System Identification – Practical Assignment 9
Instrumental variable methods

Logistics

- This practical assignment should be carried out by each student separately, if at all possible. In extremis, only if there are more students than computers, students may team up in groups of 2.
- The assignment solution consists of Matlab code. This code will be checked and run by the teacher during the lab class, and your attendance to the lab will only be registered if you have a working, original solution. Validated attendances for all the labs are necessary for eligibility to the exam. Moreover, at most two labs can be recovered at the end of the semester, which means accumulating three or more missing labs at any point during the semester automatically leads to final ineligibility.
- Discussing ideas among students is encouraged; however, directly sharing and borrowing pieces of code is forbidden, and any violation of this rule will lead to disqualification of the solution.

Assignment description

In this assignment we will study instrumental variable methods, see Part 8 of the course material, *Instrumental Variable Methods. Closed-Loop Identification.*

Each student is assigned an index number by the lecturer. Then, the student downloads the data file that forms the basis of the assignment from the course webpage:
`http://busoniu.net/teaching/sysid2018`

The file contains the identification data in variable `id`, and separately the validation data in variable `val`. From prior knowledge, it is known that the system has order n, given in variable `n` in the data file; and that the disturbance is not white noise, but colored. All polynomial orders in the models below should be set in accordance with this value of n.

In the first part of this lab, we will implement the IV method with arbitrary instruments generated by a discrete-time transfer function:

$$x(k) = \frac{D(q^{-1})}{C(q^{-1})}u(k)$$

As an example, if $C(q^{-1}) = 1 + 0.5q^{-1}$ and $D = 0.1q^{-1}$, then the IVs are computed by simulating $x(k) = -0.5x(k-1) + 0.1u(k-1)$. To solve the identification problem efficiently in Matlab, it will be useful to rewrite the IV system of equations in a form amenable to matrix left division. To that end, let us take equation (8.3) from the lecture slides and rewrite it as:

$$\left[\frac{1}{N}\sum_{k=1}^{N} Z(k)\varphi^T(k)\right]\theta = \frac{1}{N}\sum_{k=1}^{N} Z(k)y(k)$$

$$\text{or equivalently: } \tilde{\Phi}\theta = \tilde{Y}$$

where the $n\times n$ matrix $\tilde{\Phi} = \frac{1}{N}\sum_{k=1}^{N} Z(k)\varphi^T(k)$ and the $n\times 1$ vector $\tilde{Y} = \frac{1}{N}\sum_{k=1}^{N} Z(k)y(k)$. Note the tildes, which signify that these quantities are variants of the original ARX regressors and of the original system outputs, "modified" by the IVs. Your task is to implement the algorithm in a function that takes at the input the identification dataset, the model orders na and nb, and the polynomials C and D as vectors of coefficients in increasing powers of q^{-1}; and that produces at the output the IV model found, in the `idpoly` format.

Requirements:

- Identify an ARX model of orders $na = nb = n$ and inspect its quality. For this step you may use either the Matlab function `arx`, or your code developed for the ARX lab.
- Apply IV identification with the simple instruments:

  $$Z(k) = [u(k-nb-1), \dots u(k-na-nb), u(k-1), \dots, u(k-nb)]^T$$

  by using your function with appropriately chosen polynomials: $C(q^{-1}) = 1$ and $D(q^{-1}) = -q^{-nb}$.
- Apply IV identification with the instruments:

  $$Z(k) = [-\hat{y}(k-1), \dots, -\hat{y}(k-na), u(k-1), \dots, u(k-nb)]^T$$

  where the outputs $\hat{y}$ are those of the ARX model found earlier. Use again your function but now setting C and D from the ARX model. Compare the results with those obtained using ARX and using the simple instruments.
- Repeat the previous two points with the existing Matlab function `iv`, and verify that you obtain similar results to those of your function (due to algorithmic details they may not be exactly identical). Set $nk = 1$ in `iv`.

Relevant functions from the System Identification toolbox: `arx`, `iv`, `compare`. Hints: (i) Construct $\tilde{\Phi}$, $\tilde{Y}$ efficiently by summing up terms computed using matrix operations in Matlab. (ii) Once you have your polynomials A and B as vectors of coefficients in increasing powers of q^{-1}, use `idpoly(A, B, [], [], [], 0, Ts)` to generate the IV model, where `Ts` is the sampling period. (iii) Do not forget that all vectors of polynomial coefficients must always contain the leading constant coefficients (i.e. for power 0 of the argument q^{-1}), which must be 1 in C and A, and 0 in B.