System Identification – Practical Assignment 9
Instrumental variable methods

Logistics

Please reread the logistics part of lab 2, the same rules will apply to this lab. The only thing that changes is the dropbox link, which for this lab is:

`https://www.dropbox.com/request/tv5oDk8CVgG5oX3rrZQk`

Assignment description

In this assignment, your task is to implement the method of instrumental variables (IV) using instruments based on ARX outputs, and apply it to identify the DC motor. We know the DC motor exhibits delays, so we will identify IV models with orders na, nb and delay nk.

To solve the identification problem efficiently in Matlab, we will use the following form of the IV system of equations, amenable to matrix left division (backslash operator):

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

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

where the $(na + nb) \times (na + nb)$ matrix $\tilde{\Phi} = \frac{1}{N}\sum_{k=1}^{N} Z(k)\varphi^T(k)$ and the $(na + nb) \times 1$ 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 regressors and of the original system outputs, “modified” by the IVs.

Since we have delays, in the equation above the regressor vector is:

$$\varphi(k) = [-\hat{y}(k-1), \ldots, -\hat{y}(k-na), u(k-nk), \ldots, u(k-nb-nk+1)]^T$$

and the instrument vector is:

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

where the outputs $\hat{y}$ are simulated with the ARX model. Do not use predicted outputs, as those are correlated with the disturbance and will likely break the IV method! Recall that $\theta = [a_1, \ldots, a_{na}, b_1, \ldots, b_{nb}]^T$.

Each student will obtain a data set using the DC motor and will identify the system, as detailed in the following instructions.

1. To keep things simple, we will create a single, longer sequence of data containing both the identification and validation data. We will use a sampling rate of 0.01 s (10 ms). After a short range of zero inputs, apply a PRBS signal with amplitudes in the interval $[-0.8, 0.8]$ and a length of about 300 samples, followed by another range of zero inputs, and then a step signal of magnitude around 0.3 and around 70 samples in length. You can either use `idinput` or your own PRBS code from the previous labs, but in the latter case use a sufficiently large number of bits so that the signal does not repeat.

2. Apply the signal generated to the DC motor. The output is the rotational velocity. Isolate the data range corresponding to the PRBS input and copy it to new input and output vectors; this will be our identification data. **Important note**: To minimize system wear, separate the code that generates the data from the code that performs the rest of the steps below (easiest using different script sections, see *Code Sections* in the Matlab documentation), and regenerate the data only when necessary.

3. Plot and examine the data obtained.

4. Identify an ARX model with orders $na = nb$ and delay nk. For this step it is strongly recommended to use the Matlab function `arx`, in which case you will need to create a standard identification dataset with `iddata`. You can also use your own code developed in previous labs, but you may need to modify it to handle the delay which could easily make you run out of time for the steps below.

5. Identify a model with the IV method, with the same orders as for ARX, using the math described above and the ARX outputs as instrumental variables. Hints: (i) If you used `arx`, you can obtain the A and B polynomials to initialize the IV generator polynomials D and C by writing `arxmodel.A`, `arxmodel.B`; otherwise, it will likely be simpler to just simulate the ARX model and fill in Z directly with the $\hat{y}$ and u values, rather than defining polynomials C and D. (ii) Construct $\tilde{\Phi}$, $\tilde{Y}$ efficiently by summing up terms computed using matrix operations in Matlab. (iii) Don't forget to fill in zeros for negative-and-zero time steps in the vectors Z and φ.

6. Check the performance of the IV model (in simulation). Hint: You can use `idpoly(A,B,C,D,F,0,Ts)` followed by `compare` on an `iddata` object for the validation dataset. In `idpoly`, you need to specify the nk leading zeros in B, the leading 1 in A, and the sampling time. The polynomials that you don't use can be set to 1.

7. Tune $na = nb$ and the delay nk to obtain good performance. Hints: It will help to use a second-order model, even though the system is first-order; and the delay can change between different experiments.