System Identification – Practical Assignment 4
Linear Regression for Function Approximation

Logistics

This practical assignment should very preferably be carried out by each student separately. Otherwise, if there are more students than computers, *with the explicit agreement of the lab teacher for each group*, students may team up in groups of 2.

The assignment solution consists of Matlab code. Develop this code in a single Matlab script. If you need to create functions, they can be local to the script, see local functions in scripts.

The overall rules for labs are described on the website. For each particular lab, your attendance will only be registered if you have a working, original solution. The teacher will check that your code **works** during the lab class. Only after this has been done, for the **originality** check, upload your solution here:

`https://www.dropbox.com/request/6BHcqHFYZ5fsJBFXC2cl`

Upload *only once, a single m-file, named exactly according to the following pattern*:

`L4_ENgh_LastnameFirstname.m`

where `g` is your group, `h` your halfgroup, and your last and first names follow. E.g., `L4_EN32_PopAlex.m`. If you worked in a group per the above, only upload one file with both student names included. Any files that are duplicate, nonstandard, inappropriately named, or corresponding to unchecked solutions will not be considered. Files will be automatically tested for plagiarism, and any solution that fails this test will be marked copied; only solutions that pass both the working and the originality test are definitively validated. Therefore, while you are encouraged to discuss ideas and algorithms amongst colleagues, sharing and borrowing pieces of code is forbidden.

Assignment description

In this assignment we will perform function approximation with linear regression and polynomial approximators, see *Linear Regression* in the course material *Part 3 – Mathematical Background.*

A data set of input-output pairs is given, where the outputs are generated by an unknown function g. The function has one input variable and one output variable, and the output measurements are affected by noise. You will develop an approximator of this function, using a linear model with polynomial terms (basis functions). The parameters of the model will be found using the identification data set. A second data set is provided for validating the developed model. The two data sets are given in a MATLAB data file, containing one structure for each set. The training data set is named `id` and the validation data set `val`. Each of these structures contains a vector X of input samples, and the corresponding output samples in vector Y.

Each student is assigned an index number. Then, the student downloads the data file that form the basis of the assignment from the course webpage.

Requirements:

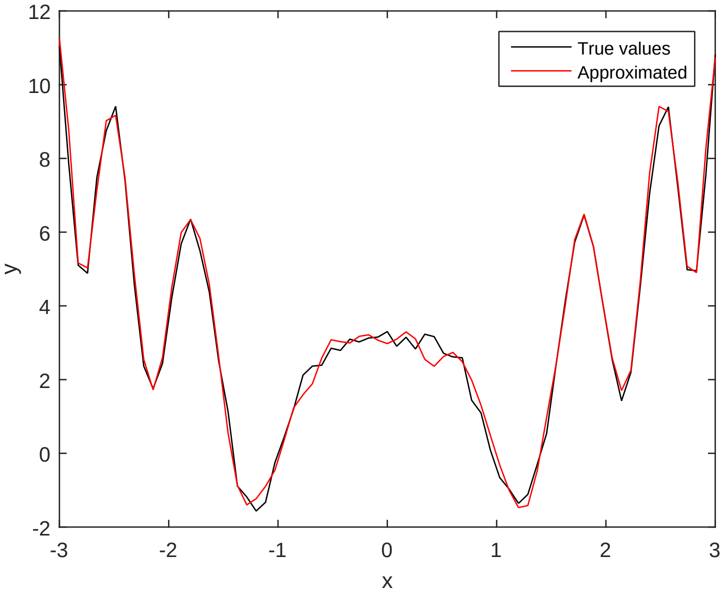
- Plot the identification data to get an idea of the function shape.
- Create a polynomial approximator of degree $n-1$, where n is the number of parameters / basis functions. Here, n should be tunable. Note there is one extra parameter for the constant term, which is why the degree is just $n-1$. For example, when $n=4$, the polynomial has degree 3 and

the approximator is:

$$\hat{g}(x) = \theta_1 + x\theta_2 + x^2\theta_3 + x^3\theta_4$$

- For any value of n, create a system of linear equations for linear regression, using the identification data. Use the matrix representation explained in the lecture. Solve this system using matrix left division, operator $\backslash$ in Matlab (or alternatively with `linsolve`). Report the MSE on the identification data.
- Validate the model on the different, validation data set: compute the approximated outputs and from those the MSE on the validation data. Show a plot of the approximated function on the validation data set, comparing to the actual outputs.
- Tune n for good performance (by trying values up to, say, 20). Performance should be evaluated by the MSE on the different, validation data set to avoid overfitting. Produce a plot of the MSE versus n and find the point where the MSE is minimal, which corresponds to the optimal polynomial degree.

Your plots will look similar to those exemplified in the next figure (except your data and fit quality may be different, of course). It is normal, and expected, to see a poor validation fit when n is too low (underfitting, approximator not powerful enough to model the dataset), and when n is too large (overfitted, approximator too powerful so it starts modeling noise, and the MSE on the validation data increases while the MSE on the identification data keeps going down).


y
x
True values
Approximated