System Identification – Practical Assignment 7
Pseudo-random binary sequences

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/8LiBLCVfFl7EIzFgEc0F`
Upload *only once, a single m-file, named exactly according to the following pattern*:
`L7_ENgh_LastnameFirstname.m`
where `g` is the group, `h` the halfgroup, and the last and first names follow. E.g., `L7_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 study the creation and properties of pseudo-random binary sequences, PRBS. See the course material, Parts VI: *Input Signals*.

Each student group is assigned an index number by the lecturer. Then, the group downloads from the course webpage the `system_simulator` function, which obtains experimental data from the system given an input signal. For the purposes of this lab, since we do not have access to the real system, this simulation takes the place of the real experiment, for both identification and validation. The function is given in obfuscated, so-called p-code, so that you can treat the simulator as an unknown system, as would be the case in a real experiment. The signature of the function is `data = system_simulator(index, u)` where `index` is your index number, `u` is the input sequence (in discrete time), and `data` is the returned experimental data as a standard object of type `iddata`.

From prior knowledge, it is known that the system to be identified has order not larger than 4, and that the disturbance does not satisfy the structural assumptions of the ARX model. Nevertheless, due to its simplicity we choose to identify an ARX model, and to compensate for the disturbance structure we take large values for the orders: $na = nb = 15$. Note that in order to identify an ARX model, the input signal should satisfy a certain persistence of excitation condition, see the lecture for the details.

Requirements:

- Generate first a validation dataset with `system_simulator`, using as input e.g. the predefined input signal `u` provided in datafile `uval.mat`. You can use this dataset to validate all the models found below.
- Write a function that generates an input signal of length N using a maximum-length PRBS with a register of a given length m, and which switches between given values a and b. Parameters N, m, a, b are given as inputs to this function, and m is limited to the range $3, 4, \ldots, 10$. Note that if $N > P$, the period of the PRBS, then the input signal will consist of several repetitions of the maximum-length PRBS (this should happen automatically, you do not need to do anything). Test this function for some values of N, m, a and b. Hint: You can use function `mod` to implement the modulo-2 summation.
- Generate an identification input signal of sufficient length (say around 300 samples) with $m = 3$, taking values $a = 0.5$ and $b = 1$. Apply this signal to the system using `system_simulator`.
- Identify an ARX model with the identification data obtained, using either the Matlab `arx` function for simplicity, or otherwise your own code from the previous lab. Compute (on paper) the order of persistent excitation for the input. Is it sufficient to identify properly the ARX parameters? Verify the quality of the ARX model on the validation data, and see if it supports your conclusion.
- Repeat the above but now with $m = 10$. Does this new data have a sufficient order of persistent excitation? Verify the quality of the ARX model.

Relevant functions from the System Identification toolbox: `arx`, `plot`, `compare`.