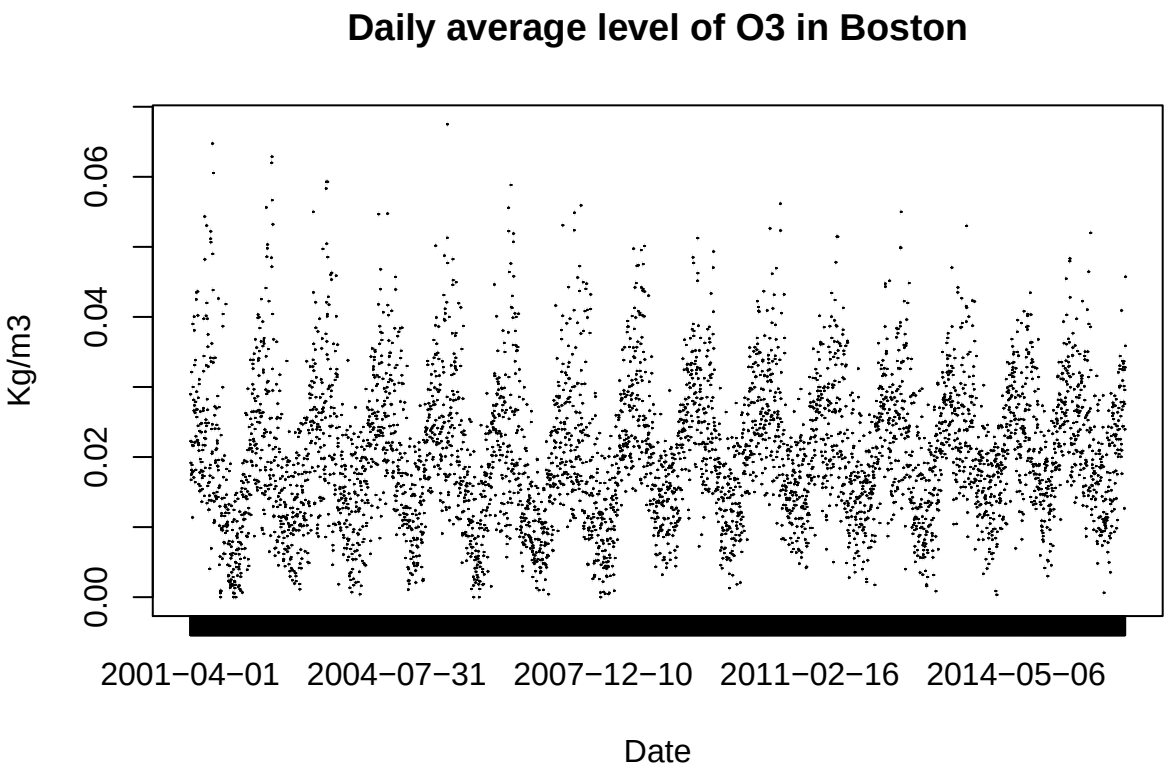
Prediction of ozone level in Boston

Lucas Emanuel Resck Domingues[*]
Lucas Machado Moschen[†]

Load and visualize



Methodology

1. Data treatment: analyse missing and duplicated data.
2. Compare the daily models: we use rollapply using 2 years + 6 days to predict the day, that is, we use $\{X_t\}_{t-2\cdot 365}^{t}$ to predict X_{t+7}. We calcute, for each $t > 2 \cdot 365 + 6$, $|\hat{X}_t - X_t|$ and after calculate $\sum_t |\hat{X}_t - X_t|$. We used Decomposition, Regression, Holt-Winters and ARMA.
3. Compare the weekly models: we use rollapply using 2 years + 4 weeks to predict the week, that is, we use $\{X_t\}_{t-2\cdot 52}^{t}$ to predict X_{t+4}. We calcute, for each $t > 2 \cdot 52 + 4$, $|\hat{X}_t - X_t|$ and after calculate $\sum_t |\hat{X}_t - X_t|$. We used Decomposition, Regression, Holt-Winters and ARMA.
4. We compare for the first the case the MAE over the 7 days of the prediction forward.
5. Compare the best models of each type for the 2 cases in the test data using the same approach.

Obs.: When applying logarithm, we applied $\log(X_t + 1)$, because there are 0 values, or almost zero, causing numerical problems. When we compare, however, we do the opposite: $\exp\{\hat{X}_t\} - 1$

[*]Escola de Matemática Aplicada
[†]Escola de Matemática Aplicada

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  O3.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  O3.Mean
## 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
```

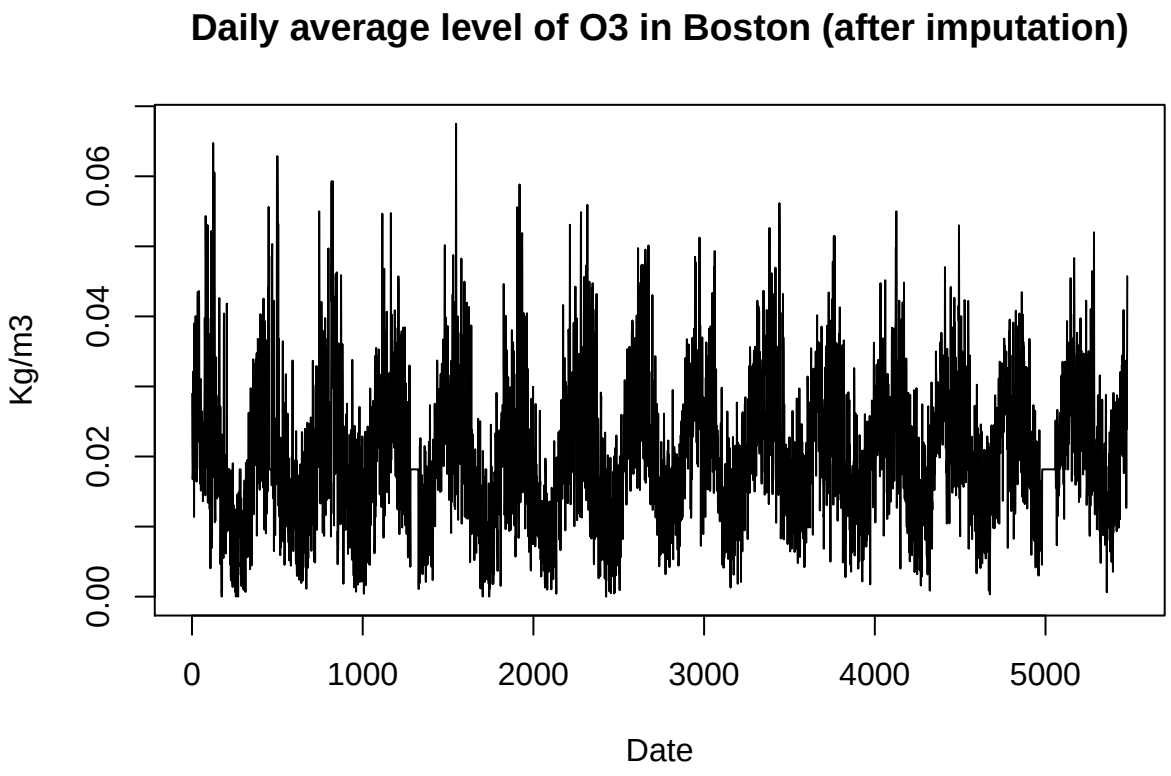
The duplicated one is easier to deal, but the missing values are harder. First we calculate the mean value between the duplicated.

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')
```



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

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

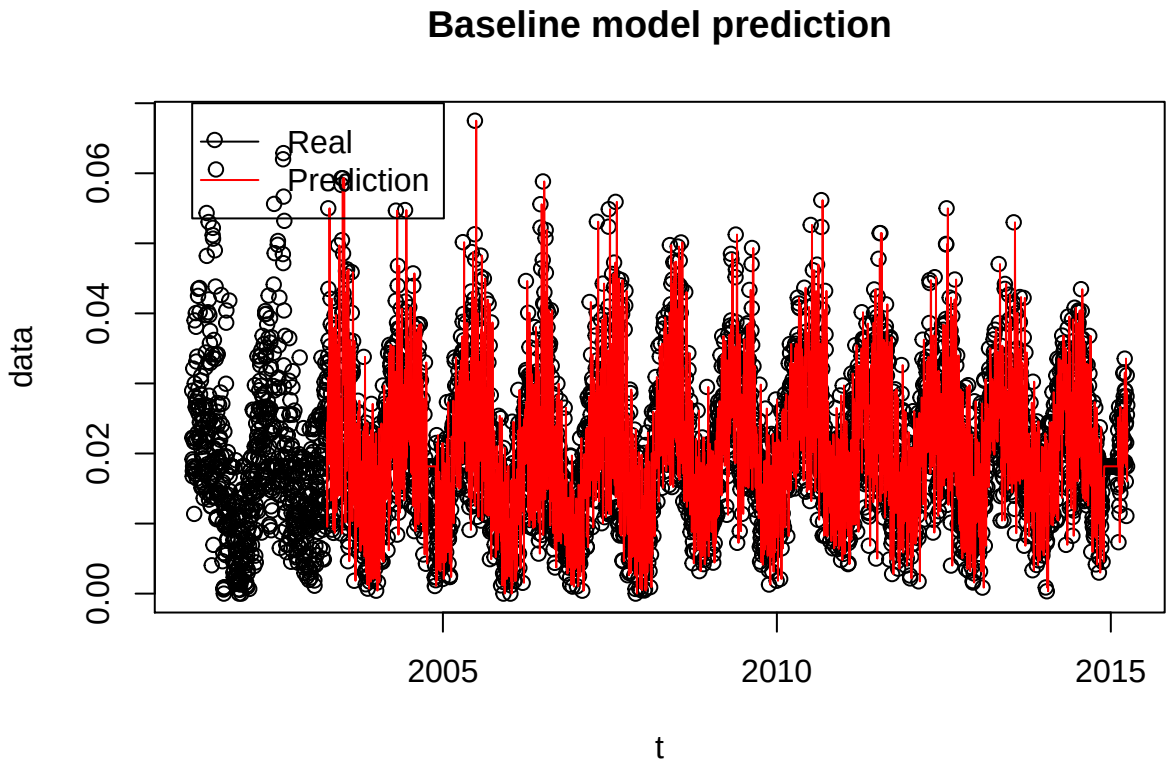
```
mae <- function(ytrue, ypred)
{
    return(mean(abs(ytrue - ypred)))
}
```

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. It gets $\hat{X}_{t+7} = X_t$.

```
## [1] 0.007844892
```



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.

```
##
```

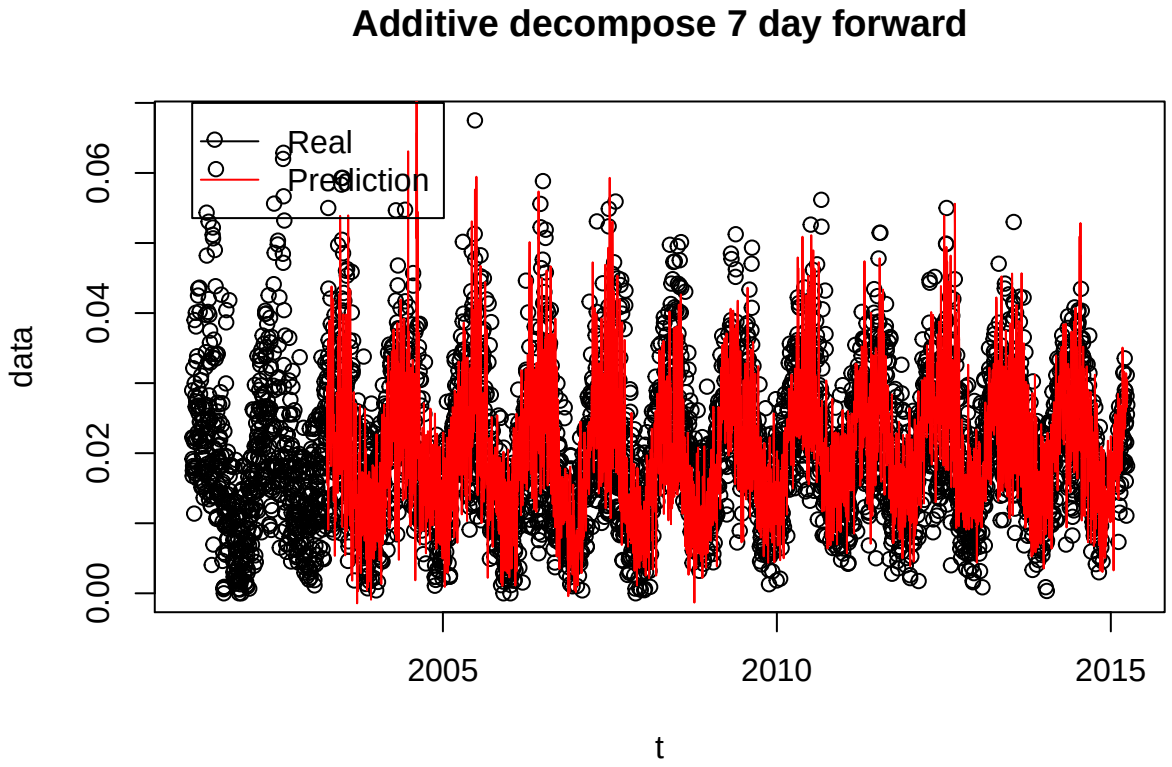
```
##  Kruskal-Wallis rank sum test
##
## data:  o3_train and g
## Kruskal-Wallis chi-squared = 32.114, df = 30, p-value = 0.3622
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  o3_train and g
## Kruskal-Wallis chi-squared = 2179.6, df = 364, p-value < 2.2e-16
```

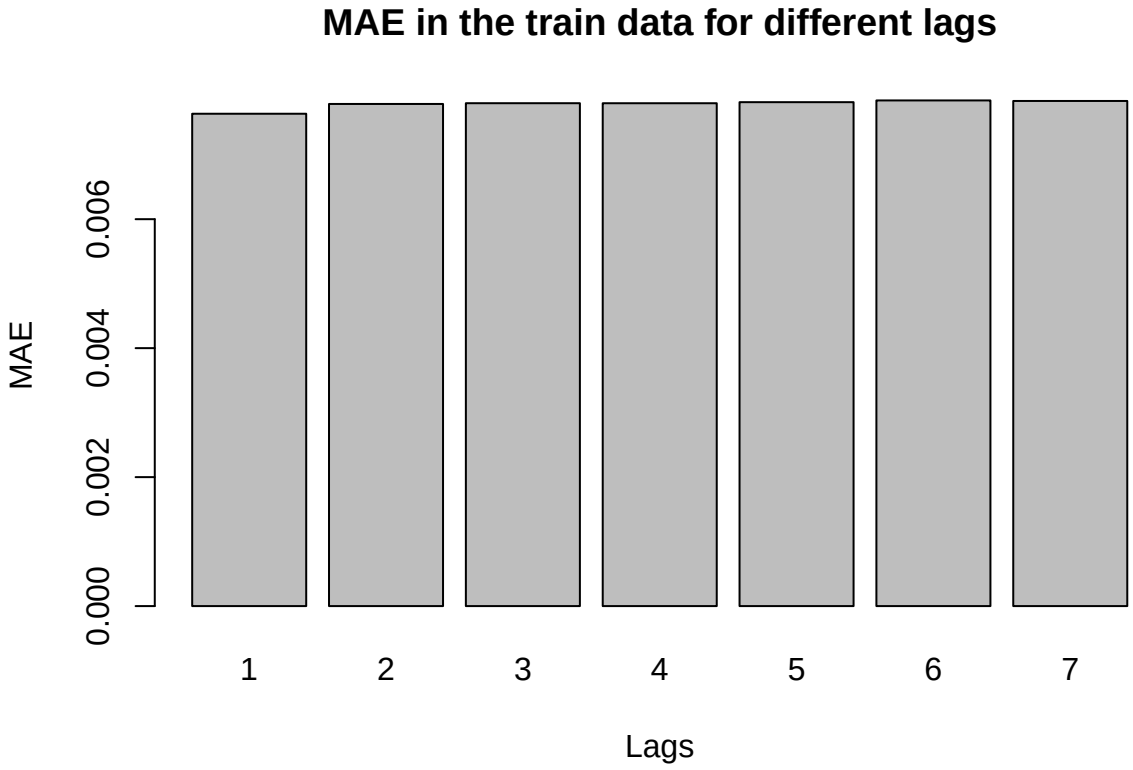
Additive model

First we analyse the MAE.

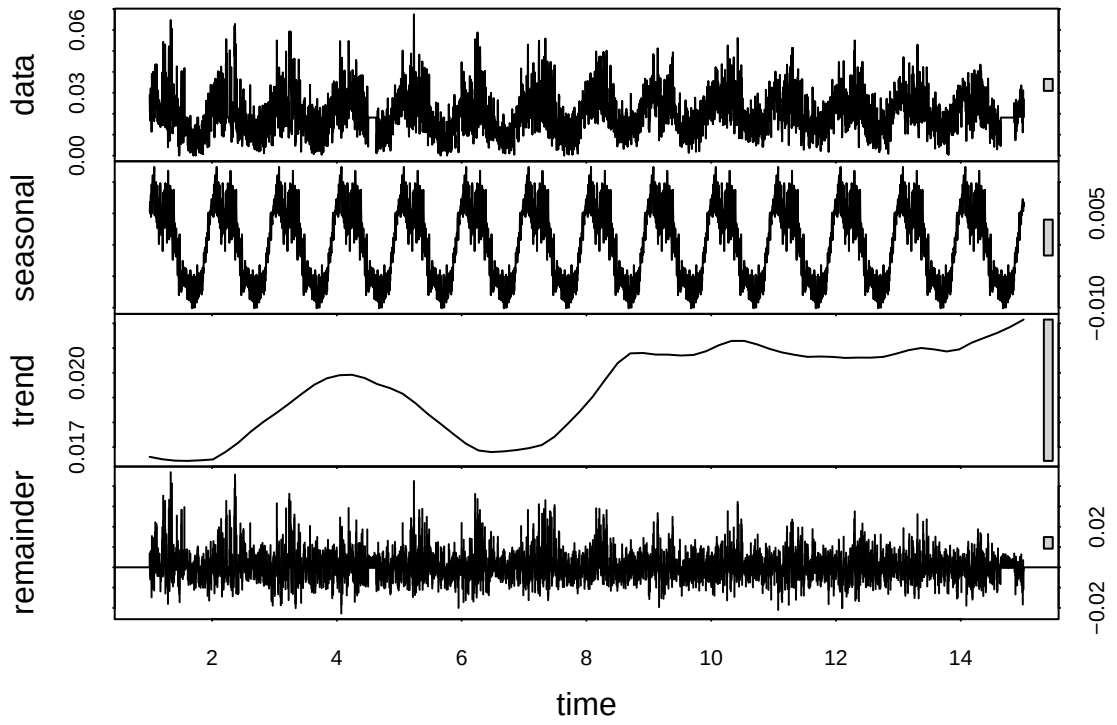
```
## [1] 0.007832648
```



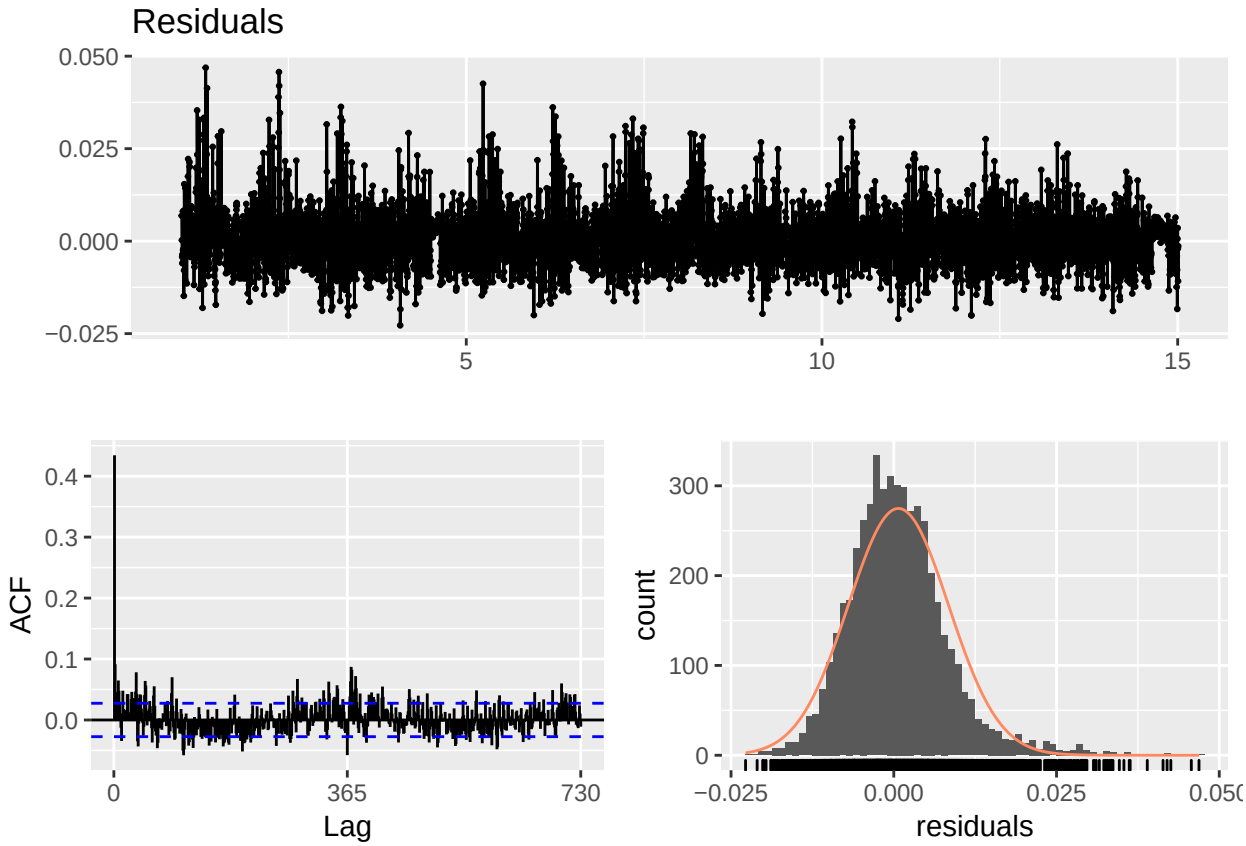
Next we compare the MAE chan changing the day from 1 to 7.



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



The ACF and the PACF of the reminder:

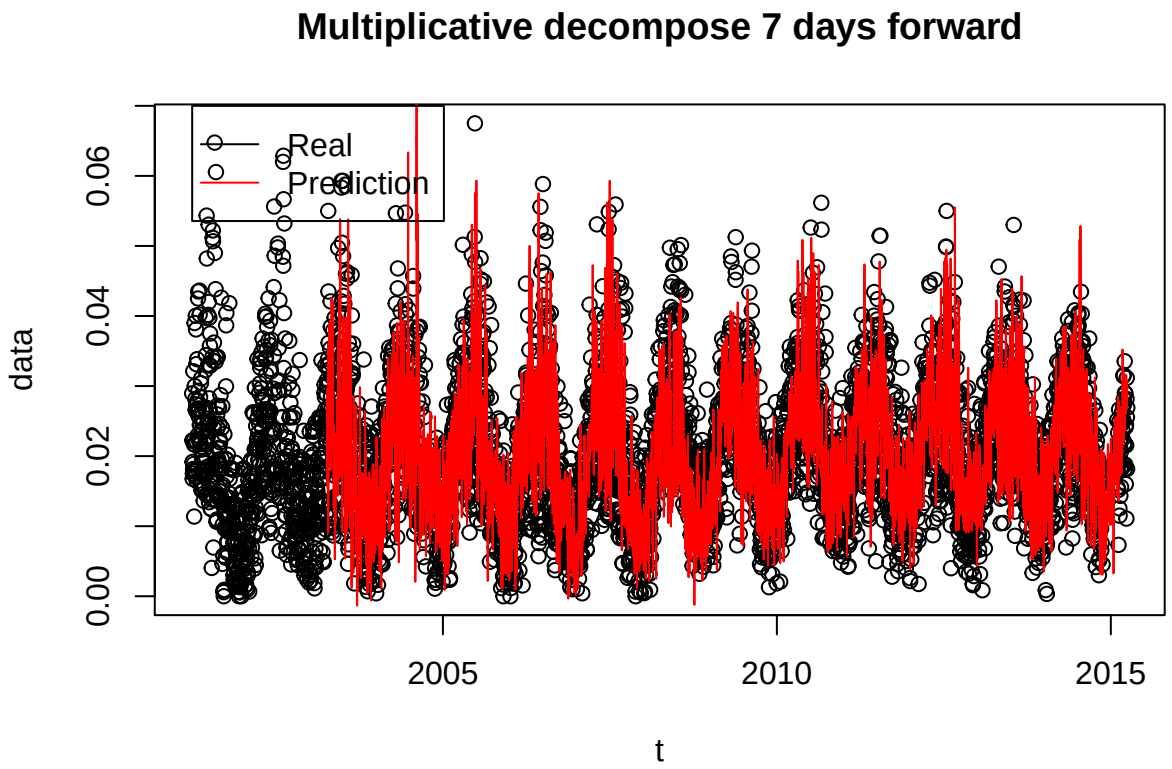


We see that there are a big spike when lag = 365. It seems not so good for a reminder. We could fit an ARMA model in this reminder yet. We also see a similar normal distribution, but the right tail is a little strange for it. It's a good model, but it can be improved.

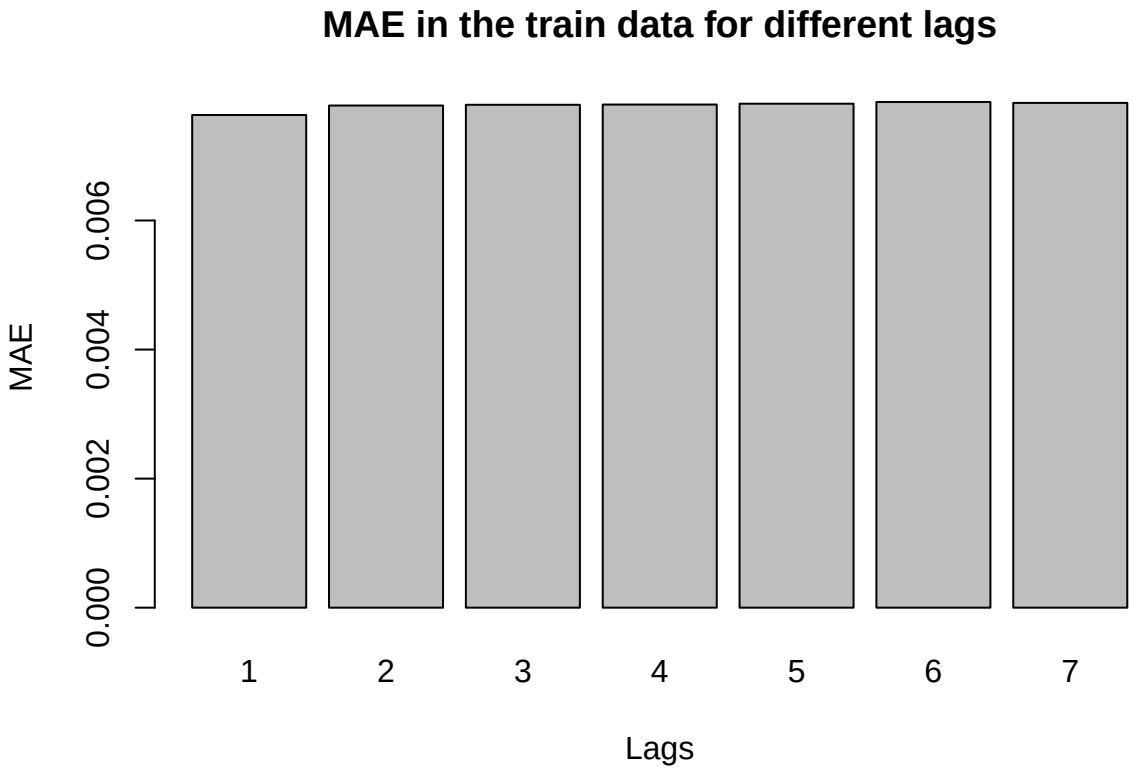
Multiplicative model

First we analyse the MAE using the `rollapply` again.

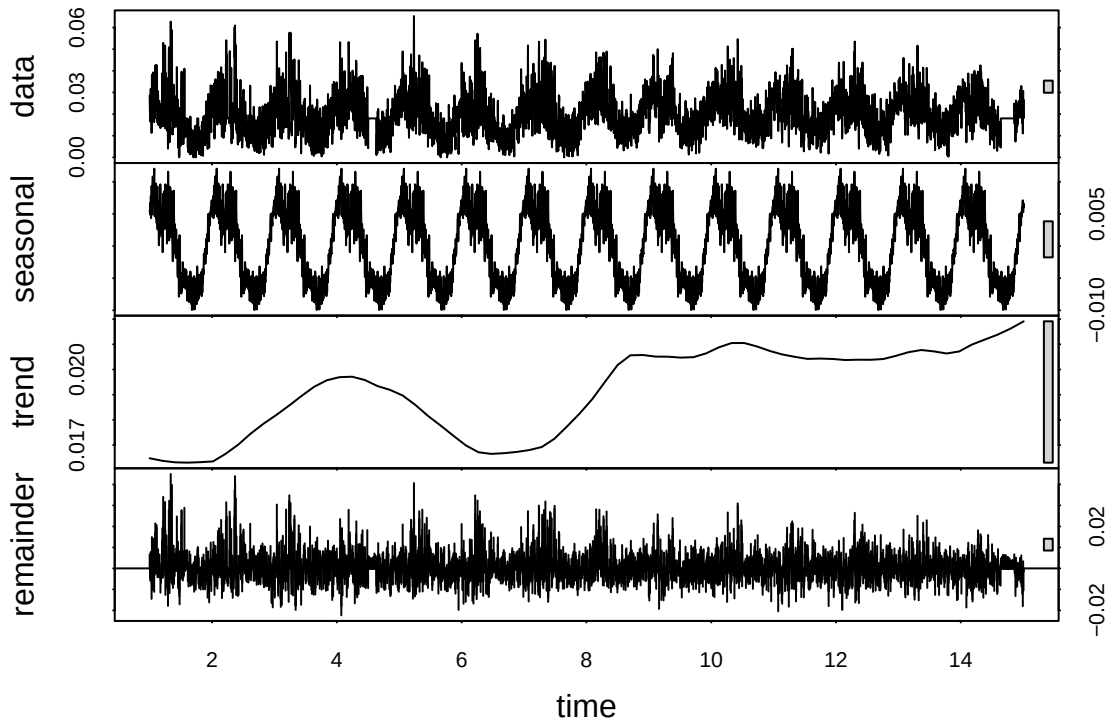
```
## [1] 0.007822273
```



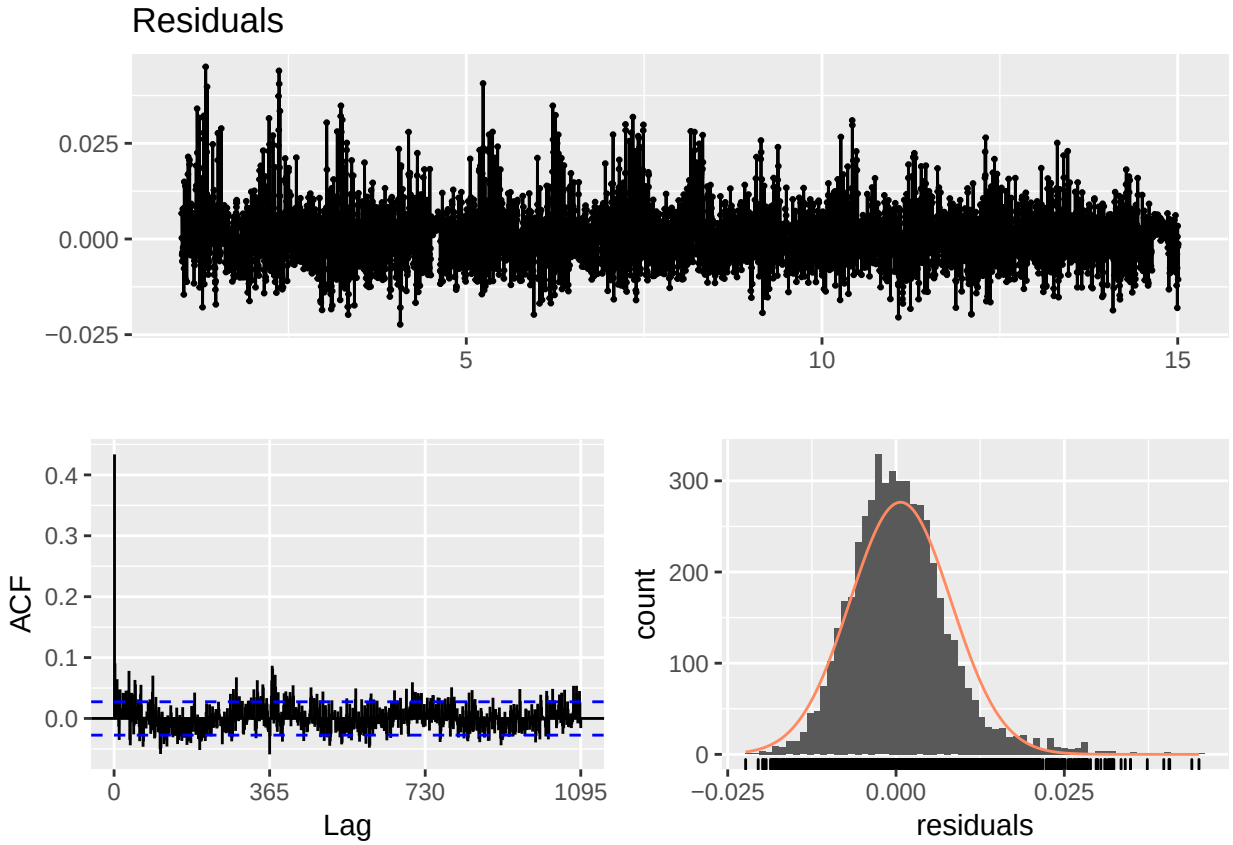
Next we compare the MAE chan changing the day from 1 to 7.



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



The ACF and the PACF of the reminder:

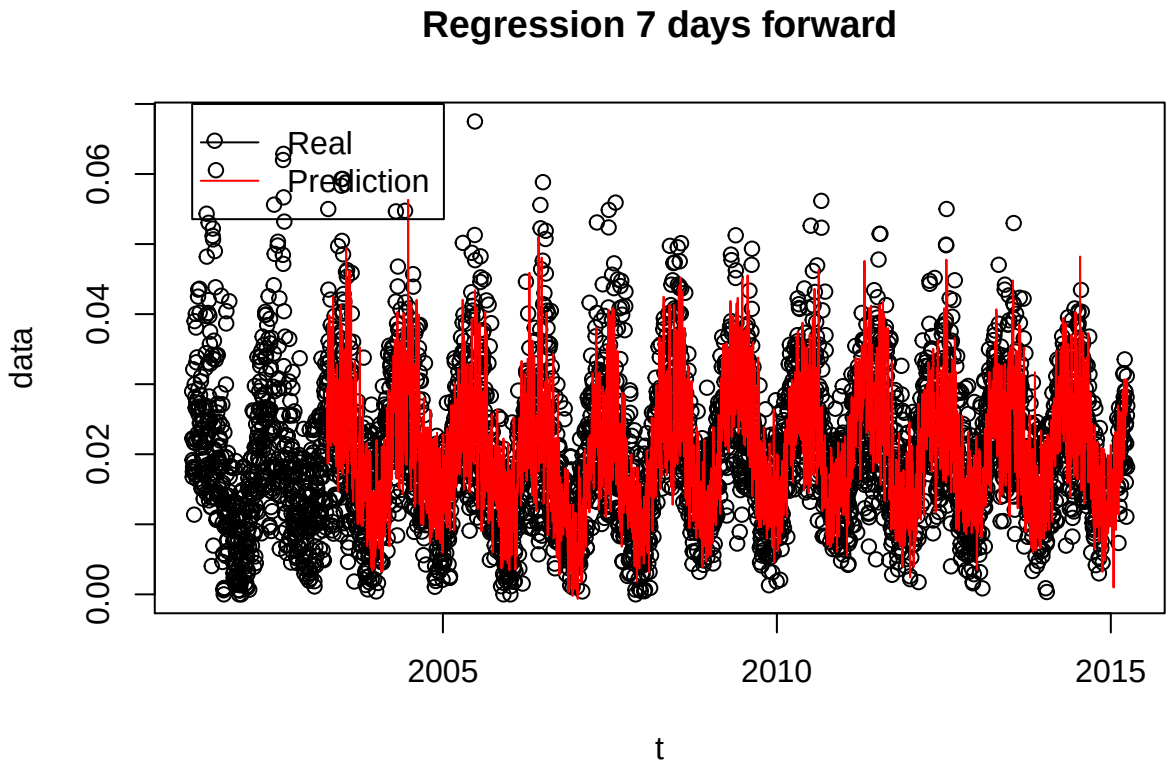


We see that there are a big spike when lag = 365. It seems not so good for a reminder. We could fit an ARMA model in this reminder yet. The same problem as before.

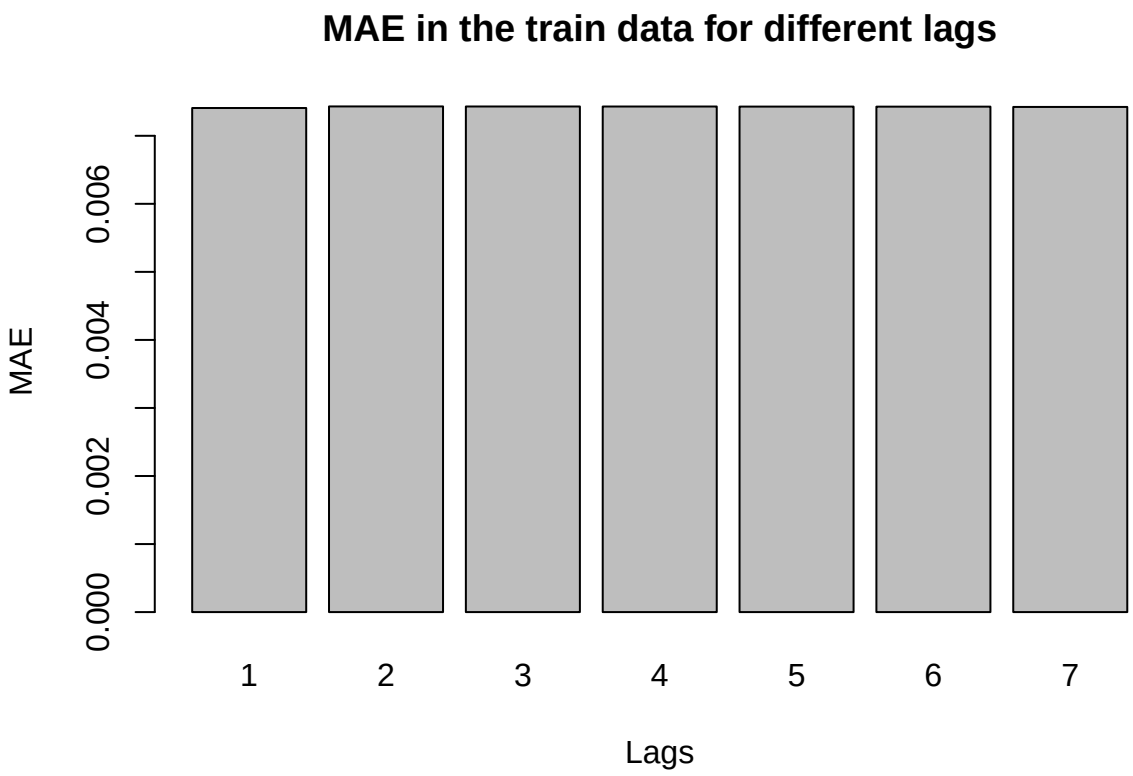
Regression

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] 0.007423855
```

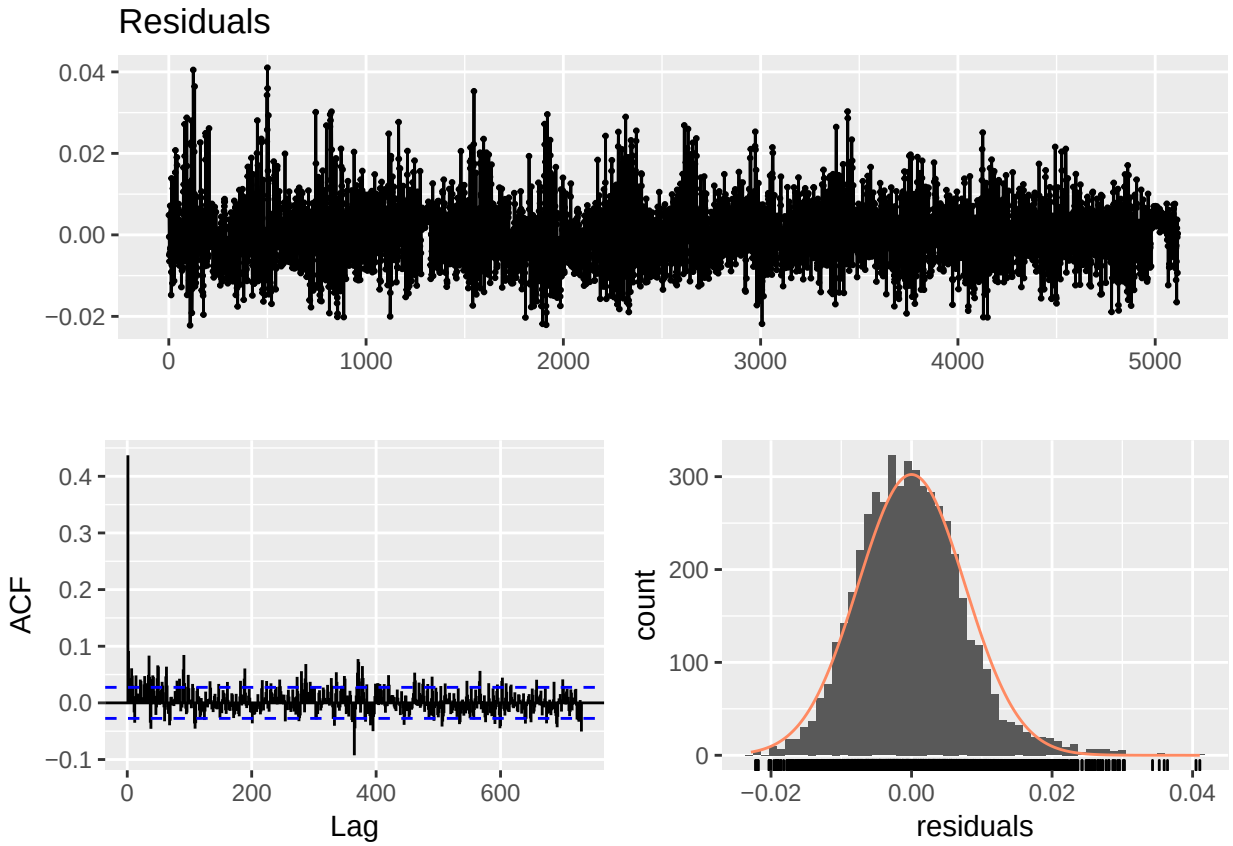


Next we compare the MAE chan changing the day from 1 to 7.



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)
```




```
##
##  Breusch-Godfrey test for serial correlation of order up to 730
##
## data:  Residuals
## LM test = 1747.3, df = 730, p-value < 2.2e-16
```

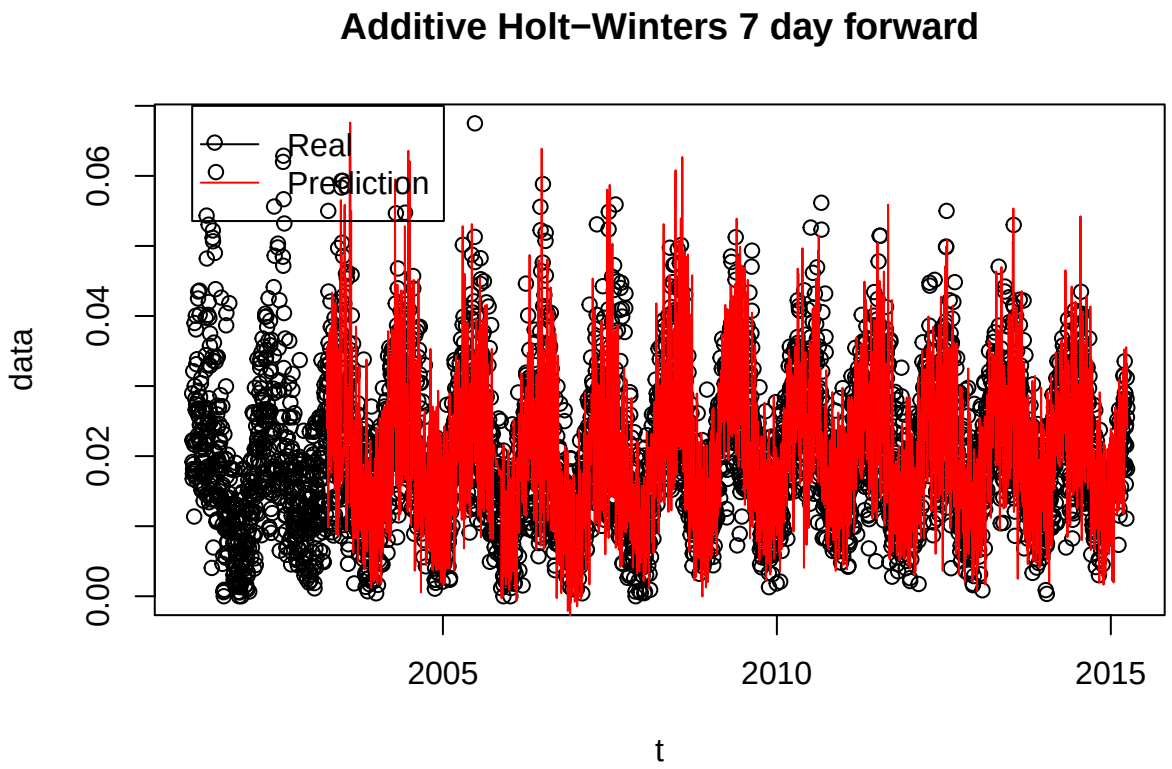
As before, wee see spikes in lag $= 365, 730$. We expect a WN to not have this.

Holt-Winters

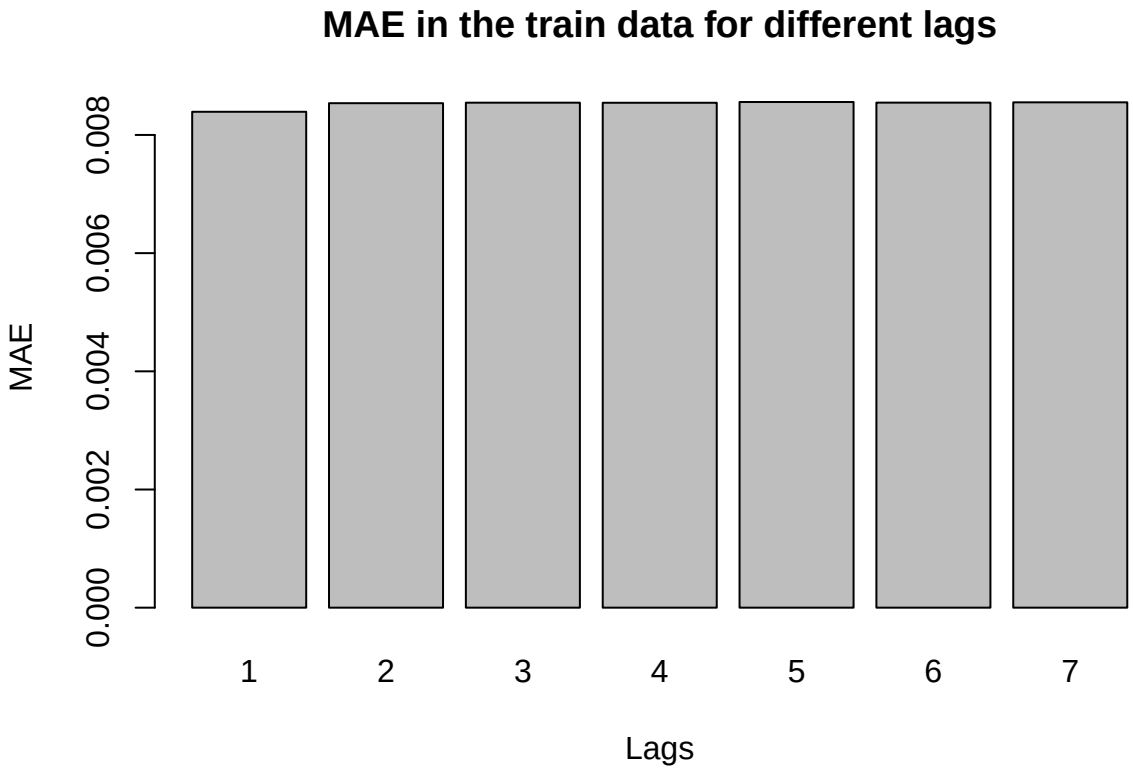
Now we will try Holt-Winters models. In fact, because of apparently seasonality, we will consider complete Holt-Winters models, both additive and multiplicative.

Additive

```
## [1] 0.00855174
```

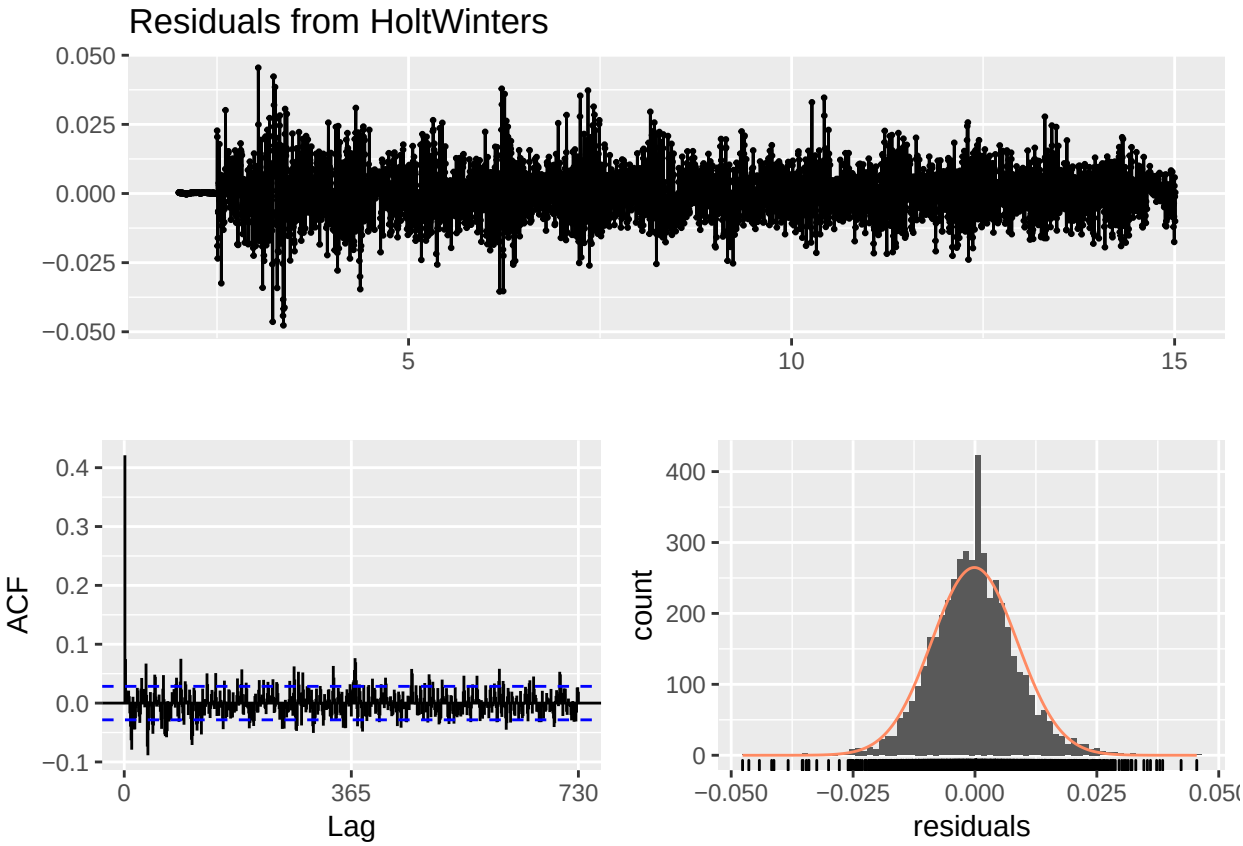


Next we compare the MAE chan changing the day from 1 to 7.



MAE is kind of good, so is the resulting graph, but we have seem better results. Let's analyse the residuals:

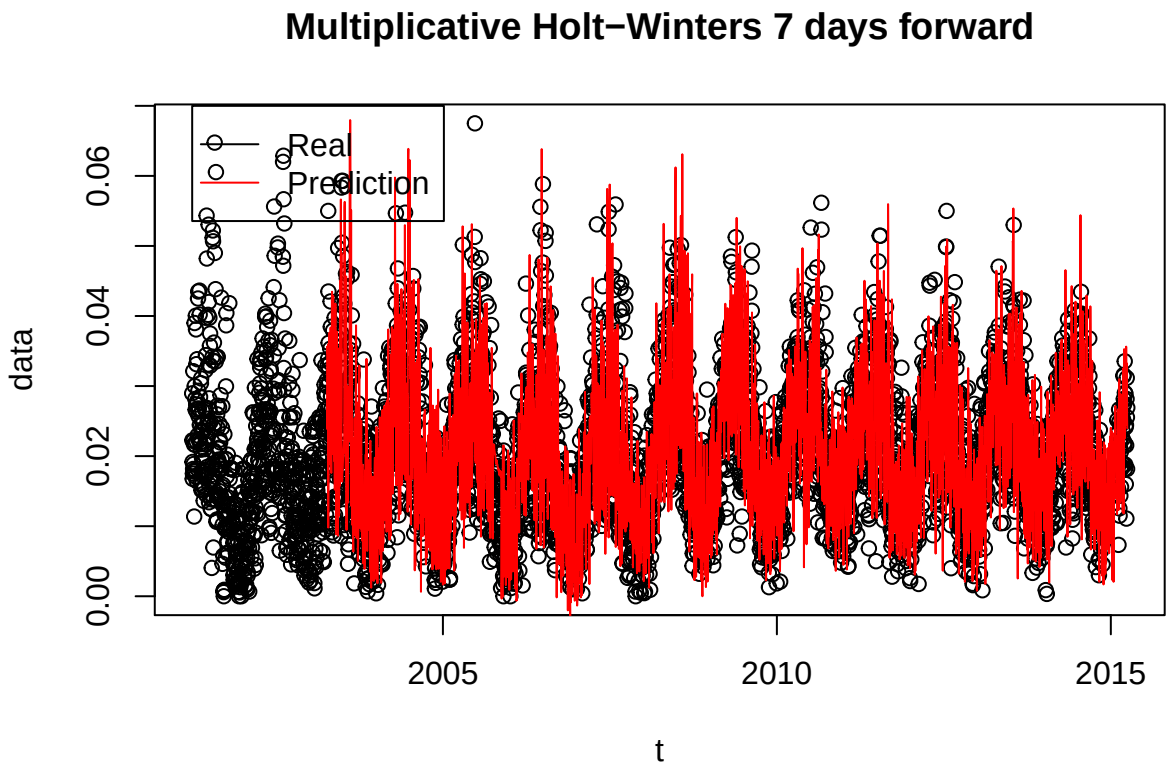
```
mod = HoltWinters(ts(o3_train, frequency = 365), seasonal = "additive")
checkresiduals(mod, lag = 2*freq, lag.max = 2*freq)
```



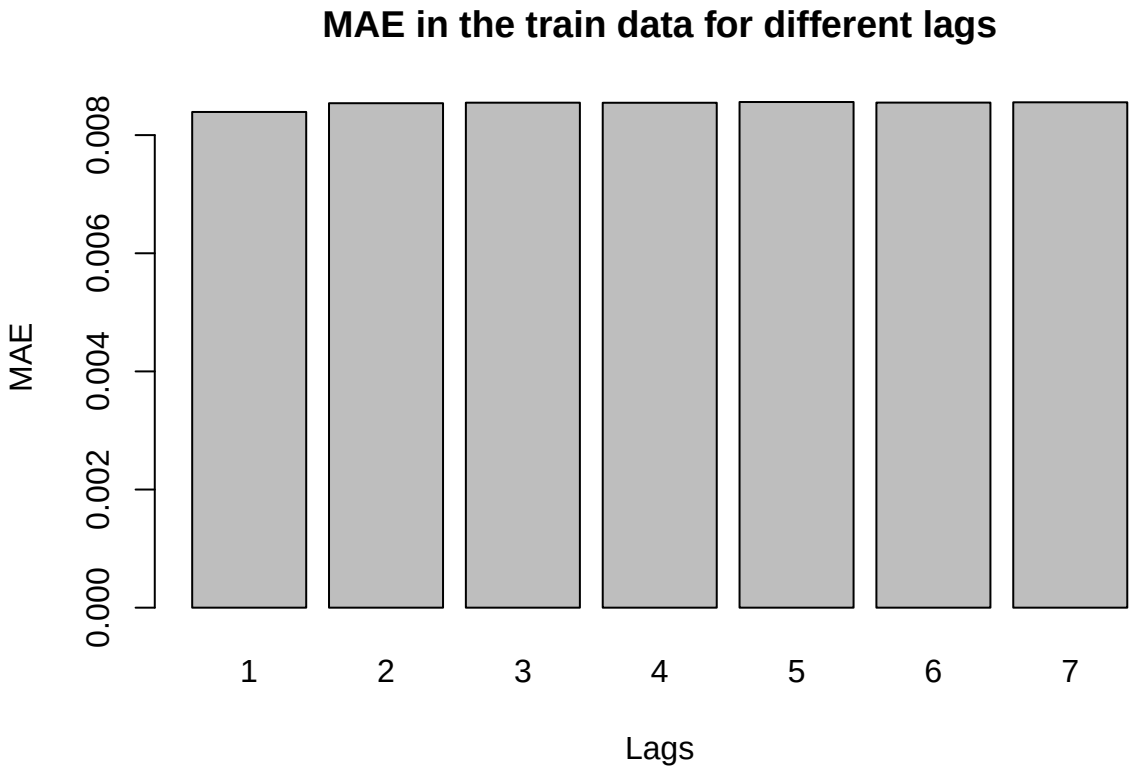
We see the optimization is a huge problem in this case, and we need more background on optimization, when we use `rollapply`. We expect around $0.05 * 730 = 37$ spikes out of the confidence interval. It seems a little higher than that, but the residuals are pretty similar to the normal distribution. The spike around lag $= 365$ is really strange, because the model seems not capture this seasonality.

Multiplicative

```
## [1] 0.008555414
```

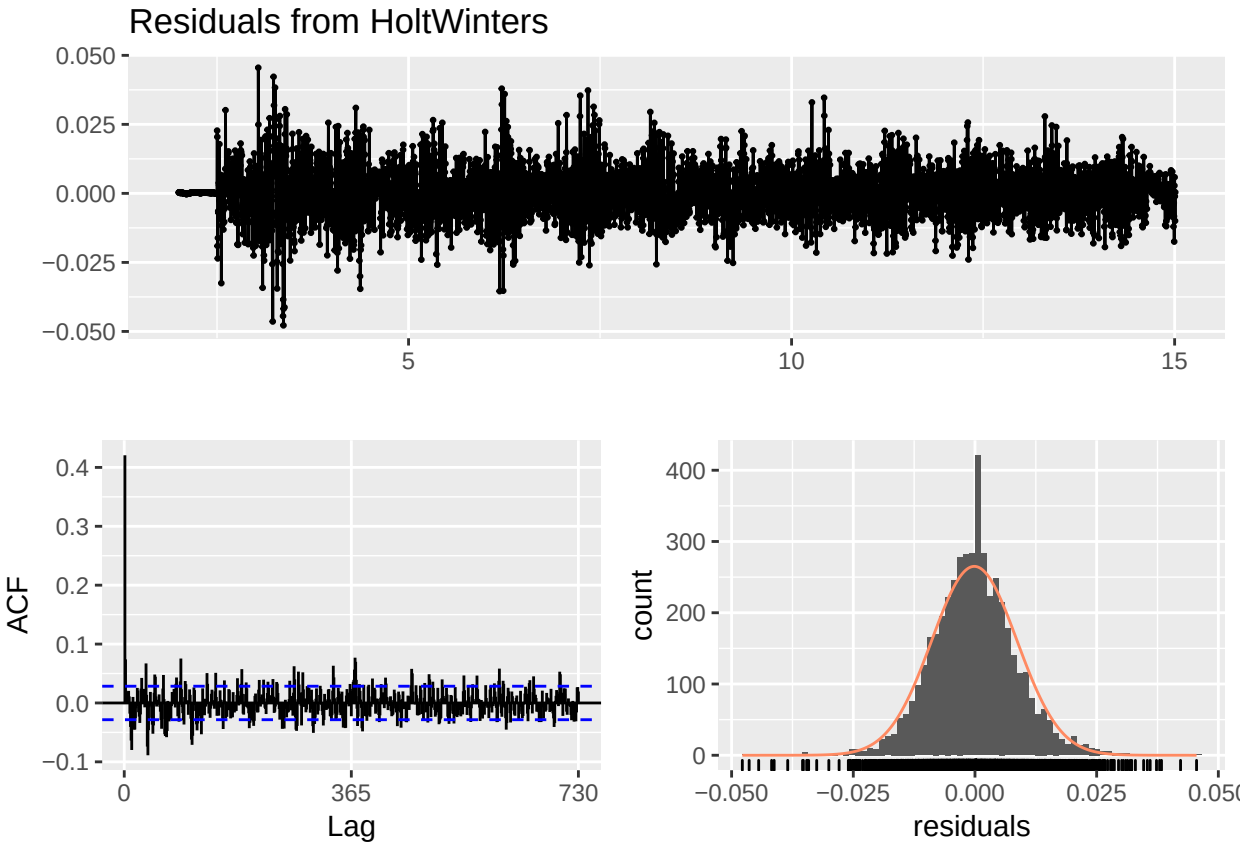


Next we compare the MAE chan changing the day from 1 to 7.



The MAE is not so bad. Let's analyse the residuals of an model fitted in all training data:

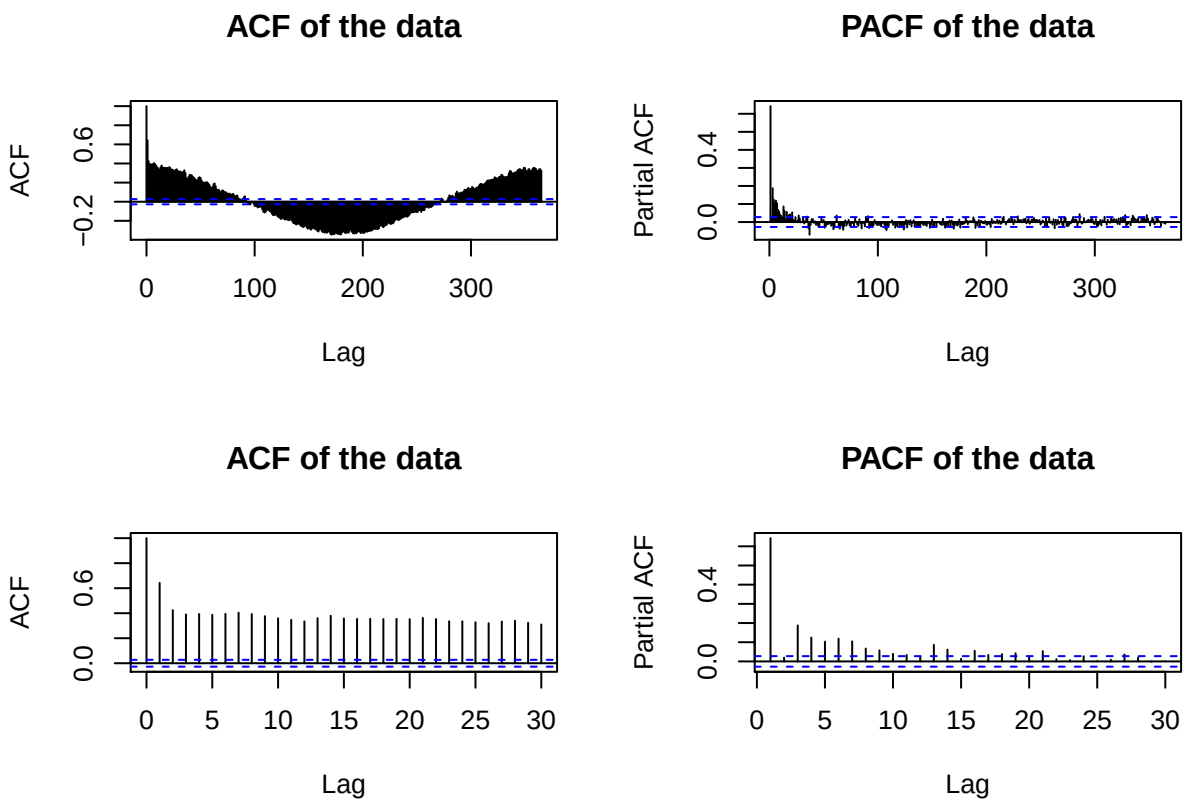
```
mod = HoltWinters(ts(o3_train+1, frequency = 365), seasonal = "multiplicative")
checkresiduals(mod, lag = 2*freq, lag.max = 2*freq)
```



Same problems as before. We see the optimization is a huge problem in this case, and we need more background on optimization, when we use `rollapply`. We expect around $0.05 * 730 = 37$ spikes out of the confidence interval. It seems a little higher than that, but the residuals are pretty similar to the normal distribution.

ARMA

We can see the ACF and PACF:



Based on these graphs, we see both graphs has a exponentially decay, the first after the $q - p = -1$ or $q - p = 0$. 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.

We tested

```
## [1] "Model ARMA(1,2)"
## [1] "AIC = -35980.5137257693"
## [1] "p-values"
##           ar1           ma1           ma2     intercept
##  0.000000e+00  0.000000e+00 1.710718e-217  4.125983e-64
## [1] ""
```

```
## [1] "Model ARMA(2,1)"
## [1] "AIC = -35428.6356484371"
## [1] "p-values"
##          ar1          ar2          ma1    intercept
## 5.901369e-01 2.561053e-09 8.409597e-13 0.000000e+00
## [1] ""
```

```
## [1] "Model ARMA(3,1)"
## [1] "AIC = -35992.7740860871"
## [1] "p-values"
```

```
##           ar1           ar2           ar3           ma1     intercept
##  0.000000e+00 2.117095e-148  4.864612e-26  0.000000e+00  5.876975e-64
## [1] ""
```

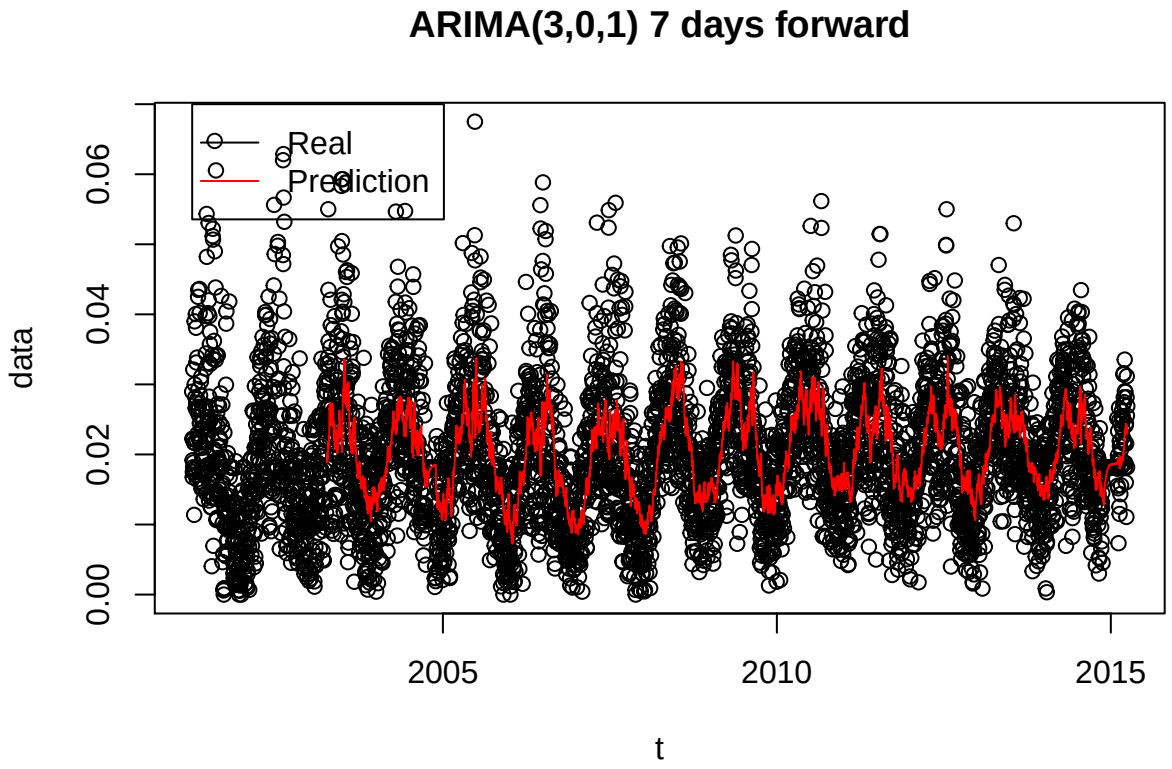
```
## [1] "Model ARMA(3,2)"
## [1] "AIC = -35994.0619496289"
## [1] "p-values"
##          ar1          ar2          ar3          ma1          ma2
## 2.336209e-47 3.447411e-03 3.418314e-02 4.333555e-17 6.240492e-02
##    intercept
## 1.017871e-62
## [1] ""
```

```
## [1] "Model ARMA(2,3)"
## [1] "AIC = -35994.1677032952"
## [1] "p-values"
##          ar1          ar2          ma1          ma2          ma3
## 1.953423e-04 8.264670e-02 4.331430e-01 6.283007e-09 9.454889e-03
##    intercept
## 1.555674e-60
## [1] ""
```

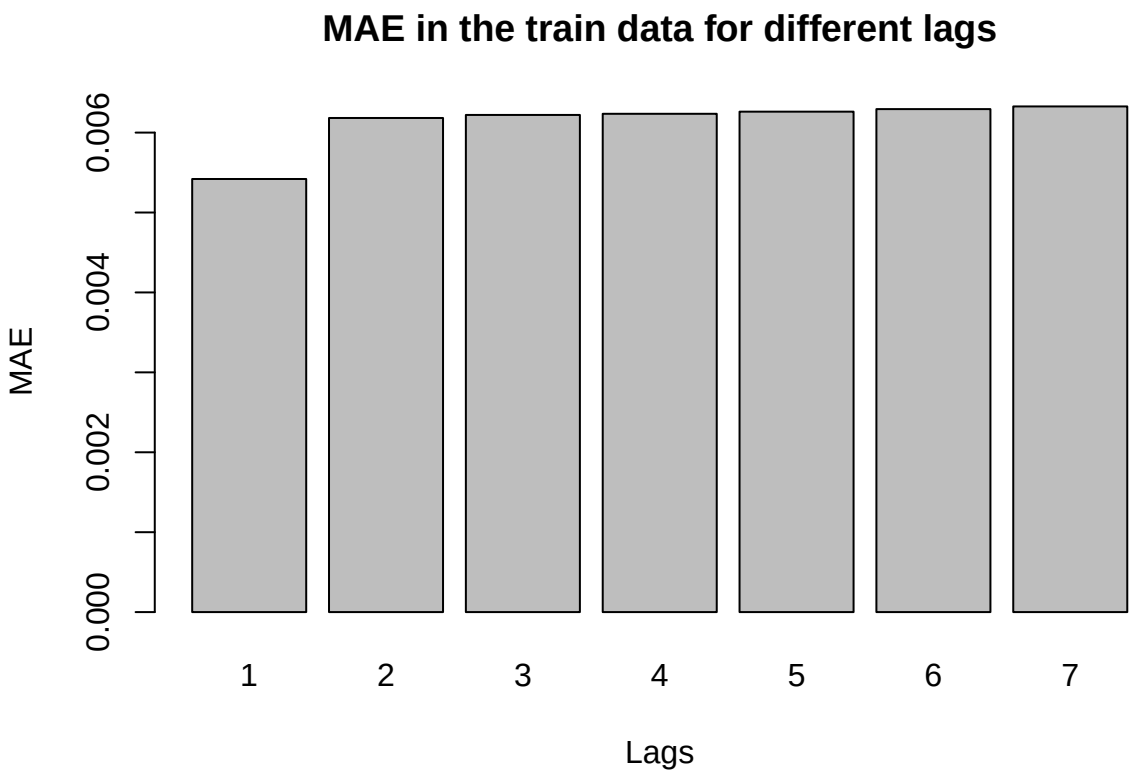
Considering $\alpha = 0.05$, only the first and third models have all parameters significant. But second and tha last have one or two non significant. Considering that, I will compare the two with minimum AIC, the third (ARMA(3,1)) and the fifth (ARMA(2,3)).

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

```
## [1] 0.006327397
```

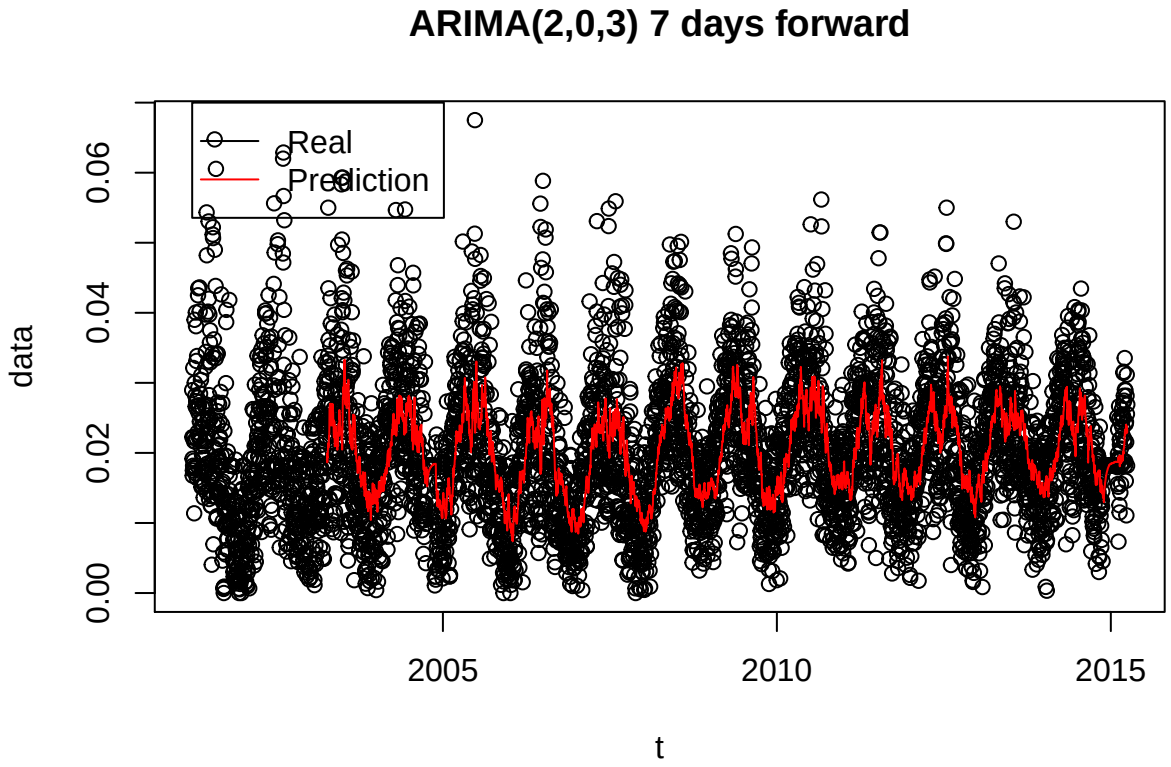


Next we compare the MAE chan changing the day from 1 to 7.

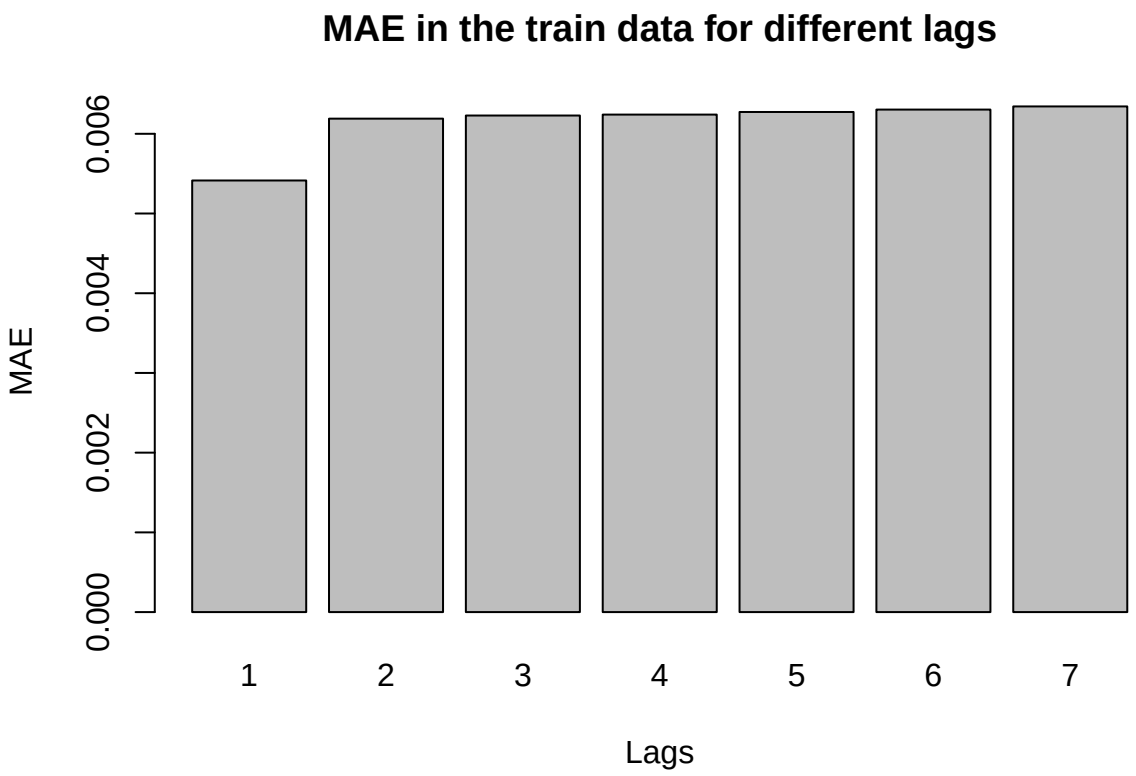


And the MAE of the ARIMA(2, 0, 3):

```
## [1] 0.00634312
```

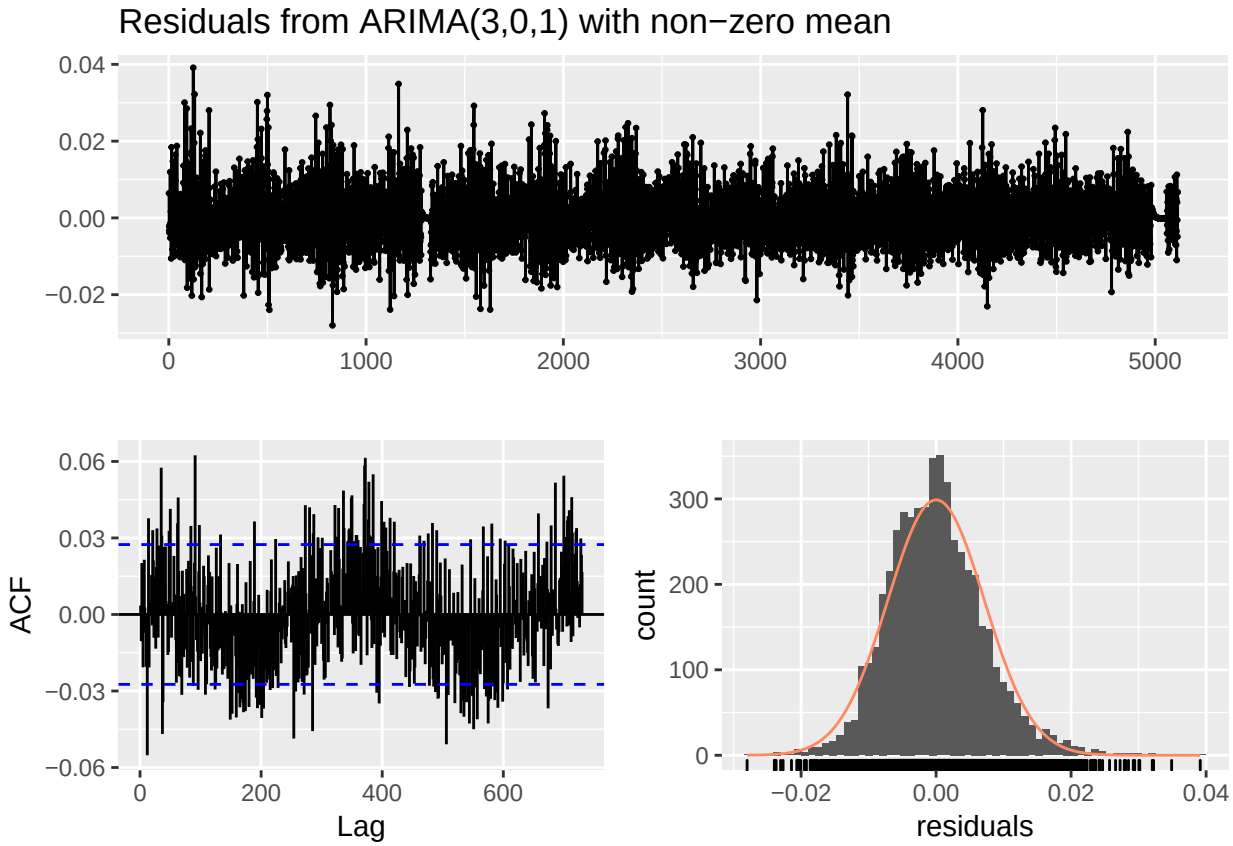


Next we compare the MAE chan changing the day from 1 to 7.



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)
```



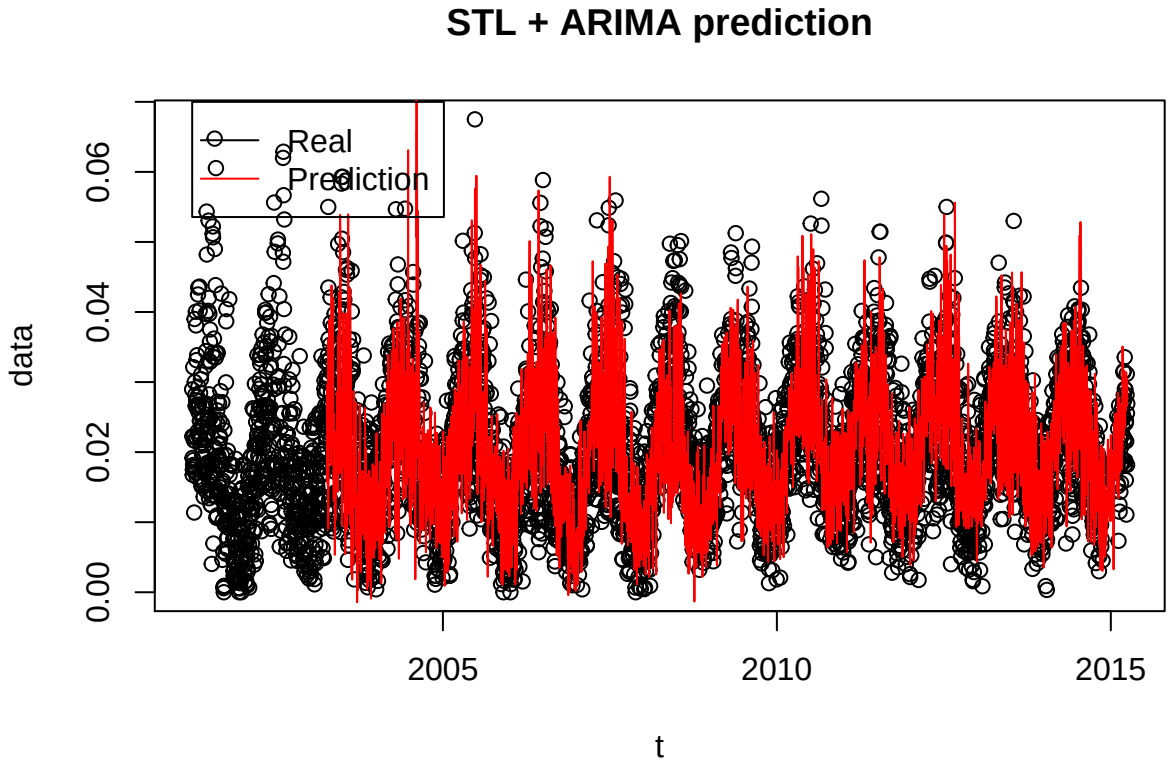

```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(3,0,1) with non-zero mean
## Q* = 1704.5, df = 725, p-value < 2.2e-16
##
## Model df: 5.   Total lags used: 730
```

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 caracteristics 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] 0.007832648
```

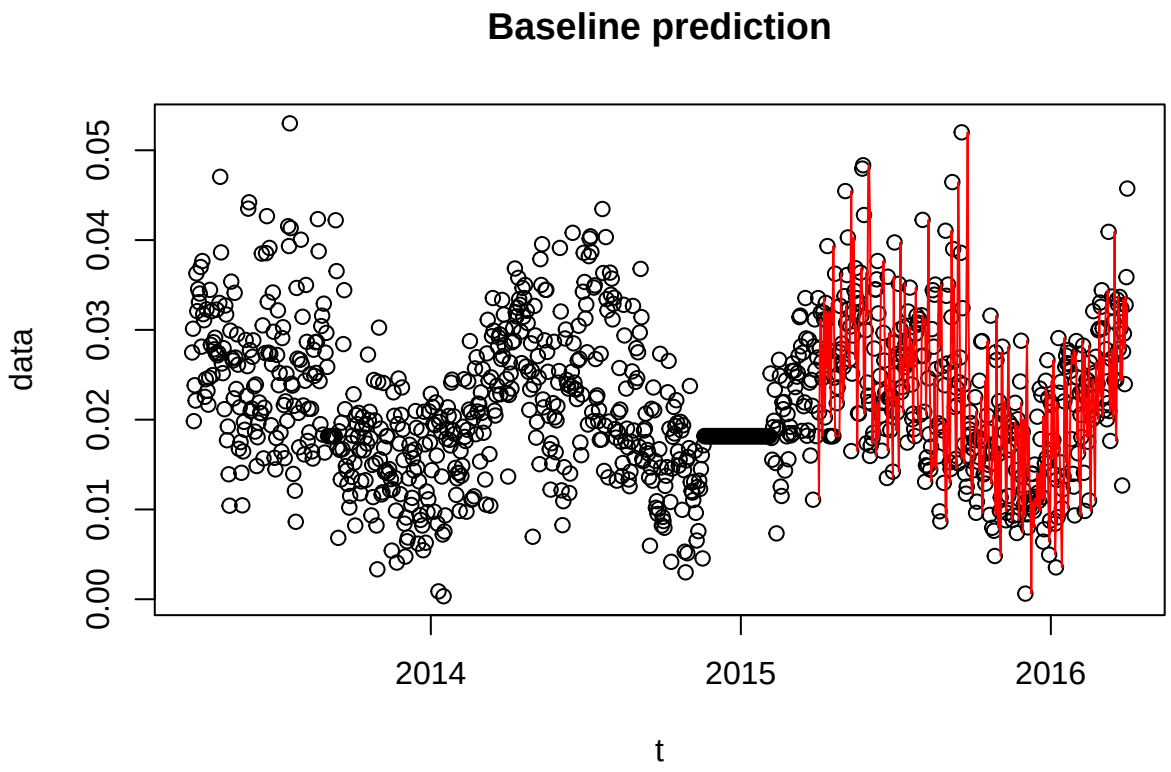


However the MAE isn't improved by this model. For that reason, this model was disregarded.

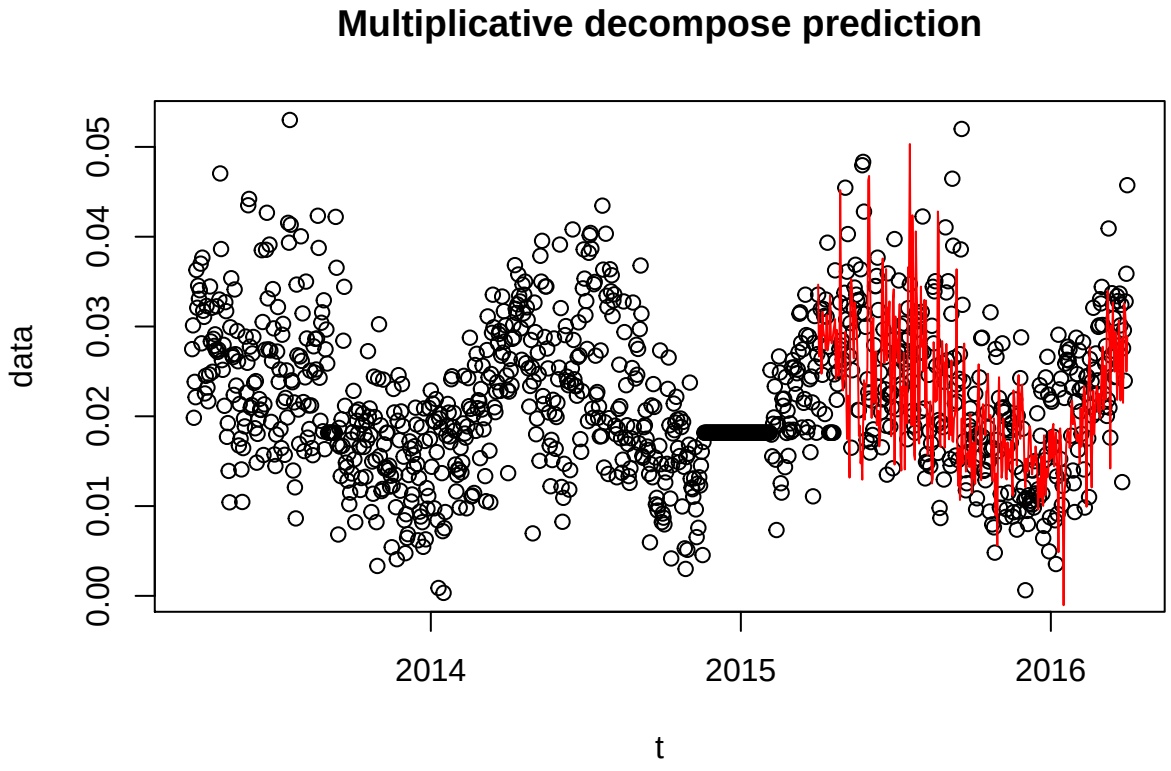
Comparing models in the test data.

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

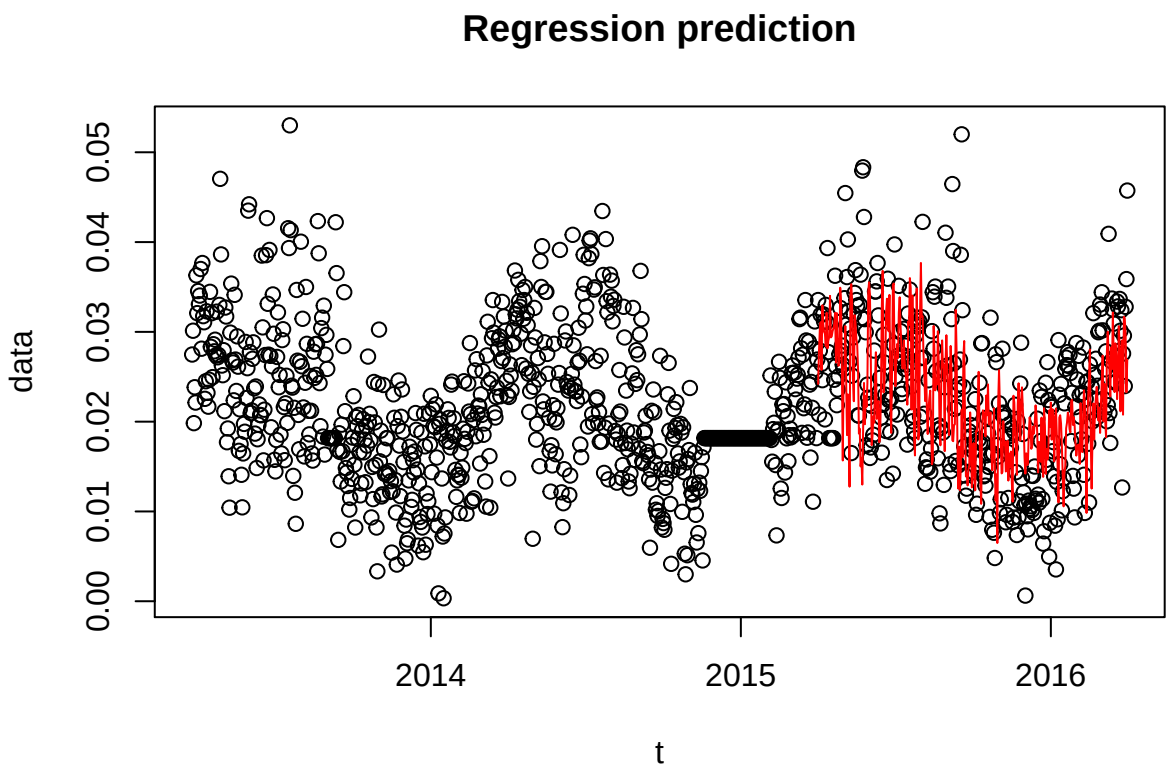
```
## [1] 0.007814345
```



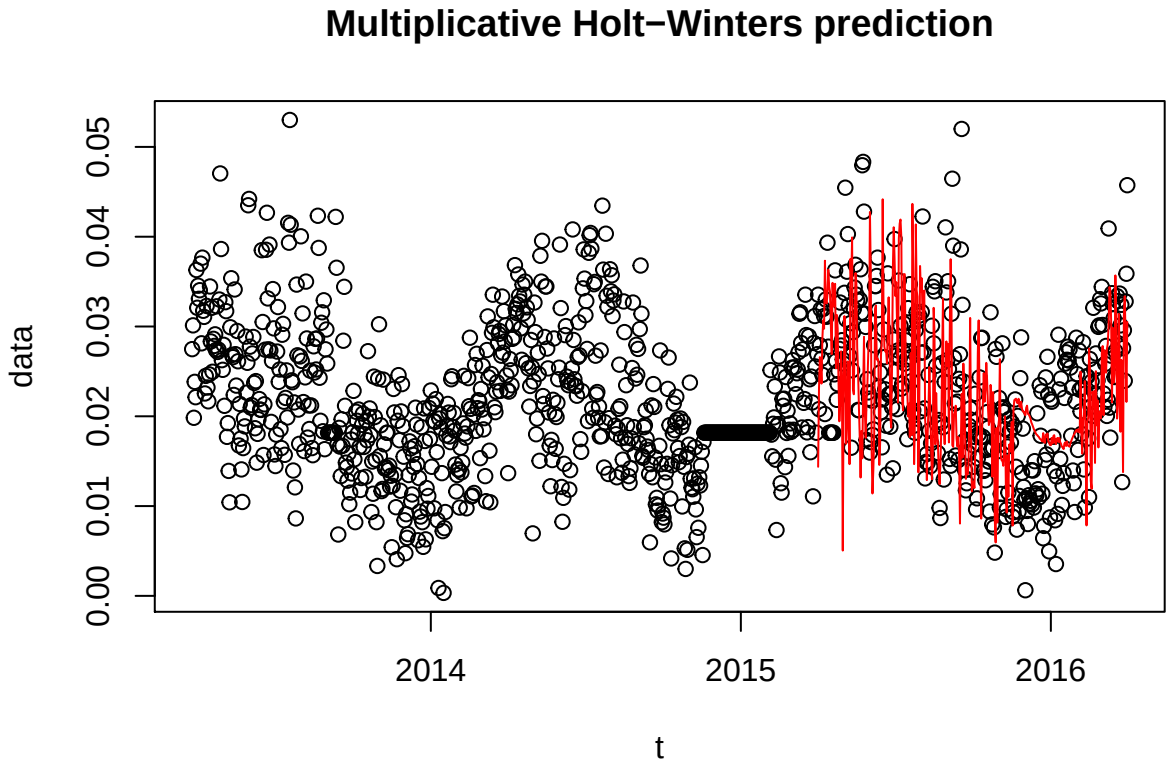

```
## [1] 0.007542956
```



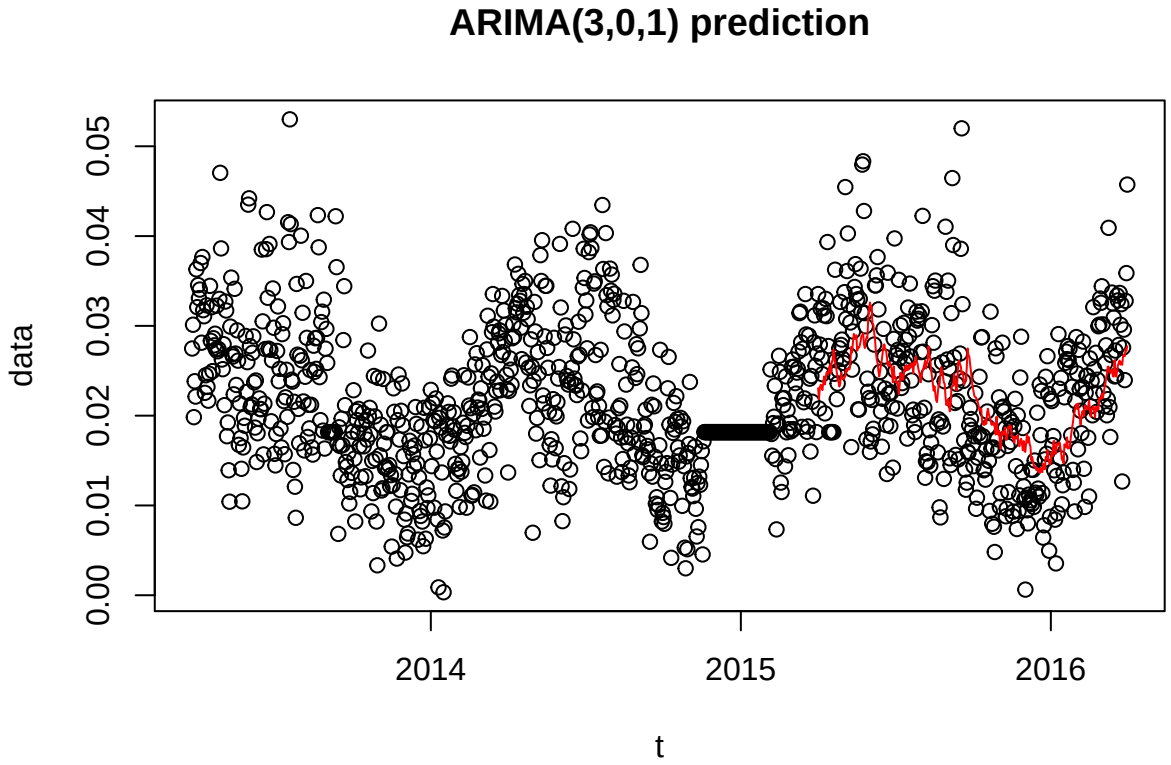

```
## [1] 0.007186968
```

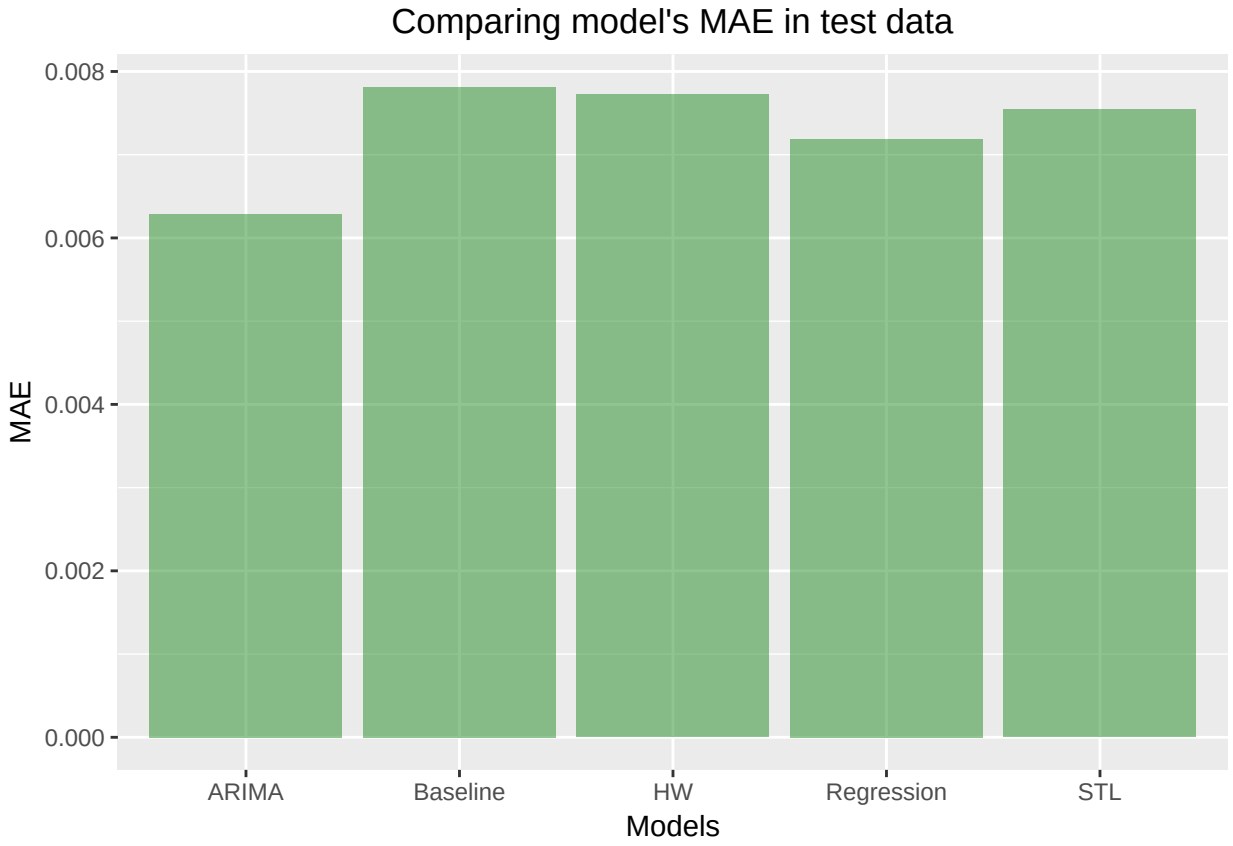



```
## [1] 0.007723339
```




```
## [1] "MAE ARIMA(3,0,1)"
## [1] 0.006289096
```





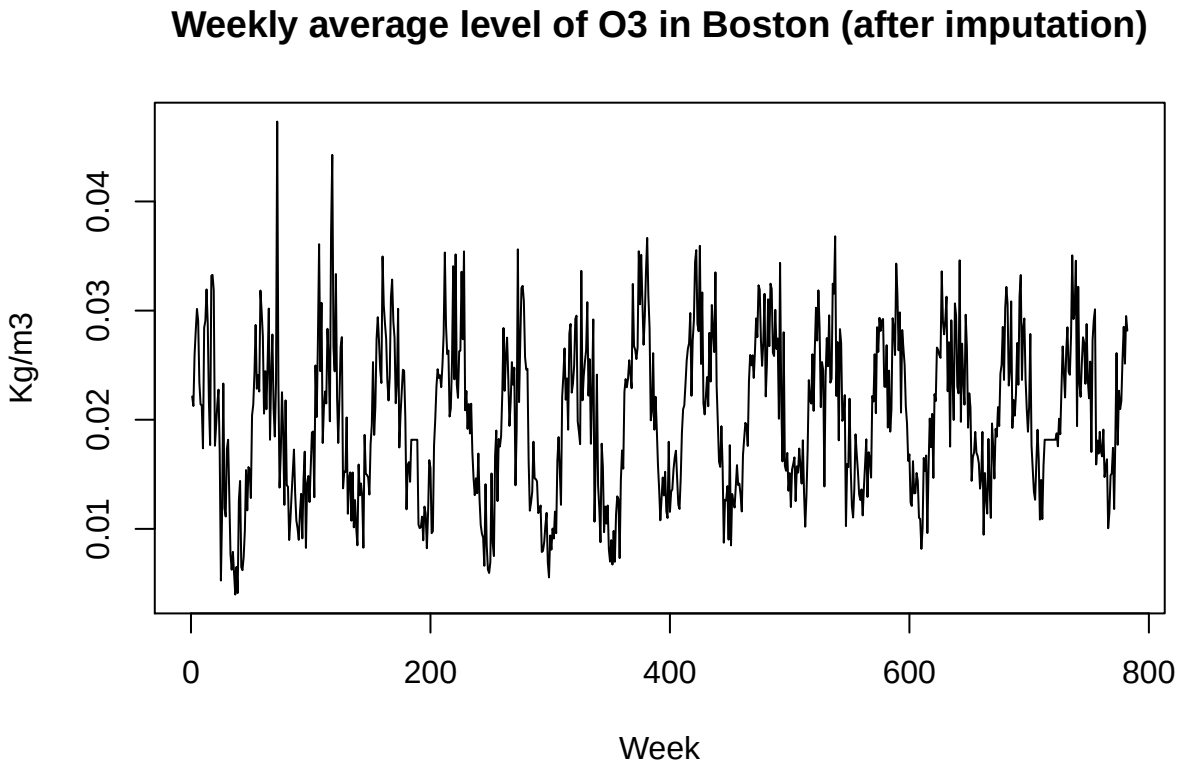
We see that our models performed reasonably, highlighted the ARIMA (in fact ARMA) model.

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)])
}
dates = seq(from = as.Date(time(o3[1])), to = as.Date(time(o3[length(o3)])), by = "weeks")[1:782]

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

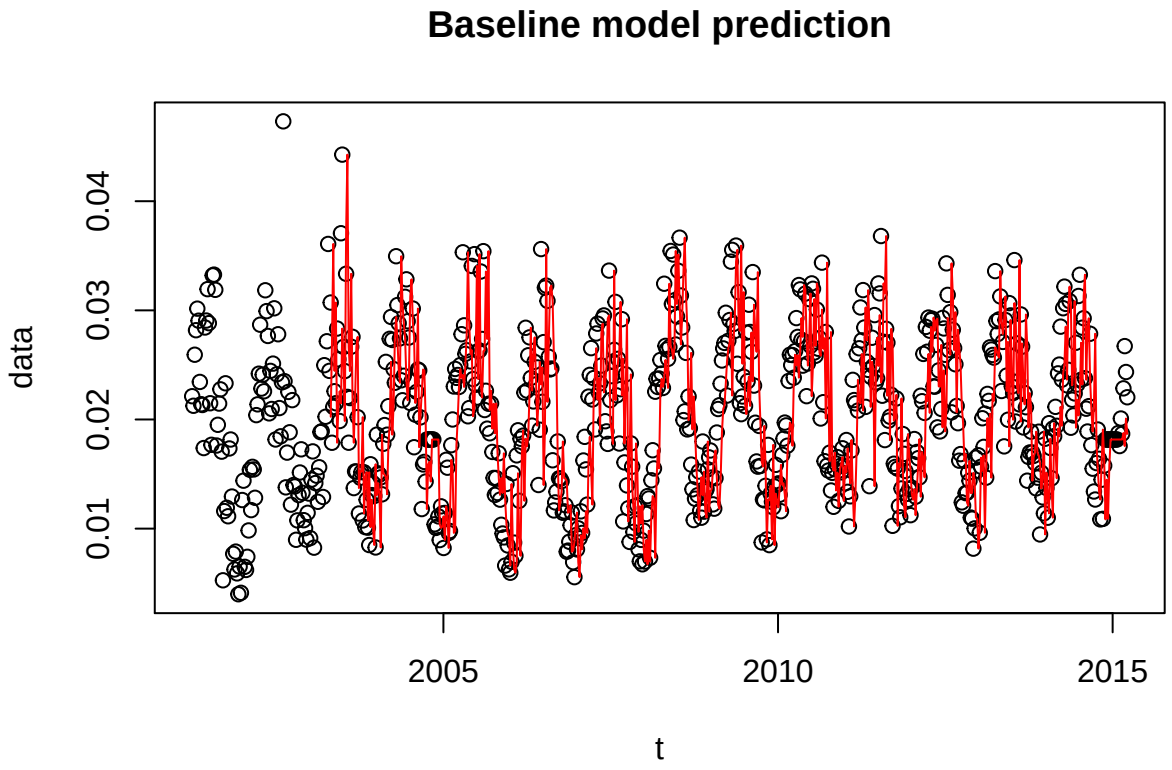



```
o3_train_week = o3_week[1:(length(o3_week)[1] - 52)]
o3_test_week = o3_week[-c(1:(length(o3_week)[1] - 52))]
```

Baseline Model

We will do the naive forecast to the baseline model.

```
## [1] 0.01008042
```



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.

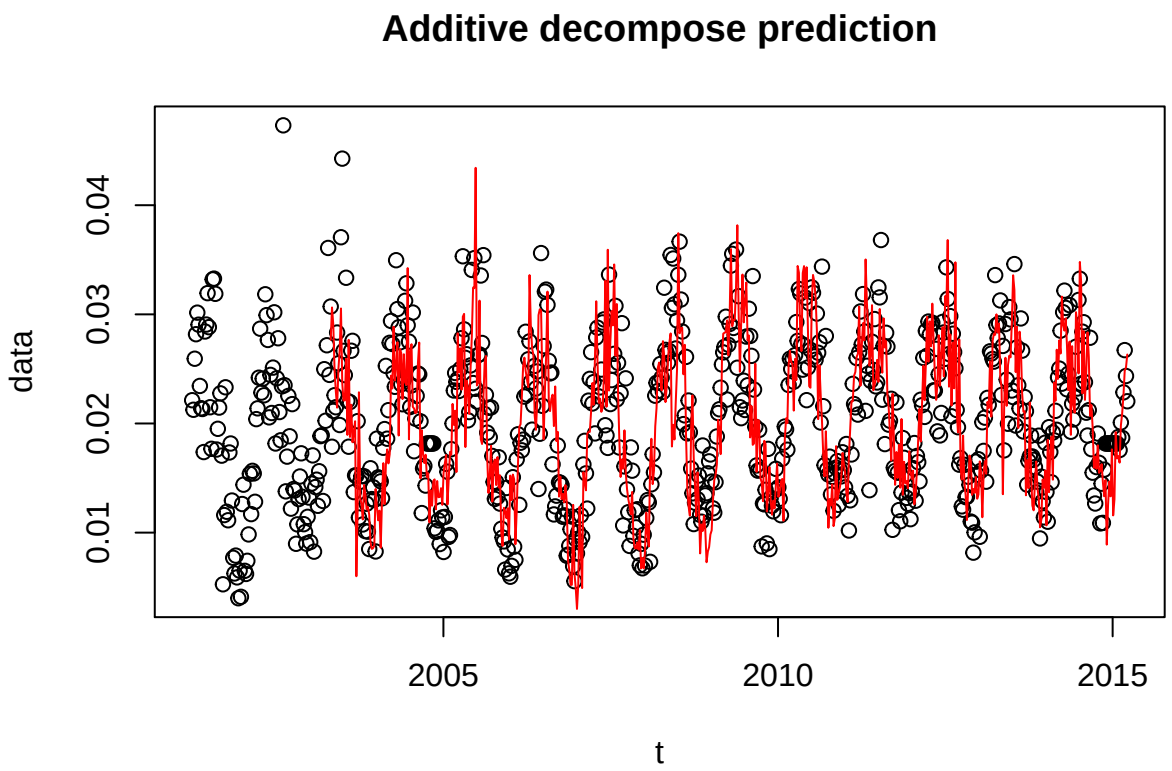
```
##
##  Kruskal-Wallis rank sum test
##
## data:  o3_train_week and g
## Kruskal-Wallis chi-squared = 0.12693, df = 3, p-value = 0.9884
```

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

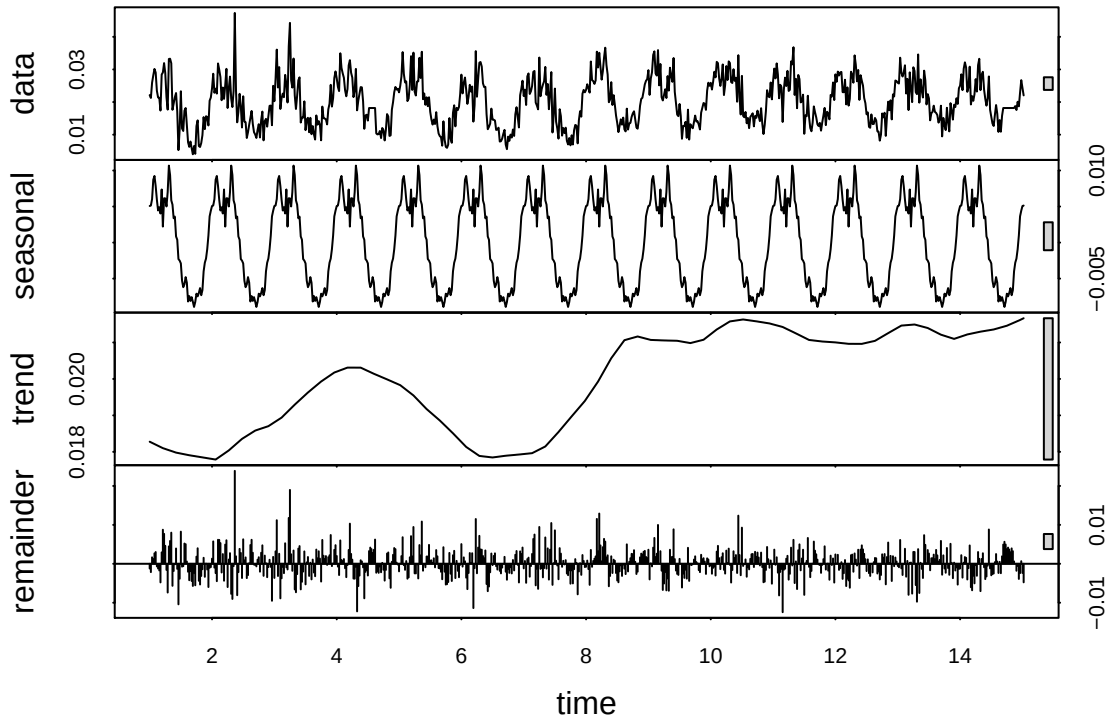
Additive model

We see the MAE of the additive decompose model:

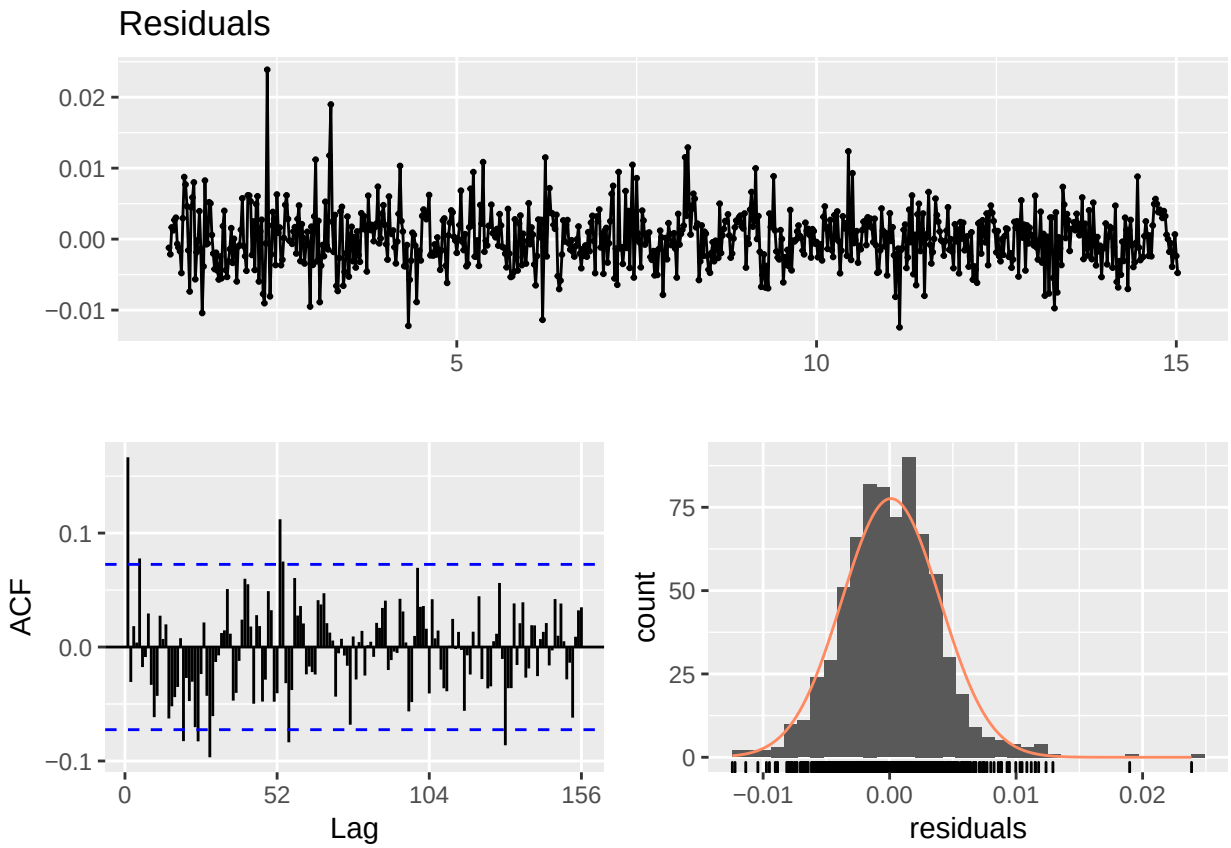
```
## [1] 0.003999828
```



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



The ACF and the PACF of the reminder:

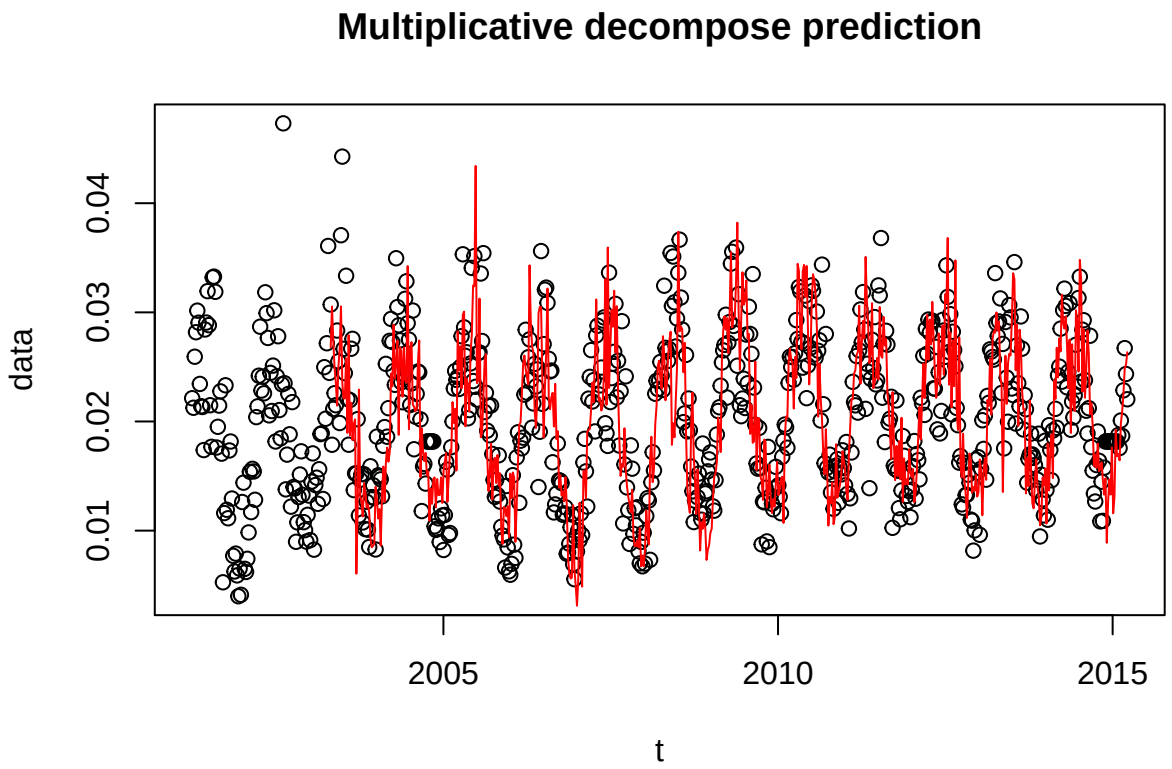


We see a big spike when lag = 52. It seems not so good for a reminder.

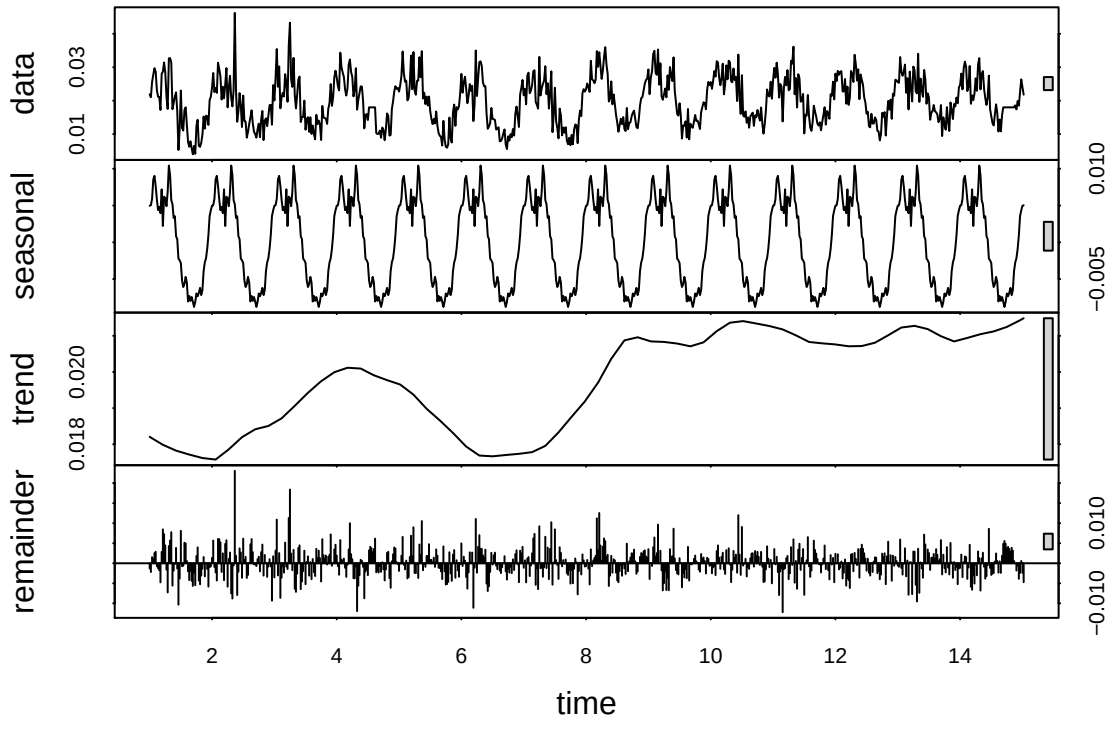
Multiplicative model

We analyse the MAE:

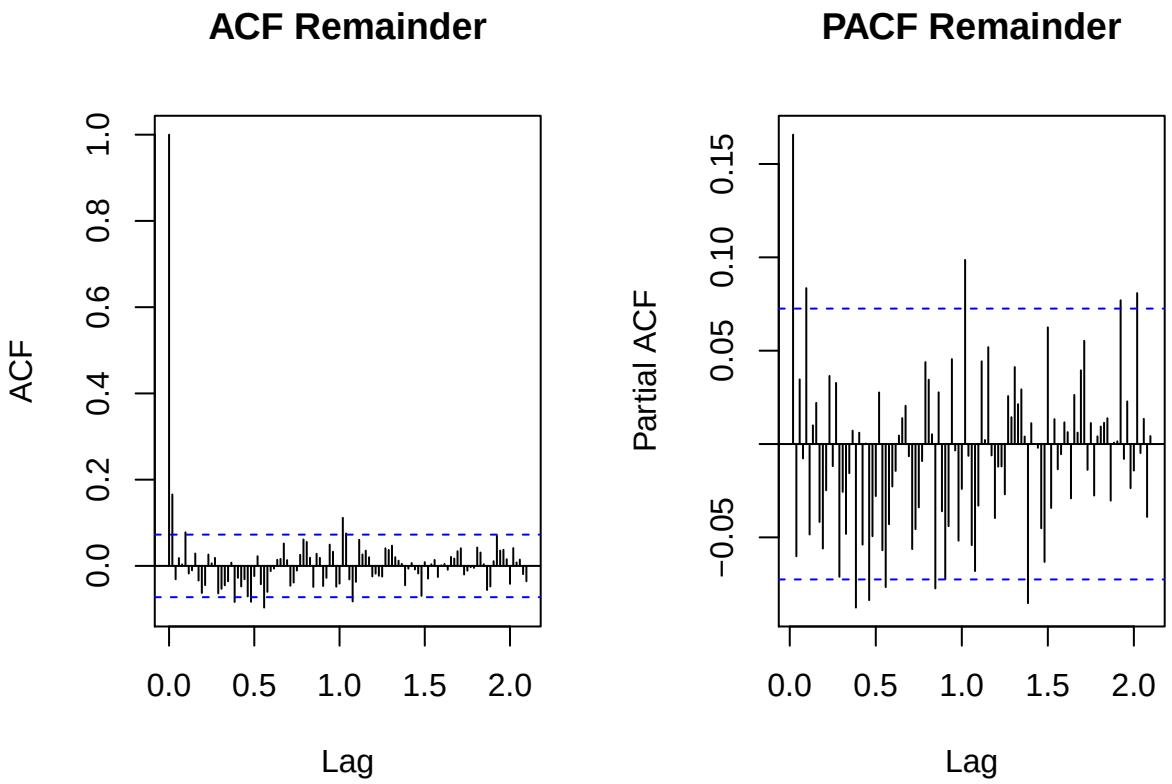
```
## [1] 0.00399206
```



Using `t.window` and analysing the reminder:



The ACF and the PACF of the reminder:

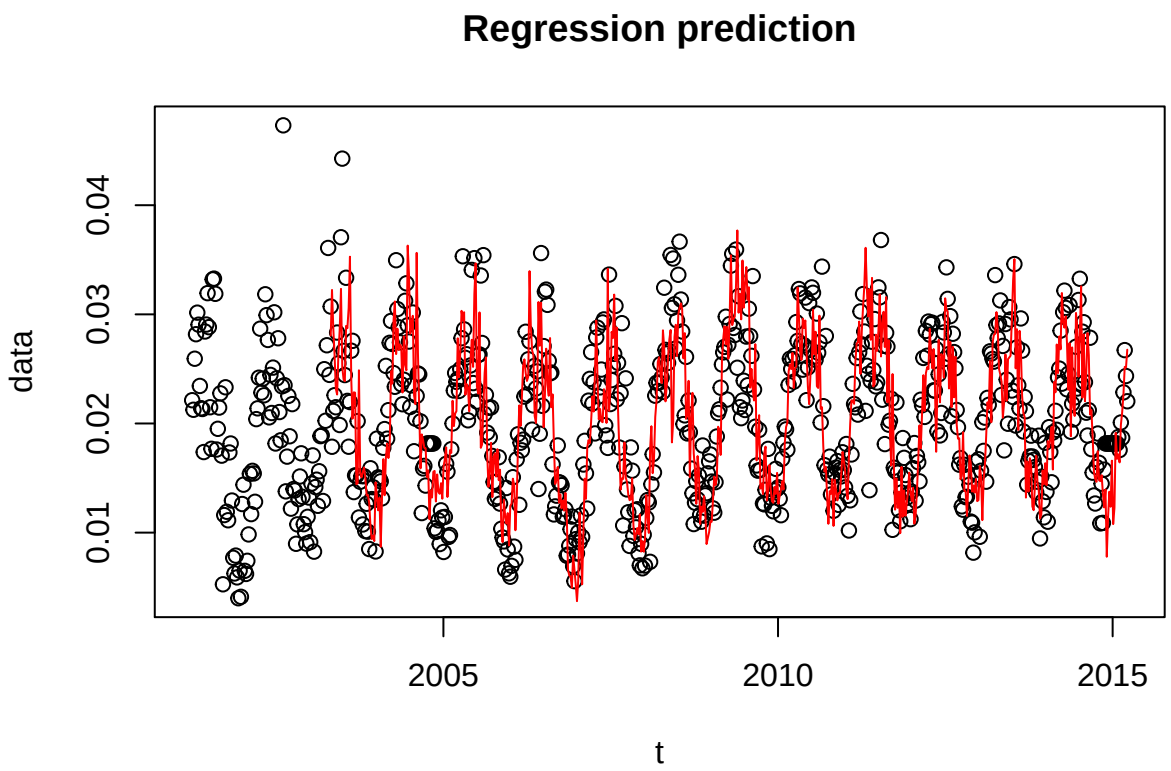


We also see the spike when lag = 52. It seems not so good for a reminder. The same as before.

Regression

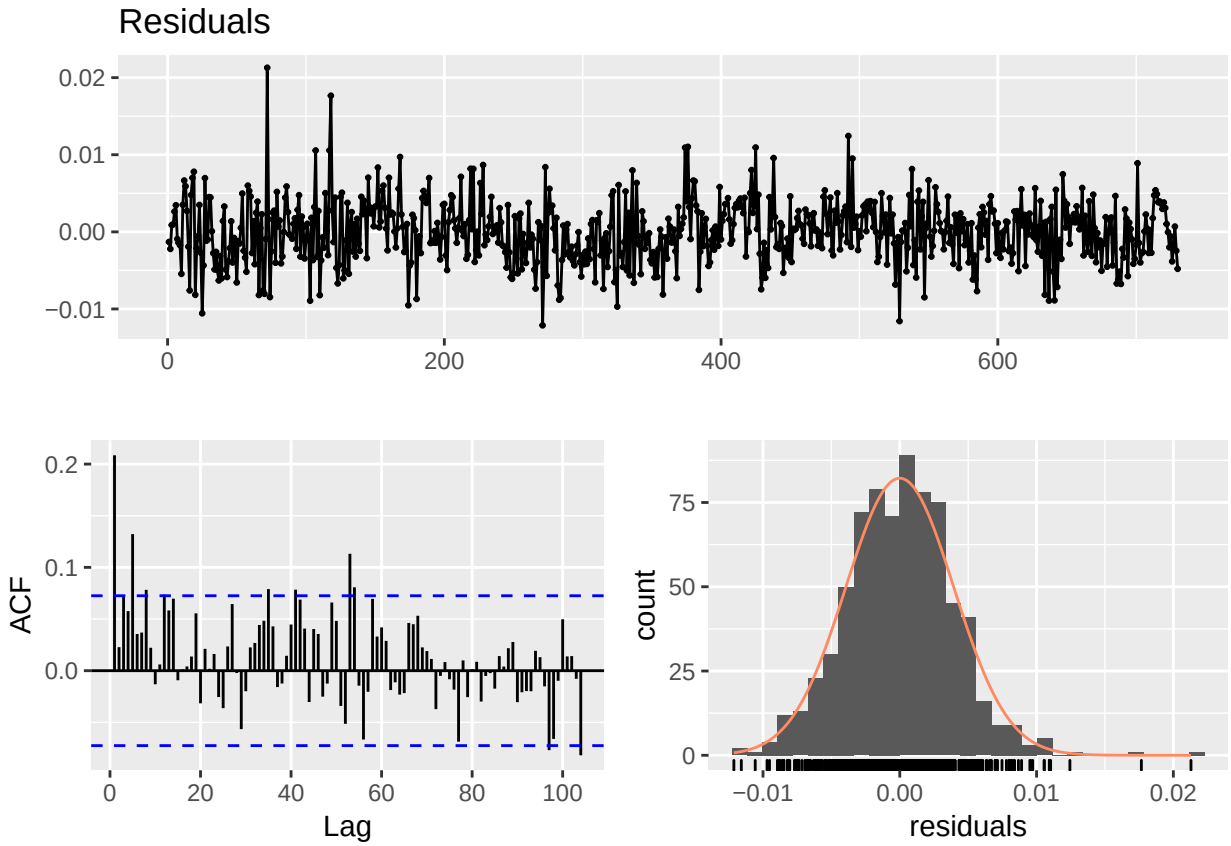
We tested for seasonalities, and we settled with 52. So we fit a regression model with the seasonality dummies. We see the MAE:

```
## [1] 0.003845637
```



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)
```




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

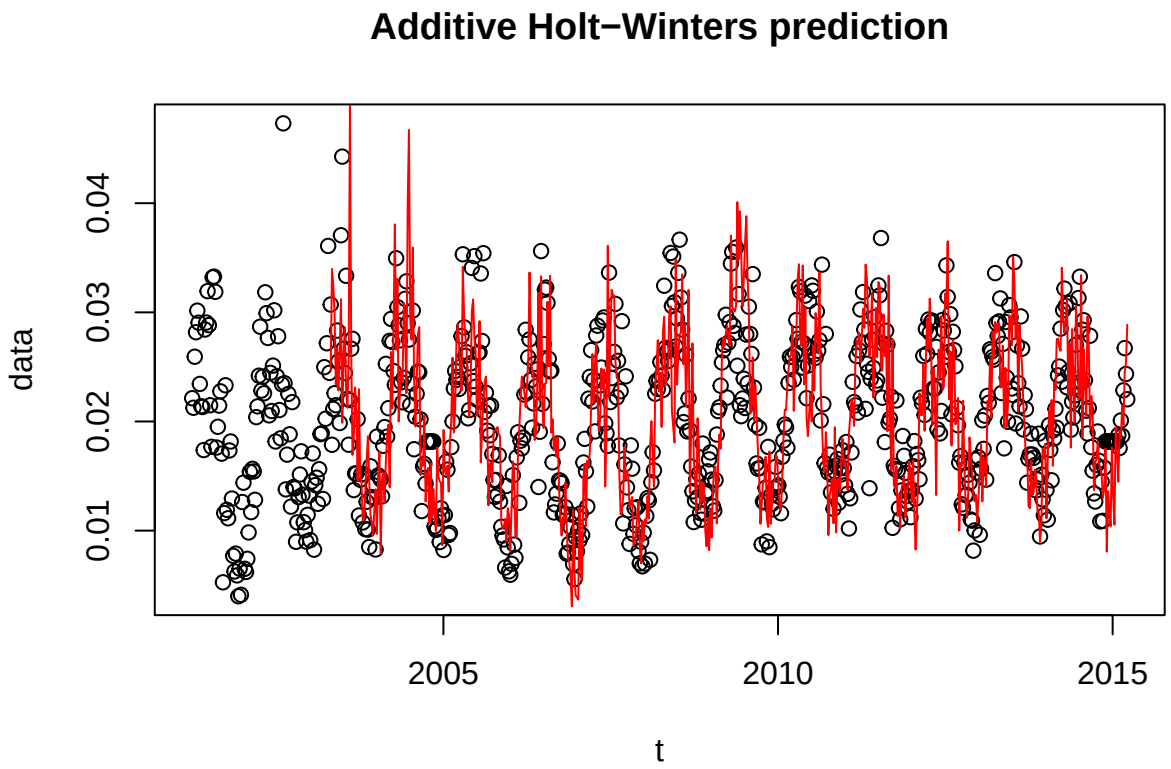
Wee see spikes in lag $= 52, 104$. Same as before.

Holt-Winters

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

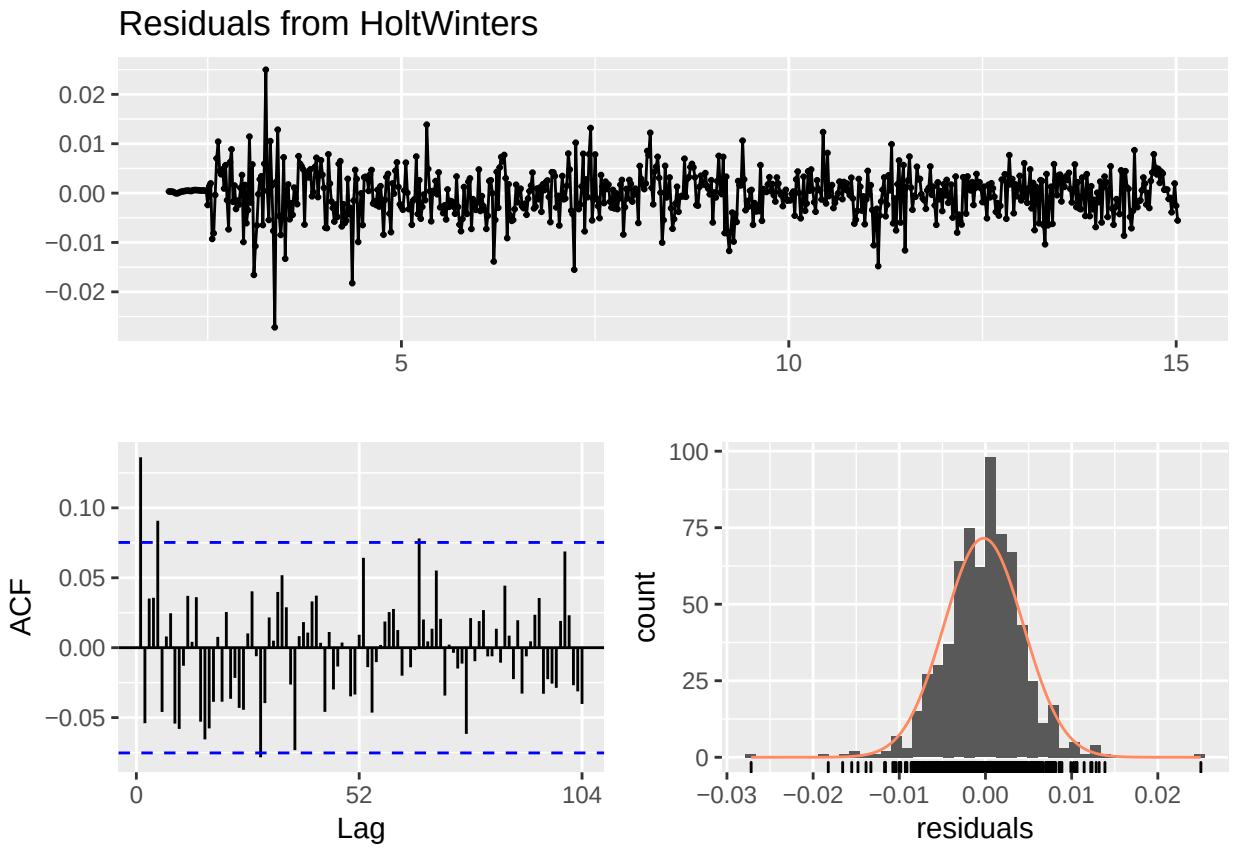
Additive

```
## [1] 0.004276606
```



MAE is not so bad, but we have seen better results. Let's analyse the residuals:

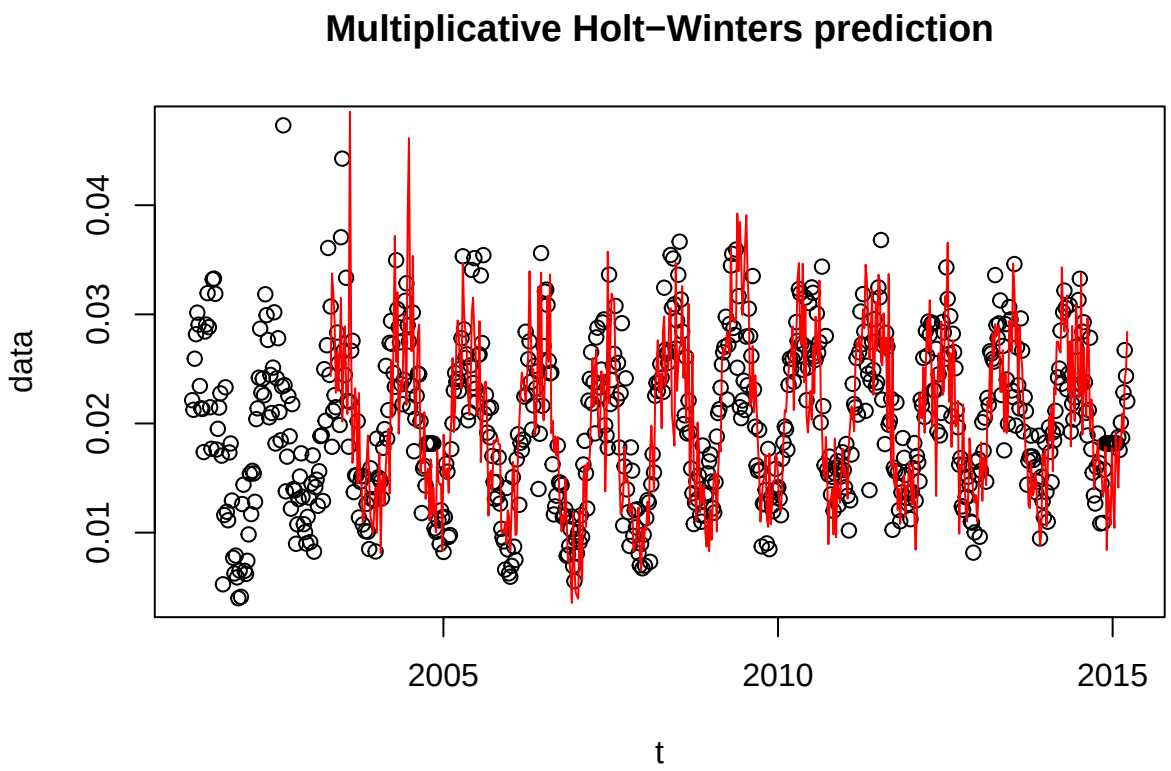
```
mod = HoltWinters(ts(o3_train_week, frequency = 52), seasonal = "additive")
checkresiduals(mod, lag = 2*freq, lag.max = 2*freq)
```



The residuals are pretty good actually. We see a norm histogram and the spikes above the lines, almost all, and we expected around 5 be higher. So, that model sounds good considering the residuals.

Multiplicative

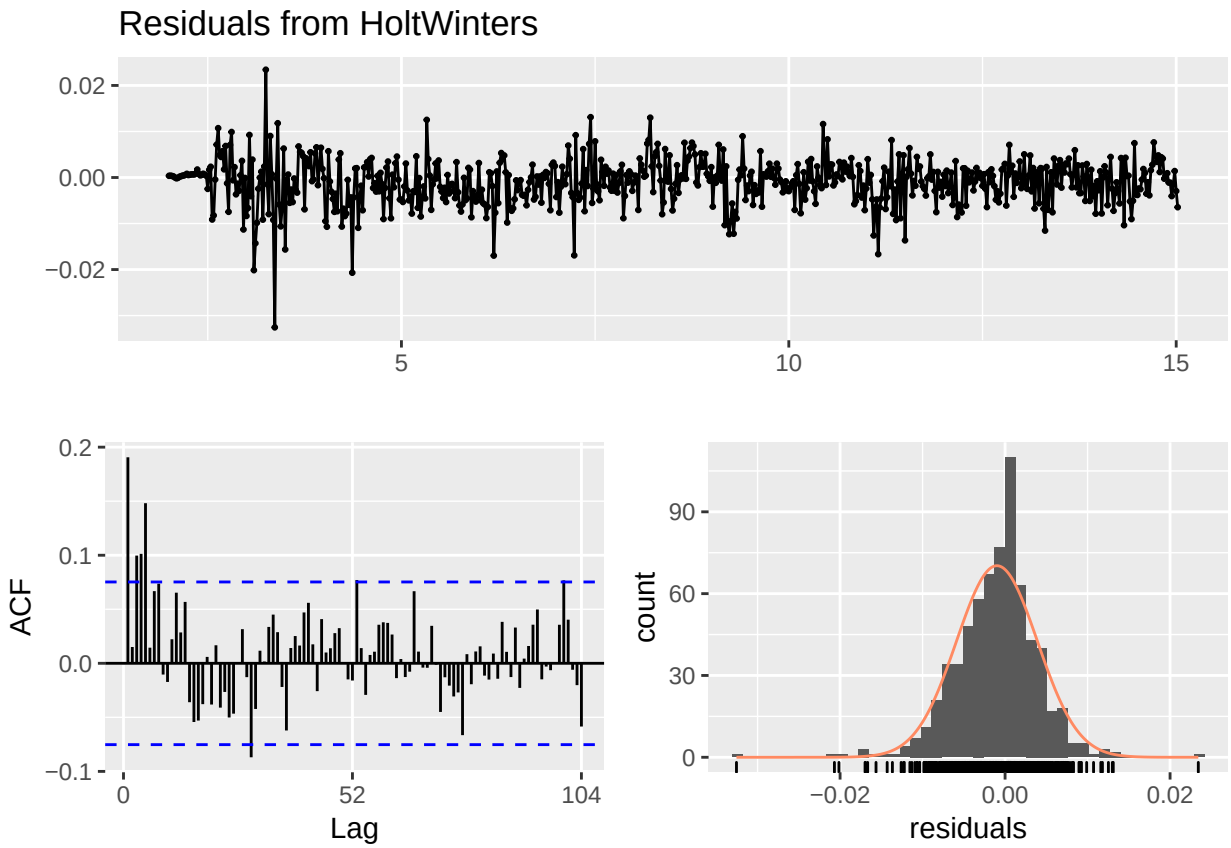
```
## [1] 0.004243044
```



We see also a reasonable MAE. Let's analyse the residuals:

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

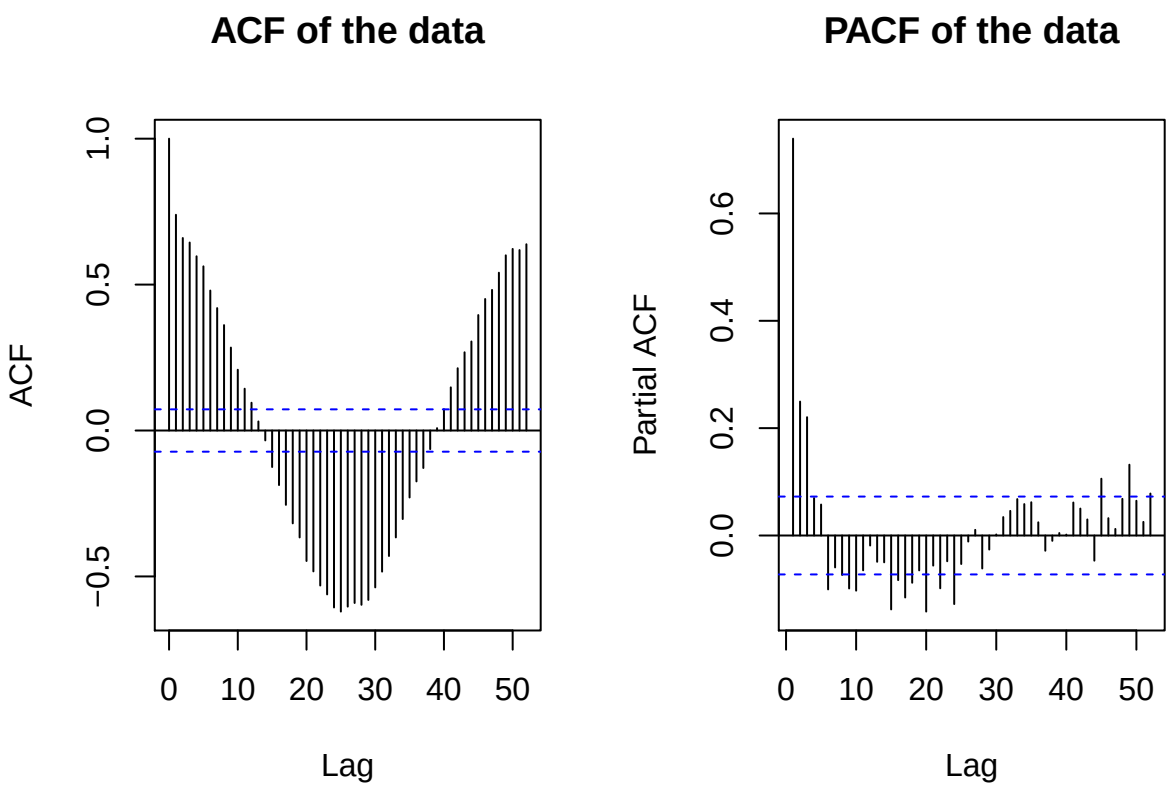
```
## Warning in modeldf.default(object): Could not find appropriate degrees of
## freedom for this model.
```



The same as before, except the fact the the ACF of the remainder seems to be a AR model, what is interesting.

ARMA

We can see the ACF and PACF:



Based on these graphs, we see both graphs has a exponentially decay, the first after the $q - p = 1$ or $q - p = 2$. 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.

```
## [1] "Model ARMA(1,2)"
## [1] "AIC = -5762.4304121375"
## [1] "p-values"
##          ar1          ma1          ma2    intercept
## 0.000000e+00 4.159117e-24 3.202857e-01 4.139337e-64
## [1] ""

## [1] "Model ARMA(2,1)"
## [1] "AIC = -5762.05454253045"
## [1] "p-values"
##          ar1          ar2          ma1    intercept
## 3.695551e-44 4.327690e-01 2.079691e-17 1.605927e-64
## [1] ""

## [1] "Model ARMA(3,1)"
## [1] "AIC = -5769.15408392066"
## [1] "p-values"
##          ar1          ar2          ar3          ma1    intercept
## 3.491202e-11 7.940421e-01 1.301972e-03 1.493624e-02 1.207618e-63
## [1] ""

## [1] "Model ARMA(1,3)"
## [1] "AIC = -5774.71251522564"
## [1] "p-values"
##          ar1          ma1          ma2          ma3    intercept
## 0.000000e+00 1.257101e-27 6.119552e-03 1.263972e-04 5.728102e-68
```

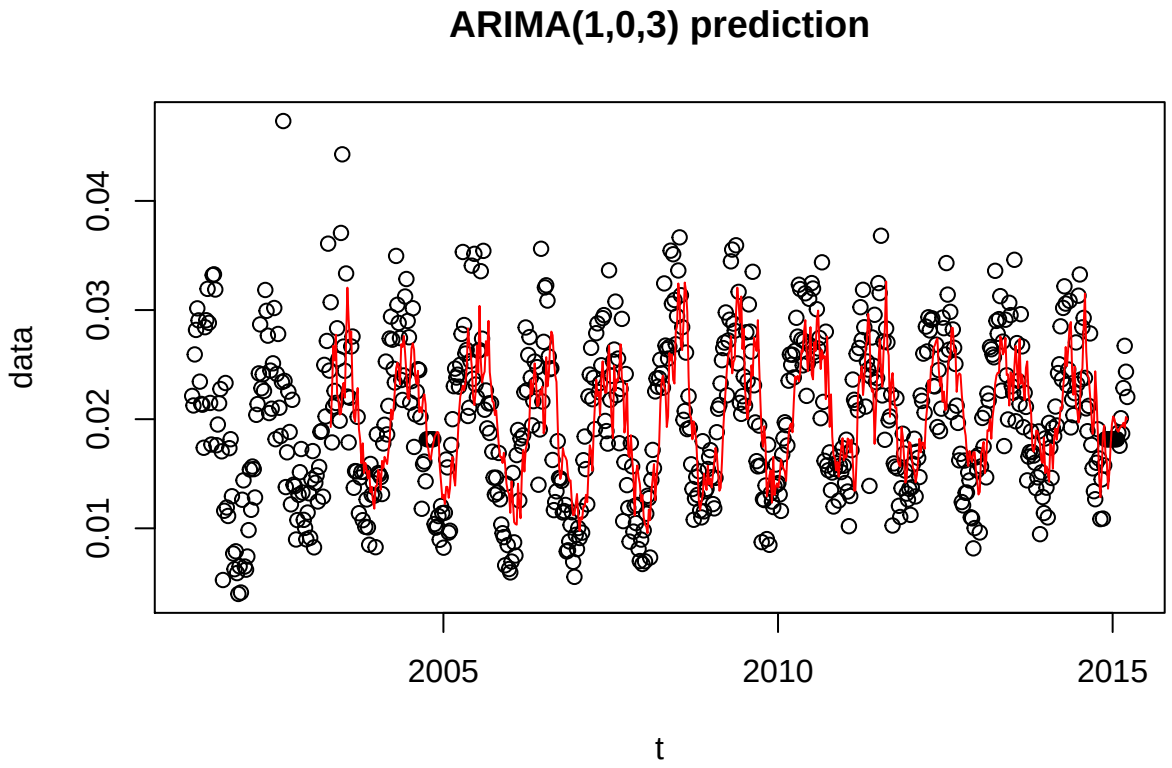
```
## [1] ""
```

```
## [1] "Model ARMA(2,3)"
## [1] "AIC = -5882.32013536996"
## [1] "p-values"
##          ar1          ar2          ma1          ma2          ma3
## 0.000000e+00 0.000000e+00 0.000000e+00 1.937176e-09 6.007339e-16
##    intercept
## 0.000000e+00
## [1] ""
```

The last three models are far the best ones. So we'll pick them out to train our model. However, the ARMA(2,3) could no be used, because of stationary problems. So:

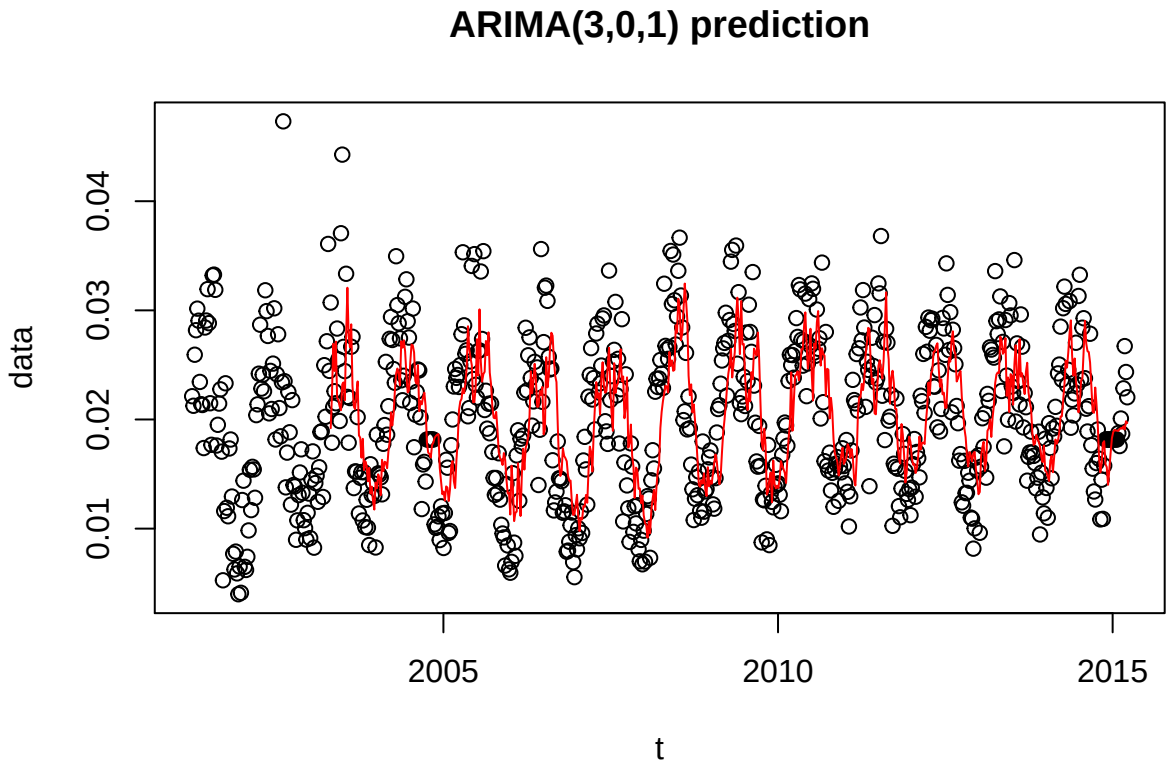
We see the MAE of the ARMA(1,3):

```
## [1] 0.004539656
```



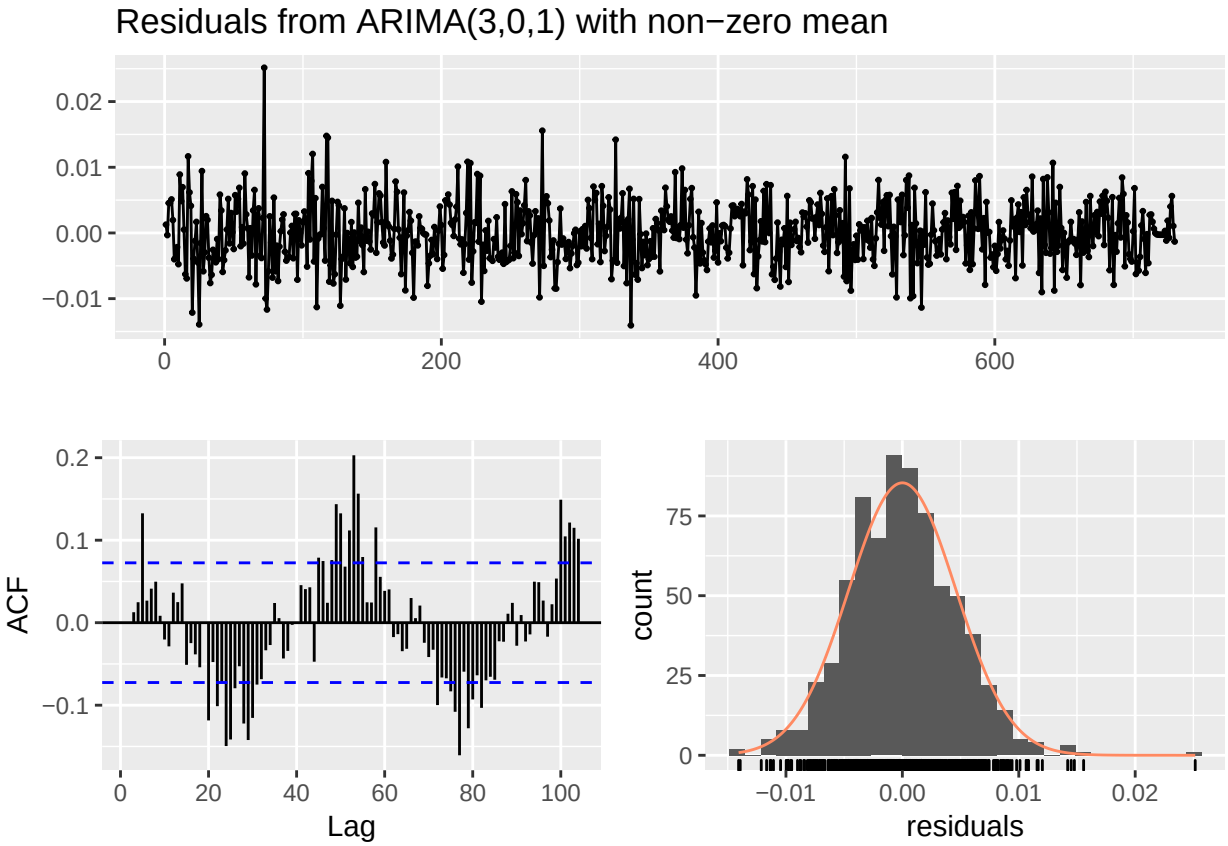
And the MAE of the ARMA(3, 1):

```
## [1] 0.004523656
```



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(3,0,1))
checkresiduals(model, lag = 2*freq, lag.max = 2*freq)
```



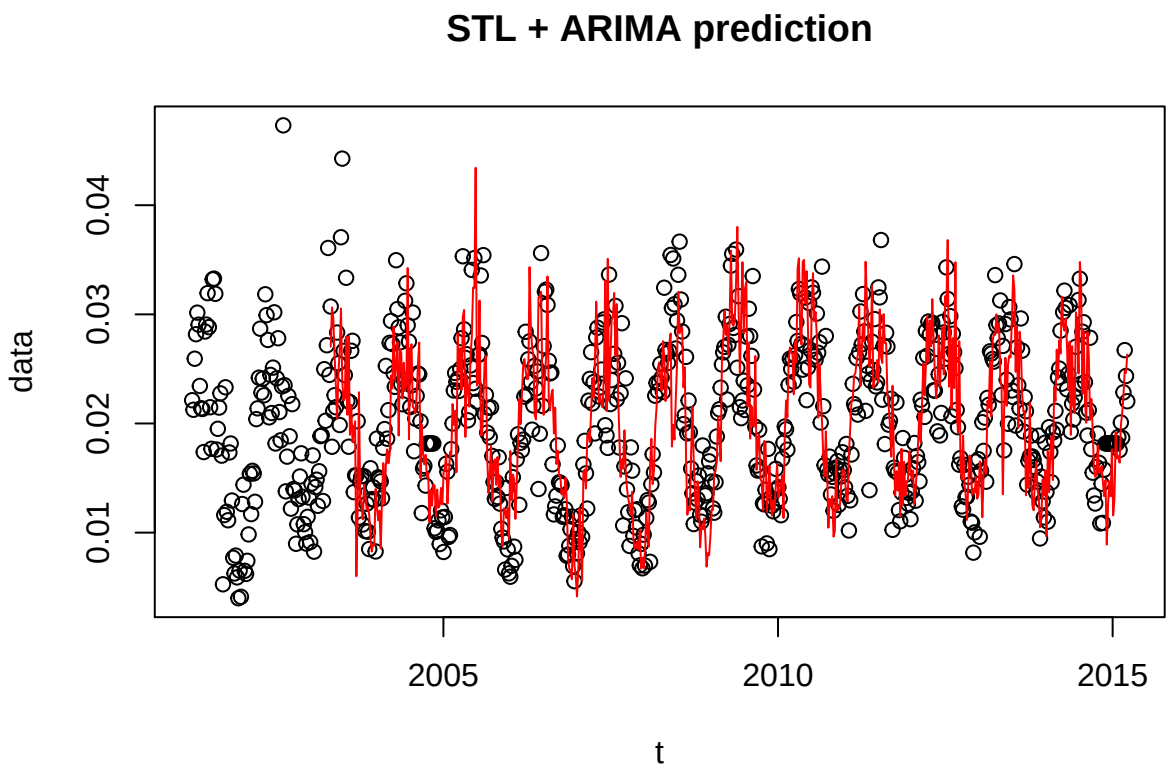

```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(3,0,1) with non-zero mean
## Q* = 452.4, df = 99, p-value < 2.2e-16
##
## Model df: 5.   Total lags used: 104
```

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 caracteristics 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] 0.003939254
```

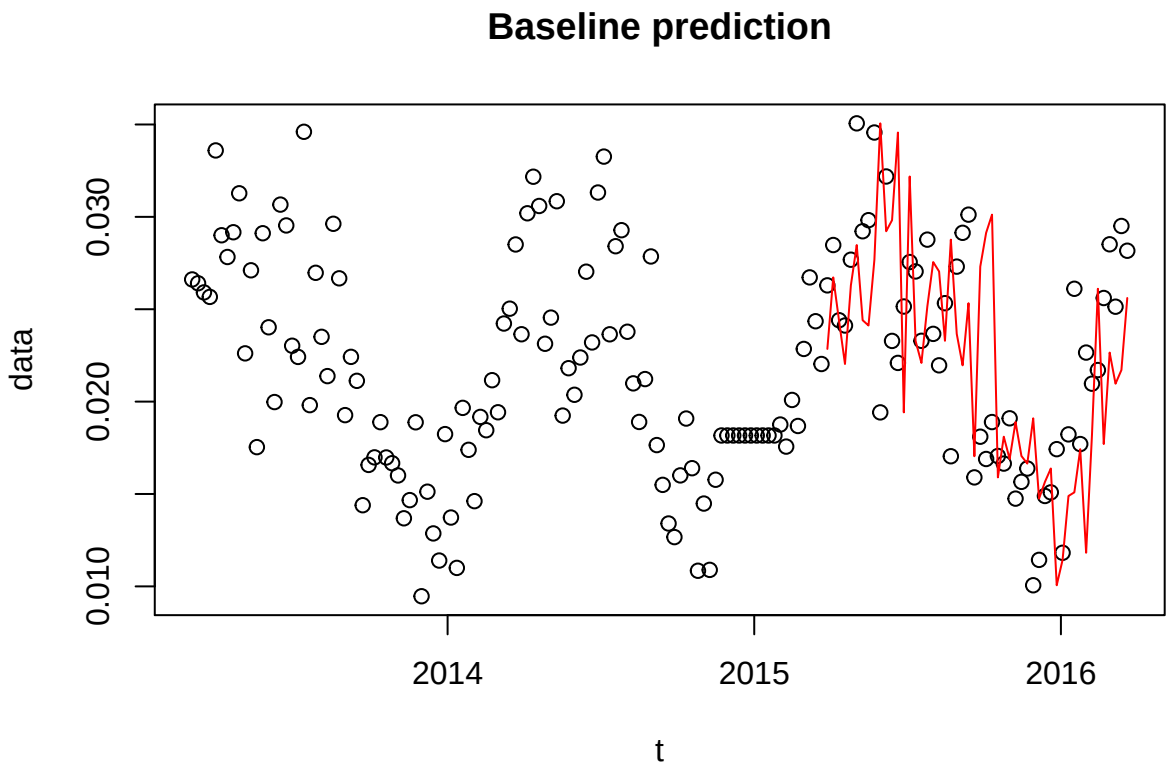


It has improved STL and ARMA model error in the predictions! So we will take it forwards.

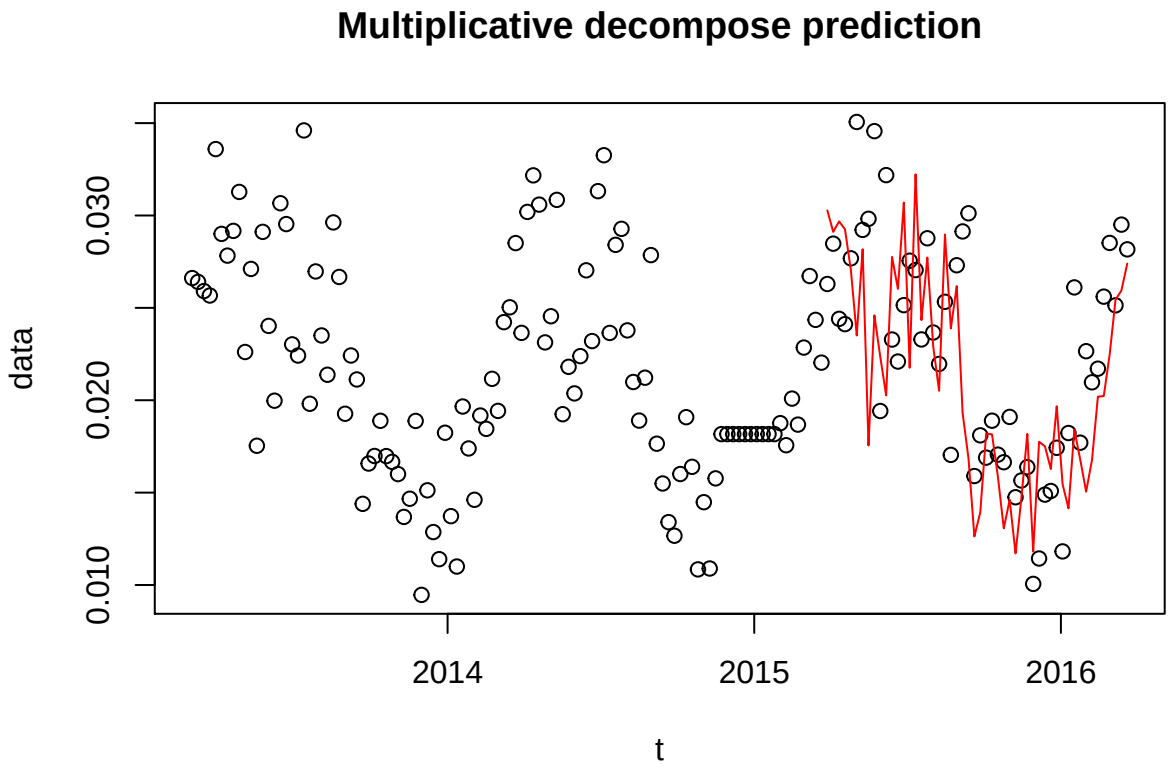
Comparing models in the test data.

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

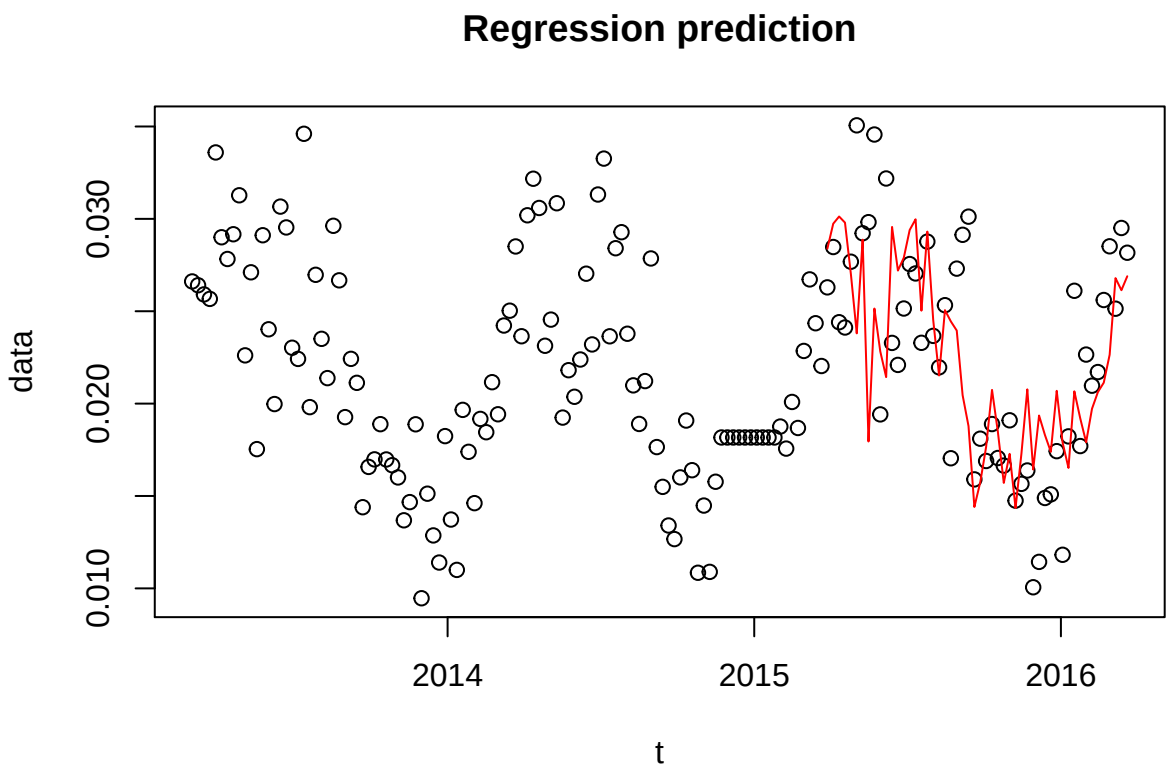
```
## [1] 0.004905538
```



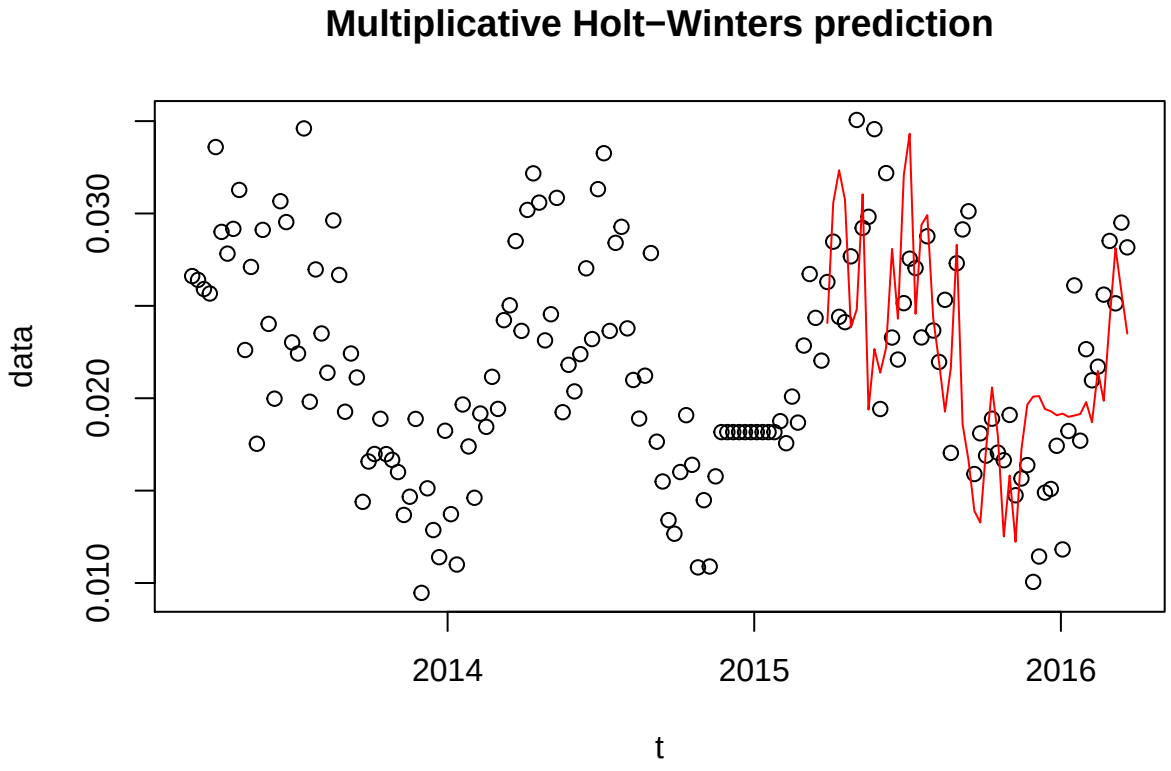

```
## [1] 0.004108195
```



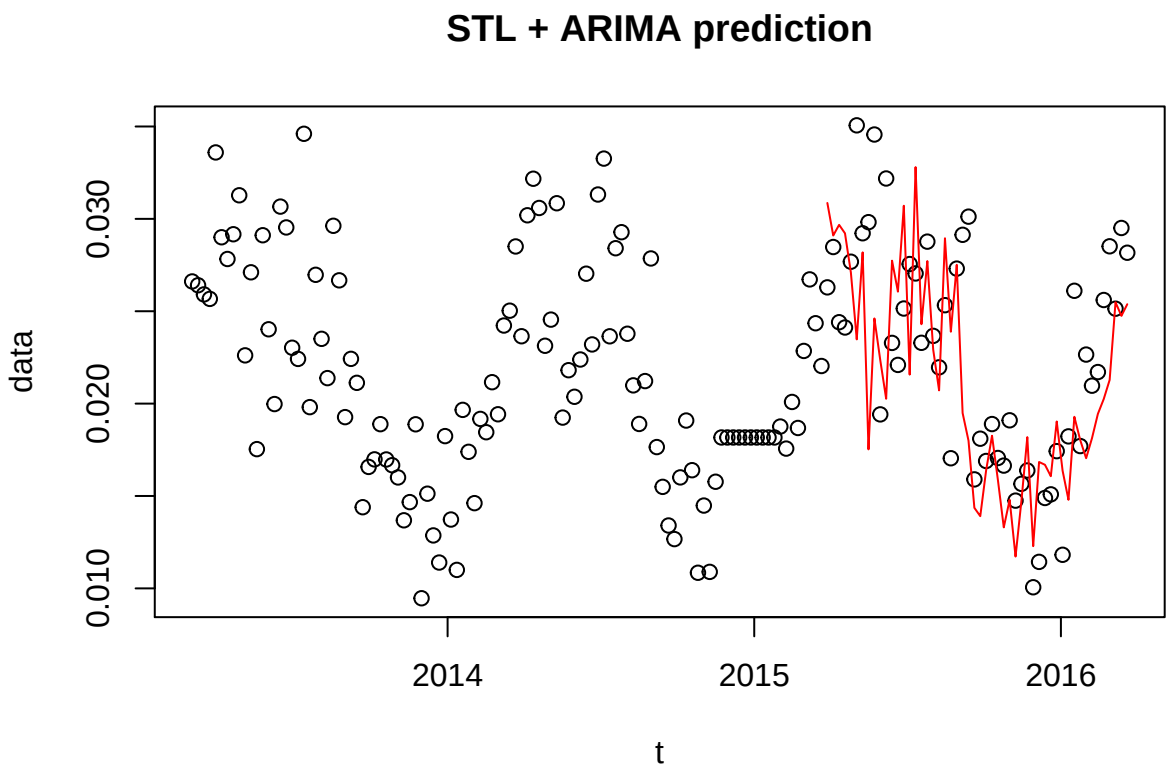

```
## [1] 0.003745867
```



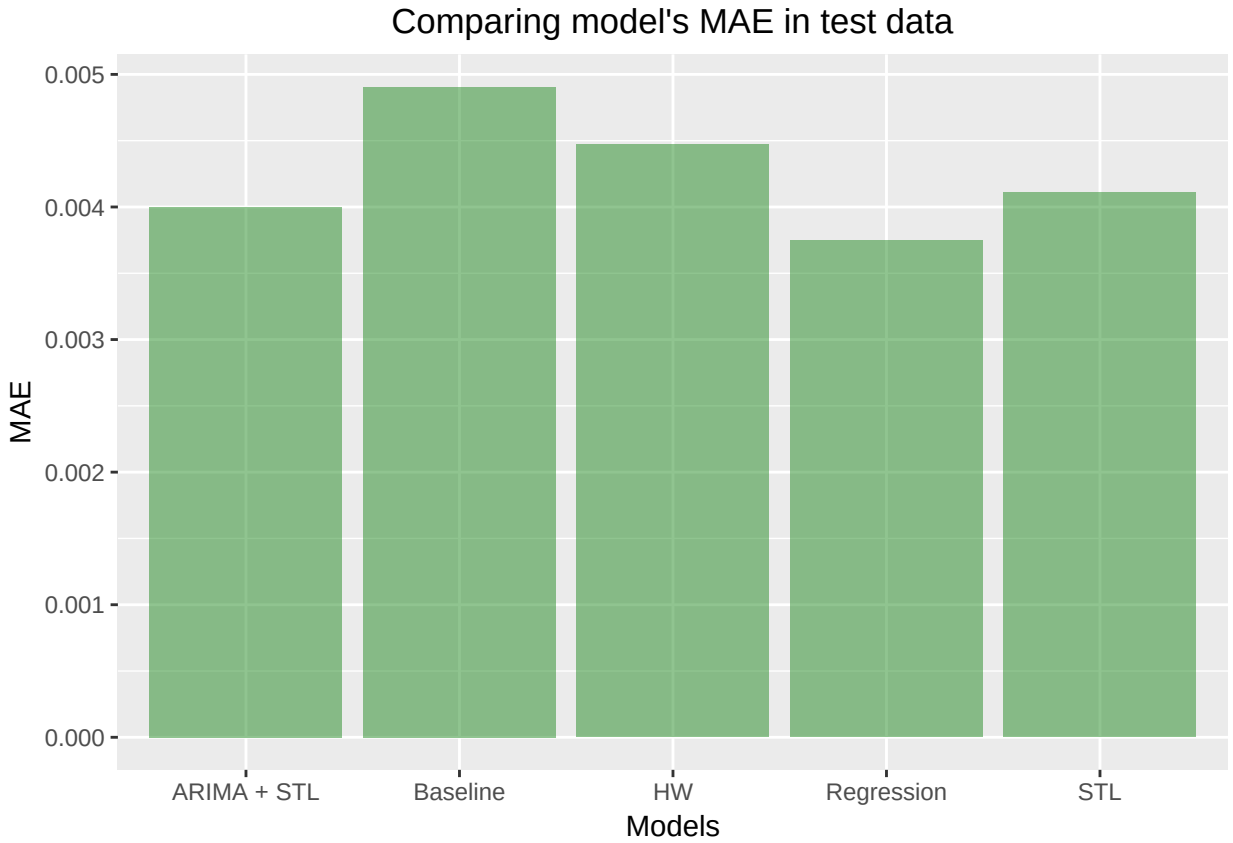

```
## [1] 0.004470292
```




```
## [1] 0.003997573
```



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



It's somewhat impressive that the model with the best performance was the regression. But the other models

also did well.