Danmarks Tekniske Universitet

02417
Time Series Analysis

A3 - Estimating ARMA Processes and Seasonal Processes

Manuel Montoya Catalá (s162706)

November 2016

Contents

Exhaust gasses from combustion engines contain NO_x which is the sum of NO and NO_2 (Sunlight and ozone affects the balance between the two). As part of a national surveillance program the NOx concentration is measured every hour at Jagtvej in Copenhagen. The sensor is located between the road and the bikelane. The data file A3_jagt_NOx.csv is made using "," as column separator. The file contains three columns: The date, the hour within day where the measurement is taken and the concentration of NO_x in $\mu g\ (NO_2 equiv)/m_3$.
You should not use the last 48 hours when estimating your model - as they should be used for testing.
Data originates from:

1 Question 3.1: Presenting the data

Plot the NO_x concentration. Consider plotting for subsets of the data to show the structure. Comment on the behaviour including considerations on stationarity and transformations.
Solution

The first thing we did is loading the dataset and preprocess it in order to be able to work with it. We used the functions substr() and POSIXlt() to join the date and time information and we reversed both arrays so that they go in increasing order of time. We then separated the data into "train" and "test" instances using the Time information, the last 48 hours will be used for testing (prediction). Once this is done we can work with our data.

First thing we will do is plotting the time series as a whole. The things we can appreciate are:

- The data points are crearly correlated, the value at time t depends highly on the value at time $t-1$.
- There is a clear seasonal (dayly) component every 24 samples, which is reasonable since the human activity and the rotation of the Earth (and everything it influences) have daily patterns.
- It seems like the peaks follow a down-trend but I would say we do not have enough samples to be sure. So I would not say so.
- We can see that there are higher values (variance) in the first 2 weeks, and we will have a bigger error in this weeks if we assume a model whose variance is constant.

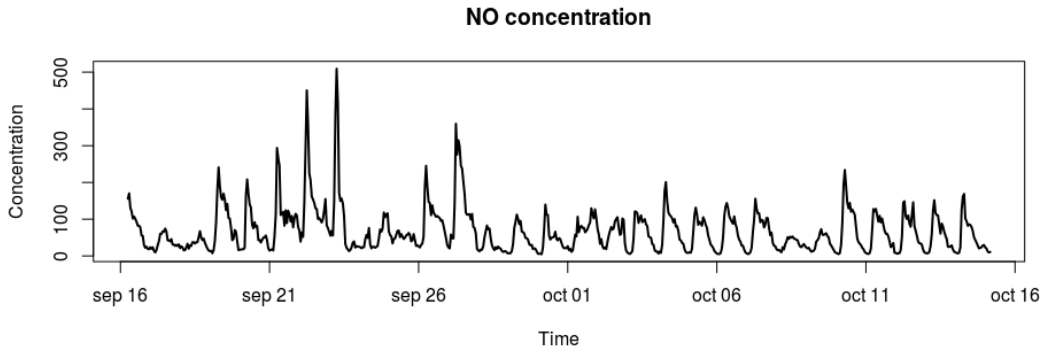

Figure 1: NO concentration for the whole training dataset

- We can also see what appears to be a weekly component. This can be more probably seen in the last 2 weeks where the last 2 days have a lot less production. This could be understandable in the way if these days are Saturday and Sunday, maybe the fabriks do not generate so much polution and there is less movements of cars and public transport.

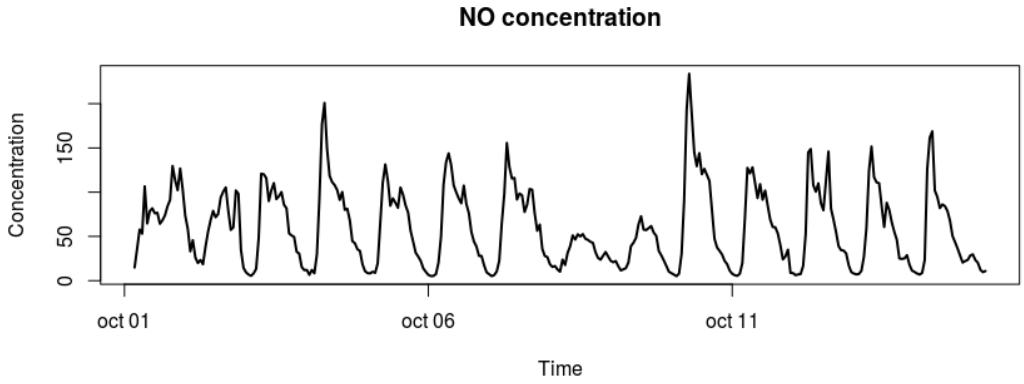

Figure 2: NO concentration for the last 2 weeks of the training set.

This process is not stationary. The fact that the mean, variance and covariance are bounded is not enough. Also, even though assumming that the data is ergodic, in order to estimate the mean and variance, the mean and variance remain constant for a big enough window and the covariance only depends on the time difference.

The problem is not that the values at one given instant depend highly on the values 24 samples before (this could be a stationary process), but rather that the values at a given point of cycle are usually the same all the time (it is not that they depend on the value happening 24 samples ago, but rather, that at every point of the cycle there is a specific mean value). That is the component that we have to remove.

We also ran an Augmented Dickey–Fuller test in R to test it using the function *adf.test()*. This method consists in a statistical test where the Null Hypothesis is $H_0 =$ "The signal is not stationary" by checking that the poles of the system lie within the unit circle in the Z domain. The p-value obtained for this test is $p = 0.031$. This is the probability of observing a datasat more extreme that the one we have given the null-hypotesis. This probability is fairly low, being the usual threshold to reject the null hypotesis $p = 0.05$, so this signal could be characterized as a stationary process since the null hypotesis is rejected, because the data obtained is very unlikely under that hypotesis. This test might fail to see that in the periodic component, the values of the signal are always very similar, so the mean value of the signal for a given point depends highly on the stage of the cycle it is in.

Transformations
Now we will apply some deterministic bijective transformations to the data in order to express it in a way that will make it easier for us to model as an ARIMA process. When we perform the prediction we will have to transform them back into the original transform.

The first transformation [**T1**] is just a **lag** $k = 1$ **differentiation**. In this case we have the signal:

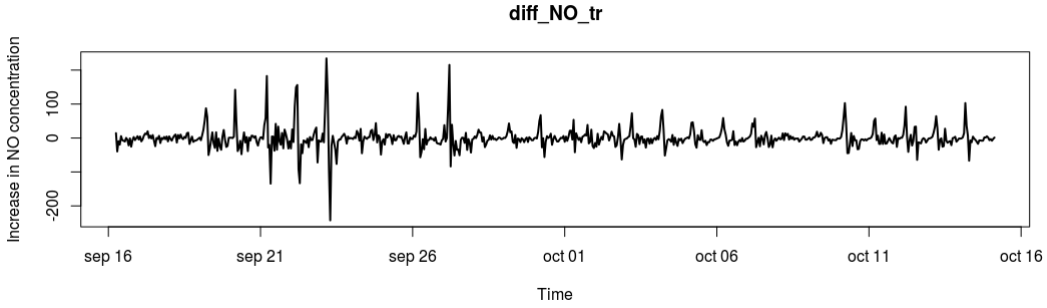

Figure 3: Differentiation Transformation of time

In the signal we can appreciate that the process has mean 0 and the variance in the first two weeks is higher than for the rest of the time as expected. This model should be easier to predict and of course it is stationary since it is a difference process from a process that was stationary at a practical level.

The second transformation [**T2**] is just the **difference of logarithms** of the signal. Taking logatithms will smooth the values, so that the resulting distribution seems more like a Guassian distribution, reducing the heavy tails. In this case we have the signal:

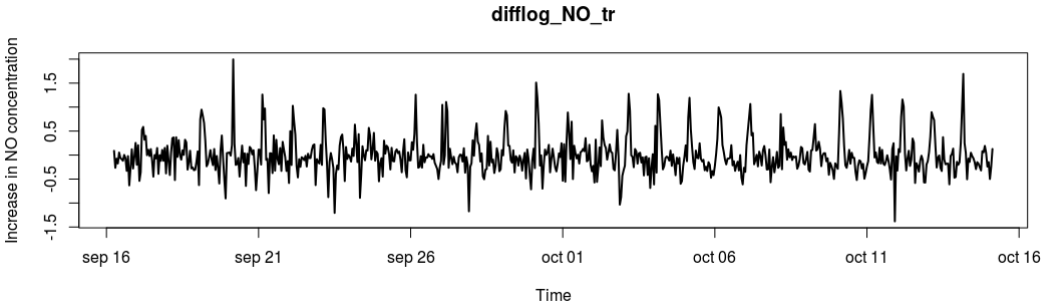

Figure 4: Transformation [T2]

As we can appreciate, the signal obtained is more uniform, we can still see the dayly seasonaly. The signal is still stationary because we have just applied a logarithm to it.

As a third transformation [**T3**] we could perform the differentiation of the season component, for this purpose we can perform **differentiation with** $lag = 24$ .In this case we have the signal:

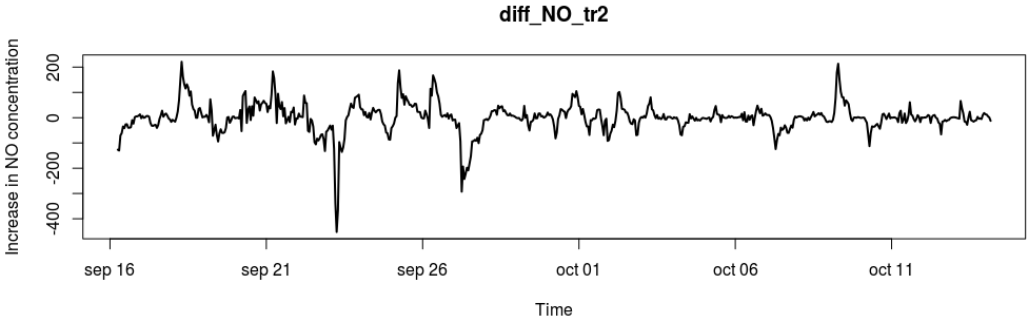

Figure 5: Transformation [T3]

As a final approach [**T4**] we could perform both differentiations, the one with lag 1 and the one with lag 24.

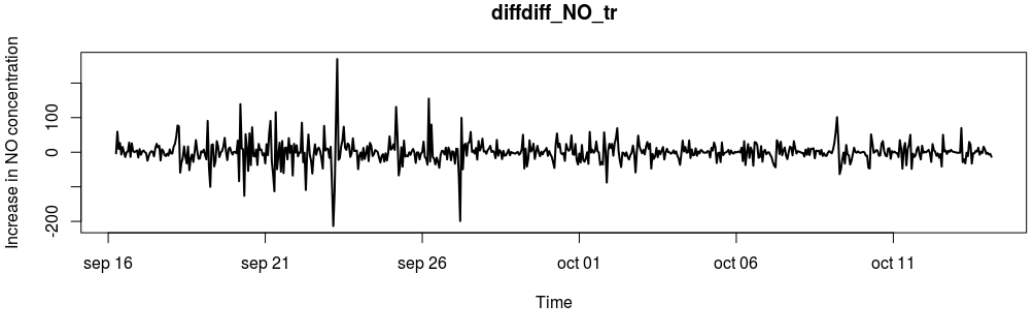

Figure 6: Transformation [T4]

This is the signal that resembles the most to noise, even though at the beggining, the variance seems to be greater. It is a good starting point for the analysis of the signal.

2 Question 3.2: ACF and PACF

Estimate the autocorrelation function and the partial autocorrelation function of the NOx concentration and if relevant also for series derived from the concentration, e.g. transformations

Solution

For the original concentration of NO_2 we have the following ACF and PAFC. The next image shows these values up to lag 200 since we wanted to visualize the weekly component at lag 7*24, nevertheless, it is not clearly appreciated in the data, it might be just because we have insufficient samples and it is a huge lag.

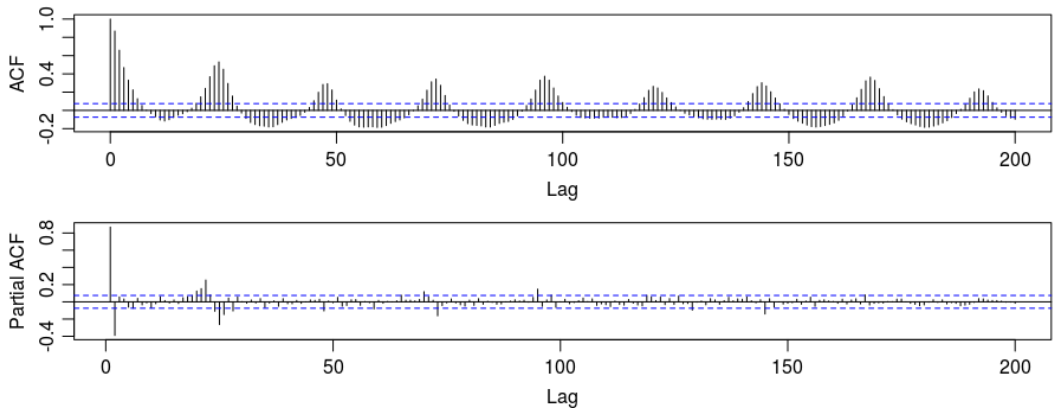

Figure 7: ACF and PACF coefficients for training values of the NO signal

In the ACF we can see an initial exponential decrease, which indicates AR component and we see a periodic trend of lag 24. In the PACF we can appreciate a high AR component at lag 1 and 2 and also a minor component around lag 24. So this model contains AR and seasonal components.

For the [**T1**] transformation, we get the next ACF and PACF coefficients. We can observe a patterns similar to those oberved for the model $A(0,0,1) \times (1,0,0)_{24}$ seen in the previous assignment, that makes up suspect a MA model with an AR periodic component would be a good candidate to model this sequence. It is found that the logarithm version of this has a simmiliar ACF and PACF but in the ACF, the model of the arma model is 2, there is a new component created, so we stick to this model as one of the finalists.

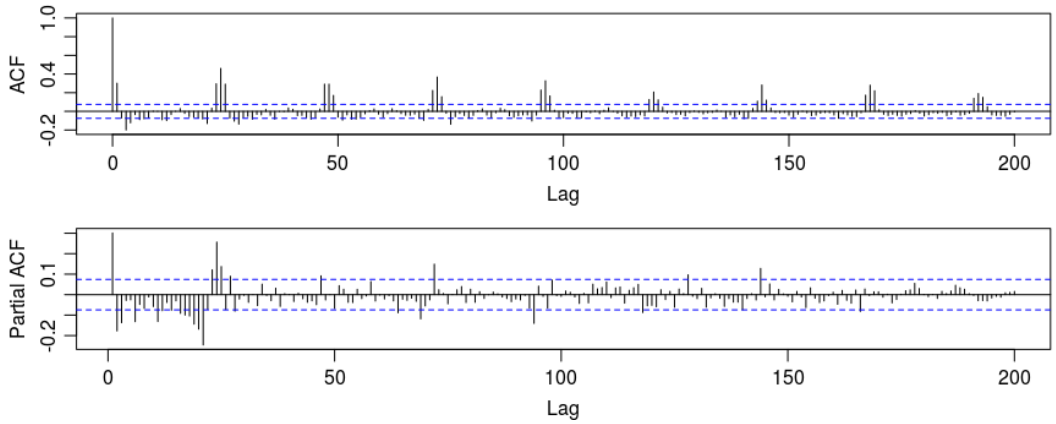

Figure 8: ACF and PACF coefficients for training values of the diff signal of NO

For the difference of the logarithm of the signal we get a very similar graph, since the signal obtained is a smoother version of the original.

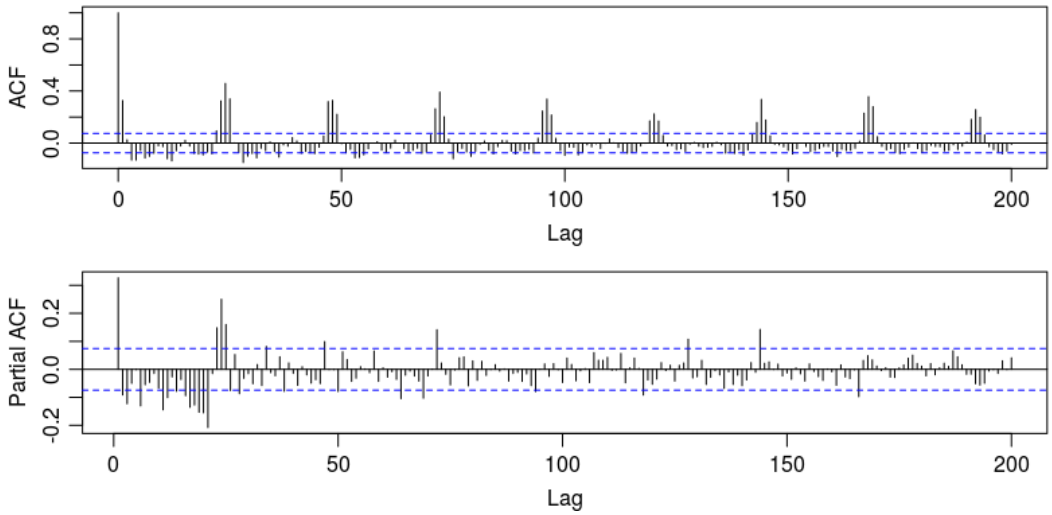

Figure 9: ACF and PACF coefficients.

Now we plot ACF and PACF values for the [**T4**], where we differentiated respect to both lag 1 and lag 24. As we can see the ACF values are almost extinkt, bur in the ACF there are peacks at 24 and 48 so there are periodic AR components. This is more characteristic maybe of a $A(0,0,0) \times (2,0,1)_{24}$ seen in the previous assignment.

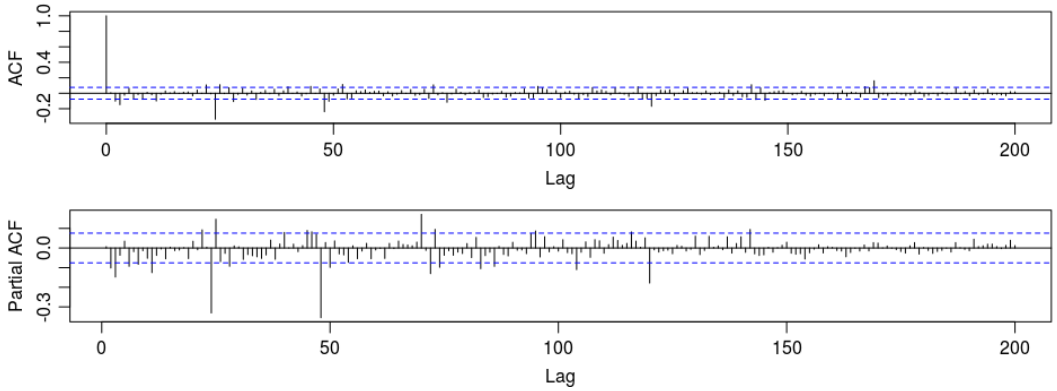

Figure 10: ACF and PACF coefficients

3 Question 3.3: Model selection

Select an initial model structure. Estimate the parameters. Validate the model. Consider tests for lower model order. Consider updating the model structure.
Argue for the choices you make. Remember that the model building process is an iterative process and you should always consider stepping back and reconsider your choices

Solution

From the results obtained in the previous questions we conclude that as first approach, the ARIMA system that will be best express the time series is a seasonal ARIMA with period $s = 24$ with a difference of order 1 for both the normal and seasonal component ($d = 1, D = 1$). Using the notation of the book we have the general model.

$$A(p, 1, q) \times (P, 1, Q)_{24}$$

We will try out different values of p, q, P, Q, and evaluate their residuals for the training data points. But first we will try models that do not take into account this and model the data without using differences.

The first basic model we will try, for comparison with the best one we can find is a naive model where we do not take into account the properties of the sigal that we have seen in the previous questions. We just use a seasonal model with AR and MA components in the normal and seasonal parts.

Model $A(2, 0, 2) \times (2, 0, 2)_{24}$

This model contains 8 parameters in total 2 MA and 2 AR for the seasonal component and 2 MA and 2 AR for the normal component. The obtained **coefficients** $\hat{\theta}_i$ are shown in the next table, along with their estimated standard deviation $\hat{\sigma}_{\hat{\theta}_i}$ and the p-value obtained under the null hypotesis H_0: "The parameter is zero, $\hat{\theta}_i = 0$". Under this hypotesis, the statistic $T = \hat{\theta}_i / \hat{\sigma}_{\hat{\theta}_i}$ follows a t-student distribution with $N - p - q - 1$ degrees of freedom. If the p-value is lower than $p = 0.05$ we can reject the null hypotesis and assume that the value is significant, $\hat{\theta}_i \neq 0$.
As we can see in the figure below, the coefficients ar1 and ar2 are the components with the highest p-value so we cannot assume that they are distinct from 0, the rest of the parameters have a very low p-value, specially sar1, meaning that there is a strong seasonal AR component.

Coeff	ar1	ar2	ma1	ma2	sar1	sar2	sma1	sma2
$\hat{\theta}_i$	0.275	0.443	0.769	0.14617	0.695	0.3030	-0.4645	-0.486
$\hat{\sigma}_{\hat{\theta}_i}$	0.22	0.204	0.214	0.0488	0.112	0.112	0.097	0.095
p-value	0.2	0.03	3.4 e-04	2.8e-03	8.6 e-10	6.8e-03	2.2e-06	3.8e-07

Table 1: Coefficients model $A(2, 0, 2) \times (2, 0, 2)_{24}$

In the following, we will perform an analysis of the residual signal, in which we will boil down the residuals to a set of values that will give us different information about them. We will use this information see the properties of each system and how they compare to each other.

The next image shows the residuals obtained for the estimated parameters, along with their **ACF and PACF** up to lag $k = 200$. As we can see, the variance of the residual in the first week is bigger, this was predictible because in the original signal, the values in this timeframe are oddly higger than in the rest of the signal, and since we do not consider a trend in the model (or varying variance), these points are wrongly estimated with a higher error. The ACF signal is pretty good, with some small peaks at lags multiples of $s = 24$ but not quite significant. The PACF shows some minor damping oscillations that become as high as 0.10 sometimes in labs multiple of $s = 24$ but are also not very significant.

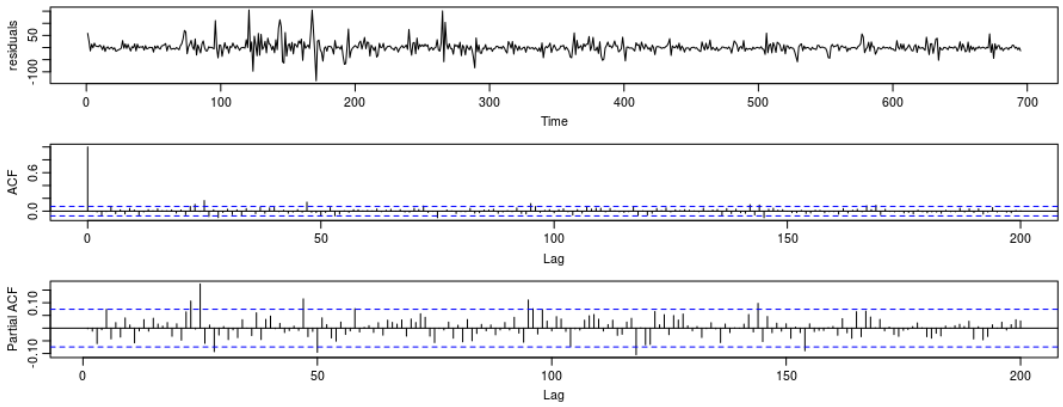

Figure 11: ACF and PACF coefficients of the residual

In a White Noise process, the **probability of change in the sign** of the signal is 0.5 so we run a statistical test whehe the null hypotesis is that H_0: The signal follows a White Process, and thus, the change of sign in the signal follow a Binomial distribution with probability of a change in sign $p = 0.5$. If the p value is lower than 0.05, we can dischard the null hypotesis. In this case, the p-value is $p = 0.0527$ which is in the threshold, so we can be sure of rejecting the null hypotesis. The signal could be white noise according to this statistic.

If the residual is actually white noise, then the values of its autocorrelation function will follow a Gaussian distribution $\hat{\rho}_\epsilon(k) \sim N(0, 1/N)$ for all $k > 0$. So we can perform a statistical test where the null hypotesis is $H_0 =$ The signal is white noise, so the **autocorrelation values should be white noise** as well. Under this hypotesis, the sum of the squared estimated autocorrelation values should follow a χ^2_h distribution with $h = m - n$ degrees of freedom where m is the number of lags used and n the number of parameters of the model. For this model, using the autocorrelation values up to lag 15, the p-value is $p = 2.3e - 06$, so we can reject the null hypotesis. Meaning that this test tells us that the residual can not be considered white noise.

The next image shows the p-values for the **Ljung-Box statistic**. This statistic works as follows: Since we are trying to check if the residual follows a white noise distribution, we perform a statistical test in which the null hypothesis H_0 is that there is no autocorrelation in the errors, that means the ACF values are 0 or any value small enough to be cause by the randomness of sampling. The statistic used under the null hypotesis is:

$$Q = n\,(n+2) \sum_{k=1}^{h} \frac{\hat{\rho}_k^2}{n-k}$$

where Q follows a χ^2_h chi-squared distribution with h degrees of freedom.
The P value indicates the probability of obtaining a result at least as unusual as the one in the samples, assuming that the null hypothesis is true. $Pvalue = P(x > D|H_0)$. If the P-value is big, that means that the probability of obtaining the data giving the null hypotesis is big, which could validate the null hypotesis. In this case, we want big values to not reject the hypotesis $H_0 =$ "The signal is not correlated ".
As we can see in the image, the correlations up to lag 24 have p-values higher that 0.05 which in the literature is enough to not have enough evidence to reject the null hypotesis. But as we advance, we see that the null hypotesis is rejected. This is expected since correlation usually appear at high lags. There is a weird happening in lags from 5 to 12 where the p-value are lower for some reason, this indicates that the correlation at this lags could exist in the residual.

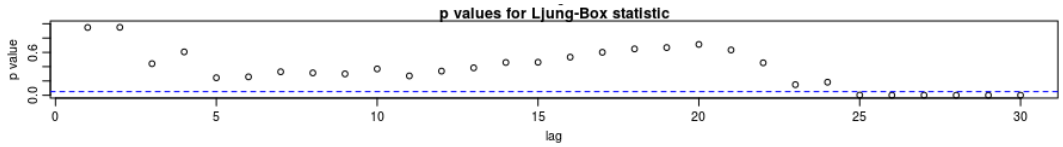

Figure 12: Ljung-Box statistic

The next figure shows the **Histogram and the QQ plot** of the residual. We can appreciate that the distribution has heavier tails than the normal distribution and even some outliers (points that are very unlikely under the gaussian assumption). So the noise distribution is not specially gaussian.

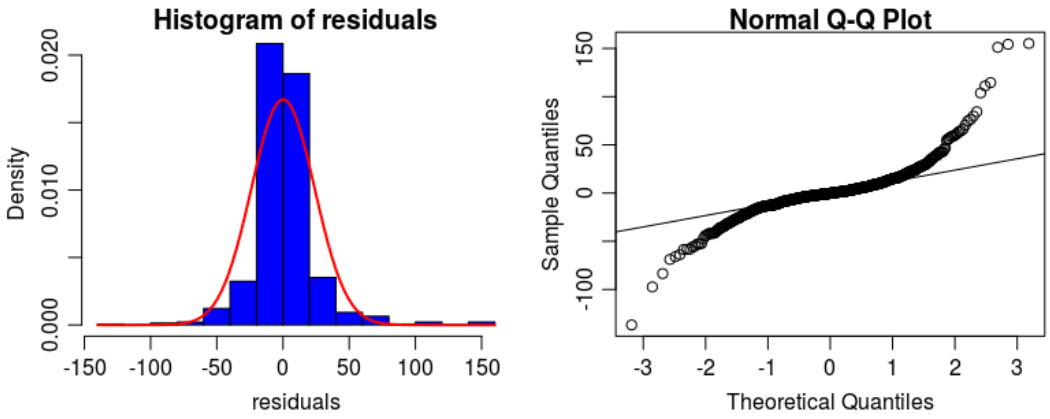

Figure 13: Histogram and Normal QQ plot

Next, we show the residuals obtained, against a **simulated random noise signal** obtained using the same variance as the estimated for the residual. As we stated many times, we can see that in the first half, the variance is higher and in the second half, the variance is lower. Since the variance changes over time, we cannot consider it a white noise process. Also, in the residurals, the errors seem to spike from time to time, which also indicates that the residual is not White Noise.

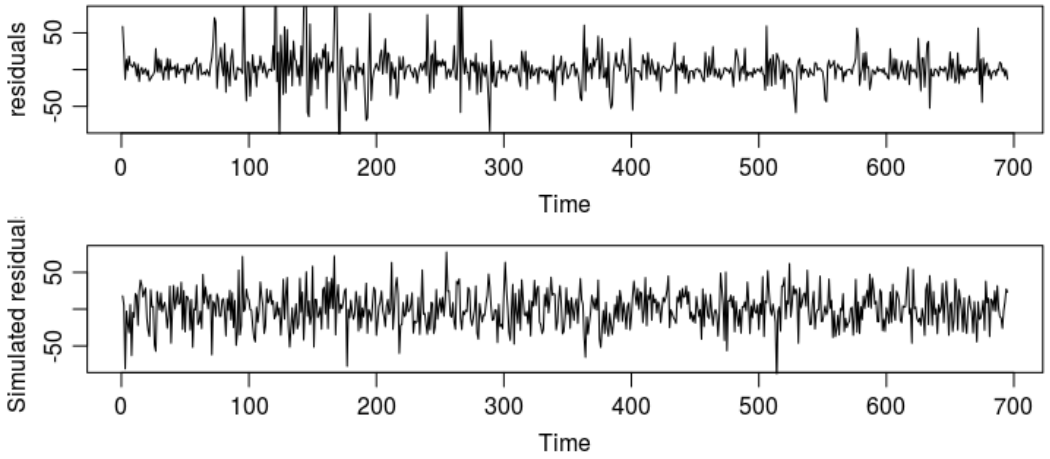

Figure 14: Simulated residual

The next image shows the **cummulative periodogram** of the signal. We can express any signal as a sum sinusoidal functions of different frequency, modulus and phase using the Fourier Transform. The Fourier Transform of a White noise signal is a constant function of frequency, so the contribution of each frequency component is the same, thus, the cummulative sum of the contribution over the frequencies should be a straight line. The cummulative periodogram allows us to see how energy is distributed along the frequency components of our signal, and check if the signal has the frequency properties of white noise.

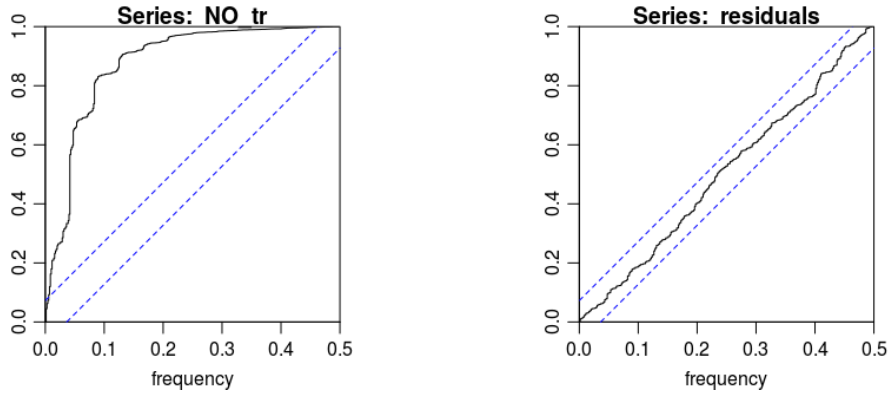

Figure 15: Cummulative Periodogram of the original signal and the residual

We can see that the energy of the frequency components of the original data is not evenly distributed so we can conlclude (as we knew) that the original data is not white noise. The residuals do seem to follow the white noise distribution in the frequency domain, so according to this test the residual could be white noise. Too bad we have seen other tests that say otherwise.

Among these parameters, we can also find others that tell us more information about the system. Many of them are not very informative on their own but they can be informative compared to other models. The table below shows a **set of parameters** that give more information about the performance of the system, these parameters are:

- **Variance of the Residual** σ_ϵ^2: Estimated variance of the residual over the training samples.
- $S(\Theta_i)$: Mean of sum of the squared errors of the estimation of the training data. It is the equation (6.99) divided by the number of traning samples.
- **-loglik**: Negative Log likelihood of the data given the model.
- **AIC**: Akaike's Information Criteria. Bayesian. Given by equation (6.102) in the book.
- **BIC**: Bayesian Information Criteria. Bayesian criteria that takes into account the complexity of the system. Given by equation (6.103) in the book.

Parameter	σ_ϵ^2	$S(\Theta)$	-loglik	AIC	BIC
Value	568	568.35	3214	4424	4460

Table 2: Extra parameters for model $A(2,0,2) \times (2,0,2)_{24}$

A final parameter we will obtain for the process is the sum of the square error for the test samples. Since our model can suffer from overfitting, relying on the sum of squared error of the trianing samples is not a good practice, this error will decrease as we increase the number of parameters of our model but our validation error can increase. The mean square error for the test samples is:

$$S^{'}(\Theta) = 1697$$

We can see that it is about 3 times bigger than the mean squared error for the training samples so might have overfitting (or that the test samples do not follow the training samples distribution).

Model $A(0,0,2) \times (2,0,2)_{24}$

For the sake of comparing nested models and test the important of adding and removing parameters, we will consider now the model $A(0,0,2) \times (2,0,2)_{24}$ in which we removed the non-seasonal AR-component because it was the less significant in the previous model. We will not go into detail as we did in the previous model, we will just point out the last parameters. We can appreciate that when we reduce the number of parameters the residual variance increases.

Parameter	σ_ϵ^2	$S(\Theta)$	-loglik	AIC	BIC
Value	718	718	3305	4582	4609

Table 3: Extra parameters for model $A(0,0,2) \times (2,0,2)_{24}$

The validation mean squared error is:

$$S^{'}(\Theta) = 2219$$

Which is a big decay from the previous model value. Thus it is a good idea to include the AR components.

We want to say that we tried to implement the $A(1,0,2) \times (2,0,2)_{24}$ model, but for some reason, some of the variances of the parameters were negative, so we couldnt perform a proper study. (Maybe because the process was seen as no stationary ?)

Model $A(0,1,1) \times (0,1,1)_{24}$

Now we go with models that use the differentiation of normal and seasonal component. This model has been the one chosen from this kind because it uses the least significant parameters while obtaining the

same results. Comparing this example with the previous one, we can see how the differentiation can significantly reduce the number of parameters needed for the model. Since this model is not nested with the previous one, we will not make a significance test.
This model contains 2 parameters in total, 1 MA and seasonal component and 1 MA for the normal component. It seems that since we have differentiated the signal both in the normal and seasonal components, the AR part of the model has become less significant. The obtained **coefficients** are shown in the next table. As we can see in the figure below, all coefficients have significant values.

Coeff	ma1	sma1
$\hat{\theta}_i$	0.1614	-0.915
$\hat{\sigma}_{\hat{\theta}_i}$	0.04	0.031
p-value	5.6e-05	3.1e-123

Table 4: Coefficients model $A(0,1,1) \times (0,1,1)_{24}$

The next image shows the residuals obtained for the estimated parameters, along with their **ACF and PACF** up to lag $k = 200$. As we can see, the variance of the residual in the first week is still bigger.The ACF signal is pretty good, with some small peaks at lags multiples of $s = 24$ but not quite significant. The PACF shows some minor damping oscillations that become as high as 0.10 sometimes in lags multiple of $s = 24$ but are also not very significant.

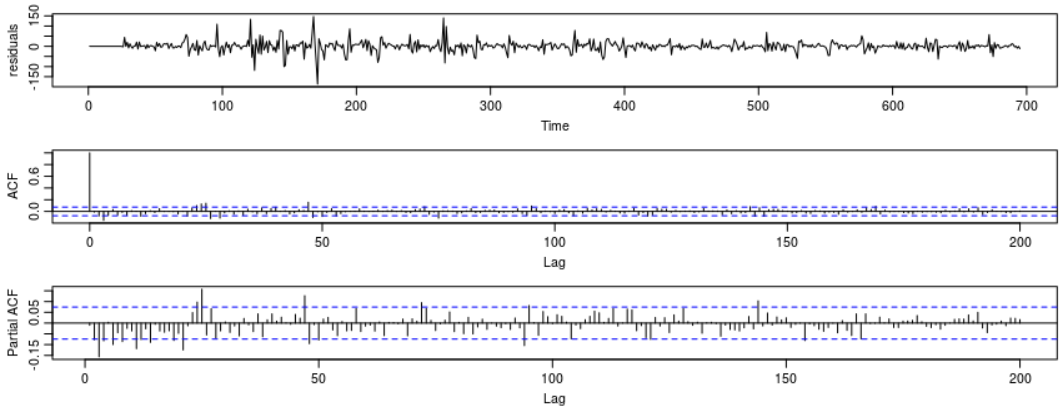

Figure 16: ACF and PACF coefficients for the residual

The p-value for the **"change in sign test"** the is $p = 0.6763$ so we cannot reject the null hypotesis that the noise is Gaussian. Good stuff.

The p-value for the test of "the autocorrelation values are white noise" is $p = 0.00014$, so we can reject the null hypotesis. Meaning that this test tells us that the residual can not be considered white noise.

The next figure shows the **Histogram and the QQ plot** of the residual. We can appreciate that the distribution has heavier tails than the normal distribution and even some outliers (points that are very unlikely under the gaussian assumption).

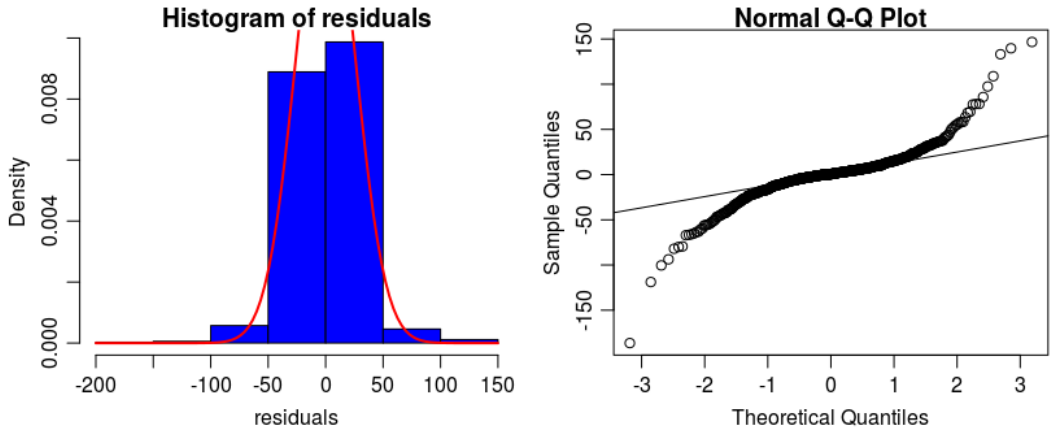

Figure 17: Histogram and QQ plot

Next, we show the residuals obtained, against a **simulated random noise signal** obtained using the same variance as the estimated for the residual. As we stated many times, we can see that in the first half, the variance is higher and in the second half, the variance is lower.

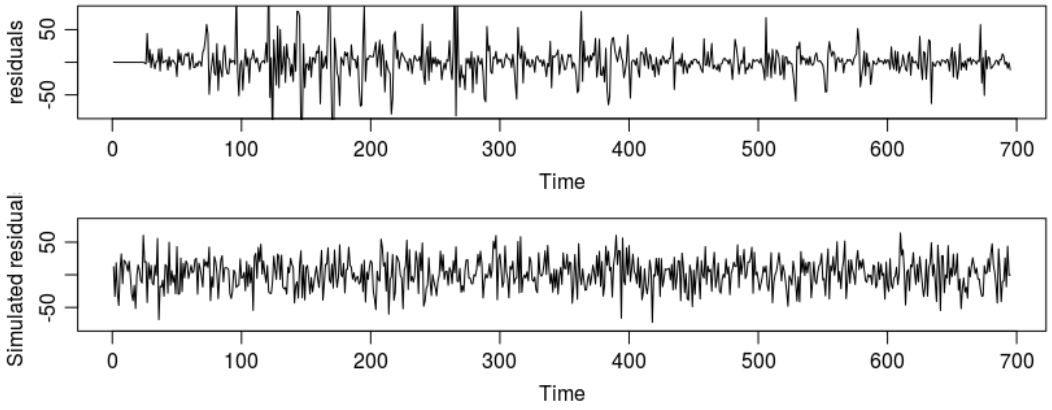

Figure 18: ACF and PACF coefficients for training values of the NO signal

The next image shows the **cummulative periodogram** of the signal. We can see the sabe things as in the previous model.

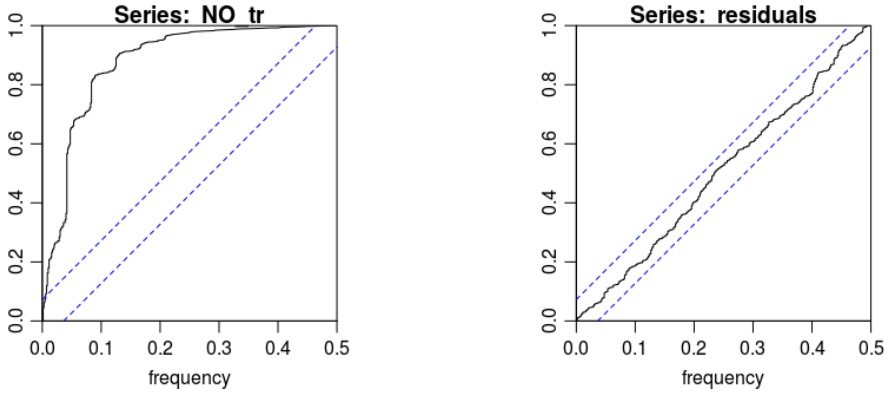

Figure 19: Simulated noise

The rest of the parameters for this model are displayed in the following table. Compared to the first presented model we have worse values for the variance of the error σ_ϵ^2, $S(\Theta)$ and the information criteria AIC and BIC but we get a better likelihood for the data given this model.

Parameter	σ_ϵ^2	$S(\Theta)$	-loglik	AIC	BIC
Value	666	642	3150	4497	4506

Table 5: Extra parameters for model $A(0,1,1) \times (0,1,1)_{24}$

A final parameter we will obtain for the process is the sum of the square error for the test samples. Since our model can suffer from overfitting, relying on the sum of square error of the trianing samples is not a good practice, this error will decrease as we increase the number of parameters of our model.
The validation mean squared error is:

$$S^{'}(\Theta) = 1430$$

Which is a big improvent from the one obtained in first model. This shows that we can significantly reduce the number of parameters needed for our model obtaining better results if we use the differentiation. Since the first model has better training residual variance but worse validation mean squared error, we can conclude that the first model performs overfitting

4 Question 3.4: Predictions

Use the model you have developed for predicting the NOx concentration 48 hours ahead and include prediction limits. Compare with the data that was left out. Include a table with the 1h, 24h and 48h predictions.

Solution

The next image shows part of the training samples and the test samples in black and the predicted samples along with $\pm\hat{\sigma}_\epsilon$ in blue for the $A(0,1,1)\times(0,1,1)_{24}$ model. We can appreciate how the uncertanty (standard deviation) of the prediction grows as we move away from the trianing samples.

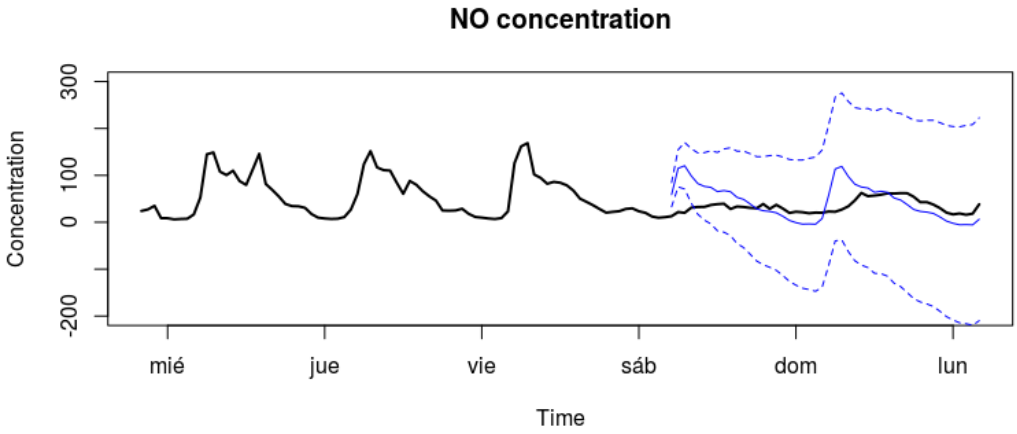

Figure 20: Prediction for the $A(0,1,1)\times(0,1,1)_{24}$ model

As we said earlier, it looks there is a weekly component that makes saturdays and sundays have lower values of concentration, since we are not modelling this seasonality and the test samples are these two days we can see we make more error than we should. But we can see how the predictions can model the daily component. We can also see that the estimation can incluse also negative values of the signal, we know this is not possible but we have not put this knowledge into the algorithm. The next table shows the prediction values asked:

Time	1 hour	24 hours	48 hours
Prediction	58.92	7.79	6.37
Standard deviation	25.83	146.2	216.3

Table 6: Predictions $A(0,1,1)\times(0,1,1)_{24}$

5 Appendix: About the logarithm transformation

I have had a hard time using the logarithmic transformation of the input data in order to model better the sequence. So I will try to explain here some results and thoughts that I have come up with. In the book, it is recomended to perform a transformation of the original signal depending of the properties of its Mean to Range graph, the kind of transformation is given by equation (6.111).

The next figure shows the graph for different values of the window l. If a relation between the mean and the range is found, then a transformation could be benefitial, this relation means that there are areas in the time series with values higher than usual, which could be explained with a higher variance. Performing this transformation we seek to "saturate" the points with higher variance, so that they are not as high in the new domain. Also any exponential trend will be transformed into a linear trend that can be easily eliminated by differentiation.

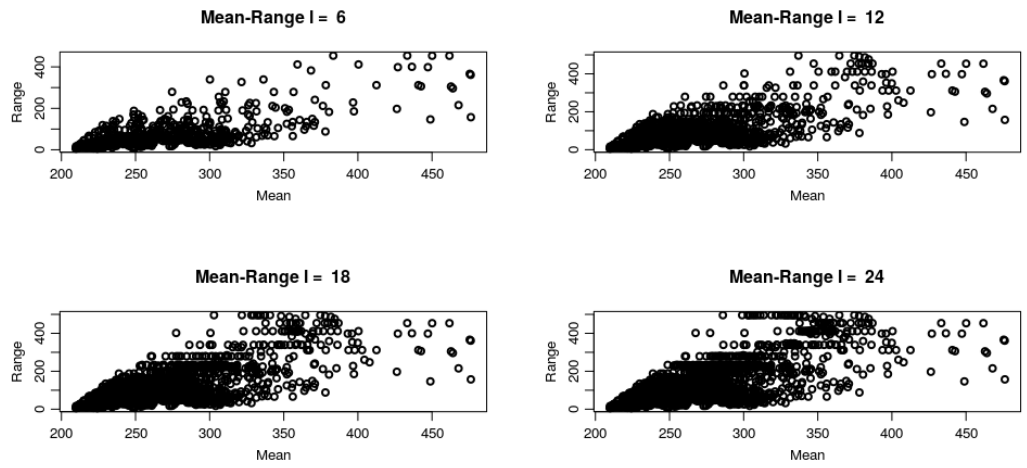

Figure 21: Mean Range graphs

The proposed transfomation given the linear relation between the range and the mean is the logarithmic transformation. The log function and its derivative can be seen in the next figure. As we can see, the log transformation is most linear at $x = 1$, saturating values higher than it and increasing values that are lower. We can see that its derivative is $1/x$. This transformation would lower the tails of the original signal.

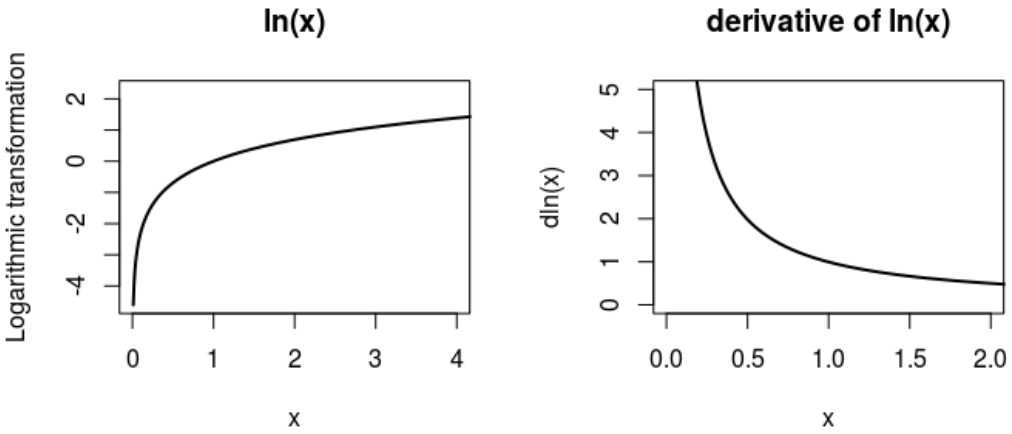

Figure 22: Logarithmic transformation

Of course this transformation could only be applied to a strictly positive signal. Once it is obtained, the output signal is usually differentiated with lag $k = 1$ obtaining the signal:

$$Y_n = log(X_n) - log(X_{n-1}) = log\left(\frac{X_n}{X_{n-1}}\right)$$

A common practice is, instead of doing this, obtain the returns of the original signal, and move them to the linear part of the logarithm transformation adding 1.

$$Y_n = log(1 + r_n) = log\left(1 + \frac{X_n - X_{n-1}}{X_{n-1}}\right) = log\left(\frac{X_n}{X_{n-1}}\right)$$

Which ends up being the same as the previous equation. Once again, we have to hope that the process X_n is either strictly positive or strictly negative, otherwise, the value could be negative or infinity. We will asume that this is the case, if the signal does not meet this requirement, I guess you can always add a constant value to the process so that it does.

This transformation is highly dependent on the mean of the orginial signal, lets say we add a constant value μ_* to the whole process. This will produce a smoothing on the signal since the division will tend to 1, thus being in the symetric, linear part of the transformation.

$$Y_n = log\left(\frac{X_n + \mu_*}{X_{n-1} + \mu_*}\right)$$

Maybe we can play with this value to obtain the best model for our data.

Model $A(1,0,1) \times (1,1,1)_{24}$

We have fitted a model $A(1,0,1) \times (1,1,1)_{24}$ to our transformed signal, using the normal logarithmic transformation, notice that we do not differentiate again in the normal component because the transformation already has the differentiation. We can obtain the residuals and prediction of the model respect to the original signal, by just reversing the transformation.

The residuals of this model in the non-transformed domain, and its ACF and PACF can be seen in the next graph. As we can see, the residuals have lower absolute values than those obtained in the models that did not operate in the transformed domain, so we the transformation produced an improvement, that means that the residual of the ARIMA model resembles more guassian noise than in the transformed domain, than in the original domain.

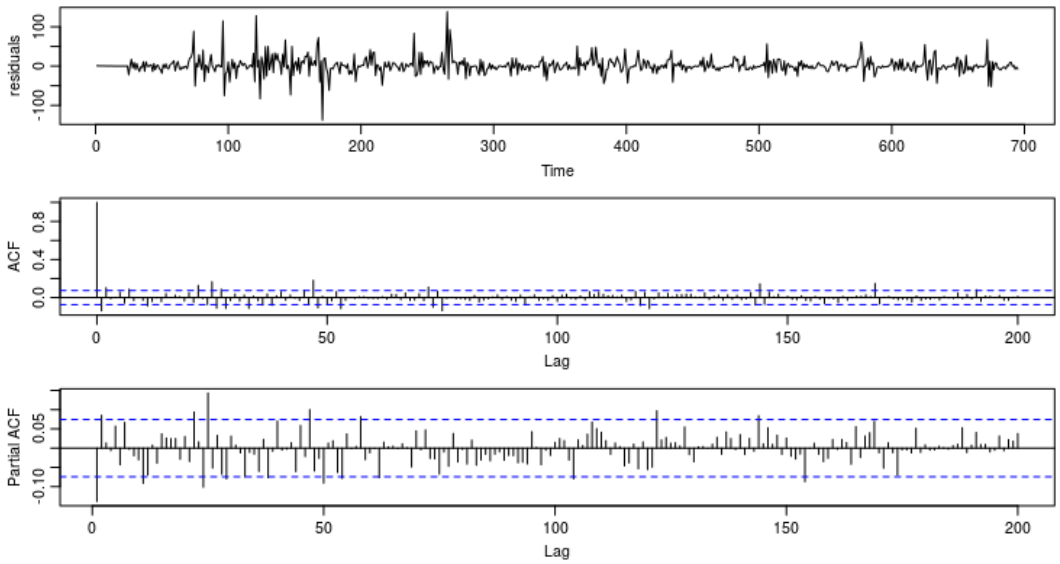

Figure 23: ACF and PACF of the residual

We can calculate the variance of the noise for the training samples and for the testing samples, being this values $S(\Theta) = 420$ and $S^{'}(\Theta) = 568$. Both are simmilar and huge improvements over the previous models, thus meaning that this model is the one that better models the signal. In the next figure we can see the predictions for this model, once they have been transformed to the original domain.

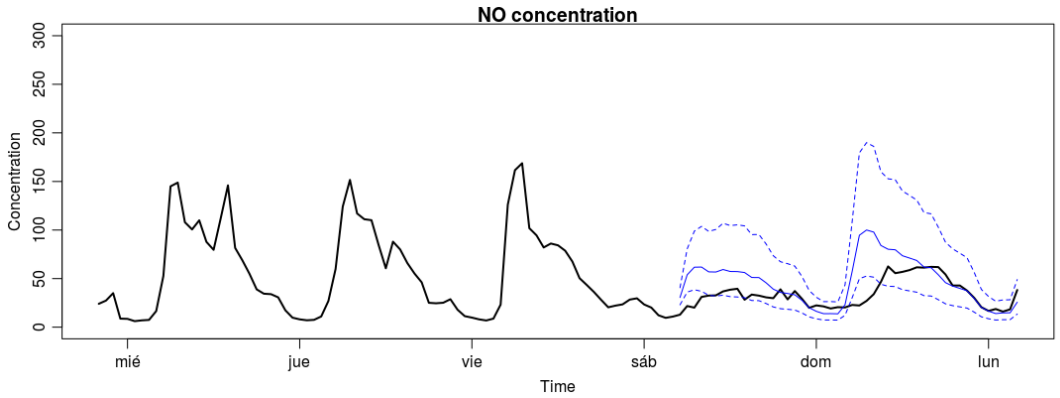

Figure 24: Prediction for the Transformed model

In can be observed that the standard deviation is not symetric since it has been de-transformed to the original domain, where the value of the signal must be positive. In this case, we have been able to transmit the information that the value cannot take negative values.

Finally we will play a bit with μ_*, we will add more mean value to the process, so that the $\left(\frac{X_n+\mu_*}{X_{n-1}+\mu_*}\right)$ is closer to 1 and therefore lies more in the linear part of the logarithm. With $\mu_* = 0$ we obtain $S(\Theta) = 418$ and $S^{'}(\Theta) = 869$. The next plot shows the prediction along with its $\pm$ standard deviation lines. As we can see, they are still asymetrical but they are more symetrical than in the previous model. The test error is worse but it just might be because the test samples oddly enough are the saturday and sunday.

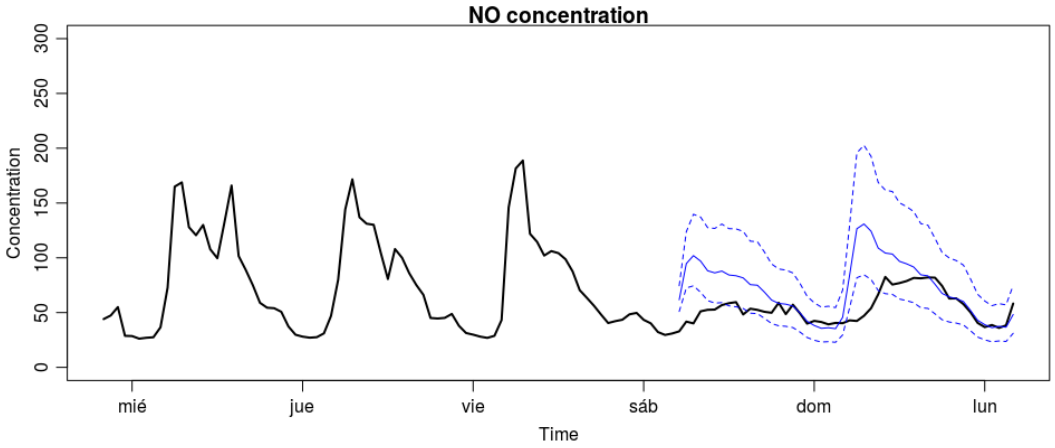

Figure 25: Final prediction

6 R code

This assignment has been entirely coded in R, since some variables are reused from one Question to the next, here I presen the code for all questions jointly. The code is fully commented and easy to follow.

```
plot_timeSeries <- function(X,Y, name, xlabel, ylabel, lwd, gp){
  # gp: If set to 1, then it generates a physical image
  if (gp == 1){
    png(file = name,width=800, height = 600, res=130)
  }
  # Size and aspect ratio could be width=400,height=350,res=45
  plot(X, Y,
       type = "l",                      # Draw as a line
       lwd= lwd,                          # Line width
       main= name, # Title of the graph
       xlab=xlabel,              # x label
       ylab=ylabel

       )                   # y label

  if (gp == 1){
    dev.off()
  }
}

plot_acfpacf <- function(X, name, Nlag, gp){
  # gp: If set to 1, then it generates a physical image
  if (gp == 1){
    png(file = name,width=800, height = 600, res=130)
  }
  par(mfrow=c(2,1), mar=c(3,3,1,1), mgp=c(2,0.7,0))
  acf(X, lag.max = Nlag )
  pacf(X, lag.max = Nlag )

  #title(name);

  if (gp == 1){
    dev.off()
```

```
  }
}

plot_SigACFPACF <- function(residuals, Nlag){
  par(mfrow=c(3,1), mar=c(3,3,1,1), mgp=c(2,0.7,0))
  plot(residuals, type="l")
  acf(residuals, lag.max = Nlag)
  pacf(residuals, lag.max = Nlag)
}

################ Question 3-1 ######################
## Loading and preparing the data:
myData = read.csv("./A3_jagt_NOx.csv", sep = ",") # read the csv
date = as.matrix(myData[,1])    # Get the date
hour = as.matrix(myData[,2])    # Get the hour
NO = as.matrix(myData[,3])      # Get the NO concentration

Nsam = dim(NO)[1]               # Number of samples

# Convert date and hour to timestamp
# casa = as.matrix(strsplit(hour, "-"))[1,]    # Useless
hour <-substr(as.character(hour),2,3)
myTimeStamp <- as.POSIXlt(paste(as.character(date),hour),format="%d-%m %H")

## We reverse the data !!
myTimeStamp = rev(myTimeStamp)
NO = rev(NO)

NO = NO + 200    # Supermean


# Vars for plotting
minNO = min(NO)
maxNO = max(NO)
rangeNO = maxNO - minNO

### Split data into train and test
Ntst = 48
tr_indx = 1:(Nsam - Ntst)  # Indexes for estimating
Ntr = length(tr_indx)
tst_indx = (Ntr + 1):Nsam   # Indexes for estimating
Ntst = length(tst_indx)

# Obtain data
myTimeStamp_tr = myTimeStamp[tr_indx]
myTimeStamp_tst = myTimeStamp[tst_indx]

NO_tr = NO[tr_indx]
NO_tst = NO[tst_indx]

## Do the plotting and saving it into an image
plot_timeSeries(myTimeStamp, NO, "NO concentration", "Time", "Concentration", 2, 0)

Nmaxsam = Ntr #
plot_timeSeries(myTimeStamp_tr[1:Nmaxsam], NO_tr[1:Nmaxsam], "NO concentration",
                "Time", "Concentration", 2, 0)
## Do the plotting and saving it into an image
Nmaxsam = 24*7*2
plot_timeSeries(myTimeStamp_tr[(Ntr -Nmaxsam):Ntr], NO_tr[(Ntr -Nmaxsam):Ntr],
                "NO concentration", "Time", "Concentration", 2, 0)
```

```
############## Transformations of the data !###############
diff_NO_tr = diff(NO_tr)
diff_NO_tr2 = diff(NO_tr, lag = 24)
log_NO_tr = log(NO_tr)

# The next two are the same
difflog_NO_tr = diff(log_NO_tr)  # -1 due to the return
normdiff_NO_tr = log( 1 + diff_NO_tr / NO_tr[-Ntr]) # -1 due to the return

## Difference twice!!
diffdiff_NO_tr = diff(diff_NO_tr, lag = 24)

## Get rid of part of the seasonal component !!
n = 24   # Differentiation
diff_NO_tr2 = NO_tr[-1] - 0.9*NO_tr[-Ntr]
Ntr_aux = Ntr - 1
diff_NO_tr2 = diff_NO_tr[(n+1):(Ntr_aux)] - 0.8*diff_NO_tr[1:(Ntr_aux-n)]

################################################
######  PLOTTING
par(mfrow=c(1,1))

plot_timeSeries(myTimeStamp_tr, log_NO_tr, "logNOtr", "Time",
                "Increase in NO concentration",2,0)

plot_timeSeries(myTimeStamp_tr[-Ntr], diff_NO_tr, "diffNOtr", "Time",
                "Increase in NO concentration",2,0)

plot_timeSeries(myTimeStamp_tr[-Ntr], difflog_NO_tr, "difflogNOtr", "Time",
                "Increase in NO concentration",2,0)
plot_timeSeries(myTimeStamp_tr[1:(Ntr-25)], diff_NO_tr2, "diff_NO_tr2", "Time",
                "Increase in NO concentration",2,0)

plot_timeSeries(myTimeStamp_tr[1:(Ntr-25)], diffdiff_NO_tr, "diffdiff_NO_tr",
                "Time", "Increase in NO concentration",2,0)
plot_timeSeries(myTimeStamp_tr[1:(Ntr-25)], diff_NO_tr2, "diff_NO_tr2",
                "Time", "Increase in NO concentration",2,0)

#######################################
###### CHECK STATIONARITY
library(tseries)
adf.test(NO_tr, alternative = "stationary",k = 24)
adf.test(diff_NO_tr, alternative = "stationary",k = 24)
adf.test(log_NO_tr, alternative = "stationary",k = 24)
# Since the p-value is less than 0.01, and being the alrernatyive hyp
# that the signal is stationary. The p-value, the probability of observing
# data more extreme that ours given the null-hypotesis
# (opposite to alternative hypotesis)
# Since the probability of observing data that is more extreme is very low,
# Then the null hypotesis is rejected, because the data obtained is very unlikely
# under the null hypotesis.

## RANGE-MEAN


par(mfrow=c(2,2), mgp=c(2,0.7,0))
LenS = 12
Ranges = c()
Means = c()

lens = c(4,12,18,24)
```

```
for( LenS in lens){
  for (i in 1:(Ntr - LenS)){
    window = NO_tr[i:(i+LenS)]
    range = max(window) - min(window)
    mean = mean(window)
    Ranges = c(Ranges, range)
    Means = c(Means, mean)
  }
  plot(Means,
       Ranges,
       lwd= lwd,                          # Line width
       main= paste("Mean-Range l = ", toString(LenS)), # Title of the graph
       xlab="Mean",            # x label
       ylab="Range")
}

################ Question 2 ######################
# We just use the ACF and PACF functions to calculate the coefficients.

## Original data !!
Nlag = 200
plot_acfpacf(NO_tr,"NOtr_ACFPACF.png",Nlag, 0)
plot_acfpacf(log_NO_tr,"NOtr_ACFPACF.png",Nlag, 0)
plot_acfpacf(diff_NO_tr,"diff_ACFPACF.png",Nlag, 0)
plot_acfpacf(difflog_NO_tr,"diff_ACFPACF.png",Nlag, 0)
plot_acfpacf(diff_NO_tr2, "diffdiff_ACFPACF.png",Nlag, 0)
plot_acfpacf(diffdiff_NO_tr, "diffdiff_ACFPACF.png",Nlag, 0)

dev.off()
################ Question 3 ######################
# Let us try to fit a model
# Model with ARIMA(2,0,2)(2,0,2) with seasonal component of order 1
# (0,1,1)(0,1,1)

fit1 <- arima(log_NO_tr, order=c(1,0,1), include.mean = FALSE, method="ML",
              seasonal = list(order = c(1,1,1),period=24))

# require(forecast)
# ARIMAfit <- auto.arima(NO_tr, approximation=FALSE,trace=FALSE)
# Summary(ARIMAfit)

plot(fit1$coef)    # Plot the coefficientes obtained.
residuals = fit1$residuals
coef = fit1$coef
varMatrix_coef = fit1$var.coef

N_coef = length(coef)

# Calculate variance of the coefficients
vars_coef = c(1:N_coef)
for (i in 1:N_coef){
  vars_coef[i] = varMatrix_coef[i,i]
}
sigma_coef = sqrt(vars_coef)
T_value_coef = coef/sigma_coef
degrees_freedom = Ntr - 1 - N_coef
p_values = 2*pt(-abs(T_value_coef),df=degrees_freedom)

## More param
sigma_res = fit1$sigma2
loglik = fit1$loglik
```

```
## Error
SS <- sum(residuals^2)/Ntr  # Sum of square errors of the residual model 1

AIC =  Ntr * log(SS) + 2 * N_coef
BIC = Ntr * log(SS) +  N_coef * log(Ntr)

#### PREDICT
prediction = predict(fit1,48)
SSval = sum((prediction$pred - NO_tst)^2)/Ntst

par(mfrow=c(1,1))

#fit1 <- arima(diff_NO_tr, order=c(25,0,1), include.mean = FALSE, method="ML")
#plot(fit1£coef)    # Plot the coefficientes obtained.
#residuals = fit1£residuals
# This is not very good because it also obtaines valus for the inbetween ACF.

## Properties
## we want that collection of plots a lot so let's make a function:

view_residuals = 0
if (view_residuals == 1){
  Nlag2 = 30
  #### Graph1: Signal + ACF + PACF
  plot_SigACFPACF(residuals, Nlag)
  tsdiag(fit1, gof.lag = Nlag2)  # You plot the residuals and their ACF and JunlgeBox
  #  Ljung-Box Q null hypothesis is that there is no autocorrelation in the errors
  #where  \chi _{1-\alpha ,h}^{2}} \chi_{1-\alpha,h}^2 is the alpha-quantile of the chi-squared
  # distribution with h degrees of freedom
  #### Graph2: Histogram and sample quantiles
  par(mfrow=c(1,2))

  # Histogram and draw gaussian
  hist(residuals,probability=T,col='blue')
  curve(dnorm(x,sd = sqrt(fit1$sigma2)), col=2, lwd=2, add = TRUE)
  # Do the samples quantities
  qqnorm(fit1$residuals)
  qqline(fit1$residuals)

  ## Graph 3: Simulate Residuals
  par(mfrow=c(2,1))
  ts.plot(residuals, ylim = c(-80,80))
  ts.plot(ts(rnorm(length(residuals))*sqrt(fit1$sigma2)),
          ylab='Simulated residuals',ylim = c(-80,80))
}

test_residuals = 0
if (test_residuals == 1){
  ### TESTS FOR THE RESIDUAL

  ### Binomial test
  # Confidence interval of the number of change of signs.
  n.residuals <- length(residuals)
  (n.residuals-1)/2
  sqrt((n.residuals-1)/4)
  (n.residuals-1)/2 + 1.96 * sqrt((n.residuals-1)/4) * c(-1,1)

  ### Binomial test
  # P-value using binomial distribution that the number of sign changes is 1/2 probable
  (N.sign.changes <- sum( residuals[-1] * residuals[-n.residuals]<0 ))
```

```
  bt = binom.test(N.sign.changes, n.residuals-1)
  p_value = bt$p.value
  int_conf = bt$conf.int

  ### Check if the sum of squared acf values follow a chi-square distribution.
  acfvals <- acf(residuals, type="correlation", plot=FALSE)$acf[2:24] # Get acf residuals
  test.stat <- sum(acfvals^2) * (length(residuals)-1)
  ## Numerical issue ... next line is better
  1 - pchisq(test.stat, length(acfvals)-length(fit1$coef))
  prob = pchisq(test.stat, length(acfvals)-length(fit1$coef), lower.tail = FALSE)

   ## Log likelihood of the data
  loglik = fit1$loglik
  # Informaiton creiterias
  res = residuals
  Nres = length(res)
  nparam = length(fit1$coef)

  VarEstimate = sum(res^2)/(Nres - nparam)
  VarEstimate2 = fit1$sigma2

  # Cumulative periodogram. The cumulative value of the frequency components of the signal.
  # If the signal is noise, then it should be a straing line since all the
  # frequency componentes should have the same value.


  par(mfrow=c(1,2))
  cpgram(NO_tr)
  cpgram(residuals)

}

compare_models = 0
if (compare_models == 1){

  # Likelihood ratio test
  prob_chisq = pchisq(-2* ( fit1$loglik - fit2$loglik ), df=1, lower.tail = FALSE)

  # F-test for lower order models
  s1 <- sum(residuals^2)  # Sum of square errors of the residual model 1
  s2 <- sum(residuals^2) # Sum of square errors of the residual model 2
  n1 <- 3    # Number of param for model 1
  n2 <- 4    # Number of param for model 2

  # Calculate the F-value of the models
  F_value = pf( (s1-s2)/(n2-n1) / (s2/(length(residuals)-n2)),
                df1 = n2 - n1, df2 = (length(residuals)-n2), lower.tail = FALSE)
}

#### PREDICT

predic_shit = 1
if (predic_shit == 1){
  prediction = predict(fit1,48)
  VarEstimate = sum((prediction$pred - NO_tst)^2)

  Nbefore = 80

  lwd = 2

  plot(myTimeStamp[(Ntr -Nbefore):(Ntr+Ntst)],
```

```
       NO[(Ntr -Nbefore):(Ntr+Ntst)],
       type = "l",                    # Draw as a line
       lwd= lwd,                        # Line width
       main= "NO concentration", # Title of the graph
       xlab="Time",             # x label
       ylab="Concentration",
       ylim = c(-200,300))

  lines(myTimeStamp_tst,  prediction$pred, col="blue")
  lines(myTimeStamp_tst,  prediction$pred + 1 * prediction$se, col="blue", lty=2)
  lines(myTimeStamp_tst,  prediction$pred - 1 * prediction$se, col="blue", lty=2)

  #legend(myTimeStamp[Ntr],200, legend = c("Actual values","Predicted values"))

  prse = c(prediction$se[1], prediction$se[24], prediction$se[48])
  pr = c(prediction$pred[1], prediction$pred[24], prediction$pred[48])

}

########################################### LOGGING ###################################
###########################################################################################
###########################################################################################
###########################################################################################
###########################################################################################
No = 1000
x = (1:No)/(No/10)
lnx = log(x)
dlnx = diff(lnx)*(No/10)
#dlnx = 1/x[-No]

par(mfrow=c(1,2))

plot(x,
     lnx,
     type = "l",                    # Draw as a line
     lwd= lwd,                        # Line width
     main= "ln(x)", # Title of the graph
     xlab="x",             # x label
     ylab="Logarithmic transformation",
     xlim = c(0,4))

plot(x[-No],
     dlnx,
     type = "l",                    # Draw as a line
     lwd= lwd,                        # Line width
     main= "derivative of ln(x)", # Title of the graph
     xlab="x",             # x label
     ylab="dln(x)",
     xlim = c(0,2),
     ylim = c(0,5))



##### IF WE ARE USING LOG OF THE INITIAL ISGNAL #############3

NO_tr_estimation = exp(-(fit1$residuals - log_NO_tr))
residuals = (NO_tr - NO_tr_estimation)
SS <- sum(residuals^2)/Ntr  # Sum of square errors of the residual model 1

plot_SigACFPACF(residuals, Nlag)
tsdiag(fit1, gof.lag = Nlag2)  # You plot the residuals and their ACF and JunlgeBox
```

```
par(mfrow=c(1,2))

# Histogram and draw gaussian
hist(residuals,probability=T,col='blue')
curve(dnorm(x,sd = sqrt(fit1$sigma2)), col=2, lwd=2, add = TRUE)
# Do the samples quantities
qqnorm(fit1$residuals)
qqline(fit1$residuals)

par(mfrow=c(2,1))
plot_timeSeries(myTimeStamp_tr, log_NO_tr, "logNOtr", "Time",
                "Increase in NO concentration",2,0)

plot_timeSeries(myTimeStamp_tr, NO_tr_estimation, "logNOtr", "Time",
                "Increase in NO concentration",2,0)



############### LOGSHIT ONLY !!!!! ######################
prediction = predict(fit1,48)
NO_tst_pred = exp(prediction$pred)
residuals_tst = (NO_tst - NO_tst_pred)

VarEstimate = sum((NO_tst - NO_tst_pred)^2)/(Ntst)


Nbefore = 80

lwd = 2
par(mfrow=c(1,1))
plot(myTimeStamp[(Ntr -Nbefore):(Ntr+Ntst)],
     NO[(Ntr -Nbefore):(Ntr+Ntst)],
     type = "l",                      # Draw as a line
     lwd= lwd,                          # Line width
     main= "NO concentration", # Title of the graph
     xlab="Time",              # x label
     ylab="Concentration",
     ylim = c(0,300))

lines(myTimeStamp_tst,  NO_tst_pred, col="blue")
lines(myTimeStamp_tst,  exp((prediction$pred + 1 * prediction$se)), col="blue", lty=2)
lines(myTimeStamp_tst,  exp((prediction$pred - 1 * prediction$se)), col="blue", lty=2)

#legend(myTimeStamp[Ntr],200, legend = c("Actual values","Predicted values"))

prse = c(prediction$se[1], prediction$se[24], prediction$se[48])
pr = c(prediction$pred[1], prediction$pred[24], prediction$pred[48])
```