Identification of Hammerstein-Wiener Models

Group 4:

Bjørnar Ø. Kaarevik, Chinmay A. Patwardhan, Nicole Quattrini, Andreas G. Tufte, Giacomo Melloni

Hammerstein-Wiener Models

INPUT NONLINEARITY

Adjusting input signals

Sound engineer tuning instruments.

LINEAR BLOCK

Dynamic part of the system

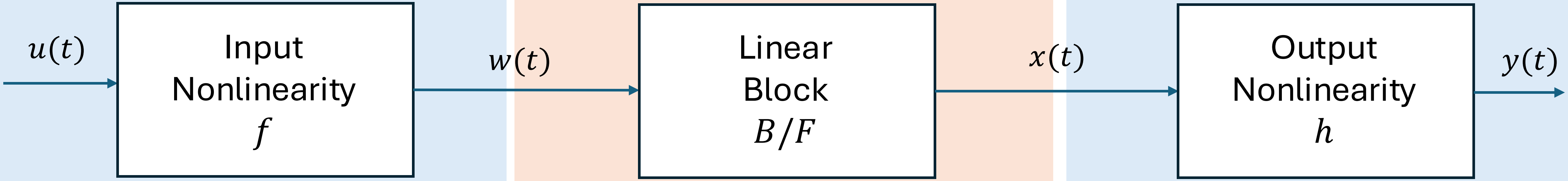
The band killing it.

OUTPUT NONLINEARITY

Adjusting output signals

Sound engineer adding effects for the recording.

HW Model Structure



- f = nonlinear function that transforms input data $u(t)$ as

$$w(t) = f\big(u(t)\big)$$

  - $w(t)$ = internal variable, output of Input Nonlinearity block + has same dimensions as $u(t)$

- B/F = linear transfer function, transforms $w(t)$ as

$$x(t) = (B/F)w(t)$$

  - $x(t)$ = internal variable, output of the Linear block + has same dimensions as $y(t)$

- h = nonlinear function, maps output of linear block $x(t)$ to system output

$$y(t) = h(x(t))$$

Reference: [What Are Hammerstein-Wiener Models? - MATLAB & Simulink - MathWorks Nordic]

Use cases: anywhere nonlinear dynamic systems exist

1. Control System Design
   - Adaptive control
   - Model Predictive Control (MPC)

2. Chemical and Process industries
   - Reactor control
   - Distillation column control

3. Biomedical Engineering
   - Neuromuscular and Biomechanical systems
   - Representation of Pharmacokinetics

Example: Two tank system

The model contains:

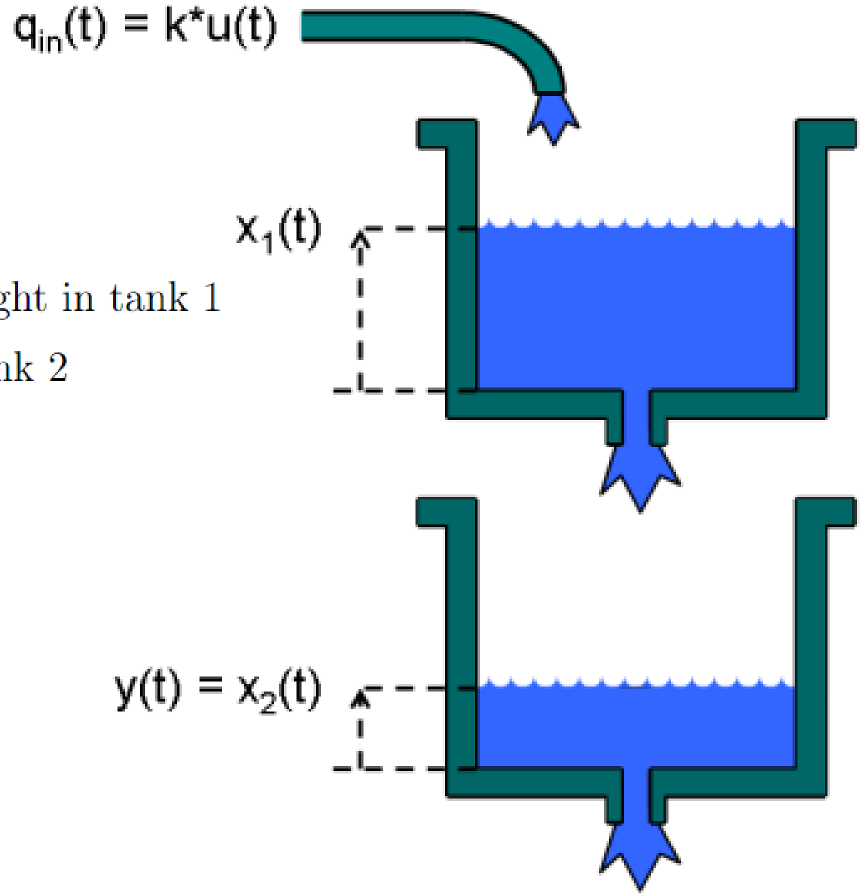
$q_1(t) \propto u(t)$ Linear pump or valve

$q_2(t) \propto \sqrt{x_1(t)}$ Flow from tank 1 to tank 2 is proportional to the square of height in tank 1

$q_3(t) \propto \sqrt{x_2(t)}$ Flow out of tank 2 is proportional to the square of height in tank 2

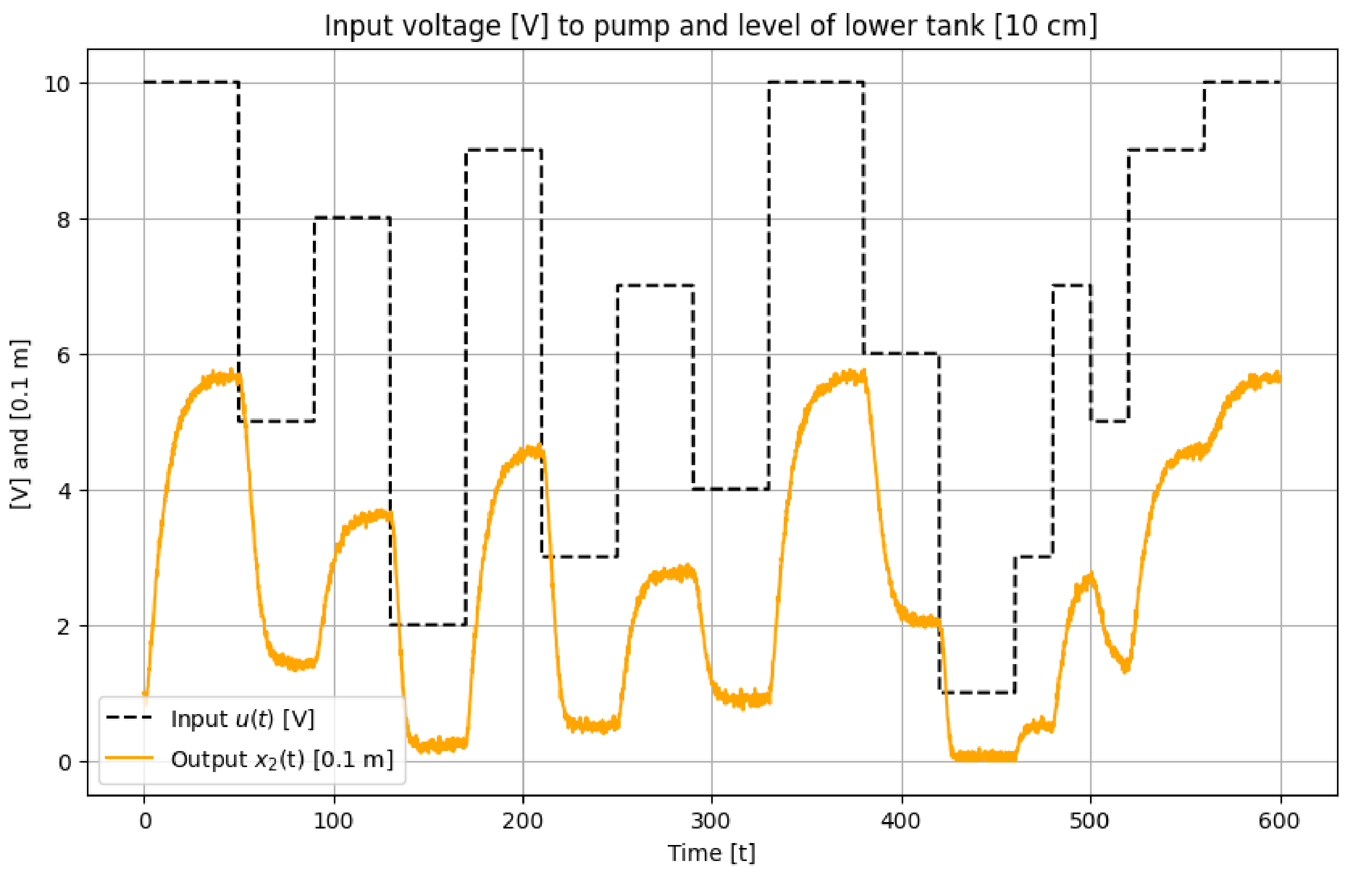
$u(t) = x_2(t)$ Output is the linear level in tank 2

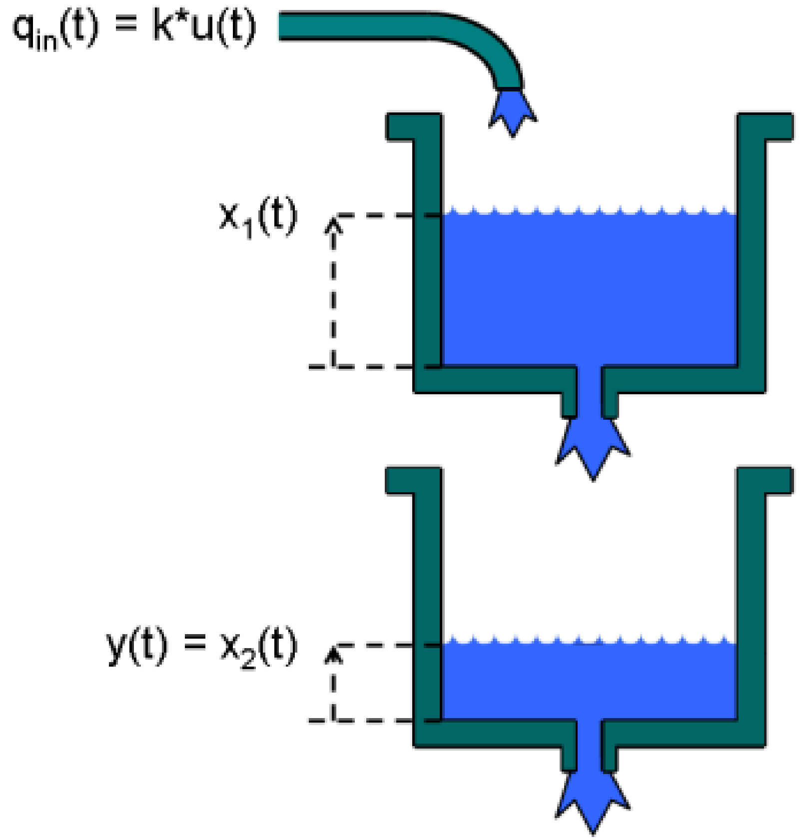
-> Model contains both linear and nonlinear behavior



System Identification Toolbox MATLAB:
https://se.mathworks.com/help/ident/ug/two-tank-system-single-input-single-output-nonlinear-arx-and-hammerstein-wiener-models.html

Input Output relation





We see that the level in tank 2 stabilizes for some flow input defined by u(t)

Model test: NARX

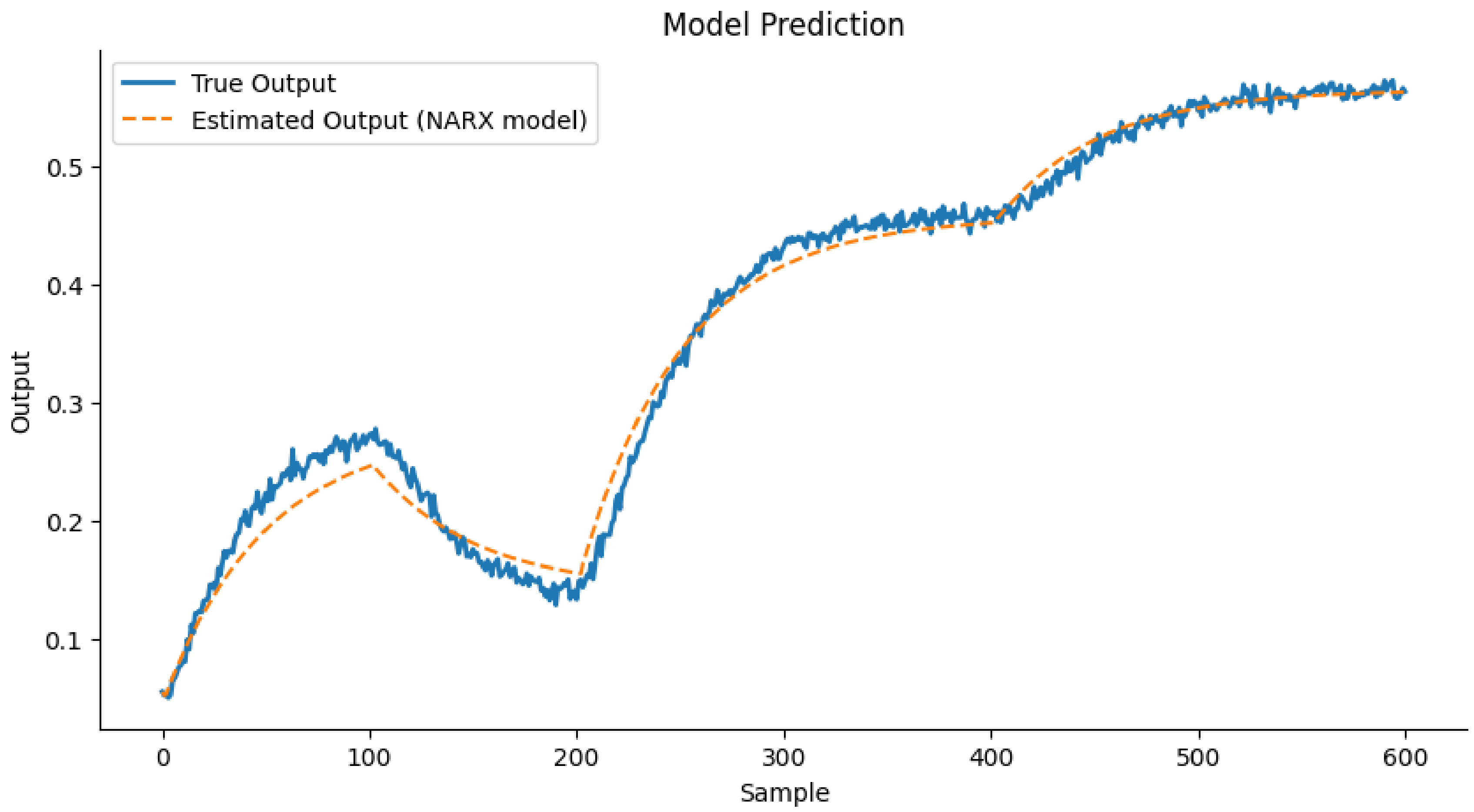
- Used for **non-linear** scenarios (n.l. relationships between Input-Output)

$$y(t) = F\left(y(t-1), y(t-2), \ldots, y(t-n_y),\ u(t-1), u(t-2), \ldots, u(t-n_u)\right) + e(t)$$

where **y** is the output, **u** is the input, ***F*** is some nonlinear function, e is the error term.

- **Polynomial model** is applied
- **FROLS** (Forward Regression Orthogonal Least Squares) for regressors selections (i.e. determine model structure of n.l. system)

Model prediction: NARX



Two-stage optimal HW-method

- Chapter 3 in *Block-oriented Nonlinear System Identification*: "An Optimal Two-stage Identification Algorithm for Hammerstein–Wiener Nonlinear System" (Giri and Bai, 2010)

- This method is implemented in Python
- It uses **a priori information about nonlinear** input and output

Consider a scalar stable discrete time nonlinear dynamic system represented by

$$y(k) = \sum_{i=1}^{p} a_i \{ \sum_{l=1}^{q} d_l g_l [y(k-i)] \} + \sum_{j=1}^{n} b_j \{ \sum_{t=1}^{m} c_t f_t [u(k-j)] \} + \eta(k) \tag{3.1}$$

where $y(k)$, $u(k)$ and $\eta(k)$ are the system output, input and disturbance at time k respectively. The $g_l(\cdot)$'s and $f_t(\cdot)$'s are non-linear functions and

$$a = (a_1, ..., a_p)', \; b = (b_1, ..., b_n)', \; c = (c_1, ..., c_m)', \; d = (d_1, ..., d_q)' \tag{3.2}$$

denote the system parameter vectors. The model (3.1) may be considered as the system where two static nonlinear elements N_1 and N_2 surround a linear block. It is different from the well-known Wiener–Hammerstein model [2] where two linear blocks surround a static nonlinear element and also different from the Hammerstein model discussed in [3, 4, 7, 8, 9] composed of a static nonlinear element followed by a linear block.

The purpose of identification is to estimate unknown parameter vectors a, b, c and d from the observed input-output measurements. Through out the chapter, f_i's $(i = 1, 2, ..., m)$ and g_j's $(j = 1, 2, ..., q)$ are assumed to be *a priori known smooth* nonlinear functions and the orders q, n, p and m are assumed to be known as well.

https://link.springer.com/book/10.1007/978-1-84996-513-2

Algorithm breakdown

$$\theta_{\mathrm{ls}} = (\Phi^T \Phi)^{-1} \Phi^T Y$$

Stage 1: Least Squares Estimation

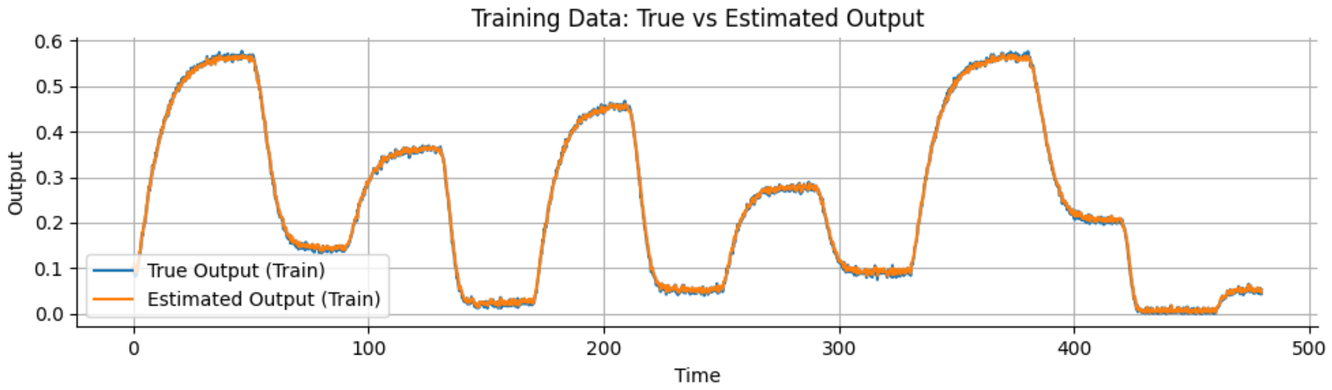
Obtains initial estimates of model parameters. Phi is a regressor containing transformed input and output values

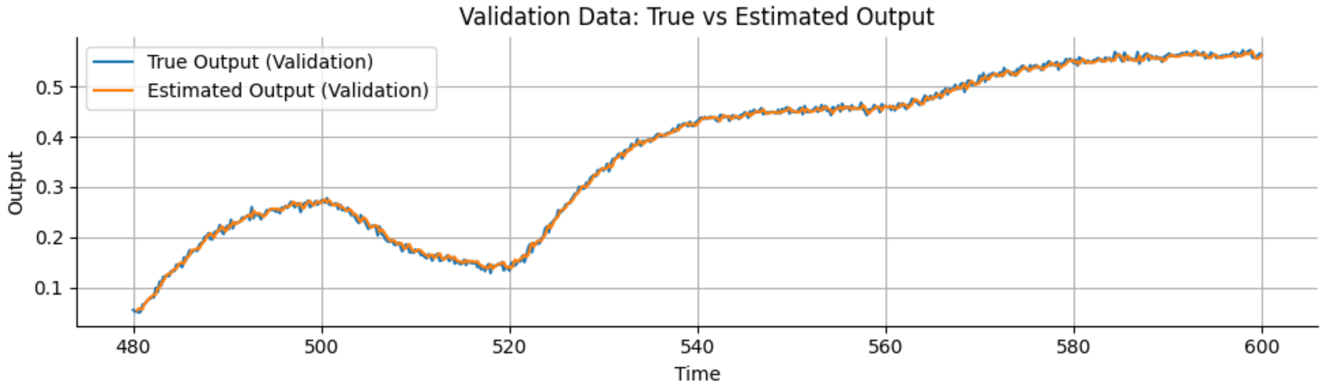
Stage 2: Singular Value Decomposition (SVD)

Decompose the LS estimate with SVD to find (a, b, c, d)-vectors (see last slide)

- The two-stage method is efficient because it **breaks down a high-dimensional problem** (finding a, b, c, d last slide) into simpler steps. First by over-parameterizing, then refining the solution using SVD.
- It is **globally optimal** for white noise disturbances and converges to the true system parameters.

Two-stage HW-method (something's went wrong in the code..)

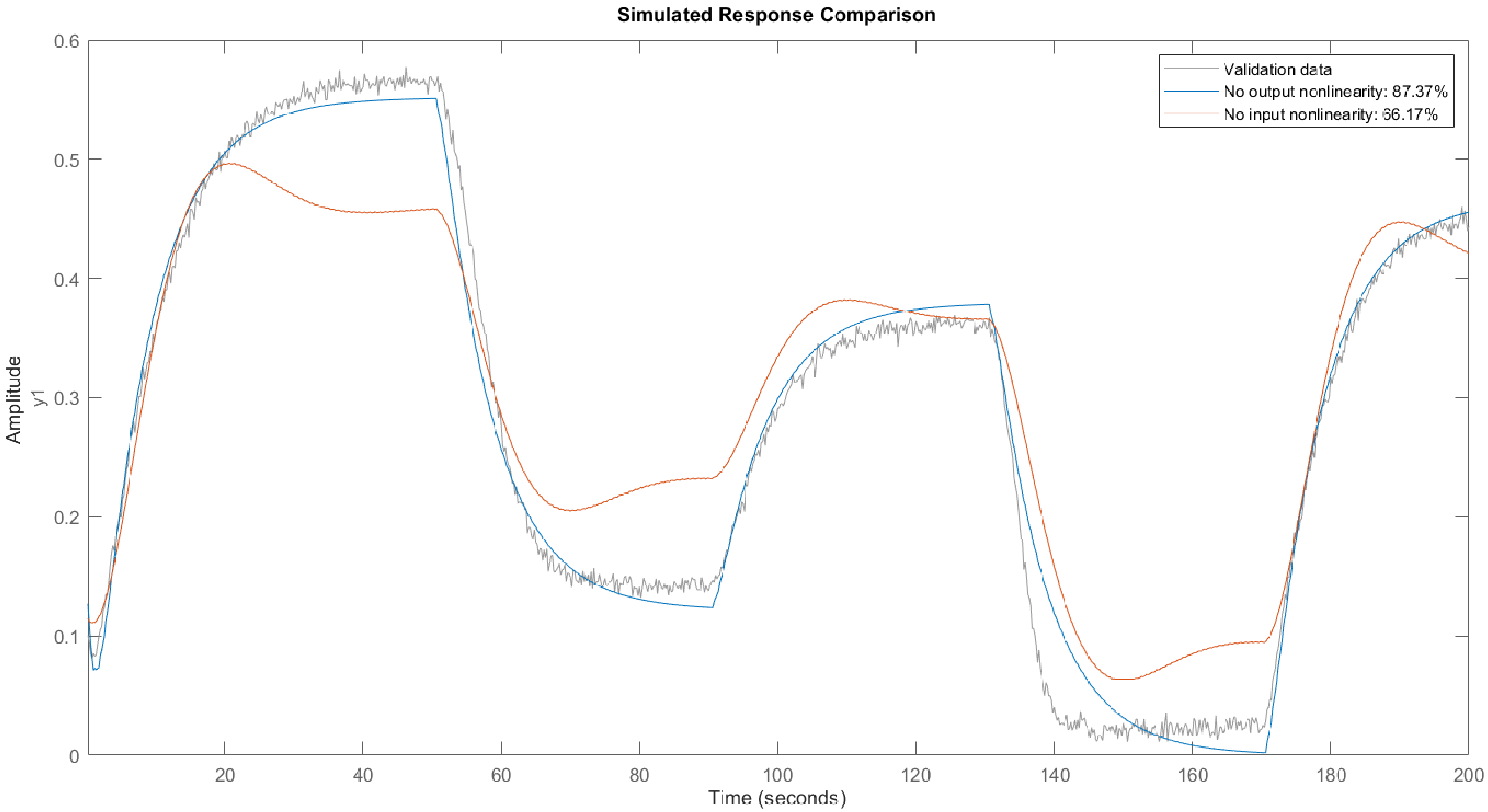




It looks like the method uses past output for predictions

-> **Lessons learned:** have insight into your own code

This is how the method could perform..



Our provided notebook

- NARX method works
- Still some errors in the two-stage optimal HW-method..

Python notebook:
https://colab.research.google.com/drive/1nZfpeLsaYgD7Tjnc3Wf794Z5b_G_pZGm#scrollTo=luIdu73RkWyG

Almost any nonlinear system can be approximated as a Hammerstein-Wiener system!

Pros

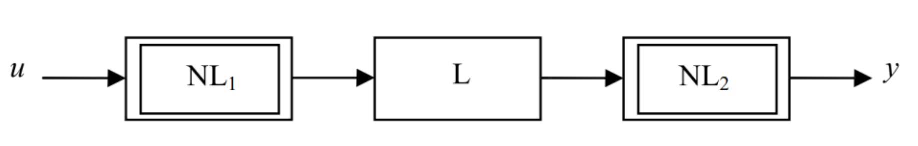
- Can account for nonlinear *actuators* (Hammerstein)
- Can account for nonlinear *sensors* (Wiener)
- Wide range of applications

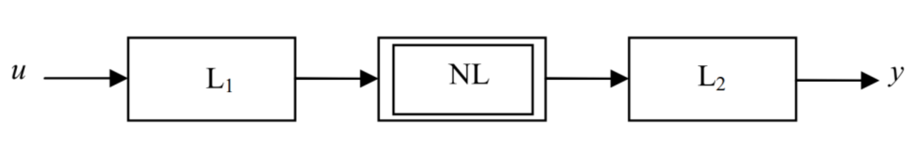
Cons

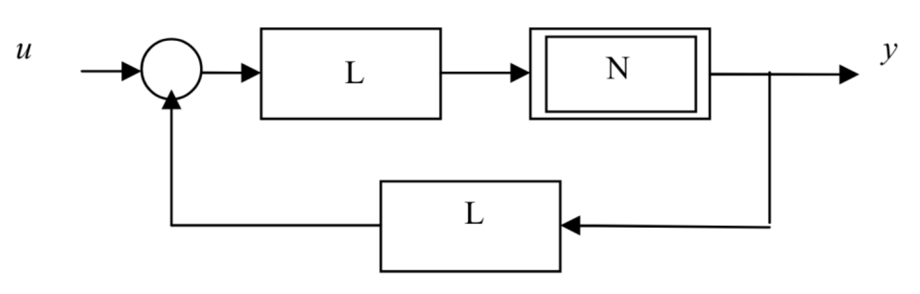
- Figuring out the nonlinearities a priori can be hard.
- Approximating nonlinearities without a priori knowledge is also hard.
- Just because it *can* be approximated, doesn't mean it is *practical*.

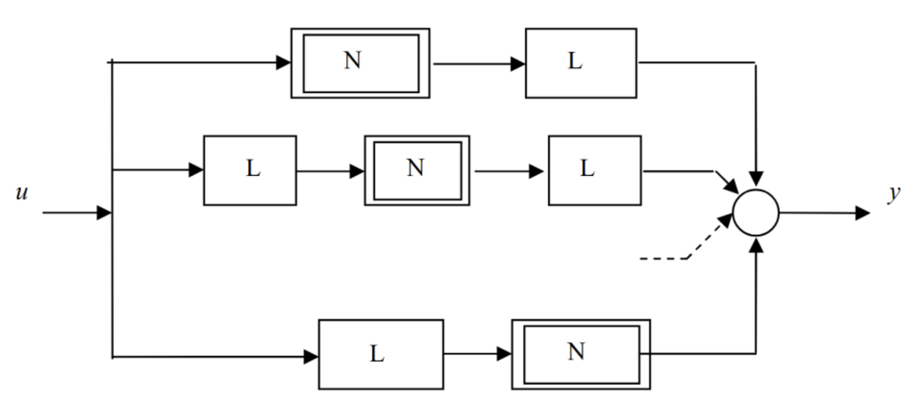
Other models

- Hammerstein-Wiener is *NLN*
- Wiener-Hammerstein is *LNL*
- There are also:
    - Hammerstein: *NL*
    - Wiener: *LN*
- And coupled systems
    - Feedback linearity
    - Feedback nonlinearity
    - Multiple channels
    - ... and more









Summary

- When?
  - Nonlinear actuator/sensor characteristics
  - Complicated timeseries behavior
- Why?
  - *"Almost any nonlinear systems can be modeled by a HW-system"*

- How?
  - Matlab, mostly (`nlwh` and `idnlhw`)
  - Maybe Python (https://sysidentpy.org/ , *2024*)