TUTORIAL 7 - REAL DATA IDENTIFICATION OF AN ACTUAL RADIATOR-ROOM SYSTEM AIMED TO VIRTUAL SENSOR DESIGN

Stefano Malan & **Cosimo Greco**
Dip. di Automatica e Informatica
Politecnico di Torino

Outline

- Introduction
- Plant description
- Models choice: SSV models; I/O models
- Experimental setup and data
- Estimation method: Operational aspects; Software tools
- Identification and validation results
- Virtual sensors: State space solution; Input output solution
- Matlab scripts

INTRODUCTION

Introduction

Many reasons drives to energy management and saving in buildings

- Reduce pollution (respect protocols target)
- Reduce operating costs
- Maximize user comfort
- Fairly rearrange cost distribution on actual consumption: right accounting
- Give a sense of responsibility to final user
- Create and maximize final user's energy awareness
- ...

Introduction

Reduce monitoring and right accounting costs and complexity:

- use virtual or soft sensors, based on a identified model, to substitute actual sensors

First step to design a Virtual Sensor:

- identify a suitable mathematical model of the system able to provide, as an output, the signal to be measured: open loop VS

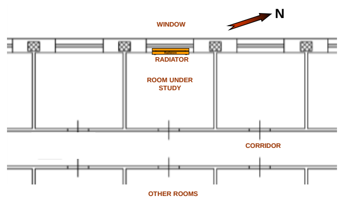
Second step to design a Virtual Sensor:

- use Control Theory to design a closed loop VS

PLANT DESCRIPTION

Plant description

- University office room, 12 square meters
- Heating system: radiator connected to central heating unit
- Input water temperature: set by the central heating unit
- Water flow: can be regulated by means of a valve



Plant description

Research aim:

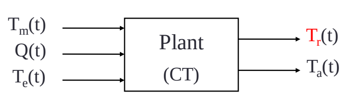
- set up a methodology for micro-accounting heating energy consumption
- reduce the need of physical sensors → virtual sensor
- 3 measurements needed to compute Heating Power:
  1. water flow Q
  2. water input temperature T_m
  3. water output temperature T_r

$$P_h = Q\rho c(T_m - T_r)$$

- tutorial aim:
  identify a model of the overall radiator-room system to suitably design a temperature T_r virtual sensor

Plant description

Overall continuous time (CT) system definition



Inputs:

- radiator input water temperature T_m
- heating water flow Q
- external environment temperature T_e

Outputs:

- radiator output water temperature T_r
- room temperature T_a

Plant description

Further inputs:

- surrounding rooms temperatures → negligible effects

State variable model:

- physical states: homogeneous bodies temperatures
- state number: model order
- spatial discretization: great number of states
- equations: heat exchanges equilibriums, physical parameters

⇓

Simple models with few constraints

Model order as a "black-box" identification parameter

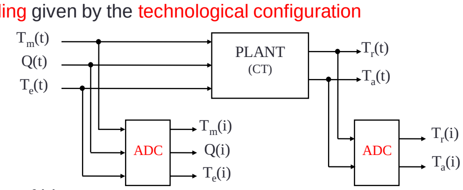
MODELS CHOICE

SSV models

I/O models

Models choice

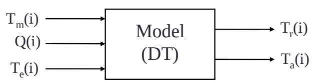
The continuous time (CT) dynamic system is represented by a discrete time (DT) model due to inputs and outputs sampling given by the technological configuration



ADC:
number of bits N
sampling interval T_s
sampling instant $t_i = iT_s$, with i the DT variable

Models choice

DT dynamic model representing the CT dynamic plant



Model characteristic:
- linear
- lumped parameters

Model classes:
- SSV : state space variable
- I/O : input-output

Models choice: SSV models

Time domain model

$$\begin{cases} x(i+1) = Ax(i) + Bu(i) \\ y(i) = Cx(i) + Du(i) \\ \quad \text{given the initial condition } x(0) \end{cases}$$

Z transform domain model

$$\begin{cases} zx(z) = Ax(z) + Bu(z) + zx(0) \\ y(z) = Cx(z) + Du(z) \end{cases}$$

In stationary conditions, for an asymptotically stable system, the initial condition is no more taken into account

Models choice: I/O models

Z transform domain model

$$y(z) = G(z)u(z) + y_0(z)$$

The elements of $G(z)$ are transfer functions all having the same denominator $D_c(z)$ of degree n

The elements of $y_0(z)$ are rational proper functions all having the same denominator $D_c(z)$ of degree n

In stationary conditions, for an asymptotically stable system, the initial condition is no more taken into account

$$y(z) = G(z)u(z)$$

Models choice: I/O models

Z transform domain model explicit form

$$\begin{bmatrix} y_1(z) \\ y_2(z) \\ \vdots \\ y_{ny}(z) \end{bmatrix} = \frac{1}{D_c(z)} \begin{bmatrix} N_{1,1}(z) & N_{1,2}(z) & \cdots & N_{1,nu}(z) \\ N_{2,1}(z) & N_{2,2}(z) & \cdots & N_{2,nu}(z) \\ \vdots & \vdots & \cdots & \vdots \\ N_{ny,1}(z) & N_{ny,2}(z) & \cdots & N_{ny,nu}(z) \end{bmatrix} \cdot \begin{bmatrix} u_1(z) \\ u_2(z) \\ \vdots \\ u_{nu}(z) \end{bmatrix} + \frac{1}{D_c(z)} \cdot \begin{bmatrix} N_{01}(z) \\ N_{02}(z) \\ \vdots \\ N_{0ny}(z) \end{bmatrix}$$

Consider a single input – single output model

$$y_j(z) = G_{jk}(z)u_k(z) = \frac{N_{jk}(z)}{D_c(z)} u_k(z) =$$

$$= \frac{b_{jk,n} z^n + b_{jk,n-1} z^{n-1} + \ldots + b_{jk,1} z + b_{jk,0}}{z^n - a_{n-1} z^{n-1} - \ldots - a_1 z - a_0} u_k(z)$$

Models choice: I/O models

The derived difference equation is

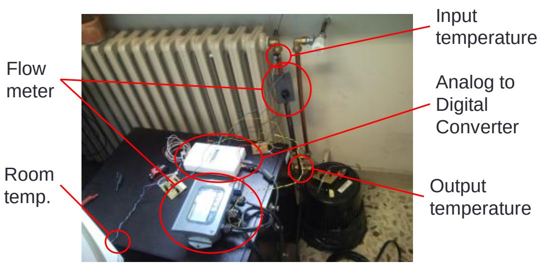
$$y_j(i) = a_{n-1} y_j(i-1) + a_{n-2} y_j(i-2) + \ldots + a_0 y_j(i-n) +$$
$$+ b_{jk,n} u_k(i) + b_{jk,n-1} u_k(i-1) + \ldots + b_{jk,0} u_k(i-n)$$

The model for a multi input – single output therefore is

$$\begin{aligned} y_j(i) &= a_{n-1} y_j(i-1) + a_{n-2} y_j(i-2) + \ldots + a_0 y_j(i-n) + \\ &+ b_{j1,n} u_1(i) + b_{j1,n-1} u_1(i-1) + \ldots + b_{j1,0} u_1(i-n) + \\ &+ b_{j2,n} u_2(i) + b_{j2,n-1} u_2(i-1) + \ldots + b_{j2,0} u_2(i-n) + \\ &+ b_{j3,n} u_3(i) + b_{j3,n-1} u_3(i-1) + \ldots + b_{j3,0} u_3(i-n) + \\ &\vdots \\ &+ b_{jn_u,n} u_{n_u}(i) + b_{jn_u,n-1} u_{n_u}(i-1) + \ldots + b_{jn_u,0} u_{n_u}(i-n) \end{aligned}$$

EXPERIMENTAL SETUP AND DATA

Experimental setup and data



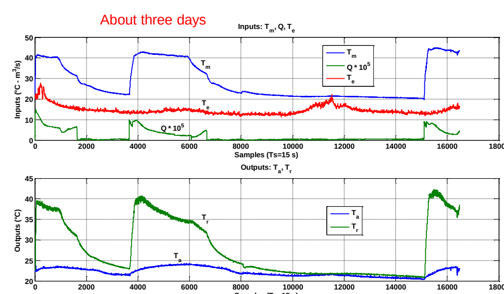
Experimental setup and data

- Temperature sensor National Semiconductor LM35
  - scale 10 mV/°C, range −40 °C ÷ +110 °C, accuracy ± 0.2 °C
  - heating water input and output copper pipes (error < 0.07 °C)
  - room
  - external
- Water flow transducer Dynasonic TFX Ultra
  - ultrasonic, not invasive
  - output current 4 ÷ 20 mA → water velocity 0 ÷ $V_{w,max}$
  - corresponding acquired voltage given by a 100 Ω resistor
  - water flow Q obtained multiplying by 49 mm² pipe section
- Acquisition board National Instruments USB-6211
  - range ± 10 V, conversion accuracy 16 bit, LabView interface

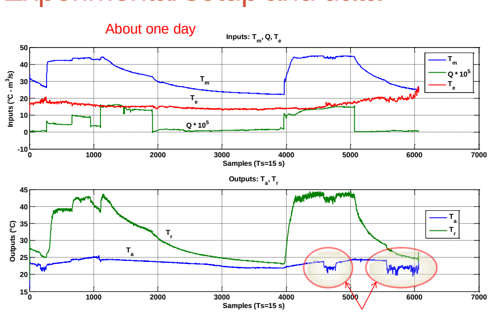
Experimental setup and data

- Acquisition period: lasting a few days, in the last decade of March, in Torino, Piemonte, north west of Italy
- Command: water flow regulated by a manual valve
- Disturbances:
  - heating water input temperature (centrally imposed)
  - external environment temperature, opening and closing windows
- Sampling interval: 1 or 5 seconds, subsequently decimated to 15 seconds

Experimental setup and data



Experimental setup and data



ESTIMATION METHOD

Operational aspects
Software tools

Estimation method

SISO system, time domain

$$y(i) = a_{n-1}y(i-1) + a_{n-2}y(i-2) + \ldots + a_0 y(i-n) + \\ + b_{n-1}u(i-1) + b_{n-2}u(i-2) + \ldots + b_0 u(i-n)$$

in matrix form

$$y(i) = \varphi'_{i-1}\theta$$

previous input and output samples

$$\varphi'_{i-1} \doteq [y(i-1) \;\; y(i-2) \;\; \ldots \;\; y(i-n) \;\; u(i-1) \;\; u(i-2) \;\; \ldots \;\; u(i-n)]$$

$m \doteq 2n$ parameters to be identified

$$\theta' \doteq [a_{n-1} \;\; a_{n-2} \;\; \ldots \;\; a_0 \;\; b_{n-1} \;\; b_{n-2} \;\; \ldots \;\; b_0]$$

Estimation method

Hp: time invariant parameters

Rewrite $y(i) = \varphi'_{i-1}\theta$ for $y(i-1), \ldots, y(i-N+1)$ and obtain

$$M_{i-1}\theta = y_i$$

N equations in m unknown

$$\underbrace{\begin{bmatrix} \varphi'_{i-N} \\ \vdots \\ \varphi'_{i-2} \\ \varphi'_{i-1} \end{bmatrix}}_{\doteq M_{i-1}} \cdot \theta = \underbrace{\begin{bmatrix} y(i-N+1) \\ \vdots \\ y(i-1) \\ y(i) \end{bmatrix}}_{\doteq y_i}$$

Note: the above equation error term was skipped

Estimation method

If $N \gg m$ solve as a least square optimization

$$\min_{\theta}\left\{\left\| y_i - M_{i-1}\theta \right\|^2\right\} \longrightarrow \hat{\theta} = (M'_{i-1}M_{i-1})^{-1}M'_{i-1}y_i$$

the solution $\hat{\theta}$ exists if

$$\operatorname{rank}\{M_{i-1}\} = m$$

this implies particular constraints on the experimental conditions under which the I/O samples were acquired

Estimation method: operational aspects

a. MIMO systems: same equation $M_{i-1}\theta = y_i$ but different construction of matrices;

for example for a second order, 2 inputs, 2 outputs system

$$M_{i-1} = \begin{bmatrix} y_1(i-N) & y_1(i-N-1) & u_1(i-N) & u_1(i-N-1) & u_2(i-N) & u_2(i-N-1) & 0 & 0 & 0 & 0 \\ & & & & & \vdots & & & & \\ y_1(i-1) & y_1(i-2) & u_1(i-1) & u_1(i-2) & u_2(i-1) & u_2(i-2) & 0 & 0 & 0 & 0 \\ y_2(i-N) & y_2(i-N-1) & 0 & 0 & 0 & 0 & u_1(i-N) & u_1(i-N-1) & u_2(i-N) & u_2(i-N-1) \\ & & & & & \vdots & & & & \\ y_2(i-1) & y_2(i-2) & 0 & 0 & 0 & 0 & u_1(i-1) & u_1(i-2) & u_2(i-1) & u_2(i-2) \end{bmatrix}$$

$$\vartheta = [a_1 \;\; a_0 \;\; b_{11,1} \;\; b_{11,0} \;\; b_{12,1} \;\; b_{12,0} \;\; b_{21,1} \;\; b_{21,0} \;\; b_{22,1} \;\; b_{22,0}]'$$

$$y_i = \begin{bmatrix} y_1(i-N+1) \\ \vdots \\ y_1(i) \\ y_2(i-N+1) \\ \vdots \\ y_2(i) \end{bmatrix}$$

Estimation method: operational aspects

b. SSV models: it is possible to set up the least-squares estimation problem or use Matlab routines

c. Offset in the I/O variables: the system dynamics usually develop in the neighbourhood of a constant value, then

1. detrend inputs and outputs; not always possible
2. use an affine model and estimate the offset $\bar{Y}$

I/O model, SISO case

$$y(i) = a_{n-1}y(i-1) + a_{n-2}y(i-2) + \ldots + a_0 y(i-n) + \\ + b_{n-1}u(i-1) + b_{n-2}u(i-2) + \ldots + b_0 u(i-n) + \bar{Y}$$

Estimation method: operational aspects

c. Offset in the I/O variables

Same equation $y(i) = \varphi'_{i-1}\theta$ but different construction of vectors

$$\theta' = [a_{n-1} \;\; a_{n-2} \;\; \ldots \;\; a_0 \;\; b_{n-1} \;\; b_{n-2} \;\; \ldots \;\; b_0 \;\; \bar{Y}]$$

$$\varphi'_{i-1} = [y(i-1) \;\; y(i-2) \;\; \ldots \;\; y(i-n) \;\; u(i-1) \;\; u(i-2) \;\; \ldots \;\; u(i-n) \;\; 1]$$

a constant value input equal to 1 surely makes the identification problem numerically ill conditioned

it is necessary to "perturb" it with some random noise

Estimation method: operational aspects

c. Offset in the I/O variables

SSV models

$$\begin{cases} x(i+1) = Ax(i) + Bu(i) + \bar{X} \\ y(i) = Cx(i) + Du(i) + \bar{Y} \end{cases}$$

$\bar{X}$ and $\bar{Y}$ are constant to be identified in the "augmented" system

$$\begin{cases} x(i+1) = Ax(i) + \begin{bmatrix} B & \bar{X} \end{bmatrix} \begin{bmatrix} u(i) \\ 1 \end{bmatrix} \\ y(i) = Cx(i) + \begin{bmatrix} D & \bar{Y} \end{bmatrix} \begin{bmatrix} u(i) \\ 1 \end{bmatrix} \end{cases}$$

Estimation method: operational aspects

d. Acquired data absolute values / numerical ill conditioning

Numerical values given as input to the identification algorithm must be about of the same magnitude to prevent numerical ill conditioning, to satisfy algorithm requirements

Often these are not satisfied not because of a requirement lack structurally related to the considered problem, but only because of a numerical ill conditioning that can be overcome by a suitable numerical values scaling

For example the values of the water flow Q is multiplied by a factor 10^5

Estimation method: software tools

- *Matlab* environment plus the Identification Toolbox
- *Scilab* environment
- *C* language

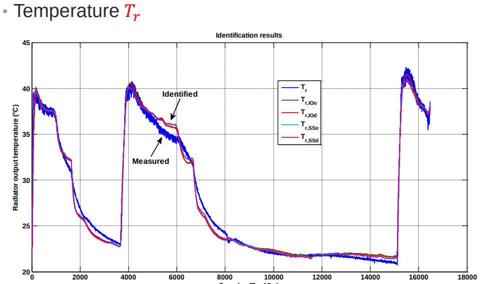
IDENTIFICATION AND VALIDATION RESULTS

Identification results

- Two data sets: first for identification, second for validation
- Four typologies of models
  - IOd I/O models, without offset, identified from detrendized data
  - IOo I/O models, with offset, identified from not detrendized data
  - SSd SSV models, without offset, identified from detrendized data
  - SSo SSV models, with offset identified, from not detrendized data
- Other combinations are possible
- Model order identified by trial and error
  - low order
  - second order suitable to describe the system
  - same number of zeroes

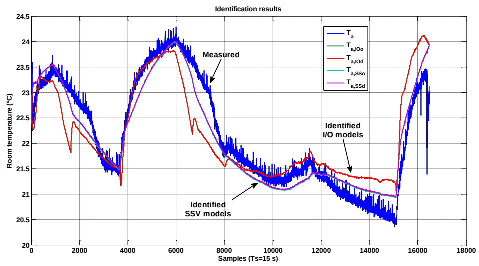
Identification results

- Temperature T_r



Identification results

- Temperature T_a



Identification results

- Numerical evaluation: standard deviation

$$\sigma = \sqrt{\frac{\sum_{i=1}^{l}\left(y(i) - y_s(i)\right)^2}{l}}$$

Model	IOo	IOd	SSo	SSd
σ_r for T_r	0.7308	0.7324	0.6644	0.6667
σ_a for T_a	0.5341	0.5341	0.3035	0.3016

Identification results

- Transfer matrix

$$\begin{bmatrix} T_r(z) \\ T_a(z) \end{bmatrix} = \begin{bmatrix} G_{11}(z) & G_{12}(z) & G_{13}(z) \\ G_{21}(z) & G_{22}(z) & G_{23}(z) \end{bmatrix} \begin{bmatrix} T_m(z) \\ 10^5 \cdot Q(z) \\ T_e(z) \end{bmatrix}$$

$$\begin{bmatrix} T_r(z) \\ T_a(z) \end{bmatrix} = \begin{bmatrix} \frac{N_{11}(z)}{D_C(z)} & \frac{N_{12}(z)}{D_C(z)} & \frac{N_{13}(z)}{D_C(z)} \\ \frac{N_{21}(z)}{D_C(z)} & \frac{N_{22}(z)}{D_C(z)} & \frac{N_{23}(z)}{D_C(z)} \end{bmatrix} \begin{bmatrix} T_m(z) \\ 10^5 \cdot Q(z) \\ T_e(z) \end{bmatrix}$$

Identification results

- IOo model

$N_{11}(z) = -0.010176\,(z-1.729)\,(z+0.5303);\quad N_{21}(z) = -0.00018483\,(z-103.2)\,(z-0.8742)$
$N_{12}(z) = 0.0027284\,(z^2 + 1.566z + 1.162);\quad N_{22}(z) = 0.0024779\,(z-1.793)\,(z+0.6046)$
$N_{13}(z) = -0.013828\,(z-0.8036)\,(z-0.3412);\quad N_{23}(z) = -0.012715\,(z-1.125)\,(z-0.2814)$
$D_C(z) = (z-0.9845)\,(z+0.126)$

- IOd model

$N_{11}(z) = -0.0099935\,(z-1.745)\,(z+0.5253);\quad N_{21}(z) = -0.00018402\,(z-103.7)\,(z-0.8742)$
$N_{12}(z) = 0.0027247\,(z^2 + 1.558z + 1.173);\quad N_{22}(z) = 0.002478\,(z-1.793)\,(z+0.6044)$
$N_{13}(z) = -0.013841\,(z-0.8043)\,(z-0.3396);\quad N_{23}(z) = -0.012715\,(z-1.125)\,(z-0.2815)$
$D_C(z) = (z-0.9845)\,(z+0.1259)$

Identification results

- SSo model

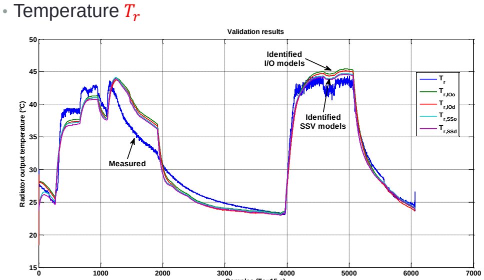
$N_{11}(z) = -0.030646\,(z-1.411)\,(z-0.9985);\quad N_{21}(z) = 0.017656\,(z-0.9942)\,(z-0.9521)$
$N_{12}(z) = 0.0034834\,(z+2.341)\,(z-0.9989);\quad N_{22}(z) = 0.0020483\,(z-1.033)\,(z-0.9221)$
$N_{13}(z) = -0.015802\,(z-0.9992)\,(z-0.8513);\quad N_{23}(z) = -0.0092313\,(z-1.011)\,(z-0.9759)$
$D_C(z) = (z-0.9986)\,(z-0.9798)$

- SSd model

$N_{11}(z) = -0.030404\,(z-1.414)\,(z-0.9985);\quad N_{21}(z) = 0.017674\,(z-0.9942)\,(z-0.9522)$
$N_{12}(z) = 0.0034558\,(z+2.366)\,(z-0.9989);\quad N_{22}(z) = 0.0020333\,(z-1.033)\,(z-0.922)$
$N_{13}(z) = -0.015828\,(z-0.9992)\,(z-0.8516);\quad N_{23}(z) = -0.0092488\,(z-1.011)\,(z-0.9759)$
$D_C(z) = (z-0.9986)\,(z-0.9798)$

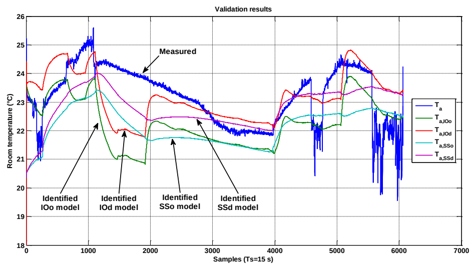
Validation results

- Temperature T_r



Validation results

- Temperature T_a



VIRTUAL SENSORS

State space solution
Input output solution

Virtual sensors

- Virtual Sensors are mathematical models able to give, in real time, the numerical value of a physical signal, that is not directly measured by an actual sensor
- Any previous model: open loop VS
- Closed loop VS to correct differences between the model and the actual system, to make the virtual measure convergent to the unknown real one
- The correction is obtained by suitably feedback the comparison of the simulated and measured values of the second output T_a

Virtual sensors

Generalized dual-input/dual-output (DIDO) model

- Two inputs u_1 and u_2
- One plant output y_{s1} to be estimated
- One plant output y_{s2} measured
- Two model outputs y_1 and y_2

Working assumptions

- u_1, u_2 and y_{s2} are measurable
- y_{s1} is not measurable, it is to be reconstructed or estimated

Virtual sensors: state space solution

VS as classical Luenberger observer

$$\begin{cases} \hat{x}(i+1) = A\hat{x}(i) + B\begin{bmatrix} u_1(i) \\ u_2(i) \end{bmatrix} + L\left[y_{s2}(i) - C_2\hat{x}(i)\right] \\ \hat{y}_1(i) = C_1\hat{x}(i) \end{cases}$$

gain L are selected to obtain a closed loop asymptotically stable system with suitable time constants

Estimation error $e_2 \doteq y_{s2} - \hat{y}_2 = y_{s2} - C_2\hat{x}$

gain L are chosen in order to ensure $\hat{y}_2 \to y_{s2}$ and to guarantee also $\hat{y}_1 \to y_{s1}$

Virtual sensors: state space solution

Closed loop VS

$$\begin{cases} \hat{x}(i+1) = (A - LC_2)\,\hat{x}(i) + \begin{bmatrix} B & L \end{bmatrix}\begin{bmatrix} u_1(i) \\ u_2(i) \\ y_{s2}(i) \end{bmatrix} \\ \hat{y}_1(i) = C_1\,\hat{x}(i) \end{cases}$$

gain L are chosen in order to ensure that the closed loop matrix $(A - LC_2)$ is asymptotically stable

Note: the measured plant output y_{s2} becomes one of the Virtual Sensor inputs

Virtual sensors: input output solution

Input output model $G(z)$ identified from acquired data

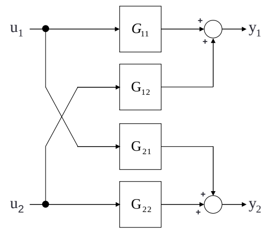
$$Y(z) = G(z)U(z)$$

where

$$Y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

$$U = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$G = \begin{bmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{bmatrix}$$



Virtual sensors: input output solution

Input output model $G(z)$

tf's G_{ij} have same denominator, different numerators

$$G_{ij} = \frac{N_{ij}}{D}, \ \forall i \text{ and } \forall j \ \rightarrow G = \frac{1}{D}\begin{bmatrix} N_{11} & N_{12} \\ N_{21} & N_{22} \end{bmatrix} = \frac{N_G}{D}$$

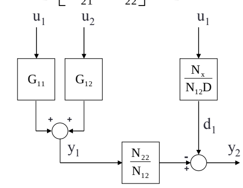
it can be computed

$$y_2 = \frac{N_{22}}{N_{12}} y_1 - \frac{N_x}{N_{12} D} u_1$$

where

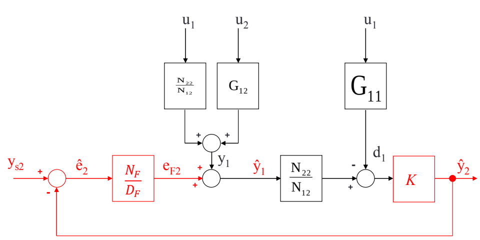
$$N_x = \det(N_G)$$

$$y_1 = G_{11} u_1 + G_{12} u_2$$



Virtual sensors: input output solution

Closed loop VS: filter N_F / D_F and gain K are chosen in order to ensure $\hat{y}_2 \rightarrow y_{s2}$ and to guarantee also $\hat{y}_1 \rightarrow y_{s1}$



MATLAB SCRIPTS

Matlab scripts

Identification and Validation

- IdE_ordN_zerM_inpP_outQ Main
  - ARXdAH I/O Identification, detrendized data
  - ARXoAH I/O Identification, affine model
  - ICC Initial Conditions Computation
  - MaFiMoRe Make Figure More Readable
  - n4sid (Identification Toolbox) SSV Identification

Virtual Sensor Design (state space solution)

- ViSeDe_L linear model
- ViSeDe_LY affine model

Note: if interested in getting the scripts, contact the authors

THANK YOU FOR YOUR KIND ATTENTION

Questions, please …