Exercise on semi- and non-parametric models

Summer school 2019 DTU - CITIES and NTNU - ZEN:

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

August 25, 2019

Introduction

In this exercise the basics of non-parametric modelling techniques are introduced through an example where the heat load of a building is modelled. The aim is to give a hands-on experience with the techniques for modelling non-linear effects - using statistical models, where the non-linear functions are modelled without specifying the exact parametric function.

An example of a non-linear *parametric* function is

$$f(x;a) = x^a$$

which has the parameter a. If a is positive and not 1 then its non-linear and smooth. A *non-parametric* function is also non-linear and smooth, but its mathematical formula cannot be expressed explicitly and thus it doesn't have parameters. However, in the process of fitting a non-parametric function to data, several types of parameters are used - we could call them "smoothing parameters". They determine some characteristics of the function, mostly related to how much the function can "bend" and thus can much it can follow the data: If it doesn't follows data adequately, then the function (i.e. model) becomes "under-fitted" to the data, but if it follows data too much, then the model is said to be "over-fitted" to data. The model which is balanced and thus neither under- and over-fitted is called a "suitable model".

The following data is available for three houses:

- P_t (`P` in data) the heat load for the building (W)
- T_t^e (`Te` in data) the external temperature (°C)
- G_t (`G` in data) the global radiation (W/m^2)
- W_t^s (`Ws` in data) the wind speed (m/s)

The observations are 2 day average values. The internal air temperature was not measured, but during the winter period the heating was thermostatically controlled and thus the internal air temperature can be assumed constant.

In the questions below base spline and kernel functions are introduced. The exercise is then to apply them to build and fit non-parametric models to the data - first by "manually" tuning the smoothing parameters and then automizing this using cross-validation techniques. Finally, it will be shown how the techniques can be used realize a so-called semi-parametric model.

Q1: Plot the data and fit a linear regression model

Open the script "`q1_read_data_and_lm.R`". Get familiar with the data and fit the linear regression with heat load P_t as model output and external temperature T_t^e as input

$$P_t = \beta_0 + \beta_1 T_t^e + \varepsilon_t$$

where β_0 and β_1 are the model parameters and ε_t is the error. Until the last question, we do not consider G_t and W_t^s.

How does a linear regression model fit the data? When using the winter period only? When including the entire period?

So check if there are patterns left in the residuals, i.e. check the assumption that the residuals are independent and identically distributed (i.i.d.).

Q2: Base splines intro

Now open the script "`q2_base_splines.R`". The aim of this question is to give an idea of how base splines behave and what the effects of smoothing parameters are. Play around with the `bs()` function.

Try to vary the degrees of freedom (`df`). What happens to the base splines generated?

Try to vary the degree (`degree`). What happens to the base splines generated?

Give the knot points directly (using the `knots` argument). Give the knots as the quantiles of `x`, what happens when `degree = 1` and what happens when `degree = 3`? (exactly how the base splines are generated when `degree > 1` is left out of scope of the exercise, just think about that it is done using some "optimal" procedure).

Try with some non-equidistant `x` sequence, such that the quantiles are not equidistant. What happens with the base splines?

Q3: Base splines model

Go and open the script "`q3_spline_model.R`". Now we want to calculate the base splines as a function of the external temperature. And then use them as input to a linear regression model. In this way, it becomes a non-linear model

$$P_t = \beta_0 + f(T_t^e) + \varepsilon_t,$$

where β_0 is the intercept and $f()$ becomes a spline function. The characteristics of $f()$ depends on how the base splines are generated, hence it does not have any explicit parameters - only the smoothing parameters of the base splines.

Try to fit a model, is it linear or how is it shaped?

Try to change the degrees of freedom (`df`), what happens?

Q4: Kernel functions

Go and open the script "`q4_kernels.R`". Another way to make non-parametric models is to use locally weighted regression. To do this we need a kernel function. There exists many kernel functions, one of the simplest is the triangular kernel

$$k(x_1, x_2, h) = \begin{cases} 1 - \frac{|x - x_i|}{h} \text{ for } |x - x_i| < h, \\ 0 \text{ for } |x - x_i| \geq h, \end{cases}$$

where x_1 and x_2 are two points and h is the *bandwidth*.

Try to calculate and plot the triangular kernel and play around with the parameters. How do they affect the shape of the kernel?

Try to calculate and plot the Epanechnikov kernel and play around with the parameters. How do they affect the shape of the kernel?

Q5: Locally weighted model with a kernel function

Go and open the script "`q5_kernel_model.R`". Fit a locally weighted model for a single point and predict the heat load.

Fit for a sequence and make the plot of the function. Try to change the bandwidth h. How does that change the estimated function between T_e and P?

Q6: Tuning of the smoothing parameters

Go and open the script "`q6_model_tuning.R`". Now we face the challenge of finding the optimal values for the smoothing parameters, either the bandwidth h in the kernel or the degrees of freedom for the base splines. If the model is over-fitted it varies too much (the function is too adapted to the training data and bends around too much), and on the other hand if it is under-fitted, then it is not “bending” and adapting enough to the data.

One approach is to do a cross-validation optimization of a score function. In the case of estimating the (conditional) mean value, the score function should almost always be the Root Mean Squared Error (RMSE).

The most robust way is to do *leave-one-out*:

- Fit the model without observation i
- Predict observation i

Do this for all the observations and then calculate the score function using the predictions.

In this way we can find the right balance between under- and over-fitting (i.e. the suitable model).

Carry out leave-one-out cross validation. Try to change the bandwidth. What happens to the RMSE score?

Of course we cannot use our time doing manually optimization, so use an optimizer to optimize the bandwidth. Does the result look reasonable for the locally weighted model?

Use leave-one-out cross validation for the base spline model. Does the result look reasonable?

For the base spline models a model selection criteria, such as AIC or BIC can be used. Try that for the base spline model and compare. Do you get the same results?

Q7: Semi- and conditional parametric models (optional for reporting)

Until now, we have calculated the weights and base splines using the input to the model, namely T_t^e. What if we used another variable, but still fitted the same model. What will then happen?

Lets try to calculate the weights using the time t. By doing that we actually allow the coefficients in the model to change as a function of time. We would usually in this case add a t to the parameters

$$P_t = \beta_{0,t} + \beta_{1,t} T_t^e + \varepsilon_t,$$

indicating that they change over time.

This model can of course also be fitted using base splines.

Go and open the script "`q7_semi_parametric_models.R`".

Describe how the coefficients change over time.

What happens with the coefficients during the summer?

Can you explain the result in relation to how the heating system of the building is operating?

Q8: More aspects (optional for reporting)

In this question we will deal with two aspects:

- How to apply a 2. order local model and use another type of kernel function
- Investigate the effect of external temperature, conditional on the wind speed

First, go and open "`q8_more_aspects.R`".

In the first part: 2. order inputs are included into the model, hence it is now a local polynomial model. By including these, the curvature of the function is better estimated, hence this can, when the function is "bending" a lot, lead to a better fit.

Try changing the formula `frml` to find the model which minimizes the cross-validated score.

What is the best model you can find, does it have any second order inputs?

What happens to the bandwidth found with cross-validation when a 2. order term is included?

Compare the results of the `tri()` and `epanechnikov()` kernel function. Does is seem like one of them lead to slightly better fits?

In the second part: Investigate the effect of the external temperature conditional on the wind speed.

How does the coefficient for T_e change as a function of W_s?

Can you explain these results based on your knowledge from physics about building heat transfer?

Software for non- and semi-parametric modelling

It can be an advantage to use packages in R, since they often provide more advanced functionality and can make the workflow easier – most popular seems to be `np` for

kernel regression and `gam` for spline models. Of course packages are available in other languages, e.g. Python or C++, use some search engine to find the them.