Time Series Assignment (Group 12)

Ignacio Amaya, Adrián Ramírez and Hugo Santana

8 de enero de 2016

Contents

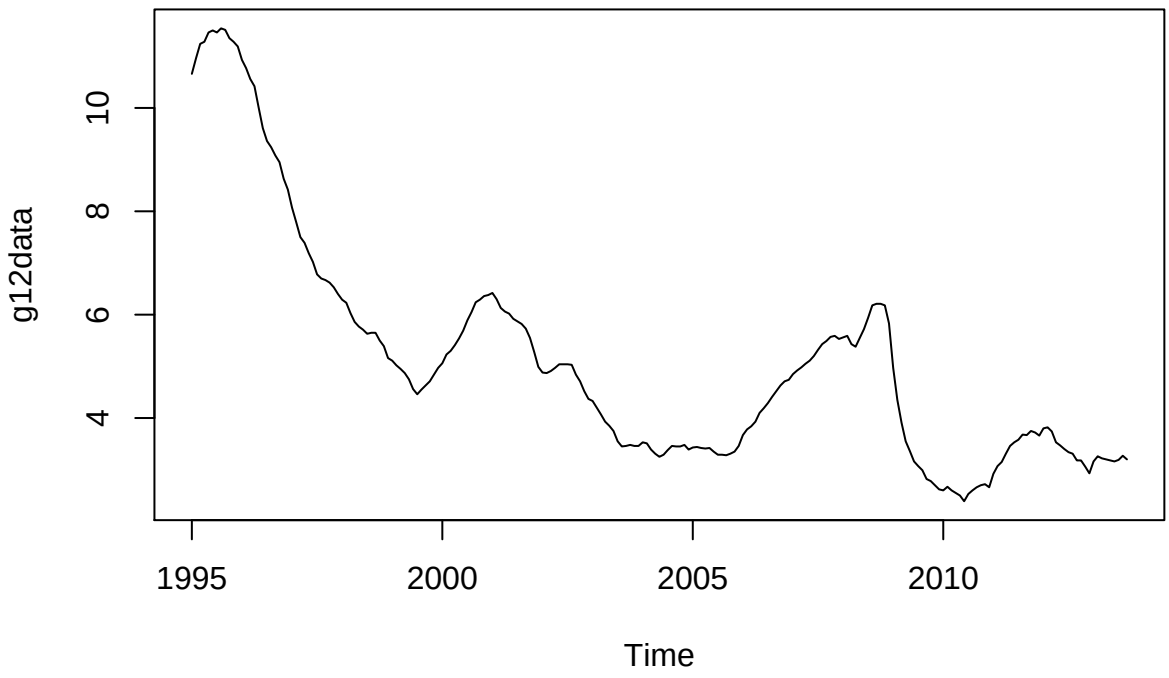
Overview

The time series we are going to study gathers the monthly interest rate of unsubsidised loans for home purchase (each value is expressed as a percentage). The data goes from January 1995 to September 2013, covering 18 years. This is a large amount of time, so we can consider our time series to be representative.

This data has been obtained from ww.bde.es (Banco de España website).

Firstly, we are going to plot the series and briefly comment on the main characteristics observed (trend, seasonality, stationarity...).

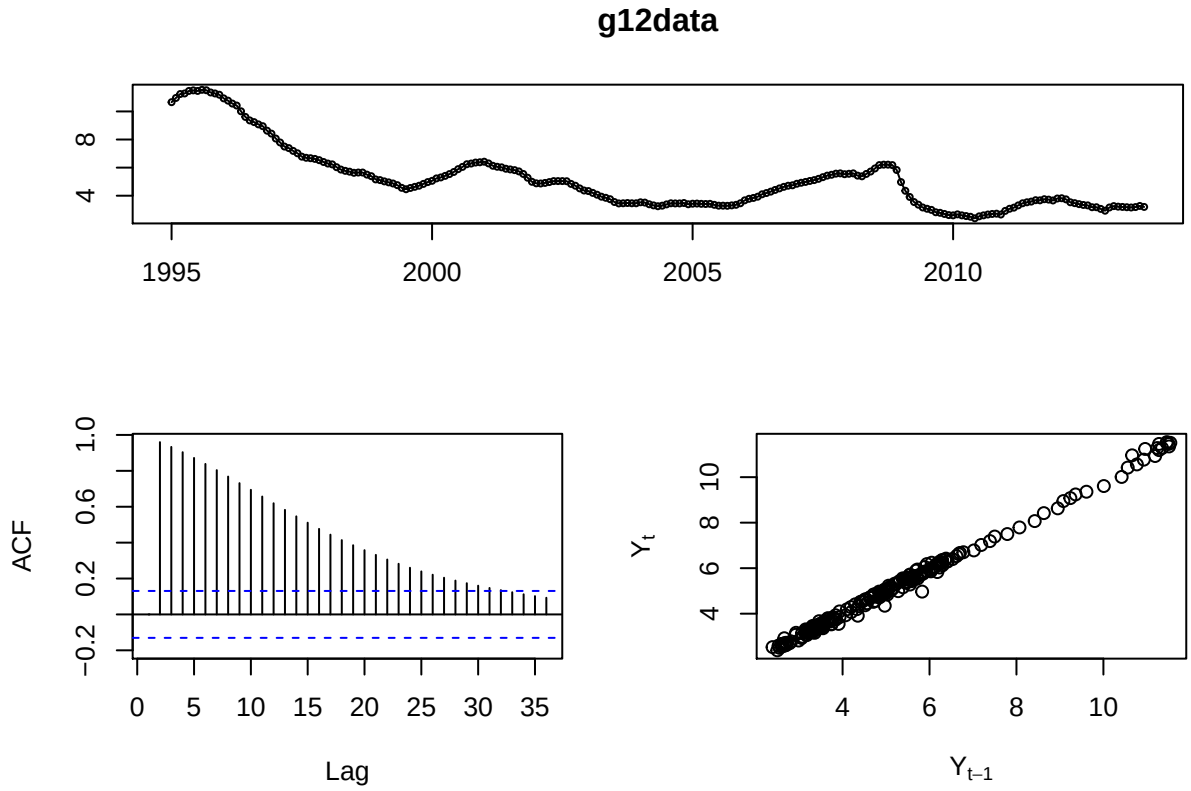
```
g12data=ts(dataAsigment2$Tipo,start=c(1995,1), end=c(2013,9),frequency=12)
plot(g12data)
```



As we can see, our series is clearly non-stationary because the mean and variance are not constant over time. We can't observe a clear seasonal pattern, but there's a decresing trend throughout the time. We can also observe that approximately each five years the trend is reversed for a small period of time, but as this doesn't happen on regular basis every year, we can't consider this as seasonality. The cause of this cyclic pattern is surely due to the cyclic nature of capitalism.

We can plot this same series in a more complex and fancy way to get more information about our data. In this plot we have the autocorrelation plot in the bottom left corner and a lagged scatterplot in the bottom right corner.

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



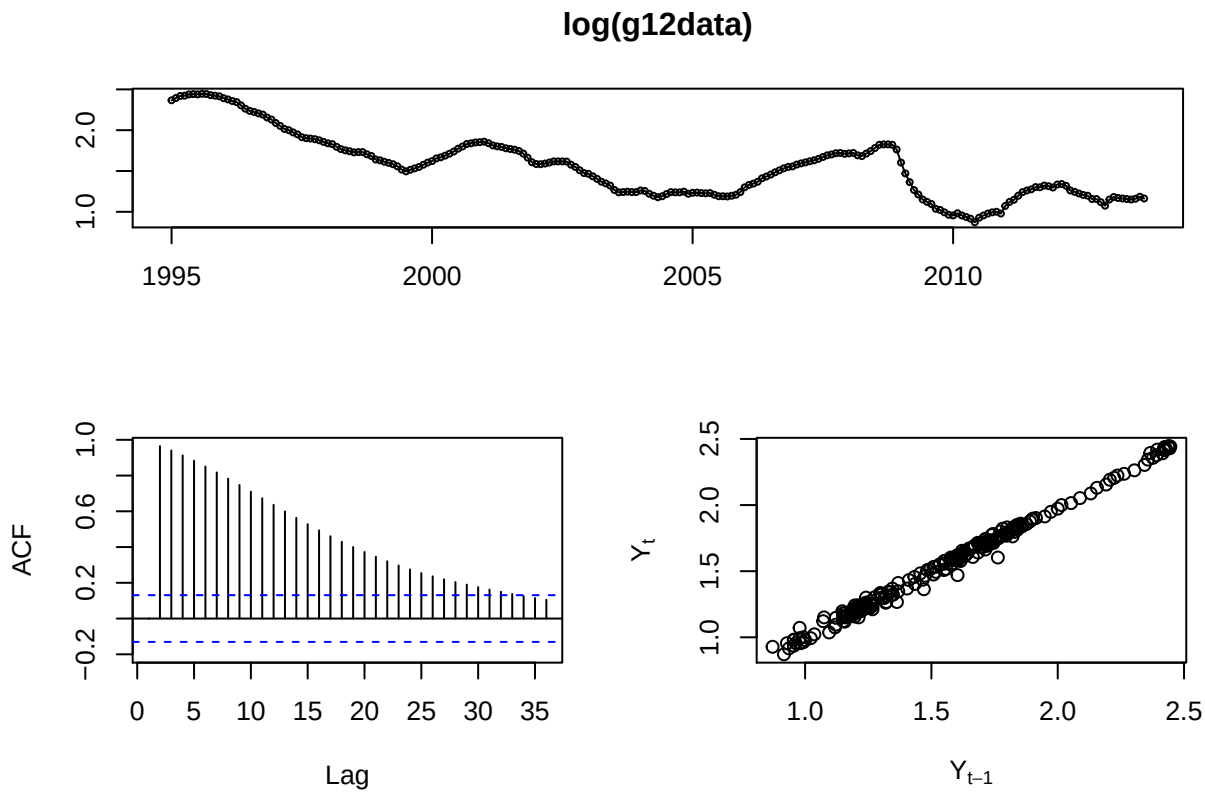
Checking the autocorrelation plot we clearly see that our series lacks randomness because the values are not close to zero. The good thing is that our values are decaying over time (though very slowly) which is a property of the ACF plot in stationary series. In the lagged scatterplot we can also check that our series is non-stationary because data compared with previous one spaced by a predefined time unit lag is highly correlated.

Sometimes applying logartithms can improve the stationary of our series. We are also going to try a BoxCox transformation is possible (with lambda between 0 and 2).

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

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

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



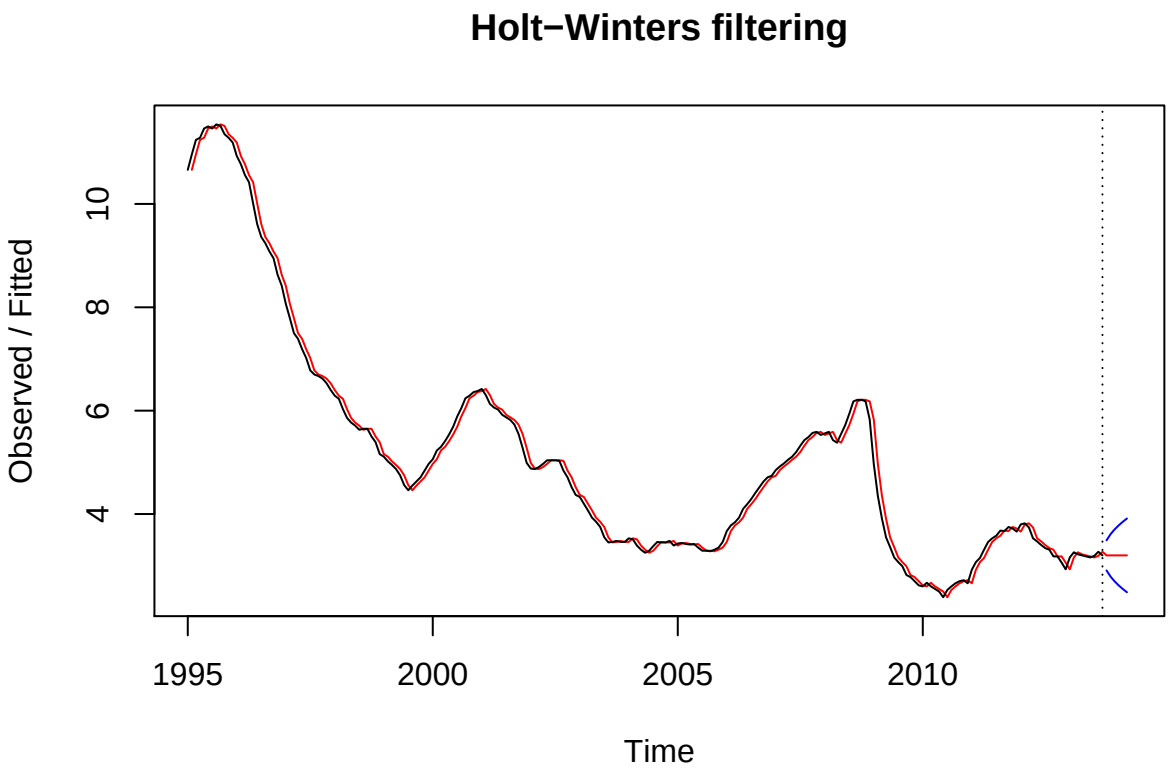
The lagged scatterplot has improved a little bit because the data is more uniformly distributed but the series is still far from being stationary. This improvement is so slight that doesn't justify taking logs because that increases the complexity of our model, which should be avoided when possible. Besides, the BoxCox value is not close to zero so there are not any good BoxCox transformation for the lambda interval specified.

Analysis

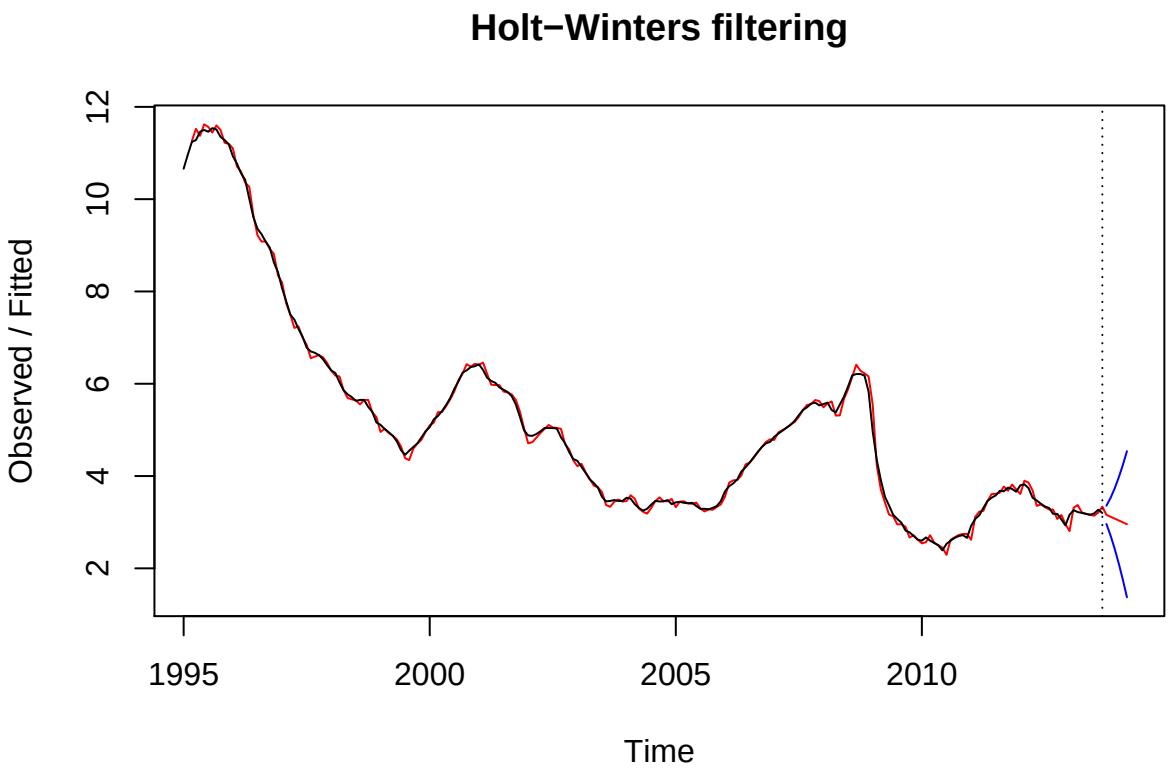
Smoothing

We can use smoothing methods to model our series. As our time series has trend but no seasonality the best model to use is a holt doulbe exponential smoothing (it uses two paramenters, alpha and beta). Plotting it along a simple exponential smoothing and a triple exponential smoothing we can observe that it is the best fit, which tells us that our assumptions about the trend and seasonality were correct.

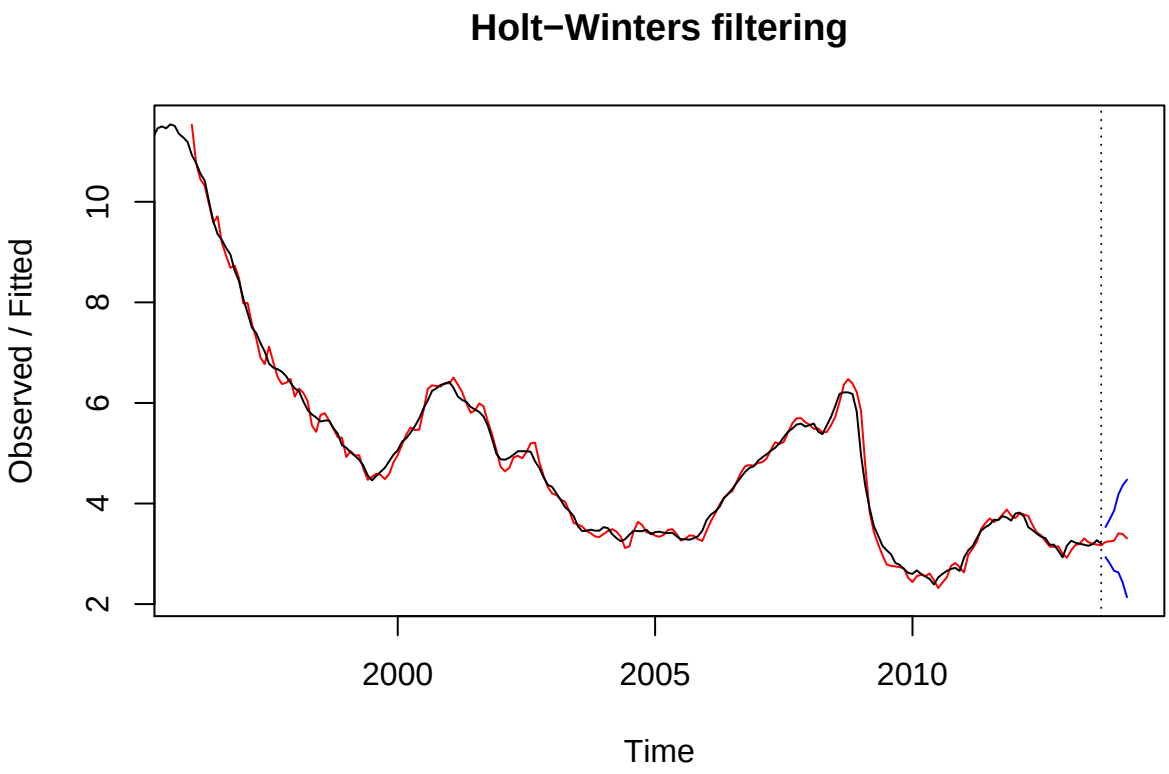
```
smoothing1=HoltWinters(g12data,gamma=FALSE,beta=FALSE)
smoothing2=HoltWinters(g12data,gamma=FALSE)
smoothing3=HoltWinters(g12data)
forecast1=predict(smoothing1, n.ahead=6, prediction.interval=TRUE, level=0.95)
forecast2=predict(smoothing2, n.ahead=6, prediction.interval=TRUE, level=0.95)
forecast3=predict(smoothing3, n.ahead=6, prediction.interval=TRUE, level=0.95)
plot(smoothing1, forecast1)
```




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



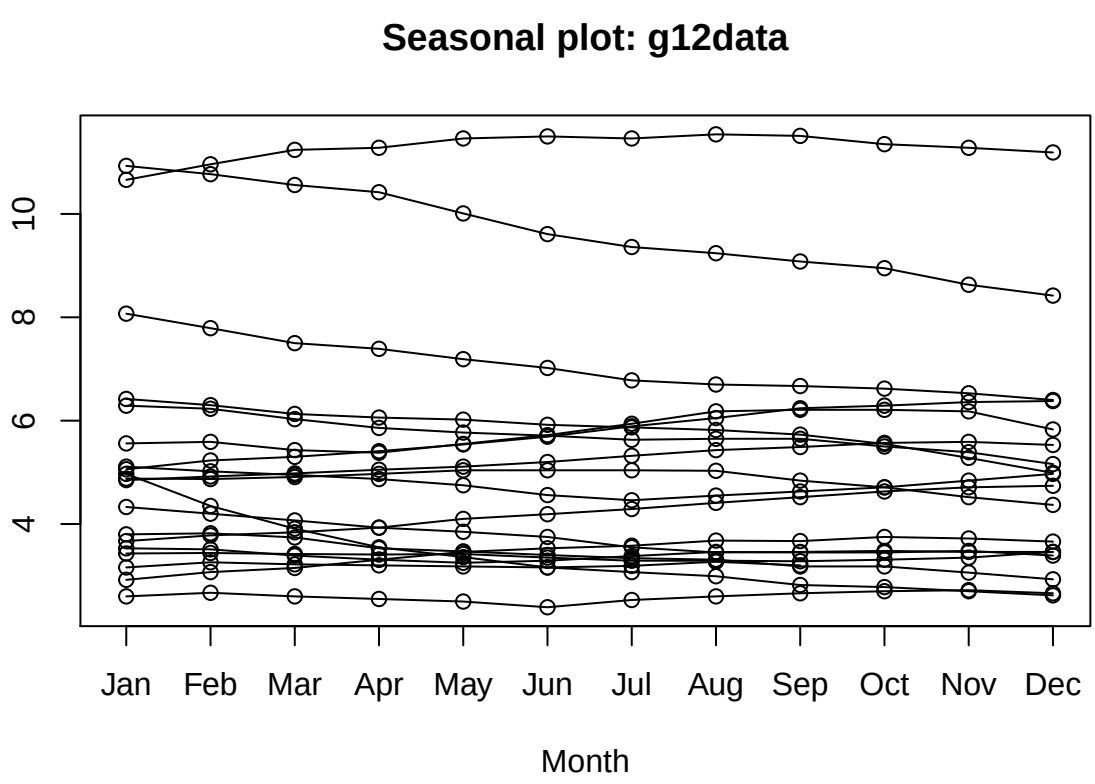

```
plot(smoothing3, forecast3)
```



Seasonality plots

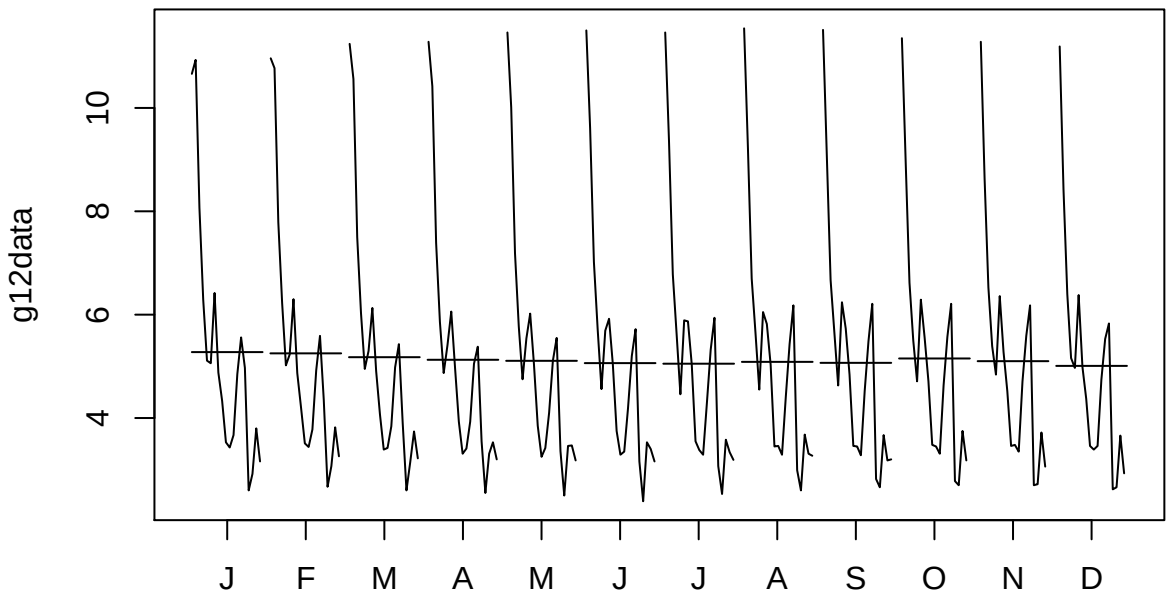
To finish testing our sesonal and trend assumptions we can check the monthplot and sesonplot.

```
par(mfrow=c(1,1))
seasonplot(g12data)
```



In the sesonplot all the months for each year are plotted, so is easy to find a seasonality component. In our case there isn't any clear evidence supporting seasonality.

```
monthplot(g12data)
```

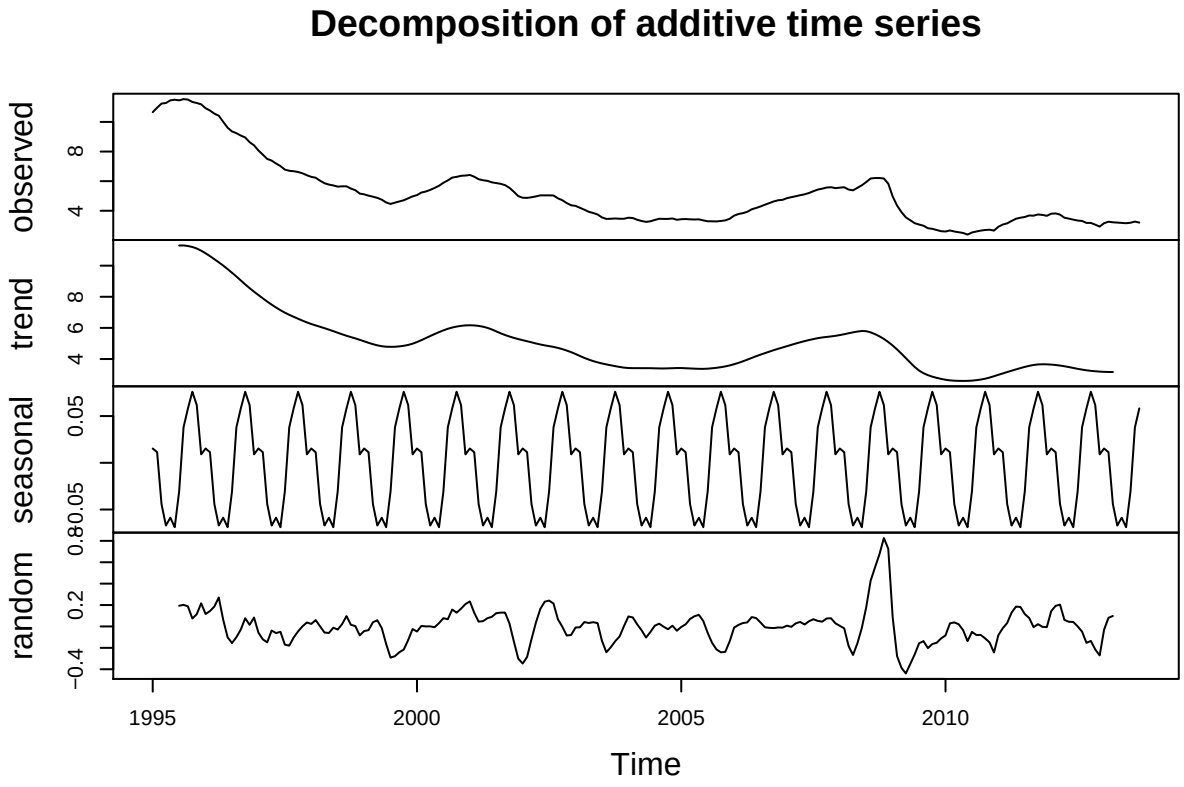


In the monthplot we have subseries for each month of the year with its mean. All the subplots are similiar due to the trend, but there's no change in the mean (which is constant), showing there is no seasonality.

Decomposition

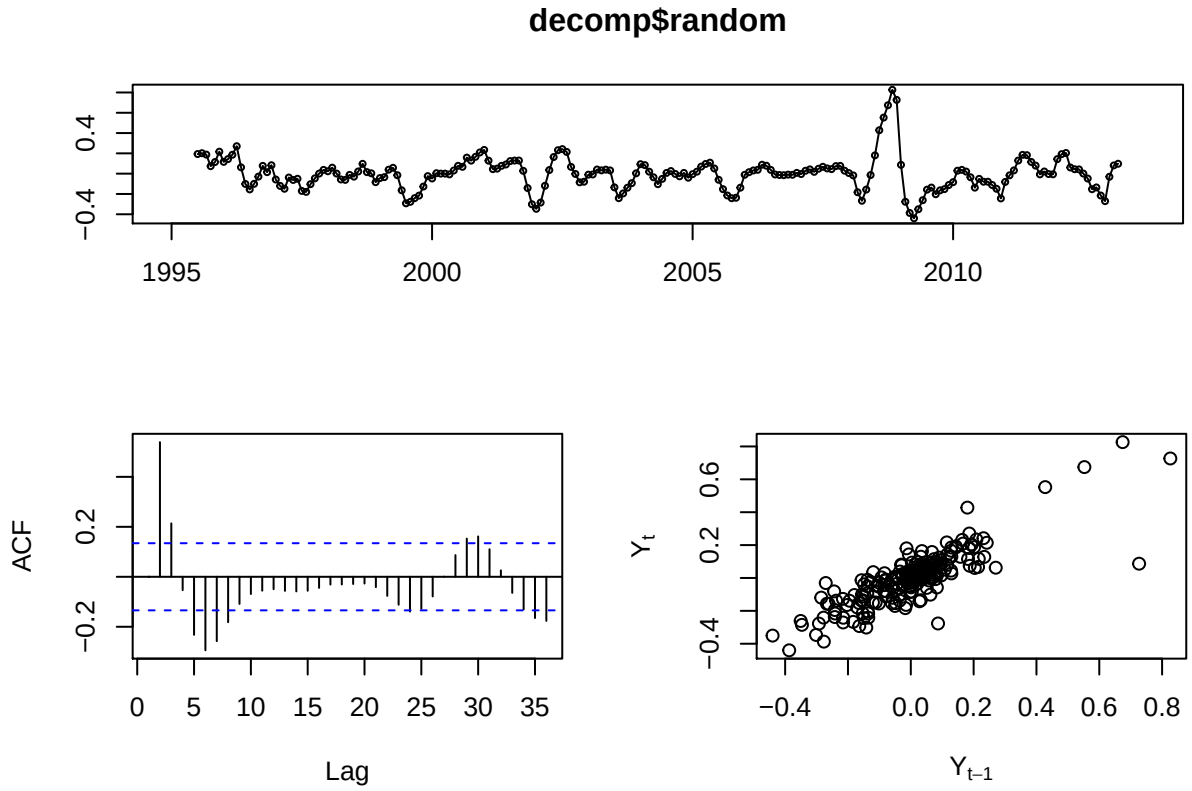
We can obtain a plot of the decomposition of the series. As we don't have seasonality it doesn't make too much sense to use decomposition methods because they assume a seasonality. The multiplicative decomposition is picked over the additive one when there is variance in our series. In our case, as we don't have seasonality we will use the additive model because it is simpler.

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



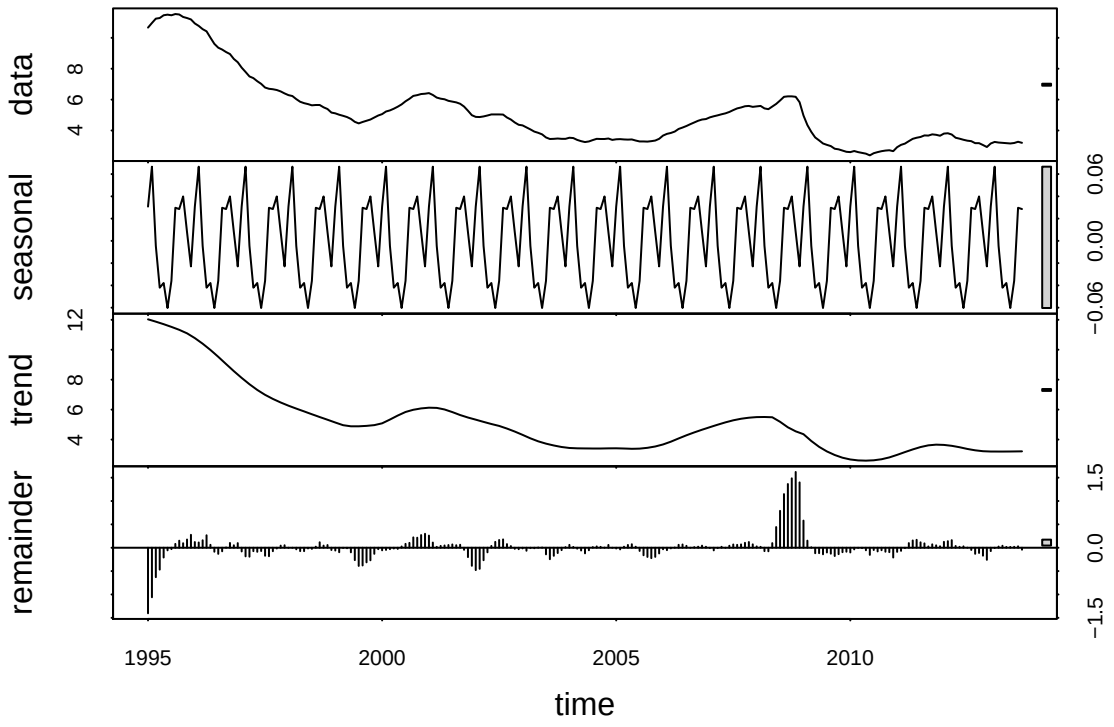
The trend is detected successfully, but the seasonal component is adding noise.

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



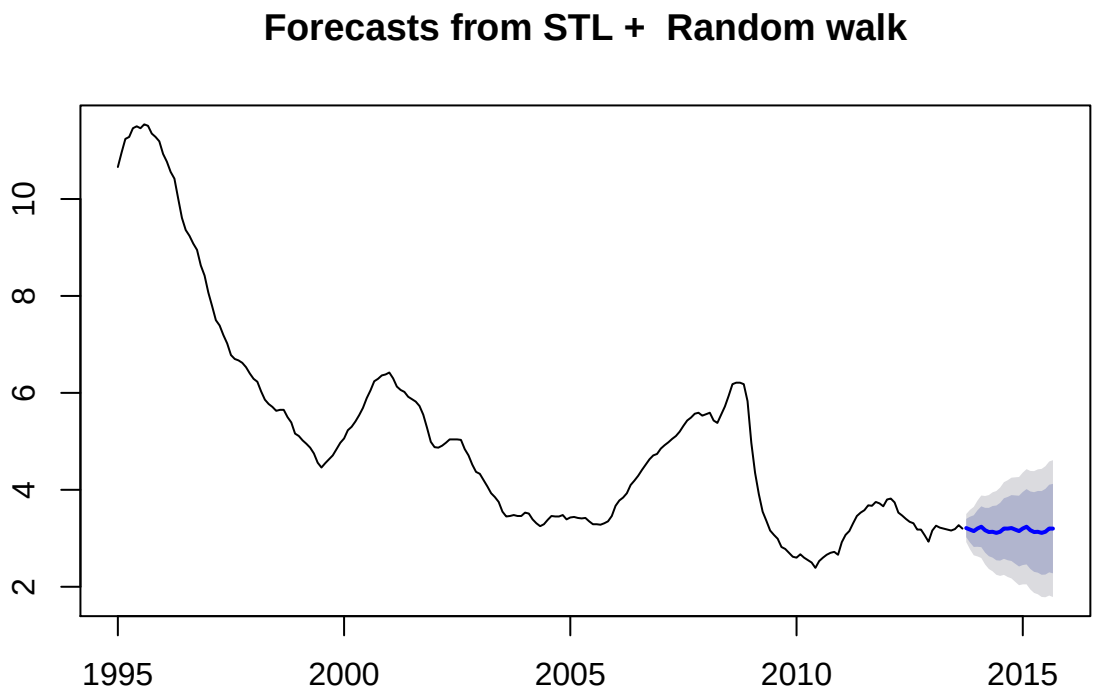
We can see that the lagged scatterplot is still showing correlation and that in the ACF plot the decreasing of the values is not fast enough. However, there has been an imporvement because the trend is taken into account. Let's try to forecast future values with this model. We'll first decompose it using another R function and then forecast two years with this decomposition model.

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



The remainder doesn't look like white noise because there are sinusoidal patterns that tell us there's correlation among the values.

```
fcst1=forecast(stl, method="naive")
plot(fcst1)
```




```
fcst1$mean
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul
## 2013
## 2014 3.202207 3.238077 3.166870 3.129341 3.133552 3.111097 3.135501
## 2015 3.202207 3.238077 3.166870 3.129341 3.133552 3.111097 3.135501
##           Aug      Sep      Oct      Nov      Dec
## 2013                    3.211342 3.179297 3.148463
## 2014 3.201159 3.200000 3.211342 3.179297 3.148463
## 2015 3.201159 3.200000
```

The values predicted for 2014 and 1015 are exactly the same. A better model for our series should be found, so we are going to use an arima model in the next section.

Arima models

We are going to use differentiation to achieve stationarity. In order to see when to stop differentiating we are going to check the standard deviation values.

The best value for the standard deviation is ahieved after differentiation two times (if we go on derivating the sd goes up).

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

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

```
sd(diff(g12data)) #better
```

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

```
sd(diff(diff(g12data))) #Best
```

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

```
sd(diff(diff(diff(g12data,12)))) #Worse
```

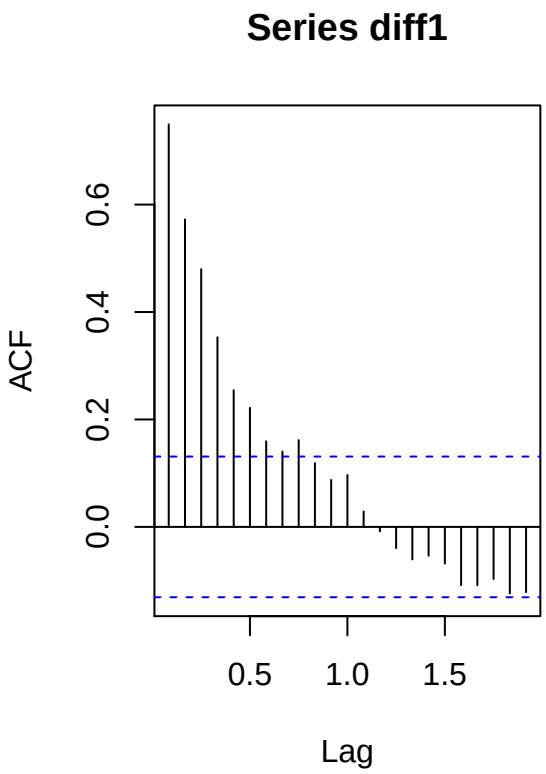
```
## [1] 0.1311037
```

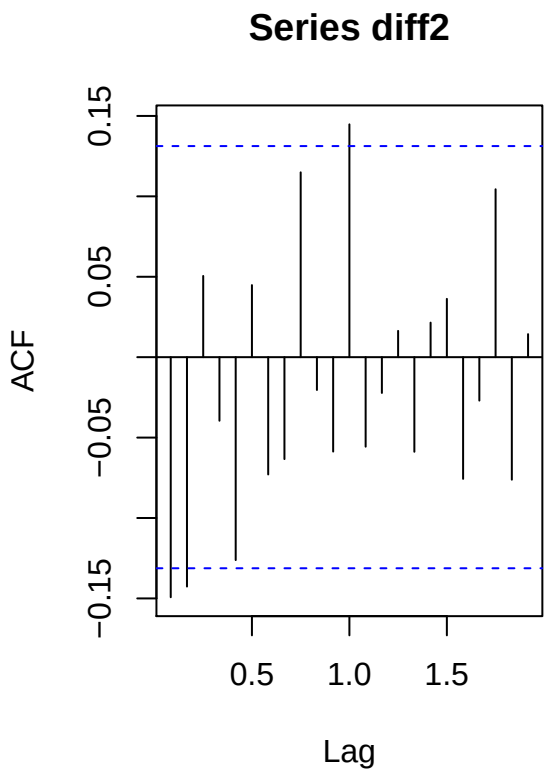
```
sd(diff(diff(diff(g12data)))) #Worse
```

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

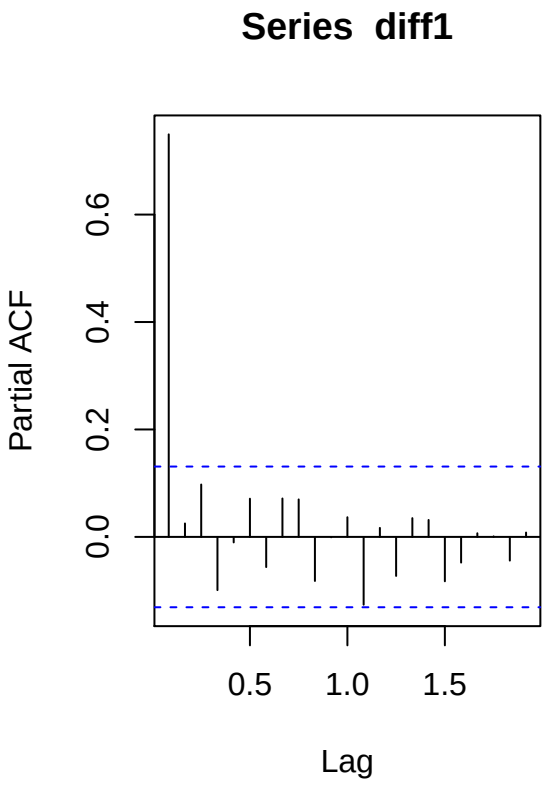
It was obvious that differentiating using seasonality was not going to work, but we tried it anyway to check it. The improvement of the sd from one derivation to two is not really that high, but the complexity of the model incresases. That's why maybe it won't be advisable to use two derivations. Let's compare the plots of them to check the ACF and the partial ACF.

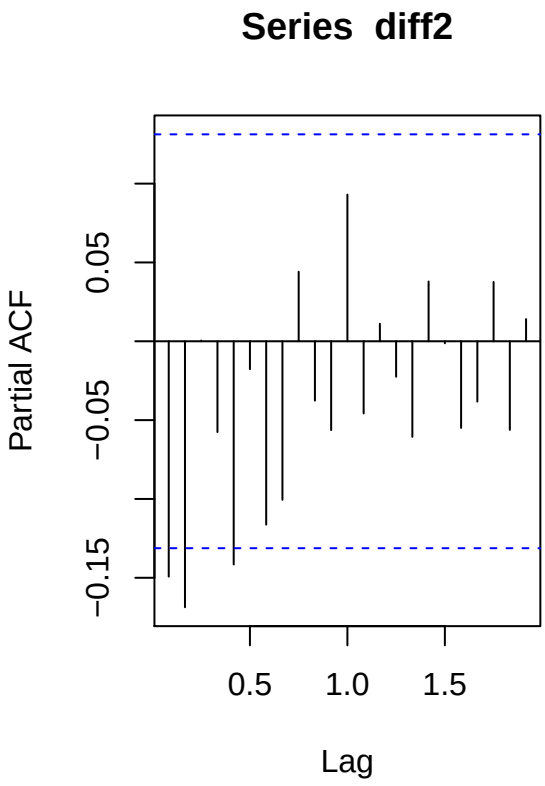
```
diff1=diff(g12data)
diff2=diff(diff(g12data))

par(mfrow=c(1,2))
acf(diff1)
acf(diff2)
```



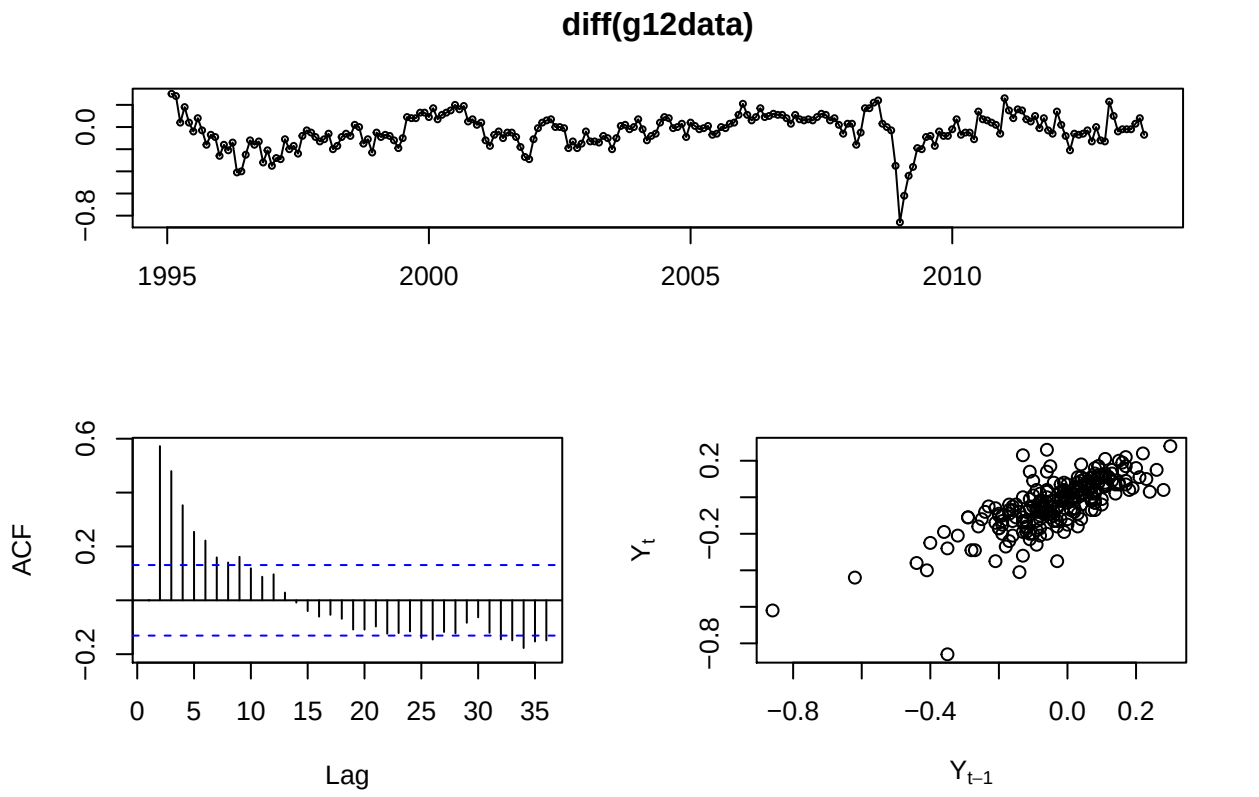



```
par(mfrow=c(1,2))
pacf(diff1)
pacf(diff2)
```

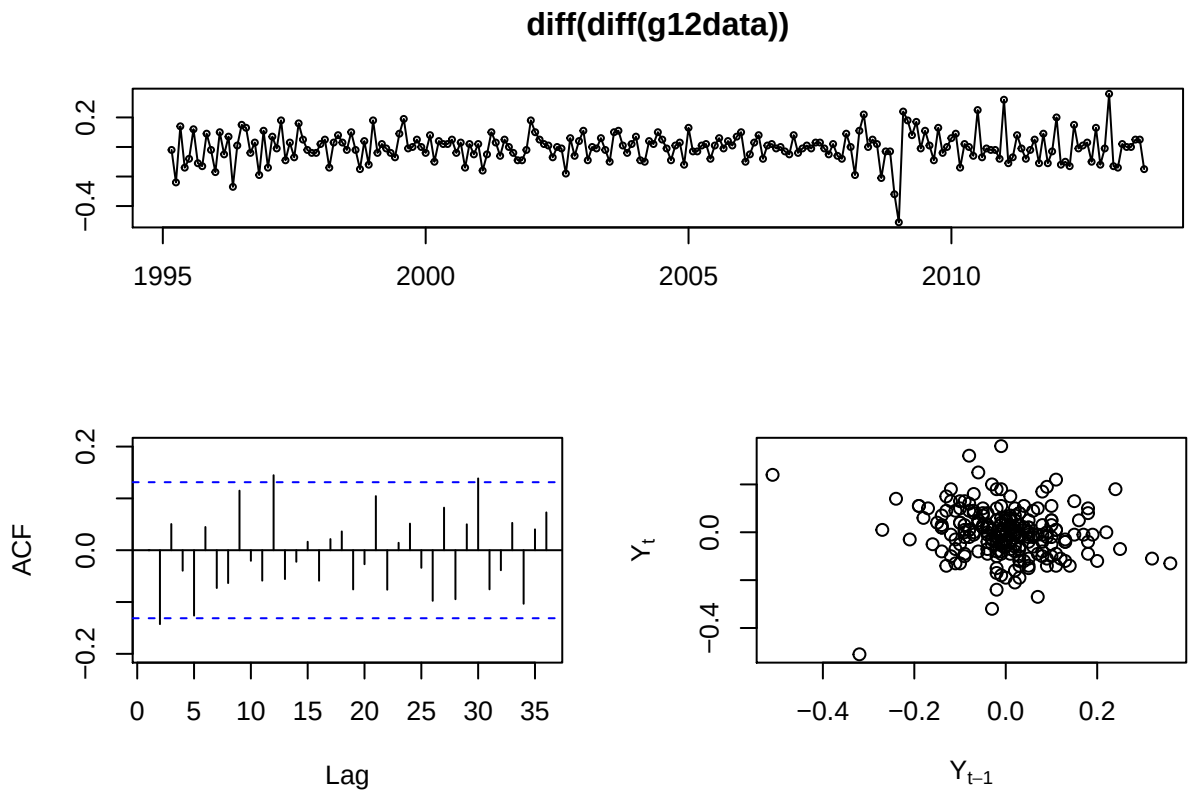





```
tsdisplay(diff(g12data), plot.type="scatter")
```




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



We can observe in the ACF that when differentiating a second time the values are much lower, but with only one differentiation we still have a significant decay and values tend to zero, though not as fast as we would like. In the PACF plot we can see that when differentiating one time there's a spike in the first value, which is a good sign. This can't be observed when differentiation two times, so this might mean that only one differentiation step is needed. Making the model more complex if we don't have significant improvements is not worth it. Still, we are going to try ARIMA models for both values of d to check if this assumption is correct.

The p-value when checking if it is stationary is very low, which indicates our differentiated series have a stationary behaviour.

```
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
```

```
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
```

ARIMA models have 6 parameters that we have to tune. Our model has the following structure ARIMA(p,d,q)(P,D,Q) were the upper case parameters refer to the seasonal component. The meaning of the letters is the following:

- p->past observations
- q->past errors
- d->diferentiation order

We detected earlier that d=1 or d=2, so we will have to try both. As we didn't detect any sesonal pattern and because the sd was worse when seasonal differentiating, we have that the seasonal differentiation order is zero (D=0).

We've tried several different ARIMA models for d=1, D=0 and d=2, D=0. The best results are the following ones.

```
g12arima.1=Arima(g12data,order=c(1,2,1),include.mean=1,seasonal=list(order=c(1,0,1)))
g12arima.2=Arima(g12data,order=c(2,1,0),include.mean=1,seasonal=list(order=c(1,0,1)))
g12arima.3=Arima(g12data,order=c(1,1,0),include.mean=1,seasonal=list(order=c(1,0,0)))
```

We can check how good they are using the AIC measure. The smallest the AIC, the better. The third one is the best one. This one was found using my-automatic.R script from the lab class, which go over all the possible options and selects the better one. We will keep this one because it is also the simplest model.

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

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

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

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

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

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

We can compare our model with the one from the auto arima function.

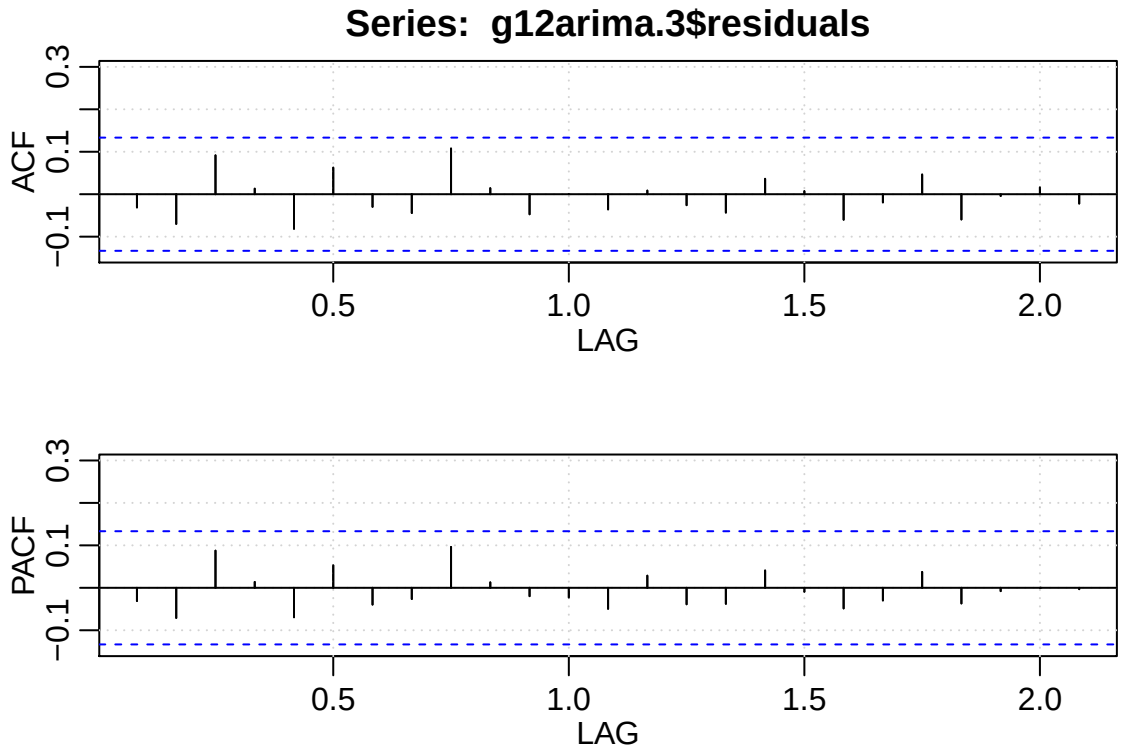
```
auto.arima(g12data)
```

```
## 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
```

All our previous models were better than this one. Also, the model we got using auto.arima is more complex (which is bad).

```
acf2(g12arima.3$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
```

As we can see both the acf and the partial acf are almost zero, indicating that this is a good model for our series.

We still need to check if there are correlation between the coeffients.

```
cov2cor(g12arima.3$var.coef)
```

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

As this value is not greater than 0.8, there are no correlation.

We also need to test the independence of the residuals because we want them to be similar to the residuals from white noise. We are going to apply the LB test

```
LB.test(g12arima.3)
```

```
##
##  Box-Ljung test
##
## data:  residuals from  g12arima.3
## X-squared = 9.8561, df = 10, p-value = 0.4532
```

This test indicates that our residuals appear to be white noise.

We also need to test the normality of the residuals. The Jarque-Bera test gives us a very small p-value, so our residuals apparently are not normal. However, in some of the exmaples in the labs (as in the bubble example) this also happened.

```
jarque.bera.test(residuals(g12arima.3))
```

```
##
##  Jarque Bera Test
##
## data:  residuals(g12arima.3)
## X-squared = 505.13, df = 2, p-value < 2.2e-16
```

We can also check the independence of the time series using the Box-Pierce and Ljung-Box tests. As our value is greater than 0.05 everything seems alright.

```
Box.test(residuals(g12arima.3),lag=12)
```

```
##
##  Box-Pierce test
##
## data:  residuals(g12arima.3)
## X-squared = 9.4269, df = 12, p-value = 0.6661
```

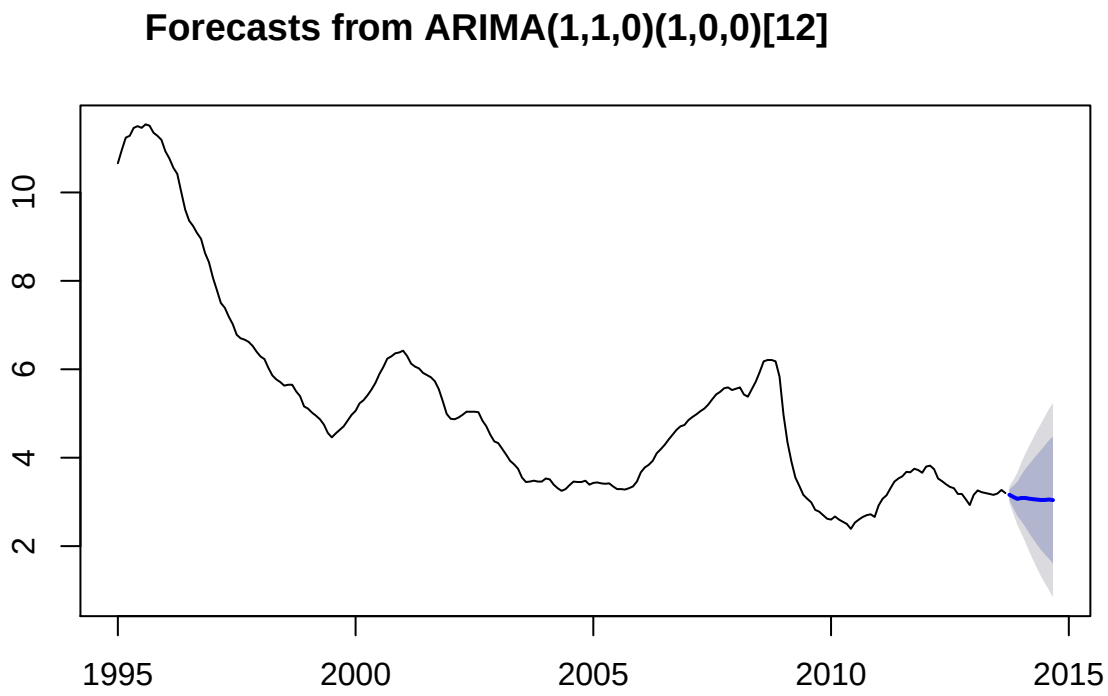
Sometimes the normality of the residuals can be solved taking logarithms. However, as we saw at the beggining of this report in this case taking logs doesn't improve things.

```
g12arima.3log=Arima(log(g12data),order=c(1,1,0),include.mean=1,seasonal=list(order=c(1,0,0)))
jarque.bera.test(residuals(g12arima.3log))
```

```
##
##  Jarque Bera Test
##
## data:  residuals(g12arima.3log)
## X-squared = 612.57, df = 2, p-value < 2.2e-16
```

Let's forecast with our arima model.

```
par(mfrow=c(1,1))
plot(forecast(g12arima.3,h=12))
```



The values of this forecast with their error are:

```
forecast(g12arima.3,h=12)
```

```
##          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
```

Conclusions

We've studied our time series and applied to it the best arima model we could find. One of the problem in our series is the lack of seasonality or periodicity, which are limitations to find pattern from previous values in the model. In a seasonal time series with a clear pattern each year it is easier to predict the values for the next year. However, our series depends on the fluctuation of the economy, which is rather unpredictable.

It is very difficult to find a model that accurately predicts future values. The forecast we got is not very informative and wouldn't be of much use in a more rigorous study.

Summarizing, the forecast we have can be used to predict values close in time, but as we go further the error grows and the predictions are not so accurate. However, as we have the error associated with our predictions and because we checked our assumptions to be correct, we can trust this results and use them accordingly for further studies. Having more data related with this topic would also help to find more patterns and have better predictions.