tseries

Jose Luis Contreras Santos and Antonio Javier González Ferrer

December 21st, 2016

Loading the data

```
library(forecast)
library(astsa)
library(tseries)
library(car)
library(astsa)

visados.df <- read.csv("../data/data_g2.csv", header=T, sep=",", stringsAsFactors = FALSE)
# Create a Time Series object, starting from 1997 and seasonal data.
visados.ts <- ts(visados.df$Visados, start=c(1997,1), end=c(2013, 8), frequency = 12)
```

Part 1

Jose –>

Part 2

Decomposition methods describe the trend and seasonal factors in a time series. We will use this information for understanding the data at hand before building a more sophisticated model, for instance, an ARIMA model.

The Multiplicative decomposition is one of the basic decomposition methods and it is useful when the seasonal variance of the data increases over time. As we will see in section X, such hypothesis will be closer to the structure of the dataset than the simple Additive decomposition. The Multiplicative decomposition method is defined as follows:

$$x_t = T_t * S_t * I_t$$

where x_t is the time series at time t, T_t is the trend, S_t is the seasonality and I_t is the irregular part or remainder. Essentially, the goal is to find a decomposition such that the remainder behaves like white noise, that is to say, the remainder acts as a stationary time series by itself.

There are different ways to model the trend and the seasonality components. R provides the stl() function for seasonal decomposition of time series based on LOESS, a smoothing procedure that is based on local, weighted robust regression. The aim of this procedure is to reduce potentially the disturbing influence of outliers. By default, the command stl() performs an additive decomposition but it can be easily transformed to a multiplicative decomposition taking logarithm of the input data.

```
# Multiplicative decomposition
visados.stl <- stl(log(visados.ts), s.window="periodic")
plot(visados.stl)
```

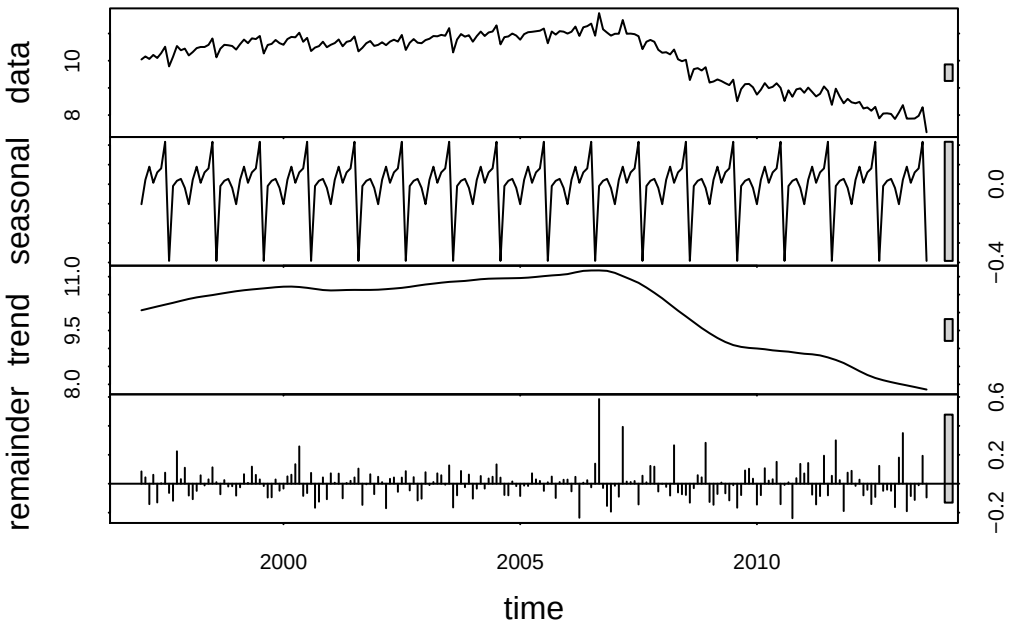


Figure 1: Pedro picapiedra

The second parameter essentially controls the seasonal effects to be estimated in as averages of de-trended values. The plot shows the observed series, the seasonal pattern, the smoothed trend line, and the remainder part of the series. Note that the seasonal pattern is a regularly repeating pattern. The grey bars facilitate the interpretation by covering the same span of the y-scale in all the charts.

Nevertheless, the remainder does not look like white noise. Looking at the ACF plot, some significant autocorrelation coefficients can be distinguished outside the limits. Then, there is still information left in the remainder that should be used in computing forecasts.

```
# Extracting and ploting the remainder part.
visados.stl.remainder <- visados.stl$time.series[,c("remainder")]
tsdisplay(visados.stl.remainder)
```

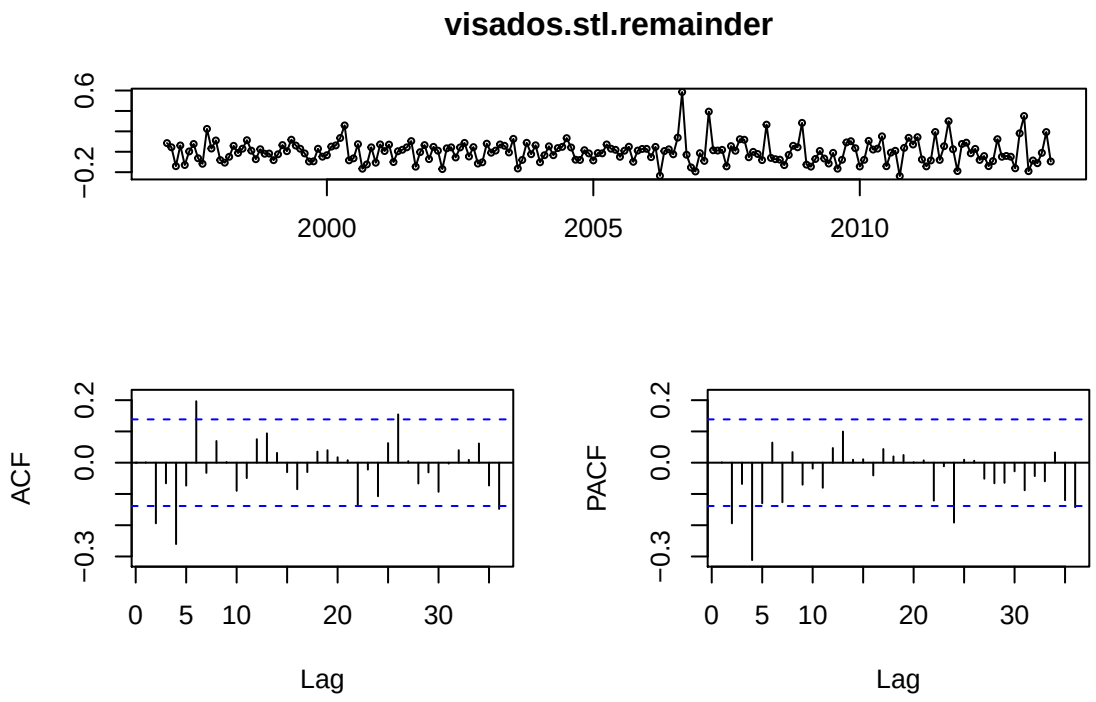


Figure 2: Pedro picapiedra

The further assumption is checked using the Box-Pierce test for examining the null hypothesis of independence in the remainder. The p-value is close to 0 and hence there exist significant autocorrelations for the first 12 lags.

```
# Box-Pierce test for independence of a time series.
Box.test(visados.stl.remainder, lag=12)
```

```
##
##  Box-Pierce test
##
## data:  visados.stl.remainder
## X-squared = 35.136, df = 12, p-value = 0.0004454
```

Let us forecast future values based on the last analysis of the trend and seasonal factors.

```
plot(forecast(visados.stl, method="naive"))
```

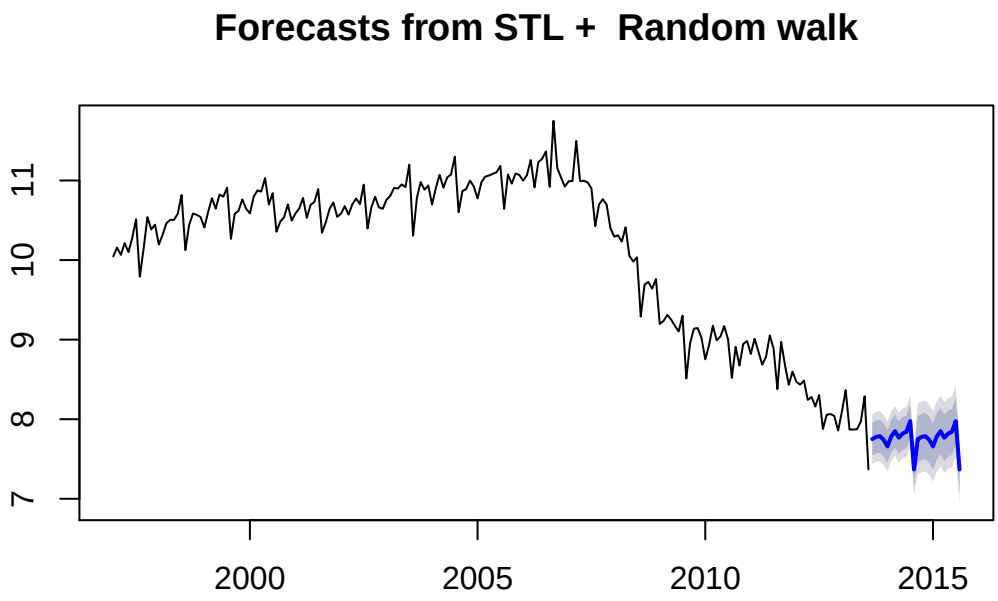


Figure 3: Pedro picapiedra

–>

Part 3

Description

blablabla ARIMA blablbla

Data transformation

First of all, let us decide on whether to use the original variable or to work with the log transformation of it. The log transformation might improve the stationary of the time series. We will select the latter option if the amplitude of the seasional changes increases with the overall trend, that is, if the variance is not constant across the series. In the following figure you can observe that the shape of the log plot is smoother and its variance is more stable than the original variable. Therefore we will use the log of the house sales variable from now on. Notice that the Box Cox transformation suggests to use a lambda close to 0 too.

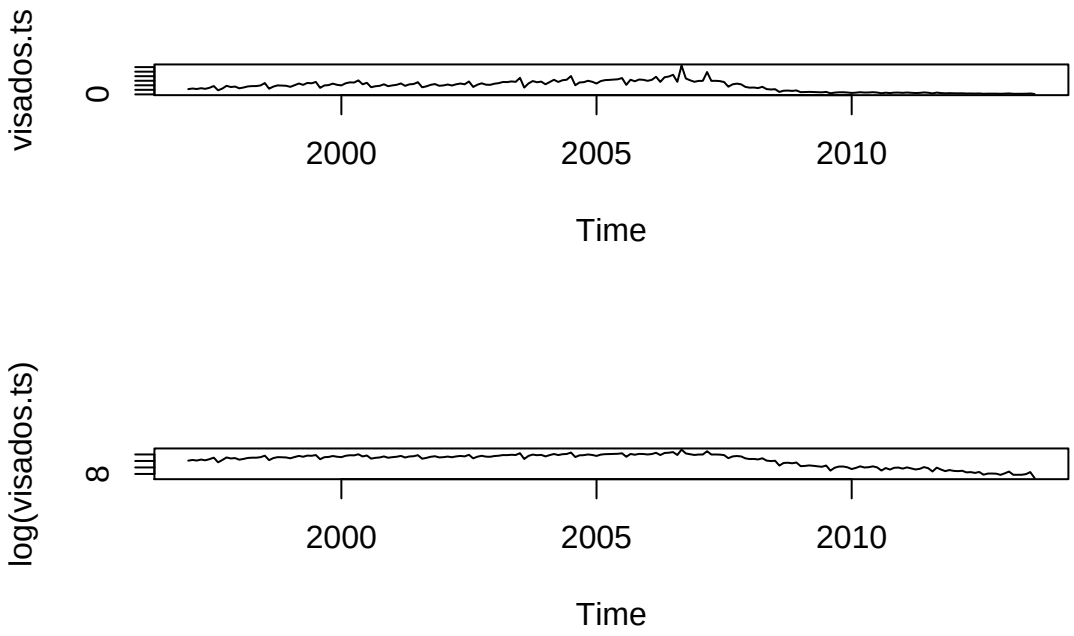


Figure 4: Pedro picapiedra

Seasonality

The nature of the dataset leads us to set an anual seasonal component, $s = 12$. Analyzing the periodogram of Figure 5, we can clearly observe a high peak at point 0 of the x-axis. This peak is related to a long repeated cycle over the time, in our case, a 12-monthly seasonality. On the other hand, the seasonplot() and monthplot() functions might help us identifying this seasionality. Firstly, the season plot shows how the ups and downs are similar over the years. Secondly, the month plot reveals a strong seasonality pattern in August, where the sales decay. This drop should be adjusted into the model if we would like to see whether there is a real trend if the house sales go down in August.

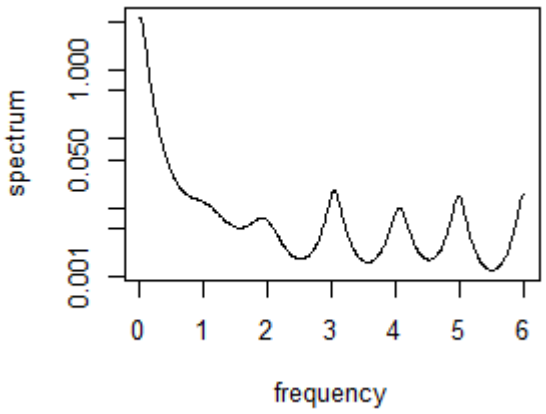


Figure 5: Periodogram of log(visados.ts)

```
par(mfrow=c(1,2))
seasonplot(log(visados.ts))
monthplot(log(visados.ts))
```

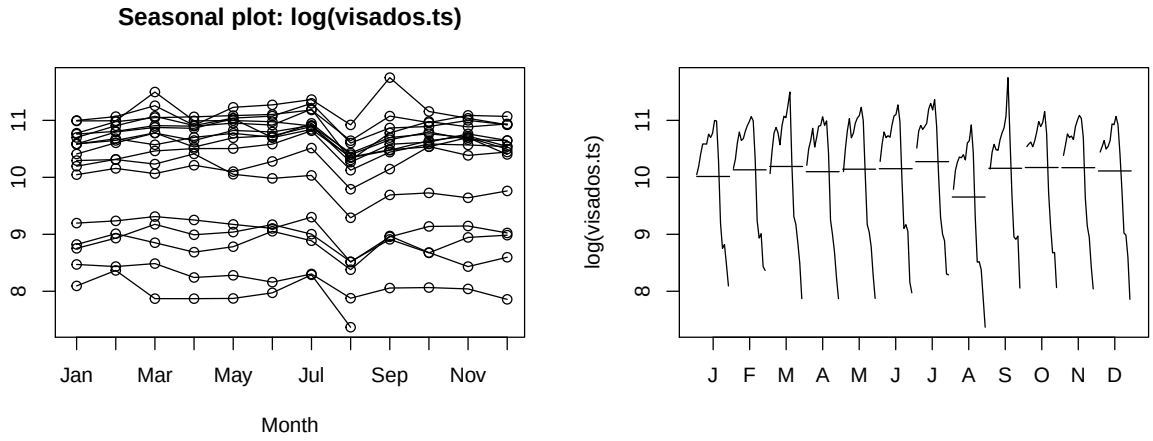


Figure 6: Pedro picapiedra

Differencing

The next step in fitting an ARIMA model is to detect the order of differencing needed to make the time series stationary. We will employ the adf.test to check stationarity and the standard deviation values to set the stop condition (the lower, the better.)

Let us suppose that we omit the seasonality analysis from the last section. Since the original ACF plot (Figure X) has positive autocorrelations out to a high number of lags, we will apply one order of diferencing.

```
# Initial standard deviation
sd(log(visados.ts))
```

```
## [1] 1.006753
```

```
# tsdisplay(log(visados.ts)) #this plot should already be in the report.

# One order of differencing.
sd(diff(log(visados.ts)))
```

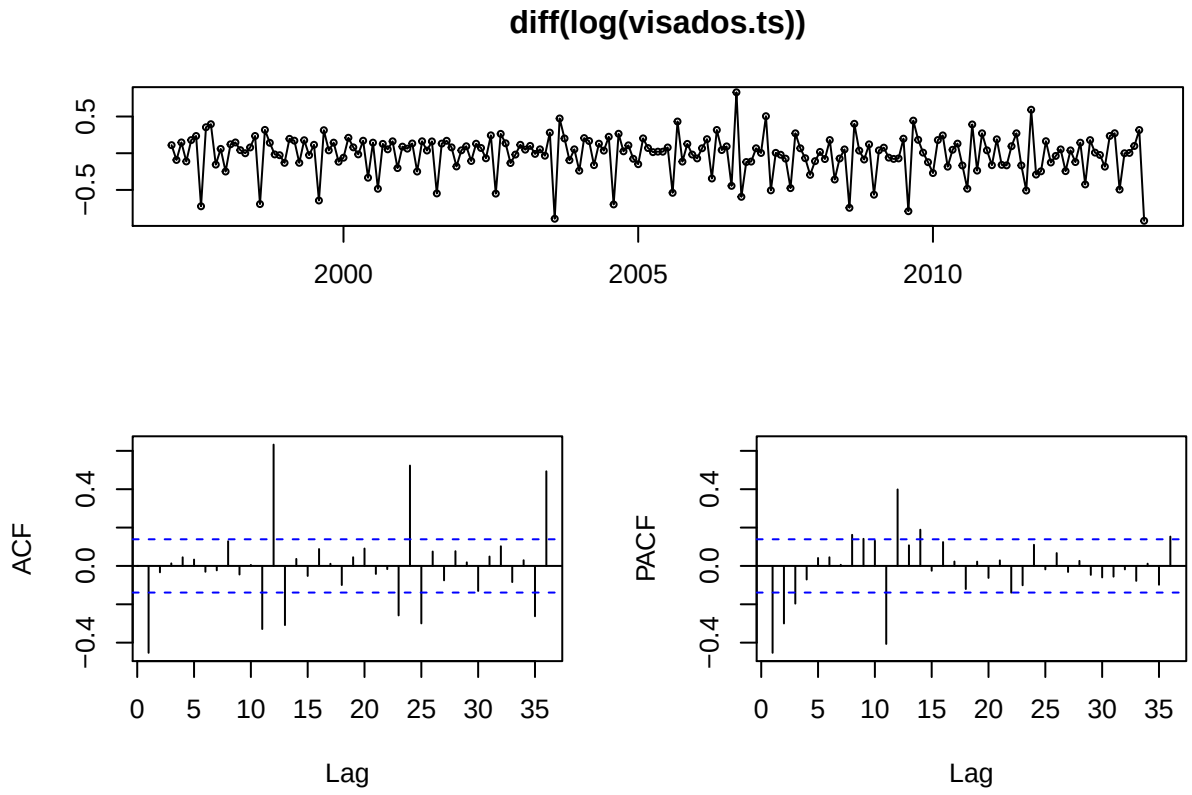
```
## [1] 0.2719039
```

```
adf.test(diff(log(visados.ts)))
```

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

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff(log(visados.ts))
## Dickey-Fuller = -7.5577, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

```
tsdisplay(diff(log(visados.ts)))
```



We achieve stationarity due to we have evidence to reject the null hypothesis in favor of the alternative hypothesis of stationarity in the Augmented Dicket-Fuller test. However, notice the high peaks of autocorrelations in the lags 12, 24 and 36. This is a clear evidence of monthly seasonality. Therefore, we omit this first step and we apply directly a first seasonal difference in order to see if the time series is described by a seasonal random walk.

```
# One order of seasonal differencing.
sd(diff(log(visados.ts), 12))
```

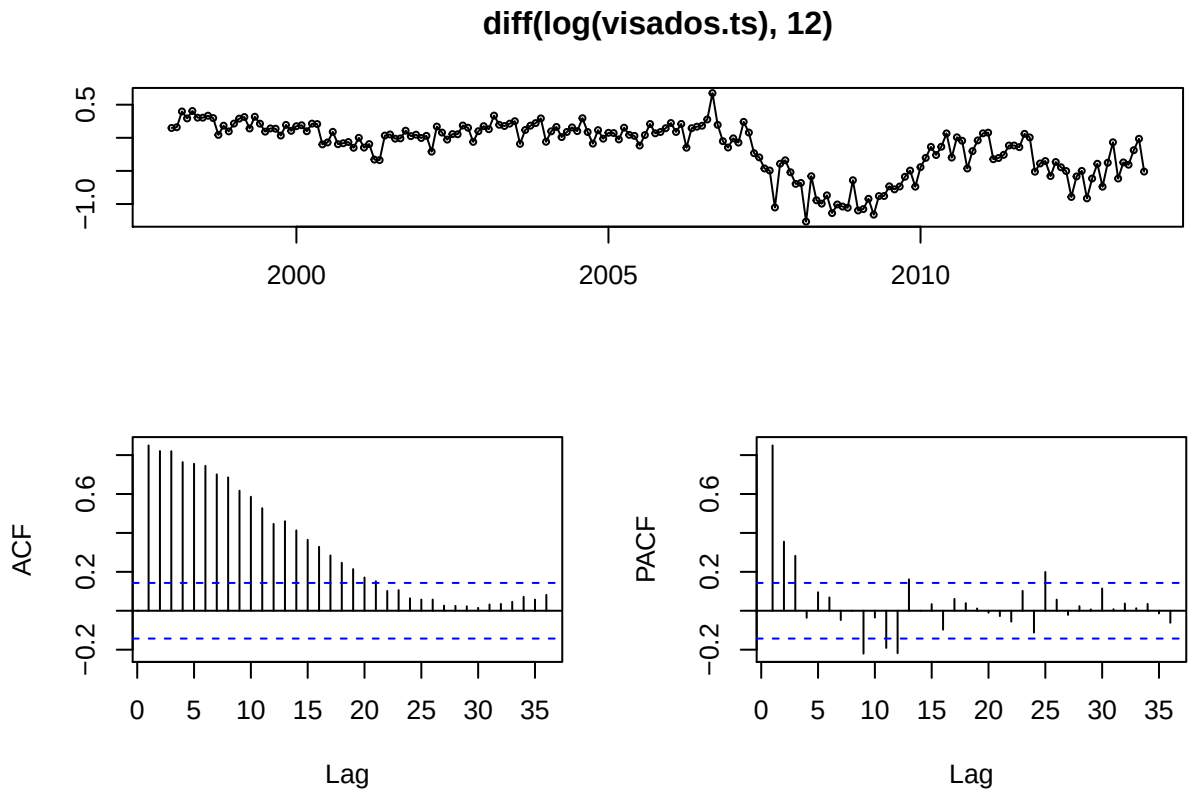
```
## [1] 0.39003
```

```
# Checking stationarity
adf.test(diff(log(visados.ts), 12))
```

```
##
##  Augmented Dickey-Fuller Test
```

```
##
## data:  diff(log(visados.ts), 12)
## Dickey-Fuller = -1.8166, Lag order = 5, p-value = 0.6529
## alternative hypothesis: stationary
```

```
tsdisplay(diff(log(visados.ts), 12))
```



A seasonal random walk is defined as $\hat{Y}_t = Y_{t-12} + \mu$, where μ is the average annnual trend. In this case, there are not evidence to reject the null hypothesis in the adf.test() and the ACF plot still has many positive autocorrelations. As we have already seen, this is an evidence to apply one order of non-seasonal difference.

```
# One order of non seasonal differencing and one order of seasonal differencing.
sd(diff(diff(log(visados.ts), 12)))
```

```
## [1] 0.2116118
```

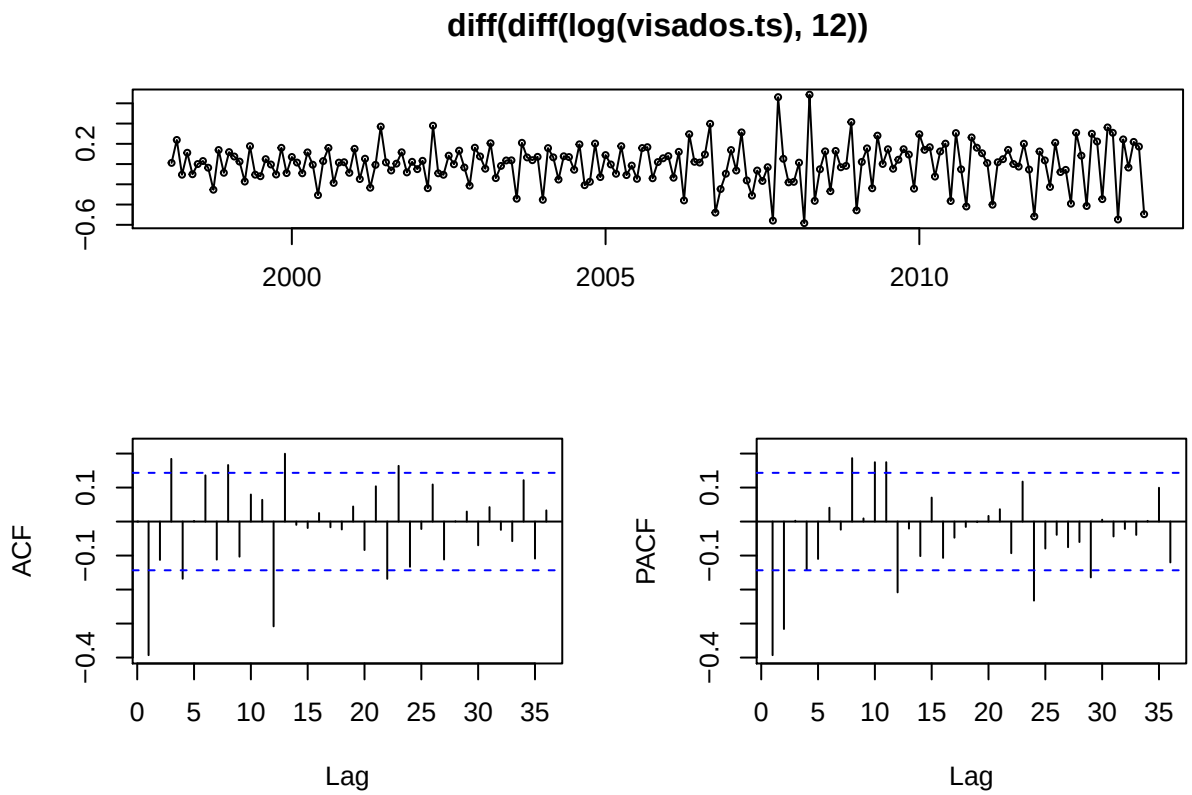
```
adf.test(diff(diff(log(visados.ts), 12)))
```

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

##
##  Augmented Dickey-Fuller Test
##
## data:  diff(diff(log(visados.ts), 12))
```

```
## Dickey-Fuller = -6.8037, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

```
tsdisplay(diff(diff(log(visados.ts), 12)))
```



We obtain stationarity again, with a lower standard deviation and the peaks in the lags 12 and 24 are considerably smaller. The times series is now modeled as a seasonal random trend. Comparing this model with the seasonal random walk, they both predict that next year's seasonal cycle will have the same pattern. In contrast, the seasonal random trend considers that the future trend will be equal to the most recent year-to-year trend, instead of the average year-to-year trend (μ in the model). The seasonal random trend is defined as $\hat{Y}_t = Y_{t-12} + Y_{t-1} - Yt - 13$, which is equivalent to an $ARIMA(0,1,0)(0,1,0)_{12}$ model.

To conclude, we stop here since another order of differencing does not improve the standard deviation. On the other hand, notice

```
sd(diff(diff(diff(log(visados.ts), 12))))
```

```
## [1] 0.3522928
```

Let us study what values of differencing the functions ndiffs and ndsdiff suggest. These results should be taken into account with a grain of salt, and just use them to support our analysis.

```
ndiffs(log(visados.ts), test="kpss")
```

```
## [1] 2
```

```r
ndiffs(log(visados.ts), test="adf")
```

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

```r
nsdiffs(log(visados.ts), m=12)
```

```
## [1] 0
```

Surprisingly, depending on the test we use to estimate the number of differences required to make the time series stationary, we obtain a different value. This extra differencing order is a particular case where the KPPS test detects that the first order difference of the time series is still not stationary but trend-stationary, and another order of differencing is needed. In contrast, the ADF test just rejects the null hypothesis of present of a unit root. On the other hand, nsdiffs indicates the need of non order of seasonal differencing.