Time series modeling, prediction and methodology

David Andersson

January 14, 2016

Abstract

This paper will be about creating models for time series, estimating parameters and making predictions. I will not venture to deep into the mathematical background, but rather the methodology and potential issues and ambiguity encountered when creating models. The data used to create the models will be heat power consumption, using air and water temperature as inputs.

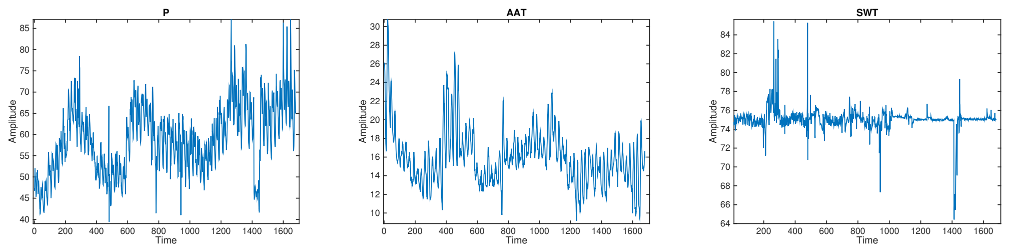

Figure 1: Output and inputs

1 Introduction

The time series output in question is the power consumption of heating (P) in the city of Esbjerg in Denmark, and the input is supply water temperature (SWT) and ambient air temperature (AAT) for the same city. The measurement is from the years 1989 and 1990, but I will focus on the data from 89. The setting used is a model with air temperature as input (M1), air and water as input (M2) and also a recursive estimation of the first model (M3). Figure 1 shows plots of the data when choosing the first 1680 data points. This will account for about 10 weeks of data, making it an even 70 24 hour cycles. The calculations will be preformed using Matlab and the material is based on the book ”An introduction to Time Series Modeling” [1] from Andreas Jakobsson. Andreas has also provided most of the functions not native to Matlab.

The tools used are the Auto correlation Function plot (ACF) with the option of trimming away outliers, the Partial Auto correlation Function plot (PACF) and the normal probability plot with added support for T-distribution. The ACF and PACF plots does not include the lag 0, since it is irrelevant to the task. An example of the plots are give in figure 3 Where ACF is subplot 1, PACF is subplot 2 and normplot is subplot 3. The T-distributed addition to normplot is given in subplot 3 in figure 5.

An addition to this one also have the whiteness test. The test is actually a collection of tests, namely: Ljung-Box-Peirce, MacLeod-Li, Monti and sign test. An example is shown in table 1. The tests and theory is based on the notion that residuals are Gaussian distributed. This will not always hold, which will be apparent later, but I have still used the Gaussian 95% confidence interval of $1.96\sqrt{N}$ since i only use it as a indication.

The method part of the paper will be sectioned by the different steps i have used when creating the models. I will incorporate all variations of the models under each section, and later in the ”Result” section give them a more autonomous explanation.

The paper expects a basic knowledge of time series and modeling.

2 Method

The basic methodology I'm using is the following sequence:

- Outliers
- Transform
- Trends
- Periodicity
- Model order selection
- Parameter estimation
- Prediction ...

This process is iterative, meaning that one will probably have to revisit the previous steps when the current model will undoubtedly fail.

2.1 Outliers

Measuring data is rarely precise. Often there will be some sensor malfunction or the process which one is measuring behaves in a way that's not significant for the general behavior. In these cases one has outliers in the data. On the first glance it seems as if P has some major outliers for example $x\{479\ 942\}$ in figure 1. But closer inspection will show that the outliers can be explained by the SWT data which will be discussed in "Model 2 - Air and water as input".

Obviously in M2 where I have access to SWT these deviations will be included in the model. But in M1 and M3 where only AAT is used, there is no corresponding contribution. This raises the question if one should consider the them as outliers and just simply remove theme. Since they seem non-periodic and account for a significant fluctuation in amplitude I chose to do this in M1.

One has a variety of choices dealing with outliers such as interpolation, linear recreation or using the Kalman filter. Since I had few (if any) outliers I chose to use one of Matlabs native function.

2.2 Transform

One assumption one makes in these settings is that the time series are reasonably stationary. However in real data, this does not always hold. To make the data more stationary in the variance, and hopefully more Gaussian, one uses transforms. To decide which transform to use on y_t one have to find a λ which maximizes the log-likelihood function:

$$L(\lambda) = -\frac{N}{2} log\left\{\hat{\sigma}_y(\lambda)\right\} + (\lambda - 1)\sum_{N}^{t=1} log(y_t) \tag{1}$$

Where $\hat{\sigma}_y(\lambda)$ denoting the estimated standard deviation of the transformed data, using the parameter λ.

This is preformed with the given Matlab function bcNormalPlot. Which is the Box-Cox normality plot showing $L(\lambda)$ as a function of λ. Visually or by searching one can find the max in this plot. In the book a few standard transforms with corresponding λ are given [1, p. 124], which I have used when modeling.

Using this on the section of the data P which is to be used for modeling I got a fairly consistent indication that the transform $\sqrt{y_t}$ should be used. When doing the same test on for example AAT I got a indication that $y_t^{-1/2}$ or y_t^{-1} should be used. Since P is the data which is most central to the model, I chose to go with the square root. Trying it out it did not give a very good result, so instead I used the logarithm. It's supposed to be a very common and stable transform. when doing the test on smaller batches of data, a lot of it also indicated that i should use the logarithm. Choosing a transform means that all the data input and output should be subjected as well. This gave me some problems, since AAT sometimes gave back complex numbers for the 89 data. In these cases I removed the transform which also worked quite well. One example of the bcNormalPlot is given in figure 2. The actual result was 0.3838 which is closes to 0.5 in the transform tables, which means one should use squared root.

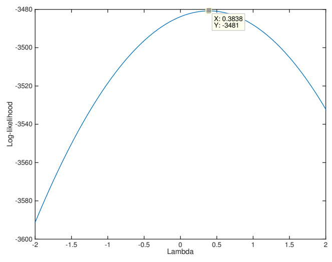

Figure 2: Box-Cox normality plot.

2.3 Trends

Some data have trends. To deal with this one can use a differentiation filter:

$$A(Z)(1 - z^{-1})^d y_t = C(z)e_t \tag{2}$$

Where d is the number of differentiations. In this case the power consumption, one could imagine, would have piece wise linear or exponential trend, using more power in the winter than in the summer e.t.c. But I found that I got a better result not removing the trend, but rather focusing on the later topic of seasonality. I.e, when first removing the season, I found that no significant trend was present. Also, the broad strokes of the trends was also accounted for by the negative of the input AAT.

2.4 Periodicity

In the 89 data one could see a strong seasonality in P and AAT, and in 90 also in SWT. The period was a cycle of 24, which seems reasonable since one uses less power and the air is cooler at night. I assume that the cold also influences the water temperature, or maybe the water suppliers lowers the temperature when people do not use as much warm water. An example of the period in a ACF plot can be seen in figure 3.

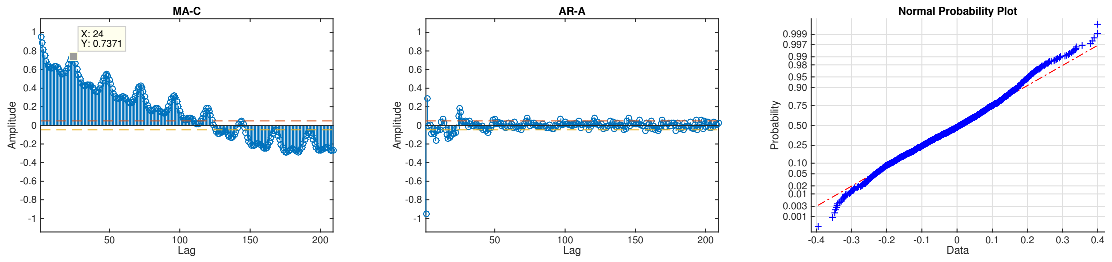

Figure 3: ACF, PACF and normplot for P, using the log transform and subtracting the mean.

To account for this the step of removing seasonality is required. Similar to removing a trend one uses:

$$A(Z)(1 - z^{-s})y_t = C(z)e_t \tag{3}$$

Where s is the season.
Removing the trend I get a result shown in figure 4.

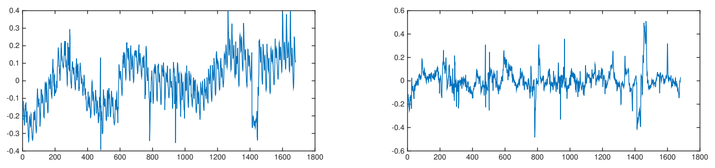

Figure 4: Transformed and deseasoned versions of P.

2.5 Model order selection

Once The season and the trends are removed and whats left should be data that is without apparent structure. To proceed one need to make use of another strategy.

Using an Autoregressive–moving-average model with exogenous inputs model (ARMAX) one want to find polynomials that describe the data in the form:

$$A(z)y_t = B(z)x_{t-d} + C(z)e_t \tag{4}$$

d being the delay between input and output. And where A,B,C are polynomials in accordance with:

$$P(z) = p_0 + p_1 z^{-1} + ... + p_n z^{-n} \tag{5}$$

n being the model order.
With the tools one should decide the A polynomial (AR) using the PACF, and the C polynomial (MA) using ACF for P. Also, one should choose the delay d for B (X) polynomial for AAT and SWT using the cross correlation between P and AAT and P and SWT.

So, when preforming the above steps and filtered the output signal, what is left is a residual which is to be decided by the ACF and PACF. Without going to much in to the math the rules are quite simple.

If one has a AR(p) the ACF should look like a dampened exponential and/or sine function, and the PACF should be zero for values larger than p.

If one has a MA(q) the ACF should be zero for values larger than q, and the PACF should look like a dampened exponential and/or sine function.

If one has and ARMA(p,q) both the ACF and PACF look like a dampened exponential and/or sine function for values larger than $|p - q|$.

As mentioned above I have made the assumption that the data is Gaussian, and thus used the 95% confidence interval of $1.96\sqrt{N}$. I have also employed the rule of thumb to look at $N/4$ of the data, since looking at data further is not relevant. These values are inputted into the plotting function. As I also pointed out in the introduction there is an ACF function that trims away outliers when plotting. To get an even better overview I used this in parallel to the normal ACF.

The resulting residual from the de-seasoning can be seen in figure 5. Where one can see a dampened exponential in the ACF and values being zero after q = 2 except for a season of 25 which also needs to be accounted for in the model. One could incorporate this season in the de-seasoning step, but I found that it was easier and gave better result to take it in to account in the model order selection step. When doing a de-seasoning, in this case on the A polynomial, one has to do a similar operation on the C polynomial. The method i used was to make a C filter which included $[24-1,\ 24,\ 24+1]$ which worked very well for this application, even thou the 23 polynomial was a bit redundant at times.

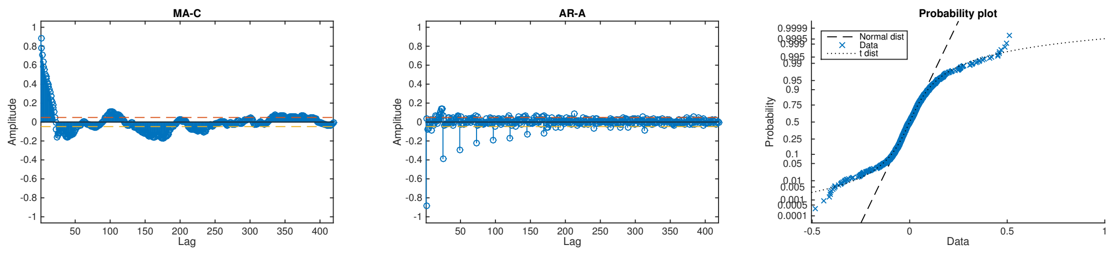

Figure 5: ACF PACF and T-normplot for removed season

Using a general model such as Box-Jenkins, choosing the B-polynomial follows some pretty straight forward rules. However, using the ARMAX there is a lot of guesswork. There is, as mentioned, the question of delay where one could use the cross correlation function. Using the cross correlation on SWT and P and AAT and P results in the graphs in figure 6. As one can see, there is a clear indication that AAT has a delay of 11, and SWT has a delay of 0. Of course this has to be tested out, seeing as the plots might not be entirely accurate.

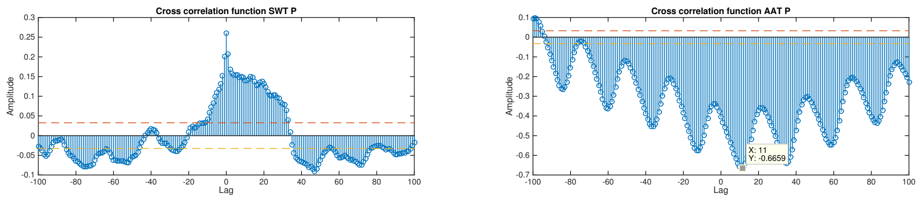

Figure 6: The cross correlation for SWT and P and AAT and P respectively

One thing I'm still not sure bout is the fact that a contribution at lag 7-8 showed up when making a k=1 step prediction. It was quite weak, but still, the models often preform better taking it into account. One idea is that there is a seasonality that is not very strong, and by that does not show up in the ACF and PACF, or maybe it's masked by the 24 seasonality. One guess is that it's due to the fact that people work, sleep and have spare time, in 8 hour cycles. One thing to note is that when using less data, for example 5 weeks, the contribution was less persistent and did not need to be in the model.

2.6 Parameter estimation

When estimating the parameters there are a verity of estimators to choose from. One common estimator when using ARMAX is PEM (Prediction Error Method).

The idea is to minimize the difference between the measurement and the one step prediction of the measurement:

$$\epsilon_{t+1|t}(\Theta) = y_{t+1} - \hat{y}_{t+1|t}(\Theta) \tag{6}$$

Where $\epsilon_{t+1|t}(\Theta)$ is the prediction error, $\hat{y}_{t+1|t}(\Theta)$ is the one step prediction and (Θ) is the parameter vector of the model for y_t. More details are given at [1, p. 170].

Using PEM in Matlab gives an estimate of the parameters, and also a variance which can help one determine if the parameters are significant to the model. The function to find the variance is "present", using the model, which is a idpoly object, as an input. This will be discussed further in the Result section.

In addition to this the recursive model also needs a recursive estimator. For this task I have used the recursive PEM, either using a setting employing the Kalman filter, or a forgetting factor $\lambda \subset \{0..1\}$ Where 0 uses no data and 1 uses all data. This is for tracking dynamics in the system.

2.7 Prediction

Prediction is about making a educated guess of future values. Given data that is dependent on earlier data, good model and a good estimation of parameters, one will get a result and also a variance of how good the prediction was.

For this case I will use a optimal linear predictor. This predictor will be such that it is formed as a linear combination of earlier observations:

$$\hat{y}_{t+k|t} = \sum_{l=0}^{n} w_l y_{t-l} \tag{7}$$

Which holds for a normal distributed process, which one assume the process is from theorem 6.2 [1, p. 170].

For the case of ARMAX the optimal predictor of y_{t+k} is:

$$\hat{y}_{t+k|t}(\Theta) = E\left\{y_{t+k}|\Theta\right\} = \hat{F}(z)E\left\{x_{t+k}|\Theta\right\} + \frac{\hat{G}(z)}{C(z)}x_t + \frac{G(z)}{C(z)}y_t \tag{8}$$

Where G, $\hat{G}$ and $\hat{F}$ comes form:

$$\begin{gathered}[F,G] = deconv(conv([1 zeros(1,k-1)],C),A) \\ BF = conv(B,Fk) \\ [\hat{F},\hat{G}] = deconv(conv([1 zeros(1,k-1)],BF),C)\end{gathered} \tag{9}$$

More information is found at [1, p. 235-240].

When analyzing the prediction for k steps in to the future there are a few things to consider regarding the residual. First of all for $k = 1$, the residual should be white. For $k > 1$ the residual should resemble a MA(k-1). This conclusion is drawn from the fact that the prediction error is:

$$\epsilon_{t+k|t} = y_{t+k} - \hat{y}_{t+k|t} = F(x)e_{t+k} \tag{10}$$

3 Result

3.1 Model 1 - Air as input

One thing I wanted to try out was to eliminate the outliers mentioned before. In the end I eliminated the points $[290, 291, 292, 293, 478, 479, 480, 481, 482]$, and also experimented with other further away in time. My first try was to restore the values with the Kalman filter. So i made a model, and used the model to make a one step prediction for the eliminated values. This was not very successful since I had estimated my parameters on data with the outliers replaced by the previous sample. In stead I turned to the Matlab command "misdata" which seemed to give a reasonable result. "misdata" uses a default model and does an estimation using that. The result when removing the outliers was very good, i got white residual with ease, passing all test. Also the prediction improved, going from a variance of 0.0011 to 7.4195e-04.

The model I ended up with without removing the outliers is shown below in equation (11). One should note that the B-polynomial is not very significant, but it still gave the best predictions. I tried to adjust the filter size and delay but I could not find a better model.

$$A(z) = 1 - 1.444(+/-0.1251)z^{-1} + 0.6034(+/-0.125)z^{-2} - 0.1159(+/-0.02901)z^{-3}$$
$$+0.05928(+/-0.02702)z^{-7} - 0.08766(+/-0.02642)z^{-8}$$

$$B(z) = -0.005601(+/-0.002655)z^{-11} + 0.006414(+/-0.00265)z^{-12} \quad (11)$$

$$C(z) = 1 - 0.4578(+/-0.125)z^{-1} + 0.001078(+/-0.009966)z^{-7} + 0.008371(+/-0.009832)z^{-8}$$
$$-0.9335(+/-0.009496)z^{-24} + 0.4585(+/-0.1151)z^{-}25$$

The predicting k steps for $k = [1, 4, 24]$ is shown in figure 7.

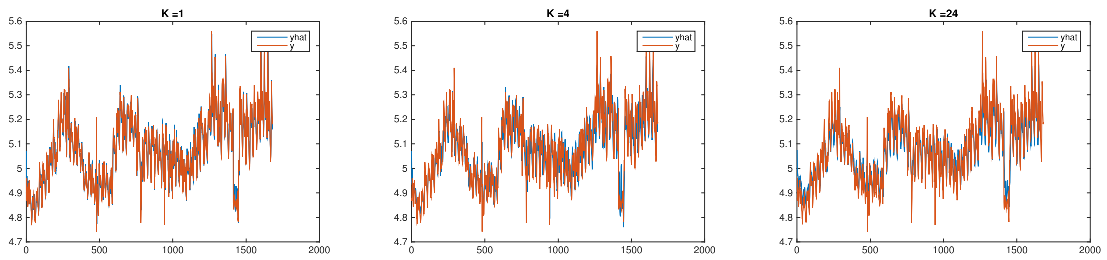

Figure 7: P for k = [1 4 24]

To test out the model on other data I shifted the data by steps of 100. i.e $(1680+100*n : 1680+100*n)$. Without re-estimating the model i got white k=1 residual for n up to 3. When re-estimating i got white residual up to 7. After that the data behaves very odd, and my model was not sufficient. For the 90 data I had some luck with $n = 3-8$. But i had to change the delay of the input. I also have to note that for some of datasets I had to remove the log transform since it gave complex values. I think all that data would work reasonably well if outliers were removed and the transforms were recalculated. One thing i noticed was that adding or removing an extra AR and MA parameter really helped with making the model more general, with this strategy I got the model to work with more data when re-estimating.

Figure 8 shows the residual for k = 1 on the same data I used for estimation. The residual is run through the whiteness test discussed earlier and gets full score except from the McLeod-Li test which is known to be extremely sensitive to non Gaussianity. The result can be seen in table 1. About 4.55% of the residuals are outside of the 95% confidence interval.

Table 1: Whiteness Test

Ljung-Box-Pierce	1	white if 30.38 <36.42
McLeod-Li	0	white if 100.08 <36.42
Monti	1	white if 28.71 <36.42
Sign change	0	white if 0.46 in [0.48,0.54]

As discussed earlier I made a whiteness test when removing the outliers The result is shown in table 2. I could also remove a lot of the parameters in A and C and still got a white residual.

Table 2: Whiteness Test

Ljung-Box-Pierce	1	white if 33.42 <36.42
McLeod-Li	1	white if 10.37 <36.42
Monti	1	white if 33.69 <36.42
Sign change	1	white if 0.49 in [0.48,0.54]

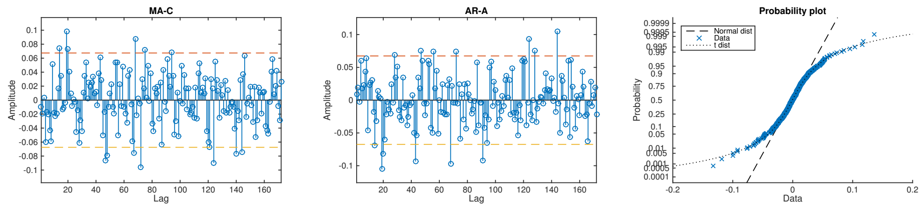

Figure 8: ACF, PACF and T-normplot for residual k = 1.

Figure 9 shows the residual for k = 4 on the same data I used for estimation. One can see that there is indication that this is a MA(k-1) since the ACF has clear contributions on the first three values. The following values are however not zero.

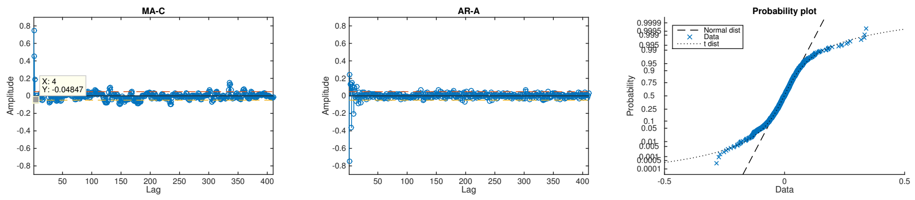

Figure 9: ACF, PACF and T-normplot for residual k = 4.

Figure 10 shows the residual for k = 24 on the same data I used for estimation. No real conclusions can be drawn from the residual except that it barely resembles a MA(k-1).

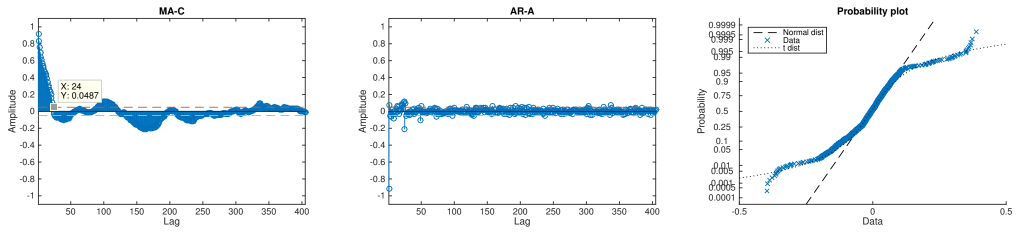

Figure 10: ACF, PACF and T-normplot for residual k = 24.

I also made a forecast, using all the data I had when modeling and compared it with an additional 72 datapoints form P. As one can see it is a quite close fit with a variance of 0.0148. The result can be seen in figure 11.

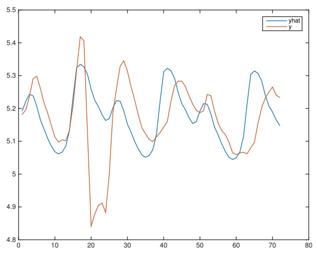

Figure 11: 72 step forecast using model 1.

I also made a forecast using the data and model where outliers had been removed. The result was actually better with a variance of 0.0143.

3.2 Model 2 - Air and water as input

The second model was formed using waters as a second input. On first glance this seemed like a really good idea. Looking at figure 12

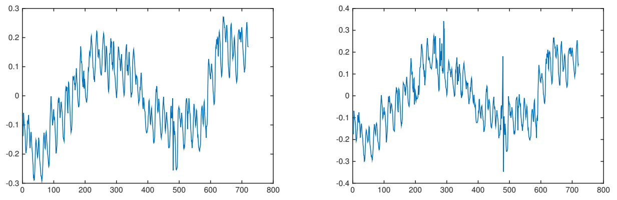

Figure 12: P with and without the removal of SWT.

which is showing $P - SWT * 1.5$ one can clearly see that the spiky contribution in P is explained by SWT. However, making a model for this proved difficult. I first tried using the same model as before, just adding the extra input. I found that doing so diminished the validity of the AAT B-polynomial (B2) and gave very poor result regardless of what polynomial I used for SWT (B1). The question was not so much if I could get a white residual on estimation, but rather the prediction. The trick was to use quite a lot of parameters for B1. When I finally found a polynomial that worked, it actually preformed very well, the estimation was better than that of the first model.

The polynomial I ended up with was:

$$A(z) = 1 - 1.839(+/-0.04808)z^{-1} + 0.9578(+/-0.0547)z^{-2} - 0.1172(+/-0.02792)z^{-3}$$

$$B1(z) = 1.179(+/-0.07186) - 2.009(+/-0.1502)z^{-1} + 0.9167(+/-0.1475)z^{-2} - 0.09287(+/-0.08274)z^{-3}$$

$$B2(z) = 0.0003228(+/-0.0002267)z^{-11} \quad (12)$$

$$\begin{aligned} C(z) = 1 - 0.8869(+/-0.04295)z^{-1} - 0.05551(+/-0.02396)z^{-7} + 0.03876(+/-0.02168)z^{-8} \\ -0.6173(+/-0.02103)z^{-24} + 0.521(+/-0.03407)z^{-25} \end{aligned}$$

As one can see the B2 polynomial is very insignificant, but it was still needed to make a good prediction.

I also tried to use the model on other data with the same setup is in M1, it turned out that it was much harder. For n = 1-4 the result was quite good, but after that I had to make adjustments that clearly made it into a different model.

The plot from the The k=1 step prediction can be seen in figure 13.

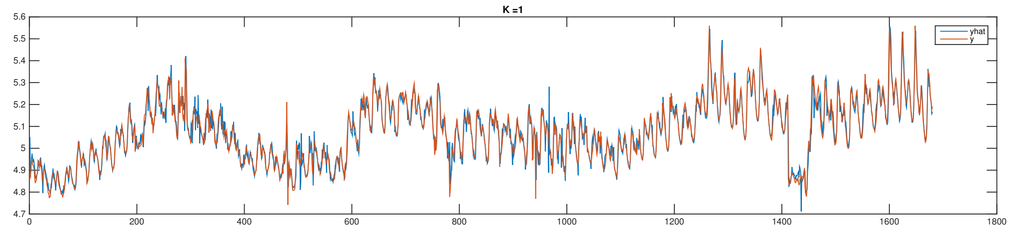

Figure 13: k = 1 step prediction.

As one can see there are some unexplainable structure in $\hat{y}$ which seem to be due to the SWT input. My guess is that there are outliers in the already spike prone data which is masked. If one could eliminate these it should be possible to get a better prediction.

The variance I got from the residual was 0.0011 which is about the same as when I did not remove the outliers from the previous model M1.

Figure 14 shows the residual for k = 1 on the same data I used for estimation. The result of the whiteness test can be seen in table 3. About 4.78% of the residuals are outside of the 95% confidence interval.

Table 3: Whiteness Test

Ljung-Box-Pierce	1	white if 27.15 <36.42
McLeod-Li	0	white if 441.58 <36.42
Monti	1	white if 25.26 <36.42
Sign change	1	white if 0.48 in [0.46,0.54]

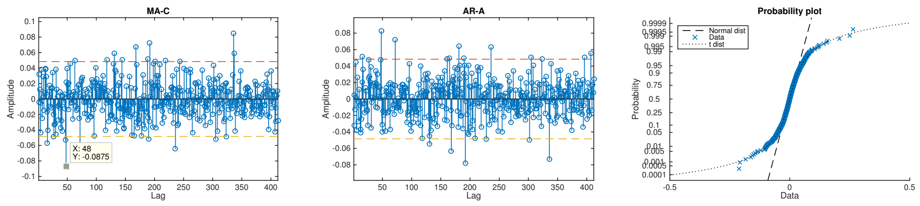

Figure 14: ACF, PACF and T-normplot for residual k = 1.

Figure 15 shows the residual for k = 4 on the same data I used for estimation. One can see that there is indication that this is a MA(k-1) since the ACF has clear contributions on the first three values. The following values are however not zero.

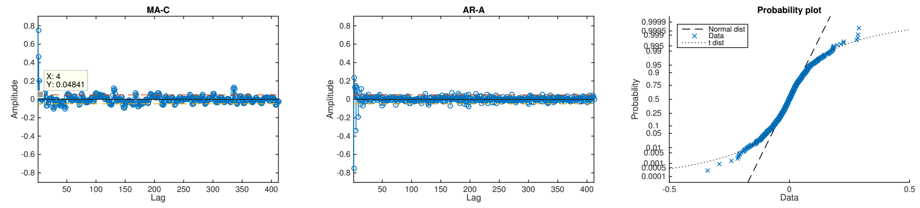

Figure 15: ACF, PACF and T-normplot for residual k = 4.

Figure 16 shows the residual for k = 24 on the same data I used for estimation. No real conclusions can be drawn from the residual except that it does not resemble a MA(k-1).

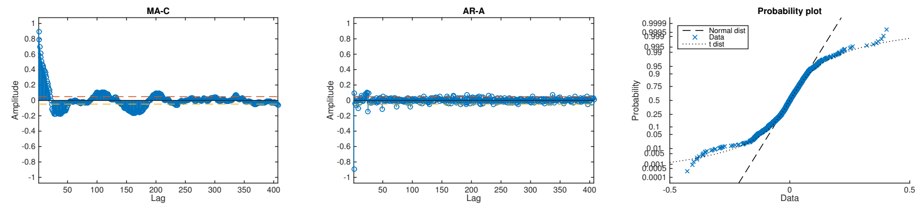

Figure 16: ACF, PACF and T-normplot for residual k = 24.

Figure 17 shows the 72 point forecast, using all the data I had when modeling. It's actually not as good as M1 with a variance of 0.0148.

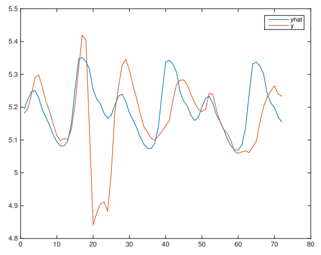

Figure 17: 72 step forecast using model 2.

3.3 Model 3 - Air as input, recursive estimation

As mentioned before I also tried a recursive estimation of the parameter using recursive PEM. In M1 I discussed the capability of my model to predict on other data. The conclusion was that a limited dataset worked fine using the same parameters, but in general a re-estimation was required. This means that there is a time dependency on the parameters, an thus they might need to be be estimated recursively. In figure 18 one can see the k=1 step prediction for the first 1680 datapoints.

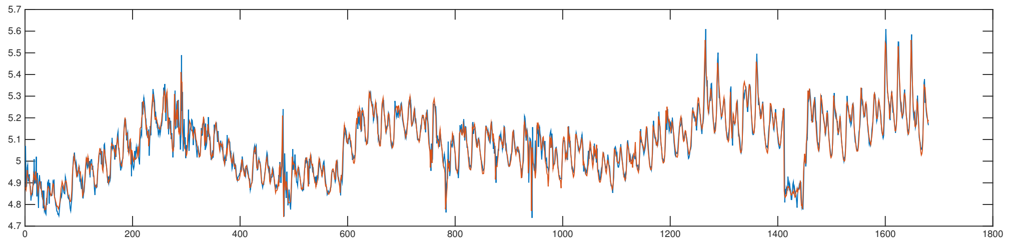

Figure 18: k = 1 step prediction result.

As one can see in figure 19 there is a difference between the initial values $t = 0$ which is the estimated parameters using PEM, and the last values, showing that the parameters do change over time.

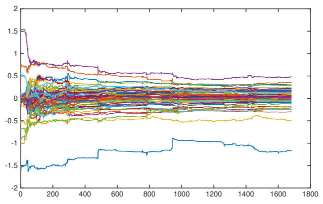

Figure 19: Recursive estimation of parameters 0:1680.

I did not manage to get a completely white residual shown in figure 20. I also tried using a forgetting factor of 0.9-0.99 which did not give me a better result.

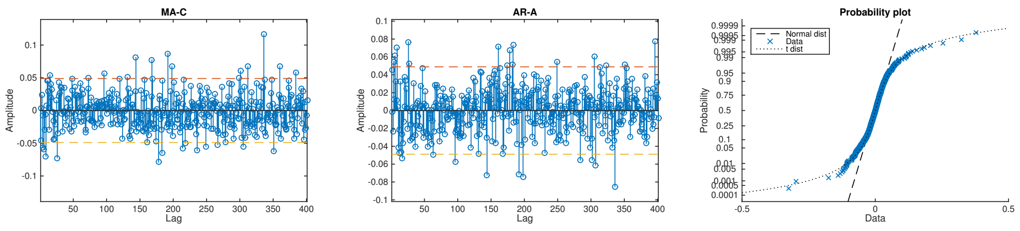

Figure 20: ACF, PACF and T-normplot for residual k = 1.

So, clearly using recursive estimation for the original data was not a success. In stead a proceeded to use a smaller dataset of 720 datapoints. This actually proved to be very good.

When using the Kalman filter there is the question of the matrix Re which contains the covariance matrix of the parameter changes per time step, which should be set. My best result was setting the values that was non-zero to 0.005 and the zero values to zero. This will result in a relatively slow change which is also consistent with the previous findings.

When doing a k=1 step prediction on the same data as I used in M1 I got even better result using recursion. Figure 21 shows the residual for k = 1 on the same data I used for estimation and the corresponding whiteness test is shown in table 4. As one can see there is a very good margin in the Ljung-Box-Peirce and Monti test. The variance I got was $6.4613e-04$ which is a 8% improvement from M1.

Table 4: Whiteness Test

Ljung-Box-Pierce	1	white if 8.01 <36.42
McLeod-Li	0	white if 218.87 <36.42
Monti	1	white if 8.03 <36.42
Sign change	1	white if 0.47 in [0.46,0.54]

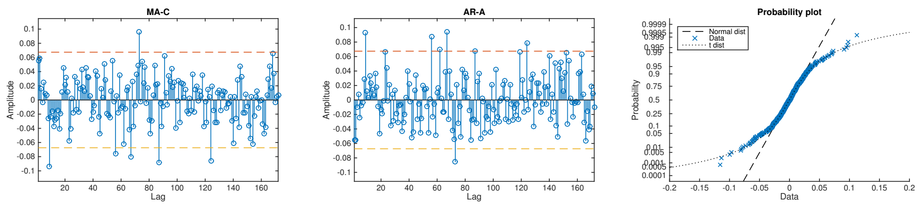

Figure 21: ACF, PACF and T-normplot for residual k = 1.

Figure 22 shows the residual for k = 4 on the same data I used for estimation. Compared to figure 9 one could see an improvements in the residual for values larger than p-1 which should be zero.

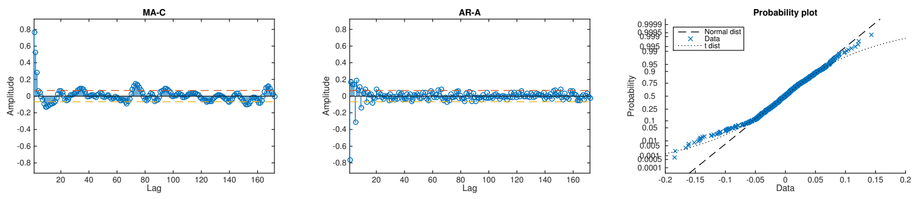

Figure 22: ACF, PACF and T-normplot for residual k = 4.

Figure 23 shows the residual for k = 24 on the same data I used for estimation. As in M2, for k = 4 the residual seems closer to zero for values larger than p-1.

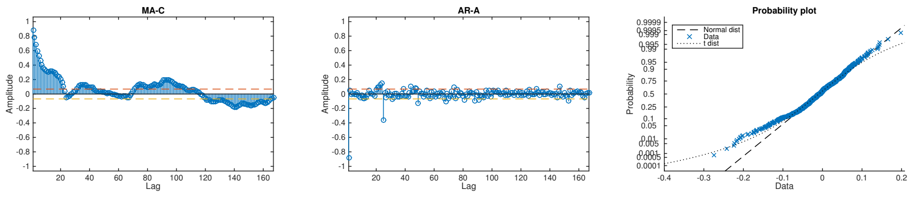

Figure 23: ACF, PACF and T-normplot for residual k = 24.

To support this I also included a plot of the parameters development through time in figure 24. There does not seem to be a big change, but apparently enough to be beneficial to do a recursive estimate.

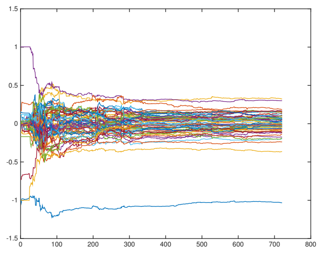

Figure 24: Recursion of parameters 0:720.

I also made a 48 datapoint forecast which gave the a variance of 0.0034. Using the same settings for M1 and M2 gave me a variance of 0.0019 and 0.0020 and a visually much better fit. The result is shown in figure 25. One thing to note is that the model also did very poorly when I tried to do a prediction using data not used in the modeling. This might not be so strange seeing as the estimation is made to fit the current sample, rather than the data as a whole.

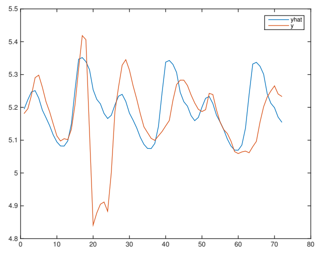

Figure 25: 48 step forecast using model 3

4 Discussion and Conclusions

So, now we have three models to solve one problem. The problem as is see it is to make good predictions, and maybe even more important, forecasts.

My choice of model is M1. All in all it was the most general one, it used the least amount of parameters and gave the rivaling forecast. Even without removing the outliers one could get a good result, which for me indicated that the potential addition from SWT could be accounted for using additional parameters.

M2 should intuitively be better, having more data which actually explains a lot of the non-stationarity in the data. I am unsure if it's because it needed more parameters, or that the data has outliers that corrupted the data, but I could not get a better result than M1. It might be the case that SWT contains behavior that is not relevant to P. When looking at how it preformed on other data, I get the feeling that my initial model was compensating for something, making it less general and not very useful.

M3 did give a very low variance and the whiteness test had a margin unrivaled by the other models. But, the forecast was very poor. As I mentioned in the ”Result” section it’s probably a result of recursion being very data specific, making it less general. I guess for other applications it’s a good method, but I can not see it being beneficial here.

I would say that the main issue I was faced with making these models were the lack of insight about the data. I had no real understanding of how the data should look, what was outliers and whats not, or if the sensors used for measuring was should be trusted. If I was to redo this project I would probably spend more time on these questions.

References

[1] Andreas Jakobsson. *An Introduction to Time Series Modeling.* Studentlitteratur, 2015.