FrontSeat Summer School on *Optimization-based Control Systems: Theory & Applications*

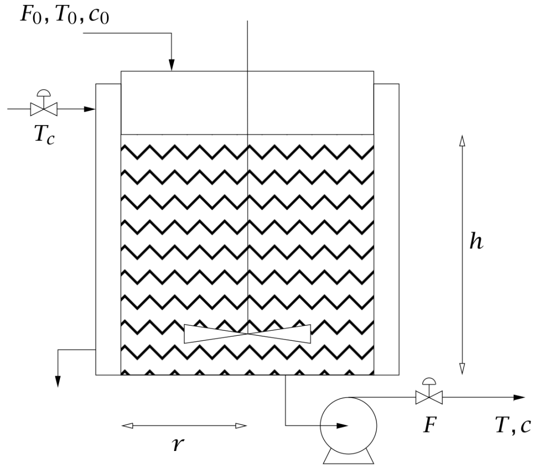
Exercise 3: System Identification with `SIPPY` and `Keras`

Riccardo Bacci di Capaci, Karol Kiŝ

This exercise PDF and all accompanying Python template codes can be downloaded or cloned from the course repository https://github.com/CPCLAB-UNIPI/FrontSeatSummerSchool (please follow the instructions in README.md).

Part 1 - CSTR

We consider the same example of Exercise 1: a nonlinear continuous stirred-tank reactor (CSTR). [1]



An irreversible, first-order reaction $A \rightarrow B$ occurs in the liquid phase and the reactor temperature is regulated with external cooling. Mass and energy balances lead to the following nonlinear model:

$$\begin{aligned}
\dot{c} &= \frac{F_0(c_0 - c)}{\pi r^2 h} - k_0 \exp\left(-\frac{E}{RT}\right) c \\
\dot{T} &= \frac{F_0(T_0 - T)}{\pi r^2 h} - \frac{\Delta H}{\rho C_p} k_0 \exp\left(-\frac{E}{RT}\right) c + \frac{2U}{r \rho C_p}(T_c - T) \\
\dot{h} &= \frac{F_0 - F}{\pi r^2}
\end{aligned}$$

with states $x = (c, T, h)$ where c is the concentration of substance A, T is the reactor temperature and h is the height. The inputs are $u = (T_c, F, F_0)$, that is, the coolant liquid temperature T_c, the outlet flowrate F (as two manipulated variables), and the inlet flowrate F_0, as a disturbance. The outputs are $y = x$ since states are assumed measurable.

Description of the environment

Among files and methods of `SIPPY`:

- `Ex_CSTR_temp.py` is the example file (a draft to be edited) containing all the functions to be recalled to perform the system identification;

[1] The example as well as the figure have been adopted from Example 1.11 in Rawlings, J.B., Mayne, D.Q. and Diehl, M., 2017. *Model predictive control: theory, computation, and design (Vol. 2).* Madison, WI: Nob Hill Publishing.

- `__init__.py` is the main file that performs system identification:
- `functionset.py` includes most of the functions used by the identification methods;
- `functionset_OPT.py` contains the nonlinear optimization problem used by optimization-based identification methods;
- `functionsetSIM.py` contains additional functions used by the Subspace Identification Methods and other useful functions for state space models.

Tasks

1. Open-Loop data collection. Around the given steady-state operating condition, define suitable input signals (e.g., GBNs of suitable amplitude, frequency, and length). Generate the corresponding system outputs, adding noises e with normal distribution and suitable variance σ_e.
2. System identification. Identify three different linear models (an ARX, an ARMAX, and an SS), by setting suitable model orders and identification methods. Plot the corresponding predicted output of each model together with the real output and report a suitable metric of error (e.g., MSE, RMSE, or EV).
3. System validation. Repeat the analysis for another suitable dataset to be used for model validation.
4. Model comparison. Compare the obtained models with the following linearized model, available in Example 1.11 of Rawlings, J.B., Mayne, D.Q. and Diehl, M., 2017. Compare in terms of time and frequency response (in open-loop) and validation performance.

$$A = \begin{bmatrix} 0.2681 & -0.00338 & -0.00728 \\ 9.703 & 0.3279 & -25.44 \\ 0 & 0 & 1 \end{bmatrix}, B = \begin{bmatrix} -0.00537 & 0.1655 \\ 1.297 & 97.91 \\ 0 & -6.637 \end{bmatrix}, B_d = \begin{bmatrix} -0.1175 \\ 69.74 \\ 6.637 \end{bmatrix},$$

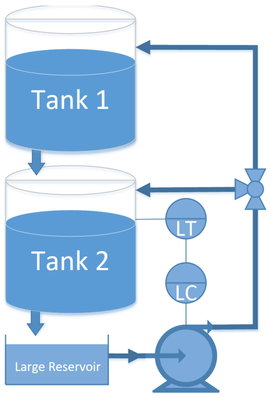
$$C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, D = \mathbf{0}$$

Part 2 - Two tank system

We consider the following two-tank system. [2] Mass balances lead to the following nonlinear model:

$$\frac{h_1}{dt} = c_1(1-\gamma)p - c_2\sqrt{h_1}$$
$$\frac{h_2}{dt} = c_1\gamma p + c_2\sqrt{h_1} - c_2\sqrt{h_2}$$

where 1 and 2 refer to the tanks, with states $x = (h_1, h_2)$, that is, the two level heights. The inputs are $u = (p, \gamma)$, with p the pump flow and γ the valve position, both between 0 and 1. The outputs are $y = x$ since states are assumed measurable; $c1 = 0.08$ and $c2 = 0.04$ are the inlet valve coefficient and the tank outlet coefficient, respectively.



Description of the environment

Among files and methods of `SIPPY`:

- `Ex_tts_temp.py` is the example file (a draft to be edited) containing all the functions to be recalled to perform the system identification.
- The other core files are the same as described in Part 1.

Tasks

1. System identification. Split the available data file `tts_data.csv` into two sets: a first part for identification, a second for validation. Identify three different linear models (an ARX, an ARMAX, and an SS), by setting suitable model orders and identification methods. Plot the corresponding predicted output of each model together with the real output and report a suitable metric of error (e.g., MSE, RMSE, or EV).
2. System validation. Repeat the model analysis for the validation dataset.
3. Parametric analysis. Repeat the identification and validation study by adding noise of different variances (σ_e) and by considering models of different orders; for the SS model apply information criteria for the best order selection. Plot and comment on the obtained results in the various cases.
4. Model comparison. Compare one of the obtained models in terms of validation performance with:
   - a model obtained from the NL system (define balance equations and collect corresponding data).
   - the model obtained with Keras.

[2] The example as well as the figure is adopted from `https://apmonitor.com/do/index.php/Main/LevelControl`.