Exercise on statistical models for forecasting in energy systems

Summer school 2018 DTU - CITIES and NTNU - ZEN

Time series analysis - with a focus on modelling and forecasting in energy systems

CITIES

Centre for IT Intelligent Energy Systems

Intro

In this exercise you will work with setting up models, which are useful for many applications of forecasting in energy systems. The models are setup in a two-stage approach, first transformations of the inputs are applied and then an estimation method is applied.

The exercise deals with first load forecasting, then solar and wind forecasting. It starts by introducing a simple linear low-pass filter for input transformation (and base splines) and then this is used together with linear regression for load forecasting. Then the recursive least squares (RLS) estimation method is introduced, and applied for load forecasting.

For solar and wind forecasting both base spline and kernel methods are used.

In the exercise numerical weather predictions (NWPs) are used as model input. The first step is to understand how they are set up. They are set up as matrices. It holds for each time t *the latest available forecasts along the row* for the variable

$$\boldsymbol{u}_t^{\mathrm{nm}} = \begin{array}{c} \begin{array}{cccc} \mathbf{k1} & \mathbf{k2} & \dots & \mathbf{k}n_k \end{array} \\ \begin{pmatrix} u^{\mathrm{nm}}_{2|1} & u^{\mathrm{nm}}_{3|1} & \dots & u^{\mathrm{nm}}_{1+n_k|1} \\ u^{\mathrm{nm}}_{3|2} & u^{\mathrm{nm}}_{4|2} & \dots & u^{\mathrm{nm}}_{2+n_k|2} \\ \vdots & \vdots & & \vdots \\ u^{\mathrm{nm}}_{t|t-1} & u^{\mathrm{nm}}_{t+1|t-1} & \dots & u^{\mathrm{nm}}_{t-1+n_k|t-1} \\ u^{\mathrm{nm}}_{t+1|t} & u^{\mathrm{nm}}_{t+2|t} & \dots & u^{\mathrm{nm}}_{t+n_k|t} \end{pmatrix} \end{array} \begin{array}{c} \leftarrow \textbf{horizon} \\ \downarrow \textbf{time} \\ 1 \\ 2 \\ \vdots \\ t-1 \\ t \end{array} \tag{1}$$

where

- t is the counter of time for equidistant time points. In this notation normalized, such that the sampling time is $t_1 - t_0 = 1$ (the time stamps are then just kept in another vector)
- t_0 is the first available time point
- n_k is the length of the forecasting horizon
- The column names are indicated above the matrix, they are simply a **k** concatenated with the value of k.

Hence, with a prediction horizon $n_k = 24$, having data from time $t_0 = 1$, then at

time $t = 100$ we would have the following matrix

$$\boldsymbol{u}^{\text{nm}}_{100} = \begin{pmatrix} \textbf{k1} & \textbf{k2} & \dots & \textbf{k24} & \leftarrow \textbf{horizon} \\ & & & & \downarrow \textbf{time} \\ u^{\text{nm}}_{2|1} & u^{\text{nm}}_{3|1} & \dots & u^{\text{nm}}_{25|1} & 1 \\ u^{\text{nm}}_{3|2} & u^{\text{nm}}_{4|2} & \dots & u^{\text{nm}}_{26|2} & 2 \\ \vdots & \vdots & & \vdots & \vdots \\ u^{\text{nm}}_{100|99} & u^{\text{nm}}_{101|99} & \dots & u^{\text{nm}}_{123|99} & 99 \\ u^{\text{nm}}_{101|100} & u^{\text{nm}}_{102|100} & \dots & u^{\text{nm}}_{124|100} & 100 \end{pmatrix} \tag{2}$$

The model output is simply a vector, so for the above example it is

$$y_t = [y_0\ y_1\ \dots\ y_{100}]. \tag{3}$$

The time t can then just be thought of as an index (in the R code we use just i and the time stamps are kept in a vector).

Question 1: Data setup and linear regression

Go and open `q1_data_setup_and_lm.R`. First the data is loaded and in steps you try to see how it is setup.

The main point is, that in order to fit a model for the k'th horizon you will need to lag the forecast input, e.g. for $k = 1$, and setup the data like

$$\boldsymbol{X}_1 = \begin{pmatrix} y_1 & \text{NA} \\ y_2 & u^{\text{nm}}_{2|1} \\ y_3 & u^{\text{nm}}_{3|2} \\ \vdots & \vdots \\ y_{t-1} & u^{\text{nm}}_{t-1|t-2} \\ y_t & u^{\text{nm}}_{t|t-1} \end{pmatrix} \tag{4}$$

and for $k = 10$

$$\boldsymbol{X}_{10} = \begin{pmatrix} y_1 & \text{NA} \\ \vdots & \vdots \\ y_{10} & \text{NA} \\ y_{11} & u^{\text{nm}}_{11|1} \\ y_{12} & u^{\text{nm}}_{12|2} \\ \vdots & \vdots \\ y_{t-1} & u^{\text{nm}}_{t-1|t-11} \\ y_t & u^{\text{nm}}_{t|t-10} \end{pmatrix} \tag{5}$$

This is actually what is called the *design matrix* in linear regression.

Now the point is, that if we want a forecast model for k steps ahead, then we can simply use `lm()` in R on this data.

Try to learn how the data is setup, divide into a training and a test set, fit a linear regression model for $k = 1$ step ahead. Does the forecast look reasonable?

Try to calculate a forecast for $k = 36$ steps ahead, try for a different house. Give an example and a summary of what you find.

Question 2

Now we do know that there are dynamics, such that the heating doesn't change immediately when the ambient temperature change, that's why we usually will use a time series model (discrete ARMAX or continuous GB), however here we introduce a slightly simplified way to do it.

Lets write up an ARX model

$$\phi(B)Y_t = \omega(B)u_t + \varepsilon_t$$

and rewrite it into transfer function form

$$Y_t = \frac{\omega(B)}{\phi(B)}u_t + \varepsilon_t$$

and let

$$Y_t = H(B)u_t + \varepsilon_t$$

where $H(B)$ is a transfer function.

We also know, that the response of a building can be modelled as an R-C network, which lead to a low-pass filtering effect. Hence, we can apply a low-pass filter to the input, and the use that in the linear regression.

Go and open `q2_lowpass_filter.R` and apply a low-pass filter to the generated on/off signal. It is the simplest first order low-pass filter with stationary gain of one

$$H(B) = \frac{1 - a_1}{1 - a_1 B} \tag{6}$$

What happens with the relation between the input and the low-pass filtered signal e.g. whats the relation between the time constant and a1?

What about the stationary gain? (i.e. the limit y approaches)

Question 3: Load forecast

Go and open `q3_load_forecast.R`. Fit a model with low-pass filtering on the inputs

$$P^{\mathrm{h}}_{t+k|t} = \beta_0 + \beta_1 H(B) T^{\mathrm{a}}_{t+k|t} + \beta_2 H(B) G_{t+k|t} + \varepsilon_t \tag{7}$$

So note that *first* the low-pass filter is applied, such that e.g. $H(B)T^{\mathrm{a}}_{t+k|t}$ are the actual values used in `lm()`.

Is the model tuned for the particular building heat dynamics?

In order to tune the low-pass filter coefficients (one for the ambient temperature and one for the solar radiation) apply an optimizer to minimize the RMSE with leave-one-out cross-validation on the test set.

Are the forecasts improved in terms of RMSE?

Finally, include a diurnal curve using base splines

$$P^{\mathrm{h}}_{t+k|t} = f(t_{\mathrm{day}}) + \beta_1 H(B) T^{\mathrm{a}}_{t+k|t} + \beta_2 H(B) G_{t+k|t} + \varepsilon_t \tag{8}$$

where $f(t_{\mathrm{day}})$ is a spline function.

Make the base splines using $bs()$. Do we get better forecasts?

(Optional) How to choose the degrees of freedom `df`? maybe use AIC or BIC?

(Out of currect scope) Use Fourier series instead as basis functions for the diurnal curve.

Question 4: Recursive least squares

Go and open `q4_recursive_least_squares.R`. In the script the recursive least squares estimation is implemented in the function `rls()`.

Give it a look through and then try the part, where data is generated from two periods, where the parameters are changed.

Does `lm()` estimate the parameters well on the combined data?

Does `rls()`?

Try to change the forgetting factor λ. What happens when it is set to 1 compared to the results obtained from `lm()`?

Is there a trade off between variance and bias (i.e. over- and under-fitting) related to λ?

Question 5: Load forecast with RLS

Now we will use RLS for fitting the coefficients, hence they can change over time. Go and open `q5_load_forecast_rls.R`, run first the first part:

- How about the forecasts, do they look fine?
- The tracked coefficients (the βs kept in θ), do they change?
- What was λ set to? it that optimal?

Run the next part plotting the first month of the training set:

- Are the forecasts good the week?
- What about the coefficients?

Now, since the forecasts are poor until the coefficients are tracked, then make a "burn-in period", which simply means that a period in the beginning of the training set is left out in the score evaluation. Run next part and tune the parameters:

- Are the forecasts good the first week?
- What about the coefficients, did they change?

Run next part:

- Comment on the results: did the forecast improve? Does the coefficients change over time?

Final part is optional: Script for setting the forecasts for multiple horizons.

Question 6: Solar forecasting

Now we can "easily" find a model which is useful for forecasting solar, e.g. the power generation on a PV panel. In the exercise, we will actually just use the observed global radiation as the solar power, hence this is somewhat a little bit simplified case. However, as presented in **?** these observations contain quite a few deviations: shadowing in the late morning hours from a chimney, some tilt of the sensor and also some saturation.

Open `q6_solar_forecast.R` and run firs part, where a very simple linear regression model is fitted:

- Do the forecasts seem to be good?
- Can you spot any systematic patterns?

Explore the residuals in the next part:

- Do you find any systematic patterns?
- What can cause the found differences in the relation between NWP global radiation G_{nwp} and the observed solar power (i.e. observed global radiation) P_{s}?

Now, define a model the relation between the solar power and the global radiation NWP is conditional on the time of day

$$P^{\text{s}}_{t+k|t} = f(t^{\text{day}}_{t+k}) G^{\text{nwp}}_{t+k|t} + \varepsilon_{t+k|t} \tag{9}$$

Fit it, using base splines and `lm()` and `rls()`, and finally with a kernel model.

- How can you decide which model is better?
- Do you find differences between the prediction performance of the models?

Question 7: Wind forecasting

Go and open `q7_wind_forecast.R`. Note: These are not real wind power measurements. The "wind power" P_w, which is used, is actually the observed wind speed put through a simple power curve function. Hence this is a "simplified" wind power, which have the basic properties of a "real" wind power – remember "real" wind power signals are potentially quite different depending on the wind turbine and wind farm properties, and surroundings etc. Hence, the signals used holds the basic properties of a wind power measurement.

Run first part and fit a linear model:

- Look at the scatter plot of the wind power P_w vs. wind speed NWP W_s (`plot(X$Ws, X$Pw)`) with the linear model fit (`abline(fit)`). Can you conclude that a linear model is suitable?

Run the next part:

- Are the forecast improving when you apply a base spline model taking into account a non-linear functional relationship between W_s and P_w?

Run next part: (`## Base spline model with rls`). So now we fit the base splines model with RLS:

- Do we achieve improvements with RLS over LM?
- If not, can you give some explanation? (i.e. somehow weird, since RLS with $\lambda = 1$ give the same result as LM, right!?)...what was the optimal λ found using the training set?

Finally, what about the kernel model? Do we achieve improvements?

So, this was a very "simplified" example, when forecasting "real" wind power of, say, a huge wind farm, then there are many things to take into account. First, surely a conditional dependence of wind direction is important, and further, a lot of information about the operational status is very important to take into account.

Question 8: Probabilistic forecasting

Finally, an example of how a probabilistic forecasting model can be setup is presented. It is very simple: Replace linear regression `lm()` with quantile regression `rq()`, and replace the RMSE score function with CRPS!

Go and open `q8_forecast_probabilistic.R`. Run both the linear and the base splines model:

- Go and find some information about the Continuous Ranked Probability Score (CRPS). When you got it, please fill out `https://en.wikipedia.org/wiki/Continuous_ranked_probability_score`!!
- Do we gain in predictive performace using the base spline model over the linear model?

Finally, an optional task is left to make a locally fitted quantile regression model (hence using a kernel). If this is of your interest, you basically have to replace `lm()` with `rq()`, and rmse() with `crpsDecomposition()$PRBS`...and yes, the same approach can be applied for load, solar, temperature, etc. forecasts!