

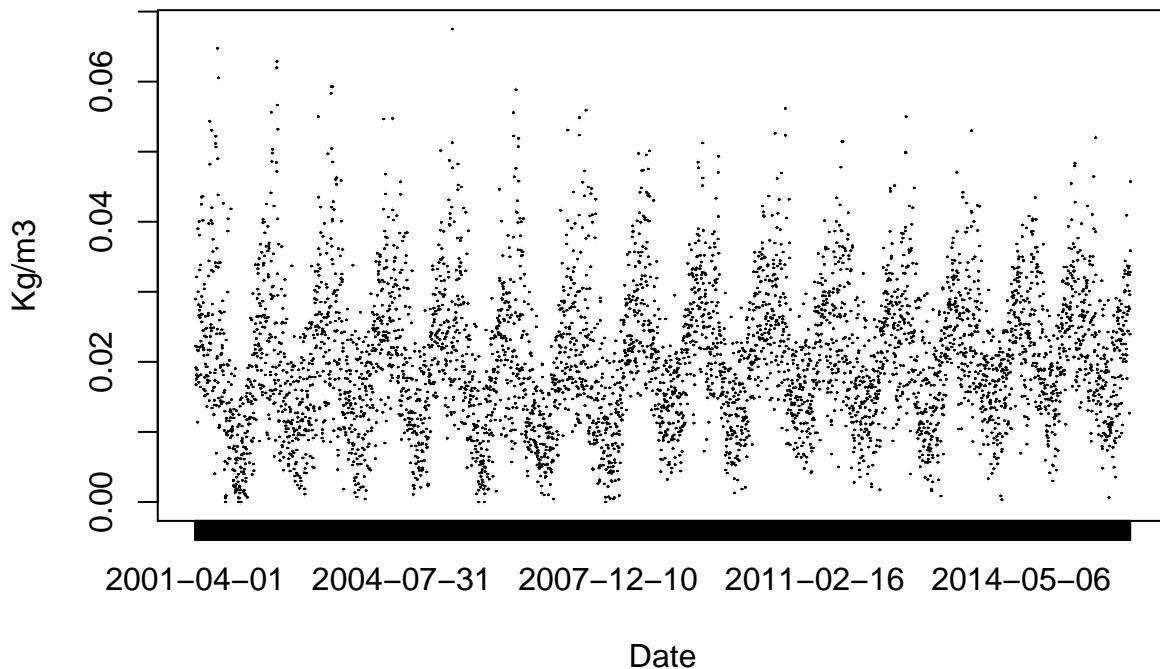
Prediction of ozone level in Boston

Lucas Emanuel Resck Domingues*

Lucas Machado Moschen†

Load and visualize

Daily average level of O₃ in Boston



Data treatment

We noticed that some days do not exist in the dataset, for example, the day August 31, 2001 does not have information in the dataset.

```
##      X City      State Site.Num Date.Local 03.Mean
## 148 148 Boston Massachusetts     42 2001-08-28 0.024583
## 149 149 Boston Massachusetts     42 2001-08-29 0.015000
## 150 150 Boston Massachusetts     42 2001-08-30 0.022333
## 151 151 Boston Massachusetts     42 2001-09-01 0.021958
## 152 152 Boston Massachusetts     42 2001-09-02 0.018750
## 153 153 Boston Massachusetts     42 2001-09-03 0.028708
```

Also, there is duplicated days, as June 9, 2002:

```
##      X City      State Site.Num Date.Local 03.Mean
```

*Escola de Matemática Aplicada

†Escola de Matemática Aplicada

```

## 412 412 Boston Massachusetts      42 2002-06-08 0.022917
## 413 413 Boston Massachusetts      42 2002-06-09 0.036190
## 414 414 Boston Massachusetts      42 2002-06-09 0.037000
## 415 415 Boston Massachusetts      42 2002-06-10 0.023389

```

The duplicated one is easier to deal, but the missing values are harder. First we calculate the mean value between the duplicated.

The rate of missing values is almost 5% of the dataset.

```
## [1] 0.04453367
```

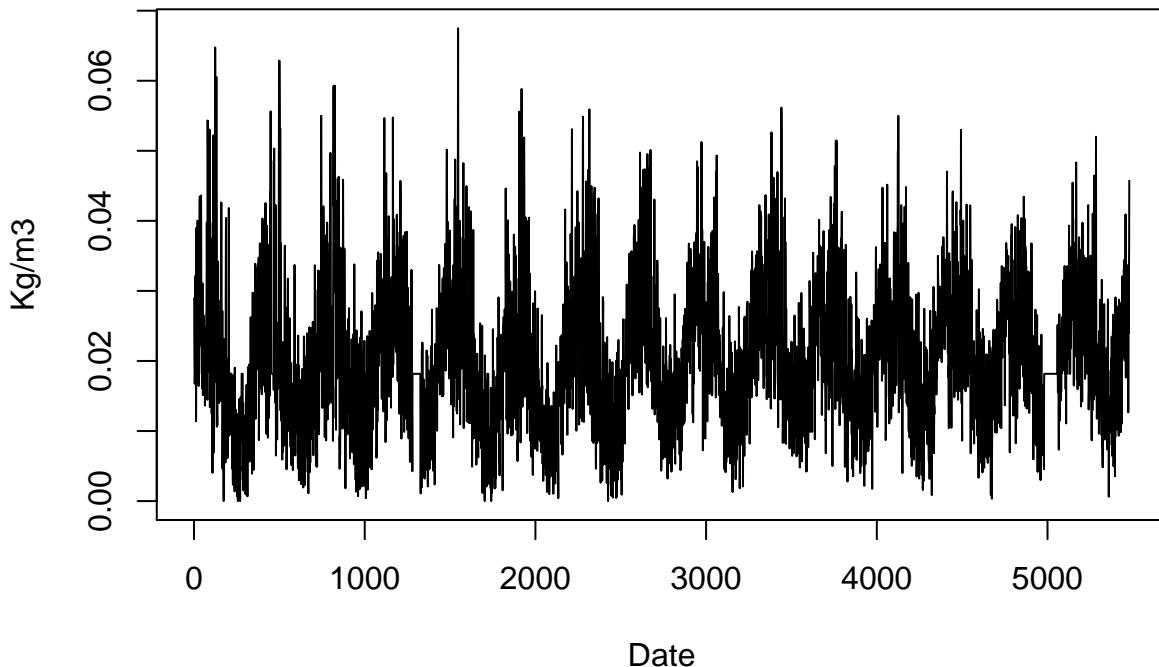
So as to solve that problem, we make a knn imputation using the month ($k = 30$)

```

o3.clean <- knn.impute(as.matrix(o3.ts), k = 30)
o3.clean <- as.ts(o3.clean)
plot(o3.clean, main = 'Daily average level of O3 in Boston (after imputation)',
      xlab = 'Date', ylab = 'Kg/m3')

```

Daily average level of O3 in Boston (after imputation)



We also separate our data between training and test because of modelling best practices.

```

o3_train = o3.clean[1:(length(o3)[1] - 365),]
o3_test = o3.clean[(length(o3)[1] - 365 + 1):length(o3)[1],]

```

Models: case 1

Now we develop some models using the train data.

The metric to compare is the Mean Absolute Error (MAE) in the predictions:

```

mae <- function(ytrue, ypred)
{
  return(mean(abs(ytrue - ypred)))
}

```

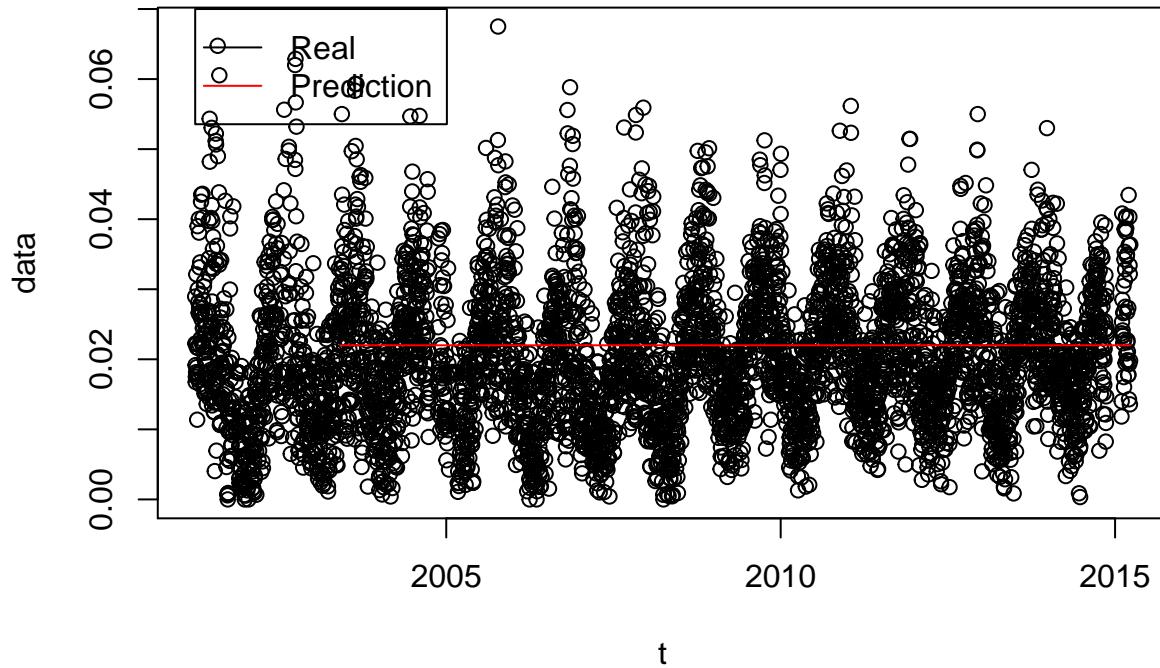
We will use `rollapply` in order to calculate the error, considering the last two years to predict one week forward.

Baseline Model

We will do the naive forecast to the baseline model.

```
## [1] 0.008015428
```

Baseline model prediction



Decompose

First of all we make a seasonality test using Kruskal-Wallis. Actually it tests whether samples originate from the same distribution. We can organize it to be samples for each corresponding day. We compare two different frequencies: monthly and yearly. The second one showed the smallest p-value, in particular less than 0.05. For that reason, we will use 365 as the seasonality.

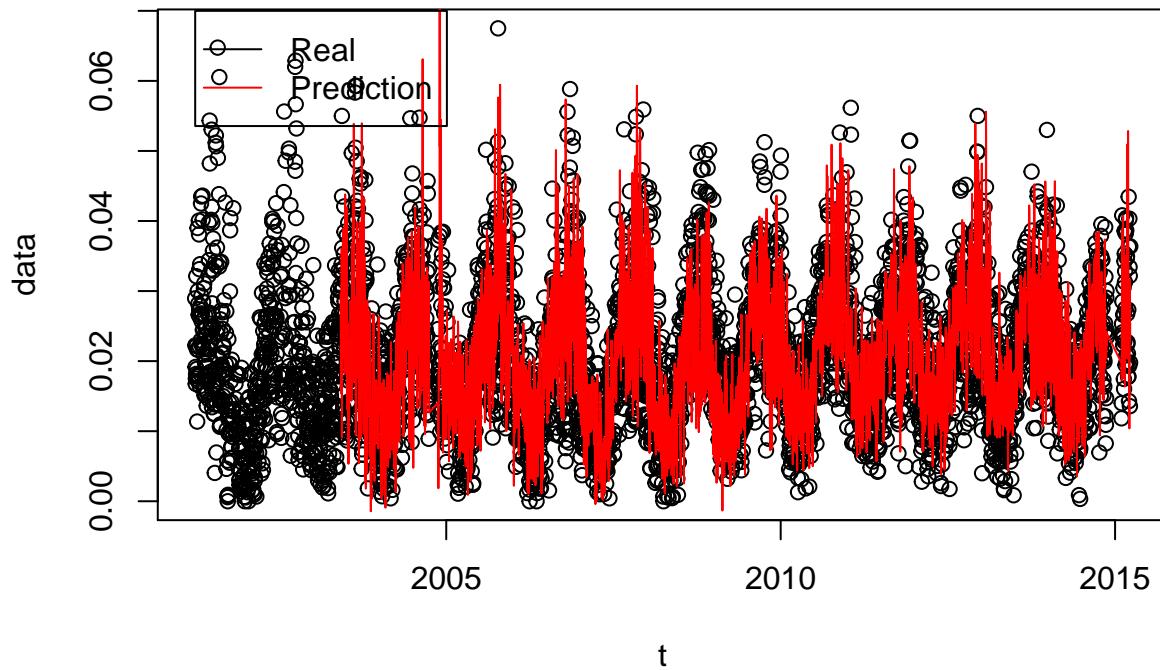
```
## 
## Kruskal-Wallis rank sum test
## 
## data: o3_train and g
## Kruskal-Wallis chi-squared = 32.983, df = 30, p-value = 0.3233
## 
## Kruskal-Wallis rank sum test
## 
## data: o3_train and g
## Kruskal-Wallis chi-squared = 2122.9, df = 364, p-value < 2.2e-16
```

Additive model

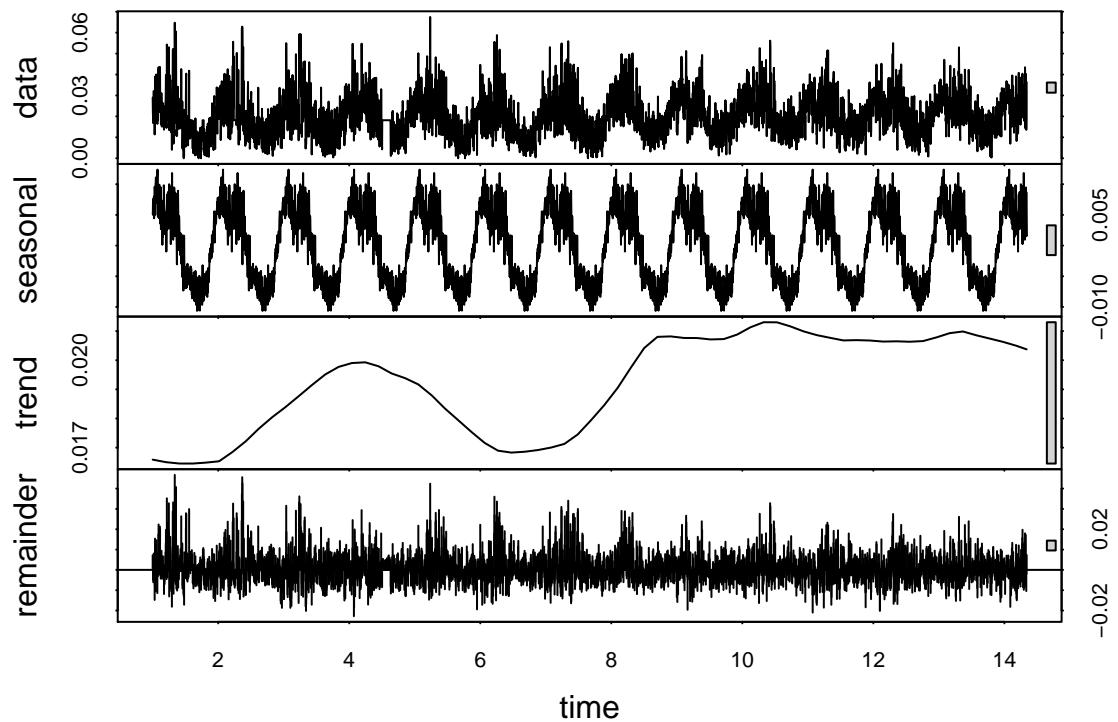
First we analyse the MAE.

```
## [1] 0.007963981
```

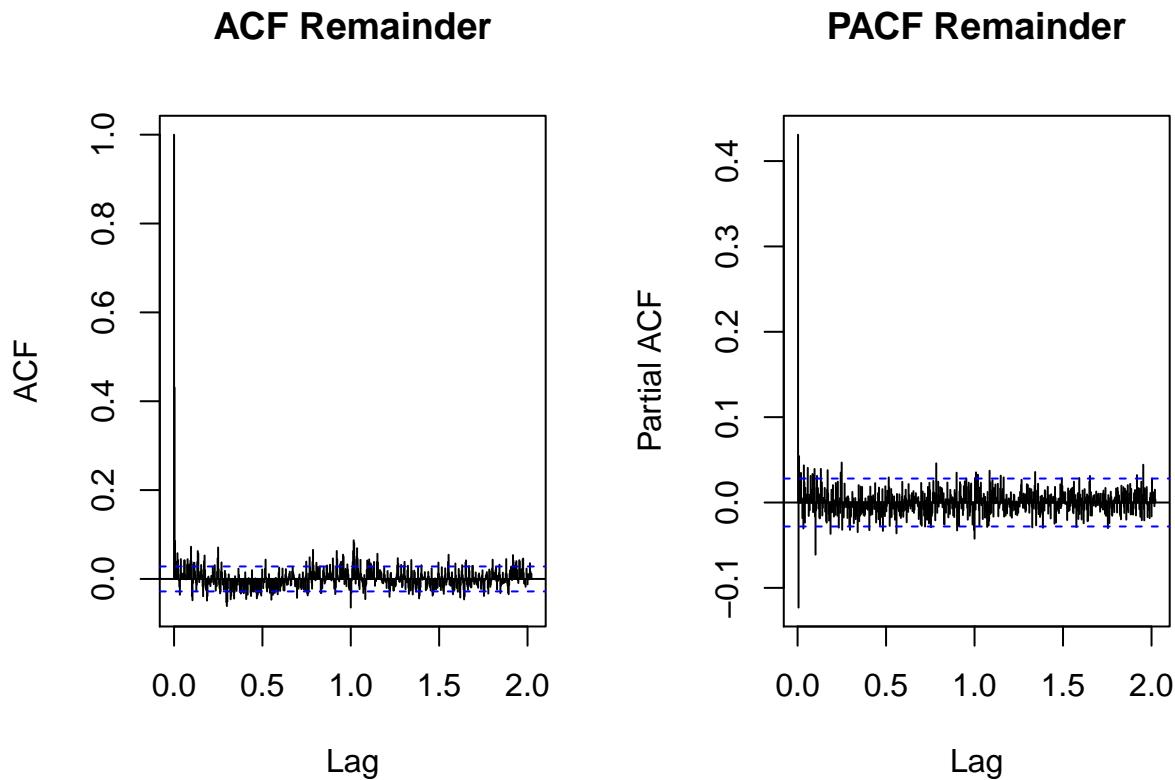
Additive decompose prediction



We also can fit the model using `t.window` and analyse the reminder of the method.



The ACF and the PACF of the reminder:



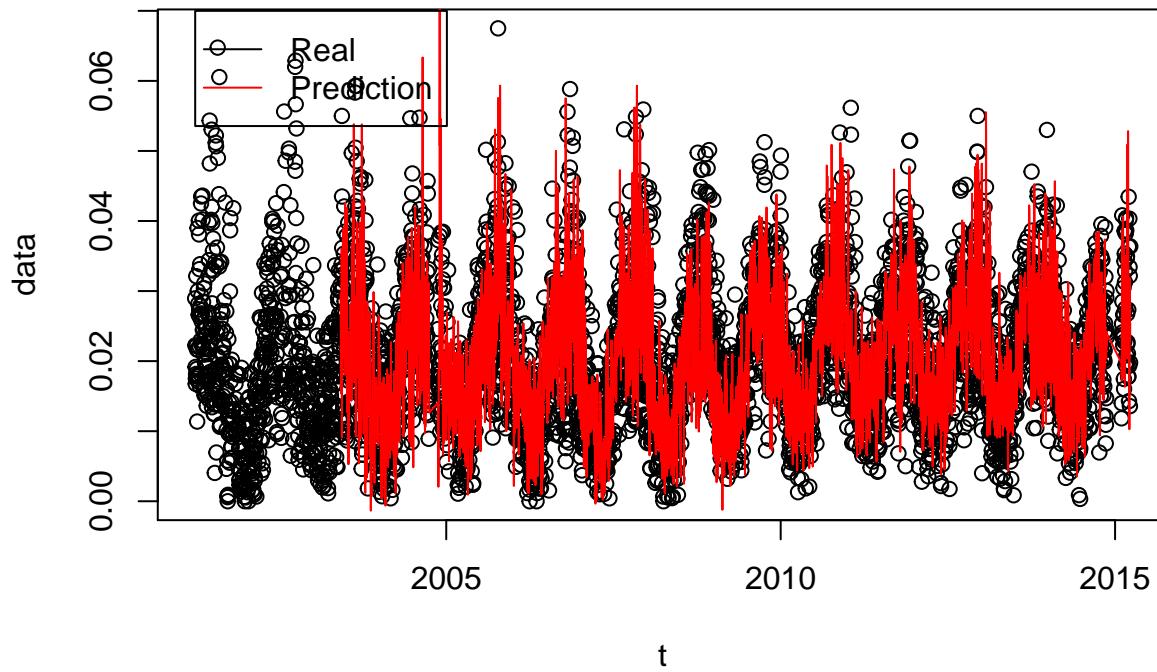
We see that there are a big spike when lag = 365. It seems not so good for a reminder. We could fit an ARMA model in this reminder yet.

Multiplicative model

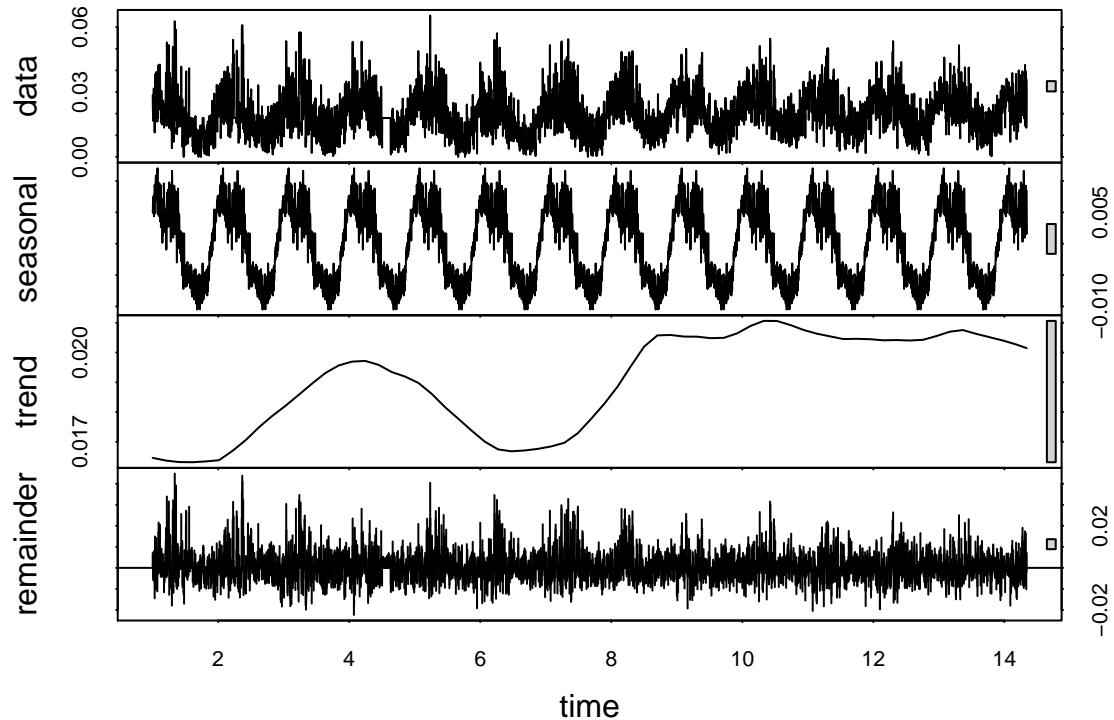
First we analyse the MAE.

```
## [1] 0.00795386
```

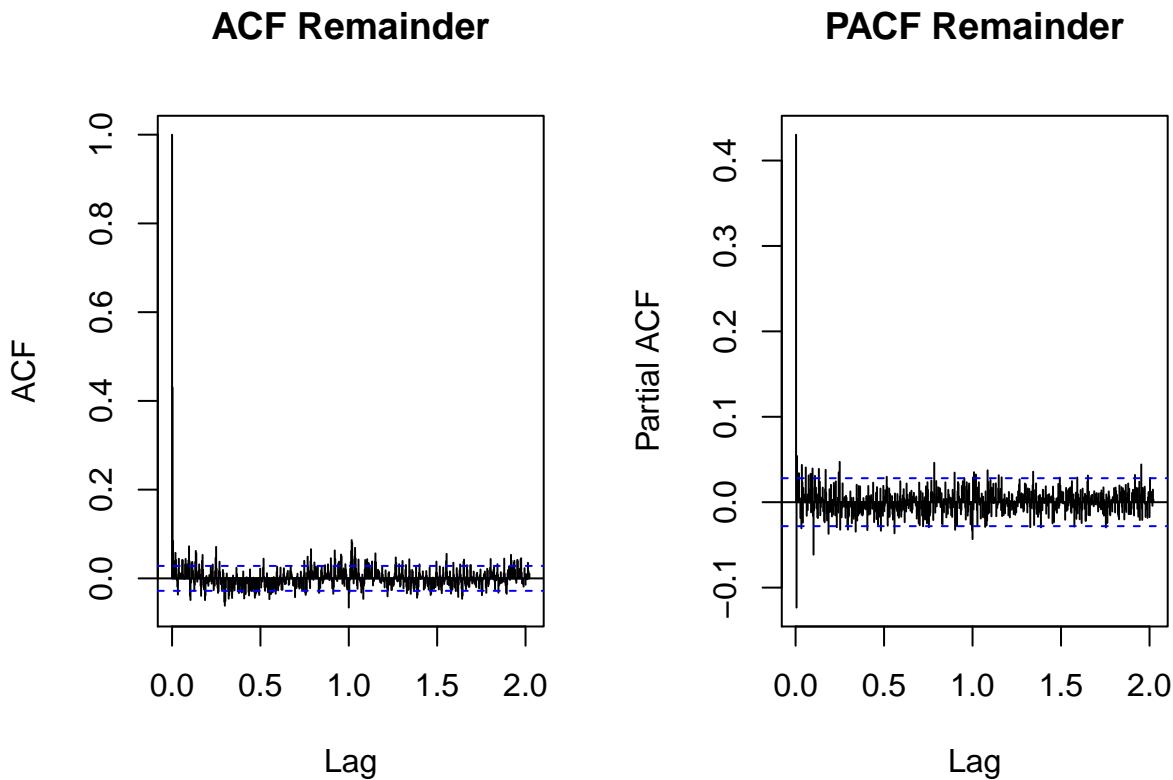
Multiplicative decompose prediction



We also can fit the model using `t.window` and analyse the reminder of the method.



The ACF and the PACF of the reminder:



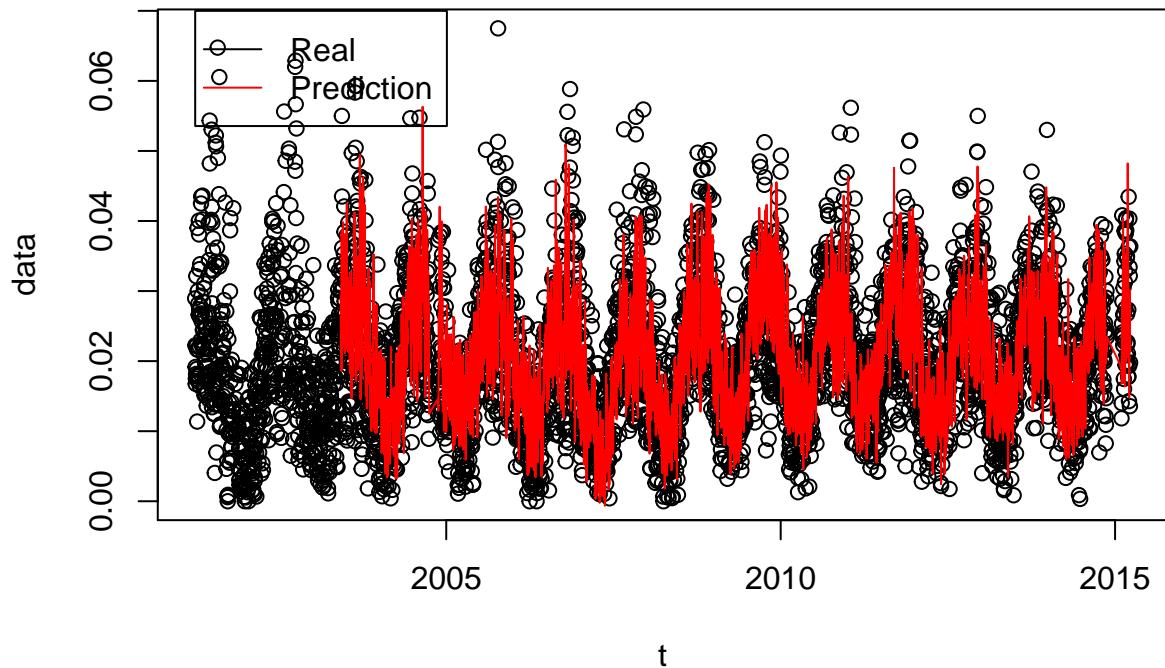
We see that there are a big spike when lag = 365. It seems not so good for a reminder. We could fit an ARMA model in this reminder yet. The same problem as before.

Regression

We tested for seasonalities, and we settled with 365. So we will fit a regression model with the seasonality dummies. We see the MAE:

```
## [1] 0.007518997
```

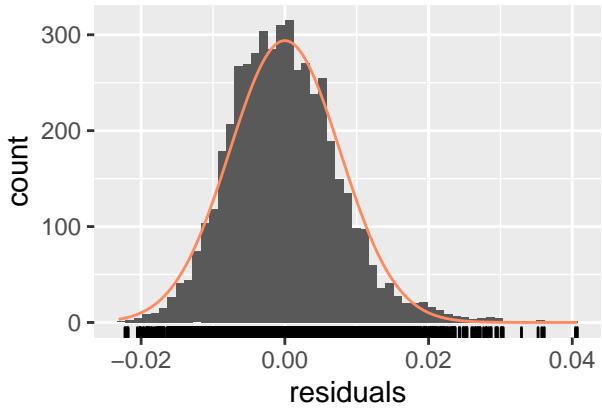
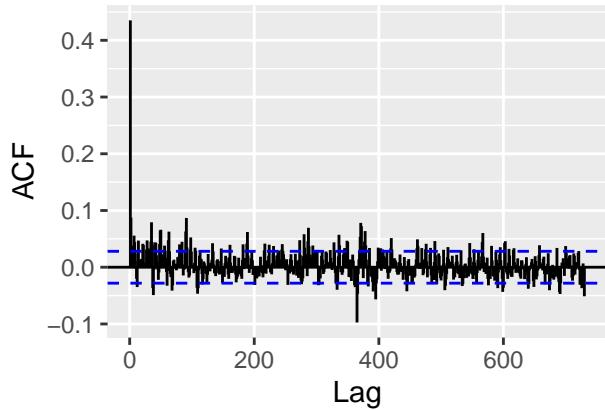
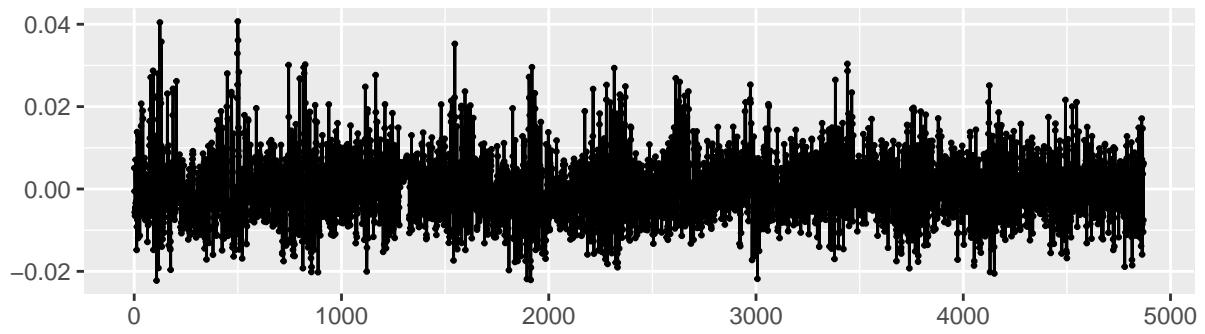
Regression prediction



Now we analyse the residuals. We fit a LM model in all training data in order to analyse it.

```
train = data.frame(  
  t = t,  
  o3_train = o3_train,  
  Q = Q  
)  
mod = lm(o3_train~t+Q, data = train)  
checkresiduals(mod, lag = 2*freq, lag.max = 2*freq)
```

Residuals



```
##  
## Breusch-Godfrey test for serial correlation of order up to 730  
##  
## data: Residuals  
## LM test = 1685.6, df = 730, p-value < 2.2e-16
```

As before, we see spikes in lag = 365, 730. We expect a WN to not have this.

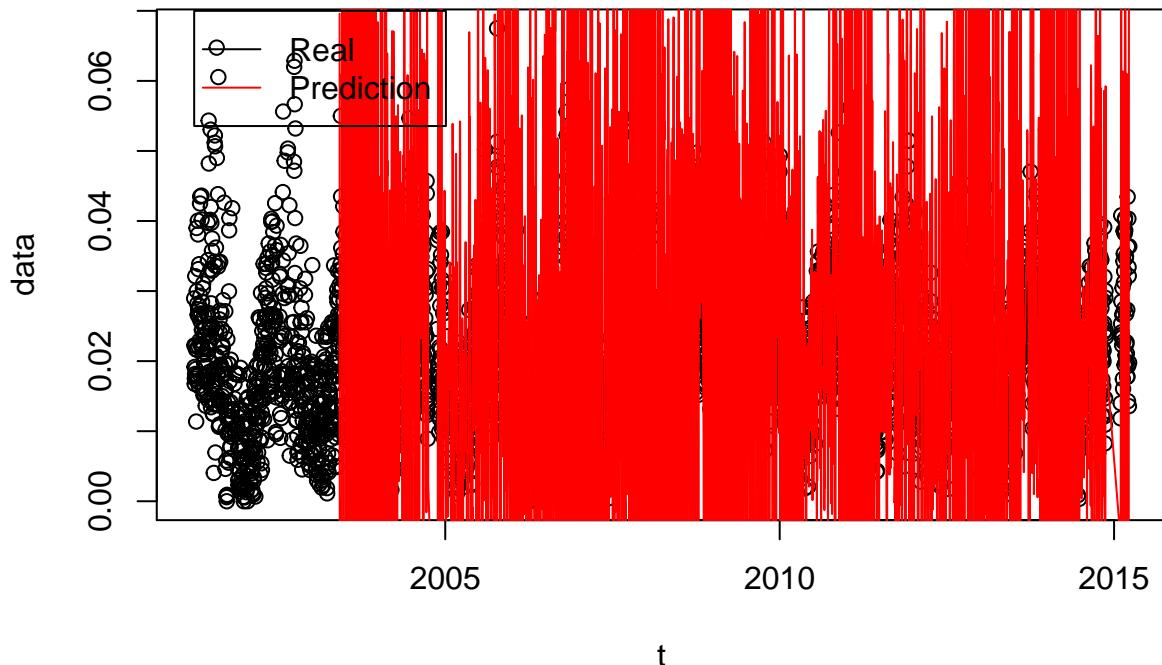
Holt-Winters

Now we will try Holt-Winters models. In fact, because of apparently seasonality, we will consider complete Holt-Winters models, both additive and multiplicative.

Additive

```
## [1] 0.03493752
```

Additive Holt–Winters prediction

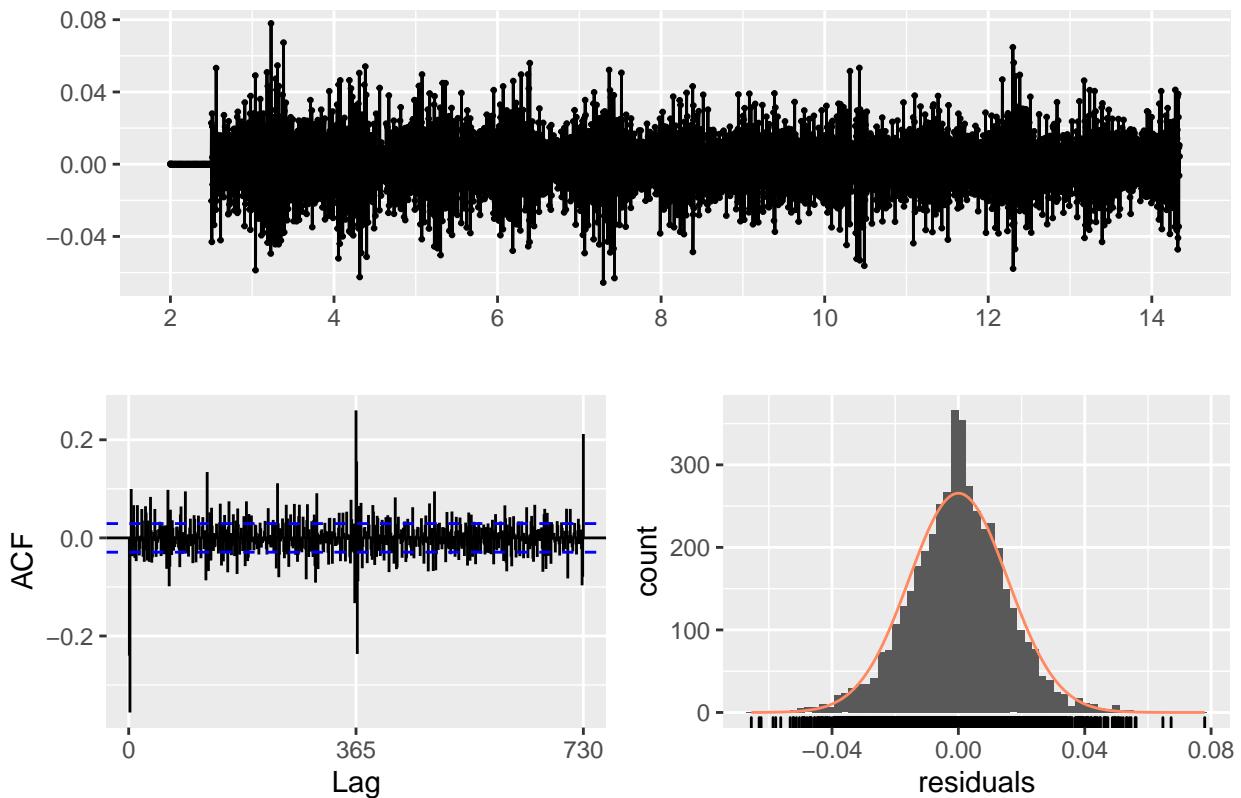


MAE is not so good, nor is the resulting graph. HW tends to “explode”. We have seem better results. Let's analyse the residuals:

```
mod = HoltWinters(ts(o3_train, frequency = 365), beta = T, gamma = T, seasonal = "additive")
checkresiduals(mod, lag = 2*freq, lag.max = 2*freq)

## Warning in modeldf.default(object): Could not find appropriate degrees of
## freedom for this model.
```

Residuals from HoltWinters

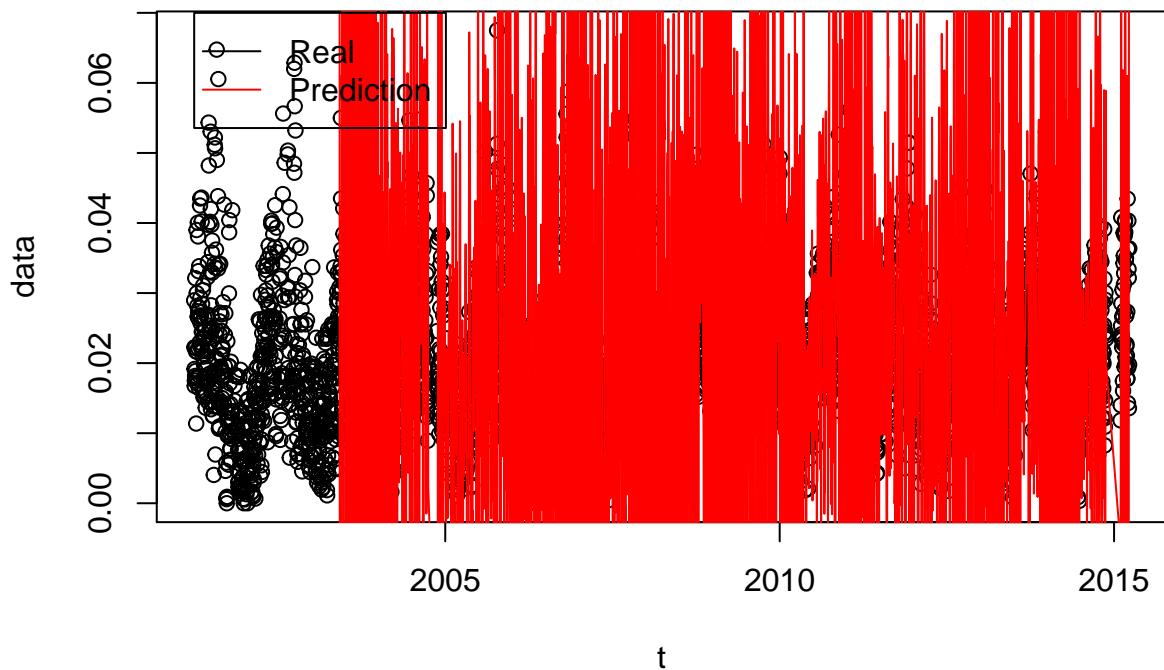


We see the same problems as before: high correlated lag = 365, 730, evidence of this not being a WN.

Multiplicative

```
## [1] 0.03485431
```

Multiplicative Holt–Winters prediction

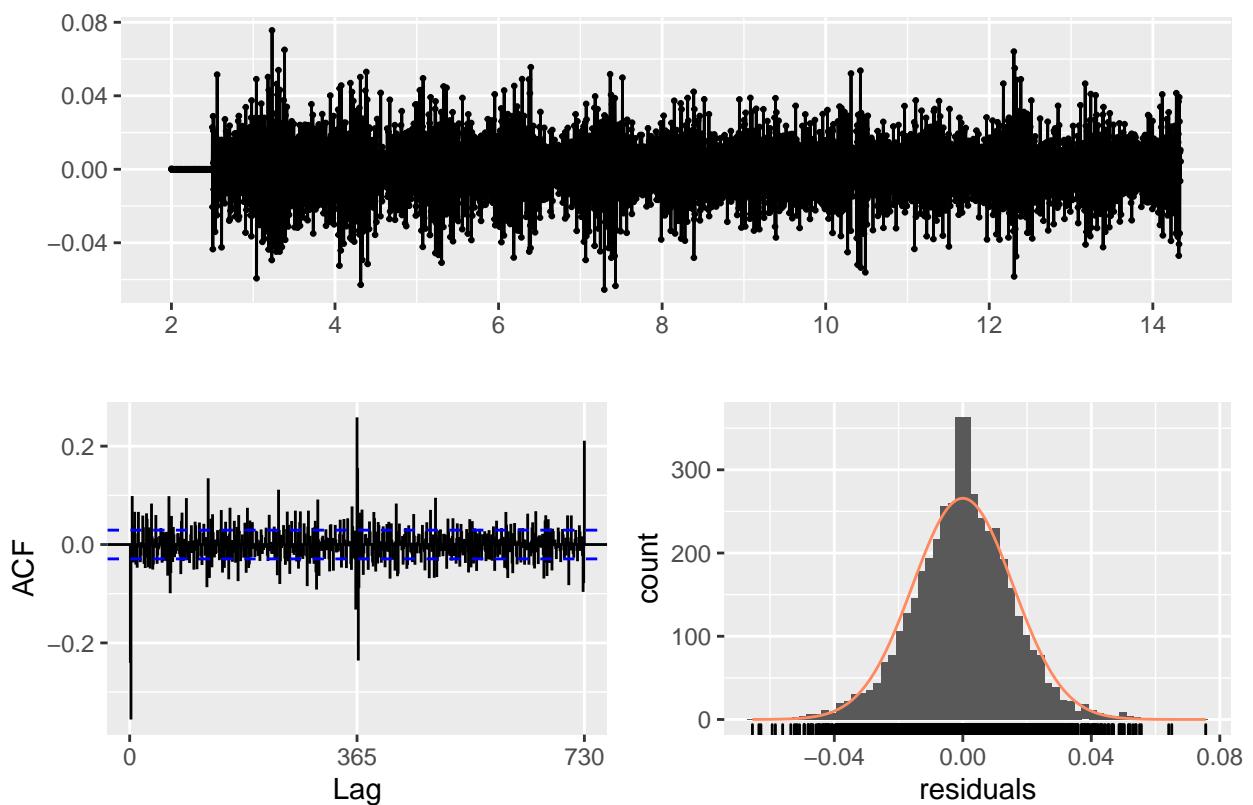


We see also a not so good MAE and graph. Let's analyse the residuals of an model fitted in all training data:

```
mod = HoltWinters(ts(o3_train+1, frequency = 365), beta = T, gamma = T, seasonal = "multiplicative")
checkresiduals(mod, lag = 2*freq, lag.max = 2*freq)

## Warning in modelfdf.default(object): Could not find appropriate degrees of
## freedom for this model.
```

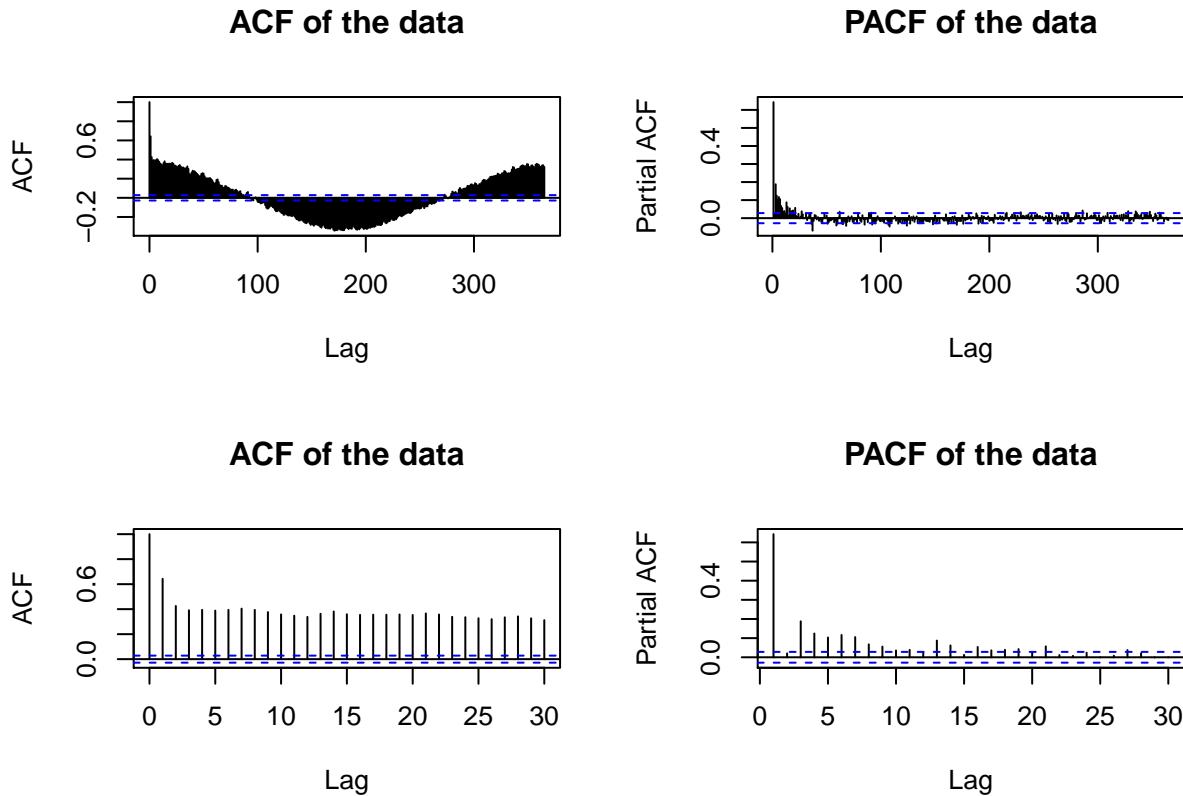
Residuals from HoltWinters



Same problems as before. Holt-Winters doesn't appear to work well in this case.

ARMA

We can see the ACF and PACF:



Based on these graphs, we see both graphs has a exponentially decay, the first after the $p - q = 1$ or $p - q = 2$. In order to identify the model, we will compare the adjusted ARMA models with different p and q . First we simply fit it to look at the Akaike Information Criteria (AIC) and the significance of the parameters estimated.

The AIC measures the goodness of fit and the simplicity of the model into a single statistic. Generally we aim to reduce the AIC.

$$AIC = 2k - 2 \ln(\hat{L}),$$

where $k = p + q + 2$ and \hat{L} is the maximum value of the likelihood for the model.

```
##
## Call:
## arma(x = o3_train, order = c(2, 1))
##
## Model:
## ARMA(2,1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -0.0256895 -0.0050812 -0.0003623  0.0043066  0.0398679
## 
## Coefficient(s):
##             Estimate Std. Error t value Pr(>|t|)    
## ar1        1.4018803  0.0150148  93.367 < 2e-16 ***
## ar2       -0.4073756  0.0146509 -27.805 < 2e-16 ***
## ma1       -0.9294688  0.0059259 -156.849 < 2e-16 ***
## intercept  0.0001116  0.0000292    3.822 0.000132 ***
```

```

## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Fit:
## sigma^2 estimated as 5.396e-05,  Conditional Sum-of-Squares = 0.26,  AIC = -34030.29
##
## Call:
## arma(x = o3_train, order = c(3, 1))
##
## Model:
## ARMA(3,1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.0280340 -0.0049450 -0.0003378  0.0043396  0.0391002
##
## Coefficient(s):
##             Estimate Std. Error t value Pr(>|t|)
## ar1        1.436e+00  1.616e-02  88.837 < 2e-16 ***
## ar2       -5.972e-01  2.357e-02 -25.337 < 2e-16 ***
## ar3        1.537e-01  1.485e-02  10.346 < 2e-16 ***
## ma1       -9.060e-01  8.615e-03 -105.163 < 2e-16 ***
## intercept 1.531e-04  3.928e-05    3.897 9.75e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Fit:
## sigma^2 estimated as 5.282e-05,  Conditional Sum-of-Squares = 0.26,  AIC = -34132.4
##
## Call:
## arma(x = o3_train, order = c(3, 2))
##
## Model:
## ARMA(3,2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.027737 -0.004941 -0.000356  0.004364  0.039341
##
## Coefficient(s):
##             Estimate Std. Error t value Pr(>|t|)
## ar1        1.2360037  0.0891579 13.863 < 2e-16 ***
## ar2       -0.3130929  0.1287091 -2.433 0.014992 *
## ar3        0.0686044  0.0422242  1.625 0.104213
## ma1       -0.7035439  0.0887237 -7.930 2.22e-15 ***
## ma2       -0.1896764  0.0832069 -2.280 0.022633 *
## intercept  0.0001719  0.0000448   3.837 0.000124 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Fit:
## sigma^2 estimated as 5.279e-05,  Conditional Sum-of-Squares = 0.26,  AIC = -34133.17
##

```

```

## Call:
## arma(x = o3_train, order = c(4, 3))
##
## Model:
## ARMA(4,3)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.0275745 -0.0049606 -0.0003479  0.0043427  0.0391448
##
## Coefficient(s):
##             Estimate Std. Error t value Pr(>|t|)
## ar1        0.9060239  0.3974686   2.279  0.02264 *
## ar2       -0.0265556  0.4953456  -0.054  0.95725
## ar3        0.1412857  0.1831459   0.771  0.44045
## ar4       -0.0326660  0.0554639  -0.589  0.55589
## ma1       -0.3723047  0.3971375  -0.937  0.34852
## ma2       -0.2997470  0.2906608  -1.031  0.30242
## ma3       -0.1781035  0.1224956  -1.454  0.14596
## intercept 0.0002418  0.0000921   2.625  0.00866 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Fit:
## sigma^2 estimated as 5.281e-05, Conditional Sum-of-Squares = 0.26, AIC = -34127.1
##
## Call:
## arma(x = o3_train, order = c(4, 2))
##
## Model:
## ARMA(4,2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.0278979 -0.0049234 -0.0003225  0.0043674  0.0393150
##
## Coefficient(s):
##             Estimate Std. Error t value Pr(>|t|)
## ar1        1.059e+00  4.642e-01   2.281  0.0226 *
## ar2       -6.008e-02  6.680e-01  -0.090  0.9283
## ar3       -4.541e-02  2.895e-01  -0.157  0.8754
## ar4       3.665e-02  8.276e-02   0.443  0.6579
## ma1       -5.259e-01  4.625e-01  -1.137  0.2555
## ma2       -3.481e-01  4.171e-01  -0.834  0.4040
## intercept 2.042e-04  9.095e-05   2.246  0.0247 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Fit:
## sigma^2 estimated as 5.281e-05, Conditional Sum-of-Squares = 0.26, AIC = -34128.87

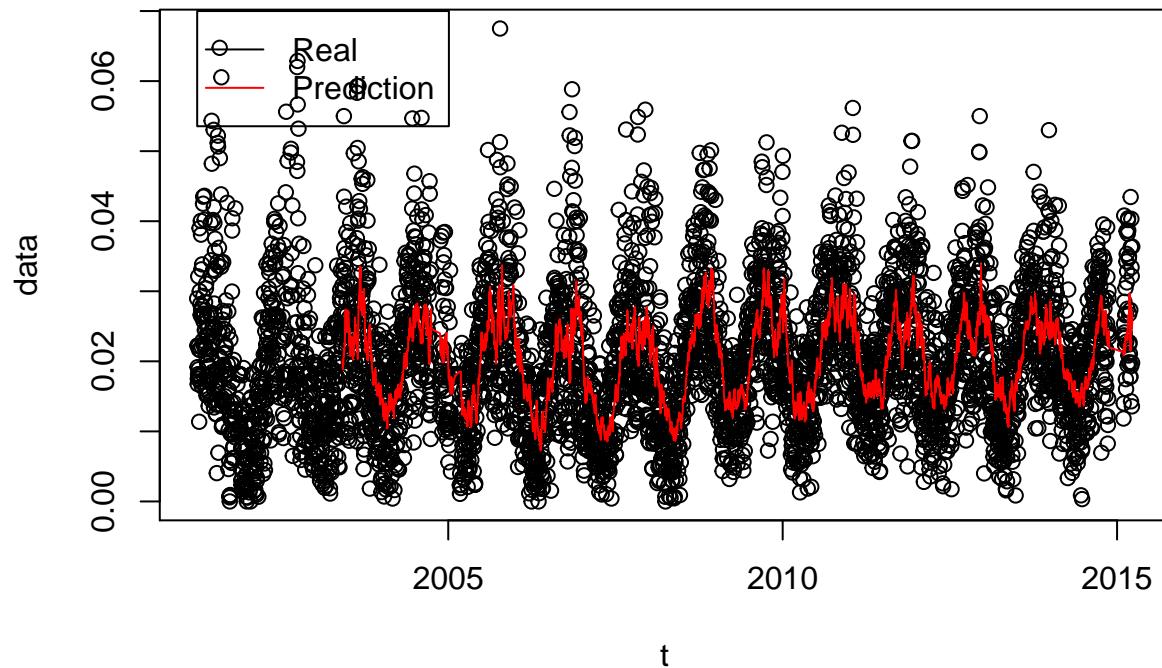
```

First we see that in the last two model, there is no statistical significance in the major part of the parameters, so we'll no consider it. The first, second and third models have significant parameters and similar AIC. In especial the 2° and 3° has the smallest AIC. So we will compare them both.

We see the MAE of the ARIMA(3, 0, 1):

```
## [1] 0.006463897
```

ARIMA(3,0,1) prediction

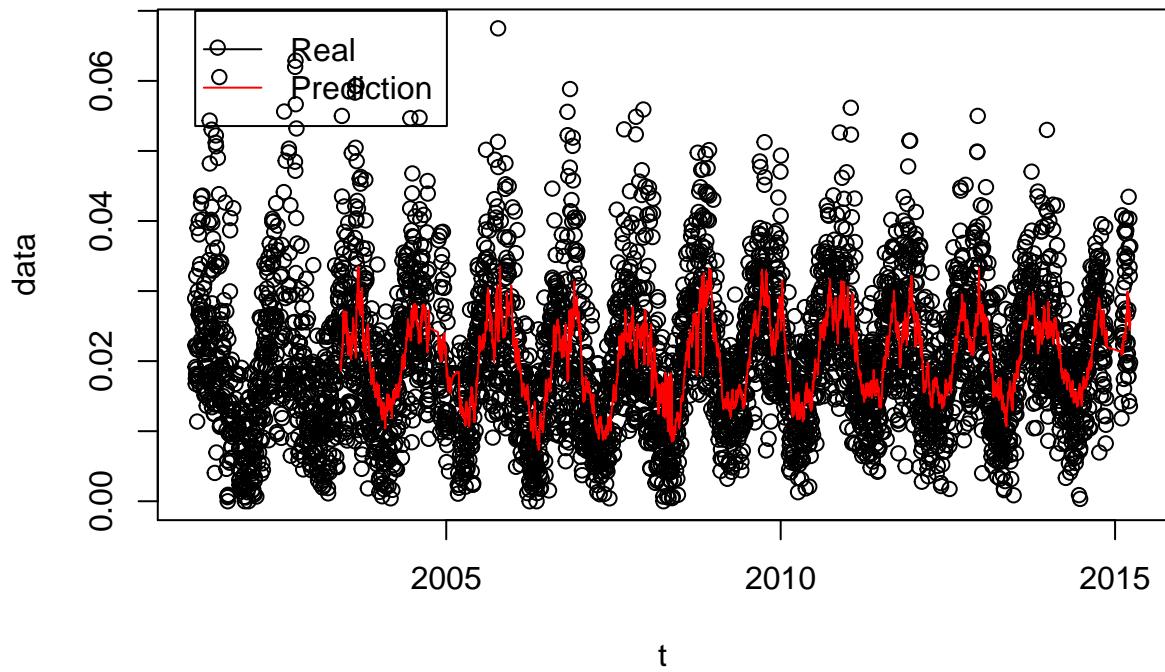


And the MAE of the ARIMA(3, 0, 2):

```
## [1] "MAE ARIMA(3,0,2)"
```

```
## [1] 0.006483669
```

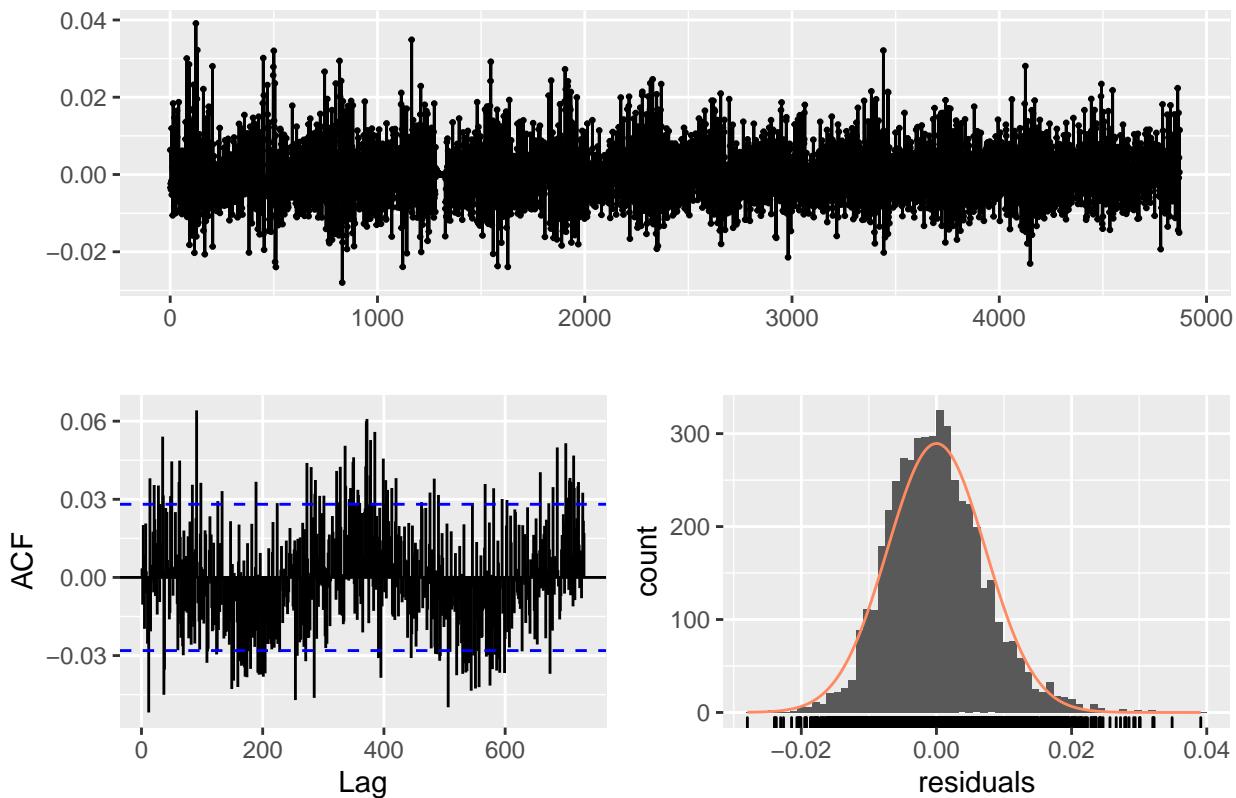
ARIMA(3,0,2) prediction



The first model seems a little better. So, let's check the residuals to observe it's problems.

```
model <- arima(o3_train, order = c(3,0,1))
checkresiduals(model, lag = 2*freq, lag.max = 2*freq)
```

Residuals from ARIMA(3,0,1) with non-zero mean



```
##  
## Ljung-Box test  
##  
## data: Residuals from ARIMA(3,0,1) with non-zero mean  
## Q* = 1627, df = 725, p-value < 2.2e-16  
##  
## Model df: 5. Total lags used: 730
```

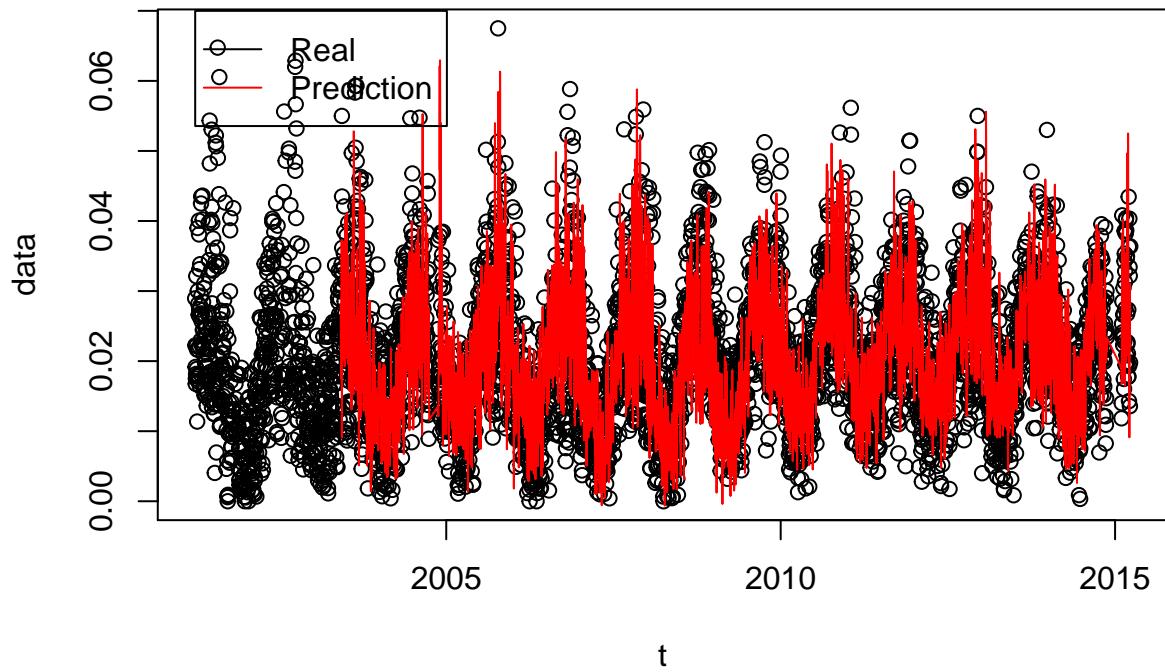
It's interesting to note the ACF has a peak around the 365, so the ARMA model did not seem to capture the seasonality. It would be better to fit an Seasonal ARIMA further. The histogram is pretty similar to normal distribution. The test made analyses if the mean is 0. It may be because the p-value is really small. That's the reason to adapt ARMA with STL, that is, fit an ARMA model in the residuals.

Adapting ARMA

The ARMA seems to fit well as we can see so far. However, it's not capturing other characteristics on the data, as seasonality. For that reason, we will combine the stl and arma model and extract the best of each one. We will decompose the series in trend and seasonality and in the reminder, we fit an arima model with `auto.arima()`.

```
## [1] 0.007855054
```

STL + ARIMA prediction



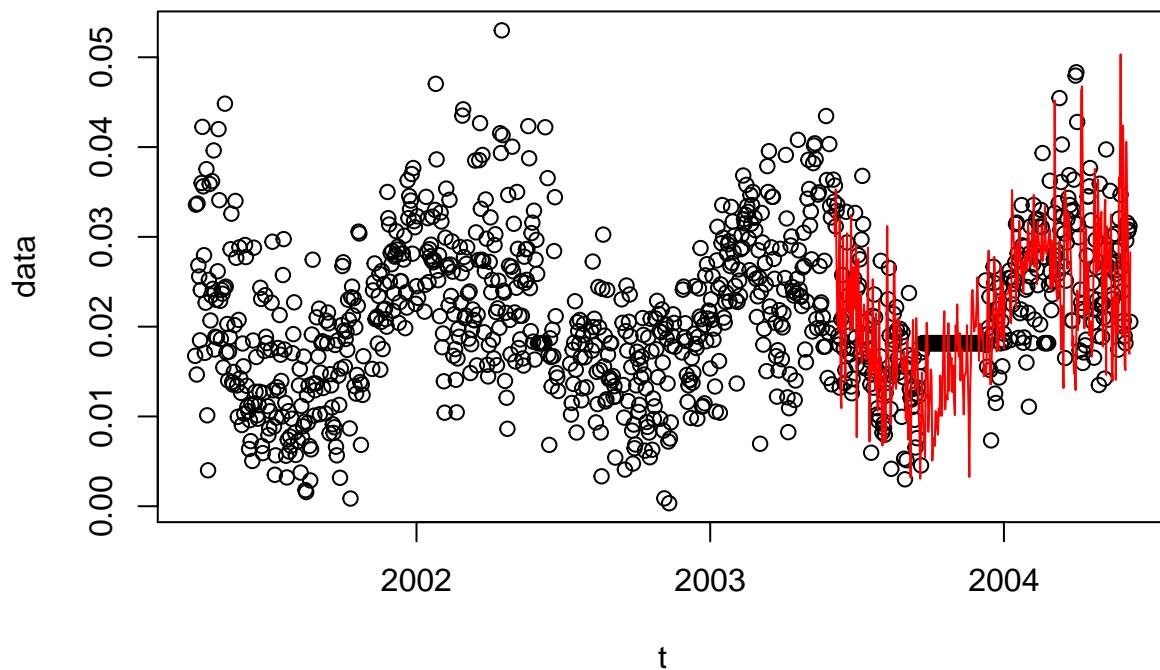
However the MAE isn't improved by this model. For that reason, this model was disregarded.

Comparing models in the test data.

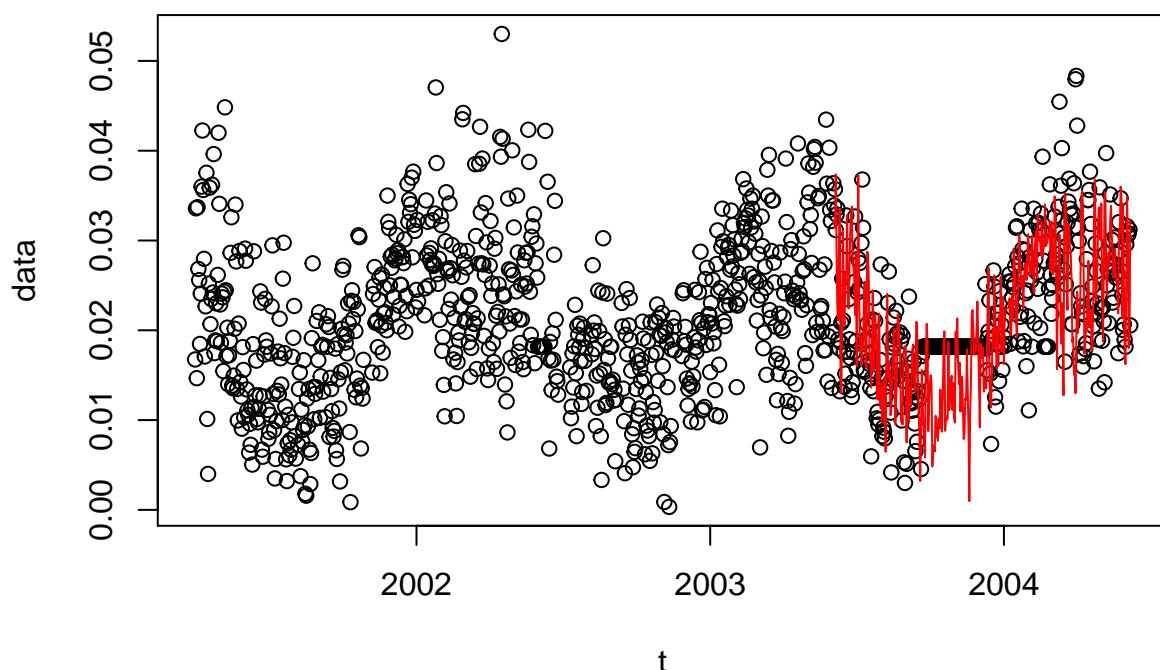
We chose some of the best models to compare with the testing data.

```
## [1] 0.006599825
```

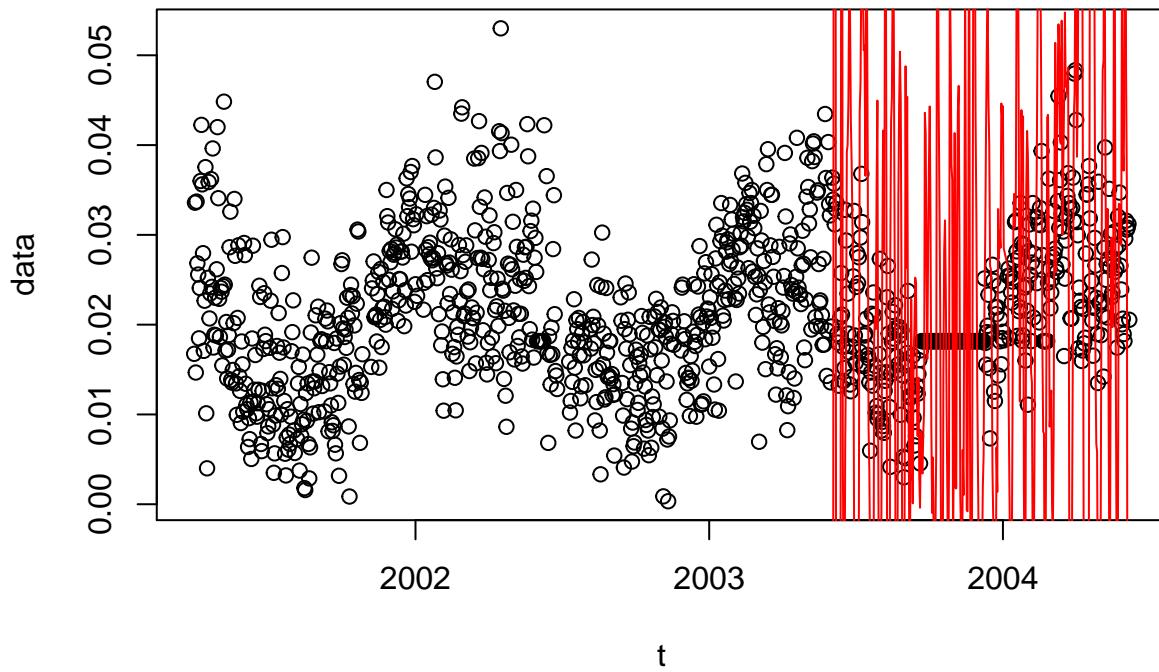
Multiplicative decompose prediction



Regression prediction

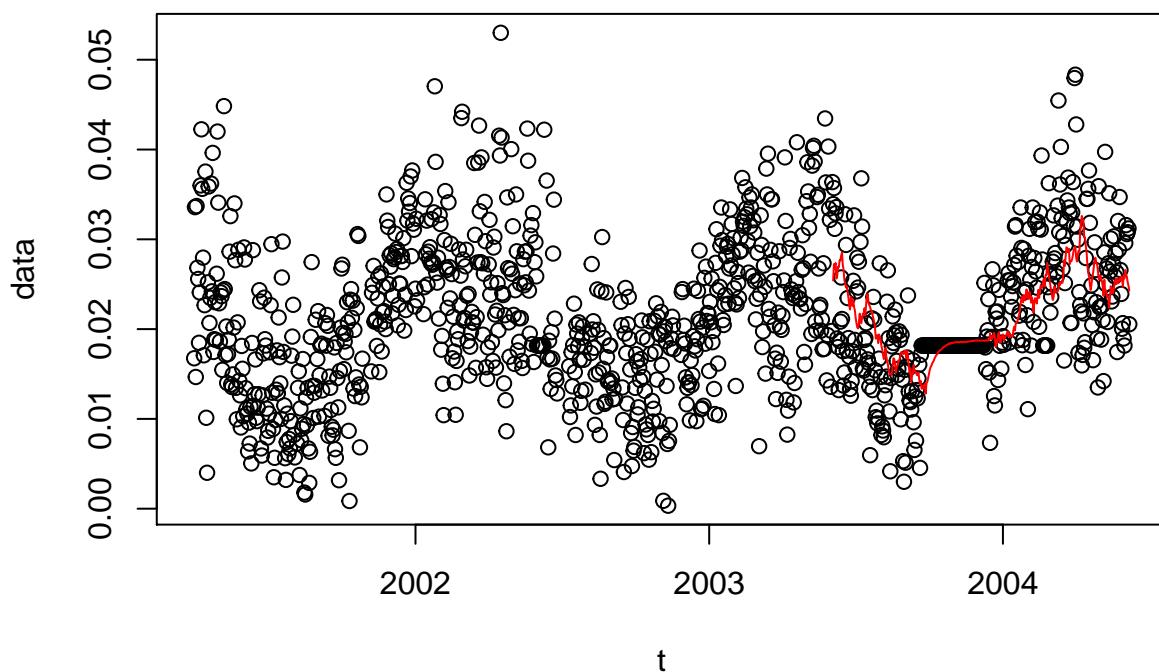


Multiplicative Holt–Winters prediction



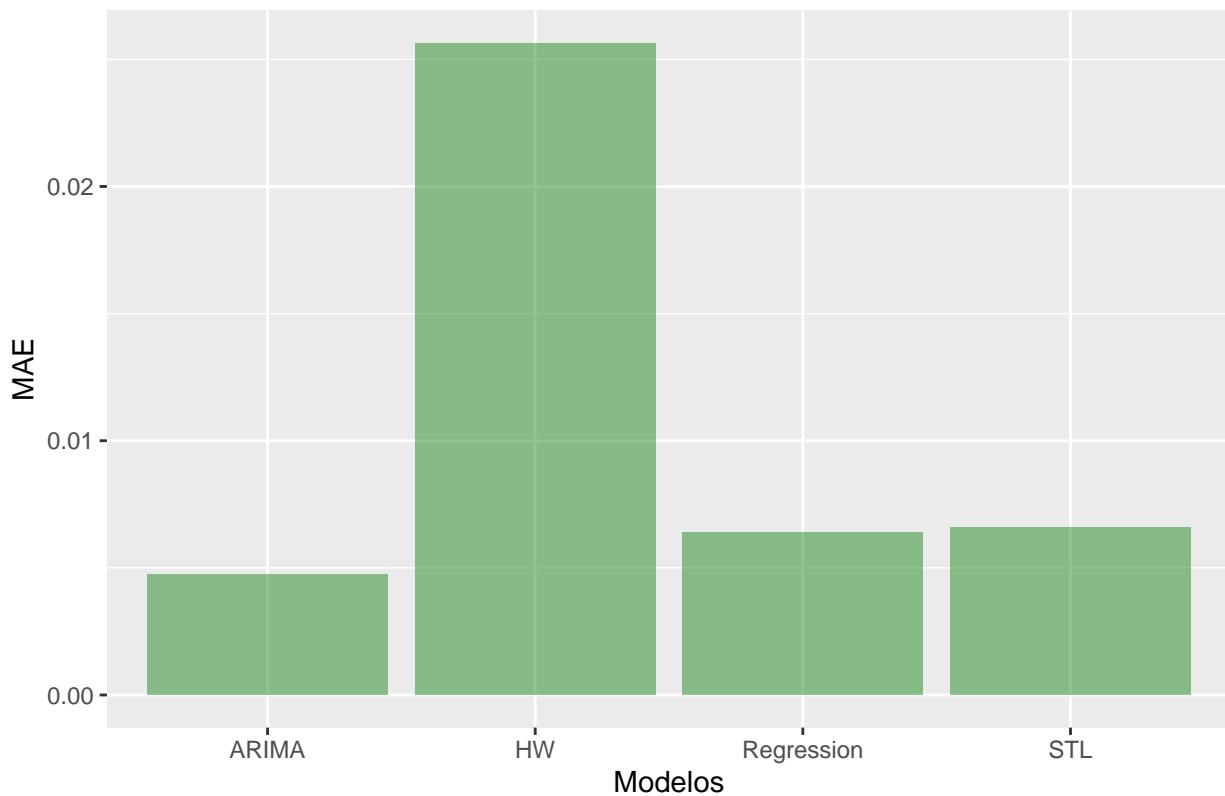
```
## [1] "MAE ARIMA(3,0,1)"  
## [1] 0.004755646
```

ARIMA(3,0,1) prediction



Finally, we have this bar graphic to compare the values:

Comparando MAE para os modelos



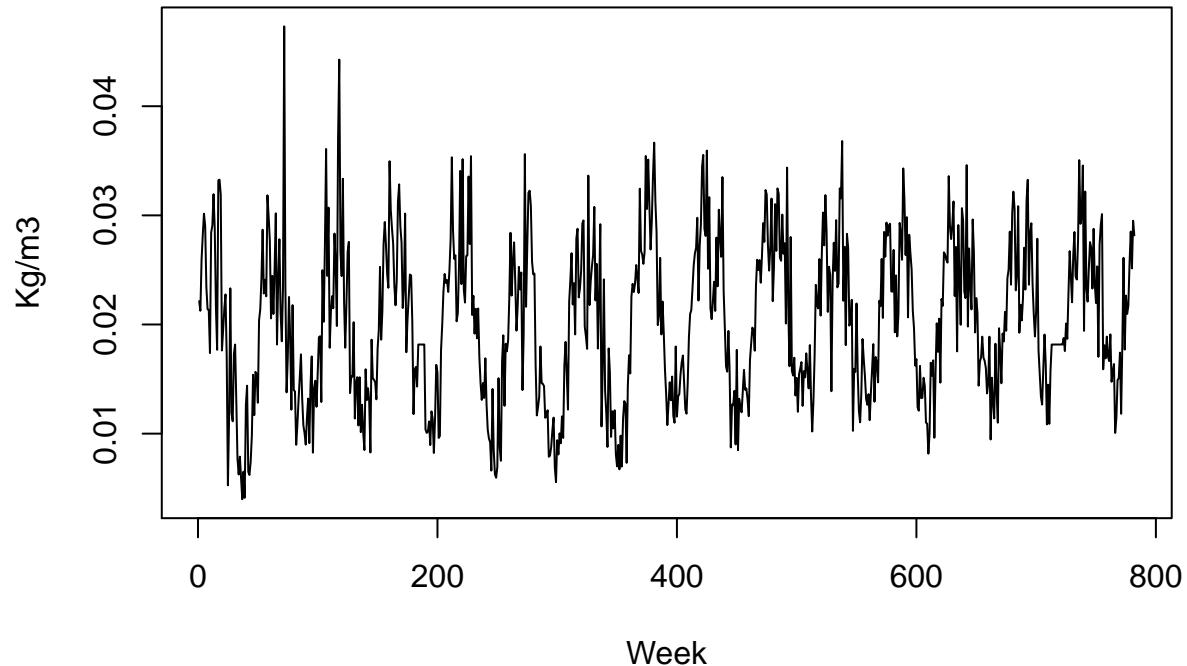
Models: case 2

In case two, we have to aggregate the diary days in a week, starting from the sunday, as requested. So we calculate the mean value in the week to be its representant. The models may be very similar to the previous. We may see less outliers. We also will separate train and test data. The first day in the data is April 1, 2001, a Sunday. So we do not worry about that.

```
o3_week <- c(1:floor(length(o3.clean)/7))
for(i in seq(1,length(o3.clean)-7, 7)){
  o3_week[ceiling(i/7)] = mean(o3.clean[i:(i+6)])
}

plot(ts(o3_week), main = 'Weekly average level of O3 in Boston (after imputation)',
      xlab = 'Week', ylab = 'Kg/m3')
```

Weekly average level of O₃ in Boston (after imputation)



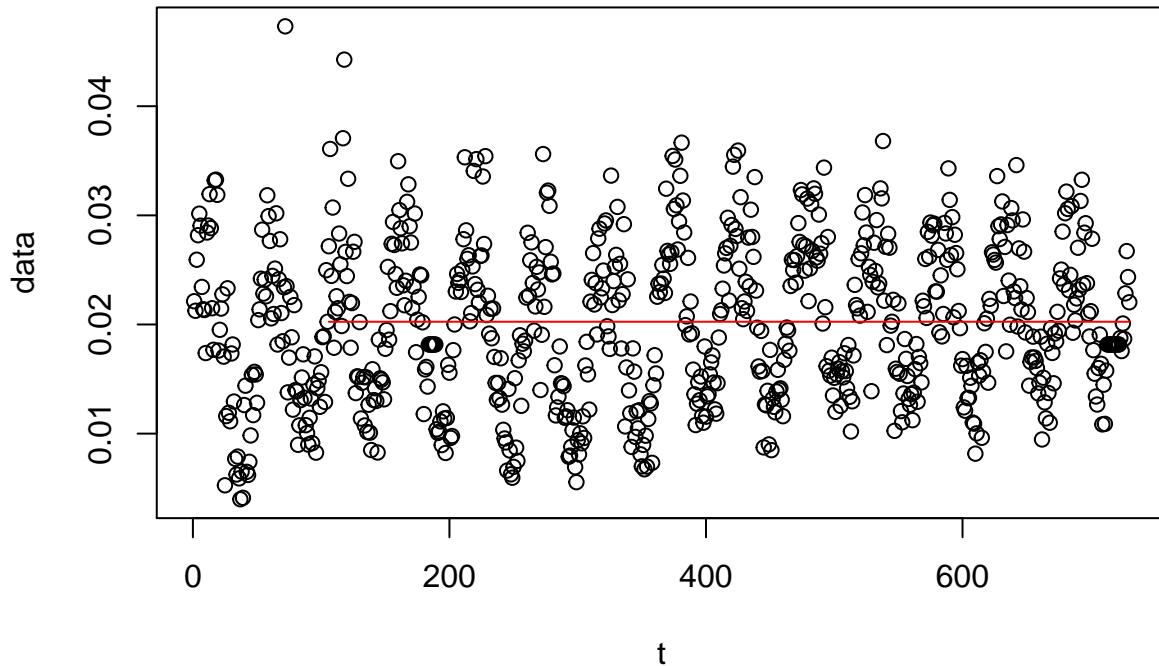
```
o3_train_week = o3_week[1:(length(o3_week)[1] - 52)]  
o3_test_week = o3_week[-c(1:(length(o3_week)[1] - 52))]
```

Baseline Model

We will do the naive forecast to the baseline model.

```
## [1] 0.008817416
```

Baseline model prediction



Decompose

We now check for seasonality considering monthly (4 records) and yearly (52 records). For seasonality of 52, we see better p-value. In fact, less than 0.05. So we will use 52.

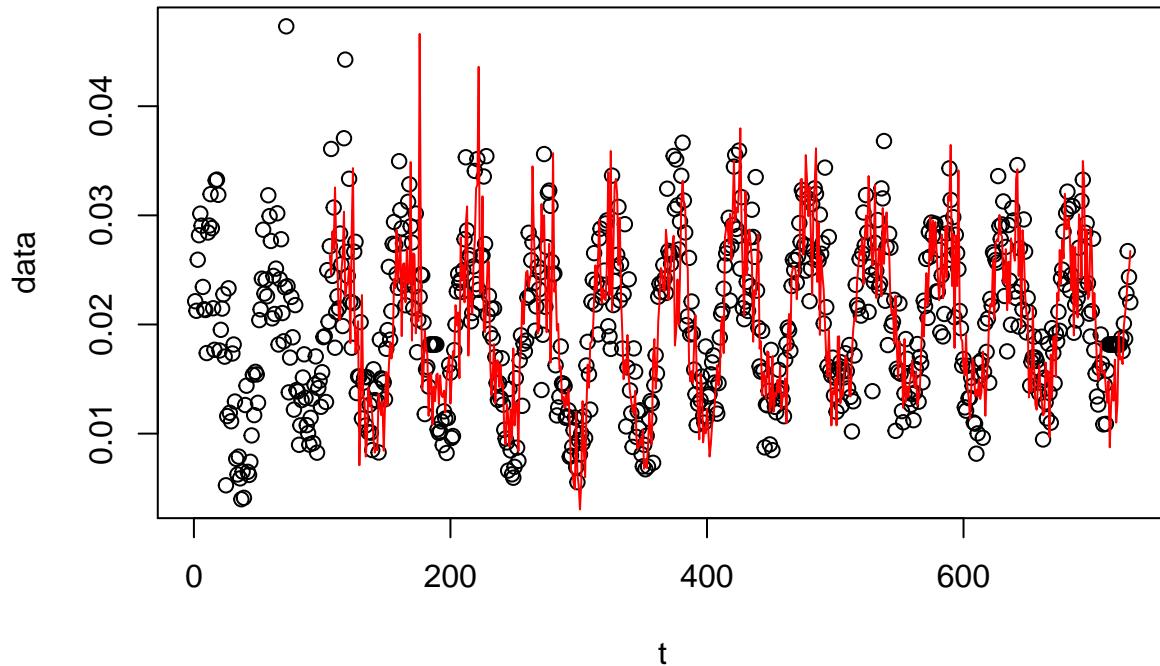
```
##  
## Kruskal-Wallis rank sum test  
##  
## data: o3_train_week and g  
## Kruskal-Wallis chi-squared = 0.12693, df = 3, p-value = 0.9884  
##  
## Kruskal-Wallis rank sum test  
##  
## data: o3_train_week and g  
## Kruskal-Wallis chi-squared = 527.25, df = 51, p-value < 2.2e-16
```

Additive model

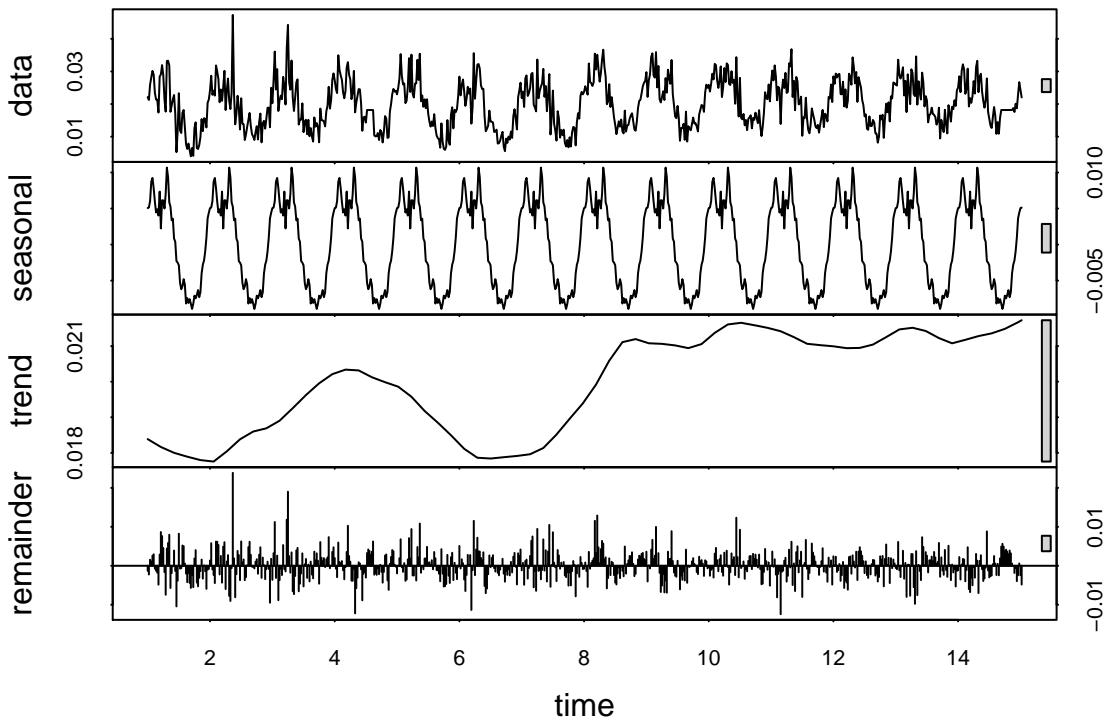
We see the MAE of the additive decompose model:

```
## [1] 0.003993143
```

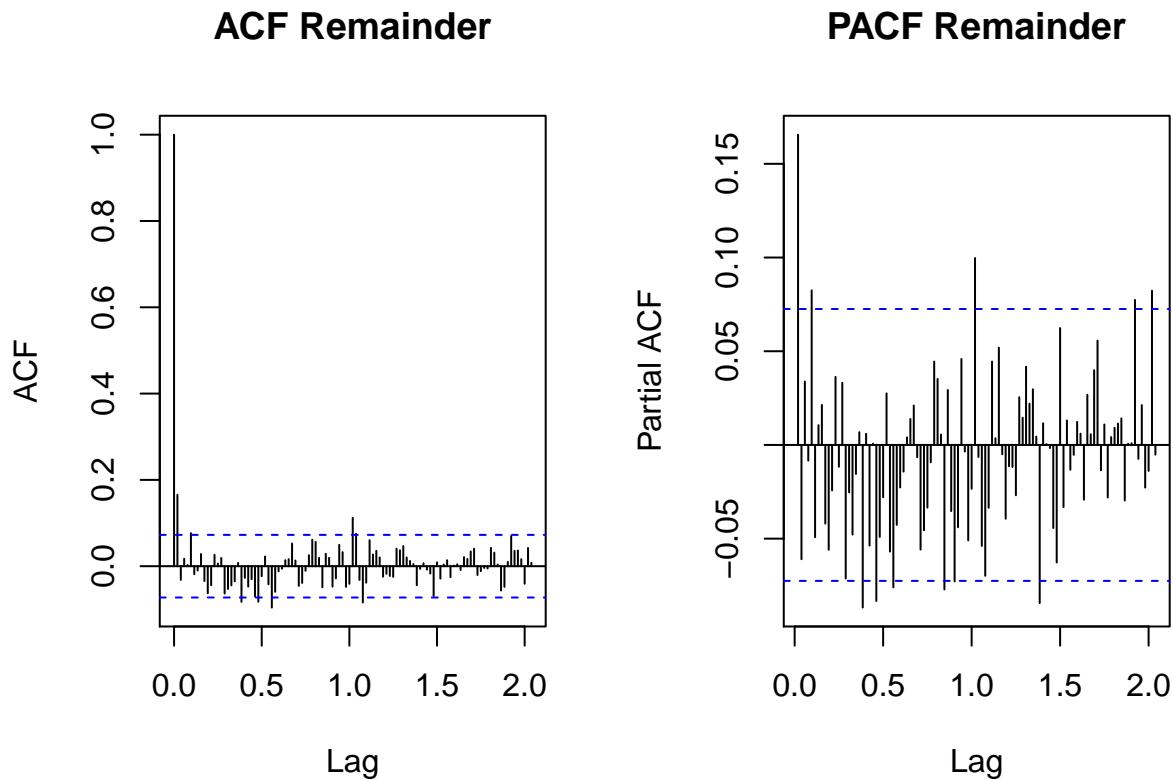
Additive decompose prediction



We fit the model using `t.window` and analyse the reminder:



The ACF and the PACF of the reminder:



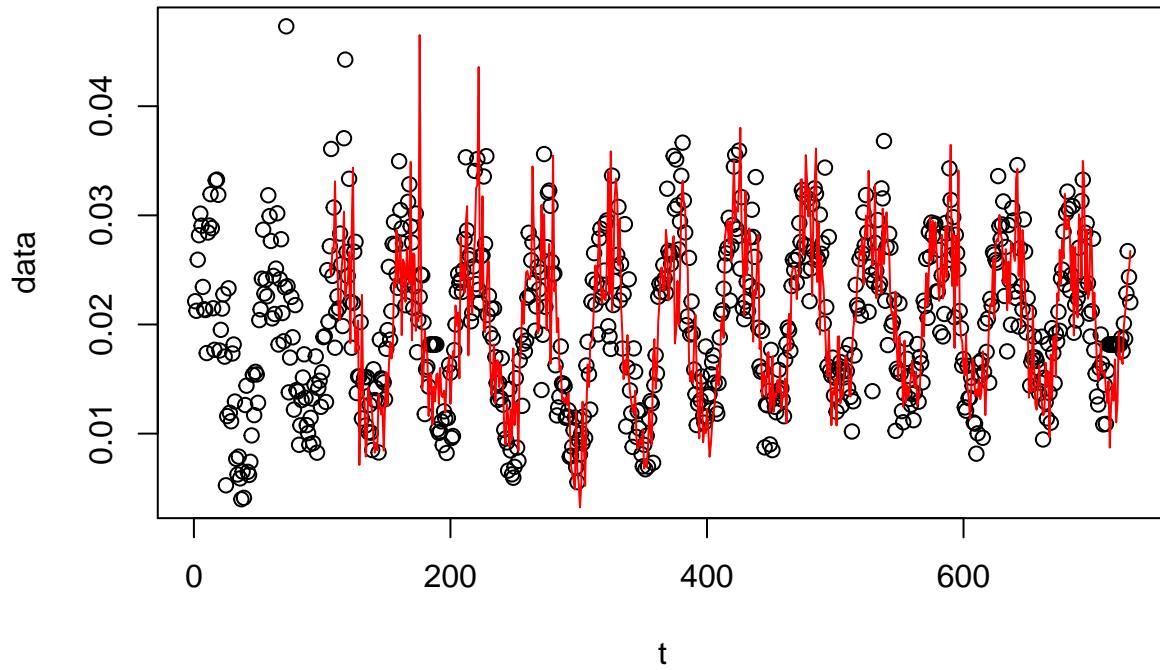
We see a big spike when lag = 52. It seems not so good for a reminder

Multiplicative model

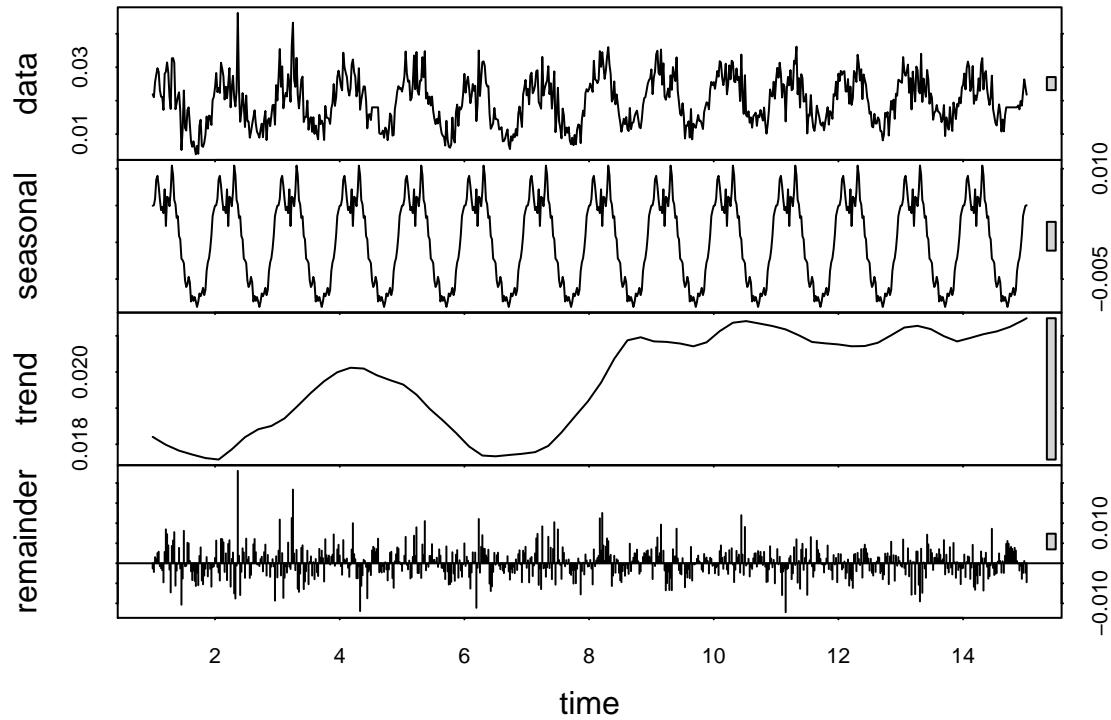
We analyse the MAE:

```
## [1] 0.003968754
```

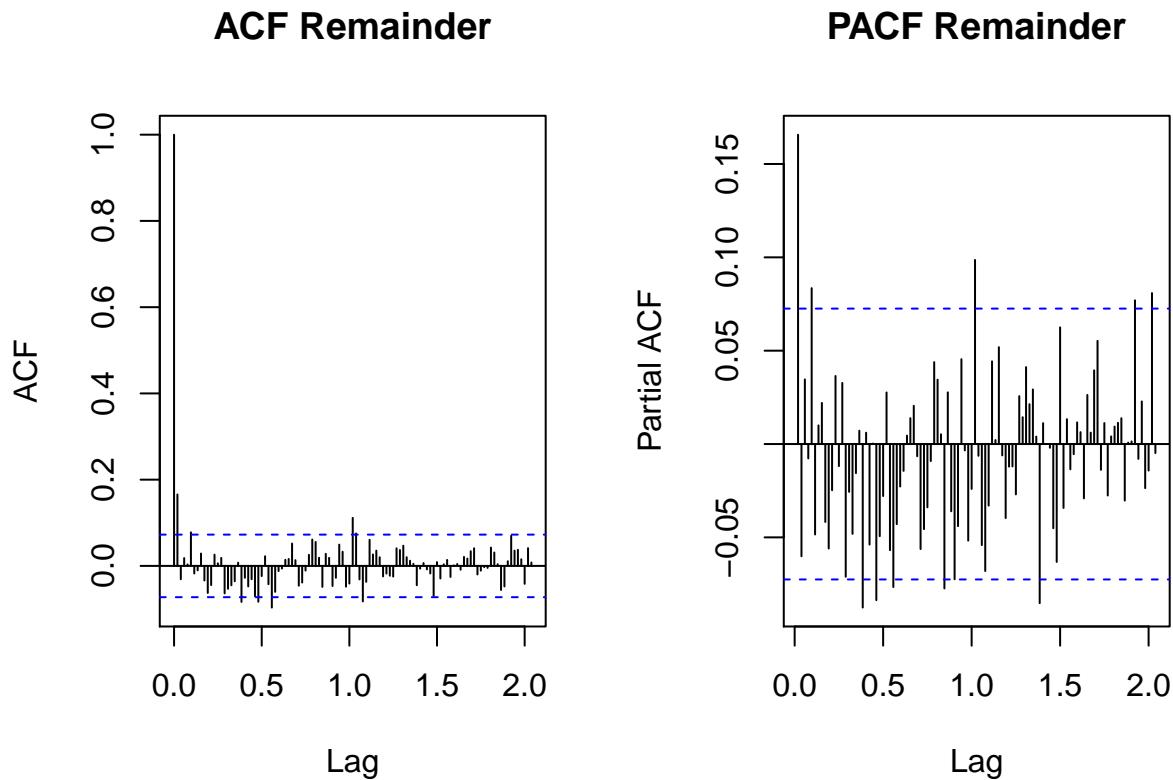
Multiplicative decompose prediction



Using `t.window` and analysing the reminder:



The ACF and the PACF of the reminder:



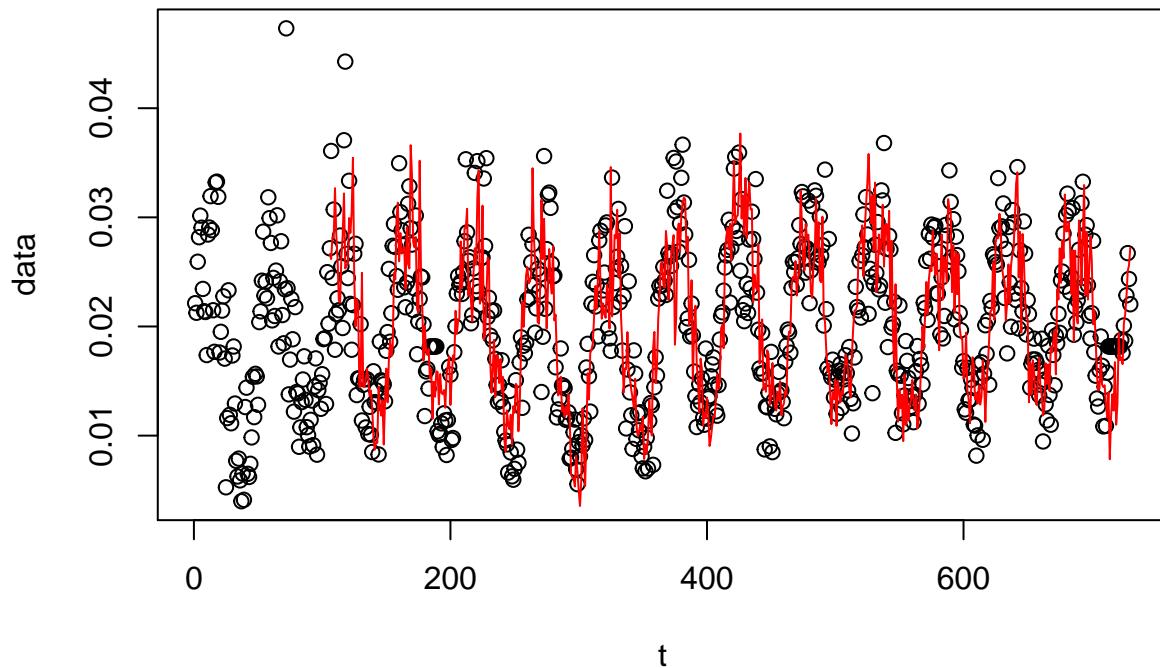
We also see the spike when lag = 52. It seems not so good for a reminder. The same as before.

Regression

We tested for seasonalities, and we settled with 52. So we fit a regression model with the seasonality dummies. We see the MAE:

```
## [1] 0.003803593
```

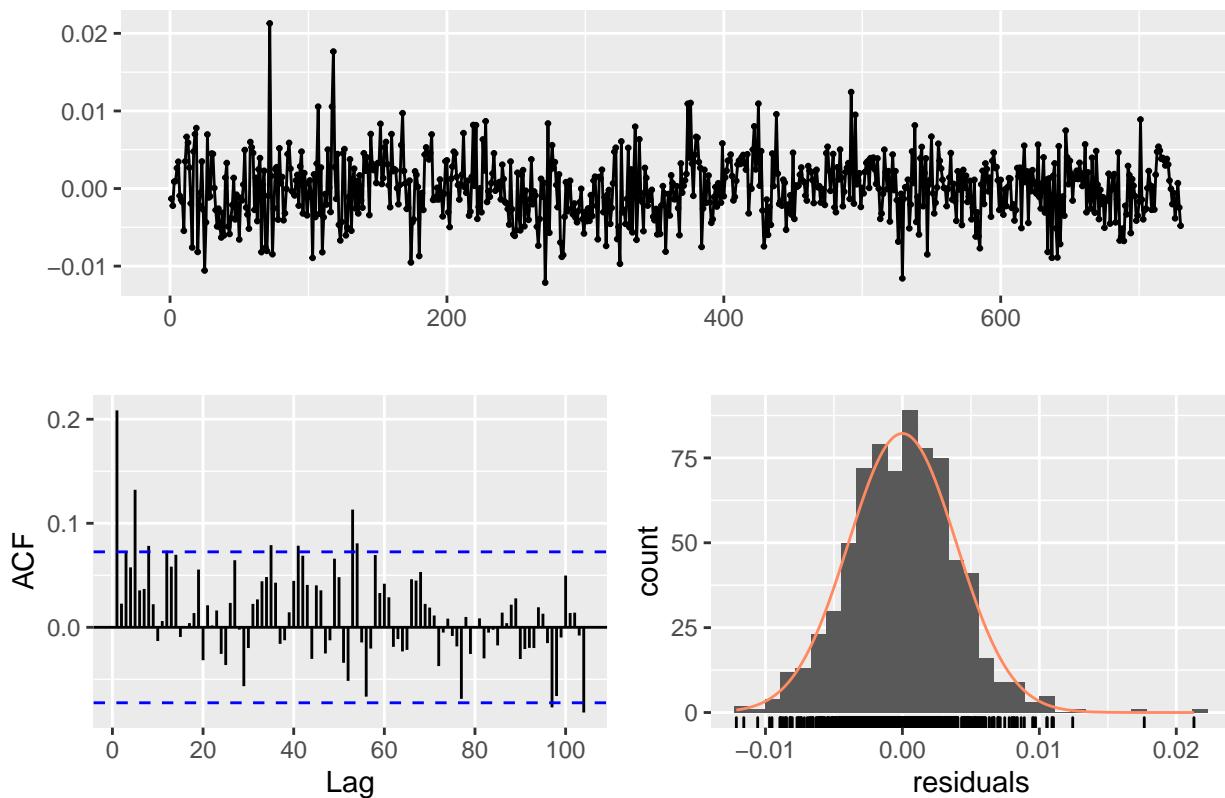
Regression prediction



Now we analyse the residuals.

```
train = data.frame(
  t = t,
  o3_train_week = o3_train_week,
  Q = Q
)
mod = lm(o3_train_week~t+Q, data = train)
checkresiduals(mod, lag = 2*freq, lag.max = 2*freq)
```

Residuals



```
##  
## Breusch-Godfrey test for serial correlation of order up to 104  
##  
## data: Residuals  
## LM test = 142.53, df = 104, p-value = 0.007267
```

We see spikes in lag = 52, 104. Same as before.

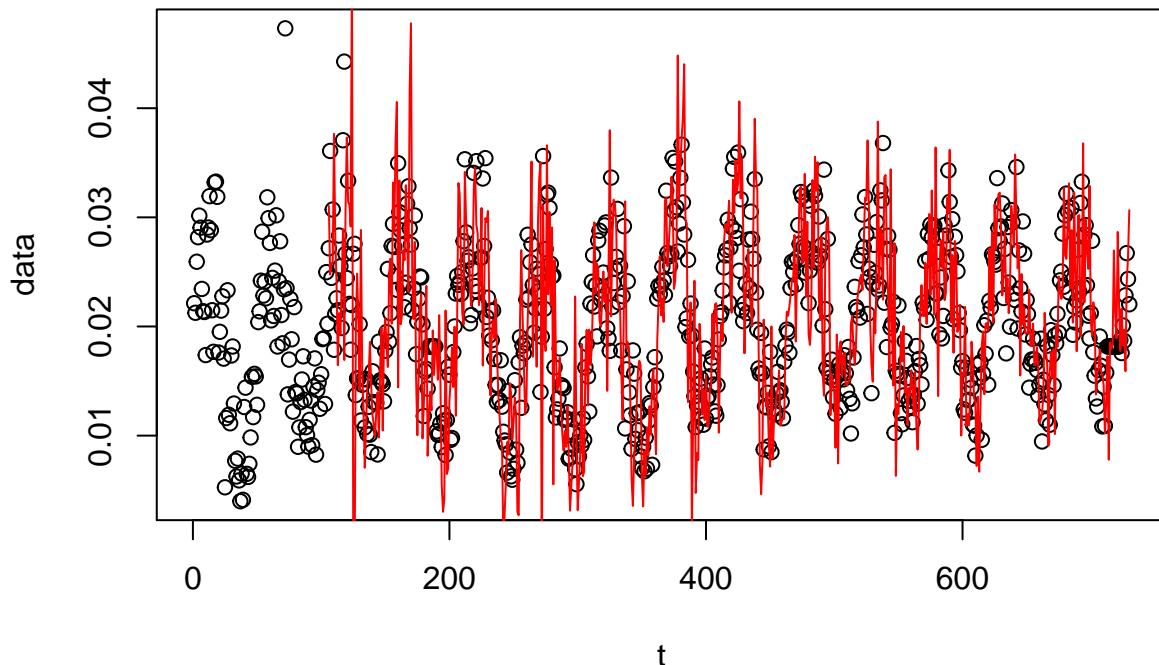
Holt-Winters

Now we will try Holt-Winters models. We will consider complete Holt-Winters models, those with seasonality.

Additive

```
## [1] 0.005158324
```

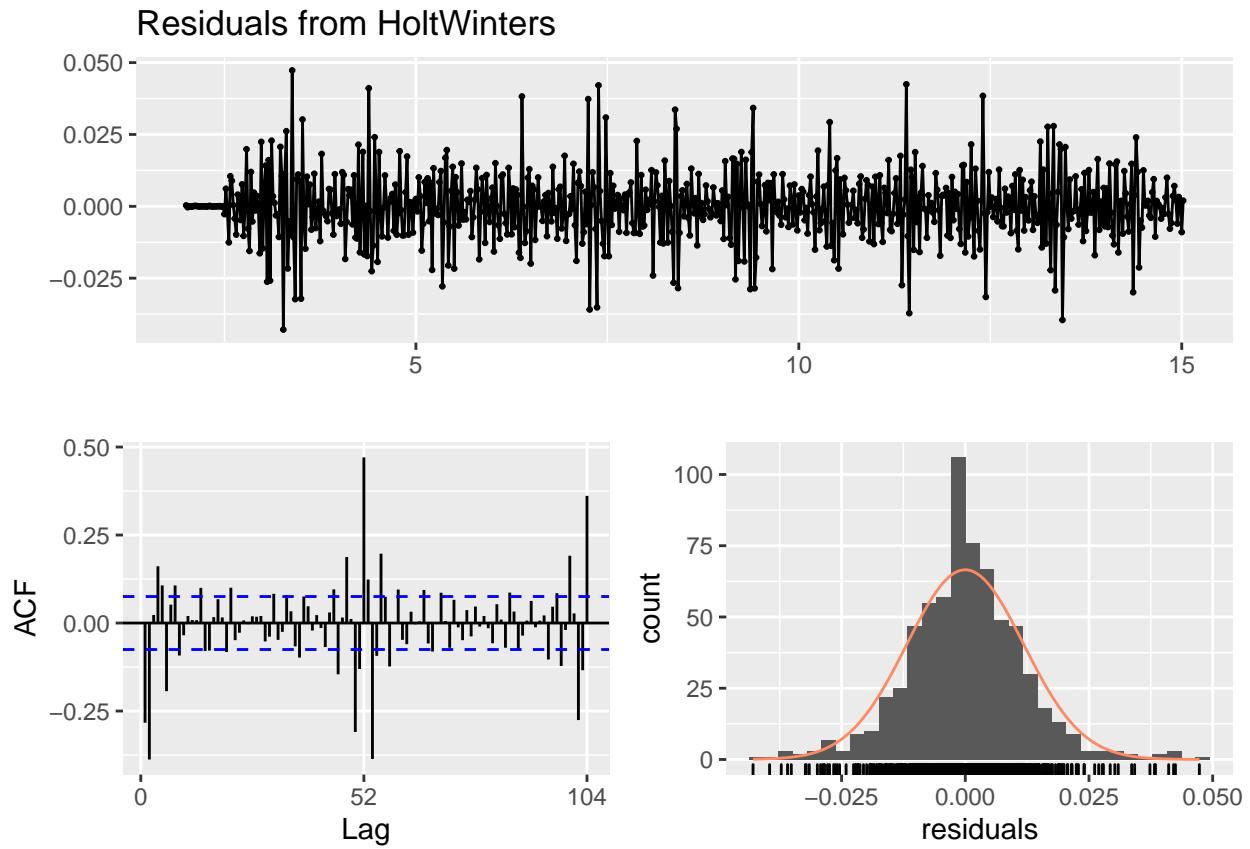
Additive Holt–Winters prediction



MAE is not so good as the other models seen so far. We see some kind of explosion in the resulting graph. We know that HW tends to “explode”. We have seem better results. Let's analyse the residuals:

```
mod = HoltWinters(ts(o3_train_week, frequency = 52), beta = T, gamma = T, seasonal = "additive")
checkresiduals(mod, lag = 2*freq, lag.max = 2*freq)

## Warning in modeldf.default(object): Could not find appropriate degrees of
## freedom for this model.
```

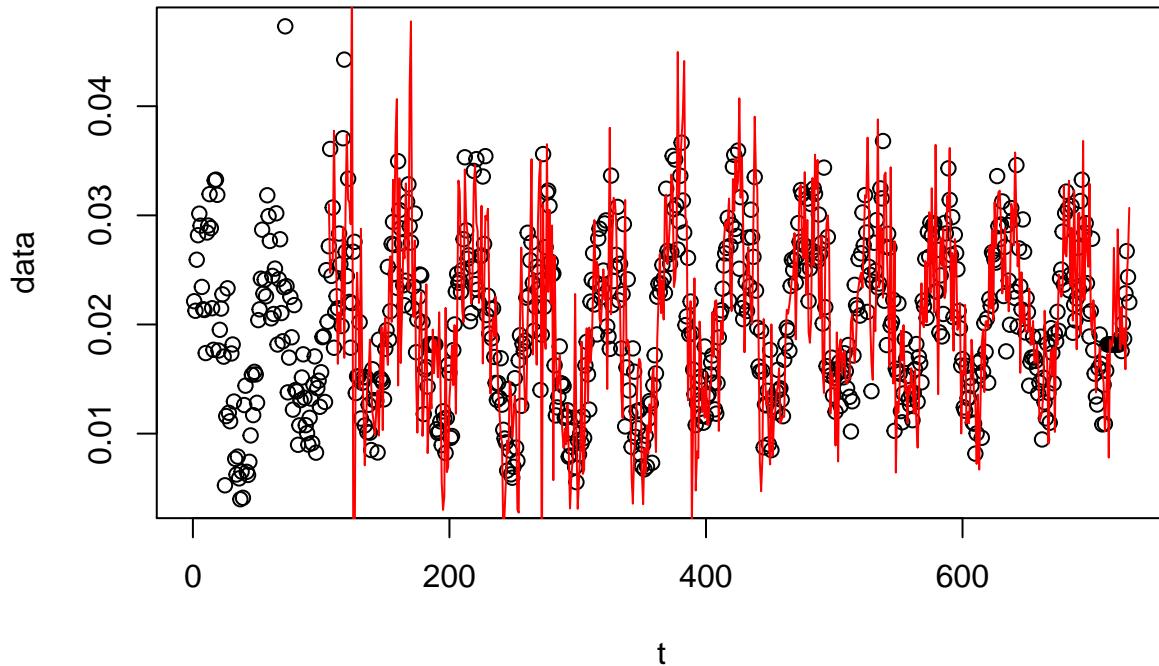


We see the same problems as before: high correlated lag = 52, 104, evidence of this not being a WN.

Multiplicative

```
## [1] 0.005168137
```

Multiplicative Holt–Winters prediction

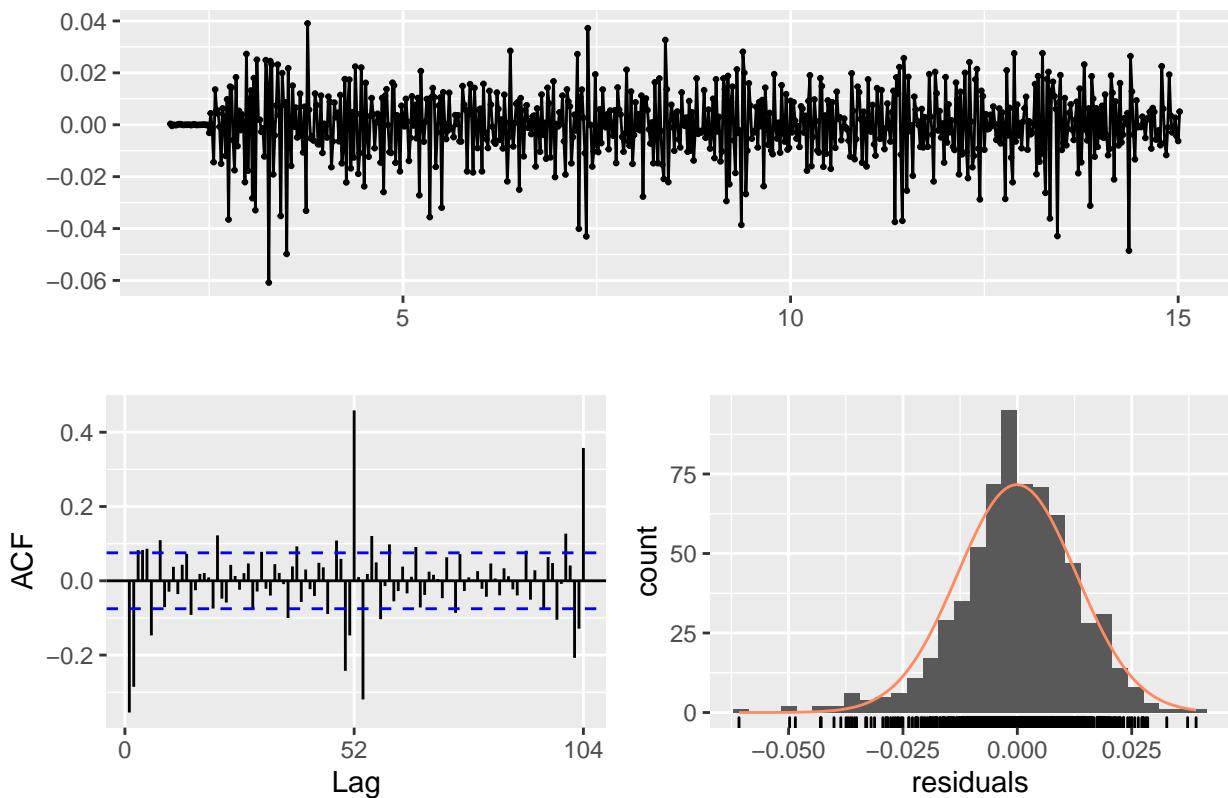


We see also a not so good MAE and graph. Let's analyse the residuals:

```
mod = HoltWinters(ts(o3_train_week, frequency = 52), beta = T, gamma = T, seasonal = "multiplicative")
checkresiduals(mod, lag = 2*freq, lag.max = 2*freq)

## Warning in modeldf.default(object): Could not find appropriate degrees of
## freedom for this model.
```

Residuals from HoltWinters

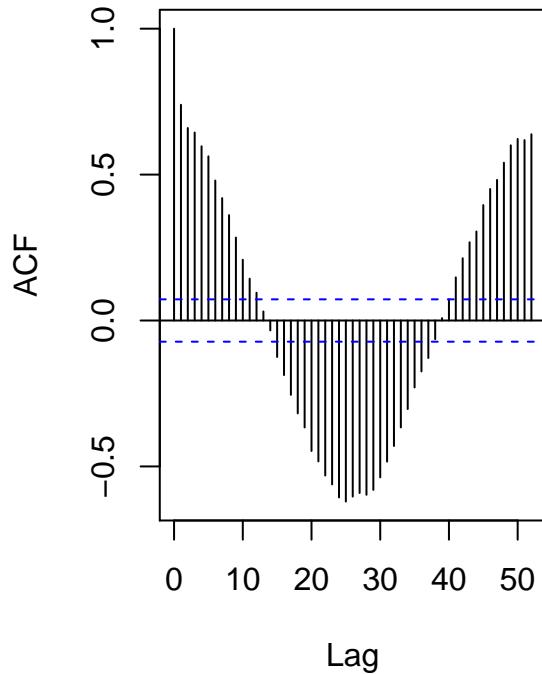


Same problems as before. Holt-Winters doesn't appear to work well in this case.

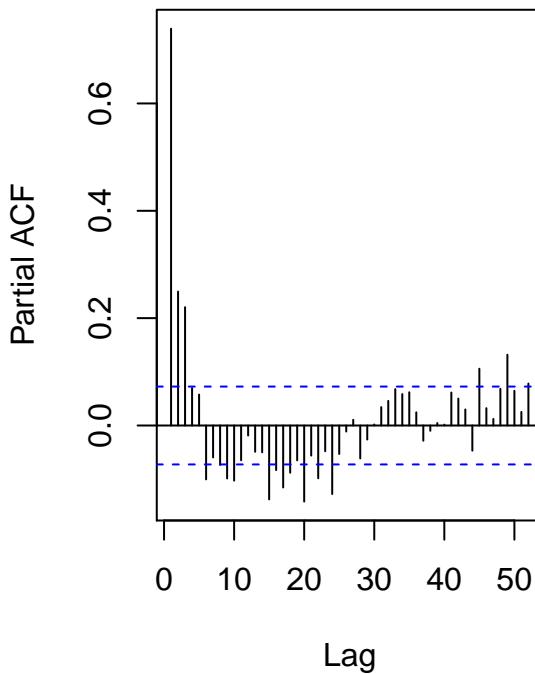
-### ARMA

We can see the ACF and PACF:

ACF of the data



PACF of the data



Based on these graphs, we see both graphs has a exponentially decay, the first after the $p - q = 1$ or $p - q = 2$. In order to identify the model, we will compare the adjusted ARMA models with different p and q . First we simply fit it to look at the Akaike Information Criteria (AIC) and the significance of the parameters estimated.

```
##
## Call:
## arma(x = o3_train_week, order = c(2, 0))
##
## Model:
## ARMA(2,0)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.0151516 -0.0032246 -0.0002512  0.0028414  0.0257926
##
## Coefficient(s):
##             Estimate Std. Error t value Pr(>|t|)
## ar1        0.5546040  0.0358395 15.475 < 2e-16 ***
## ar2        0.2498525  0.0358459  6.970 3.17e-12 ***
## intercept  0.0039316  0.0005489  7.162 7.93e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Fit:
## sigma^2 estimated as 2.257e-05,  Conditional Sum-of-Squares = 0.02,  AIC = -5732.5
##
## Call:
## arma(x = o3_train_week, order = c(2, 1))
```

```

## 
## Model:
## ARMA(2,1)
## 
## Residuals:
##      Min       1Q   Median      3Q     Max
## -0.014583 -0.003300 -0.000134  0.002970  0.025345
## 
## Coefficient(s):
##             Estimate Std. Error t value Pr(>|t|)    
## ar1        0.9751555  0.0700310 13.925 < 2e-16 ***
## ar2       -0.0463356  0.0607827 -0.762   0.446    
## ma1       -0.4952763  0.0585446 -8.460 < 2e-16 ***
## intercept  0.0014296  0.0003658  3.908 9.29e-05 *** 
## --- 
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Fit:
## sigma^2 estimated as 2.162e-05, Conditional Sum-of-Squares = 0.02, AIC = -5761.97
## 
## Call:
## arma(x = o3_train_week, order = c(3, 1))
## 
## Model:
## ARMA(3,1)
## 
## Residuals:
##      Min       1Q   Median      3Q     Max
## -0.0140809 -0.0032711 -0.0001538  0.0029207  0.0251467
## 
## Coefficient(s):
##             Estimate Std. Error t value Pr(>|t|)    
## ar1        0.7724162  0.1110272  6.957 3.48e-12 ***
## ar2       -0.0260143  0.0701409 -0.371  0.71072    
## ar3        0.1551277  0.0497065  3.121  0.00180 **  
## ma1       -0.2899243  0.1088996 -2.662  0.00776 **  
## intercept  0.0019742  0.0005461  3.615  0.00030 *** 
## --- 
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Fit:
## sigma^2 estimated as 2.133e-05, Conditional Sum-of-Squares = 0.02, AIC = -5769.92
## Warning in arma(o3_train_week, order = c(3, 2)): Hessian negative-semidefinite
## Warning in sqrt(diag(object$vcov)): NaNs produced
## Warning in sqrt(diag(object$vcov)): NaNs produced
## 
## Call:
## arma(x = o3_train_week, order = c(3, 2))
## 
## Model:
## ARMA(3,2)

```

```

## 
## Residuals:
##      Min       1Q    Median       3Q      Max
## -1.401e-02 -3.242e-03 -9.053e-05  2.926e-03  2.527e-02
## 
## Coefficient(s):
##             Estimate Std. Error t value Pr(>|t|)    
## ar1        0.6314855     NA        NA        NA    
## ar2        0.1161892     NA        NA        NA    
## ar3        0.1431114  0.0393657  3.635  0.000278 *** 
## ma1       -0.1477659     NA        NA        NA    
## ma2       -0.0693593     NA        NA        NA    
## intercept  0.0021847  0.0004249  5.142 2.71e-07 *** 
## ---      
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Fit:
## sigma^2 estimated as 2.135e-05, Conditional Sum-of-Squares = 0.02, AIC = -5767.21
## Warning in arma(o3_train_week, order = c(4, 2)): Hessian negative-semidefinite
## Warning in arma(o3_train_week, order = c(4, 2)): NaNs produced
## Warning in arma(o3_train_week, order = c(4, 2)): NaNs produced
## Call:
## arma(x = o3_train_week, order = c(4, 2))
## 
## Model:
## ARMA(4,2)
## 
## Residuals:
##      Min       1Q    Median       3Q      Max
## -0.0140735 -0.0032840 -0.0001482  0.0029149  0.0251517
## 
## Coefficient(s):
##             Estimate Std. Error t value Pr(>|t|)    
## ar1        0.5552339     NA        NA        NA    
## ar2        0.1374963     NA        NA        NA    
## ar3        0.1507390  0.0504035  2.991  0.00278 **  
## ar4        0.0362919  0.0564229  0.643  0.52009    
## ma1       -0.0750043     NA        NA        NA    
## ma2       -0.0568046     NA        NA        NA    
## intercept  0.0024065  0.0005688  4.231 2.33e-05 *** 
## ---      
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Fit:
## sigma^2 estimated as 2.134e-05, Conditional Sum-of-Squares = 0.02, AIC = -5765.57
## Warning in arma(o3_train_week, order = c(4, 3)): Hessian negative-semidefinite
## Warning in arma(o3_train_week, order = c(4, 3)): NaNs produced

```

```

## Warning in arma(o3_train_week, order = c(4, 3)): NaNs produced
##
## Call:
## arma(x = o3_train_week, order = c(4, 3))
##
## Model:
## ARMA(4,3)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.01344441 -0.0031889 -0.0001146  0.0029098  0.0249980
##
## Coefficient(s):
##             Estimate Std. Error t value Pr(>|t|)    
## ar1        0.7282249     NA        NA        NA      
## ar2        0.4180584     NA        NA        NA      
## ar3       -0.4544918    0.06666640 -6.818 9.25e-12 ***
## ar4        0.2009348     NA        NA        NA      
## ma1       -0.2669488     NA        NA        NA      
## ma2       -0.4451386     NA        NA        NA      
## ma3        0.4472807    0.1095129  4.084 4.42e-05 ***
## intercept  0.0021247    0.0004557  4.662 3.13e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Fit:
## sigma^2 estimated as 2.09e-05, Conditional Sum-of-Squares = 0.02, AIC = -5778.63

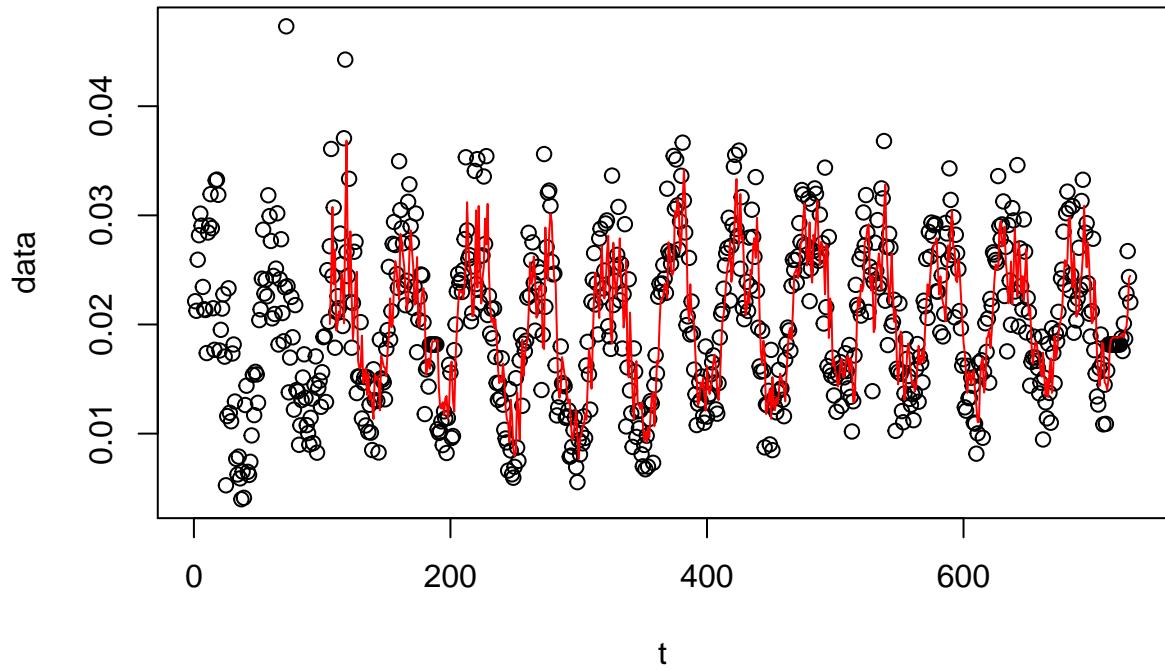
```

We see that from the second onwards, we have parameters without statistical significance. However the second and third models have good AIC either. The last three has problems with the Hessian, so the p-values could not be calculated. For that reason, we shall disregard them and compare the others. In special we'll test ARMA(3,1), AR(2) and ARMA(2,1).

We see the MAE of the AR(2):

```
## [1] 0.003525595
```

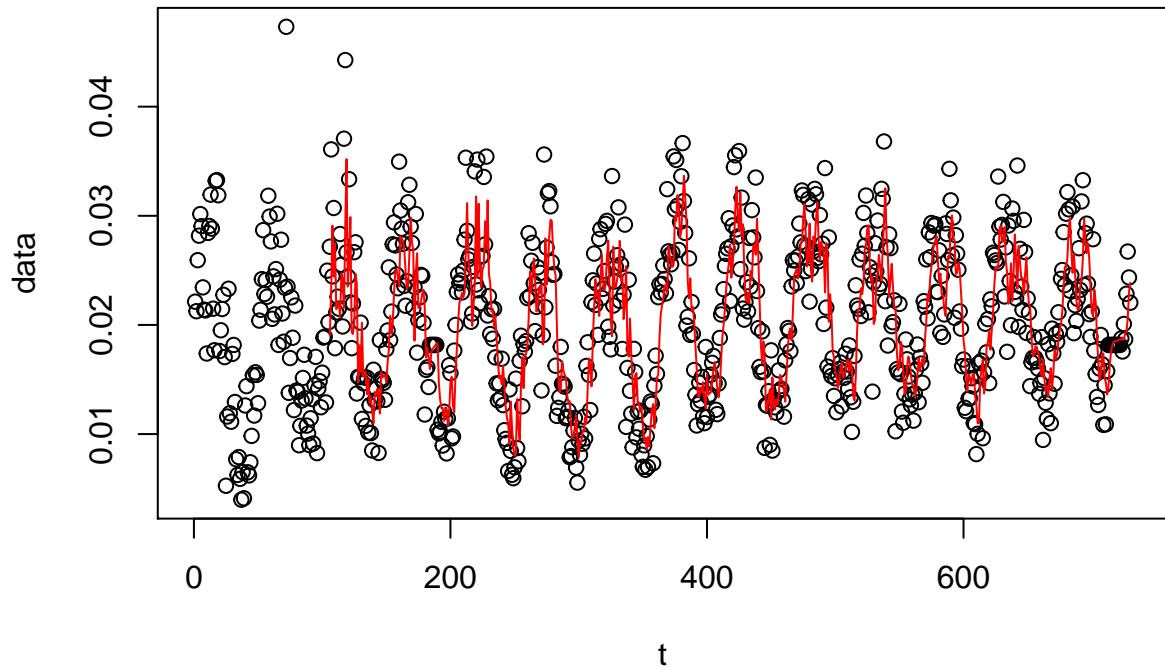
ARIMA(3,0,1) prediction



And the MAE of the ARMA(2, 1):

```
## [1] 0.003507218
```

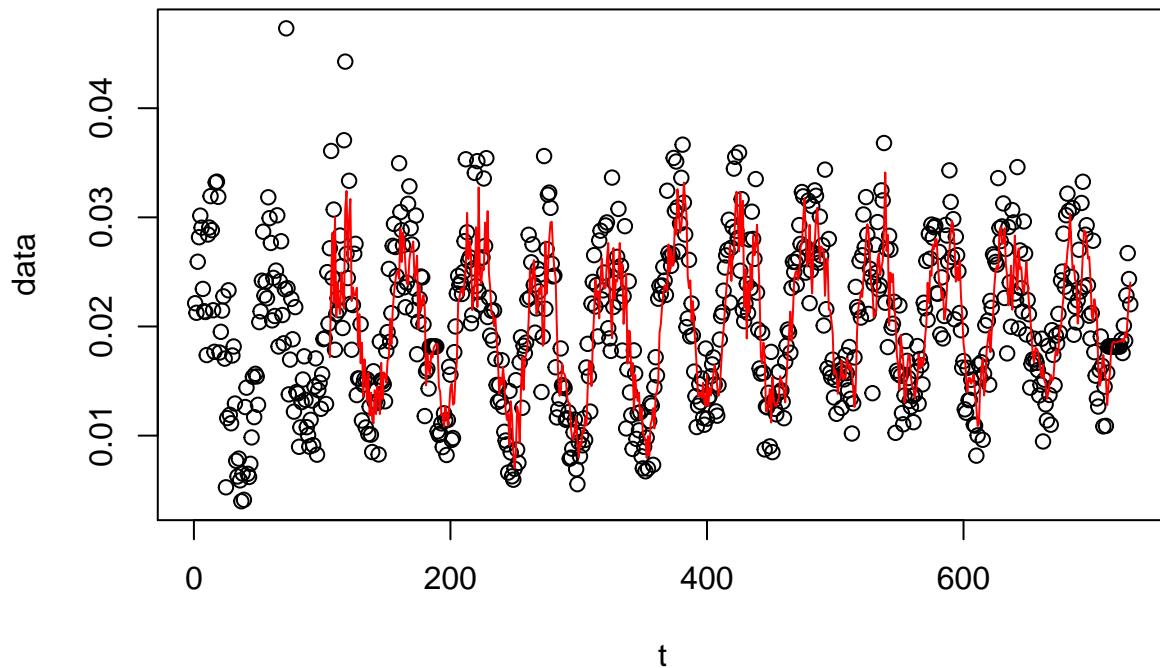
ARIMA(2,0,1) prediction



And the MAE of the ARMA(3, 1):

```
## [1] 0.003558917
```

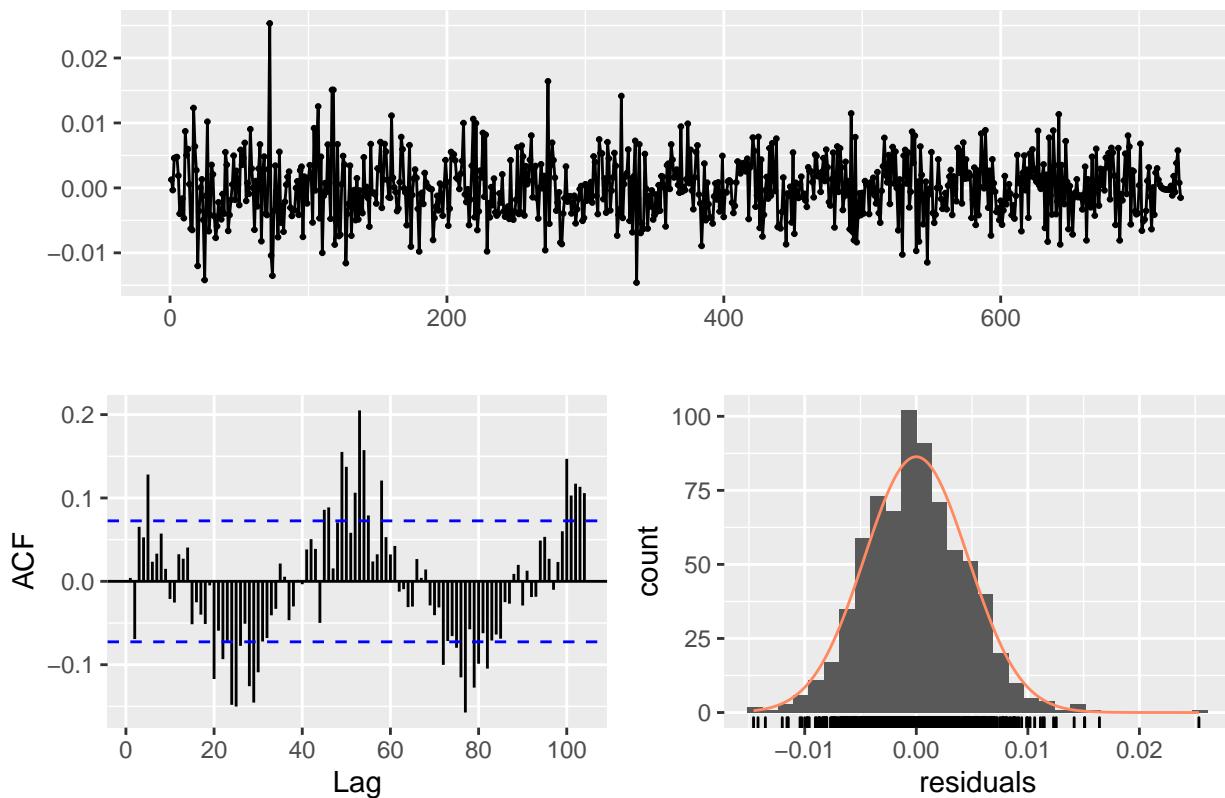
ARIMA(3,0,1) prediction



The second model seems a little better. So, let's check the residuals to observe it's problems.

```
model <- arima(o3_train_week, order = c(2,0,1))
checkresiduals(model, lag = 2*freq, lag.max = 2*freq)
```

Residuals from ARIMA(2,0,1) with non-zero mean



```
##  
## Ljung-Box test  
##  
## data: Residuals from ARIMA(2,0,1) with non-zero mean  
## Q* = 465.72, df = 100, p-value < 2.2e-16  
##  
## Model df: 4. Total lags used: 104
```

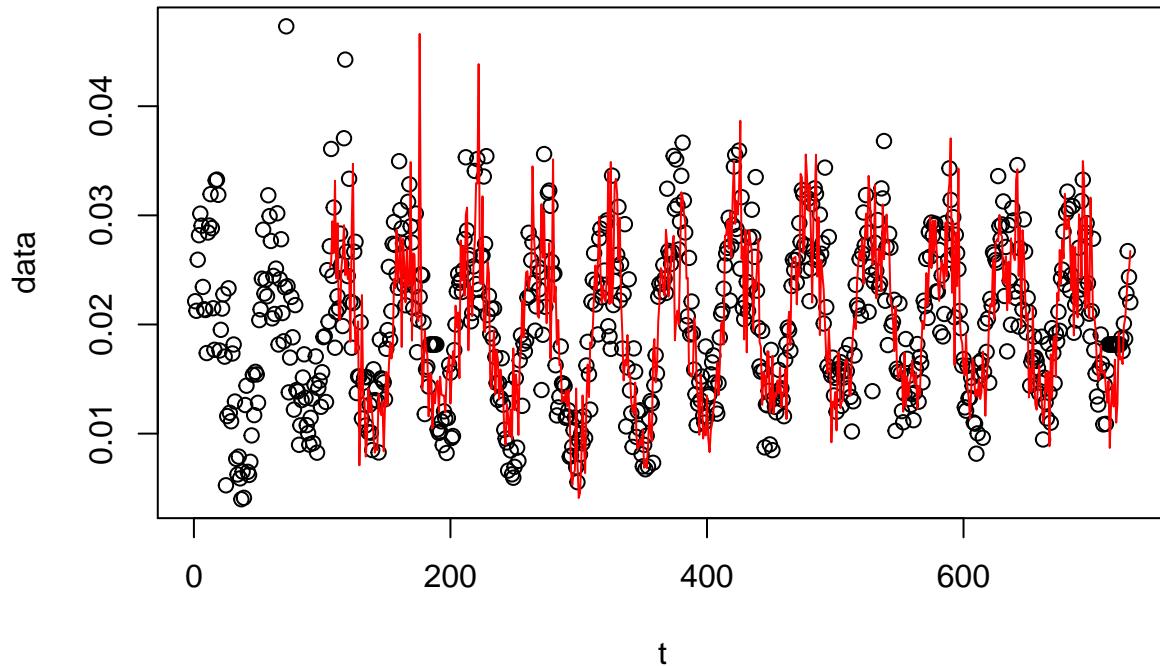
It's interesting to note the ACF has a peak around the 52, so the ARMA model did not seem to capture the seasonality. It would be better to fit an Seasonal ARIMA further. The histogram is pretty similar to normal distribution. The test made analyses if the mean is 0. It may be because the p-value is really small. The ACF also suggests that we include a high order parameter in MA and AR. However, in trying to fit an ARMA(3,2), we struggle with stationarity in AR part. For that reason, the model has it fails and alone it seems that it cannot be improved.

Adapting ARMA

The ARMA seems to fit well as we can see so far. However, it's not capturing other characteristics on the data, as seasonality as mentioned. For that reason, we will combine the STL and ARMA model and extract the best of each one. We will decompose the series in trend and seasonality and in the reminder, we fit an arima model with `auto.arima()`.

```
## [1] 0.003987221
```

STL + ARIMA prediction



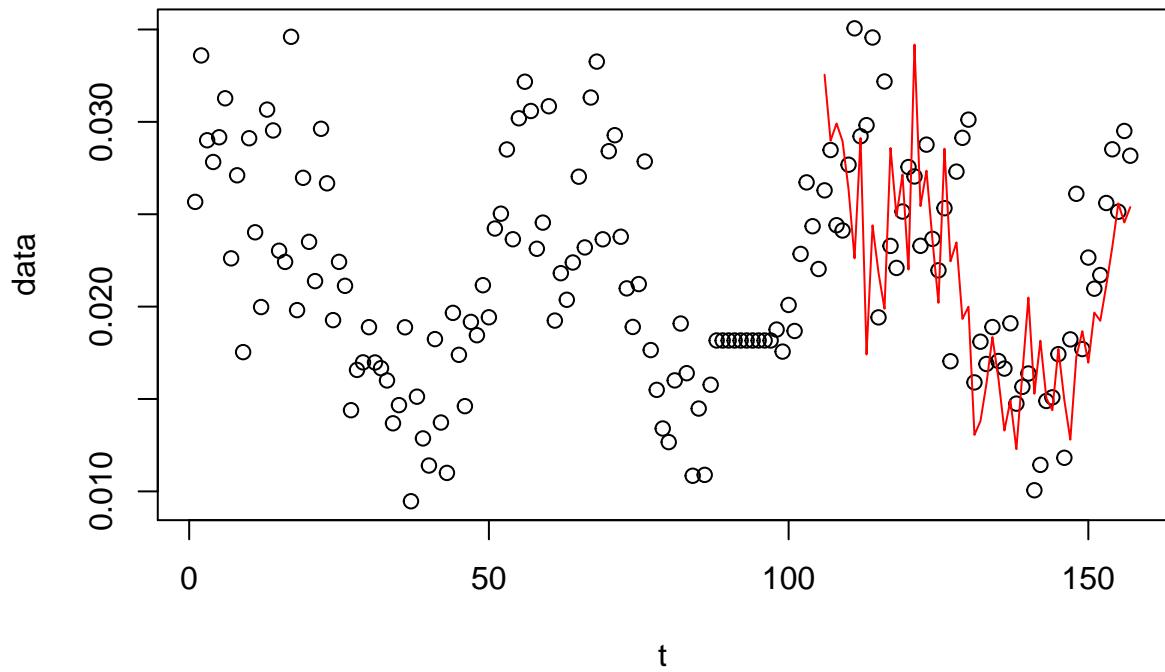
In spite of the improvement in the STL model, the MAE isn't improved by this model when compared to ARMA best model. For that reason, this model was disregarded.

Comparing models in the test data.

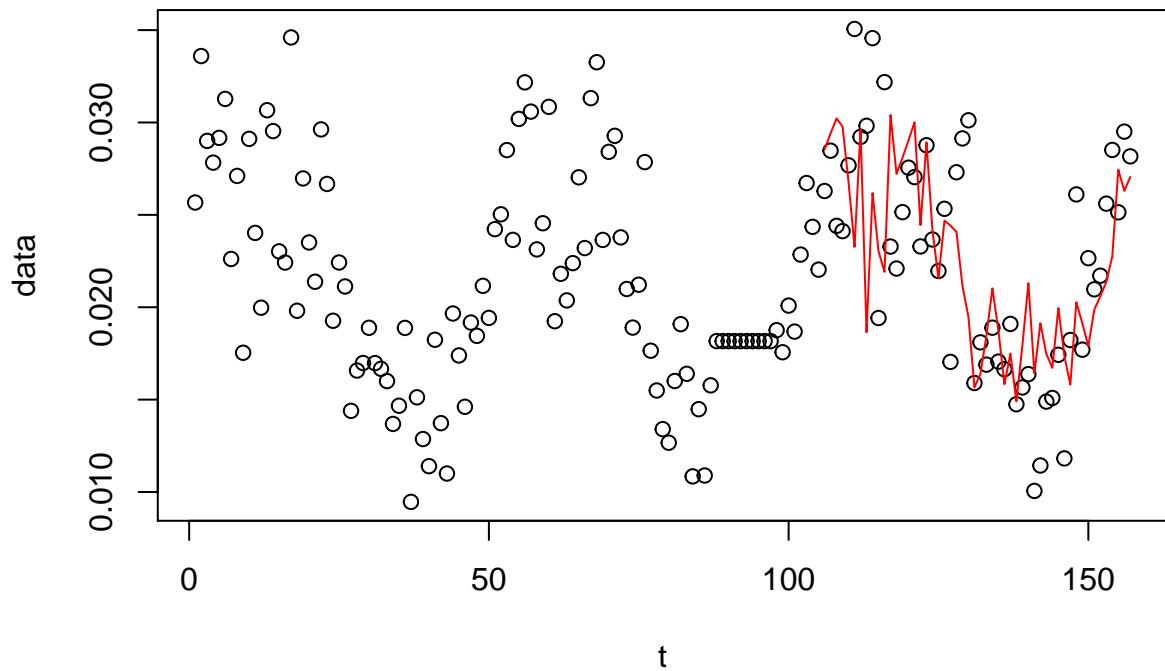
We chose some of the best models to compare with the testing data.

```
## [1] 0.004080176
```

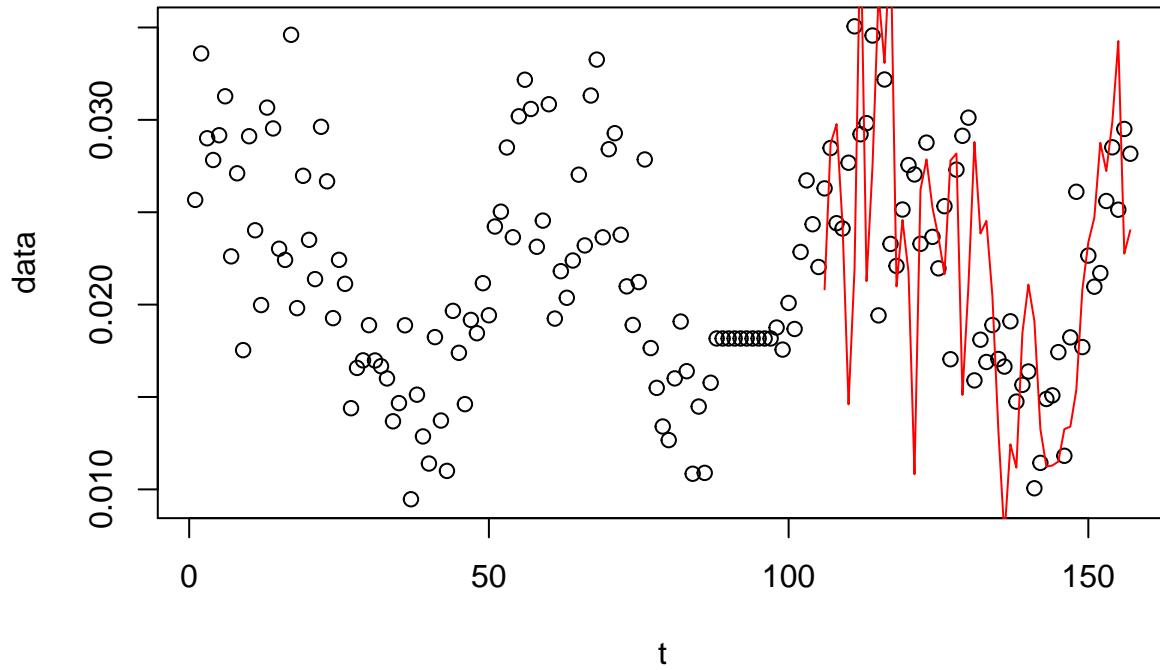
Multiplicative decompose prediction



Regression prediction

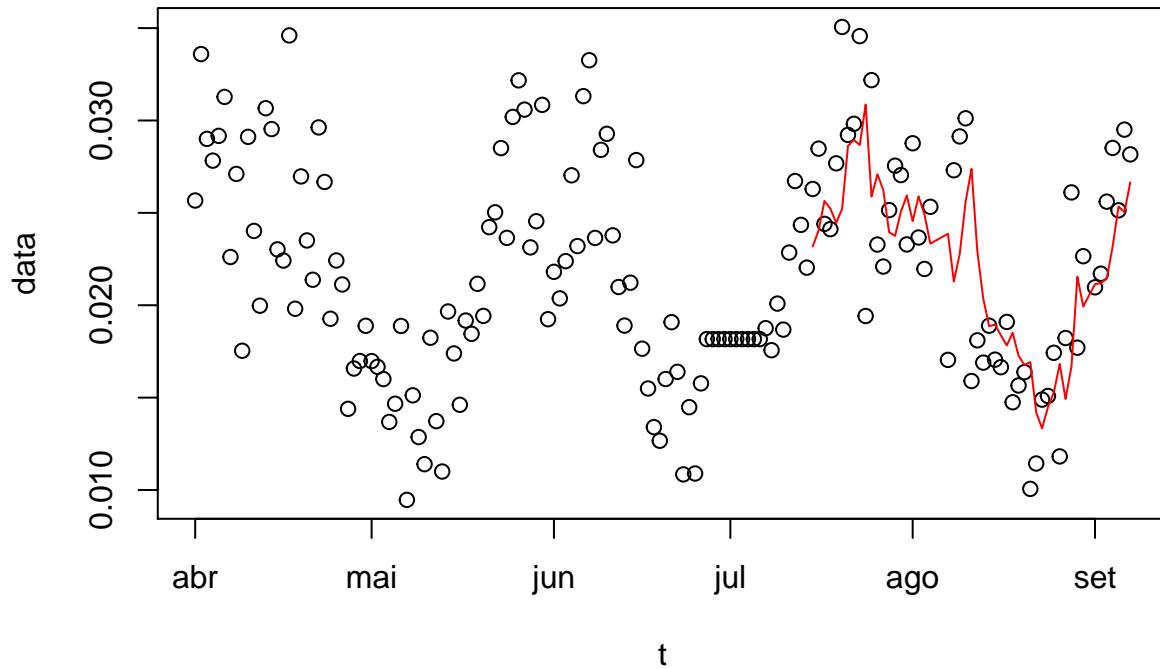


Multiplicative Holt–Winters prediction



```
## [1] 0.003572499
```

ARIMA(2,0,1) prediction



Finally, we have this bar graphic to compare the values:

Comparando MAE para os modelos

