System Identification – Practical Assignment 10: Recursive ARX identification

Logistics: Same as for previous labs.

In this assignment we will study the recursive variant of the ARX method, see the lecture: *Recursive identification methods*. You will develop a function with the exact signature:

```
[index, Pinv10, theta50, thetaN] = rarxidentify
```

Each student is assigned an index number in the set 1-8, which needs to be saved to variable `index` at the beginning of the function. The index dictates which data file the student should load. For instance, if you have index 3, you load file `lab10_3.mat`. All these datafiles are already accessible from your function code. Each file contains the identification data in variable `id`, and the validation data in variable `val`. Hint: to increase execution speed and avoid timeouts, save the vectors of inputs and outputs to separate arrays instead of always accessing the `id` object.

From prior knowledge, it is known that the system has order n, given in variable `n` in the data file; that it is of the output error (OE) type; and that it has no time delay. To account for the model mismatch we will take larger orders for the ARX model: $na = nb = 3\cdot$`n`.

- Implement the recursive ARX algorithm, see the pseudocode below with additional hints compared to the lecture. Run the algorithm on the identification data, using an initial matrix $P^{-1}(0) = \frac{1}{\delta} I_{na+nb}$ with $\delta = 0.01$, and a zero initial parameter vector $\theta(0)$. As an intermediate verification, return $P^{-1}(10)$ in output `Pinv10` of the function.
- Return in `theta50` the parameter vector $\theta(49)$ after processing 49 data points,[1] and in `thetaN` the final parameter vector $\theta(N)$ after processing all data points.
- Using function `compare`, investigate the quality of the two ARX models corresponding to these two parameter vectors *on the validation data*. Which model is better? Think about the reasons.
- Optionally, if you still have time, repeat the experiment with $\delta = 100$. Consider the results. For which value of δ are the models worse, and why?

---

initialize $\widehat{\theta}(0)$ (an $na + nb$ column vector), $P^{-1}(0)$ (a $(na + nb) \times (na + nb)$ matrix)

loop at every step $k = 1, 2, \ldots$

  retrieve $u(k)$, $y(k)$

  form ARX regressor vector: $\varphi(k) = [-y(k-1), \cdots, -y(k-na), u(k-1), \cdots, u(k-nb)]^\top$

  find prediction error: $\varepsilon(k) = y(k) - \varphi^\top(k)\widehat{\theta}(k-1)$ (a scalar)

  update inverse: $P^{-1}(k) = P^{-1}(k-1) - \frac{P^{-1}(k-1)\varphi(k)\varphi^\top(k)P^{-1}(k-1)}{1+\varphi^\top(k)P^{-1}(k-1)\varphi(k)}$

  compute weights: $W(k) = P^{-1}(k)\varphi(k)$ (an $(na + nb)$ column vector)

  update parameters: $\widehat{\theta}(k) = \widehat{\theta}(k-1) + W(k)\varepsilon(k)$

end loop

Relevant functions from the System Identification toolbox: `rarx`, `idpoly`, `compare`. Additional hints:

[1] The reference solution has an off-by-1 error, apologies.

- 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 ARX model, where `Ts` is the sampling period. Do not forget that all vectors of polynomial coefficients must always contain the leading constant coefficients (power 0 of the argument q^{-1}), which must be 1 in A and 0 in B. Keep in mind that the matrix of parameters returned by the algorithm does *not* contain these leading coefficients.

- Matlab also offers function `rarx` that you can use to verify your results, if you wish. This function takes at the input the identification dataset, the model orders na and nb and the delay nk as a vector, the `'ff', 1` arguments to configure the algorithm as in the lecture, the initial parameter vector $\theta(0)$, and the initial inverse matrix $P^{-1}(0)$. The quantity denoted `P` by documentation of the `rarx` Matlab function is actually the *inverse* matrix P^{-1} from the lecture, so be careful when setting it. The function produces at the output a matrix $\Theta \in \mathbb{R}^{N\times(na+nb)}$ containing on each row k the parameter vector $\theta(k)$: first the coefficients $a_1, \ldots, a_{na}$ of A, and then the coefficients $b_1, \ldots, b_{nb}$ of B.