System Identification – Practical Assignment 6
ARX model identification

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/jW3K8rqsqiOFzxyaqxYn`

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

`L6_ENgh_LastnameFirstname.m`

where `g` is your group, `h` your halfgroup, and your last and first names follow. E.g., `L6_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 identify ARX models (autoregressive with exogenous input), using least-squares, linear regression. See the course material, Part V: *ARX Identification*.

Each student is assigned an index number by the lab teacher. Then, the student downloads the data file that forms the basis of the assignment from the course webpage. The file contains the identification data in variable `id`, and separately the validation data in variable `val`. Both these variables are objects of type `iddata` from the system identification toolbox of Matlab, see `doc iddata`. It is known from prior knowledge that the system does not have any time delay.

Requirements:

- Plot and examine the data supplied.
- Implement ARX identification explicitly using linear regression, as described in the lecture. Recall that the regressors are $-y(k-1), \ldots, -y(k-na), u(k-1), \ldots, u(k-nb)$. Your code should work for any values of na and nb.
- Moreover, implement the simulation of the computed model for the validation data. Keep in mind that for simulation, knowledge about the real outputs of the system is not available, so we can only use previous outputs of the model itself; in particular $y(k-i)$ in the model formula must be

replaced by its previously simulated value $\hat{y}(k-i)$, for $i = 1, \ldots, na$. Hint: signals at negative or zero time steps can be taken equal to zero.

- Try to guess a system order from the step response shapes in the validation data. Set the na and nb orders of the ARX model accordingly, and identify a model with your code, on the identification data. Then, on the validation data, compare the output simulated with your model with the real output.
- If the results are poor, increase na and nb (e.g., in increments of 1, or make them twice as big, etc.) until you get a good fit.
- Optionally, if you still have time – or if you have bugs and want a known good solution – identify models with the same values of na, nb as above, but this time with the Matlab `arx` function (with $nk = 1$ since the system is known to not have a time delay). Compare the results with those that you obtained using your code, and verify that the two results are similar.

Relevant functions from the System Identification toolbox: `arx`, `plot`, `compare`. When the `ident` toolbox function has the same name as a function in another toolbox – like in the case of `compare`, which overloads the MPC toolbox implementation – write e.g. `doc ident/compare` to get the documentation of the `ident` variant. See also `doc ident` for the full documentation of the toolbox.