter   6 value -2.344469
## iter   7 value -2.344481
## iter   8 value -2.344485
## iter   9 value -2.344485
## iter  10 value -2.344485
## iter  10 value -2.344485
## iter  10 value -2.344485
## final  value -2.344485
## converged
```



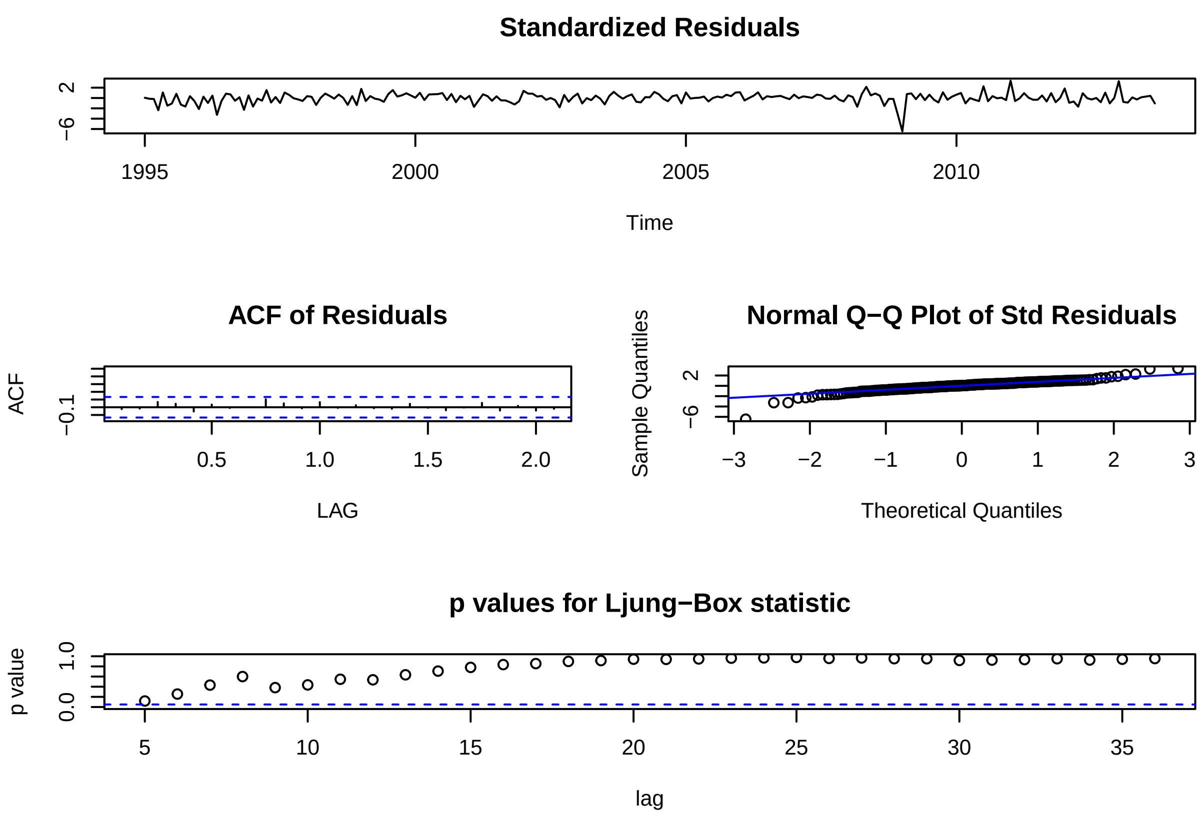
```
## $fit
##
## Call:
## stats::arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D,
##     Q), period = S), xreg = constant, optim.control = list(trace = trc, REPORT = 1,
##     reltol = tol))
##
## Coefficients:
##          ar1    sar1  constant
##       0.7679  0.1523   -0.0269
## s.e.  0.0434  0.0704    0.0318
##
## sigma^2 estimated as 0.009148:  log likelihood = 207.32,  aic = -406.64
##
## $AIC
## [1] -3.667596
##
## $AICc
## [1] -3.657899
##
## $BIC
## [1] -4.622048
```

```r
#my-automatic (d=2 D=0): 1 1 1 1 (sarima) -> more complex and less aic
sarima(g12data,1,2,1,1,0,1,12)
```

```
## initial  value -2.288639
## iter   2 value -2.302680
## iter   3 value -2.310837
## iter   4 value -2.311133
## iter   5 value -2.318350
## iter   6 value -2.322359
## iter   7 value -2.324525
## iter   8 value -2.328000
## iter   9 value -2.330996
## iter  10 value -2.336800
## iter  11 value -2.339097
## iter  12 value -2.346044
## iter  13 value -2.351003
## iter  14 value -2.354731
## iter  15 value -2.355292
## iter  16 value -2.355456
## iter  17 value -2.356229
## iter  18 value -2.356539
## iter  19 value -2.357263
## iter  20 value -2.357301
## iter  21 value -2.357314
## iter  22 value -2.357359
## iter  23 value -2.357401
## iter  24 value -2.357441
## iter  25 value -2.357454
## iter  26 value -2.357455
## iter  27 value -2.357456
## iter  27 value -2.357456
```

```
## iter  27 value -2.357456
## final  value -2.357456
## converged
## initial  value -2.329487
## iter   2 value -2.334453
## iter   3 value -2.335600
## iter   4 value -2.336010
## iter   5 value -2.336288
## iter   6 value -2.336374
## iter   7 value -2.336381
## iter   8 value -2.336411
## iter   9 value -2.336492
## iter  10 value -2.336606
## iter  11 value -2.336653
## iter  12 value -2.336697
## iter  13 value -2.336725
## iter  14 value -2.336730
## iter  15 value -2.336758
## iter  16 value -2.336784
## iter  17 value -2.336859
## iter  18 value -2.337004
## iter  19 value -2.337293
## iter  20 value -2.337326
## iter  21 value -2.337534
## iter  22 value -2.337589
## iter  23 value -2.337597
## iter  24 value -2.337598
## iter  25 value -2.337629
## iter  26 value -2.337666
## iter  27 value -2.337727
## iter  28 value -2.337835
## iter  29 value -2.337884
## iter  30 value -2.337910
## iter  31 value -2.338006
## iter  32 value -2.338013
## iter  33 value -2.338019
## iter  34 value -2.338039
## iter  35 value -2.338081
## iter  36 value -2.338359
## iter  37 value -2.338374
## iter  38 value -2.338718
## iter  39 value -2.338757
## iter  40 value -2.338781
## iter  41 value -2.338923
## iter  42 value -2.339119
## iter  43 value -2.339412
## iter  44 value -2.339925
## iter  45 value -2.340382
## iter  46 value -2.342693
## iter  47 value -2.342921
## iter  48 value -2.343036
## iter  49 value -2.343080
## iter  50 value -2.343165
## iter  51 value -2.343363
```

```
## iter  52 value -2.343736
## iter  53 value -2.343876
## iter  54 value -2.343978
## iter  55 value -2.343980
## iter  56 value -2.344027
## iter  57 value -2.344041
## iter  58 value -2.344052
## iter  59 value -2.344054
## iter  60 value -2.344054
## iter  61 value -2.344062
## iter  62 value -2.344066
## iter  63 value -2.344069
## iter  64 value -2.344069
## iter  64 value -2.344069
## iter  64 value -2.344069
## final  value -2.344069
## converged
```




```
## $fit
##
## Call:
## stats::arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D,
##     Q), period = S), include.mean = !no.constant, optim.control = list(trace = trc,
##     REPORT = 1, reltol = tol))
##
```

```
## Coefficients:
##          ar1    ma1    sar1     sma1
##       0.7897  -1.00  0.9537  -0.8819
## s.e.  0.0436   0.01  0.0610   0.1001
##
## sigma^2 estimated as 0.009001:  log likelihood = 206.3,  aic = -402.61
##
## $AIC
## [1] -3.674919
##
## $AICc
## [1] -3.664812
##
## $BIC
## [1] -4.614188
```

```
#The one obtained with my-automatic is better (less aic and less complex)
g12arima.8=Arima(g12data,order=c(1,1,0),include.mean=1,seasonal=list(order=c(1,0,0)))
auto.arima(g12data)
```

```
## Series: g12data
## ARIMA(2,1,1)(2,0,0)[12]
##
## Coefficients:
```

```
## Warning in sqrt(diag(x$var.coef)): Se han producido NaNs
```

```
##           ar1     ar2     ma1    sar1    sar2
##       -0.0471  0.6336  0.8311  0.1583  0.0089
## s.e.      NaN     NaN     NaN  0.0709  0.0762
##
## sigma^2 estimated as 0.009169:  log likelihood=207.03
## AIC=-402.06   AICc=-401.68   BIC=-381.59
```

```
AIC(g12arima.1)
```

```
## [1] -379.8211
```

```
AIC(g12arima.2)
```

```
## [1] -382.8607
```

```
AIC(g12arima.3)
```

```
## [1] -398.6795
```

```
AIC(g12arima.4)
```

```
## [1] -201.8503
```

```r
AIC(g12arima.5)
```

```
## [1] -402.608
```

```r
AIC(g12arima.6)
```

```
## [1] -404.0317
```

```r
AIC(g12arima.7) #This is the best one (it is the lowest one)
```

```
## [1] -406.672
```

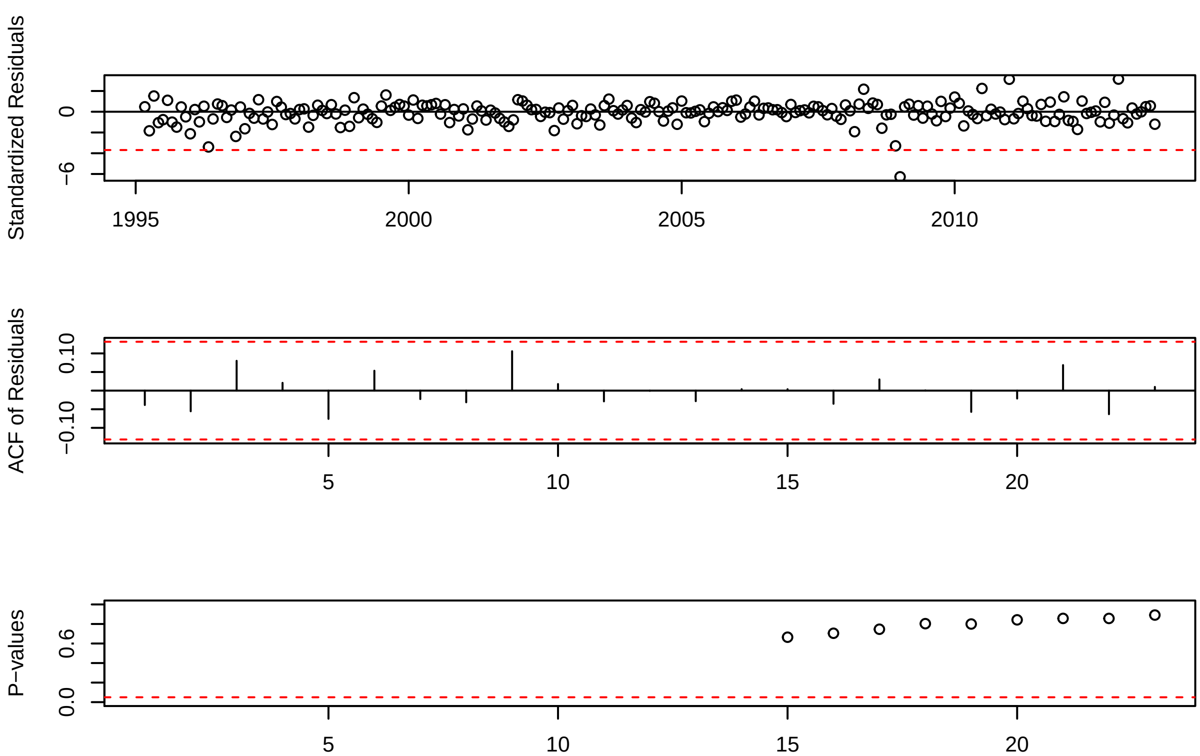
```r
AIC(g12arima.8) #This is the best one (it is the lowest one being simpler)
```

```
## [1] -407.9574
```

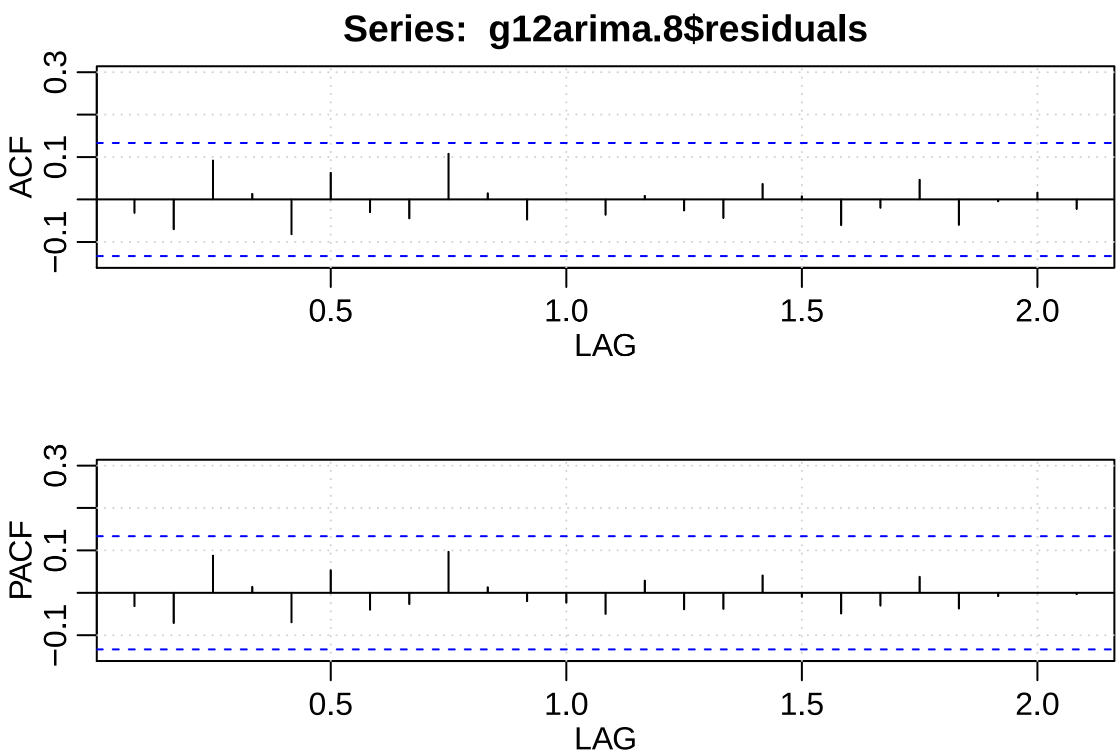
We'll keep the last one

Maybe compare them with bad ones?? WTF?

```r
tsdiag(g12arima.8)
```



```r
acf2(g12arima.8$residuals)
```




```
##          ACF  PACF
##  [1,] -0.03 -0.03
##  [2,] -0.07 -0.07
##  [3,]  0.09  0.09
##  [4,]  0.01  0.01
##  [5,] -0.08 -0.07
##  [6,]  0.06  0.05
##  [7,] -0.03 -0.04
##  [8,] -0.04 -0.03
##  [9,]  0.11  0.10
## [10,]  0.01  0.01
## [11,] -0.05 -0.02
## [12,]  0.00 -0.02
## [13,] -0.04 -0.05
## [14,]  0.01  0.03
## [15,] -0.03 -0.04
## [16,] -0.04 -0.04
## [17,]  0.04  0.04
## [18,]  0.01 -0.01
## [19,] -0.06 -0.05
## [20,] -0.02 -0.03
## [21,]  0.05  0.04
## [22,] -0.06 -0.04
```

```
## [23,]  0.00 -0.01
## [24,]  0.02  0.00
## [25,] -0.02  0.00
```

```r
#correlations between model coefficients
cov2cor(g12arima.8$var.coef) #Not greater than 0.8 so it is good
```

```
##             ar1       sar1
## ar1  1.00000000 0.01790429
## sar1 0.01790429 1.00000000
```

```r
#testing independence of the residuals (apparently they are not independent???)
#CHECK THE P-VALUES (are them white noise or not???)
LB.test(g12arima.8) #it seems ok because in the bubble example the values are the same
```

```
##
##  Box-Ljung test
##
## data:  residuals from  g12arima.8
## X-squared = 9.8561, df = 10, p-value = 0.4532
```

```r
#or
LB.test(g12arima.8, lag=20)
```

```
##
##  Box-Ljung test
##
## data:  residuals from  g12arima.8
## X-squared = 12.098, df = 18, p-value = 0.8421
```

```r
#normality of the residuals
jarque.bera.test(residuals(g12arima.8)) #less than 0.05, they are not normal (seems bad)
```

```
##
##  Jarque Bera Test
##
## data:  residuals(g12arima.8)
## X-squared = 505.13, df = 2, p-value < 2.2e-16
```

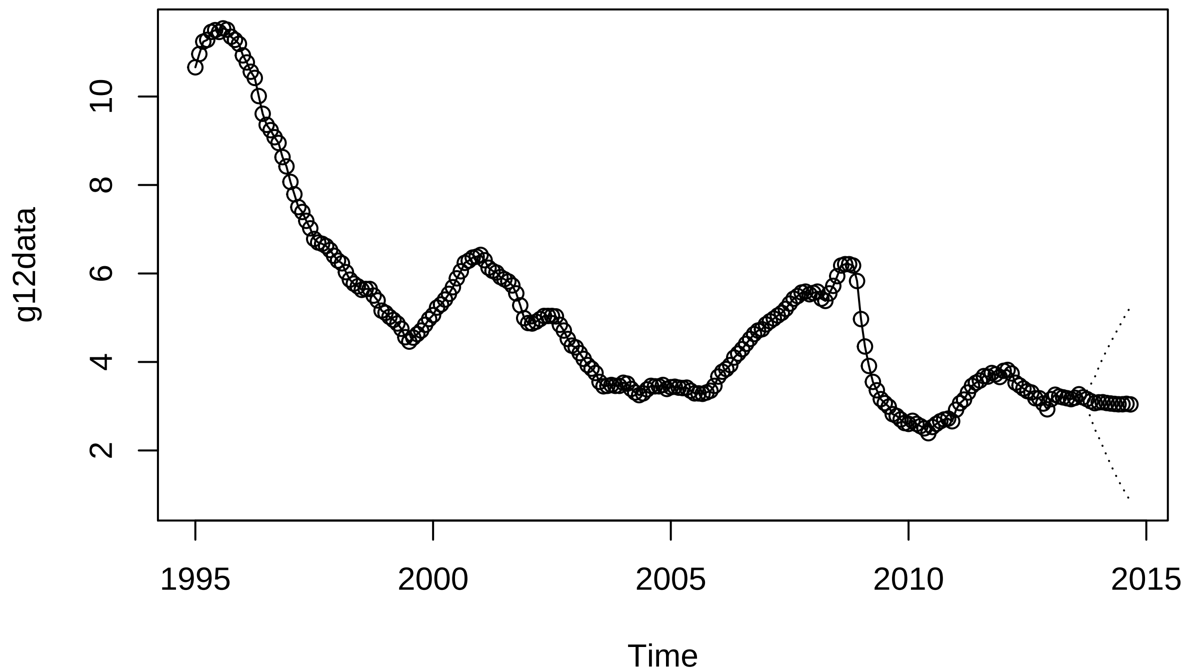
```r
#Checks the independence in a time series (greater than 0.05 seems alright)
Box.test(residuals(g12arima.8),lag=12)
```

```
##
##  Box-Pierce test
##
## data:  residuals(g12arima.8)
## X-squared = 9.4269, df = 12, p-value = 0.6661
```

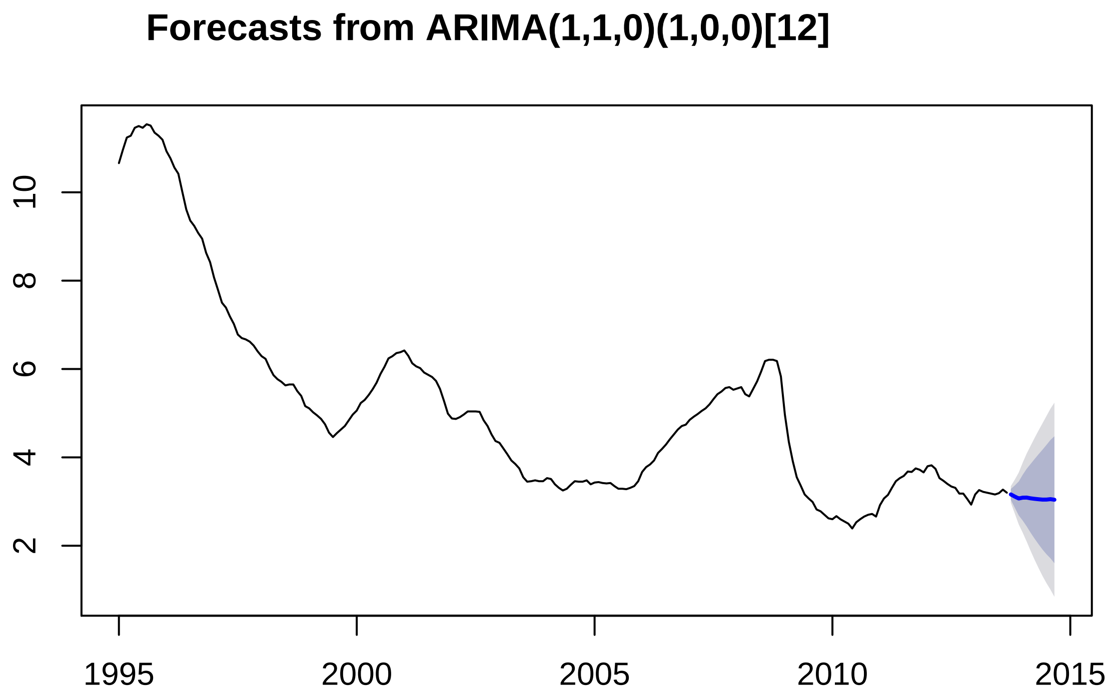
```r
#So the residuals seem normal but nott independent

## Forecasting with the model lynx.3
predict(g12arima.8,n.ahead=20)
```

```
## $pred
##           Jan      Feb      Mar      Apr      May      Jun      Jul
## 2013
## 2014 3.088048 3.090078 3.073036 3.061571 3.051973 3.043821 3.044712
## 2015 3.018141 3.017814 3.014611 3.012406 3.010583
##           Aug      Sep      Oct      Nov      Dec
## 2013                   3.161711 3.113026 3.069434
## 2014 3.054393 3.040984 3.033120 3.024014 3.016029
## 2015
##
## $se
##             Jan        Feb        Mar        Apr        May        Jun
## 2013
## 2014 0.40484209 0.50788243 0.60756348 0.70327324 0.79479915 0.88215913
## 2015 1.43162373 1.50639006 1.57925959 1.65014272 1.71902671
##             Jul        Aug        Sep        Oct        Nov        Dec
## 2013                                  0.09577884 0.19513118 0.29974721
## 2014 0.96550099 1.04504139 1.12102862 1.19901895 1.27741513 1.35516915
## 2015
```

```r
TSA::plot.Arima(g12arima.8) #TSA package
```




```
plot(forecast(g12arima.8,h=12))
```




```r
forecast(g12arima.8,h=12) #same predictions plotted with the TSA package
```

```
##          Point Forecast    Lo 80    Hi 80     Lo 95    Hi 95
## Oct 2013       3.161711 3.038966 3.284457 2.9739881 3.349434
## Nov 2013       3.113026 2.862956 3.363097 2.7305762 3.495476
## Dec 2013       3.069434 2.685292 3.453575 2.4819401 3.656928
## Jan 2014       3.088048 2.569222 3.606874 2.2945724 3.881524
## Feb 2014       3.090078 2.439200 3.740955 2.0946466 4.085509
## Mar 2014       3.073036 2.294412 3.851660 1.8822334 4.263839
## Apr 2014       3.061571 2.160290 3.962852 1.6831809 4.439961
## May 2014       3.051973 2.033397 4.070549 1.4941952 4.609751
## Jun 2014       3.043821 1.913289 4.174354 1.3148212 4.772821
## Jul 2014       3.044712 1.807373 4.282052 1.1523650 4.937059
## Aug 2014       3.054393 1.715119 4.393668 1.0061497 5.102637
## Sep 2014       3.040984 1.604328 4.477640 0.8438082 5.238160
```

We could try logs to see if the bad distribution of the residuals is fixed.

```r
g12arima.9=Arima(log(g12data),order=c(1,1,0),include.mean=1,seasonal=list(order=c(1,0,0)))
AIC(g12arima.9) #aic lower than before (by a lot)
```

```
## [1] -1074.566
```

```r
cov2cor(g12arima.9$var.coef) #Not greater than 0.8 so it is good
```

```
##           ar1     sar1
## ar1  1.000000 0.128937
## sar1 0.128937 1.000000
```

```r
#testing independence of the residuals (apparently they are not independent???)
#CHECK THE P-VALUES (are them white noise or not???)
LB.test(g12arima.9) #it seems ok because in the first example the values are the same (it is lower now
```

```
##
##  Box-Ljung test
##
## data:  residuals from  g12arima.9
## X-squared = 16.193, df = 10, p-value = 0.09425
```

```r
#or
LB.test(g12arima.9, lag=20)
```

```
##
##  Box-Ljung test
##
## data:  residuals from  g12arima.9
## X-squared = 23.981, df = 18, p-value = 0.1557
```

```r
#normality of the residuals
jarque.bera.test(residuals(g12arima.9)) #less than 0.05, they are not normal (seems bad)
```

```
##
##  Jarque Bera Test
##
## data:  residuals(g12arima.9)
## X-squared = 612.57, df = 2, p-value < 2.2e-16
```

```r
#Checks the independence in a time series (greater than 0.05 seems alright)
Box.test(residuals(g12arima.9),lag=12) #She uses it instead of jarque.vera when the other doesn't work
```

```
##
##  Box-Pierce test
##
## data:  residuals(g12arima.9)
## X-squared = 15.569, df = 12, p-value = 0.2118
```

We don’t take logs as it doesn’t improve anything