16/04

GENERAL TOPIC OF MIDA COURSE :

- collect digitally data from real systems
- build BLACK-BOX (GRAY-BOX) models from data, with emphasis on dynamical systems and control/automation-oriented applications

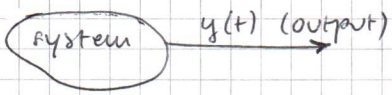
Purpose of this modelling :

- prediction
- software-sensing
- modelling for control design

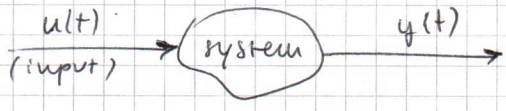
(General area: machine-learning but with the focus on modelling for control design)

SUPER SUMMARY OF MIDA 1 :

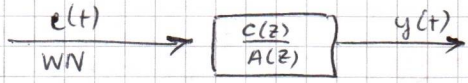
Focus :

- Time series : 

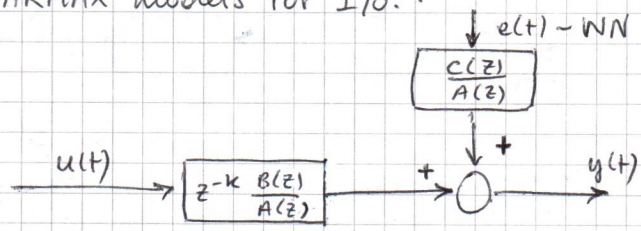
- Input/Output system (I/O) :



Model classes used in MIDA 1 :

- ARMA models for T.S. 

- ARMAX models for I/O. :



these are the models $M(\theta)$ where θ is the parameter vector (θ is the collection of coefficients of $A(z)$, $B(z)$, $C(z)$)

A parametric identification method has been used.
A performance index is defined :

$$J(\theta) = \frac{1}{N} \sum_{i=1}^{N} (y(t) - \hat{y}(t|t-1, \theta))^2$$

variance of the prediction error made by the model $\Longrightarrow$ PREDICTION ERROR METHOD (P.E.M.)

$$\longrightarrow \hat{\theta}_N = \arg\min_{\theta} J(\theta)$$

MIDA 2 : focus on I/O system (more close to real applications than time series)

Chapter 1 : NON-parametric (direct/constructive) black box identification of I/O systems using state-space models

Chapter 2 : parametric identification of black box I/O systems with a frequency domain approach

Chapter 3 : Kalman filter for software sensing using feedback on white box models

Chapter 4 : black box methods for software sensing without feedback

Chapter 5 : gray-box system identification :
- using kalman filter
- using "simulation-error methods" (S.E.M.)

Chapter 6 : minimum variance control (M.V.C.)
⟹ design of optimal feedback controllers using the theoretical background of the MIDA course

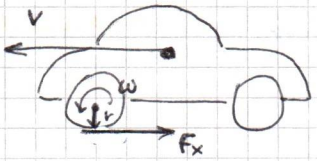
Appendix : recursive ("on-line") implementation of algorithms for system identification

Remark : we'll see some application examples / MATLAB / simulink numerical examples

---

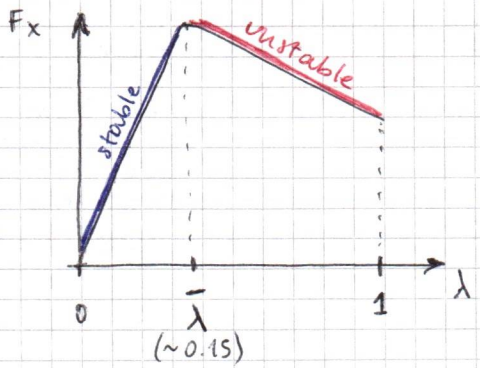
Motivation example for the course :

Consider a car :

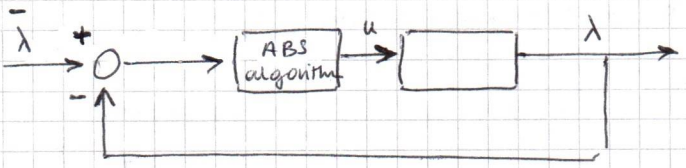


SLIP of the wheel := $\lambda = \frac{v - \omega r}{v}$, $0 \leq \lambda \leq 1$

($\lambda = 1 \Rightarrow$ locked wheel
$\lambda = 0 \rightarrow$ free rolling wheel)

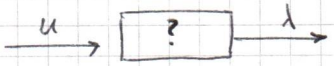


⟹ ABS control problem :
(anti-lock braking system)



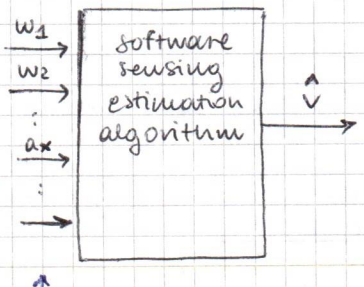
⟹ the ABS problem is to design a control loop s.t. λ is designed to be equal to $\bar{\lambda}$

Sub-problems :

1. model the system : 

2. software estimation of λ : $\lambda = \frac{(v) - \omega r}{(v)}$ → not measurable (GPS or optical sensories)

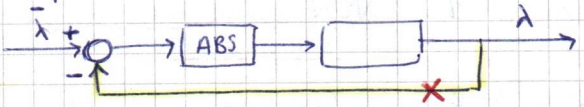
so we have to make software sensing : (estimation of v ($\hat{v}$), then λ ($\hat{\lambda}$))



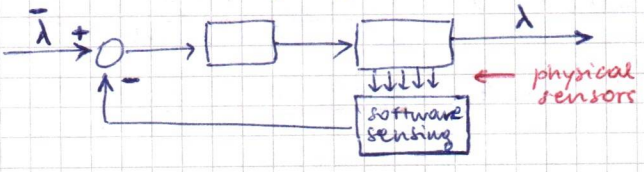
⟹ $\hat{\lambda}$

physical sensors

in practice we can't do this measurement :

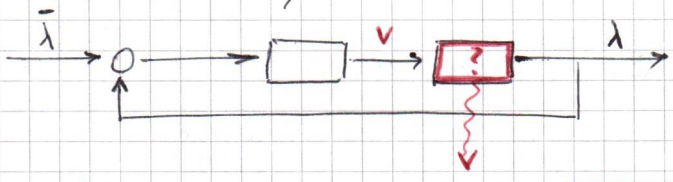


so we take physical sensors and we build software sensing algorithm :

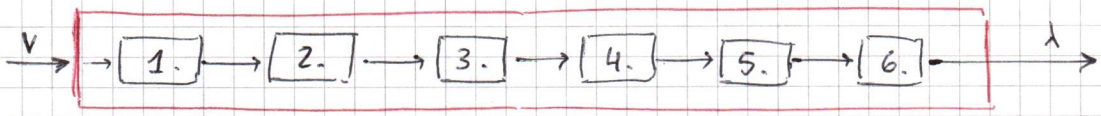


3. design the ABS control algorithm

Why black box modelling?
Let's consider the system:



V = voltage driving an electric motor of a brake-by wire actuator

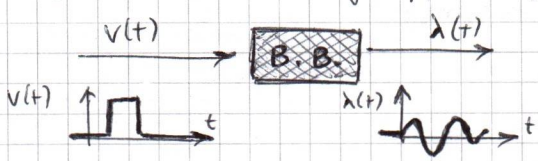


1. current dynamics of electric motor (electric)
2. position dynamics of BBW actuator (mechanical) (BBW = brake by wire actuator)
3. dynamics of the hydraulic circuit of BRAKE system (hydraulic)
4. tire dynamics (mechanical)
5. wheel rotational dynamics (mechanical)
6. vehicle full dynamics (mechanical)

$\Longrightarrow$ multi-domain system (electric, mechanical, hydraulic)

We have two options:

- white box (physical) modelling $\Longrightarrow$ write equations from "first principle"
- black box modelling: experiment $\Longrightarrow$ collect data $\Longrightarrow$ build model (machine learning approach)



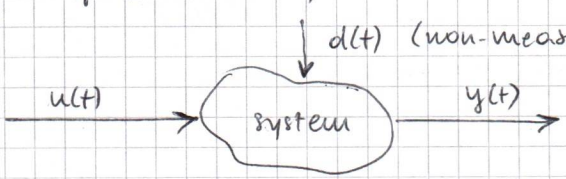
using just I/O measured data we can learn a mathematical model of the I/O behaviour of the system

---

CHAPTER 1

1

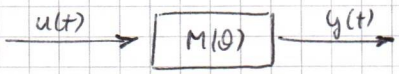
(Black-box non parametric system identification of I/O systems using state-space models)



Data: $\{u(1), u(2), \ldots, u(N)\}$, $\{y(1), y(2), \ldots, y(N)\}$

Recall of the general path of a **parametric** identification method:

1. collect data: $\{u(1), u(2), \ldots, u(N)\}$, $\{y(1), y(2), \ldots, y(N)\}$
2. select (A PRIORI) of a class / family of parametric models: $M(\theta)$, where θ is the parameter vector



3. Select (A PRIORI) a performance index (it gives an order to the quality of models). In general: $J(\theta): \mathbb{R}^{n_\theta} \to \mathbb{R}^+$ where n_θ = size of the parameter vector θ.
With the performance index we can say:
$M(\theta_1)$ is better than $M(\theta_2)$ if $J(\theta_1) < J(\theta_2)$
(we can give an order)

Example: P.E.M., $J(\theta) = \frac{1}{N}\sum_{t=1}^{N}\left(y(t) - \hat{y}(t|t-1,\theta)\right)^2$

$y(t)$: true output; $\hat{y}(t|t-1,\theta)$: 1-step predictor of the model

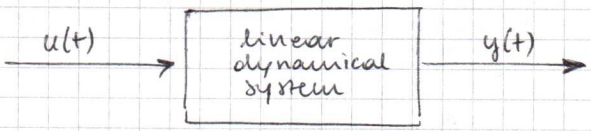
variance of the error made by the predictor based on model $M(\theta)$

4. Optimization step: minimize $J(\theta)$ w.r.t. θ:

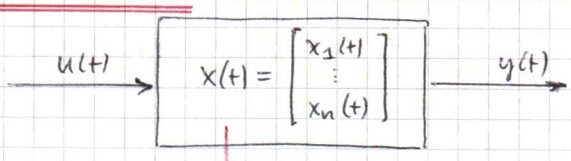
$$\hat{\theta}_N = \arg\min_{\theta} J(\theta) \implies \text{optimal model } M(\hat{\theta}_N)$$

In this chapter we are presenting a totally different system identification approach (which is not parametric):
- NO a priori model class selection
- NO performance index definition
- NO optimization task

PROLOGUE (RECALL): 3 main representations of a discrete-time, linear, dynamical system



Representation 1: STATE-SPACE REPRESENTATION



internal states of the system
(n = ORDER of the system)

$$\implies \begin{cases} x(t+1) = F\,x(t) + G\,u(t) & \leftarrow \text{"state equations"} \\ y(t) = H\,x(t) + D\,u(t) & \leftarrow \text{"output equations"} \end{cases}$$

$F = [\,n \times n\,]$:= state matrix $G = [\,n \times 1\,]$:= input matrix

$H = [\,1 \times n\,]$:= output matrix $D = [\,1 \times 1\,]$:= I/O matrix

usually is ZERO for STRICTLY-PROPER systems

we are assuming 1 input and 1 output (can be extended for multiple inputs and/or outputs)

1 input, 1 output := SISO

Note: S.S. representation is not unique for a system

$$F \longrightarrow F_1 = TFT^{-1}$$
$$G \longrightarrow G_1 = TG$$
$$H \longrightarrow H_1 = HT^{-1}$$
$$D \longrightarrow D_1 = D$$

$\forall T$ invertible square matrix

represent the same system

$\{F, G, H, D\}$ and $\{F_1, G_1, H_1, D_1\}$ are equivalent.

Example:
$$\begin{cases} x_1(t+1) = \frac{1}{2} x_1(t) + 2u(t) \\ x_2(t+1) = x_1(t) + 2x_2(t) + u(t) \\ y(t) = \frac{1}{4} x_1(t) + \frac{1}{2} x_2(t) \end{cases}$$

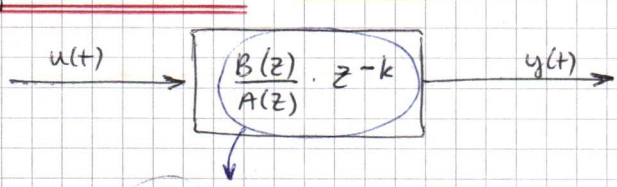
$n = 2$: $x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$

1 input : $u(t)$
1 output : $y(t)$

$$F = \begin{bmatrix} -1/2 & 0 \\ 1 & 2 \end{bmatrix} \quad G = \begin{bmatrix} 2 \\ 1 \end{bmatrix} \quad H = \begin{bmatrix} \frac{1}{4} & \frac{1}{2} \end{bmatrix} \quad D = 0$$

Representation 2 : TRANSFER FUNCTION REPRESENTATION (I/O repr.)

because $\nexists$ internal states, everything is referred to I/O



$$W(z) = \frac{B(z)}{A(z)} z^{-k} = \frac{b_0 + b_1 z^{-1} + \dots + b_p z^{-p}}{a_0 + a_1 z^{-1} + \dots + a_n z^{-n}}$$

z^{-k} := PURE DELAY

n := ORDER of the system (:= number OF poles)

$W(z)$:= DIGITAL FILTER (is a rational function of the z operator)

It's very easy to move from transfer function representation to a time domain description of the system:

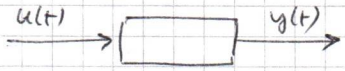
Example: $y(t) = \underbrace{\left(\frac{1 + \frac{1}{2} z^{-1}}{2 + \frac{1}{3} z^{-1} + \frac{1}{4} z^{-2}} \cdot z^{-1} \right)}_{W(z)} u(t)$

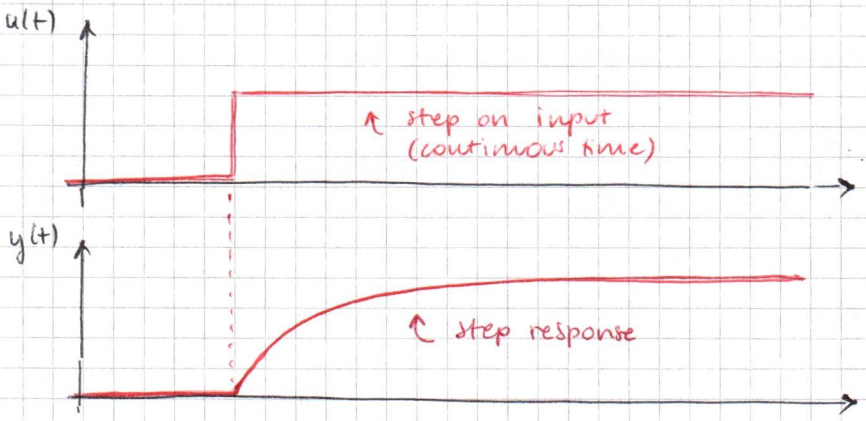
$$\Longrightarrow 2y(t) + \frac{1}{3} y(t-1) + \frac{1}{4} y(t-2) = u(t-1) + \frac{1}{2} u(t-2)$$

$$\Longrightarrow y(t) = \underbrace{-\frac{1}{6} y(t-1) - \frac{1}{8} y(t-2)}_{\text{old values of } y(t) \text{ (recursive or auto-regressive part)}} + \underbrace{\frac{1}{2} u(t-1) + \frac{1}{4} u(t-2)}_{\text{old values of input (Exogenus part)}}$$

difference equation in $u(t)$ and $y(t)$ (NO STATES)

Remark on strictly proper system:
notice that for a strictly proper system the delay $k \geq 1$. (*)

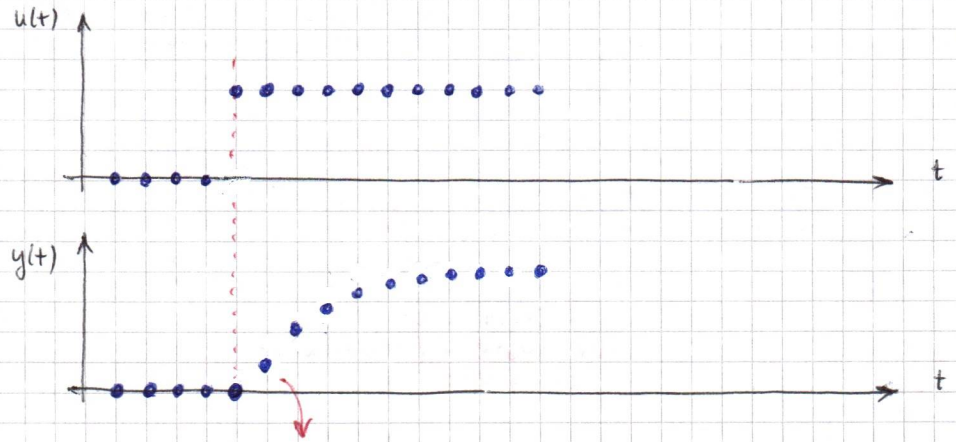




there is no jump in the output
⇒ the system is strictly proper

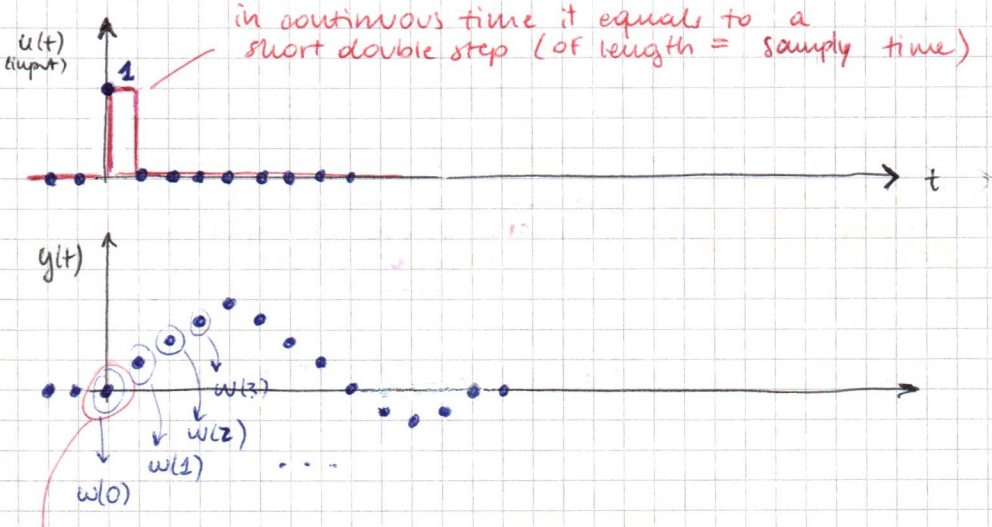
So the strictly properness is something which is physically very intuitive: whenever we make a jump step on a input, the output doesn't have a jump

What happens in the discrete time?



the first non-zero value has at least **1** step delay (that's why the delay $k \geq 1$ (*))
(this is in accord to the fact that $D\,(y(t) = Hx(t) + Du(t))$ is $= 0$)

Representation **3**: CONVOLUTION OF THE INPUT WITH THE **I**NPUT **R**ESPONSE (**IR**)



if the system is strictly proper ⟹ $w(0) = 0$

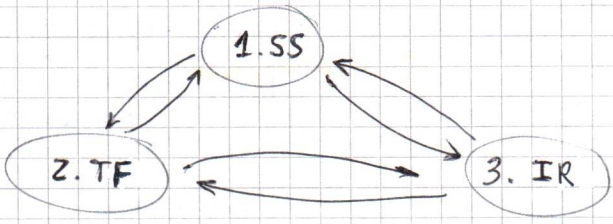
it can be proven that the I/O relationship from $u(t)$ and $y(t)$ can be written as:

$$y(t) = w(0)\,u(t) + w(1)\,u(t-1) + w(2)\,u(t-2) + \dots$$

$$\Longrightarrow \boxed{y(t) = \sum_{k=0}^{+\infty} w(k)\,u(t-k)} \quad := \text{IR representation of the system}$$

↑ convolution of I.R. with the input signal

Representations:
(the system is the same! only the repers. change)



Are we able to move from one representation to one another?

Most classical transformation: 1. ⟶ 2.

1. ⟶ 2.:

$$\begin{cases} x(t+1) = Fx(t) + Cu(t) \\ y(t) = Hx(t) + \cancel{Du(t)} \end{cases}$$

we can use the "z-operator": $z\,x(t) = x(t+1)$, $z^{-1}x(t) = x(t-1)$

$$\Longrightarrow z\,x(t) = Fx(t) + Cu(t)$$

$$\Longrightarrow x(t) = (zI - F)^{-1} G\,u(t)$$

$$\Longrightarrow y(t) = \boxed{H(zI-F)^{-1}G}\,u(t)$$

transfer function from $u(t)$ to $y(t)$:

$$\mathbf{W(z) = H(zI-F)^{-1}G}$$

Example: $F = \begin{bmatrix} 1 & 0 \\ 1/2 & 2 \end{bmatrix}$ ($n = 2$), $G = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ (1 input), $H = \begin{bmatrix} 1 & 0 \end{bmatrix}$ (1 output), $D = [0]$ (strict prop. syst.)

$$W(z) = \begin{bmatrix} 1 & 0 \end{bmatrix} \left(\begin{bmatrix} z & 0 \\ 0 & z \end{bmatrix} - \begin{bmatrix} 1 & 0 \\ 1/2 & 2 \end{bmatrix} \right)^{-1} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{\det[\cdot\cdot]} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

$$\Longrightarrow W(z) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} z-1 & 0 \\ -1/2 & z-2 \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 \end{bmatrix} \frac{1}{(z-1)(z-2)} \begin{bmatrix} z-2 & 0 \\ 1/2 & z-1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$= \frac{1}{(z-1)(z-2)} \begin{bmatrix} z-2 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$= \frac{z-2}{(z-1)(z-2)} = \underbrace{\frac{1}{z-1}}_{\text{positive power}} = \underbrace{\frac{1}{1-z^{-1}}\,z^{-1}}_{\text{negative power}}$$

Remark: $W(z) = \frac{1}{z-1}$ → we have only one pole "visible" from I/O representation (but $n=2$) (because of the cancellation)

Remember about it in controllability and observability.

2. → 1. : transfer function → state space : this transformation is not very used in practice ⟹ **realization** (name of the transform. 2.→1.) (of a T.F. into a SS. model)

Problem: SS. representation is not unique ⟹ from a single T.F. we can get ∞ different S.S. models (equivalent)

Example : control realization

$$W(z) = \frac{b_0 z^{n-1} + b_1 z^{n-2} + \dots + b_{n-1}}{z^n + a_1 z^{n-1} + a_2 z^{n-2} + \dots + a_n} \quad *$$

we assume **MONIC** denominator

* : we're forcing the assumption of strictly proper system

formula for control realization of $W(z)$:

$$F = \begin{bmatrix} 0 & 1 & & \\ & 0 & 1 & \\ & & 0 & 1 \\ -a_n & -a_{n-1} & \dots & -a_1 \end{bmatrix} \qquad G = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$$

$$H = \begin{bmatrix} b_{n-1} & b_{n-2} & \dots & b_0 \end{bmatrix} \qquad D = 0$$

Example : $W(z) = \frac{2z^2 + \frac{1}{2}z + \frac{1}{4}}{1z^3 + \frac{1}{4}z^2 + \frac{1}{3}z + \frac{1}{5}}$ → strictly proper ← $n=3$

$$F = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1/5 & -1/3 & -1/4 \end{bmatrix} \quad G = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad H = \begin{bmatrix} 1/4 & 1/2 & 2 \end{bmatrix} \quad D = 0$$

cross check:
compute $W(z) = H(zI - F)^{-1} G$

2. → 3. : It's easy: just make the ∞-long division between numerator and denominator of $W(z)$

Example : $W(z) = \frac{1}{z - \frac{1}{2}} = \frac{z^{-1}}{1 - \frac{1}{2}z^{-1}}$:

$$\begin{array}{l|l} z^{-1} & 1 - \frac{1}{2}z^{-1} \\ \hline -z^{-1} + \frac{1}{2}z^{-2} & \hline \\ \cancel{\frac{1}{2}z^{-2}} & z^{-1} + \frac{1}{2}z^{-2} + \frac{1}{4}z^{-3} \\ -\frac{1}{2}z^{-2} + \frac{1}{4}z^{-3} & \\ \hline \cancel{\frac{1}{4}z^{-3}} & \end{array}$$

$$\Longrightarrow W(z) = \frac{z^{-1}}{1 - \frac{1}{2}z^{-1}} = 0 \cdot z^{-0} + 1 z^{-1} + \frac{1}{2} z^{-2} + \frac{1}{4} z^{-3} + \dots$$

$w(0)$ $w(1)$ $w(2)$ $w(3)$ …

IR values

There is a quicker way:

$$y(t) = \frac{z^{-1}}{1-\frac{1}{2}z^{-1}}\,u(t) = \left(z^{-1}\cdot\left(\frac{1}{1-\frac{1}{2}z^{-1}}\right)\right)u(t)$$

Geometrical series: $\sum_{k=0}^{+\infty} a^k = \frac{1}{1-a}$ (if $|a|<1$)

$$\Longrightarrow\quad y(t) = \left(z^{-1}\cdot\sum_{k=0}^{+\infty}\left(\frac{1}{2}z^{-1}\right)^k\right)u(t)$$

$$\Longrightarrow\quad y(t) = \left(0 + z^{-1} + \frac{1}{2}z^{-2} + \frac{1}{4}z^{-3} + \frac{1}{8}z^{-4} + \dots\right)u(t)$$

$w(0)$ $w(1)$ $w(2)$ $w(3)$ . . .

Remark (notational remark):

1. $W(z) = \dfrac{z^{-1}}{1+\frac{1}{3}z^{-1}}$

2. $W(z) = z^{-1} + \frac{1}{2}z^{-2} + \frac{1}{4}z^{-3}$

both are transfer functions (or digital filters),

1. **IIR** filter (Infinite Impulse Response)
2. **FIR** filter (Finite Impulse Response)

$w(0) = 0,\quad w(1) = 1,\quad w(2) = \frac{1}{2},\quad w(3) = \frac{1}{4},\quad w(k) = 0\ \ \forall k \geq 4$

Conclusion:
⟹ whenever we have poles, we have infinite response, otherwise it is finite.

3. ⟶ 2. : We need to recall a fundamental theoretical definition and result.

Def. Given a discrete time signal $s(t)$ ($s(t) = 0\ \ \forall t<0$) its "Z-transform" is defined as:

$$\mathcal{Z}(s(t)) = \sum_{t=0}^{+\infty} s(t)\,z^{-t}$$

transform of a SIGNAL (function of z)

Given this definition, it can be proven that:

$$W(z) = \mathcal{Z}(w(t)) = \sum_{t=0}^{+\infty} w(t)\,z^{-t}$$

transformation formula from I.R. to T.F.

≡ the transf. fun. is the Z-transform of the input response $w(t)$

Notice that the transfer function of a system is the $\mathcal{Z}$ transform of a special signal. This signal is the Impulse Response of the system.

Remark: can this formula be used IN PRACTICE to move from **IR** to **TF**? **NO**. Because we need ∞ points of IR and IR must be noise-free.
⟹ this formula / transformation is only theoretical

1. → 3. : Let's start from: $\begin{cases} x(t+1) = Fx(t) + Gu(t) \\ y(t) = Hx(t) + 0 \end{cases}$

with initial condition: $\begin{cases} x(0) = 0 \\ y(0) = 0 \end{cases}$

We develope this in time domain:

$$\begin{cases} x(1) = Fx(0) + Gu(0) = Gu(0) \\ y(1) = Hx(1) = HGu(0) \end{cases}$$

$$\begin{cases} x(2) = Fx(1) + Gu(1) = FGu(0) + Gu(1) \\ y(2) = Hx(2) = HFGu(0) + HGu(1) \end{cases}$$

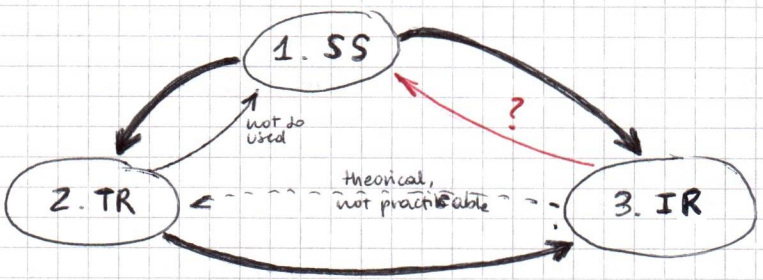
$$\begin{cases} x(3) = Fx(2) + Gu(2) = F^2Gu(0) + FGu(1) + Gu(2) \\ y(3) = Hx(3) = HF^2Gu(0) + HFGu(1) + HGu(2) \end{cases}$$

...

$$\Rightarrow \quad y(t) = \underbrace{0}_{w(0)}\,u(t) + \underbrace{HG}_{w(1)}\,u(t-1) + \underbrace{HFG}_{w(2)}\,u(t-2) + \underbrace{HF^2G}_{w(3)}\,u(t-3) + \dots$$

$$\Rightarrow \quad w(t) = \begin{cases} 0 & t = 0 \\ HF^{t-1}G & t > 0 \end{cases}$$

Summary of transformations:

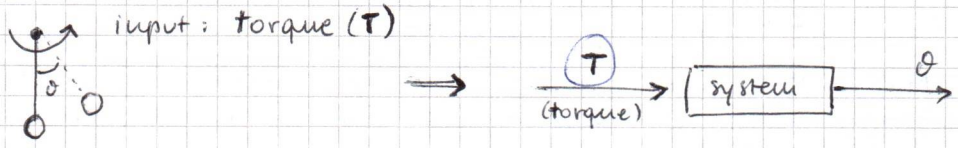


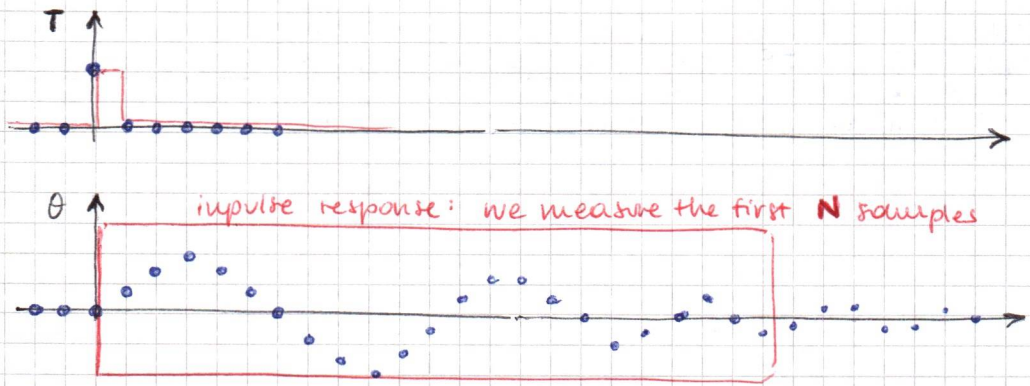
IR: easy to obtain experimentally, but difficult to get out (3.→2., 3.→1.)

Moving from IR. to S.S. is the key task of: **SUBSPACE-BASED STATE SPACE SYSTEM IDENTIFICATION (4SID methods)**

(Remark: in MATLAB system ID. toolbox: >> n4sid)

Motivation:
The original (first developed) 4SID method starts from the mesurement of the system output in a very simple experiment → "impulse experiment".





The idea is that it is experimentally easy to make this experiment.
The problem is how to identify a model $\{\hat{F}, \hat{G}, \hat{H}\}$ starting from $\{w(0), w(1), w(2), \dots, w(N)\}$.

Remark: for **tutorial** reasons we'll see the solution of this problem in 2 steps:

1. IR measurement is assumed "noise-free"
2. IR is measured with noise:

$$\tilde{w}(t) = w(t) + \eta(t) \qquad t = 0,1,2,\dots,N$$

$\tilde{w}(t)$: measured noise IR; $w(t)$: true (noise-free) I.R.; $\eta(t)$: measurement of noise (e.g. WN)

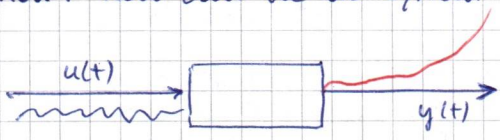
1. is easy but not realistic, 2. is the real problem.

Remark: for tutorial reasons we see in detail only **4SID** when the experiment is an impulse-experiment (1st and original version of 4SID). However, **4SID** can be extended to any input signal:

$\{u(1), u(2), \dots, u(N)\}$ generic input (sufficiently exciting)

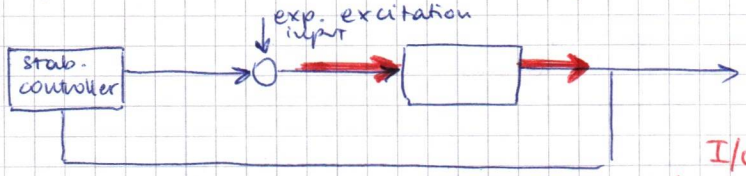
$\{y(1), y(2), \dots, y(N)\}$

Question: how can we do system identification for unstable systems?



the open loop experiment is unfeasible (because whatever is the input (impulse, steps, random sequence, WN, ..) we get divergence of the output)

So what is done is to make an experiment which is in close loop: we close the loop with the "stabilizing controller", then we inject experimental excitation input and we measure the input output sequence ( )



I/O data are collected in a closed loop experiment this is not a problem because we get anyway the I/O data

---

21/04

Before presenting the algorithm we need to recall the fundamental concepts of:
- observability
- controllability (reachability)

of a dynamical system.

Consider:

$$\begin{cases} x(t+1) = Fx(t) + Gu(t) \\ y(t) = Hx(t) \end{cases}$$

The system is FULLY OBSERVABLE from the output if and only if the observability matrix:

$$O = \begin{bmatrix} H \\ HF \\ HF^2 \\ \vdots \\ HF^{n-1} \end{bmatrix}$$

is fully rank ($\text{rank}(O) = n$).
(only refer to H and F)

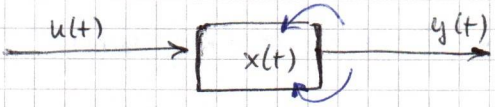
The system is fully CONTROLLABLE (/REACHABLE) from the input if and only if the controllability matrix:

$$R = [\,G \quad FG \quad F^2G \quad \cdots \quad F^{n-1}G\,]$$ is full rank $(\text{rank}(R) = n)$.

(only refer to G and F)

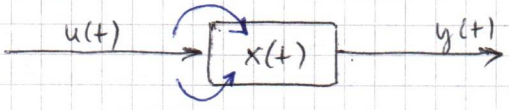
Observability of the state from the output = we can "see" the state from the output sensors:

if we look at $y(t)$ we can have a look on the state



Controllability of the state from the input = we can "control" ("move") the state using input:

the input can modify the state



Example: (observability)

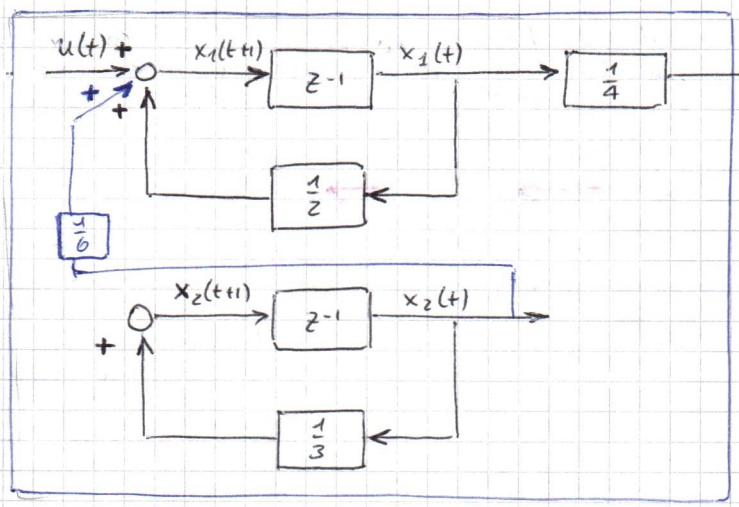
$$S: \begin{cases} x_1(t+1) = \frac{1}{2}x_1(t) + u(t) \\ x_2(t+1) = \frac{1}{3}x_2(t) \\ y(t) = \frac{1}{4}x_1(t) \end{cases}$$

$n = 2$

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}, \quad F = \begin{bmatrix} 1/2 & 0 \\ 0 & 1/3 \end{bmatrix}, \quad H = \begin{bmatrix} 1/4 & 0 \end{bmatrix}$$

$$\Longrightarrow O = \begin{bmatrix} H \\ HF \end{bmatrix} = \begin{bmatrix} 1/4 & 0 \\ 1/8 & 0 \end{bmatrix} \Longrightarrow \text{rank}(O) = 1 < 2$$

$\Longrightarrow$ the system S is not fully observable



If we look at $y(t)$ can we see (observe) x_1 and x_2? We can see x_1 but not x_2

$$S: \begin{cases} x_1(t+1) = \frac{1}{2}x_1(t) + u(t) + \frac{1}{6}x_2(t) \\ x_2(t+1) = \frac{1}{3}x_2(t) \\ y(t) = \frac{1}{4}x_1(t) \end{cases}$$

$$F = \begin{bmatrix} 1/2 & 1/6 \\ 0 & 1/3 \end{bmatrix}, \quad H = \begin{bmatrix} \frac{1}{4} & 0 \end{bmatrix} \Longrightarrow O = \begin{bmatrix} 1/4 & 0 \\ 1/8 & 1/24 \end{bmatrix}$$

$\Longrightarrow \text{rank}(O) = 2$

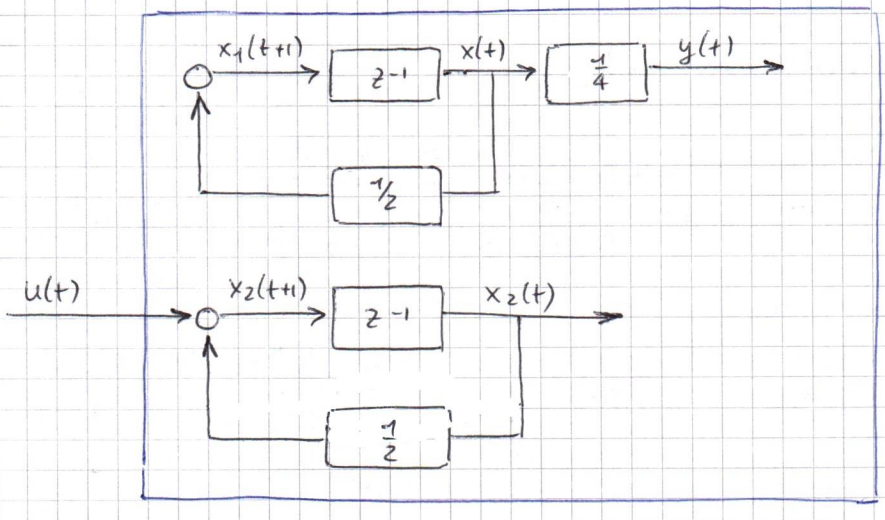
$\Longrightarrow$ fully observable

Example: (controllability)

$$S: \begin{cases} x_1(t+1) = \frac{1}{2} x_1(t) \\ x_2(t+1) = \frac{1}{3} x_2(t) + u(t) \\ y(t) = \frac{1}{4} x_1(t) \end{cases}$$

$$F = \begin{bmatrix} 1/2 & 0 \\ 0 & 1/3 \end{bmatrix}, \quad G = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \Longrightarrow R = \begin{bmatrix} 0 & 0 \\ 1 & 1/3 \end{bmatrix}$$

$\Longrightarrow$ rank $(R) = 1 < 2$
$\Longrightarrow$ not controllable



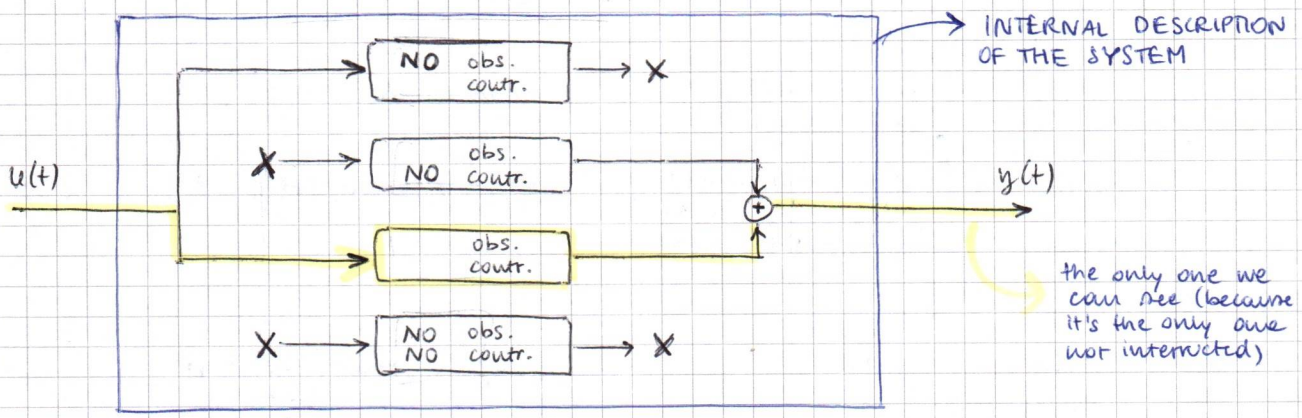
we can control $x_2(t)$ but not $x_1(t)$ with $u(t)$

$$S: \begin{cases} x_1(t+1) = \frac{1}{2} x_1(t) + \frac{1}{6} x_2(t) \\ x_2(t+1) = \frac{1}{3} x_2(t) + u(t) \\ y(t) = \frac{1}{4} x_1(t) \end{cases}$$

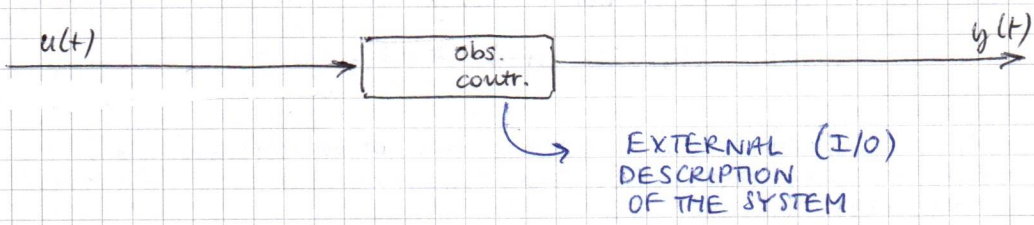
$$F = \begin{bmatrix} 1/2 & 1/4 \\ 0 & 1/3 \end{bmatrix}, \quad G = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \Longrightarrow R = \begin{bmatrix} 0 & 1/6 \\ 1 & 1/3 \end{bmatrix}$$

$\Longrightarrow$ rank $(R) = 2$
$\Longrightarrow$ fully controllable

Remark: Any system can be divided into 4 sub-systems:



$\Vert$ from the external point of view



Only observable and controllable part is visible.

Final definition (befor 4SID algorithm):

HANKEL MATRIX of order n built from I.R. $\{w(0), w(1), \dots, w(N)\}$:

$$H_n = \begin{bmatrix} w(1)^{*} & w(2) & w(3) & \cdots & w(n) \\ w(2) & w(3) & w(4) & \cdots & w(n+1) \\ w(3) & w(4) & w(5) & \cdots & w(n+2) \\ \vdots & & & & \vdots \\ w(n) & w(n+1) & & & w(2n-1) \end{bmatrix} = [\text{////}]$$

Square matrix $n \times n$

→ we need I.R. up to time $2n-1$

* note that we start from $w(1)$ not $w(0)$. $(w(0) = D)$

Now recall transformation: 1. (SS) ⟶ 3. (IR) :

$$\{F, G, H\} \longrightarrow w(t) = \begin{cases} 0 & t = 0 \\ HF^{t-1}G & t > 0 \end{cases} \qquad (D = 0)$$

⟹ we can rewrite H_n :

$$H_n = \begin{bmatrix} HG & HFG & \cdots & HF^{n-1}G \\ HFG & HF^2G & \cdots & HF^nG \\ \vdots & & & \\ HF^{n-1}G & HF^nG & \cdots & HF^{2n-2}G \end{bmatrix} = \underbrace{\begin{bmatrix} H \\ HF \\ HF^2 \\ \vdots \\ HF^{n-1} \end{bmatrix}}_{\text{matrix } O \text{ (observability matrix)}} \underbrace{\begin{bmatrix} G & FG & F^2G & \cdots & F^{n-1}G \end{bmatrix}}_{\text{matrix } R \text{ (controllability matrix)}}$$

⟹ $H_n = O \cdot R$ ⟹ H_n can be factorized into observability × controllability matrix.

Now we can eventually present the procedure (algorithm) to obtain $\{\hat{F}, \hat{G}, \hat{H}\}$ starting from a noise-free measured impulse response (I.R.) : $\{w(0), w(1), \dots, w(N)\}$.

Step 1. Build the Hankel matrix in increasing order and each time check the rank:

$H_1 = [w(1)]$

$H_2 = \begin{bmatrix} w(1) & w(2) \\ w(2) & w(3) \end{bmatrix}$ suppose rank $(H_2) = 2$

$H_3 = \begin{bmatrix} w(1) & w(2) & w(3) \\ w(2) & w(3) & w(4) \\ w(3) & w(4) & w(5) \end{bmatrix}$ suppose rank $(H_3) = 3$

$\vdots$

$H_n = [\dots]$ suppose rank $(H_n) = n$

$H_{n+1} = [\dots]$ suppose rank $(H_{n+1}) = n$

⟹ H_{n+1} is the first matrix **not** full rank

⟹ we have estimated the order n of the system

Step 2. Take H_{n+1} (square $(n+1) \times (n+1)$ matrix of rank $(H_{n+1}) = n$) and factorize into two rectangular matrices of size $(n+1) \times n$ and $n \times (n+1)$.
(Remark: this is possible because rank $(H_{n+1}) = n$ and not $n+1$)

$$H_{n+1} = \begin{bmatrix} \cdots & n \\ n+1 & \end{bmatrix} \begin{bmatrix} \cdots & n+1 \\ n & \end{bmatrix}$$

we consider this matrix as extended observability matrix O_{n+1}

we consider this matrix as extended controllability matrix R_{n+1}

$$O_{n+1} = \begin{bmatrix} H \\ HF \\ HF^2 \\ \vdots \\ HF^{n-1} \\ HF^n \end{bmatrix} \quad n+1 \quad (\leftrightarrow n)$$

$$R_{n+1} = \begin{bmatrix} G & FG & F^2G & \cdots & F^{n-1}G & F^nG \end{bmatrix} \quad n \quad (\leftrightarrow n+1)$$

Step 3. (H, F, G estimation)
Using O_{n+1} and R_{n+1} we can easily find $\hat{G}$ and $\hat{H}$:

$$H_{n+1} = \begin{bmatrix} \overbrace{H} & \cdots \\ \vdots & \end{bmatrix} \begin{bmatrix} G & \cdots \\ \vdots & \end{bmatrix}$$

$\Leftarrow$ we can simply read H from the first row of O_{n+1} and G from the first column of R_{n+1}

$$\hat{G} = R_{n+1}[:, 1]$$
$$\hat{H} = O_{n+1}[1, :]$$

What about $\hat{F}$?
Consider for example O_{n+1} (but the same can be done also starting from R_{n+1}):

$$O_{n+1} = \begin{bmatrix} H \\ HF \\ HF^2 \\ \vdots \\ HF^{n-1} \\ HF^n \end{bmatrix}$$

$$O_1 = O_{n+1}[1:n, :]$$

$$O_2 = O_{n+1}[2:n+1, :]$$

Notice that O_1 and O_2 are square $n \times n$ matrices.

Notice that O_1 and O_2 are linked by the so-called "shift invariant" property.

$$\Longrightarrow \quad O_2 = O_1 \cdot F$$

since O_1 is square and (invertible) — because it's the observability matrix and the system order is n (rank(obs. matrix) = n)

$$\Longrightarrow \quad \hat{F} = O_1^{-1} \cdot O_2$$

End of algorithm.

Conclusion: in a simple and constructive way we have estimated a state space model of the system $\{\hat{F}, \hat{G}, \hat{H}\}$ starting from a measured noise-free input response (I.R.), using only $2n+1$ samples of the I.R.

Example: $N = 200$, $n = 4$ } we can make system identification using only 9 samples of I.R.

Comment: this method was known since '60 (Kalman-Ho method). It is simple, clean, nice but useless.
Why practically useless? Because if $w(t)$ is noisy then step 1. stops with probability equal to 0 (in practice never stops). Even if we know a priori the value of n, the estimated $\{\hat{F}, \hat{G}, \hat{H}\}$ would be badly wrong.

(The method is not robust w.r.t. the measurement noise).
(which is always present in real applications)

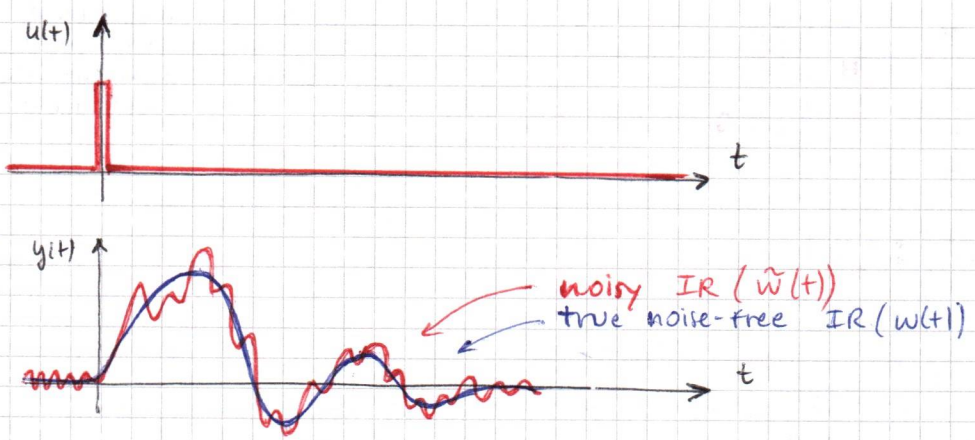
This method was sleeping until late '80, when a new numerical-algebra tool was fully developed: SINGULAR VALUES DECOMPOSITION (S.V.D.)

technique for:
- data-compression
- optim. separation of signal from noise

Now we can consider the real problem: when we have noisy measurement of I.R.

$$\tilde{w}(t) = w(t) + \eta(t) \qquad t = 0, 1, \dots, N$$

- $\tilde{w}(t)$: measured noisy I.R.
- $w(t)$: true/noise-free I.R.
- $\eta(t)$: measurement of the noise (Broad-Band noise)
- N: typical value of N is something between 100 and 1000



4SID PROCEDURE with NOISY MEAS. OF I.R.

Step 1. Build Hankel matrix from data using "one shot" all the N available data points.

$$\tilde{w}(t) \quad t = 0, 1, 2, \dots, N \implies \tilde{H}_{qd}$$

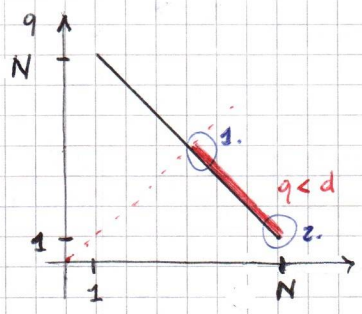
- $\tilde{H}$: we indicate that it is build with noisy data
- H_{qd}: it's not a square matrix, but a rectangular matrix $q \times d$

$$\tilde{H}_{qd} = \begin{bmatrix} \tilde{w}(1) & \tilde{w}(2) & \tilde{w}(3) & \cdots & \tilde{w}(d) \\ \tilde{w}(2) & \tilde{w}(3) & \tilde{w}(4) & \cdots & \tilde{w}(d+1) \\ \tilde{w}(3) & \tilde{w}(4) & & & \\ \vdots & \vdots & & & \\ \tilde{w}(q) & & & & \tilde{w}(d+q-1) \end{bmatrix}$$

we want to use the full dataset $\implies q + d - 1 = N$

Remark on the choice of q and d:
Let's assume that $q < d$.
We know that:

$$q + d - 1 = N \implies q = N + 1 - d$$



If $q \approx d$ the method has a better accuracy.
If $q \ll d$ the method is computationally less intensive.
} so we want 1. AND 2.

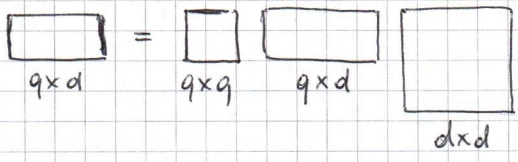
Rule of thumb: if $0.6\,d < q < d \implies$ the sensitivity of the estimation quality to the choice of q and d is very low.
$\implies$ $\boxed{q \approx 0.6\,d}$

Example: $N = 1000$: $q = 400$, $d = 601$
(why 601? $d + q = N + 1$)

22/04

Step 2. SVD of $\tilde{H}_{qd}$:

$$\tilde{H}_{qd} = \tilde{U}\tilde{S}\tilde{V}^T$$



$\tilde{U}$ and $\tilde{V}$ are **UNITARY** matrices.

(Def. M (square matrix) is unitary if:
- $\det(M) = 1$ ($\implies$ invertible)
- $M^{-1} = M^T$)

$$\tilde{S} = \begin{bmatrix} \sigma_1 & & \phi & \vdots & \\ & \sigma_2 & & & \phi \\ \phi & & \ddots & & \\ & & & \sigma_q & \end{bmatrix}$$

where $\sigma_1, \sigma_2, \ldots, \sigma_q$ are the "singular values" of $\tilde{H}_{qd}$.
they're real positive numbers sorted in decreasing order:
$\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \cdots \geq \sigma_q$.

Remark: the singular values of a rectangular matrix are a sort of eigenvalues of a square matrix. (They're sort of eigens for a rect. matrix)
in some sense, SVD is a sort of "diagonalization" of a rectangular matrix.

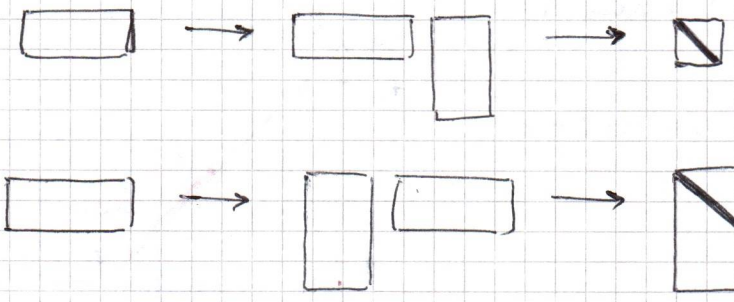
Recall that for a square matrix A:

$$\text{eig}(A) = \text{roots}\left(\det(A - \lambda I)\right)$$

If M is rectangular:

$$\underset{\text{singular values}}{SV(M)} = \sqrt{\text{eig}(MM^T)} = \sqrt{\text{eig}(M^T M)}$$

(This relation is valid only for non-zero eigenvalues)



(it's longer but the last eigenvalues are 0, so the eigenvalues $\neq 0$ of the two are equal)

How can we compute **SVD**?
The optimal numerical computation is not trivial (use ">> svd(M)").
The theoretical method for SVD computation is to make two diagonalization steps:

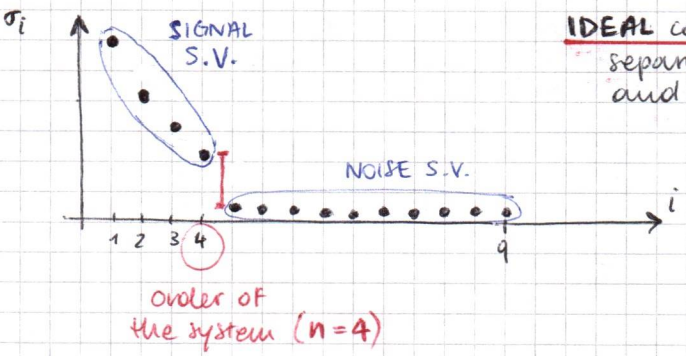
$$\underbrace{\tilde{H}_{qd}\,\tilde{H}_{qd}^{T}}_{\text{square } q\times q \text{ matrix}} = \tilde{U}\,\tilde{S}\tilde{S}^{T}\,\tilde{U}^{T}$$

$$\tilde{S}\tilde{S}^{T} = \begin{bmatrix} \sigma_1^2 & & & \\ & \sigma_2^2 & & \\ & & \ddots & \\ & & & \sigma_q^2 \end{bmatrix}$$

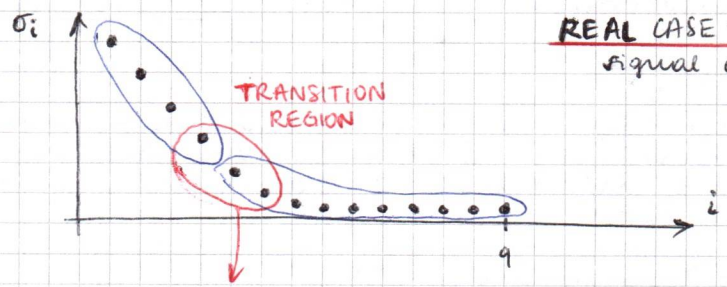
$$\underbrace{\tilde{H}_{qd}^{T}\,\tilde{H}_{qd}}_{\text{square } d\times d \text{ matrix}} = \tilde{V}\,\tilde{S}^{T}\tilde{S}\,\tilde{V}^{T}$$

$$\tilde{S}^{T}\tilde{S} = \begin{bmatrix} \sigma_1^2 & & & & \\ & \sigma_2^2 & & & \\ & & \ddots & & \\ & & & \sigma_q^2 & \\ & & & & 0 \end{bmatrix}$$

Step 3. We need to plot the singular values and cut off the 3 matrices.



IDEAL case: there is a perfect/clear separation between "signal" and "noise" (JUMP)



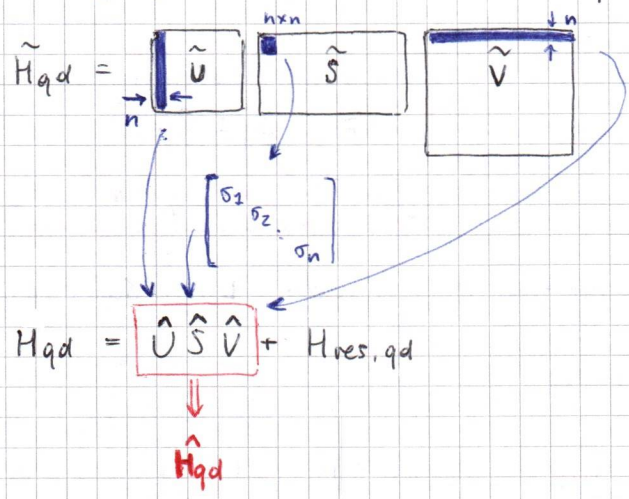
REAL CASE: no clear transition between signal and noise

n (order of the system) is in this transition region

With some empirical test we can select a good compromise between complexity, precision and overfitting.
(ex. see CROSS-VALIDATION)

What is the trade-off? We want small n because we'd have lower complexity, lower risk of overfitting, but there is less precision in the fitting. We want large n for high precision but there is more complexity and the overfitting can be an issue.

After decision of the value of n we split $\hat{U}, \hat{S}, \hat{V}$:



$\text{rank}(\tilde{H}_{qd}) = q$
$\text{rank}(\hat{H}_{qd}) = n$
$\text{rank}(H_{res,qd}) = q$

$\tilde{H}_{qd} \rightarrow \hat{H}_{qd}$: we made a major rank reduction

Step 4. The procedure can be completed with the estimation of $\{\hat{F}, \hat{G}, \hat{H}\}$ using the "cleaned" matrix $\hat{H}_{qd}$.

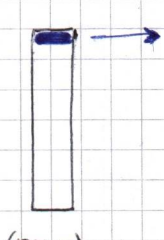
$$\hat{H}_{qd} = \hat{U}\hat{S}\hat{V} = \hat{U}\,\hat{S}^{1/2}\,\hat{S}^{1/2}\,\hat{V}^T$$

$$\hat{S}^{1/2} \rightarrow \begin{bmatrix} \sqrt{\sigma_1} & & & \\ & \sqrt{\sigma_2} & & \\ & & \ddots & \\ & & & \sqrt{\sigma_n} \end{bmatrix}$$

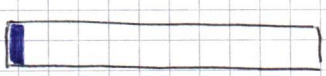
Def. $\hat{O} := \hat{U}\hat{S}^{1/2}$
$\hat{R} := \hat{S}^{1/2}\hat{V}^T$
$\rightarrow \hat{H}_{qd} = \hat{O} \cdot \hat{R}$

We can consider $\hat{O}$ and $\hat{R}$ as extended observability and reachability matrices of the system.

Note: $\underset{(q\times d)}{\hat{H}_{qd}} = \underset{(q\times n)}{\square}\ \underset{(n\times n)}{\square}\ \underset{(n\times d)}{\square} = \hat{U}\hat{S}\hat{V}^T$

$\hat{O} =$  $(q \times n)$

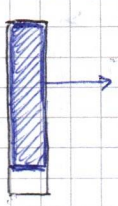
$\rightarrow \underline{\hat{H} = \hat{O}[1,:]}$

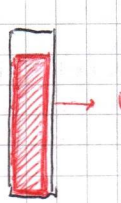
$\hat{R} =$  $(n \times d)$

$\underline{\hat{G} = \hat{R}[:,1]}$

we can use $\hat{R}$ too

What about the estimation of $\hat{F}$?
Consider for example $\hat{O}$ and use the shift invariance property:

$\hat{O} =$  $\rightarrow O_1 = O[1:q-1,:]$

$\hat{O} =$  $\rightarrow O_2 = O[2:q,:]$

Using S.I.P. (Shift Invariant Property)

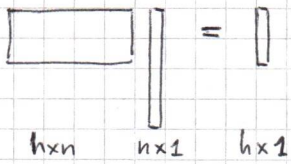
$\Longrightarrow \quad \hat{O}_1 \cdot \hat{F} = \hat{O}_2$

$\Longrightarrow$ it's not possible to say: ~~$\hat{F} = \hat{O}_1^{-1} \hat{O}_2$~~

because $\hat{O}_1$ is very far from being a square matrix

In this case we can use the approximate Least Square solution of this linear system.

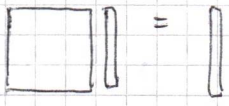
$Ax = b \quad \Longrightarrow$ 3 cases

1. 

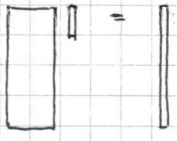
$h \times n \quad n \times 1 \quad h \times 1$

$h < n$ (h = number of equations, n = number of unknowns)

We have less equations than variables $\Longrightarrow$ the system is called "under-determined" $\Longrightarrow$ ∞-solutions

2. 

$h = n \Longrightarrow$ 1! solution (if A is invertible)

3. 

$h > n \Longrightarrow$ we have more equations than variables $\Longrightarrow$ the system is overdetermined $\Longrightarrow$ NO solutions

$\Longrightarrow$ approximate linear squares (L.S.) solution:

$Ax = b \quad \Longrightarrow \quad \underbrace{A^T A}_{\text{square matrix}} x = A^T b$

$\Longrightarrow \quad \hat{x} = \boxed{(A^T A)^{-1} A^T} \, b$

$A^\dagger$ called PSEUDO INVERSE (surrogate of A^{-1} when A is rectangular)

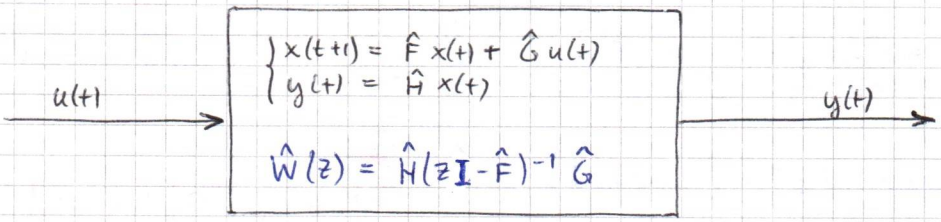
Using this pseudo-inverse method:

$\hat{O}_1 \cdot \hat{F} = \hat{O}_2 \quad \Longrightarrow \quad \hat{O}_1^T \cdot \hat{O}_1 \hat{F} = \hat{O}_1^T \hat{O}_2$

$\Longrightarrow \quad \hat{F} = (\hat{O}_1^T \hat{O}_1)^{-1} \hat{O}_1^T \hat{O}_2$

End of algorithm.

Conclusion: starting from a noisy I.R. $\{w(1), w(2), \ldots, w(N)\}$ we have estimated a model $\{\hat{F}, \hat{G}, \hat{H}\}$ in a non-parametric / constructive way.



Remark: it can be done something similar also if the measured input is generic (not impulse).

Remark: (optimality of 4SID): the method is optimal in the sense that it makes the best possible rank reduction of $\tilde{H}_{qd}$:

$$\tilde{H}_{qd} = \hat{H}_{qd} + \tilde{H}_{res,qd}$$

rank: q n q

In general there are ∞ ways to make a rank reduction.

Example: $\begin{bmatrix} 2 & 5 & 3 & 6 & 5 \\ 5 & 3 & 6 & 5 & 7 \\ 3 & 6 & 5 & 7 & 1 \end{bmatrix} = \begin{bmatrix} \quad \end{bmatrix} + \begin{bmatrix} \quad \end{bmatrix}$

rank = 3

we want a rank = 2 matrix

residual matrix with rank = 3

Stupid way: $\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 1 & 5 & 3 & 6 & 5 \\ 5 & 2 & 6 & 5 & 7 \\ 3 & 6 & 5 & 7 & 1 \end{bmatrix}$

Is this optimal? **No**, it's just one of the ∞ ways.

Goal: obtain the desired rank-reduction by discarding / trashing the minimum amount of information contained in the original matrix.

$\Longrightarrow$ SVD makes exactly this: $\tilde{H}_{res,qd}$ is the minimum possible

in the sense of the FROBENIUS NORM (*)

(*) $$|\tilde{H}_{res,qd}|_F = \sqrt{\sum_{i,j} ((\tilde{H}_{res,qd})_{i,j})^2}$$

(Example: $\left| \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \right|_F = \sqrt{1^2 + 2^2 + 3^2 + 4^2 + 5^2 + 6^2}$)

Remark: 4SID is a constructive method that can be implemented in a fully automatic way, except:

- **q** and **d** selection (not critical)
- choise of **n** (supervised by the designer, but it can be made automatic using a cross-validation method)

SVD was a historical turning point in machine-learning algorithms because it allows:

- very efficient compression of information
- very efficient separation of "important information" from noise

Exercise : (similar to an exam exercise)

Consider the following SS model:

$$F = \underbrace{\begin{bmatrix} 1/2 & 0 \\ 1 & 1/4 \end{bmatrix}}_{2\times 2}, \quad G = \underbrace{\begin{bmatrix} 1 \\ 0 \end{bmatrix}}_{1}, \quad H = \underbrace{[0 \;\; 1]}_{1}, \quad D = 0$$

$2\times 2 \Rightarrow n = 2$ (order of the system)

$\Rightarrow$ 1 input

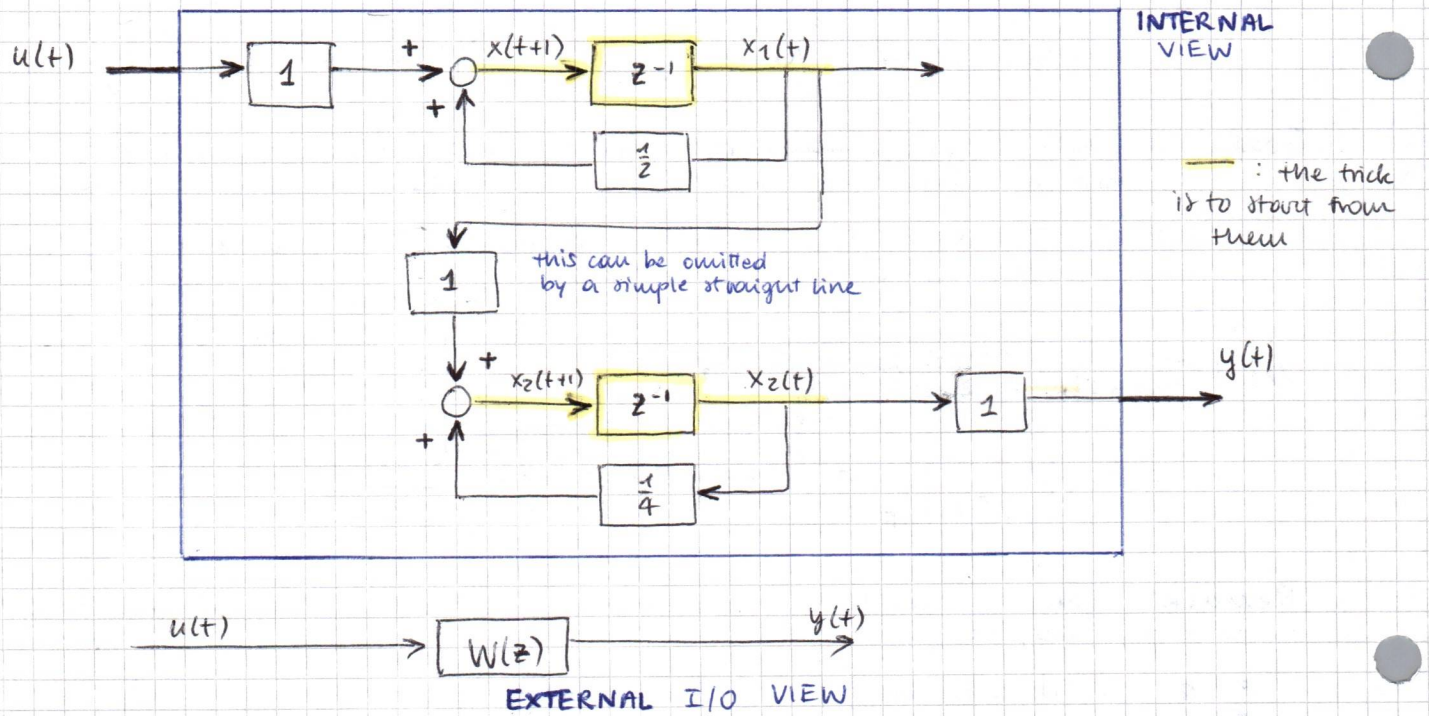
$\Rightarrow$ 1 output

$\Longrightarrow$ SISO system, order $n = 2$

- Write the time domain equations of the system in SS representation.

$$\begin{cases} x_1(t+1) = \frac{1}{2} x_1(t) + 0\, x_2(t) + 1\, u(t) \\ x_2(t+1) = 1\, x_1(t) + \frac{1}{4} x_2(t) + 0\, u(t) \\ y(t) = 0\, x_1(t) + 1\, x_2(t) \end{cases}$$

- Write the block-scheme of the SS representation.



EXTERNAL I/O VIEW

Notice that by "visual inspection" the system is fully observable and fully controllable. (From $y(t)$ we can see $x_2(t)$ and thanks to the communication we see $x_1(t)$ too. In a similar way, $u(t)$ can control $x_1(t)$ and $x_2(t)$.)

- Formal verification that the system is fully observable and fully controllable.

$$O = \begin{bmatrix} H \\ HF \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 1/4 \end{bmatrix} \qquad \text{rank}(O) = 2 = n \Rightarrow \text{fully observable}$$

$$R = [\, G \;\; FG \,] = \begin{bmatrix} 1 & 1/2 \\ 0 & 1 \end{bmatrix} \qquad \text{rank}(R) = 2 = n \Rightarrow \text{fully controllable}$$

Compute the extended ("$n+1$") O and R matrices:

$$O_3 = \begin{bmatrix} H \\ HF \\ HF^2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 1/4 \\ 3/4 & 1/6 \end{bmatrix}$$

$$R_3 = \begin{bmatrix} 1 & 1/2 & 1/4 \\ 0 & 1 & 3/4 \end{bmatrix}$$

- Compute the corresponding TF representation.

1st method: direct manipulation of SS equations:

$$\begin{cases} x_1(t+1) = \frac{1}{2} x_1(t) + u(t) \\ x_2(t+1) = x_1(t+1) + \frac{1}{4} x_2(t) \\ y(t) = x_2(t) \end{cases}$$

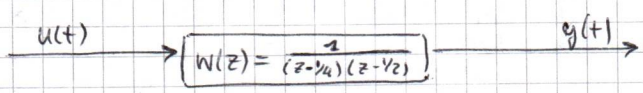
$$z x_1(t) - \frac{1}{2} x_1(t) = u(t)$$

$$\Longrightarrow \quad x_1(t) = \frac{1}{(z-\frac{1}{2})} u(t)$$

$$z x_2(t) \quad x(t) = \frac{1}{(z-\frac{1}{2})} u(t)$$

$$\Longrightarrow \quad x_2(t) = \frac{1}{(z-\frac{1}{4})(z-\frac{1}{2})} u(t)$$

$$\Longrightarrow \quad y(t) = \underbrace{\frac{1}{(z-\frac{1}{4})(z-\frac{1}{2})}}_{W(z)} x_2(t)$$



We have 2 poles: $z = 1/4$, $z = 1/2$ (same as eigenvalues of F)
(it might happen (it happens) that we have less poles than eigenvalues if there's something hidden (non-obs./non-contr. part))

↳ because in that case there are semplifications

2nd method: formula:

$$W(z) = H(zI - F)^{-1} G$$

$$= [0 \;\; 1] \left(\begin{bmatrix} z-1/2 & 0 \\ -1 & z-\frac{1}{4} \end{bmatrix} \right)^{-1} \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$= \dots = \frac{1}{(z-\frac{1}{4})(z-\frac{1}{2})}$$

- Write I/O time-domain representation.

$$y(t) = \frac{1}{z^2 - \frac{3}{4} z + \frac{1}{8}} u(t)$$

$$= \frac{z^{-2}}{1 - \frac{3}{4} z^{-1} + \frac{1}{8} z^{-2}} u(t)$$

$$\Longrightarrow \quad y(t) = \frac{3}{4} y(t-1) - \frac{1}{8} y(t-2) + u(t-2)$$

difference equation (I/O)

- Compute the first 6 values (including $w(0)$) of I.R. ($w(0), w(1), \dots, w(5)$).

We decide to compute it from **TF**. (since it's not specified)
(suggestion: in negative powers)

$$W(z) = \frac{z^{-2}}{1 - \frac{3}{4} z^{-1} + \frac{1}{8} z^{-2}}$$

We procede with the long division algorithm.

$$z^{-2} - z^{-2} + \frac{3}{4}z^{-3} - \frac{1}{8}z^{-4} \;\Big|\; 1 - \frac{3}{4}z^{-1} + \frac{1}{8}z^{-2}$$
$$\frac{3}{4}z^{-3} - \frac{1}{8}z^{-4} \qquad\qquad z^{-2} + \frac{3}{4}z^{-3} + \frac{7}{16}z^{-4} + \frac{15}{64}z^{-5}$$
$$\vdots$$

$$\Longrightarrow \quad z^{-2} + \frac{3}{4}z^{-3} + \frac{7}{16}z^{-4} + \frac{15}{64}z^{-5}$$

$$\Longrightarrow \quad w(0) = 0, \quad w(1) = 0, \quad w(2) = 1, \quad w(3) = \frac{3}{4}, \quad w(4) = \frac{7}{16}, \quad w(5) = \frac{15}{64}$$

! $W(z) = \dfrac{z^{-2}}{(\cdots)}$ (two delays) $\Rightarrow$ the first two elements of the response are 0

In fact we have: $W(z) = \dfrac{1}{(z-1/4)(z-1/2)}$
$\Rightarrow$ no zeros, two poles
$\Rightarrow$ # poles − # zeros = 2 (number of initial zeros)

- Build Hankel matrix and stop when rank is not full.

$$H_1 = [0]$$

$$H_2 = \begin{bmatrix} 0 & 1 \\ 1 & 3/4 \end{bmatrix} \qquad \text{rank}(H_2) = 2$$

$$H_3 = \begin{bmatrix} 0 & 1 & 3/4 \\ 1 & 3/4 & 7/16 \\ 3/4 & 7/16 & 15/64 \end{bmatrix} \qquad \text{rank}(H_3) = 2 \Longrightarrow \text{we stop} \Longrightarrow \text{the order of the system } (n) \text{ is } 2$$

Note that: $O_3 R_3 = H_3$ (just for check)

S

CHAPTER 2

(Parametric Black-Box System identification of I/O systems using a frequency domain approach)

So far we have seen (MIDA 1) parametric Black Box identification of I/O systems (ARMAX) and time series (ARMA).
In MIDA 2 - chapter 1 we have seen non parametric Black Box identification of I/O systems (4SID → SS representation).
The frequency domain approach is Black Box and parametric (very used in practice since it's robust and reliable).

Parametric system identification: 4 classical steps:

1. Experiment design and data pre-processing
2. Selection of parametric model class: $M(\theta)$
3. Definition of a performance index: $J(\theta)$
4. Optimization: $\hat{\theta} = \arg\min_{\theta} J(\theta)$

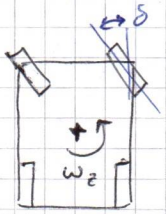
1. → A special type of experiment and data pre-processing is needed
3. → A new special performance index is needed

27/04

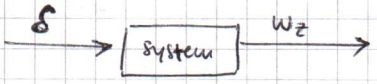
The general idea of the method is:

- make a set of "single sinusoid" (/ "single-tone") excitation experiment
- from each experiment we estimate a single point of the frequency response of the system
- fit the estimated and modeled frequency response to obtain the optimal model

Example: (vehicle from above)

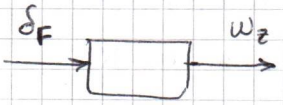


ω_z = "YAW-RATE" (rotational velocity of car around its vertical axis)



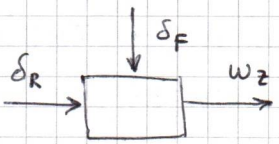
This is the dynamical relationship linking the input (steer) to the output (Yaw-rate) → very important for stability control design

We can have 3 situations:



Control variable is front steer: this application is AUTONOMOUS CAR

SISO



(Today case in high performance cars) The drivers moves δ_F (δ front): measurable disturbance and the control system makes use of δ_F (control variable)

SISO

Both δ_F and δ_R are control variables (high performance autonomous cars)

MISO (multiple inputs, single output)

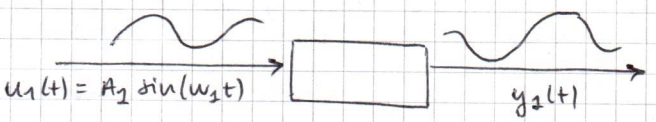
In the experiment design set we first have to select a set of excitation frequencies:

ω_H = max frequency $\Rightarrow$ $\{\omega_1, \omega_2, \omega_3, \dots, \omega_H\}$

So we have H frequencies ranging from ω_1 and ω_H: usually they are evenly spaced ("$\Delta\omega$" is constant). ω_H must be selected according to the bandwidth of the control system.

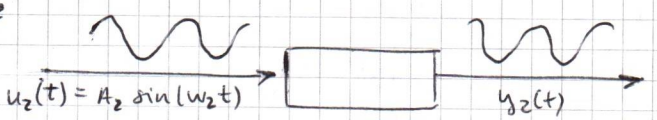
Now we make H exeperiments (H experiments)

Experiment #1: ω_1



$t = 1, \dots, N$

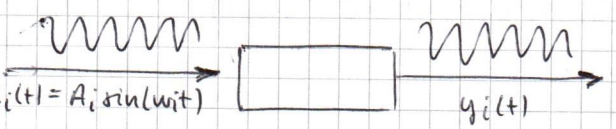
Experiment #2: ω_2



$t = 1, \dots, N$

⋮

Experiment #i: ω_i

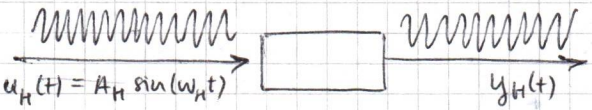


$t = 1, \dots, N$

the frequency keep increasing

⋮

Experiment #H : ω_H



$t = 1, \dots, N$

Remark: the amplitudes $A_1, A_2, \dots, A_H$ can be equal (constant) or, more frequently, they decrease as the frequency increase (this is to fulfill power constraints on the input (actuator)).

Example : (of steer)

$\delta(t)$ is the steering angle (moved by an actuator).
The requested steering torque is proportional to δ :

$$T(t) = k\,\delta(t)$$

$\Rightarrow$ steering power $\rightarrow$ $T(t)\cdot\dot{\delta}(t) = K\,\delta(t)\,\dot{\delta}(t)$ → steering velocity

If $\delta(t) = A_i \sin(\omega_i t)$ $\Rightarrow$ steering power is:

$(=)$ $K A_i \sin(\omega_i t)\cdot \omega_i A_i \cos(\omega_i t)$

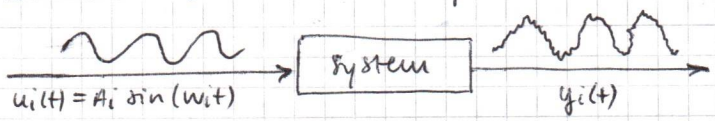
$\Rightarrow$ steering power is proportional to : $\underbrace{K A_i^2 \omega_i}$

if we have a limit to this power, this power should be constant during #1, .., #H experiments

$$\Rightarrow \quad K A_i^2 \omega = \text{constant}$$

$$\Rightarrow \quad A_i = \sqrt{\frac{\text{constant}}{K\omega_i}}$$

Now let's focus on the i-th experiment :



Remember that if the system is **LTI** (linear time invariant) the frequency response theorem says that if the input is a sinusoid of frequency ω_i $\Rightarrow$ also the output **MUST** be a pure sinusoid of frequency ω_i.

However, $y_i(t)$, in the real applications is not a perfect sinusoid :

- noise on output measurement
- noise on the system (not directly on the output)
- (small) non-linear effects (that we neglect)

In pre-processing of I/O data we want to extract from $y_i(t)$ a perfect sinusoid of frequency ω_i $\Rightarrow$ we force the assumption that the system is LTI $\Rightarrow$ the output must be a pure sinusoid of frequency ω_i (all the remaining signal is noise).

The "model" of the output signal is a sinusoid :

$$\hat{y}_i(t) = B_i \sin(\omega_i t + \varphi_i) \qquad B_i, \varphi_i \text{ are unknown}$$

} they're equal

otherwise :

$$\hat{y}_i(t) = a_i \sin(\omega_i t) + b_i \cos(\omega_i t) \qquad a_i, b_i \text{ are unknown}$$

for our purpose this version is more useful (the parameters here are linear)

The unknown parameters are a_i and b_i and we can find them by parametric identification:

$$\{\hat{a}_i, \hat{b}_i\} = \arg\min_{\{a_i, b_i\}} J_N(a_i, b_i)$$

$$J_N(a_i, b_i) = \frac{1}{N}\sum_{t=1}^{N}\left(y_i(t) - a_i \sin(w_i t) - b_i \cos(w_i t)\right)^2$$

$y_i(t)$: measured noisy output; $a_i \sin(w_i t) - b_i \cos(w_i t)$: modeled output

$y_i(t) - a_i \sin(w_i t) - b_i \cos(w_i t)$: model error

$J_N(a_i, b_i)$: sample variance of modelling error

(**note**: it's quadratic w.r.t. a_i and b_i)

since it's quadratic

$\Rightarrow$ we can solve "one-shot" and explicitely the optimization problem:

$$\begin{cases} \frac{\partial J_N}{\partial a_i} = \frac{2}{N}\sum_{t=1}^{N}\left(-\sin(w_i t)\right)\left(y_i(t) - a_i \sin(w_i t) - b_i \cos(w_i t)\right) = 0 \\ \frac{\partial J_N}{\partial b_i} = \frac{2}{N}\sum_{t=1}^{N}\left(-\cos(w_i t)\right)\left(y_i(t) - a_i \sin(w_i t) - b_i \cos(w_i t)\right) = 0 \end{cases}$$

$$\Rightarrow \begin{bmatrix} \sum_{t=1}^{N}\sin(w_i t)^2 & \sum_{t=1}^{N}\sin(w_i t)\cos(w_i t) \\ \sum_{t=1}^{N}\sin(w_i t)\cos(w_i t) & \sum_{t=1}^{N}\cos(w_i t)^2 \end{bmatrix} \begin{bmatrix} a_i \\ b_i \end{bmatrix} = \begin{bmatrix} \sum_{t=1}^{N} y_i \sin(w_i t) \\ \sum_{t=1}^{N} y_i \cos(w_i t) \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} \hat{a}_i \\ \hat{b}_i \end{bmatrix} = \boxed{\quad}^{-1} \cdot \boxed{\quad}$$

LEAST SQUARES ESTIMATION of parameters a_i, b_i

At this point we prefer to "go back" to a "sin-only" form (B_i, φ_i) of the estimated sinusoid:

$$\hat{B}_i \sin(w_i t + \hat{\varphi}_i) = \hat{a}_i \sin(w_i t) + \hat{b}_i \cos(w_i t)$$

expanded:

$$\hat{B}_i \sin(w_i t)\cos(\hat{\varphi}_i) + \hat{B}_i \cos(w_i t)\sin(\hat{\varphi}_i)$$

$$\Rightarrow \begin{cases} \hat{B}_i \cos(\hat{\varphi}_i) = \hat{a}_i \\ B_i \sin(\hat{\varphi}_i) = \hat{b}_i \end{cases}$$

$$\frac{\hat{b}_i}{\hat{a}_i} = \frac{\sin(\hat{\varphi}_i)}{\cos(\hat{\varphi}_i)} = \tan(\hat{\varphi}_i) \Rightarrow \boxed{\hat{\varphi}_i = \arctan\left(\frac{\hat{b}_i}{\hat{a}_i}\right)}$$

$$\boxed{\hat{B}_i = \frac{\frac{\hat{a}_i}{\cos(\hat{\varphi}_i)} + \frac{\hat{b}_i}{\sin(\hat{\varphi}_i)}}{2}}$$

$$\Rightarrow y_i(t) \approx \hat{B}_i \sin(w_i t + \hat{\varphi}_i)$$

(approximated! we have (in reality) pure sinusoid + noise)

Repeating the experiment and pre-processing for #1, #2, .., #H we obtain:

$$\{\hat{B}_1, \hat{\varphi}_1\}, \{\hat{B}_2, \hat{\varphi}_2\}, \ldots, \{\hat{B}_H, \hat{\varphi}_H\}$$

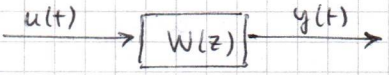
And so we can obtain:

$$\{\hat{B}_1, \hat{\varphi}_1\} \longrightarrow \frac{\hat{B}_1}{A_1} e^{j\hat{\varphi}_1}$$

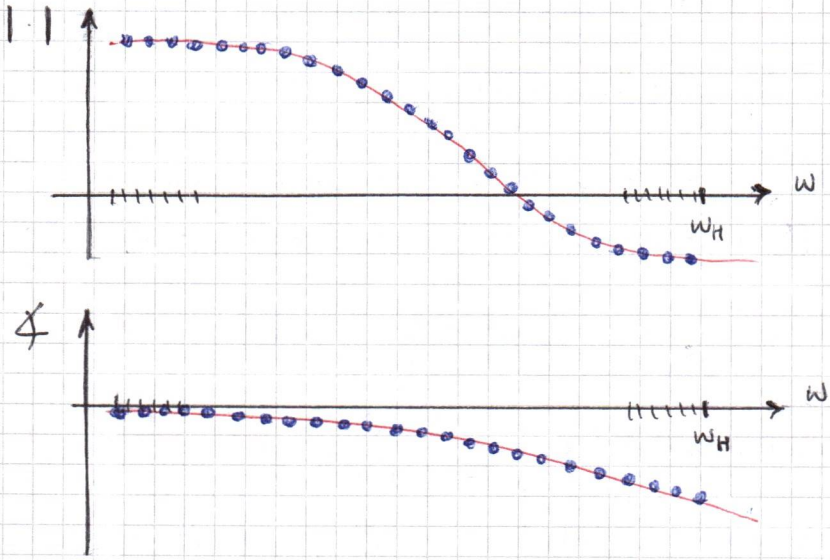
$$\vdots \qquad\qquad \vdots$$

$$\{\hat{B}_H, \hat{\varphi}_H\} \longrightarrow \frac{\hat{B}_H}{A_H} e^{j\hat{\varphi}_H}$$

H complex numbers which are the estimated H points of the frequency response of the transfer function $W(z)$ from input $u(t)$ to output $y(t)$ of the system:



Graphically:



•••: estimated point-wise BODE plot of the transfer function of the system

——: estimated transfer function (that we get only at the end of the procedure)

more precisely → "frequency response of the estimated model: $W(e^{jw}; \hat{\theta})$"

At the end of **step 1.** we have a frequency-domain dataset (H values) representing H estimated points of the frequency response of the system.

Step 2. : model class selection (we select a transfer function)

$$M(\theta): \quad W(z;\theta) = \frac{b_0 + b_1 z^{-1} + \dots + b_p z^{-p}}{1 + a_1 z^{-1} + \dots + a_n z^{-n}} \qquad z^{-1}$$

parametric model

$$\theta = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \\ b_0 \\ b_1 \\ \vdots \\ b_p \end{bmatrix}$$

Remark: as usual, we have the problem of order selection (n and p). (If we can we should use the cross-validation approach, or more simply, the visual inspection of the fitting of Bode-plots).

Step 3. : we need a new performance index (frequency domain, not time domain)

$$J_H(\theta) = \frac{1}{H} \sum_{i=1}^{H} \left(W(e^{jw_i}; \theta) - \frac{\hat{B}_i}{A_i} e^{j\hat{\varphi}_i} \right)^2 \qquad , \quad J_H : \mathbb{R}^{p+n} \longrightarrow \mathbb{R}^+$$

$W(e^{jw_i}; \theta)$: modeled frequency response

$\frac{\hat{B}_i}{A_i} e^{j\hat{\varphi}_i}$: measured frequency response

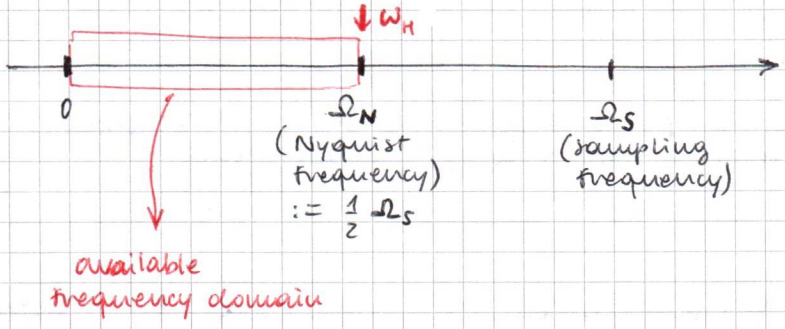
$J_H(\theta)$: estimated error variance of frequency response fitting

Step 4. : optimization.

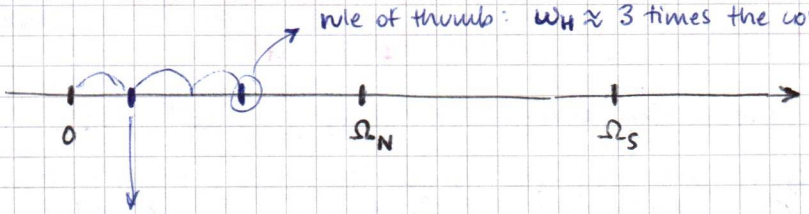
$$\hat{\theta} = \arg\min_{\theta} J_H(\theta)$$

Remark: usually $J_H(\theta)$ is a non quadratic and non convex function
$\Rightarrow$ iterative optimization method are needed (gradient descent, quasi-Newt.)

Remark: Frequency bandwith selection. ($\omega_H = ?$)
Theoretically the standard "best" solution should be: H points distributed uniformal from 0 to Ω_N := **Nyquist** frequency

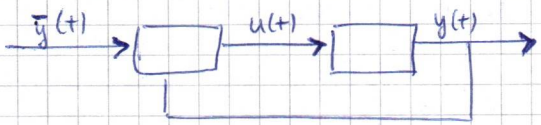


However, in practice / in real applications is better to concentrate the experimental effort in a smaller and more focused bandwith. It's generally used a rule of thumb:



(because we don't really need an accurate model till Ω_N, we need an accurate model till ω_C (for very high frequencies the control system becomes really insensitive))

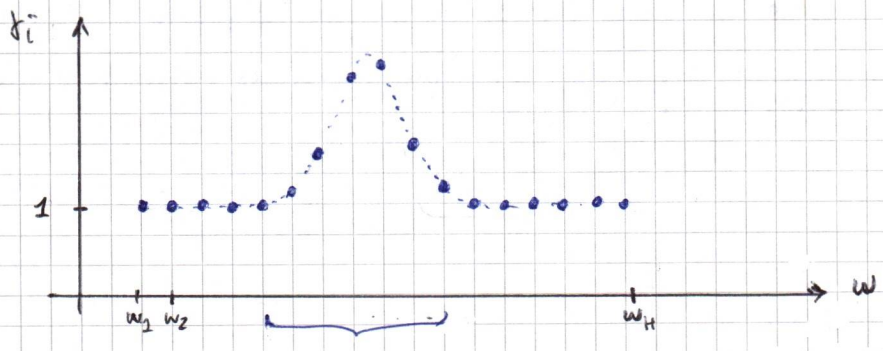
ω_C = the expected "cut off" frequency of the control system:



Example: the electronic stability control has an expected bandwith of 2 Hz.
$\Rightarrow$ we can limit our exploration to $\omega_H \simeq 6$ Hz.
(if we go beyond it's useless)

30/04

Remark: Emphasis on special frequency range.
In some case, between ω_1 and ω_H we want to be more accurate in system identification in some frequency regions (typically: around cut-off - frequency, around resonances)
$\Rightarrow$ we can use different weigets for different frequencies:



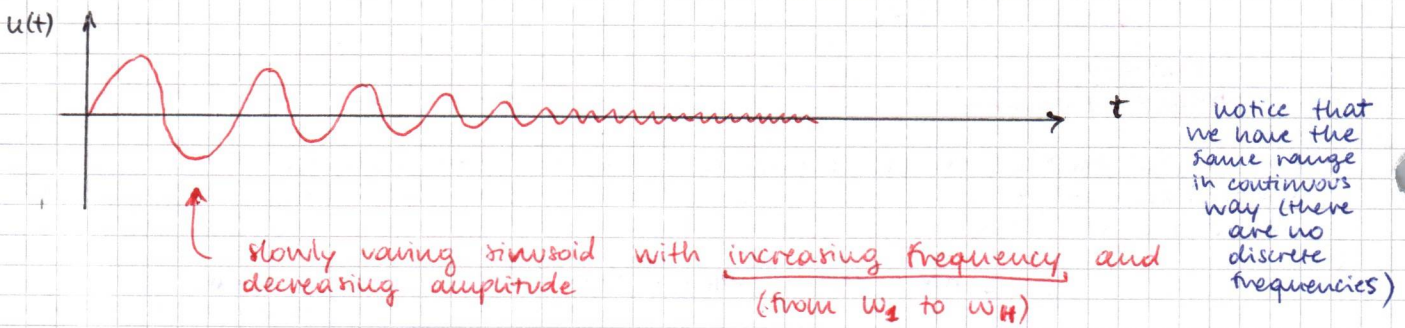
→ The performance index can be re-defined:

$$\hat{J}_H(\theta) = \frac{1}{H} \sum_{i=1}^{H} \gamma_i \left(W(e^{j\omega_i}, \theta) - \frac{\hat{B}_i}{A_i} e^{j\hat{\varphi}_i} \right)^2$$

↑ the errors are more important in the frequency regions we want to emphatize

An other trick: more dense ω_i spacing in the frequency region of special interest (not so used)

Remark: Single experiment.
Sometimes the set of **H** independent / separated single-sinusoid experiments can be replaced by a long single "sine-sweep" experiment:



This type of signal is called "SINUSOIDAL SWEEP" or "chirp signal", ...

We can at a-posteriori the signal into H pieces and then back to the standard procedure. Or, directly compute an estimation of $\hat{W}(e^{j\omega})$ as a ratio of the output and input spectra:

$$\hat{W}(e^{j\omega}) \approx \frac{\hat{\Gamma}_y(e^{j\omega})}{\hat{\Gamma}_u(e^{j\omega})} = \frac{\text{est. spectrum of the output}}{\text{est. spectrum of the input}}$$

and then we can fit the estimated $\hat{W}(e^{j\omega})$ with the model frequency response $W(e^{j\omega}, \theta)$ in the performance index (step. 3.)
(this experiment is quicker but has usually a lower signal-to-noise-ratio)

Comparison (pros and cons) between time domain (ARMAX) and frequency domain parametric methods.

frequency domain methods:

- pros:
  - robust and very reliable (we put a lot of energy on each sinusoid so the frequency response estimation is very reliable
  - intuitive (easy to understand)
  - consistent with control-design method (usually in frequency domain)
- cons:
  - more demanding for the experiments
  - no noise model is estimated, unlike ARMAX models

Notice that time domain and frequency domain methods should provide the same results if they're done correctly.

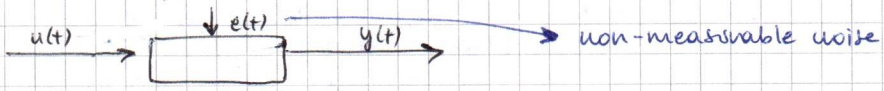
CHAPTER 3

(Kalman filter : software sensing in feedback)

we use algorithms instead of using physical sensoring

Preliminary **remark**: in MIDA 1 we have mostly used I/O (transfer function) representation:

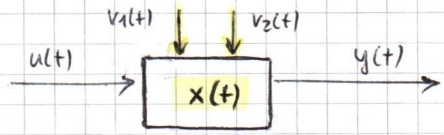
$$y(t) = \frac{B(z)}{A(z)} u(t-k) + \frac{C(z)}{A(z)} e(t) \qquad e \sim WN$$



Kalman-filter theory is fully based on state space representation:

$$\begin{cases} x(t+1) = Fx(t) + Gu(t) + v_1(t) \\ y(t) = Hx(t) + Du(t) + v_2(t) \end{cases} \qquad \begin{matrix} v_1 \sim WN \\ v_2 \sim WN \end{matrix}$$

(stochastich description : we have noise)



new: • second WN : we use a 2 noises description
• internal state : we are interested in states

Motivations and goals of Kalman-filter theory.
Given* a model description $\{F, G, H\}$ + noises variancies, with Kalman filter theory we can address the following problems:

* not a system identification technique !

moreover notice that we don't need recorded data (we didn't say "we start from $u(1), u(2), \ldots$ or $y(1), y(2), \ldots$")

1. K-steps ahead prediction of output : $\hat{y}(t+k|t)$

prediction horizon (k-steps ahead in the feature) → $t+k$; t ← present time (newest available data)

We already solved the problem in MIDA 1.

2. K-steps ahead prediction of state : $\hat{x}(t+k|t)$
(new problem since it was impossible for ARMAX)
(in ARMAX model there is no internal state)

at time t we have:
$y(t), y(t-1), y(t-2), \ldots$,
$u(t), u(t-1), u(t-2), \ldots$

3. find the FILTER of the state : $\hat{x}(t|t)$

$x(t)$ is unmeasured

given data $y(t)$, and past, $u(t)$ and past, the estimation is made at the same time
(notice that $\hat{y}(t|t)$ is non-sense : $\hat{y}(t|t) = y(t)$, and it's measured)

This problem in practice is software sensing : most important problem solved by Kalman-filter (indeed kalman-filter is named "filter")

4. **Gray box**** system identification (side benefit of kalman-filter, not the key objective of kalman filter)
(see chapter 5)

we go back to recorded datasets

** **gray box:**
we have a recorded data set: $\{y(1), .., y(N)\}$, $\{u(1), .., u(N)\}$ and the model structure (F, G, H) but with some un-known parameters:

$$\begin{cases} x(t+1) = F(\theta)\,x(t) + G(\theta)\,u(t) \\ y(t) = H(\theta)\,x(t) \end{cases}$$

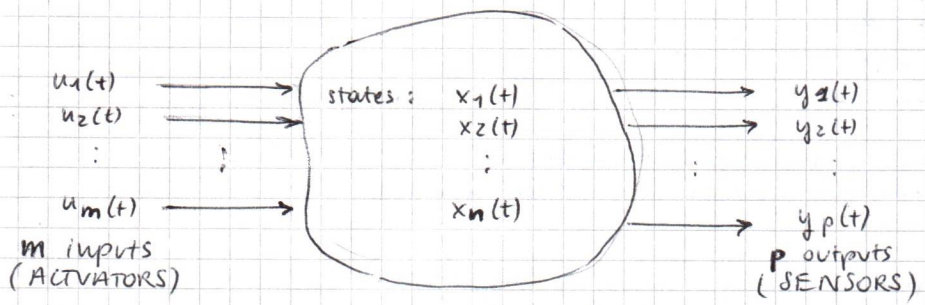
θ is a vector of some un-known parameters of the model.

Example: $F = \begin{bmatrix} 1 & 1/2 \\ a & 1/4 \end{bmatrix}$, $G = \begin{bmatrix} 1 \\ b \end{bmatrix}$, $H = [c, 1/2]$

$\longrightarrow$ $\theta = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$ $\Longrightarrow$ with kalman-filter we get (from data):

$$\hat{\theta}_N = \begin{bmatrix} \hat{a}_N \\ \hat{b}_N \\ \hat{c}_N \end{bmatrix}$$

Why kalman-filter is so useful?
Dynamical systems have this layout:



MIMO system
(**m** inputs, **p** outputs, **n** states)

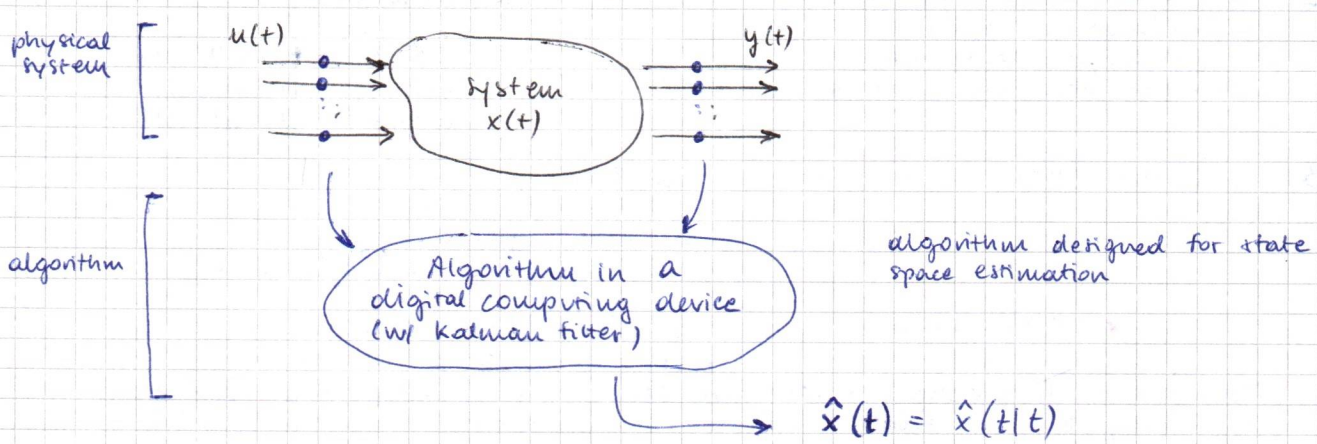
Key problem: usually $p \ll n$ $\Longrightarrow$ physical sensors are much less than system states.
Why?
- cost
- cables, power supplies
- maintenance (faults, degradation, ..)

] the physical sensors are expensive ($)

$\Longrightarrow$ not all the states are measured.
It is useful to have full measurement (physical or software) of states because we want to make:
- control design (state feedback)
- monitoring (fault detection - predictive maintenance, ..)

We solve the problem in software sensing way.

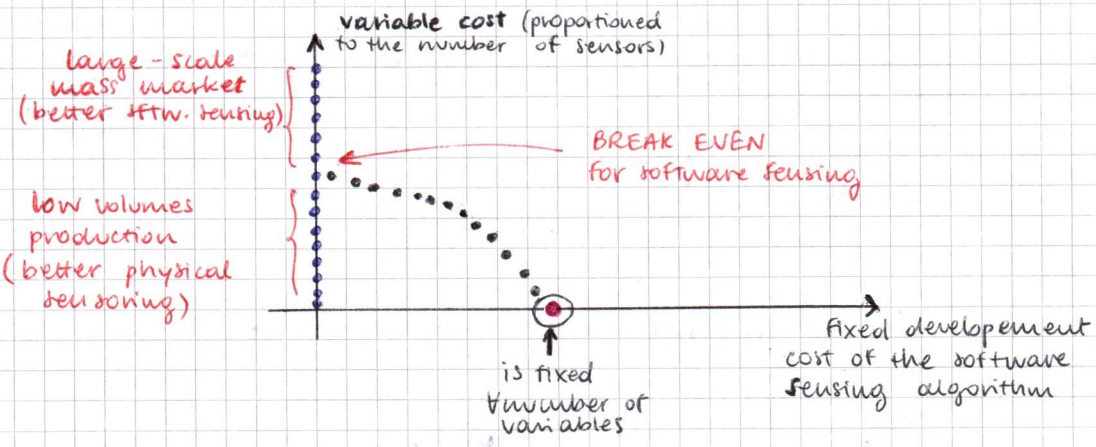
Software sensing (virtual sensing)



Dilemma of the design:
when using software sensing and when use physical sensing?

Notice that:

- in some cases there is no option (not feasible installation of the physical sensoring ⟹ software sensing)
- in most cases both options are viable
  ⟹ variable versus fixed cost:



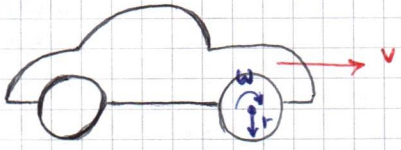
4/05

Key questions for software sensing:

1. Is software sensing feasible? We can make a test to check if it is feasible: the test is the observability of the states from measured outputs
2. Quality of estimation error? ("Noise" of measurement for sftw. sens.?)

Examples of software sensing

- Slip estimation for ABS/traction control



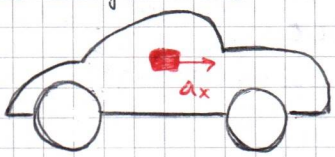
$$\lambda = \frac{v - wr}{v}$$

Problem: estimation of v.
The measure of v can be done:

- by optical sensors
- by GPS-estimated speed

(disturbancies of dust/tunnels (lost connection),..)

Both have a problem of availability (which is not guaranteed)
⟹ physical sensoring is not an option for industrial production
⟹ we need to make an estimation for the speed
Intuitive solution:
install a longitudinal accelerometer (a_x) and integrate it:



$$\hat{v}(t) = \int a_x(t)\,dt \quad \Longleftrightarrow \quad a_x(t) \rightarrow \boxed{\frac{1}{s}} \rightarrow \hat{v}(t)$$

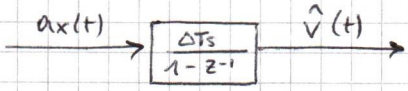
(continue time domain)

In the discrete time domain we can make a discretization using an approximation of derivative (Eulero forward method)

$$\frac{dv(t)}{dt} = a(t) \Longrightarrow \frac{dv(t)}{dt} \approx \frac{v(t+1) - v(t)}{\Delta T_s} = a_x(t)$$

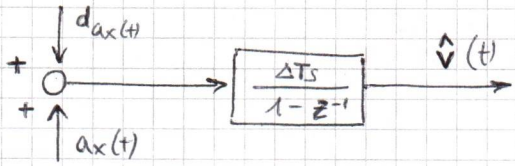
ΔT_s: sampling interval (e.g. 10 ms)

$$\Longrightarrow \quad \hat{v}(t) = \hat{v}(t-1) + \Delta T_s \, a_x(t-1)$$

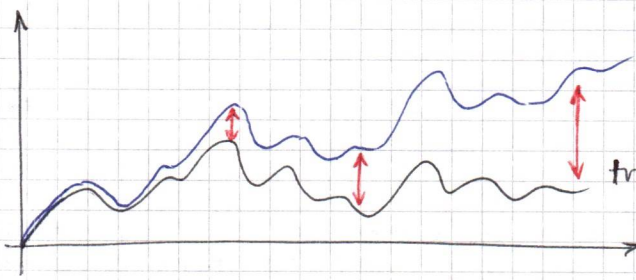


DISCRETE TIME INTEGRATOR

Unfortunately the measured signal is not $a_x(t)$ but $a_x(t) + \underbrace{d_{a_x(t)}}_{\text{measurement noise}}$ (which is the noise) :



Integrating noise generate a **drift** (since thtegrator is not asymptotically stable system).



$\hat{v}(t)$ obtained by integration : we accumulate noise, after a few seconds it becomes completely unreliable

In continuous time we have :

x = pole (0,0)

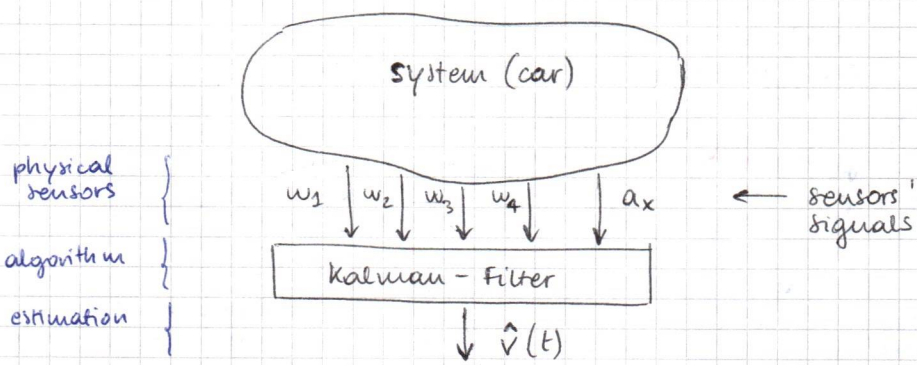
$\Rightarrow$ simply stable (not asymptotically)

In discrete time :

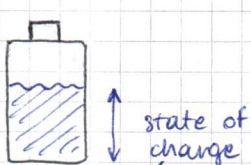
x = pole (1.0)

$\Rightarrow$ simply stable (border of asympt. stable and unstable)

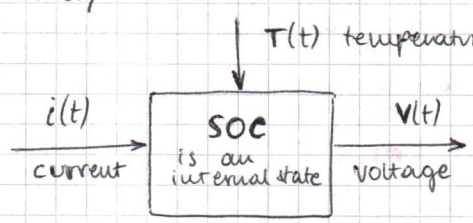
The solution is to use a Kalman-filter algorithm :



- State of charge estimation of a battery



state of charge (SOC)

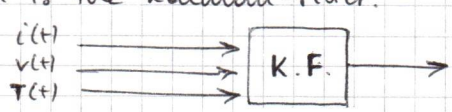


$$SOC(t) = 1 - \frac{\int i(t)\,dt}{I}$$ $\longrightarrow$ the integral starts at 100 % of SOC

I : total amount of current that can be extracted by the user of the battery

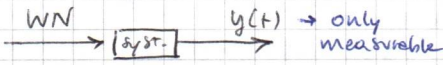
$$0 \leq SOC(t) \leq 1$$

Again, this solution is not feasible because we integrate noise on $i(t)$. The solution is the Kalman-filter.



We'll present K.F. starting from a BASIC System:

- no external inputs (~~$Gu(t)$~~) $\Rightarrow$ time series:
- linear system
- time invariant system



Then, after we describe the solution of K.F. for the basic system, we'll make the extensions to a more general system (in particular $Gu(t)$ (aka when we can use inputs to modify the system)).

Detailed description of the **basic system**:

$$\begin{cases} x(t+1) = Fx(t) + (Gu(t)) + v_1(t) & \leftarrow \text{state equation} \\ y(t) = Hx(t) + v_2(t) & \leftarrow \text{output equation} \end{cases} := S$$

$$\text{state: } x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_n(t) \end{bmatrix} \quad \left(u(t) = \begin{bmatrix} u_1(t) \\ u_2(t) \\ \vdots \\ u_m(t) \end{bmatrix} \right) \quad y(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \\ \vdots \\ y_p(t) \end{bmatrix}$$

System with n states, (m inputs), p outputs (MIMO system) (multiple input, multiple output)

If $m = p = 1 \Rightarrow$ SISO system.

Let's analyze v_1 and v_2:

$\underline{v_1(t)} \sim WN(0, V_1)$, $v_1(t) = \begin{bmatrix} v_{11}(t) \\ v_{12}(t) \\ \vdots \\ v_{1n}(t) \end{bmatrix}$ (n: consistent with the input)

(vector WN)

$v_1(t)$ is called <u>state noise</u> or <u>model noise</u> (it accounts for the modeling errors of the state equation of the system) (whenever we have a model we have errors. Most of the time because we neglect some small linearities)

1. $\mathbb{E}[v_1(t)] = \underline{0} = [0 \dots 0]^T$
2. $\mathbb{E}[v_1(t)\, v_1(t)^T] = V_1$ ($n \times n$ matrix (covariance matrix), it's symmetric and semi-definite positive $\Rightarrow V_1 \geq 0$)
3. $\mathbb{E}[v_1(t) \cdot v_1(t-\tau)^T] = 0 \quad \forall t \ \forall \tau \neq 0$ (<u>key feature of WN</u>) there is no correlation of this vector of signals at different times

$\underline{v_2(t)} \sim WN(0, V_2)$, $v_2(t) = \begin{bmatrix} v_{21}(t) \\ v_{22}(t) \\ \vdots \\ v_{2p}(t) \end{bmatrix}$ (p: consistent with the output)

$v_2(t)$ is called <u>output noise</u> or <u>measurement (sensor) noise</u>. (noise that we find on a sensor)

1. $\mathbb{E}[v_2(t)] = \underline{0}$
2. $\mathbb{E}[v_2(t)\, v_2(t)^T] = V_2$ ($p \times p$ symmetric covariance matrix, $V_2 \geq 0$. However we make the assumption that V_2 is definite positive $\Rightarrow V_2 > 0$) (we'll use this in Riccati equation of Kalman filter)
3. $\mathbb{E}[v_2(t)\, v_2(t-\tau)] = 0 \quad \forall t \ \forall \tau \neq 0$

Now we need to make assumptions on the relationship between $v_1(t)$ and $v_2(t)$:

$$\mathbb{E}[v_1(t)\, v_2(t-\tau)^T] = \underset{n \times p \text{ matrix}}{V_{12}} = \begin{cases} 0 & \tau \neq 0 \\ \text{can be } \neq 0 & \tau = 0 \end{cases}$$

in the same time there can be some correlation

Notice that V_{12} is a $n \times p$ matrix (cross variance matrix).

We assume that $v_1(t)$ and $v_2(t)$ can be correlated (only at the same time). In practice $V_{12} = 0$ is the most common assumption.

Since the system S is dynamic $\Rightarrow$ we need to define the initial cond.:

$$\mathbb{E}[x(1)] = x_0 \quad (n \times 1 \text{ vector})$$

$$\mathbb{E}[(x(1) - x_0)(x(1) - x_0)^T] = P_0 \geq 0 \quad \left(\begin{array}{c} n \times n \text{ symmetric} \\ \text{semi-definite positive} \\ \text{matrix (covariance matrix)} \end{array}\right)$$

probabilistic description of the initial condition of S

Note that if $P_0 = 0 \Rightarrow$ the initial state is perfectly known.

We finally assume that the two noises $v_1(t)$ and $v_2(t)$ are uncorrelated with the initial state:

$$\left.\begin{array}{l} x(1) \perp v_1(t) \\ x(1) \perp v_2(t) \end{array}\right\} \text{ technical assumption needed to guarantee the theoretical optimality of K.F.}$$

Now we present the basic solution: 1 step ahead prediction ($\hat{x}_1(t+1|t)$, $\hat{y}(t+1|t)$) for the basic system.
(later on we'll see all the extensions)

KALMAN-FILTER FOR THE BASIC SOLUTION OF THE BASIC SYSTEM

$$\hat{x}(t+1|t) = F\hat{x}(t|t-1) + K(t)\,e(t) \qquad \text{state equation}$$

$$\hat{y}(t|t-1) = H\hat{x}(t|t-1) \qquad \text{output equation}$$

$$e(t) = y(t) - \hat{y}(t|t-1) \qquad \text{output prediction error}$$

$$K(t) = (FP(t)H^T + V_{12})(HP(t)H^T + V_2)^{-1} \qquad \text{equation of the gain of KF}$$

$$P(t+1) = (FP(t)F^T + V_1) - (FP(t)H^T + V_{12})(HP(t)H^T + V_2)^{-1}(FP(t)H^T + V_{12})^T$$

↳ difference RICCATI equation (DRE)

These equations must be completed with 2 initial conditions (since 2 eqs. are dynamical equations):

- State equation: $\hat{x}(1|0) = \mathbb{E}[x(1)] = x_0$
- DRE: $P(1) = \text{Var}(x(1)) = P_0$

Remark (structure of $K(t)$ and DRE): notice that $K(t)$ and DRE have a "block set" structure having the form:

$$\square \cdot P(t) \cdot \diamondsuit^T + \boxed{\text{noise matrix}}$$

We have 3 different types of blocks:

- "state": $FP(t)F^T + V_1$ (F, V_1 refer to **state** equation)
- "output": $HP(t)H^T + V_2$ (H, V_2 refer to **output** equation)
- "mix": $FP(t)H^T + V_{12}$ (F state, H output, V_{12} relationship between V_1 and V_2)

Gain $K(t) = (\text{mix}) \cdot (\text{output})^{-1}$

DRE $P(t+1) = (\text{state}) - (\text{mix})(\text{output})^{-1}(\text{mix})^T$

Remark (on the Riccati equation): Riccati equation is a special type of non-linear matrix difference equation (it was studied by the count Riccati in 1700 and it turned out to be useful in '60 for the Kalman-filter). Notice that DRE is an autonomous, non linear, discrete-time, multivariable system described by a non-linear difference matrix equation.

$$\text{DRE}: \begin{cases} P(t+1) = f(P(t)) + \cancel{\text{forcing inputs}} \\ P(1) = P_0 \end{cases}$$

Remark (existence of DRE): in order to guarantee the existence of DRE $\forall t$, the only critical part is the inversion of the "output" block:

$$(\underbrace{H P(t) H^T}_{\text{pxp matrix always } \geq 0 \text{ (semi def. positive) but not guaranteed to be invertible.}} + \underbrace{V_2}_{\text{we make the assumption that } V_2 > 0 \text{ (def. positive)}})^{-1}$$

$\Longrightarrow$ the overall block is definite positive (thanks to $V_2 > 0$)

$\Longrightarrow$ the block is invertible

Remark (on the meaning of $P(t)$): it can be proven that $P(t)$ has a very important meaning:

$$P(t) = \text{Var}(x(t) - \hat{x}(t|t-1))$$
$$\triangleq \mathbb{E}[(x(t) - \hat{x}(t|t-1))(x(t) - \hat{x}(t|t-1))^T]$$

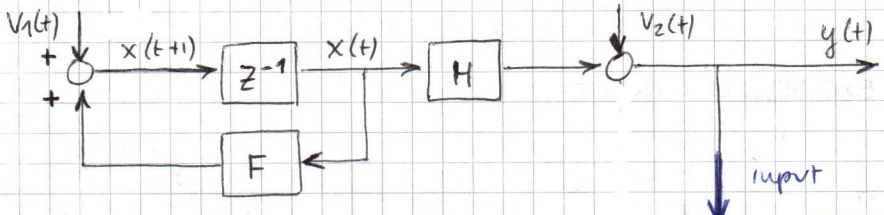
$P(t)$ is the covariance of the 1 step prediction error of the state

$P(t)$: square symmetric semi definite positive (≥ 0) matrix (covariance matrix)

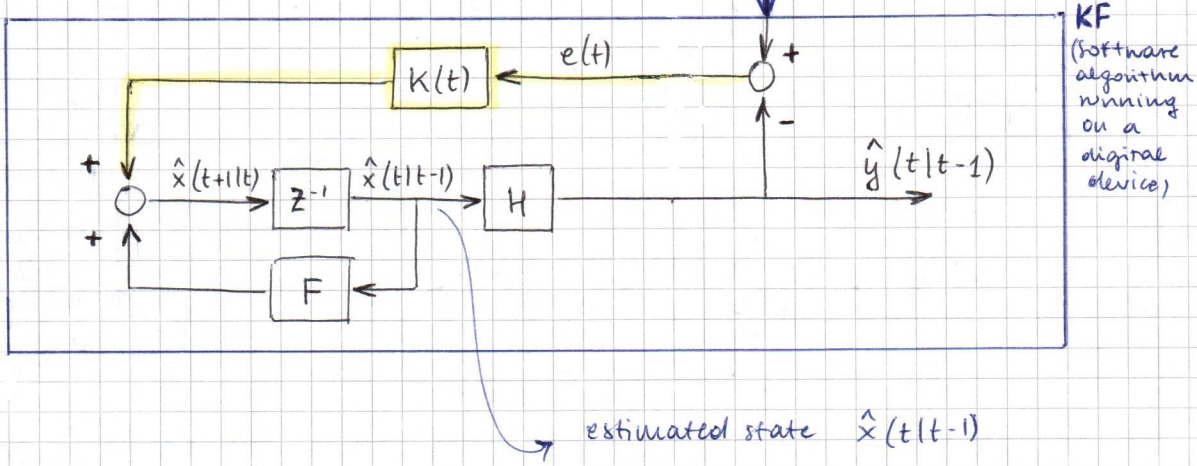
5/05

BLOCK - SCHEME REPRESENTATION OF K.F.

S:



KF:



Idea behind **KF** is simple and intuitive:

- we make a simulated replica of the system (without noises v_1 and v_2 non measurable) (they've hidden so we can't simulate them)
- we compare the true measured output with the estimated / simulated output $\hat{y}(t|t-1)$
- we make corrections on KF main equation proportional (with gain $K(t)$) to the output error in order to keep KF as close as possible to the system.

$\Longrightarrow$ we "extract" the state estimation $\hat{x}(t|t-1)$.

KF is a feedback system. The feedback here is not used for control (which is the typical thing we use feedbacks for) but for estimation.

This general idea/structure was known before KF developement. It's called "state observer".
The fundamental contribution of Kalman was to find the optimal gain $\mathbf{K(t)}$.
$K(t)$ is not a simple scalar gain (cannot be tuned empirically) but is a (may be very large) a $n \times p$ matrix $\Longrightarrow$ matrix of gains that can be very large and difficult to be tuned.
The selection of the matrix $\mathbf{K(t)}$ is very critical.
Why?

- If $K(t)$ is "too **small**" $\Longrightarrow$ the estimation is not optimal because we are "under-exploiting" the information in $y(t)$
- If $K(t)$ is "too **large**" $\Longrightarrow$ there is a risk of "over-exploiting" $y(t)$ $\Longrightarrow$ noise amplification $\Longrightarrow$ even risk of instability.

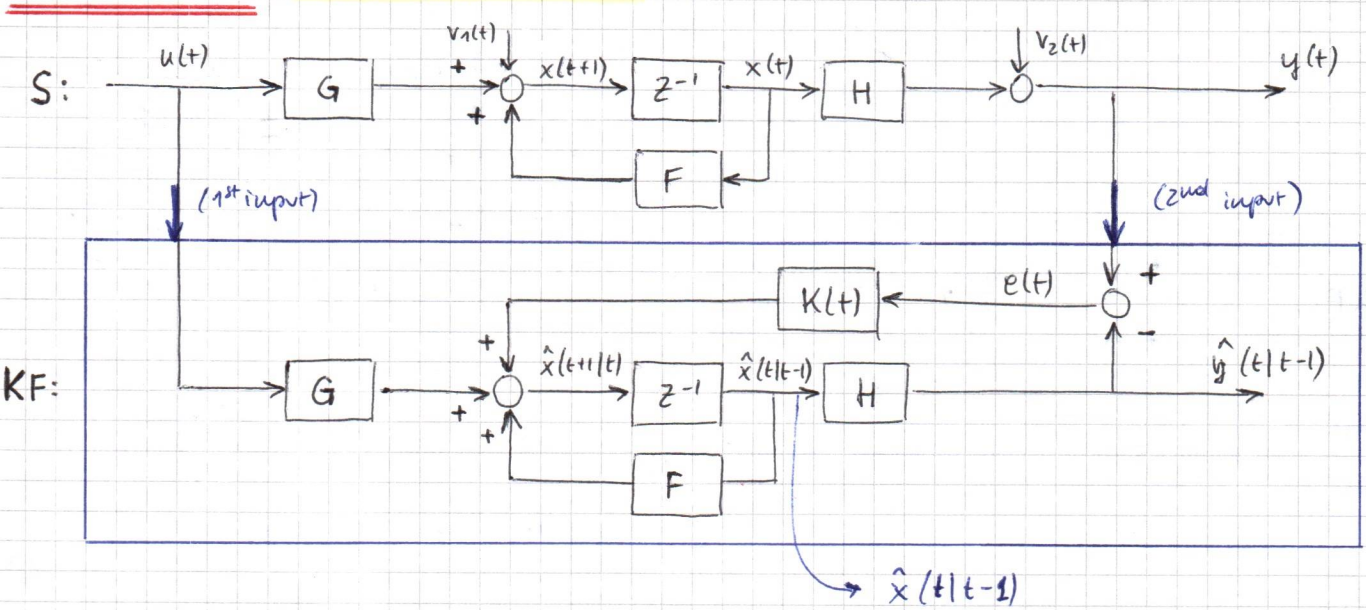
Kalman provided a theoretical optimal matrix gain $K(t)$ with a constructive / direct design from system matrices (F, G, H, V_1, V_2).

$\Longrightarrow$ the design of **KF** does not require a training dataset ("given the data $y(\cdot)$..") but a complete model of the system:

- F, G, H matrices (usually obtained with a white box physical modelling of the system **S**)
- V_1, V_2, V_{12} (noise matrices)
  - easily built from sensors specifications
  - "model noise": much more difficult to be designed (V_1 is the most critical design parameter of **KF**)

EXTENSIONS OF BASIC PROBLEM OF BASIC SYSTEMS

Extension 1.: EXOGENOUS INPUT



Notice that $K(t)$ remains the same (same formula, same value) because $P(t)$ is the covariance of the estimation (prediction) error at $x(t)$ and remains the same because $Gu(t)$ because does not introduce any additional noise or uncertainties to the system ($Gu(t)$ is totally known "deterministic" signal).

Extension 2. : MULTI-STEP PREDICTION

Assume that $\hat{x}(t+1|t)$ is known from the basic solution. We can simply obtain a multi-step prediction as:

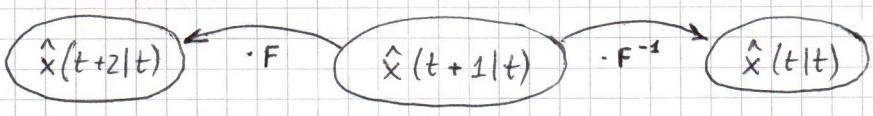
$$\hat{x}(t+2|t) = F\hat{x}(t+1|t)$$
$$\hat{x}(t+3|t) = F\hat{x}(t+2|t) = F^2\hat{x}(t+1|t)$$
$$\vdots$$

$$\boxed{\begin{aligned}\hat{x}(t+k|t) &= F^{k-1}\hat{x}(t+1|t)\\ \hat{y}(t+k|t) &= H\hat{x}(t+k|t)\end{aligned}}$$

← valid also with exogenous system (because $u(t)$ is known up to time t) but we need an additional assumption: $u(t)$ must be zero mean

Extension 3. : FILTER ($\hat{x}(t|t)$)

(we consider filter only with **no** exogenous input)



$$\longrightarrow \boxed{\hat{x}(t|t) = F^{-1}\hat{x}(t+1|t)}$$

However this formula can be used only if F is invertible.
If F is <u>not</u> invertible the filter can be obtained with a specific "FILTER" formulation of KF : "**KF in the filter form**" (not in the prediction form)

$$\begin{aligned}\hat{x}(t|t) &= F\hat{x}(t-1|t-1) + K_0(t)\,e(t)\\ \hat{y}(t|t-1) &= H\hat{x}(t|t-1)\\ e(t) &= y(t) - \hat{y}(t|t-1)\\ K_0(t) &= (P(t)H^T)(HP(t)H^T + V_2)^{-1}\\ &\text{DRE (un-changed equation)}\end{aligned}$$

+ initial condition: $\hat{x}(1|1) = x_0$

These equations are valid under the restrictive assumptions of $V_{12} = 0$.

Remark (on $K(t)$ and $K_0(t)$ when $V_{12} = 0$):

Gain of KF in predictive form: $K(t) = (FP(t)H^T)(HP(t)H^T + V_2)^{-1}$

Gain of KF in filter form: $K_0(t) = (\boxed{X}P(t)H^T)(HP(t)H^T + V_2)^{-1}$

Extension 4. : TIME-VARYING SYSTEM

$$S: \begin{cases} x(t+1) = F(t)x(t) + G(t)u(t) + V_1(t)\\ y(t) = H(t)x(t) + V_2(t)\end{cases}$$

example: aging of the system

$\{F, G, H\}$ (LTI) linear time invariant $\longrightarrow$ $\{F(t), G(t), H(t)\}$ (LTV) linear time variant

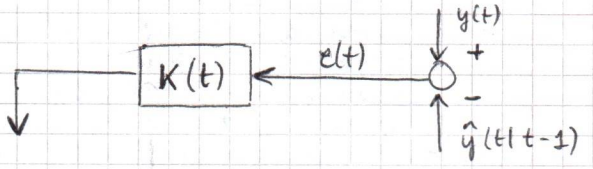
In this case KF equations are exactly the same! Just replace $F \to F(t)$, $G \to G(t)$, $H \to H(t)$ in the 5 equations of KF.
(KF, as is, is valid also for LTV systems)

Extension 5. : EXTENSION TO NON-LINEAR SYSTEMS

It's an extension much more complicated called "Extended Kalman filter (**EKF**)" (we'll see it later).

Now we consider the issue of "**Asymptotic solution of Kalman Filter**".

Observe that KF is not itself an LTI system, but it is an LTV system because the gain $K(t)$ is time varying.



the matrix $K(t)$ is time varying
⇓
the whole KF is a LTV system

The fact that KF is a LTV system is the source of two problems:

1. Checking the stability of KF algorithm is very difficult because the stability check of LTV system is not simple as the stability check of an LTI system

   Note: Consider an **LTI** system: $x(t+1) = Fx(t) + Gu(t)$. The stability check consists in verifying that the eigenvalues of F are inside the unit disk.
   Consider an **LTV** system: $x(t+1) = F(t)x(t) + G(t)u(t)$.
   In this case even if all the eigenvalues of $F(t)$ are strictly inside the unit circle at **any time** the system is not guaranteed to be asymp. stable (in practice it is **if** time-variations are "slow" like in aging)

2. Computational problem: $K(t)$ must be computed at each sampling time (e.g. every 5 ms) $\Longrightarrow$ inversion of matrix $(HP(t)H^T + V_2)$ and this matrix can be large ($p \times p$ matrix)

Because of **1.** and **2.** (1. bigger problem) in real/practical applications the asymptotic version of KF is preferred.

Basic idea:

if $P(t) \xrightarrow{\text{converges}} \bar{P}$ (steady-state value of $P(t)$) $\Longrightarrow$ $K(t) \xrightarrow{\text{converges}} \bar{K}$ (steady-state "asymptotic" value of $K(t)$)

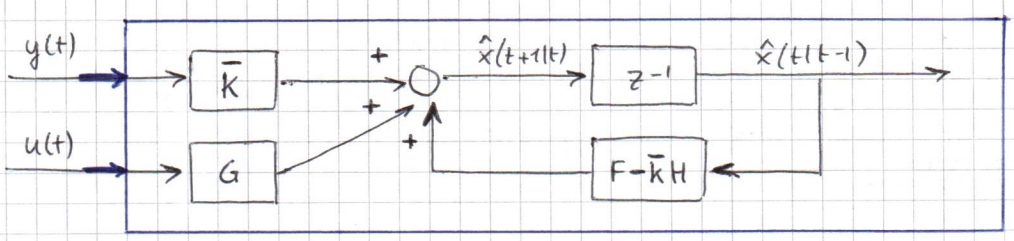
Using $\bar{K}$ $\Longrightarrow$ KF becomes an **LTI** system.

Let's analyze the asymptotic stability of KF when $\bar{K}$ is used (in this KF is LTI). (Let's assume that $\bar{K}$ does exist)

$\Longrightarrow$ analysis of the dynamics of state equation of KF:

$$\begin{aligned}
\hat{x}(t+1|t) &= F\hat{x}(t|t-1) + Gu(t) + \bar{K}e(t) \\
&= F\hat{x}(t|t-1) + Gu(t) + \bar{K}\left(y(t) - \hat{y}(t|t-1)\right) \\
&= F\hat{x}(t|t-1) + Gu(t) + \bar{K}\left(y(t) - H\hat{x}(t|t-1)\right) \\
&= (F - \bar{K}H)\,\hat{x}(t|t-1) + Gu(t) + \bar{K}y(t)
\end{aligned}$$

$(F - \bar{K}H)$: new state matrix of KF



⟹ If $\bar{K}$ does exist, the KF is asymptotically stable if and only if all the eigenvalues of $(F-\bar{K}H)$ are strictly inside the unit circle.

Notice that the stability of the **system** is related to matrix **F** and the stability of **KF** is related to matrix **$(F-\bar{K}H)$**.

⟹ KF can be asymptotically stable even if the system is unstable.

What about the existence of $\bar{K}$?

$$\bar{K} = (F\bar{P}H^T + V_{12})(H\bar{P}H^T + V_2)^{-1}$$

$\bar{K}$ exists if $\bar{P}$ exists

⟹ we need to check the convergence properties of DRE.

Remember that to find the equilibrium of a dynamical autonomous system:

Continuous time	Discrete time
$\dot{x} = f(x)$	$x(t+1) = f(x(t))$
Eq. condition: $\dot{x} = 0$	Eq. condition: $x(t+1) = x(t)$
⟹ $f(\bar{x}) = 0$	⟹ $f(\bar{x}) = \bar{x}$

this is what we need to solve because DRE is an autonomous discrete-time system:

$$\bar{P} = f(\bar{P})$$

$$\bar{P} = (F\bar{P}F^T + V_1) - (F\bar{P}H^T + V_{12})(H\bar{P}H^T + V_2)^{-1}(F\bar{P}H^T + V_{12})^T$$

this is a non linear algebraic equation known as **ALGEBRAIC RICCATI EQUATION** (A.R.E.)

If a state $\bar{P}$ solution of DRE does exist, it must be a solution of ARE
⟹ we have 3 questions:

- existence?
- convergence?
- stability?

7/05

1. **EXISTENCE**: Does DRE have semi definite positive solutions?
2. **CONVERGENCE**: If 1. is yes, does the DRE converge to $\bar{P}$? (In principle $\bar{P}$ can be an equilibrium point for DRE but not an "attractor")
3. **STABILITY**: If 1. is yes, if 2. is yes, is the corresponding $\bar{K}$ such that the KF is asymptotically stable? (All eigen $(F-\bar{K}H)$ are strictly in the unit circle?)

Answer to this questions is difficult. We need two fundamental theorems (KF asymptotich theorems). They provide sufficient conditions for 1., 2., 3.

1st Asymptotic theorem:

Assumptions:
- $V_{12} = 0$
- the system is asymptotically stable: all the eigenvalues of F are strictly inside the unit circle

Then:
- ARE has one and only one sdp solution: $\bar{P} \geq 0$
- DRE converges to $\bar{P}$ $\forall P_0 \geq 0$
- the corresponding $\bar{K}$ is such that KF is asymptotically stable

For the 2nd theorem we need to recall observability / reachability (= controllability) properties.

Observability: the pair (H, F) is observable if:

$$O = \begin{bmatrix} H \\ HF \\ \vdots \\ HF^{n-1} \end{bmatrix}$$ is full rank (rank $= n$). (observability of the state (x) from the output (y))

Controllability from noise (not from the exogenous input!):

$$x(t+1) = Fx(t) + \underbrace{(Gu(t))}_{\text{irrelevant}} + \underbrace{v_1(t)}_{\text{is an input, we need controllability from this input}} \qquad v_1 \sim WN(0, V_1)$$

Let's focus on: $x(t+1) = Fx(t) + v_1(t)$: it's always possible to factorize $V_1 = \Gamma_1 \Gamma_1^T$

$$\Longrightarrow \quad x(t+1) = Fx(t) + \Gamma w(t) \quad \Longrightarrow \quad w(t) \sim WN(0, I)$$

Check:
$$\text{Var}(\Gamma w(t)) = \mathbb{E}[\Gamma w(t) \cdot w(t)^T \Gamma^T]$$
$$= \Gamma \, \mathbb{E}[w(t) w(t)^T] \, \Gamma^T$$
$$= \Gamma \, \text{Var}(w(t)) \, \Gamma^T$$
$$= \Gamma I \Gamma^T = \Gamma \Gamma^T = V_1 = \text{Var}(v_1)$$

(Quick numerical **example**:

$$x(t+1) = \tfrac{1}{2} x(t) + v_1(t) \qquad v_1 \sim WN(0, 4)$$
$$\Longrightarrow x(t+1) = \tfrac{1}{2} x(t) + 2w(t) \qquad w \sim WN(0, 1) \qquad \Gamma = 2, \ \Gamma^2 = 4 = V_1 \ )$$

We can say that the state $\mathbf{x}$ is fully controllable / reachable from input noise $v_1(t)$ if and only if:

$$R = [\Gamma \quad F\Gamma \quad F^2\Gamma \quad \cdots \quad F^{n-1}\Gamma] \text{ is full rank } (n).$$

2nd Asymptotic theorem:

Assumptions:
- $V_{12} = 0$
- (F, H) is observable
- (F, Γ) is controllable ($V_1 = \Gamma \Gamma^T$)

Then:
- AER has 1! definite positive solution $\bar{P} > 0$
- DRE converges to $\bar{P}$ $\forall P_0 \geq 0$
- the corresponding $\bar{K}$ is such that KF is asymptotically stable

(we get more than in 1st theorem because the solution of AER is def. positive)
($\bar{P} > 0$ stronger than $\bar{P} \geq 0$)
$\Rightarrow$ if $\bar{P} > 0$ it can be invertible, for instance

These theorems are very useful in practice because we can fully avoid the convergence analysis of DRE (which is very difficult).
Remember that theorem 1 and 2 provide just sufficient conditions.

Example: (Numerical exercise on KF)

$$S: \begin{cases} x(t+1) = \frac{1}{2}x(t) + v_1(t) \\ y(t) = 2x(t) + v_2(t) \end{cases} \qquad \begin{matrix} v_1 \sim WN(0, \frac{19}{20}) \\ v_2 \sim WN(0, 1) \end{matrix} \Big\} \; v_1 \perp v_2 \Rightarrow V_{12} = 0$$

$x(t) \perp v_1$, $x(t) \perp v_2$

Question: find, if exists, the steady (asymptotic) ($\hat{x}(t+1|t)$ and $\hat{x}(t|t)$)

$n = 1 \Longrightarrow x(t) = x_1(t)$

$F = \frac{1}{2}$, $G = 0$, $H = 2$, $V_1 = \frac{19}{20}$, $V_2 = 1$, $V_{12} = 0$

we can try to use asymptotic theorems

1st step: compute **DRE**

$$P(t+1) = (FP(t)F^T + V_1) - (FP(t)H^T + V_{12})(HP(t)H^T + V_2)^{-1}(FP(t)H^T + V_{12})^T$$

$$P(t+1) = \left(\frac{1}{4}P(t) + \frac{19}{20}\right) - \frac{\left(\frac{1}{2}P(t)\cdot 2\right)^2}{4P(t)+1}$$

$$= \frac{P(t)^2 + \frac{1}{4}P(t) + \frac{19}{5}P(t) + \frac{19}{20} - P(t)^2}{4P(t)+1}$$ ← they must disappear (if not there is an error)

$$= \frac{81P(t) + 19}{80P(t) + 20}$$

$P(t+1) = f(P(t))$ (general structure of an autonomous nonlinear dynamical recursive function)

2nd step: compute and solve **ARE**

$$\bar{P} = \frac{81\bar{P} + 19}{80\bar{P} + 20} \Longrightarrow 80\bar{P}^2 + 20\bar{P} = 81\bar{P} + 19$$

$$80\bar{P}^2 - 61\bar{P} - 19 = 0$$

$$\Longrightarrow (\bar{P} - 1)(80\bar{P} + 19) = 0$$

$$\Longrightarrow \bar{P}_1 = 1, \quad \bar{P}_2 = -\frac{19}{80}$$

no interest (< 0, no physical meaning (it's a variance))

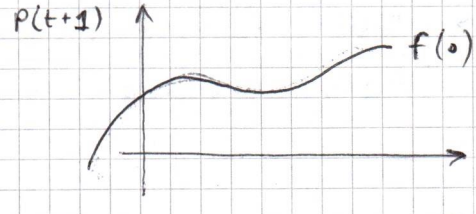
$\Longrightarrow \bar{P}_1 = 1$ 1! definitive positive solution of ARE

Question: Does the DRE converge to $\bar{P} = 1 \;\; \forall P_0 \geq 0$?
There are two methods to address this question:

- direct analysis of DRE
- use asymptotic theorems

- Direct DRE convergence analysis:

DRE: $P(t+1) = f(P(t)) \Longrightarrow$ we need to plot $f(\cdot)$ in the $P(t) - P(t+1)$ plane:

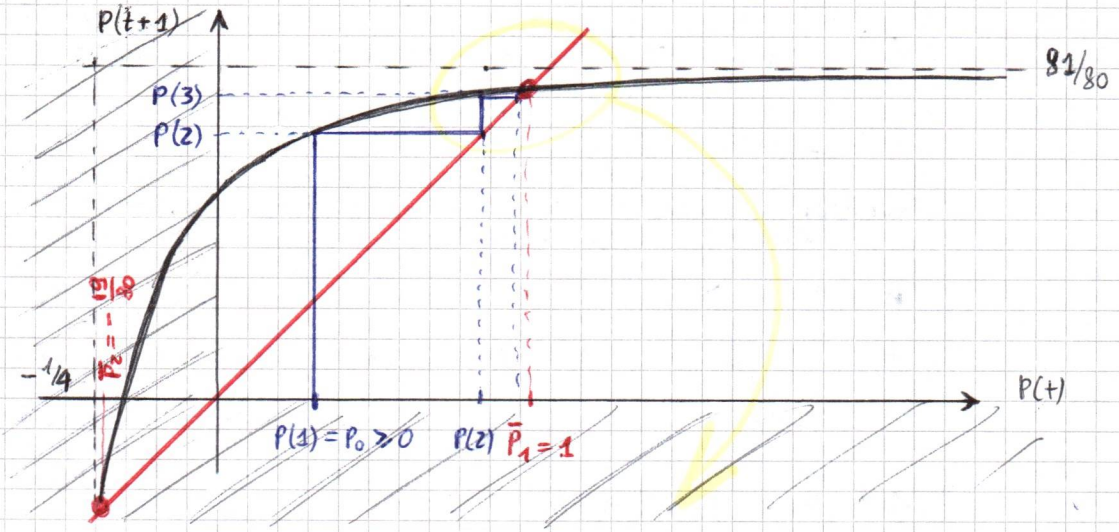


$$P(t+1) = \frac{81\,P(t) + 19}{80\,P(t) + 20}$$

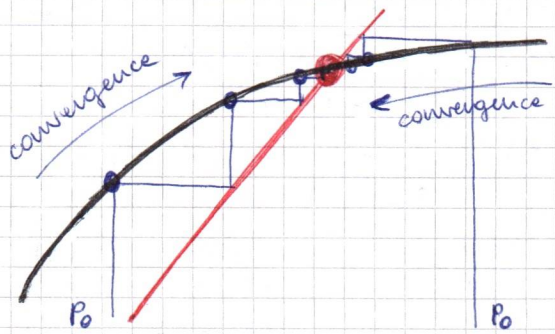
Vertical asymptotical value: $P(t) = -\frac{20}{80} = -\frac{1}{4}$

Horizontal asymptotica value: $P(t+1) = \frac{81}{80}$

$+ \begin{cases} P(t) = 0 \\ \Rightarrow P(t+1) = \frac{19}{20} \end{cases}$



Zoom:



whatever we choose as P_0, we have convergence at $\bar{P}_1 = 1$

By direct analysis of DRE analysis/inspection we can conclude that $\forall P_0 \geq 0$ DRE always converges to $\bar{P}_1 = 1$

If $n = 1 \Rightarrow$ the direct inspection is feasible, but it's very difficult for $n > 1$. With theorems we can skip this challenging steps.

- Asymptotic theorems method: (if $V_{12} \neq 0$ we **must** do direct analysis)

$V_{12} = 0$, $F = \frac{1}{2}$ ($\Rightarrow$ S is asymptotically stable)

$\Rightarrow$ theorem 1 is fullfilled

Observability matrix of (F, H): $O = [2]$, $\text{rank}(O) = 1$
$\Rightarrow$ the system is observable from output.

Controllability from noise $v_1(t)$: $V_1 = \frac{19}{20}$, $\Gamma = \sqrt{\frac{19}{20}}$, $\Gamma^2 = V_1$
Controllability of (F, Γ):
$R = [\sqrt{19/20}]$, $\text{rank}(R) = 1$ $\rightarrow$ the system is controllable from noise $v_1(t)$

$V_{12} = 0$, (F, H) observable, (F, Γ) controllable

$\Rightarrow$ theorem 2 is fullfilled

$\Rightarrow$
- ARE has 1! solution $\bar{P} > 0$ (definite positive)
- DRE $\xrightarrow{t \to \infty}$ $\bar{P}$ $\quad \forall P_0 \geq 0$
- The corresponding $\bar{k}$ makes the KF asymptotically stable

Compute $\bar{K}$:

$$\bar{K} = (F\bar{P}H^T + V_{12})(H\bar{P}H^T + V_2)^{-1}$$

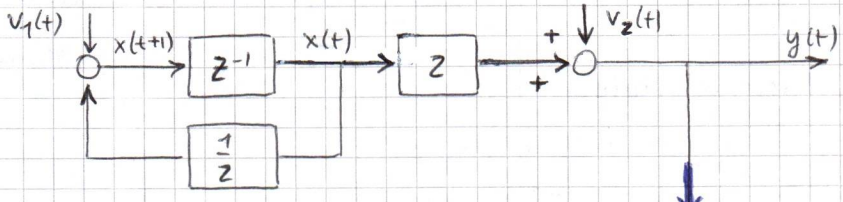
$$\bar{K} = \left(\tfrac{1}{2}\cdot 1\cdot 2 + 0\right)(2\cdot 1\cdot 2 + 1)^{-1} = \tfrac{1}{5}$$

Re-check the asymptotic stability of KF (already guaranteed by theorems):

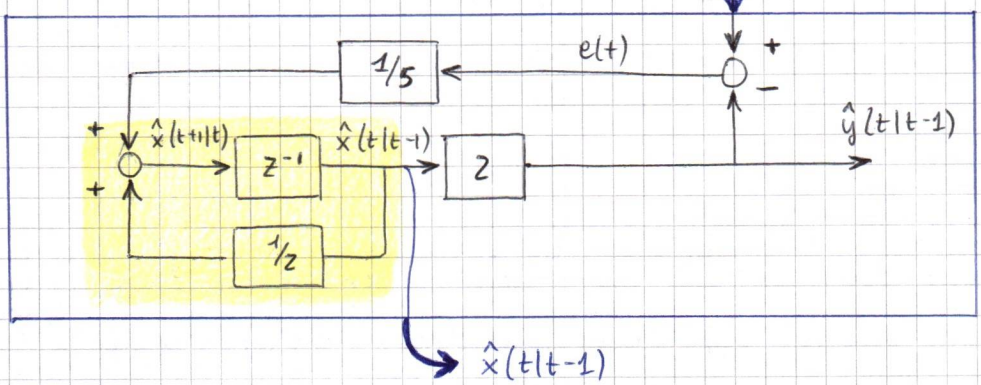
$$F - \bar{K}H = \tfrac{1}{2} - \tfrac{1}{5}\cdot 2 = \tfrac{5-4}{10} = \tfrac{1}{10} \implies$$ this eigenvalue/pole of KF is asympt. stable

Plot the block scheme of system S and KF:

S:

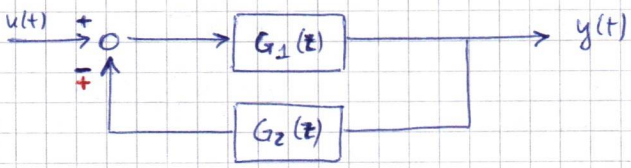


KF:



Question: find the transfer function from $y(t)$ to $\hat{x}(t|t-1)$

Recall: transfer function from block-scheme of feedback system:



Transfer function from $u(t)$ to $y(t)$?

$$y(t) = G_1(z)\left(u(t) \mp G_2(z)\, y(t)\right)$$

$$y(t)\left[1 \pm G_1(z)\, G_2(z)\right] = G_1(z)\, u(t)$$

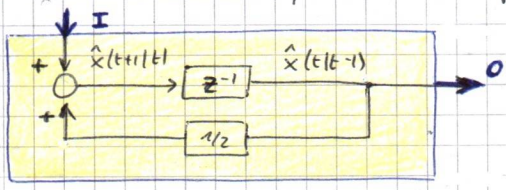
$$\implies y(t) = \frac{G_1(z)}{1 \pm G_1(z)G_2(z)}\, u(t)$$

direct line from input to output

loop function

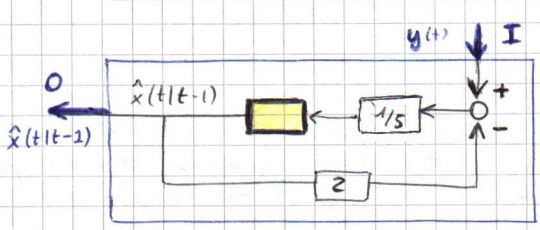
transfer function

A KF is made by 2 nested loops:



transfer function:

$$\frac{z-1}{1-\frac{1}{2}z^{-1}}$$



transfer function:

$$\frac{\frac{1}{5}\left(\frac{z^{-1}}{1-\frac{1}{2}z^{-1}}\right)}{1 + 2\cdot\frac{1}{5}\left(\frac{z^{-1}}{1-\frac{1}{2}z^{-1}}\right)} = (*)$$

$$(*) = \frac{1/5\ z^{-1}}{1 - \frac{1}{10} z^{-1}} \qquad \left(\hat{x}(t|t-1) = \frac{\frac{1}{5} z^{-1}}{1 - \frac{1}{10} z^{-1}}\, y(t) \right.$$

or in time domain:

$$\hat{x}(t|t-1) = \frac{1}{10}\, \hat{x}(t-1|t-2) + \frac{1}{5}\, y(t-1)$$

(predictor 1 step of state))

Predictor of the output:

$$\hat{y}(t|t-1) = H\, \hat{x}(t|t-1)$$

$$= z \cdot \frac{1/5}{1 - \frac{1}{10} z^{-1}}\, y(t-1)$$

consistency

asymptotic-stable pole in $z = \frac{1}{10}$

Filter of state: $\hat{x}(t|t)$

$$\hat{x}(t|t) = F^{-1}\, \hat{x}(t+1|t)$$

$$= \left(\frac{1}{z}\right)^{-1} \hat{x}(t+1|t)$$

$$= z \cdot \frac{1/5}{1 - \frac{1}{10} z^{-1}}\, y(t)$$

consistency

11/05

Remark (on white noise): In the formulas of KF there is a requirement that $v_1(t)$ and $v_2(t)$ must be white noises. However, in many practical applications this assumption can be too demanding. $\Longrightarrow$ we need a "workaround" to deal with practical applications when this assumption is not valid.

Trick of the extended system (presentation with an example):

Example:

$$\begin{cases} x(t+1) = a\, x(t) + \eta(t) \\ y(t) = b\, x(t) + v_2(t) \end{cases}$$

$\eta(t)$ NOT a white noise

$v_2(t) \sim WN(0,1)$

A model of $\eta(t)$ is given (we need it for the trick):

$$\eta(t) = \frac{1}{1 - c z^{-1}}\, e(t) \qquad e \sim WN(0,1), \quad e \perp v_2 \text{ (uncorrelated)}$$

$\eta(t)$ is not WN but is a AR(1) stochastic process.
We cannot apply KF formulas to this system (since it requires that all the noises are white).
We can procede as follow:

$$\eta(t) = c\, \eta(t-1) + e(t)$$

$$\eta(t+1) = c\, \eta(t) + e(t+1)$$

Define: $v(t) = e(t+1)$ $\longrightarrow$ we can do this because they're all remote noises so there is no problem in shifting the time

$v(t) \sim WN(0,1)$, $v \perp v_2$

$$\Longrightarrow \eta(t+1) = c\, \eta(t) + v(t)$$

Trick: extension of the state vector:

$$\left.\begin{array}{l} x(t) \longrightarrow x_1(t) \\ \eta(t) \longrightarrow x_2(t) \end{array}\right\} \Longrightarrow X(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

$\Longrightarrow$ the extended system is:

$$\begin{cases} x_1(t+1) = a_1 x_1(t) + x_2(t) \\ x_2(t+1) = c\, x_2(t) + v(t) \\ y(t) = b\, x_1(t) + v_2(t) \end{cases} \longrightarrow \text{state equation of the dynamic of the noise}$$

$n = 2, \quad F = \begin{bmatrix} a & 1 \\ 0 & c \end{bmatrix}, \quad H = [b \;\; 0]$

$v_1 = \begin{bmatrix} 0 \\ v(t) \end{bmatrix}, \quad v_1 \sim WN(0, V_1), \quad V_1 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$

$v_2 \sim WN(0, 1)$

$V_{12} = 0$

Now we can apply KF formulas to this extended system.

Extension 5. : Extension of KF to **non linear systems**.

Consider a system with non-linear dynamics:

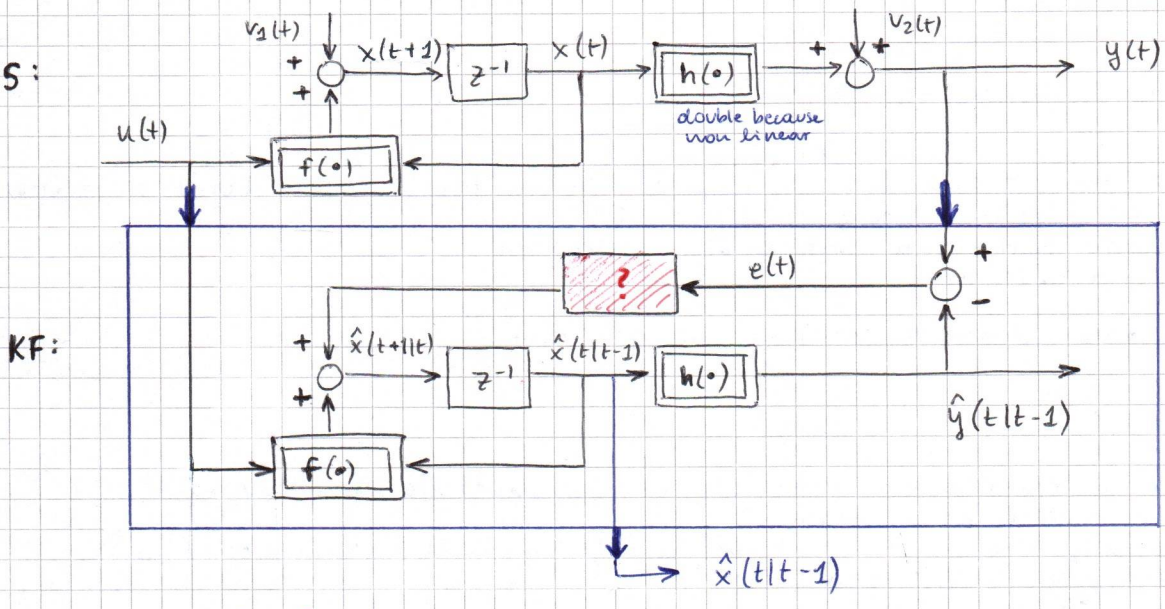
$$S: \begin{cases} x(t+1) = f(x(t), u(t)) + v_1(t) \\ y(t) = h(x(t)) + v_2(t) \end{cases}$$

in case of exogenous signal

where **f** and **h** are non linear functions of $x(t)$ and $u(t)$ (smoothness class: C^1 or higher (we must compute the derivative)).

Example: $S: \begin{cases} x(t+1) = \frac{1}{2} x(t)^5 + u(t)^3 + v_1(t) \\ y(t) = e^{x(t)} + v_2(t) \end{cases}$

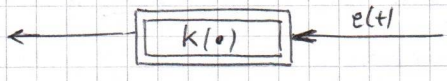
How can we design a KF in this case?
Follow basic idea of observer.



KF GAIN? (?)

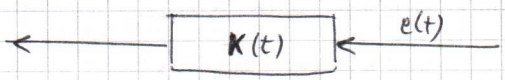
For the "gain block" of KF we have two different types of solution:

1. (most natural and intuitive) the gain is a non linear function:



(the output of the gain block is a non linear function of $e(t)$: $K(e(t))$)

2. the gain is a linear time varying (LTV) function:



$K(t)\, e(t)$

$\neq$

The second solution is less "intuitive" but it's the most effective ⟹ we can re-use in full KF formulas (with small adjustments). Remember that KF can be applied to LTV systems.

⟹ In practice extended KF idea is to make a time varying linear local approximation of a non linear time variant system.

K(t) in EKF (Extended KF) can be computed as : ([redacted])

$$K(t) = \left(F(t)\,P(t)\,H(t)^T + V_{12}\right)\left(H(t)\,P(t)\,H(t)^T + V_2\right)^{-1}$$

and P(t) can be computed from DRE:

$$P(t+1) = \left(F(t)P(t)F(t)^T + V_1\right) - \left(F(t)\,P(t)\,H(t)^T + V_{12}\right)\left(H(t)P(t)H(t)^T + V_2\right)^{-1}\left(F(t)P(t)H(t)^T + V_{12}\right)^T$$

Equations of K(t) and DRE are the usual formulas of KF. The only difference are F(t) and H(t), which are time-varying matrices defined as follow:

$$F(t) = \left.\frac{\partial f(x(t),u(t))}{\partial x(t)}\right|_{x(t)=\hat{x}(t|t-1)}, \qquad H(t) = \left.\frac{\partial h(x(t))}{\partial x(t)}\right|_{x(t)=\hat{x}(t|t-1)}$$

EKF is the time varying solution of KF where F(t) and H(t) are local linearized matrices computed around the last available state prediction $\hat{x}(t|t-1)$.

Procedure to implement EKF:
At time t :

- take the last available state prediction $\hat{x}(t|t-1)$ (computed at previous step)
- using $\hat{x}(t|t-1)$ compute F(t) and H(t)
- compute K(t) and update DRE equation
- compute $\hat{x}(t+1|t)$

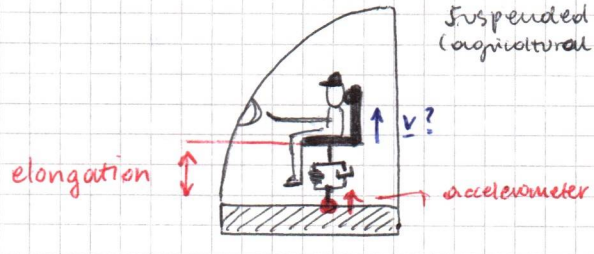
Iterate the same procedure at each sample time.

Remarks :

- EKF is very powerful since can be applied to non linear systems (major extension of KF).
- Obviously EKF does not have a "steady-state" (asymptotic) solution
- Main problems/issues of EKF:?
  Same as LTV KF :
  - very difficult / almost impossible to have a theoretical guarantee of aEKF stability (in practice: extensive empirical tests to check its stability (never 100% guarantee of stability))
  - computational load (at each time H(t), F(t), K(t), P(t) must be computed "on-time")

  EKF is largely used today with some limitations in:
  - safety-critical applications
  - mission critical applications

Example : (almost real problem) : Exemplification of KF full procedure (full set up of a Kalman filter).

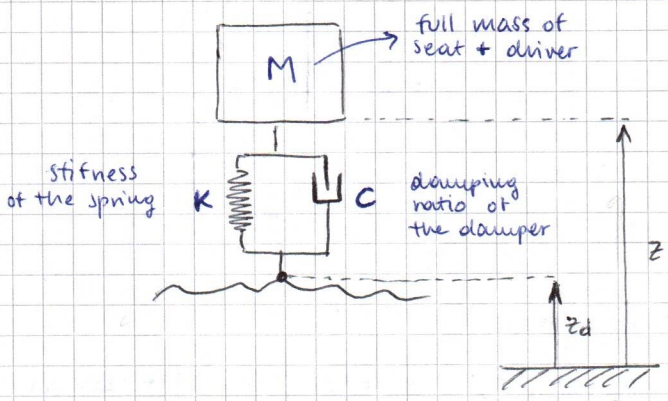


Suspended seat in the cabin of an off-highway vehicle (agricultural tractor, earth-moving machine, ...)

Physical sensors:
- vertical accelerometer based on the cabin
- elongation sensor between the basis of the cabin and the seat

Problem: estimation of the seat vertical speed (used for suspension control).

Schematic representation of the system / parameter / variables:



z_d : height of the basis of cabin
z : height of the seat

Sensors:
- acceleration : $\ddot{z}_d$ (+ noise)
- elongation : $z - z_d$ (+ noise)

Model of the system dynamics (phisical white box model in continuous time domain):

"core" model equation :
(force balance in vertical direction)

$$M\ddot{z} = -c\frac{d}{dt}(z - z_d) - k(z - z_d) + \text{gravity force} - Mg$$

(neglected since constant)

damping force "dissipative"

spring (restoring) conservative force

$$M\ddot{z} = -c(\dot{z} - \dot{z}_d) - k(z - z_d) \qquad (2^{nd} \text{ order system})$$

Since we measure $\ddot{z}_d$, the overall dimension of the system is 4
$\Longrightarrow$ vector of states variables:

$$x(t) = \begin{bmatrix} z \\ \dot{z} \\ z_d \\ \dot{z}_d \end{bmatrix} := \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \end{bmatrix}$$

Input : $u(t) = \ddot{z}_d$ (measurable disturbance given as independent exogenous input)

Output: $y(t) = z - z_d$ (sensor)

Full model in state-space form :

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -\frac{c}{M}(x_2 - x_4) - \frac{k}{M}(x_1 - x_3) \\ \dot{x}_3 = x_4 \\ \dot{x}_4 = u \\ y(t) = x_1 - x_3 \end{cases} \Rightarrow \begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -\frac{k}{M}x_1 - \frac{c}{M}x_2 + \frac{k}{M}x_3 + \frac{c}{M}x_4 \\ \dot{x}_3 = x_4 \\ \dot{x}_4 = u \\ y(t) = x_1 - x_3 \end{cases}$$

Normal form of state-space repr. :

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases}$$

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -k/m & -c/m & k/m & c/m \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \qquad C = \begin{bmatrix} 1 & 0 & -1 & 0 \end{bmatrix}$$

Next: discretization (we use digital system).

Choise of sampling time: Δ
(for this application can be $\approx 5\,ms$)

Eulero forward approximation of derivative:

$$\dot{x}(t) \approx \frac{x(t+1) - x(t)}{\Delta}$$

for example: for the first equation:

$$\frac{x_1(t+1) - x_1(t)}{\Delta} = x_2(t) \implies x_1(t+1) = x_1(t) + \Delta x_2(t)$$

So it becomes:

$$\begin{cases} x_1(t+1) = x_1(t) + \Delta x_2(t) + v_{11}(t) \\ x_2(t+1) = -\frac{\Delta k}{M} x_1(t) + \left(-\frac{\Delta c}{M} + 1\right) x_2(t) + \frac{\Delta k}{M} x_3(t) + \frac{\Delta c}{M} x_4(t) + v_{12}(t) \\ x_3(t+1) = x_3(t) + \Delta x_4(t) + v_{13}(t) \\ x_4(t+1) = x_4(t) + \Delta u(t) + v_{14}(t) \\ y(t) = x_1(t) - x_3(t) + v_2(t) \end{cases}$$

This is the <u>normal state space form in discrete time</u>, which is:

$$\begin{cases} x(t+1) = Fx(t) + Gu(t) \\ y(t) = Hx(t) \end{cases} \quad : \quad F = \begin{bmatrix} 1 & \Delta & 0 & 0 \\ -\frac{\Delta k}{M} & -\frac{\Delta c}{M} + 1 & \frac{\Delta k}{M} & \frac{\Delta c}{M} \\ 0 & 0 & 1 & \Delta \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad G = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \Delta \end{bmatrix}, \quad H = \begin{bmatrix} 1 & 0 & -1 & 0 \end{bmatrix}$$

We need to add noises!

Noise on state equations:

$$v_1(t) = \begin{bmatrix} v_{11}(t) \\ v_{12}(t) \\ v_{13}(t) \\ v_{14}(t) \end{bmatrix} \qquad v_2(t) \quad :$$

Assumptions:
- all white noises
- all uncorrelated

$v_2(t) \sim WN(0, V_2)$ → can be estimated by datasheet of elongation sensor

$v_1(t) \sim WN(0, V_1)$

where:

$$V_1 = \begin{bmatrix} \lambda_1^2 & 0 & 0 & 0 \\ 0 & \lambda_2^2 & 0 & 0 \\ 0 & 0 & \lambda_3^2 & 0 \\ 0 & 0 & 0 & \lambda_4^2 \end{bmatrix}$$

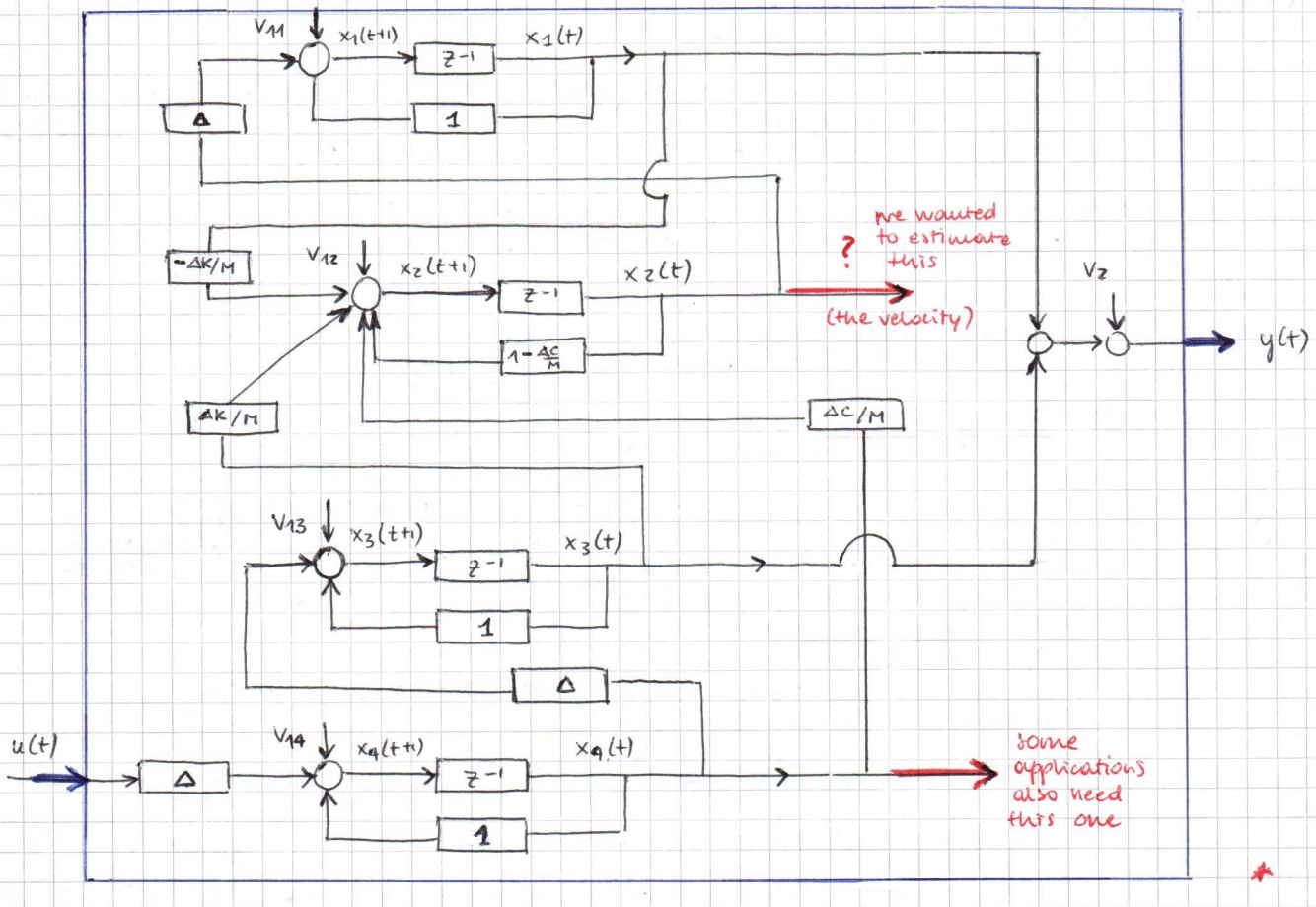
→ can be estimated by datasheet of accelerometer sensor

we expect small model errors (so we can say $\lambda_1^2, \lambda_2^2, \lambda_3^2$ are small) and we can even make the simplifying assumption:

$\lambda_1^2 = \lambda_2^2 = \lambda_3^2 = \lambda^2$ → tuned empirically

(there is no reason to assume that they're different)

Block scheme of the system:



To make it:

$x_i(t+1)$ → z^{-1} → $x_i(t)$

1.
2. feedbacks
3. noises
4. links between the different equations
5. output

Now we can compute O and R:

$$O = \begin{bmatrix} H \\ HF \\ HF^2 \\ HF^3 \end{bmatrix} \quad \text{and} \quad R = [\,G \quad FG \quad F^2G \quad F^3G\,]$$

From visual inspection of block scheme we expect full observability from output and full controllability from input (every state is influenced by the input and every state affects the output).
(Exercise:) If we compute the transfer function from $u(t)$ to $y(t)$ we expect 4 poles ($\equiv$ the system is fully visible from the output).
(check stability, the system is asymptotically stable.

KF at this point can be applied.
Notice that: theorem 1 and theorem 2 are valid
$\Longrightarrow$ we can directly jump to ARE solution ($\rightarrow \bar{P} \rightarrow \bar{K}$)

The external output does not introduce any additional uncertainty, so does not effect the **DRE** and the **gain** (that's why there is no trace of G and the input in that)

Example : (numerical example) :
Direct optimization of gain K.
Consider :

$$\begin{cases} x(t+1) = 2x(t) \\ y(t) = x(t) + v(t) \end{cases} \qquad v \sim WN(0,1)$$

→ the system is unstable, moreover the state equation is unit free

Compute the steady state (= asymptotic) predictor of the state: $\hat{x}(t+1|t)$
Two methods:

1. Direct optimization
2. KF theory

1. Direct solution

Let's start from the standard observer structure:

$$\begin{cases} \hat{x}(t+1|t) = 2\hat{x}(t|t-1) + \underbrace{K}_{\text{constant } (=\tilde{K})}\,(y(t) - \hat{y}(t|t-1)) \\ \hat{y}(t|t-1) = \hat{x}(t|t-1) \end{cases}$$

feedback correction

Optimal K?
We can try by directly minimize the variance of the state prediction error :

$$\Rightarrow \min \operatorname{Var}\left(x(t) - \hat{x}(t|t-1)\right)$$

State prediction error expression:

$$x(t+1) - \hat{x}(t+1|t) = 2x(t) - \left[2\hat{x}(t|t-1) + K\left(y(t) - \hat{y}(t|t-1)\right)\right]$$

$x(t+1)$ and $2x(t)$: system equation; $\hat{x}(t+1|t)$ and the bracketed term: observer structure equation

$$= 2x(t) - 2\hat{x}(t|t-1) - K\left(x(t) + v(t) - \hat{x}(t|t-1)\right)$$
$$= (2-K)\left(x(t) - \hat{x}(t|t-1)\right) - Kv(t)$$

Def. $\eta(t) := x(t) - \hat{x}(t|t-1)$

$$\Rightarrow \underbrace{\eta(t+1) = (2-K)\eta(t) - Kv(t)}_{\text{dynamic equation of state prediction error}} \qquad v \sim WN(0,1)$$

AR(1) process (↑), we can rewrite it as:

$$\eta(t) = \frac{1}{1-(2-k)z^{-1}}\, e(t) \qquad e(t) = -k\,v(t)$$
$$e(t) \sim WN(0, K^2)$$

AR(1) in canonical form (↑)

Easy to find the variance of $\eta(t)$: (since it's an AR(1) process)

$$\gamma_\eta(0) = \operatorname{Var}(\eta(t)) = \operatorname{var}\left(x(t) - \hat{x}(t|t-1)\right) = \frac{K^2}{1-(2-K)^2}$$

by minimizing this function w.r.t. K :

$$\frac{\partial \operatorname{Var}(\eta(t))}{\partial K} = 0 \Rightarrow \dots \Rightarrow (K_1 = 0)\ (?)$$ no feedback correction

$$\boxed{K_2 = \frac{3}{2}}$$ main solution

Both are acceptable solutions

2. KF theory

We can re-do the problem solution with KF theory

from S:

$F = 2, \quad H = 1, \quad V_1 = 0, \quad V_2 = 1, \quad V_{12} = 0, \quad \Gamma = 0$ (since $v_1 = 0$)

We check the theorems:

Thm 1:
- $V_{12} = 0$
- F is not asympt. stable

$\Rightarrow$ we can't use theorem 1

Thm 2:
- $V_{12} = 0$
- (F, Γ) not reachable
- (F, H) observable

$\Rightarrow$ we can't use theorem 2

$\Longrightarrow$ we cannot skip the analysis of DRE (the conditions for thm. are only sufficient, we can have a solution of steady state KF anyway)

DRE:

$$P(t+1) = 4P(t) - \frac{(2P(t))^2}{P(t)+1} \quad \dots$$

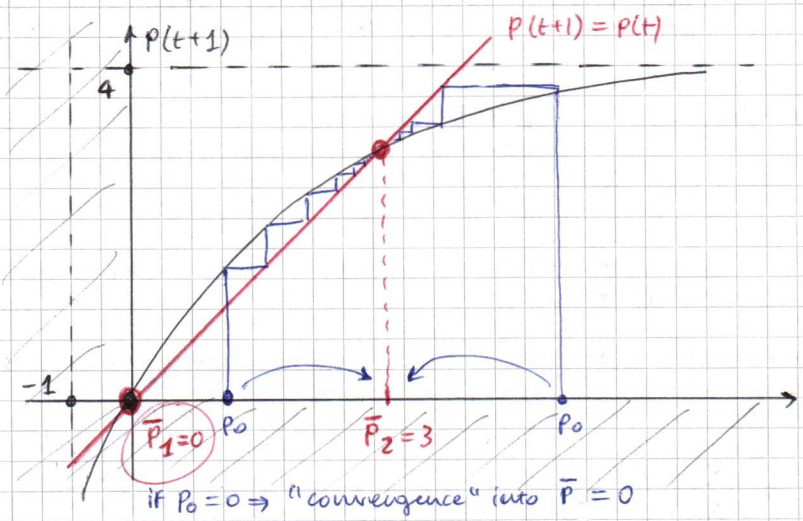
$$P(t+1) = \frac{4P(t)}{P(t)+1}$$

Find ARE solutions (≥ 0):

$$\bar{P} = \frac{4\bar{P}}{\bar{P}+1} \longrightarrow \dots \longrightarrow \begin{cases} \bar{P}_1 = 0 \\ \bar{P}_2 = 3 \end{cases} \longrightarrow \begin{cases} \bar{K}_1 = 0 \\ \bar{K}_2 = 3/2 \end{cases}$$

same solutions as before

DRE convergence analysis: (plot the DRE)



Notice that: if we start from $P_0 = 0 \Longrightarrow$ means that we have no uncertainty in the knowledge of $x(1)$. Moreover state equation is noise free $\longrightarrow$ we do not need feedback, the prediction is perfect:

$$\hat{x}(t+1|t) = 2\hat{x}(t|t-1)$$

perfect initial condition (no noise) $\rightarrow$ open loop solution

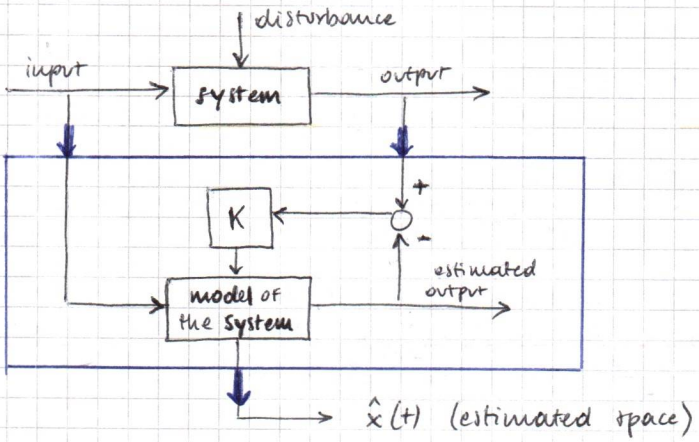
Feasible but ideal situation.

The standard solution (no perfect initial conditions) is when $K = 3/2$ (closed loop solution)

CHAPTER 4

(Software sensing (virtual sensing, variable estimation) with black box methods)

In Chapter 3 we have seen classical technology of software sensing based on KF.



Main features of this approach:

- a (white-box / physical) model must be available
- no need ("in principle") of a training data-set including measurements of the state to-be-estimated
- It is a feedback-estimation algorithm (feedback correction of the model using estimated output error)
- constructive method (non parametric / no optimization included)
- can be used ("in principle") to estimate space which are impossible to be measured (also at prototyping / training / design stage)

Are there other classes of software-sensing techniques?
Yes, black-box approaches with "learning / training" from data ("machine learning" approaches).

In this chapter we'll see black-box approaches focusing on the architecture (we don't need new algorithms, just use something we have already studied).

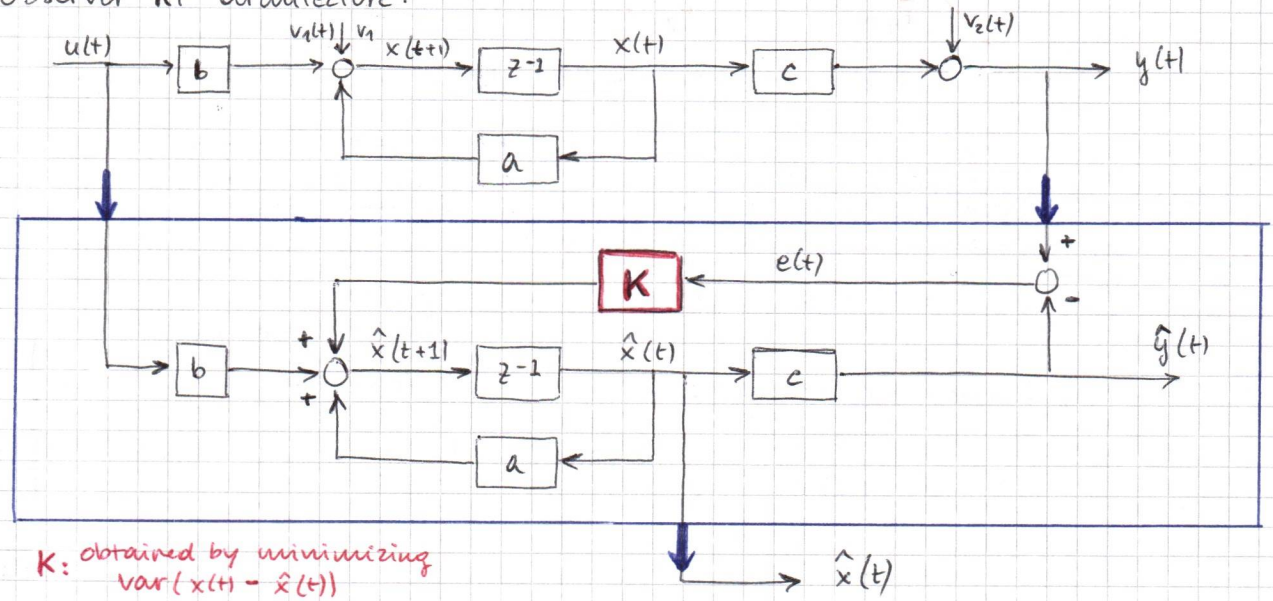
Let's start with the case of **L.T.I.** systems.
(Linear Time Invariant)

Let's consider a simplified case (SISO system with one state) to understand the approach.

$$S: \begin{cases} x(t+1) = a\,x(t) + b\,u(t) + v_1(t) \\ y(t) = c\,x(t) + v_2(t) \end{cases} \qquad \begin{matrix} v_1 \sim WN \\ v_2 \sim WN \end{matrix}$$

Problem: estimation of $\hat{x}(t)$ from measured signals $u(t)$ and $y(t)$.

Observer KF architecture:



K: obtained by minimizing $\mathrm{var}(x(t) - \hat{x}(t))$

Let's find the relationship between $u(t) \rightarrow \hat{x}(t)$, $y(t) \rightarrow \hat{x}(t)$:

$$\hat{x}(t) = \frac{b \cdot \frac{z^{-1}}{1-az^{-1}}}{1 + KC\,\frac{z^{-1}}{1-az^{-1}}}\,u(t) + \frac{K\,\frac{z^{-1}}{1-az^{-1}}}{1 + KC\,\frac{z^{-1}}{1-az^{-1}}}\,y(t)$$

$$= \dots$$

$$= \frac{b}{1+(KC-a)z^{-1}}\,u(t-1) + \frac{K}{1+(KC-a)z^{-1}}\,y(t-1)$$

we can estimate black box these two transfer functions from data

14/05

KF is a sophisticated way to build these (•) transfer functions from a W.B. (white box) model → we can estimate these T.F. directly from data.

We can adopt a black box approach to estimate these T.F.

Data set (for training):

$\{u(1), u(2), u(3), \dots, u(N)\}$

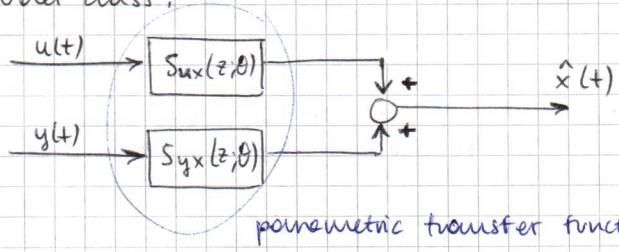
$\{y(1), y(2), y(3), \dots, y(N)\}$

$\{x(1), x(2), x(3), \dots, x(N)\}$ → in the supervised training approach we need measurement of the state to be estimated (physical sensor for $x(t)$ only for the design / training of the software sensor)

We can use 4SID for a direct non parametric identification of dynamics $u, y \rightarrow x$

or we can use a classic parametric system identification approach.

Select model class:



parametric transfer functions with parameter vector θ

The performance index is:

$$J_N(\theta) = \frac{1}{N}\sum_{t=1}^{N}\left(x(t) - \left(S_{ux}(z,\theta)\,u(t) + S_{yx}(z;\theta)\,y(t)\right)\right)^2$$

$J_N(\theta)$: variance of the estimation error

$x(t)$: measured state

$S_{ux}(z,\theta)\,u(t) + S_{yx}(z;\theta)\,y(t)$: estimated state from models

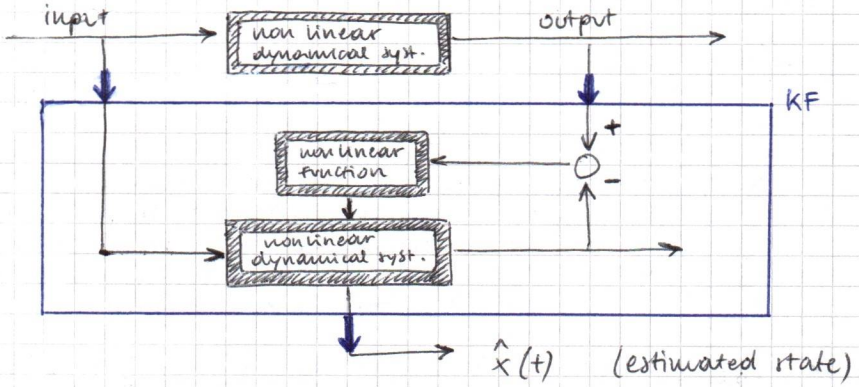
Optimization: $\hat{\theta}_N = \arg\min_{\theta} J_N(\theta)$

$\Rightarrow$ we obtain: $\begin{cases} S_{ux}(z;\hat{\theta}_N) \\ S_{yx}(z;\hat{\theta}_N) \end{cases}$ ← it represent the black box virtual sensor (because if we feed these two transfer functions with the data $u(t)$ and $y(t)$ they give us back the estimated state)

Comparison table between KF and black box software tuning:

	KF	BB	
Need of a white box physical model of S	Yes	No	bad / good
Need of a training dataset	(No) *	Yes	* theoretically no, practically yes
Interpretability of the result	Yes	No	
Easy re-tuning for a similar but different system	Yes	No	
Accuracy of the estimation	Good	Very good	
Can be used also in case of unmeasurable states	Yes	No	(we need x(t) measurement for training)

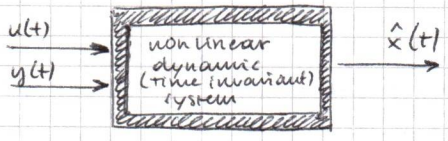
When the system is nonlinear the problem becomes more complicated.
Let's start again by taking inspiration from KF (Extended Kalman filter more precisely, the one for



Notation: ▭ : linear static dynamic ; ▭ : static nonlinear system ; ▭ : dynamic nonlinear system

Remark: In KF the EKF extension uses the trick of a time varying linear gain K(t) but the obvious and natural choise is a nonlinear gain (static nonlinear function).

The content of the ▭^{KF} box is:



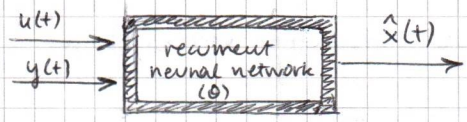
Filter of the state: $\hat{x}(t) = \hat{x}(t|t)$
(we are allowed, if needed, to use data up to present time t)

The problem is again the black box identification (supervised learning) of a nonlinear dynamic system starting from a measured training dataset:

$$\{\{u(1), u(2), \dots, u(N)\}, \{y(1), y(2), \dots, y(N)\}, \underbrace{\{x(1), x(2), \dots, x(N)\}}_{\text{physical sensor needed for training}}\}$$

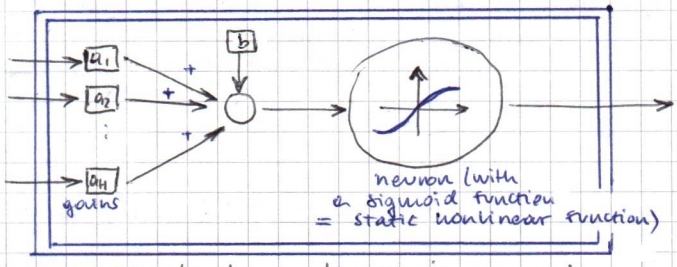
Architecture **1.** (most general):

we use a recurrent neural network.



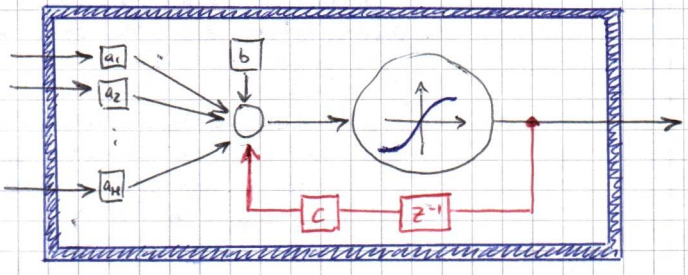
Remember that the difference between a static and a recurrent network is the neuron:

Static function of a neuron: (I/O relationship of a static neuron:)



nonlinear static

We can upgrade to a dynamic neuron:

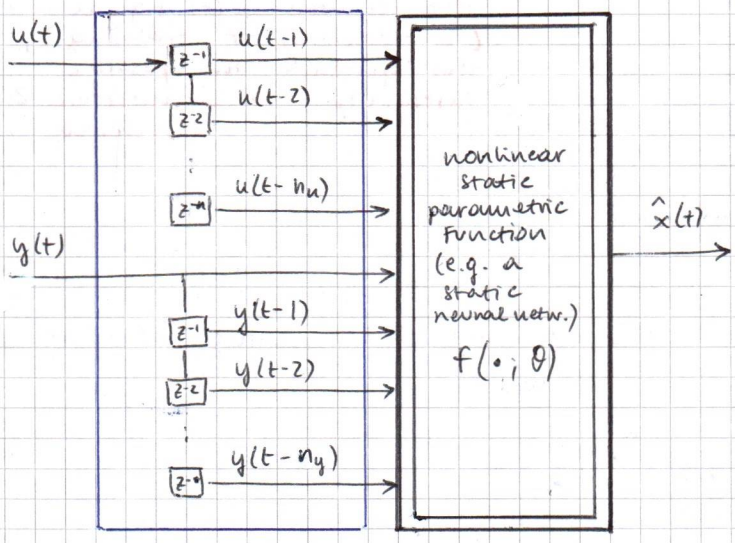


nonlinear dynamic

This is the most general approach but practically seldom used → major issues of stability and convergence of training.

Architecture 2.

Split the system into a static nonlinear system and linear dynamics (with FIR architecture (finite impulse response architecture)).



linear dynamic system (n_u, n_y are design values)

nonlinear static system (to be estimated θ)

Remark: Notice that in principle $\hat{x}(t)$ can depend on $y(t)$, whereas we know that $\hat{x}(t)$ can only depend on $u(t-1)$ (and other past values)

Remark: In case of a MIMO system with:

m inputs: $v(t) = \begin{bmatrix} v_1(t) \\ \vdots \\ v_m(t) \end{bmatrix}$, **p** outputs: $y(t) = \begin{bmatrix} y_1(t) \\ \vdots \\ y_p(t) \end{bmatrix}$

and **n** states: $x(t) = \begin{bmatrix} x_1(t) \\ \vdots \\ x_n(t) \end{bmatrix}$

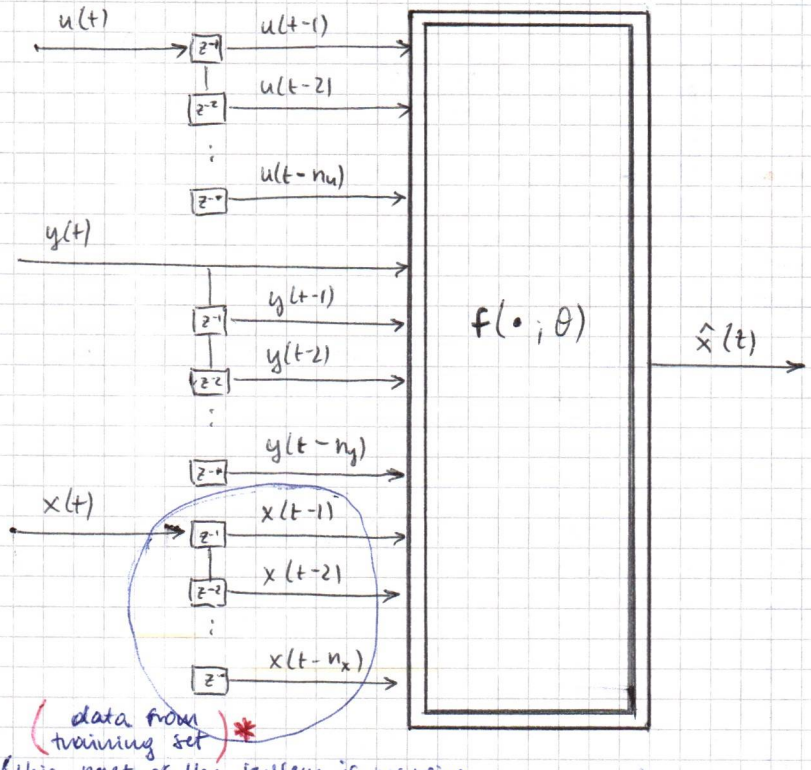
the estimation problem is the search of the optimal parameter vector θ for the function:

$$f(\cdot;\theta) : \mathbb{R}^{m \times n_u + p \times (n_y+1)} \longrightarrow \mathbb{R}^n$$

Notice that: the estimation (training) of this function $f(\cdot;\theta)$ is much more easier than the estimation of a recurrent neural network.
Moreover: stability is guaranteed (the full system is FIR (finite impulse response))

Architecture 3.

Static nonlinear function plus linear dynamics but with a IIR scheme



(this part of the system is recursive, this is why it's an INFINITE IMPULSE RESPONSE architecture)

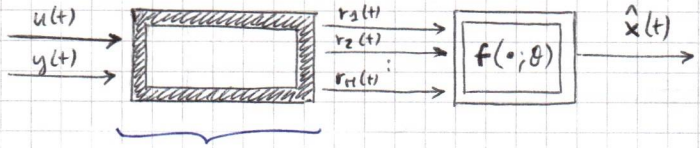
potential advantage wrt. architecture 2: n_u and n_y are smaller

potential disadvantage: this architecture is not guaranteed to be stable by construction (it depends on f)

* in production we feedback the delayed $\hat{x}(t)$ signal (in production ≠ training), this generate the possible instability ⟹ architecture 2 is preferred

Architecture 4.

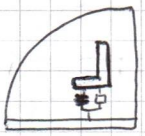
Separation of system dynamics and a static nonlinear system using regressors build from physical knowledge.

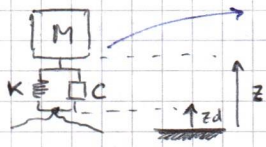


system (can be dynamic and nonlinear) that builds the regressors (signals $r_2(t), .., r_H(t)$) from physical signals $u(t)$ and $y(t)$ using physical knowledge/understanding of the system
("it's more "human-driven" ⟶ "art of machine learning/ software tuning")

The idea is to facilitate the job of $f(\cdot;\theta)$ by presenting at its input a smaller and more meaningful set of signals (regressors).

Example: (continue example at the end of chapter 3)





→ estimation (// software sensing) problem: $\dot{z}$? (one of the states)

Model (key equation of the system):

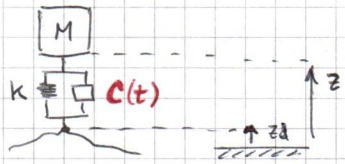
$$M\ddot{z} = -c(t)(\dot{z} - \dot{z}_d) - k(z - z_d)$$

$\ddot{z}_d$

$y(t) = z - z_d$

→ measurable input (accelerometer)
→ measurable output (elongation sensor)

$c(t)$: now we assume a "semi-active" suspension ⇒ $c(t)$ can be electronically changed (control variable)



We can solve the problem with KF (chapter 3) or we can make an experiment and collect training data:

dataset:

$c(t) \rightarrow \{c(1), c(2), \ldots, c(N)\}$

$z(t) - z_d(t) \rightarrow \{(z(1) - z_d(1)), \ldots, (z(N) - z_d(N))\}$

$\ddot{z}(t) \rightarrow \{\ddot{z}(1), \ddot{z}(2), \ldots, \ddot{z}(N)\}$

and for **design only**:

$\dot{z}(t) \rightarrow \{\dot{z}(1), \dot{z}(2), \ldots, \dot{z}(N)\}$ → measured but just for training

we need the variable we want to estimate (to software sense)

we put a sensor only to get some training data (so for design) and then in production we remove it

(Example: rotational angle for a motocycle: can we measure it? Yes. Can we use the measurements in production? No. ⇒ we want the software sensoring, but for the training phase we need the measurements)

18/05

Back to the main equation:

$$M\ddot{z} = -k(z - z_d) - c(t)(\dot{z} - \dot{z}_d)$$

: sensors

we integrate it:

$$\dot{z} = -\frac{k}{M}\int \underbrace{(z - z_d)}_{r_1(t)\ \text{elongation}}\,dt - \frac{1}{M}\int \underbrace{c(t)}_{r_2(t)\ \text{control variable}}(\dot{z} - \dot{z}_d)\,dt$$

variable to be estimated

$r_1(t)$ and $r_2(t)$ are primary regressors (directly linked with $\dot{z}(t)$)
$r_3(t)$ is a secondary regressor (can help $r_1(t)$)

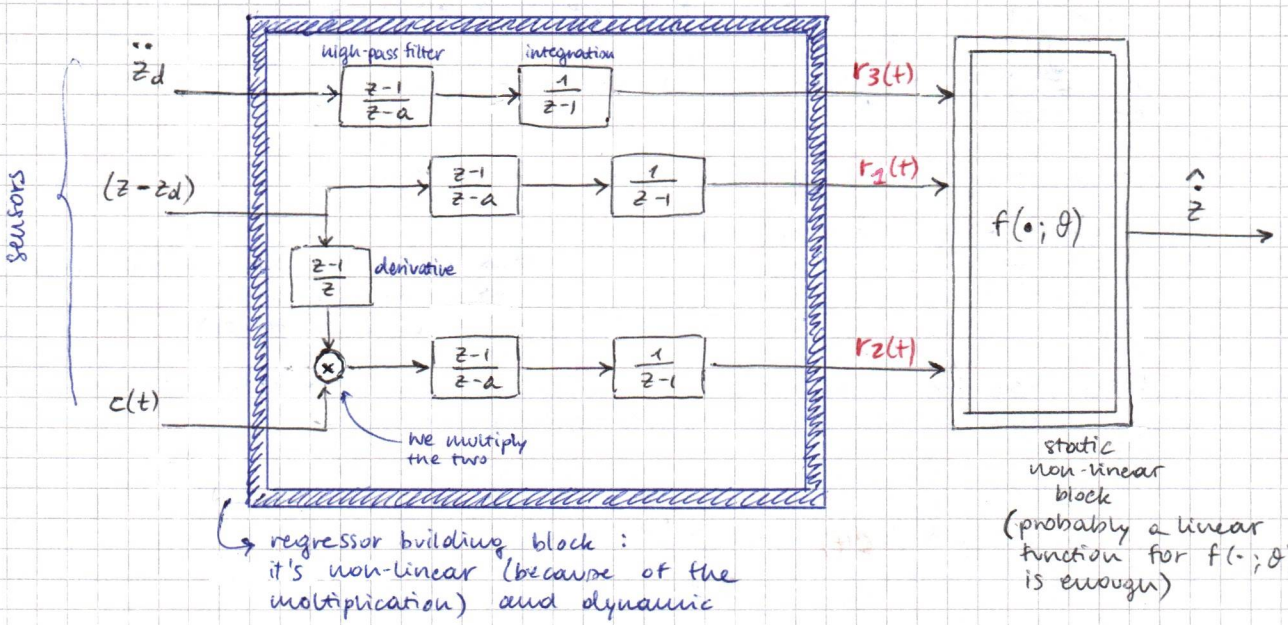
we also consider:

$$\dot{z}_d = \int \underbrace{\ddot{z}_d}_{r_3(t)\ \text{acceleration of disturbance}}\,dt$$

Since these regressors are obtained by integration, to avoid drifting (by DC-components of noise integration) → we have high-pass the inputs with high-pass filters:

$$\frac{z-1}{z-a}$$ (digital high-pass filter)

Full filter scheme:



Conclusions:
In case of black box software sensing with non-linear systems the problem can be quite complex.
Using "brutal force" approach (1 dynamical neural network) is usually doomed to failure.
The best is to gain some insight into the system and build some "smart" regressors before "black box mapping".

CHAPTER 5

(Gray-box system identification)

2 approaches:
- K.F. (Kalman Filter)
- S.E.M. (Simulation Error Method)

GRAY-BOX SYSTEM IDENTIFICATION USING KF

KF is not a system identification method, it's a variable estimation approach (software sensing / observer). However we can use it for gray-box system identification ("side benefit" of KF).

Problem definition: we have a model, typically build as a white-box model using first principles:

$$\begin{cases} x(t+1) = f(x(t), u(t); \theta) + v_1(t) \\ y(t) = h(x(t); \theta) + v_2(t) \end{cases}$$

$v_1(t)$: model noise; $v_2(t)$: output/sensor noise

f, h are linear/non linear functions depending on some unknown parameter θ (with a physical meaning (ex: mass, resistence, friction coeff., ...)). The problem is to estimate $\hat{\theta}$.

KF solves this problem by transforming the unknown parameters in extended states $\longrightarrow$ KF makes the simultaneous estimation of $\hat{x}(t|t)$ (classic KF problem) and $\hat{\theta}(t)$ (parameter identification problem). (θ originally is not a variable, we transform it into a variable)

Trick: state extension:

$$\begin{cases} x(t+1) = f(x(t), u(t); \theta) + v_1(t) \\ \theta(t+1) = \theta(t) + v_\theta(t) \\ y(t) = h(x(t); \theta) + v_2(t) \end{cases}$$

The new extended state version is: $x_E(t) = \begin{bmatrix} x(t) \\ \theta(t) \end{bmatrix}$

→ the unknown parameters are transformed in unknown variables

The new equation "we create" ($\theta(t+1) = \theta(t) + v_\theta(t)$):

- it's a fictitious equation (not a physical equation)
- the core dynamics are $\theta(t+1) = \theta(t)$: this is the equation of something which is constant $\longrightarrow$ this is exactly the nature of $\theta(t)$ which is indeed a constant vector of parameters
- we need the fictitious noise $v_\theta(t)$ in order to force KF to find (look for) the right value of θ (if no noise in this equation, KF probably would stay fixed on the initial condition. We tell KF: "do not rely on initial conditions")
- Notice: this equation is not of an asymptotically stable system but of a simply stable system $\longrightarrow$ no problem because KF can deal with non-asymptotically stable systems

The design choice is the choise of the covariance matrix of $v_\theta(t)$:

$v_\theta(t) \sim WN(0, V_\theta)$ vector of withe noises.

We make the empirical assumptions:

$v_1 \perp v_\theta$, $v_2 \perp v_\theta$

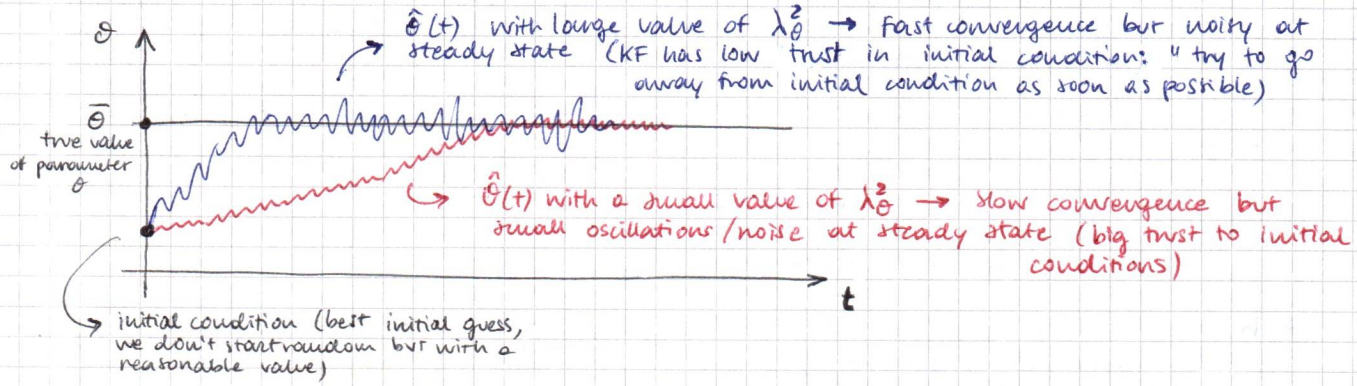
(there is no reason for $v_\theta(t)$ to be correlated with $v_1(t)$ and $v_2(t)$).

$$V_\theta = \begin{bmatrix} \lambda_{1\theta}^2 & & \phi \\ & \lambda_{2\theta}^2 & \\ \phi & & \lambda_{n_\theta\theta}^2 \end{bmatrix} \in \mathbb{R}^{n_\theta \times n_\theta}$$

n_θ = # parameters (unknown) in θ

Usually it is assumed that: $\lambda_{1\theta}^2 = \lambda_{2\theta}^2 = \dots = \lambda_{n_\theta\theta}^2$
$\Rightarrow$ we assume that $v_\theta(t)$ is a set of independent white noises with the same variance λ_θ^2 (tuning parameter / design parameter, tuned empirically).

Influence of the choise of design parameter λ_θ^2:

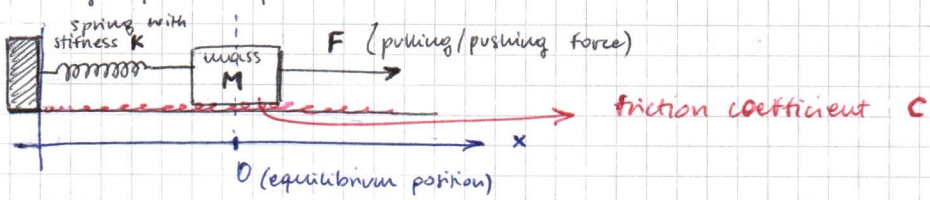


$\rightarrow$ λ_θ^2 is selected according to the best compromise for our specific application.

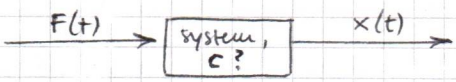
Notice: this trick can work in principle with any number of unknown parameters (e.g. 3 sensors, 10 states, 20 parameters).
In practice it works well only on a limited number of parameters. (e.g. 3 sensors, 5 states, 2 parameters).

The extrem way of using KF is black box identification.
(theoretically it works, practically no: wrong tool applied in the wrong way)

Example: (KF for oneway-box parameter estimation)



Input variable: $F(t)$
Output measured variable: position $x(t)$
Parameters K and M are known (measured) but C is unknown.



Problem: estimate C with KF.
(Remember that:)
Using KF we do not need a training dataset

Step 1: model (WB) the system:

$$\ddot{x} M = -Kx - c\dot{x} + F$$: differential continuous time linear equation

$\rightarrow$ 2nd order equation: we need 2 state variables
is a mechanical system so the most natural ones are:
- position $x_1(t) = x(t)$
- speed $x_2(t) = \dot{x}(t)$

We rewrite the syst. in state-space form $\Rightarrow$

$$\begin{cases} \dot{x}_1(t) = x_2(t) \\ M\dot{x}_2(t) = -Kx_1(t) - Cx_2(t) + F(t) \\ y(t) = x_1(t) \end{cases}$$

$$\Rightarrow \begin{cases} \dot{x}_1(t) = x_2(t) \\ \dot{x}_2(t) = -K/M\ x_1(t) - C/M\ x_2(t) + \frac{1}{M} F(t) \\ y(t) = x_1(t) \end{cases}$$

Step **2** : discretization :

Eulero forward : $\dot{x}(t) \approx \frac{x(t+1) - x(t)}{\Delta}$ (Δ sampling time)

$$\Rightarrow \begin{cases} \frac{x_1(t+1) - x_1(t)}{\Delta} = x_2(t) \\ \frac{x_2(t+1) - x_2(t)}{\Delta} = -\frac{K}{M} x_1(t) - \frac{C}{M} x_2(t) + \frac{1}{M} F(t) \\ y(t) = x_1(t) \end{cases}$$

Standard SS form : $\begin{cases} x(t+1) = Fx(t) + Gu(t) \\ y(t) = Hx(t) \end{cases}$

$$\Rightarrow \begin{cases} x_1(t+1) = x_1(t) + \Delta x_2(t) \\ x_2(t+1) = -\frac{K\Delta}{M} x_1(t) + \left(1 - \frac{C\Delta}{M}\right) x_2(t) + \frac{\Delta}{M} F(t) \\ y(t) = x_1(t) \end{cases}$$

(C → unknown)

→ linear 2nd order discrete time system

Step **3** : state extension :

$x_3(t) := C(t)$

So the extended system becomes:

$$\begin{cases} x_1(t+1) = x_1(t) + \Delta x_2(t) + v_{11}(t) \\ x_2(t+1) = -\frac{K\Delta}{M} x_1(t) + \left(1 - \frac{x_3(t)\Delta}{M}\right) x_2(t) + \frac{\Delta}{M} F(t) + v_{12}(t) \\ x_3(t+1) = x_3(t) + v_{13}(t) \\ y(t) = x_1(t) + v_2(t) \end{cases}$$

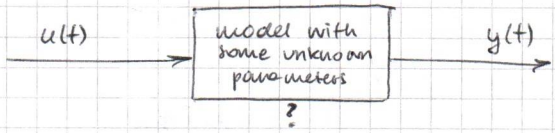
Then we add the noises s.t. :

$$V_1 = \begin{bmatrix} \lambda_{11}^2 & & \emptyset \\ & \lambda_{12}^2 & \\ \emptyset & & \lambda_{13}^2 \end{bmatrix} \qquad V_2 = \lambda^2$$

Now it's ready for KF application.
Notice that we get at the same time $\hat{x}(t)$ and $\hat{c}(t)$ (we get both state estimation and parameters estimation).

Notice that we need extended KF (EKF) → even if the original system was linear, state extension moved to a non linear system (since we have $x_2(t) \cdot x_3(t)$).

Are there alternative ways to solve gray-box system identification problems? Yes.
Commonly (and intuitive) method is a parametric identification approach based on simulation error method (S.E.M.)



1. Collect data from an experiment:

$$\{\{\tilde{u}(1), \tilde{u}(2), .., \tilde{u}(N)\}, \{\tilde{y}(1), \tilde{y}(2), .., \tilde{y}(N)\}\}$$

($\sim$ = measured data)

2. Define the model structure:

$$y(t) = \mathcal{M}(u(t); \bar{\theta}; \theta)$$

- $u(t)$: input signal
- θ: set of UNKNOWN parameters (there can be some bounds: $\theta_{min} < \theta < \theta_{max}$)
- $\bar{\theta}$: set of known parameters (mass, resistence, ..)
- $\mathcal{M}$: mathematical model (linear or nonlinear) usually written from first principles equations (physical laws, ..)

3. Performance index definition:

$$J_N(\theta) = \frac{1}{N}\sum_{t=1}^{N}\left(\tilde{y}(t) - \mathcal{M}(\tilde{u}(t); \bar{\theta}; \theta)\right)^2$$

$\tilde{y}(t)$: measured output; $\tilde{u}(t)$: measured input; $\mathcal{M}(\tilde{u}(t); \bar{\theta}; \theta)$: simulated output

Sample variance of the simulation error

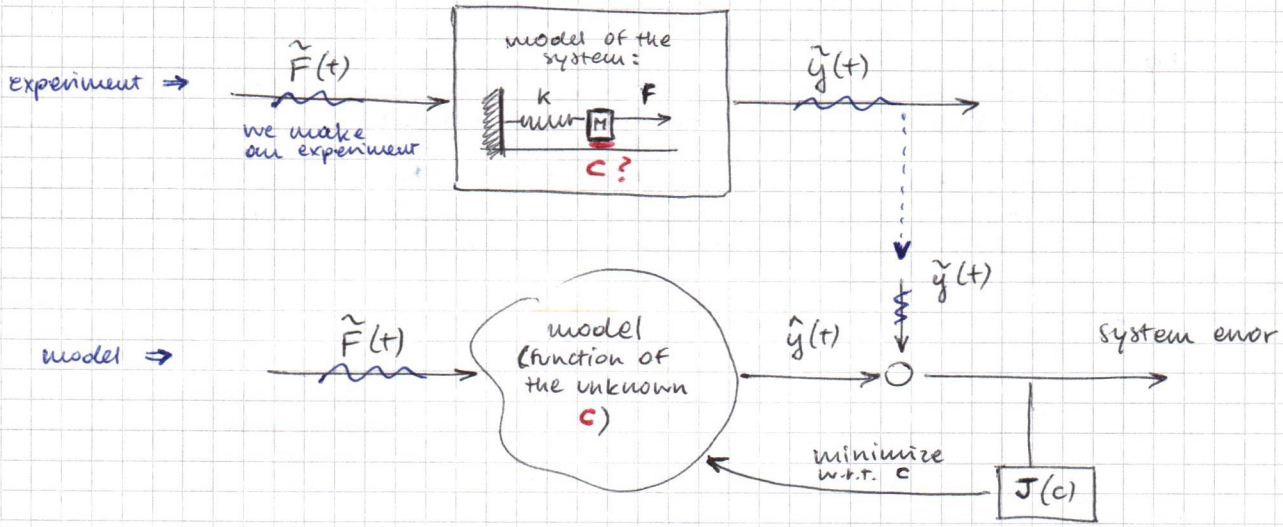
4. Optimization:

$$\hat{\theta}_N = \arg\min_{\theta}\{J_N(\theta)\}$$

Usually:
- no analytic expression of $J_N(\theta)$ is available
- each computation of $J_N(\theta)$ requires an entire simulation of the model from $t=1$ to $t=N$
- usually $J_N(\theta)$ is a non quadratic and non convex function $\Rightarrow$ iterative and randomized optimization methods must be used.

$\Rightarrow$ Intuitive but computationally very demanding

ex.



Question: can S.E.M. be applied also to black box methods? (Yes)

Consider the following example:
we collect data: $\{\tilde{u}(1), \tilde{u}(2), \ldots, \tilde{u}(N)\}$, $\{\tilde{y}(1), \tilde{y}(2), \ldots, \tilde{y}(N)\}$ and then we want to estimate from data this I/O model:

$$y(t) = \frac{b_0 + b_1 z^{-1}}{1 + a_1 z^{-1} + a_2 z^{-2}} u(t-1), \qquad \theta = \begin{bmatrix} a_1 \\ a_2 \\ b_1 \\ b_2 \end{bmatrix}$$

In time domain:

$$y(t) = -a_1 y(t-1) - a_2 y(t-2) + b_0 u(t-1) + b_1 u(t-2)$$

Using **P.E.M.** : (Prediction error method)

$$\hat{y}(t|t-1) = -a_1 \tilde{y}(t-1) - a_2 \tilde{y}(t-2) + b_0 \tilde{u}(t-1) + b_1 \tilde{u}(t-2)$$

(all measured data)

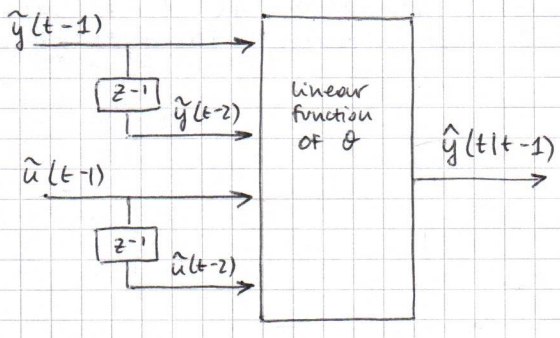
$$\Longrightarrow \underset{\text{(PEM)}}{J_N(\theta)} = \frac{1}{N} \sum_{t=1}^{N} \left(\tilde{y}(t) - \hat{y}(t|t-1;\theta) \right)^2 = \frac{1}{N} (\text{linear w.r.t. } \theta)^2 = \frac{1}{N} (\text{quadratic w.r.t. } \theta)$$

$$= \frac{1}{N} \sum_{t=1}^{N} \left(\tilde{y}(t) + a_1 \tilde{y}(t-1) + a_2 \tilde{y}(t-2) - b_0 \tilde{u}(t-1) - b_1 \tilde{u}(t-1) \right)^2$$

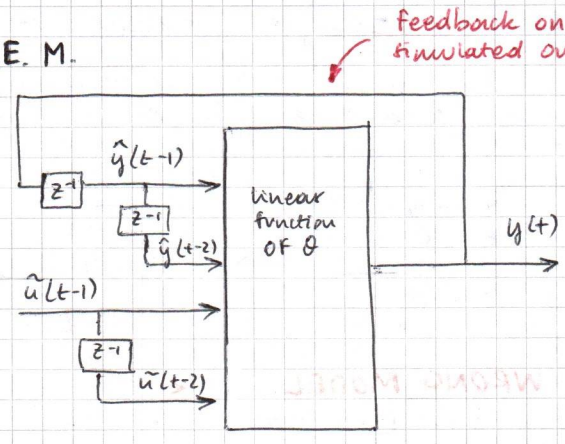
quadratic function of θ
(the minimization is very simple)

Corresponding block scheme:

P.E.M.



S.E.M.



Using **S.E.M.** :

$$\hat{y}(t) = -a_1 \hat{y}(t-1) - a_2 \hat{y}(t-2) + b_0 \tilde{u}(t-1) + b_1 \tilde{u}(t-2)$$

values of simulated output; measured

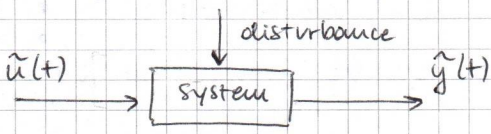
$$\Longrightarrow J_N(\theta) = \frac{1}{N} \sum_{t=1}^{N} \left(\tilde{y}(t) - \hat{y}(t;\theta) \right)^2$$

$$= \frac{1}{N} \sum_{t=1}^{N} \left(\tilde{y}(t) + a_1 \hat{y}(t-1) + a_2 \hat{y}(t-2) - b_0 \tilde{u}(t-1) - b_1 \tilde{u}(t-2) \right)^2$$

highly non linear w.r.t. θ (since $\hat{y}(t-1)$ and $\hat{y}(t-2)$ are not meas.)
(non quadratic and non convex)

P.E.M. approach looks much better ! (?) , **but** :

1. do not forget the noise :



: $\tilde{y}(t)$ also contains some disturbance

PEM is much less robust w.r.t. noise ⟹ we must include a model of the noise in the estimated model ⟹ ARMAX model

↳ noise model

If ARX :

$$y(t) = \frac{b_0 + b_1 z^{-1}}{1 + a_1 z^{-1} + a_2 z^{-2}} u(t-1) + \frac{1}{1 + a_1 z^{-1} + a_2 z^{-2}} e(t)$$

part of model of noise ($e \sim WN$)

$$\hat{y}(t|t-1) = b_0 u(t-1) + b_1 u(t-2) - a_1 y(t-1) - a_2 y(t-2)$$

linear in the parameters vector

If ARMAX :

$$1 \longrightarrow 1 + c_1 z^{-1} + \dots + c_m z^{-m}$$

⟹ $J_N(\theta)$ is highly non linear (and it becomes a problem of the same complexity of S.E.M.)

2. very sensitive to samply time choice.

Remember that when we write at discrete time $y(t)$ we mean $y(t \cdot \Delta)$ (which means "t times the samply time Δ").

$$\hat{y}(t|t-1) = -a_1 \tilde{y}(t-1) - a_2 \tilde{y}(t-2) + b_0 \tilde{u}(t-1) + b_1 \tilde{u}(t-2)$$

If Δ is very small ⟹ the difference between $\tilde{y}(t)$ and $\tilde{y}(t-1)$ becomes very small and it tends to 0 ⟹ the P.E.M. optimization tends to provide this "trivial" solution :

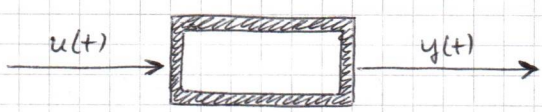
$$\left.\begin{array}{l} a_1 \longrightarrow -1 \\ a_2 \longrightarrow 0 \\ b_0 \longrightarrow 0 \\ b_1 \longrightarrow 0 \end{array}\right\} \Longrightarrow \hat{y}(t) \approx \tilde{y}(t-1)$$

WRONG MODEL

due to the fact that the recursive part of the model is using past measures of the output instead of past values of the model outputs

---

Summary of system identification methods for I/O systems :



- collect a dataset for training (if needed)
- choose a model domain :
  - linear static
  - non-linear static
  - linear dynamic
  - non-linear dynamic

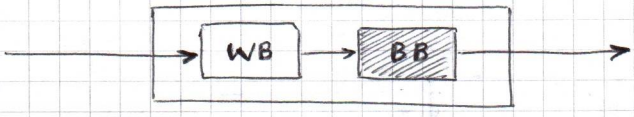
  } gray box / black box
- estimation method ⟶ constructive (4SID)
  ⟶ parametric optimization : } • P.E.M. • S.E.M.
  ⟶ filters (state extension of K.F.)

Better black box for system identification or software sensing or white box? No simple answer. It depends on goals and type of applications.

B.B.
- very general
- very flexible
- make maximum use of data
- no (or little) need of domain know-how

W.B.
- very useful when we are the system designer (not only the control algorithm designer)
- provide more insight in the system

Gray Box can sometimes be obtained by Hybrid systems :
(a part is a black-box, a part is a white-box) :



CHAPTER 6

25/05

(Minimum variance control (Design and analysis of feedback system))
It's not system identification and not software sensing.

- control design is the main motivation to system identification and software sensing
- MVC is based on the mathematics of system identification and software sensing (prediction theory)

Setup of the problem :

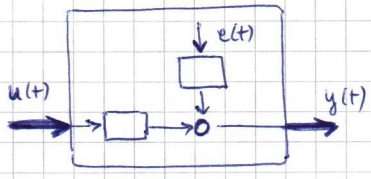
Consider a generic ARMAX model :

$$y(t) = \underbrace{\frac{B(z)}{A(z)}\,u(t-k)}_{\text{I/O part of model}} + \underbrace{\frac{C(z)}{A(z)}\,e(t)}_{\text{noise model}} \qquad e \sim WN(0, \lambda^2)$$

$$B(z) = b_0 + b_1 z^{-1} + \dots + b_p z^{-p}$$

$$A(z) = 1 + a_1 z^{-1} + \dots + a_m z^{-m}$$

$$C(z) = 1 + c_1 z^{-1} + \dots + c_n z^{-n}$$

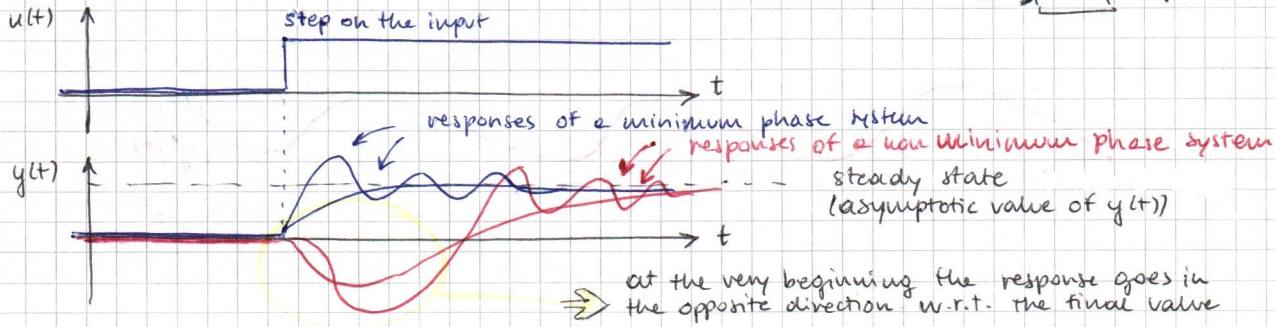


Assumptions :
- $C(z)/A(z)$ is in canonical form
- $b_0 \neq 0 \implies$ means that k is the actual delay of the system
- $B(z)/A(z)$ is "minimum phase"

$B(z)/A(z)$ is said to be "minimum phase" if all the roots of $B(z)$ (the numerator) are strictly inside the unit circle.

(denominator → poles → stability
numerator → zeros → minimum phase)

Feature of a non-minimum phase system :



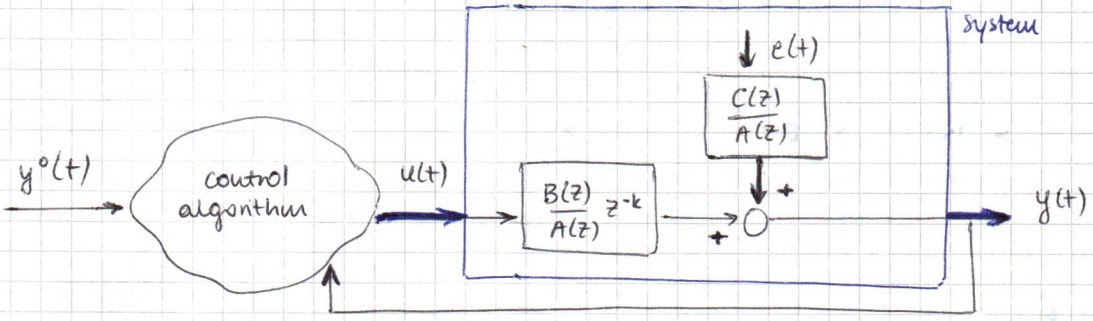
⇒ at the very beginning the response goes in the opposite direction w.r.t. the final value

Intuitively is very difficult to control non-min. phase systems → we can take the wrong decision if we react immediately.
Also for a human is difficult:

example: steer → roll dynamics in a bycicle

Design of controllers for non-min. phase is difficult and requires special design techniques (no MVC but generalized MVC (Minimum Variance Control))

The problem we wish to solve is optimal tracking of the desired behaviour of the output:



In a more formal way → MVC tries to minimize this performance index:

$$J = \mathbb{E}\left[(y(t) - y^{\circ}(t))^2 \right] \quad :=$$ variance of the tracking error (this is why is called "Minimum Variance Control")

Some additional (small) technical assumptions:

- $y^{\circ}(t)$ and $e(t)$ are not correlated ($y^{\circ}(t) \perp e(t)$) [usually easily fullfilled]
- we assume that $y^{\circ}(t)$ is known only up to time t (present time)
  → we have no preview of the future desired $y^{\circ}(t)$
  ($y^{\circ}(t)$ is totally unpredictable:

$$\hat{y}^{\circ}(t+k|t) = y^{\circ}(t) \quad : \text{ at present time } t \text{ is simply } y^{\circ}(t)$$

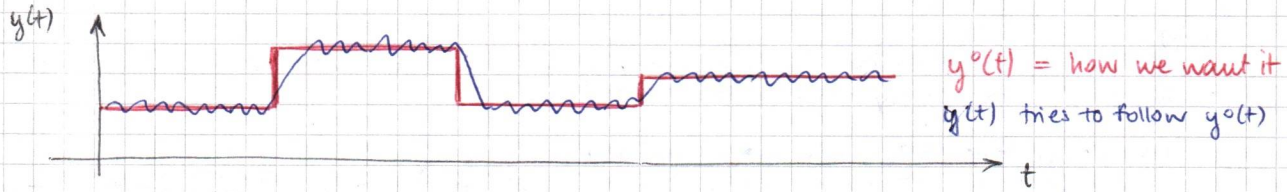
(the best prediction is to keep it constant)

Remark: There are 2 sub-classes of control problems:

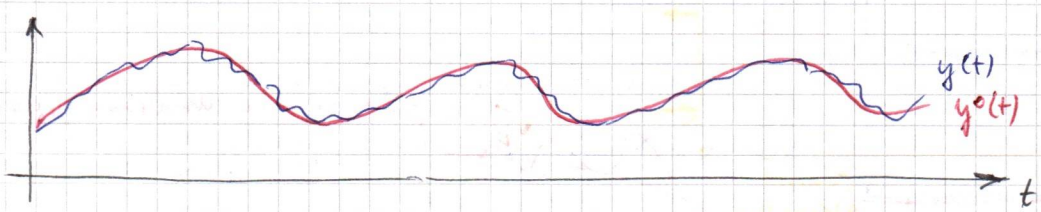
1. when $y^{\circ}(t)$ is constant or stepwise ⟹ "regulation problem"
2. when $y^{\circ}(t)$ is varying ⟹ "tracking problem"

graphical examples:

- regulation problem:
  cruise control in a car : $y^{\circ}(t)$ = desired speed (we want it constant)



- tracking problem:
  trajectory of a robot arm (angle joint)



Bottom up way of presenting MVC:

- simplified problem 1.
- simplified problem 2.
- general solution

Simplified problem 1.

$$S: \; y(t) = a\,y(t-1) + b_0\,u(t-1) + b_1\,u(t-2)$$

$$\Longrightarrow \; y(t) = \frac{b_0 + b_1 z^{-1}}{1 + a z^{-1}}\,u(t-1) + \cancel{\text{noise}} \quad \text{(noise free)}$$

and we assume that: $y^0(t) = \bar{y}^0$ ($\rightarrow$ "regulation problem")

assumptions:
- $b_0 \neq 0$
- roots of numerator must be inside the unit circle

To design the minimum variance controller we must minimize:

$$J = \mathbb{E}\left[\left(y(t) - y^0(t)\right)^2\right]$$

since there is no noise we can remove "$\mathbb{E}$" → since the $\mathbb{E}$ makes sense when we have stochastic variables, but here $\nexists$ noise $\Longrightarrow$ there is nothing stochastic

$$\Longrightarrow \; J = \left(y(t) - y^0(t)\right)^2$$
$$= \left(y(t) - \bar{y}^0\right)^2$$
$$= \left(a\,y(t-1) + b_0\,u(t-1) + b_1\,u(t-2) - \bar{y}^0\right)^2$$
$$= \left(a\,y(t) + \underbrace{b_0\,u(t) + b_1\,u(t-1)}_{(b_0 + b_1 z^{-1})\,u(t)} - \bar{y}^0\right)^2$$

TIME-SHIFT (possible since J is invariant) because we are interested in $u(t)$

derivative w.r.t. $u(t)$:

$$\frac{\partial J}{\partial u(t)} = 2\left(a\,y(t) + b_0 u(t) + b_1 u(t-1) - \bar{y}^0\right)(b_0)$$

not $(b_0 + b_1 z^{-1})$

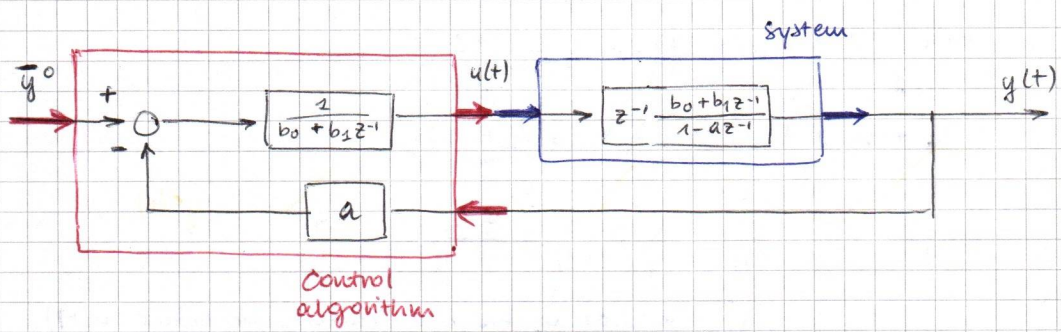
Why the derivative is just b_0?
We are at present time t: at time t the control algorithm must take a decision on the value of $u(t)$.
At time t: $y(t), y(t-1), \ldots$, $u(t-1), u(t-2), \ldots$ are no longer variables but numbers!

$u(t-1), u(t-2), \ldots$ no longer objects of optimization

$$\frac{\partial J}{\partial u(t)} = 0$$

$$\Longrightarrow \; a\,y(t) + b_0\,u(t) + b_1\,u(t-1) - \bar{y}^0 = 0$$

$$\Longrightarrow \; u(t) = \left(\bar{y}^0 - a\,y(t)\right) \cdot \frac{1}{b_0 + b_1 z^{-1}} \quad := \quad \textbf{CONTROL ALGORITHM}$$



Simplified problem 2.

$k=1$

$S: \quad y(t) = a\,y(t-1) + b_0\,u(t-1) + b_1\,u(t-2) + e(t) \qquad e(t) \sim WN(0, \lambda^2)$

The reference variable is $y^\circ(t)$ ($\rightarrow$ "tracking problem")

Assumptions:
- $b_0 \neq 0$
- $(b_0 + b_1 z^{-1})$ is minimum phase

The performance index is:

$$J = \mathbb{E}\left[(y(t) - y^\circ(t))^2\right]$$

The fundamental "trick" to solve this problem is to re-write $y(t)$ as:

$$y(t) = \underbrace{\hat{y}(t|t-1)}_{\text{predictor of } y(t) \text{ at time } t-1} + \underbrace{\varepsilon(t)}_{\text{correspond. pred. error}}$$

Since $k=1$: $\varepsilon(t) = e(t) \Longrightarrow y(t) = \hat{y}(t|t-1) + e(t)$

$$\Longrightarrow J = \mathbb{E}\left[(\hat{y}(t|t-1) + e(t) - y^\circ(t))^2\right]$$
$$= \mathbb{E}\left[((\hat{y}(t|t-1) - y^\circ(t)) + e(t))^2\right]$$
$$= \mathbb{E}\left[(\hat{y}(t|t-1) - y^\circ(t))^2\right] + \underbrace{\mathbb{E}\left[e(t)^2\right]}_{\text{constant}} + 2\,\mathbb{E}\left[(\hat{y}(t|t-1) - y^\circ(t))\,e(t)\right]$$

- $e(t) \perp y^\circ(t)$ by assumptions
- $\hat{y}(t|t-1) \perp e(t)$ by construction of the predictor

Notice that:

$$\underset{u(t)}{\arg\min}\left\{\mathbb{E}\left[(\hat{y}(t|t-1) - y^\circ(t))^2\right] + \lambda^2\right\} = \underset{u(t)}{\arg\min}\ \mathbb{E}\left[(\hat{y}(t|t-1) - y^\circ(t))^2\right]$$

↑ because λ^2 is a constant and does not depend on $u(t)$

$\Longrightarrow$ the best result is: $\hat{y}(t|t-1) = y^\circ(t)$

Now we must compute the 1-step predictor for the system:

$$S: \quad y(t) = \frac{b_0 + b_1 z^{-1}}{1 - a z^{-1}}\,u(t-1) + \frac{1}{1 - a z^{-1}}\,e(t)$$

$\rightarrow$ ARMAX $(1, 0, 1+1)$ = ARX $(1, 2)$

$k=1$, $B(z) = b_0 + b_1 z^{-1}$, $A(z) = 1 - a z^{-1}$, $C(z) = 1$

General solution for 1-step predictor of ARMAX: (MIDA 1)

$$\hat{y}(t|t-1) = \frac{B(z)}{C(z)}\,u(t-1) + \frac{C(z) - A(z)}{C(z)}\,y(t)$$

$$\Longrightarrow \hat{y}(t|t-1) = \frac{b_0 + b_1 z^{-1}}{1}\,u(t-1) + \frac{1 - 1 + a z^{-1}}{1}\,y(t)$$

$$\Longrightarrow \hat{y}(t|t-1) = (b_0 + b_1 z^{-1})\,u(t-1) + a\,y(t-1)$$

Suppose now:

$$\hat{y}(t|t-1) = y^\circ(t) \qquad (\text{or} \quad \hat{y}(t+1|t) = y^\circ(t+1))$$

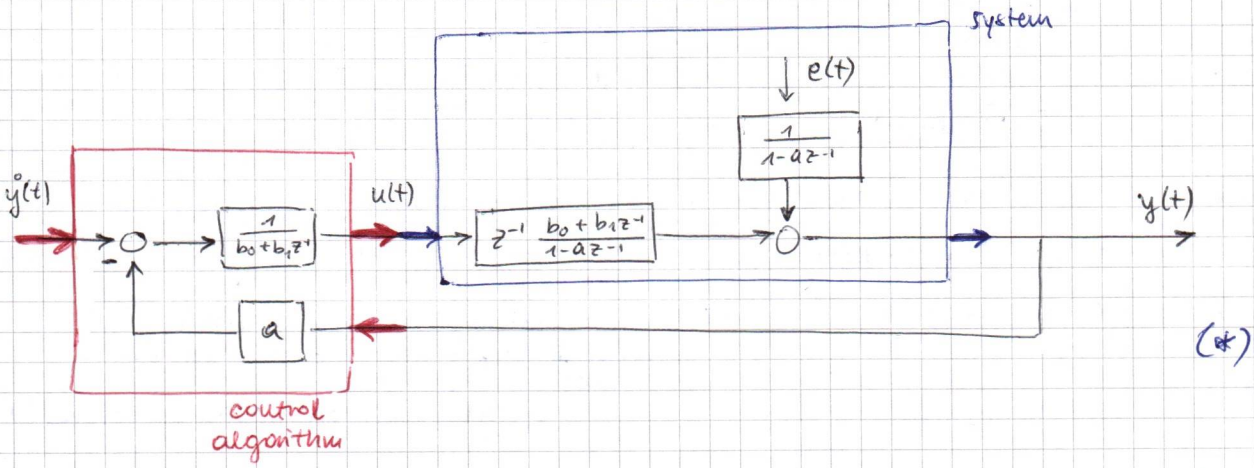
↑ we're making the shift because we want $u(t)$

$$\Longrightarrow \quad b_0\,u(t) + b_1\,u(t-1) + a\,y(t) = y^\circ(t)$$

$$\Longrightarrow \quad u(t) = \underbrace{(y^\circ(t+1)}_{\text{not available at time } t \text{ (no preview)}} - a\,y(t)) \cdot \frac{1}{b_0 + b_1 z^{-1}}$$

$\Longrightarrow$ we replace with the last available

$$\Longrightarrow \quad u(t) = (y^\circ(t) - a\,y(t)) \frac{1}{b_0 + b_1 z^{-1}} \quad := \text{CONTROL ALGORITHM}$$

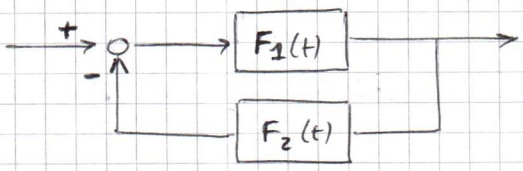


(*)

27/05

Analysis of the control system.
Analysis of stability? Analysis of performance?

For stability let's recall a result on feedback system:



Assumptions: negative feedback!

to check the closed-loop-stability:

- compute the so called "loop function": $L(z) = F_1(t)\,F_2(t)$
- build a "characteristic polynomial": $\chi(z) = L_N(z) + L_D(z)$ ← numerator and denominator of L
- find the roots of $\chi(z)$

$\Longrightarrow$ closed loop system is asymptotically stable if and only if all the roots of $\chi(z)$ are strictly inside the unit circle.

Applied in our system: (*)

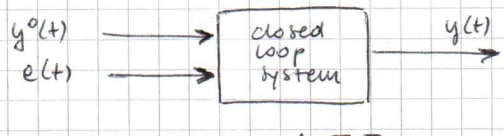
$$L(z) = \frac{1}{b_0 + b_1 z^{-1}} \cdot \frac{z^{-1}(b_0 + b_1 z^{-1})}{1 - a z^{-1}} \cdot a \qquad \text{(do not simplify)}$$

$$\chi(z) = a z^{-1}(b_0 + b_1 z^{-1}) + (1 - a z^{-1})(b_0 + b_1 z^{-1})$$

$$\pm (b_0 + b_1 z^{-1})(\cancel{a z^{-1}} + 1 - \cancel{a z^{-1}}) = b_0 + b_1 z^{-1} = B(z)$$

$\Longrightarrow$ closed loop is asymp. stable thanks to minimum phase assumption (which guarantees that the roots of $B(z)$ are strictly inside the unit circle)

Performance analysis:



since the system is L.T.I. (linear time invariant) we can use the "super position principle" = we can see the 2 inputs as acting indipendently

$$y(t) = \underbrace{F_{y^\circ y}(z)}_{\text{transfer function from } y^\circ \text{ to } y}\, y^\circ(t) + \underbrace{F_{ey}(z)}_{\text{t.f. from } e \text{ to } y}\, e(t)$$

Compute:

$$F_{y^\circ y}(z) = \frac{\boxed{\frac{1}{b_0+b_1z^{-1}} \cdot \frac{z^{-1}(b_0+b_1z^{-1})}{1-az^{-1}}}}{1 \;(+)\; \boxed{\frac{1}{b_0+b_1z^{-1}} \; z^{-1} \; \frac{b_0+b_1z^{-1}}{1-az^{-1}} \cdot a}} = [\dots] = z^{-1}$$

(direct line from input to output; (+) negative feedback; boxed: loop function)

$$F_{ey}(z) = \frac{\frac{1}{1-az^{-1}}}{1 + \boxed{\text{loop function}}} = [\dots] = 1$$

Notice that the closed loop system has a very simple closed-loop behaviour:

$$y(t) = z^{-1}\, y^\circ(t) + 1 \cdot e(t)$$

$$y(t) = y^\circ(t-1) + e(t) \longrightarrow$$ $y(t)$ follows exactly $y^\circ(t)$ (but with 1-step delay), disturbed by noise $e(t)$ (best possible solution (optimal control))

General solution to M.V. control problem: (Minimum Variance)

$$S: \quad y(t) = \frac{B(z)}{A(z)}\, u(t-k) + \frac{C(z)}{A(z)}\, e(t) \qquad e(t) \sim N(0, \lambda^2)$$

- $b_0 \neq 0$
- $B(z)$ has all roots strictly inside the unit circle (S is in minimum phase)
- $C(z)/A(z)$ is canonical representation
- $y^\circ(t) \perp e(t)$
- $y^\circ(t)$ is un-predictable (no-preview) $\longrightarrow$ $\hat{y}^\circ(t+1|t) = y^\circ(t)$

Goal: $\min J = \min \mathbb{E}\left[\underbrace{(y(t) - y^\circ(t))^2}_{\text{tracking error}}\right]$

"Trick": $y(t) = \hat{y}(t|t-k) + \varepsilon(t)$

$$\Longrightarrow J = \mathbb{E}\left[(\hat{y}(t|t-k) + \varepsilon(t) - y^\circ(t))^2\right]$$

$$= \mathbb{E}\left[((\hat{y}(t|t-k) - y^\circ(t)) - \varepsilon(t))^2\right]$$

$$= \mathbb{E}\left[(\hat{y}(t|t-k) - y^\circ(t))^2\right] + \mathbb{E}\left[\varepsilon(t)^2\right] - 2\,\mathbb{E}\left[(\hat{y}(t|t-k) - y^\circ(t))\,\varepsilon(t)\right]$$

$$= \mathbb{E}\left[(\hat{y}(t|t-k) - y^\circ(t))^2\right] + \left[\begin{array}{c}\text{function of}\\ \text{only } e(t)\\ (\perp u(t))\end{array}\right]$$

$\varepsilon(t) \perp y(t)$ (construction)
$\varepsilon(t) \perp y^\circ(t)$ (assumptions)

$\Longrightarrow$ optimal solution: $\hat{y}(t|t-k) = y^\circ(t)$

$\Longrightarrow$ we need to compute: $\hat{y}(t|t-k)$

General formula for ARMAX predictor:

Re-write: $\frac{C(z)}{A(z)} = E(z) + \frac{\tilde{R}(z)\,z^{-k}}{A(z)}$

$E(z)$: solution of long division $C(z)/A(z)$ of k steps

$\tilde{R}(z)$: residual of long division $C(z)/A(z)$ of k steps

$$\hat{y}(t|t-k) = \frac{B(z)E(z)}{C(z)}u(t-k) + \frac{\tilde{R}(z)}{C(z)}y(t-k)$$

We shift ahead k steps:

$$\hat{y}(t+k|t) = \frac{B(z)E(z)}{C(z)}u(t) + \frac{\tilde{R}(z)}{C(z)}y(t)$$

Now impose: $\hat{y}(t+k|t) = y^0(t+k)$

at time t we don't know $y^0(t+k)$ $\Longrightarrow$ we replace it with $y^0(t)$

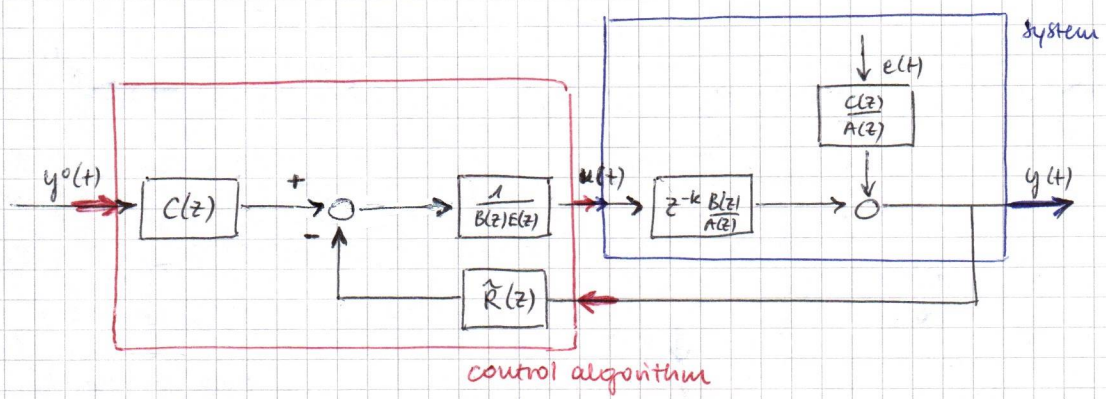
$$\Longrightarrow \hat{y}(t+k|t) = y^0(t)$$

$$\Longrightarrow \frac{B(z)E(z)}{C(z)}u(t) + \frac{\tilde{R}(z)}{C(z)}y(t) = y^0(t)$$

$\Longrightarrow$ control algorithm by making $u(t)$ explicit:

$$u(t) = \frac{1}{B(z)E(z)}\left(C(z)y^0(t) - \tilde{R}(z)y(t)\right) \quad := \text{CONTROL ALGORITHM}$$

(M.V. CONTROL GENERAL FORMULA)



Stability check:

$$L(z) = \frac{1}{B(z)E(z)} \cdot \frac{z^{-k}B(z)}{A(z)}\tilde{R}(z)$$

$$\chi(z) = L_N(z) + L_D(z)$$

$$= z^{-k}B(z)\tilde{R}(z) + B(z)E(z)A(z)$$

$$= B(z)\underbrace{\left(z^{-k}\tilde{R}(z) + E(z)A(z)\right)}_{C(z)}$$

$$= B(z)C(z)$$

$\Longrightarrow$ System in closed loop is asymp. stable if and only if:

- all roots of $B(z)$ are stable $\rightarrow$ system in min. phase $\Rightarrow$ ✓ ✓
- all roots of $C(z)$ are stable $\rightarrow$ system in canonical rep. $\rightarrow$ ✓

Performance analysis:

$$y(t) = F_{y^0y}(z)\,y^0(t) + F_{ey}(z)\,e(t)$$

$$F_{y^0y}(z) = \frac{C(z)\cdot\frac{1}{B(z)E(z)}\cdot\frac{z^{-k}B(z)}{A(z)}}{1 + \left[\frac{1}{B(z)E(z)}\frac{z^{-k}B(z)}{A(z)}\tilde{R}(z)\right]} = \dots = z^{-k}$$

$$Fey(z) = \frac{\frac{C(z)}{A(z)}}{1 + \left[\frac{1}{B(z)A(z)} \frac{z^{-k}B(z)}{A(z)} \hat{K}(z)\right]} = \dots = E(z)$$

At closed loop the behaviour is very simple:

$$y(t) = y^o(t-k) + E(z)\,e(t)$$

$y(t)$ exactly follows the desired value $y^o(t)$ (but with k-steps delay) and it's disturbed by noise $E(z)\,e(t)$

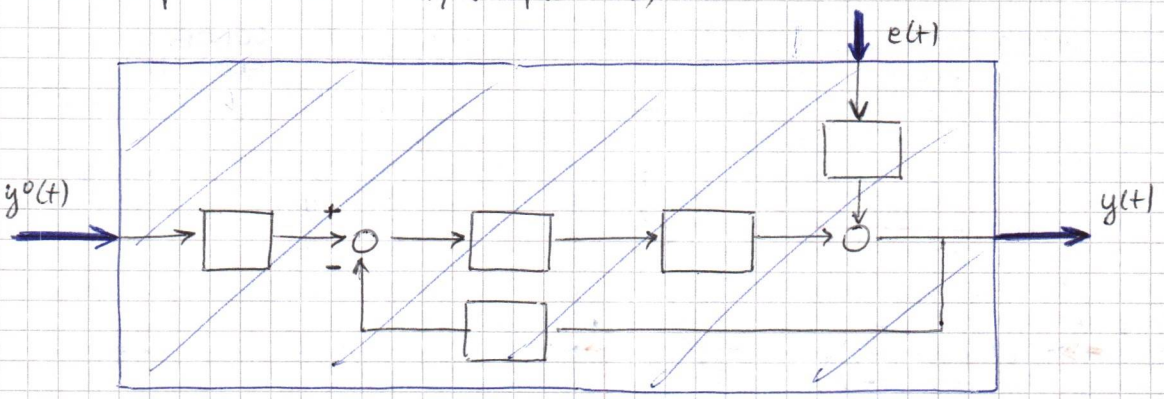
best possible solution

prediction error k-steps ahead

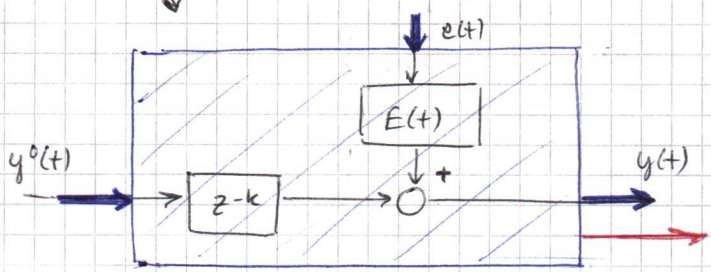
The ideal perfect behaviour would be:

$$y(t) = y^o(t) + 0$$

Remark: closed loop behaviour is very simple (?)



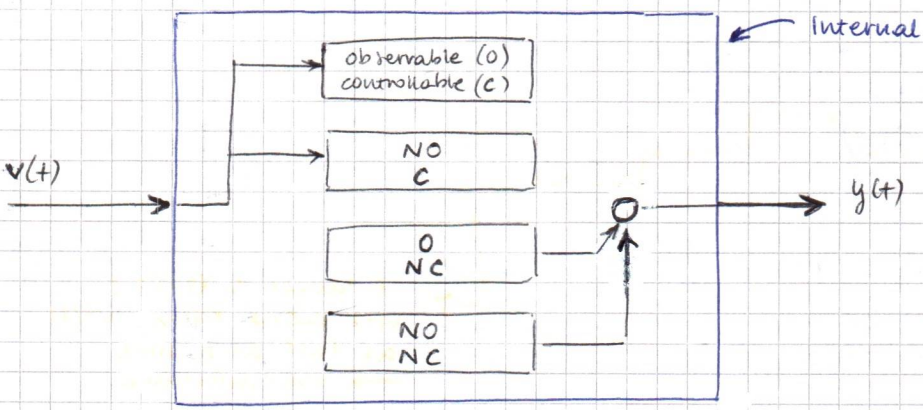
equivalent I/O perspective:



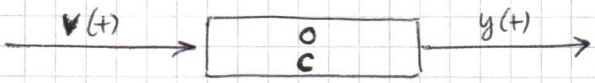
from the outside these two are completely indistinguible

Where are all the poles of the original system?

Remember:



equivalent to: (external view):



MV controller pushes all the system poles into the N.O. (not observable) or N.C. (not controllable) part of the system (it makes internal cancellation). It is a problem? No, we verified that this is internally asymptotic stable.

Main limits of M.V.C. ?

- Can be applied only to minimum phase systems
- We cannot moderate the control / activation effort
- We cannot design a specific behaviour from $y^0(t)$ to $y(t)$

To overcome these limits there is an extension of M.V.C. : G.M.V.C.. (generalized M.V.C.)

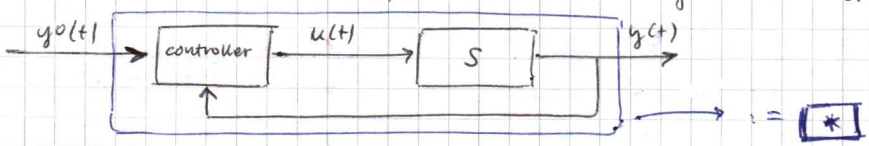
The main difference is the extension of the performance index:

MVC : $J = \mathbb{E}\left[(y(t) - y^0(t))^2 \right]$

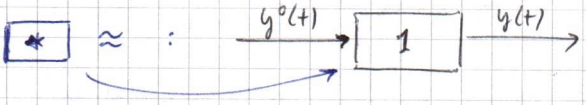
GMVC: $J = \mathbb{E}\left[\left(P(z)\, y(t) - y^0(t) + Q(z)\, u(t) \right)^2 \right]$

- $P(z)$: transfer function called "Reference model"
- $Q(z)$: transfer function that, multiplied by $u(t)$, makes a penalty to big values of $u(t)$: $Q(z)$ big means control solutions "moderates" the use of $u(t)$

Remark (on the reference model $P(z)$): Consider the general feedback system:

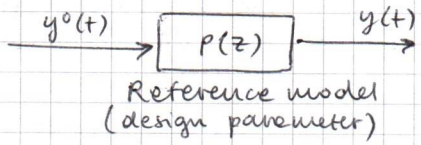


typical goal: obtain the best possible tracking:



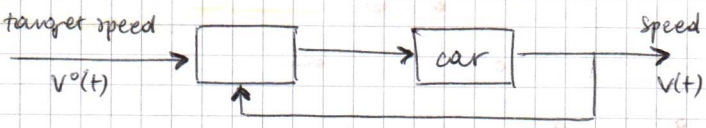
However, in many cases, fast/perfect tracking is not the best solution, but the best solution is to track a "reference model"

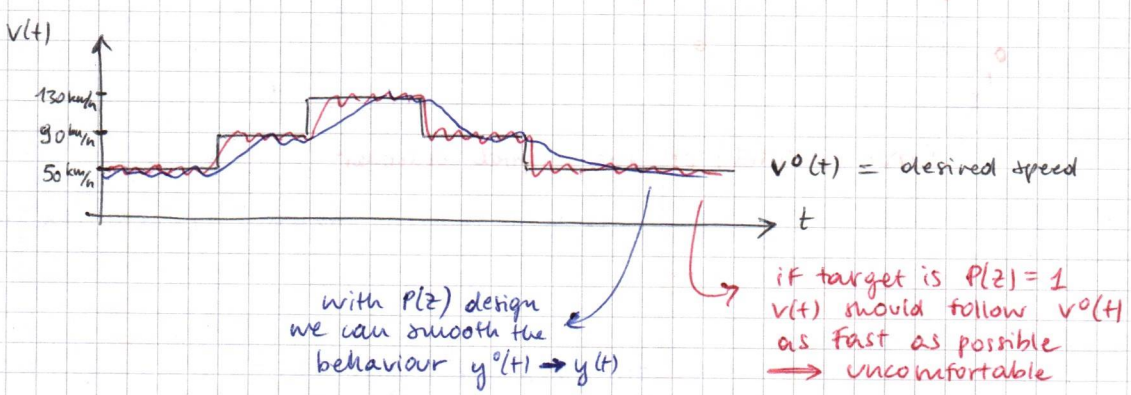
Instead of trying to obtain [1] we try to obtain [P(z)]



Reference model (design parameter)

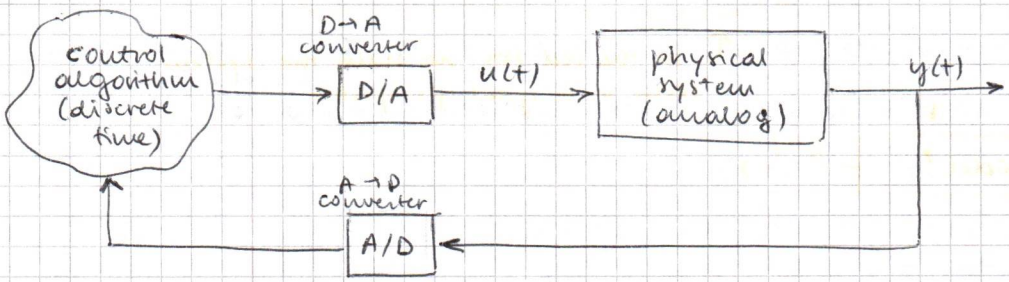
Example: cruise control in a car:



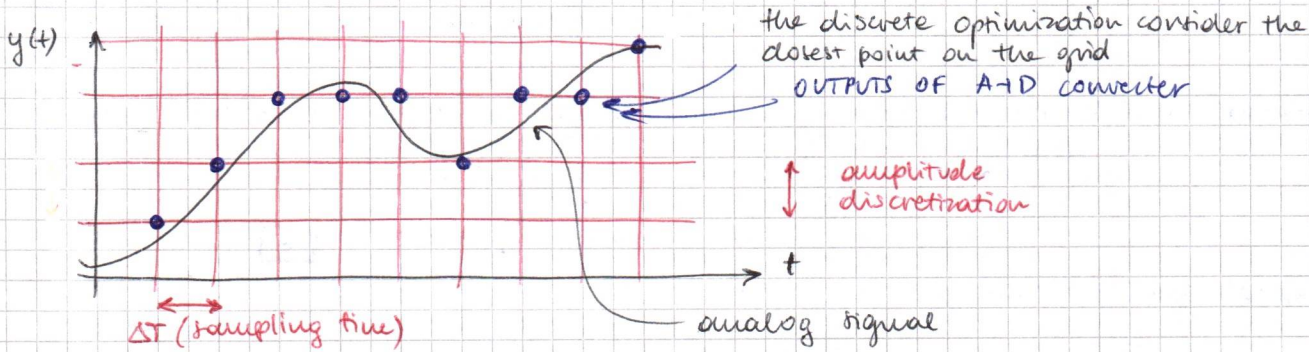


APPENDIX TO CHAPTER 6: 04/06

DISCRETIZATION OF AN ANALOG SYSTEM
(valid for both identification and control)



A → D converter : (analog → digital)

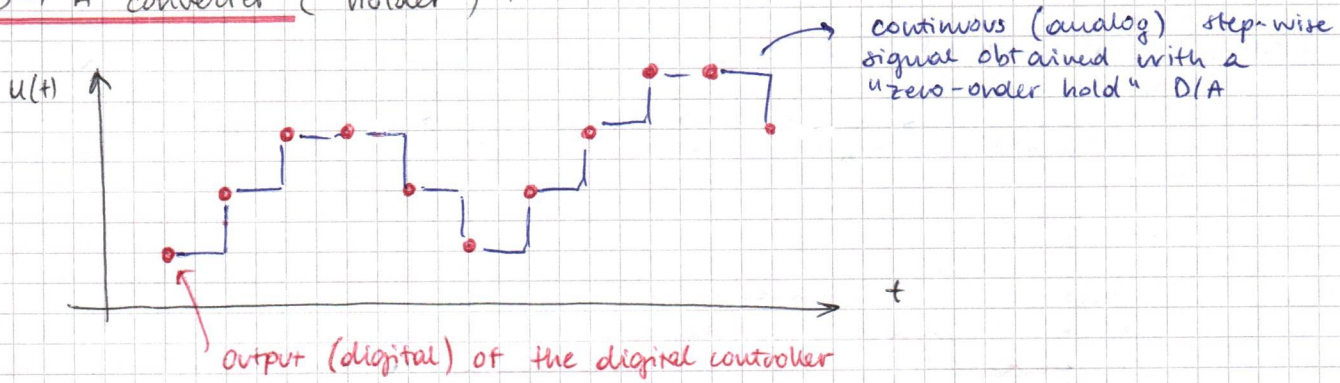


Time discretization: ΔT = sampling time
Amplitude discretization: total number of levels used for discretization

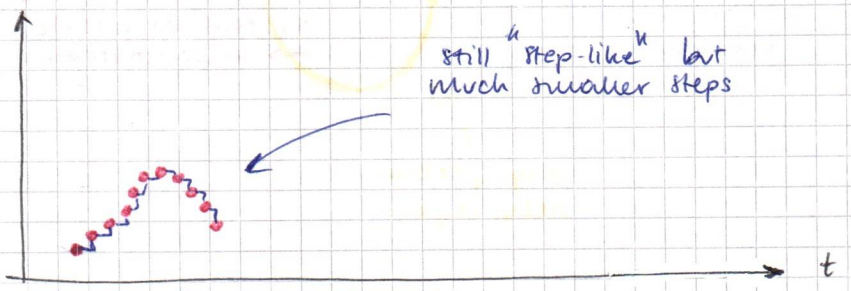
(**Ex.** "10-bits discretization" $\Longrightarrow 2^{10} = 1024$ levels of amplitude)

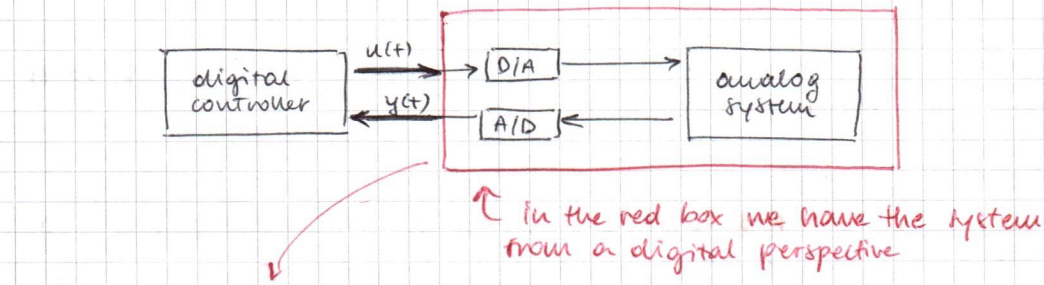
High quality A/D converter :
- can use small ΔT
- high number of levels (16 bits, ..)

D → A converter ("holder") :

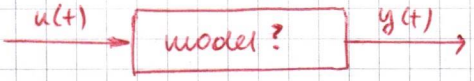


If ΔT is sufficiently small the step wise analog signal is very similar to a smooth analog signal:





↑ in the red box we have the system from a digital perspective



Model ? 2 cases:

- we make BB (Black Box) system identification from measured data → directly estimate a discrete time model
- we have a physical WB model (continuous model) → we need to discretize it

The most used approach is the state-space transformation

$$S: \begin{cases} \dot{x} = Ax + Bu \\ y = Cx + (Du) \end{cases} \xrightarrow{\text{sampling time } \Delta T} S: \begin{cases} x(t+1) = Fx(t) + Gu(t) \\ y(t) = Hx(t) + (Du(t)) \end{cases}$$

A, B, C, D — continuous time

F, G, H, D — discrete time

Transformation formulas:

$$F = e^{A\Delta T}$$

$$G = \int_0^{\Delta T} e^{As} B \, ds$$

$$H = C$$

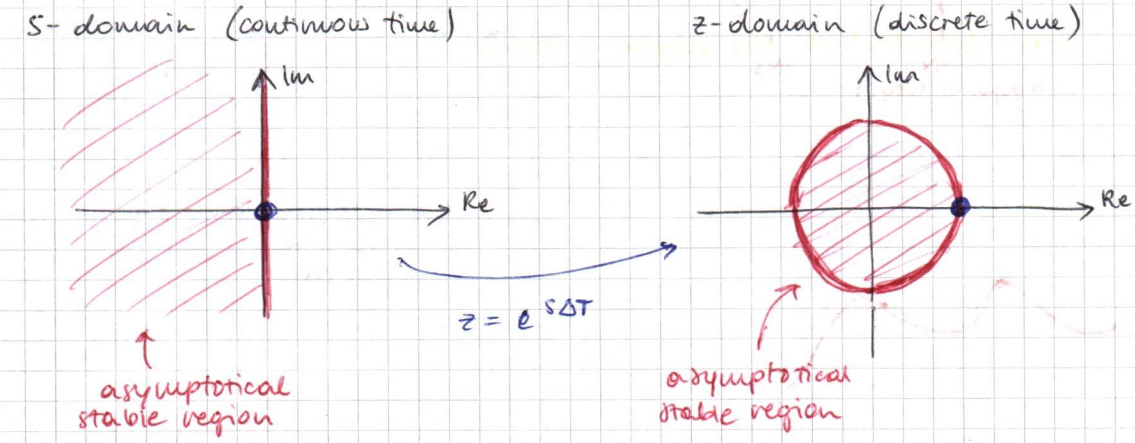
$$D = D$$

Remark: How the poles of the continuous time system are transformed? (eig(A) → eig(F) ?). It can be proven that the eigenvalues (/poles) follow the "sampling transformation rule":

$$z = e^{s \cdot \Delta T} \Longrightarrow \lambda_F = e^{\lambda_A \Delta T}$$

z - domain (discrete time)

s - domain (continuous time)



How the zeros of the systems in continuous time are transformed in zeros in discrete time? No simple rule like poles!

We can only say:

$$G(s) = \frac{\text{polynomial in } s \text{ with } h \text{ zeros}}{\text{polynomial in } s \text{ with } k \text{ poles}}$$ if $G(s)$ is strictly proper $\Rightarrow$ $k >> h$

we apply the discretization rule:

$$G(z) = \frac{\text{polynomial in } z \text{ with } k-1 \text{ zeros}}{\text{polynomial in } z \text{ with } k \text{ poles}} \longrightarrow G(z) \text{ with relative degree} = 1$$

$\Longrightarrow$ we have "new" $k-h-1$ zeros that are generated by the discretization. They're called "**hidden zeros**". Unfortunately, these hidden zeros are frequently outside the unit circle.
$\Longrightarrow$ $G(z)$ is non minimum phase even if $G(s)$ is minimum phase
$\Longrightarrow$ we need for instance GMVC to design the control system

Another simple discretization technique frequently used is the discretization of the derivative $\dot{x}$.

Euler backward:

$$\dot{x}(t) \simeq \frac{x(t) - x(t-1)}{\Delta T} = \frac{x(t) - z^{-1}x(t)}{\Delta T} = \frac{z-1}{z\,\Delta T}\, x(t)$$

Euler forward:

$$\dot{x}(t) \simeq \frac{x(t+1) - x(t)}{\Delta T} = \frac{z-1}{\Delta T}\, x(t)$$

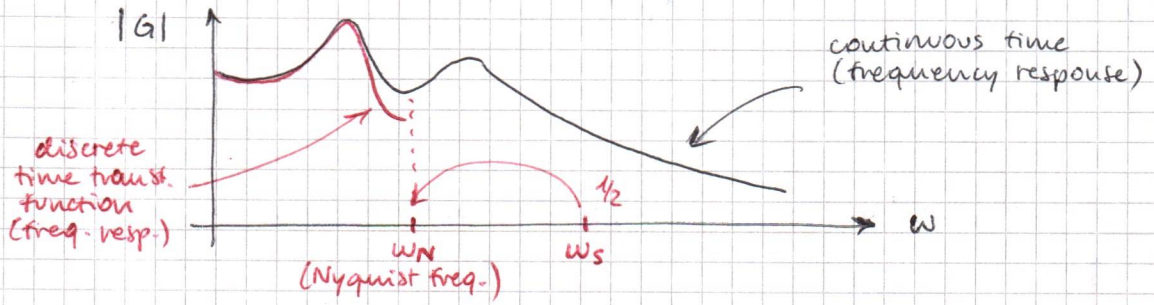
General formula for this approach:

$$\dot{x}(t) = \left[\frac{z-1}{\Delta T} \quad \frac{1}{\alpha z + (1-\alpha)} \right] x(t)$$

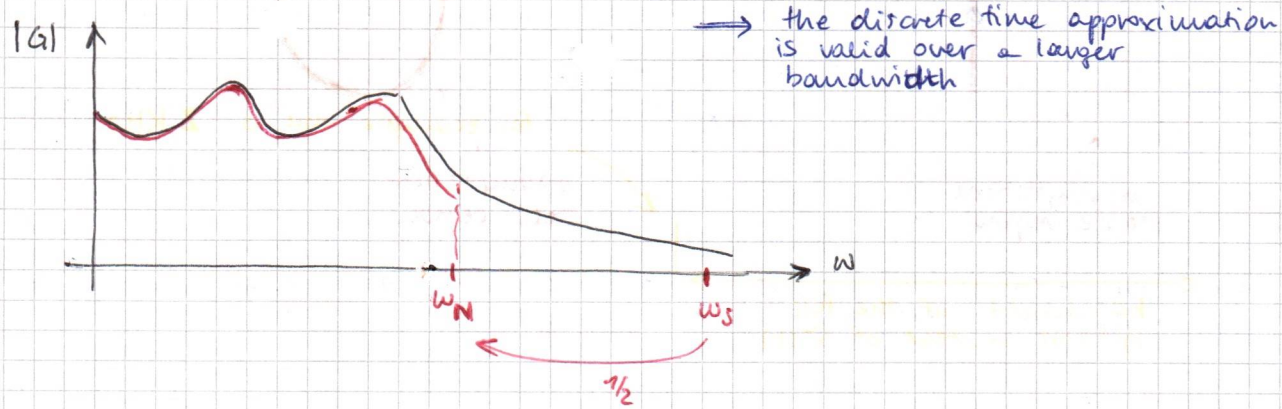
$0 \le \alpha \le 1$ $\longrightarrow$ special cases:

- $\alpha = 0$ $\Longrightarrow$ Euler forward
- $\alpha = 1$ $\Longrightarrow$ Euler Backward
- $\alpha = 1/2$ $\Longrightarrow$ TUSTIN METHOD

Critical choise is ΔT (sampling time) ($\Longleftrightarrow$ or sampling frequency $f_s = \frac{1}{\Delta T}$).
The general intuitive rule is: the smaller ΔT is, the better.



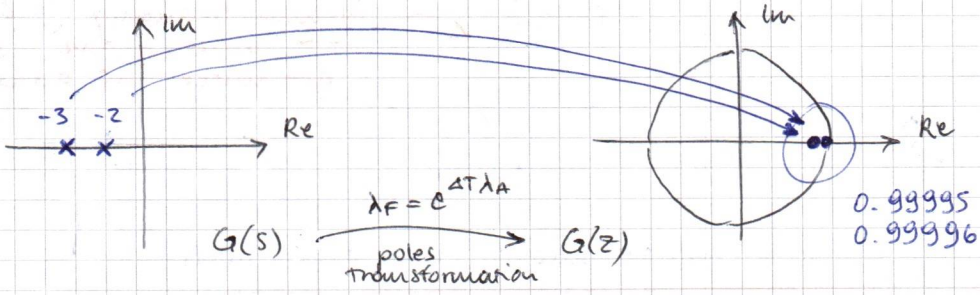
If ΔT is smaller (w_s larger):



$\Delta T \longrightarrow f_s = \frac{1}{\Delta T} \longrightarrow \omega_s = \frac{2\pi}{\Delta T} \longrightarrow \begin{cases} f_N = \frac{1}{2} f_s \\ \omega_N = \frac{1}{2} \omega_s \end{cases}$

Hidden problems of a "too small" ΔT :

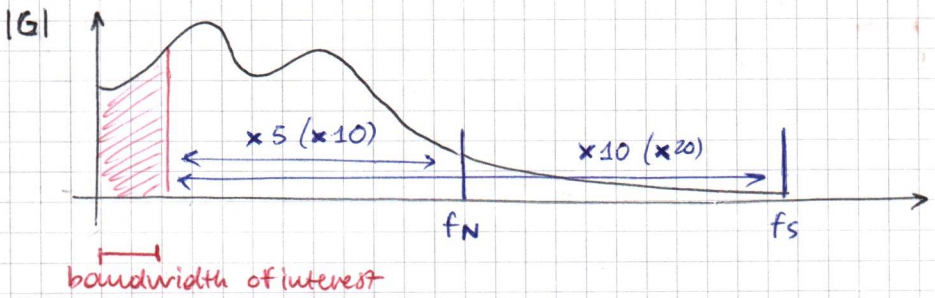
- sampling devices (A/D and D(A) cost
- computational cost → update an algorithm every 1 μs is much heavier than updating at 1 ms
- cost of memory (if data logging is needed)
- numerical precision cost (again is a hidden computational cost)



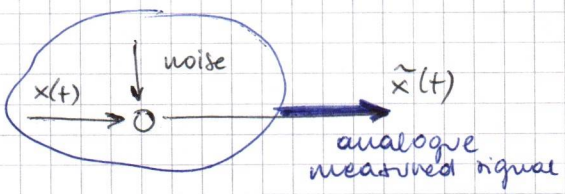
if ΔT is very small ($\to 0$) we "squeez" all the poles very close to (1,0)

⟹ we need very high numerical precision (⇒ use a lot of digits) to avoid instability

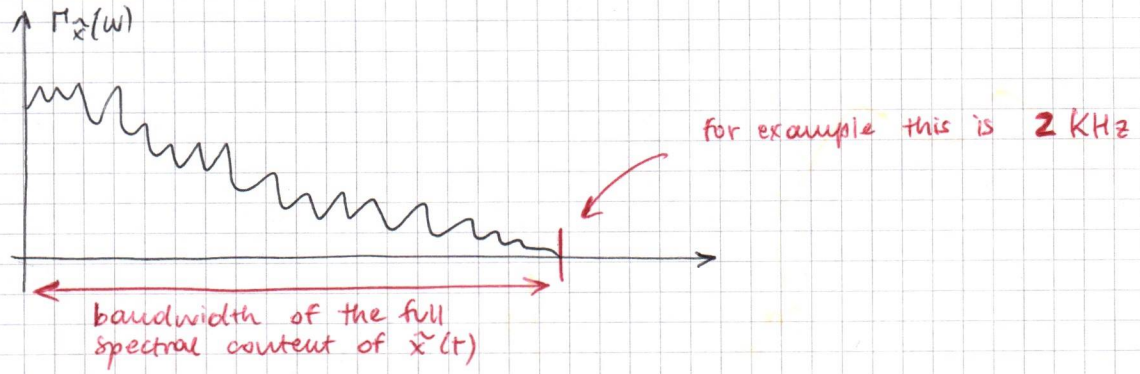
Rule of thumb of control engineers :
f_s is between 10 (min) and 20 (max) times the system bandw. we are interested in



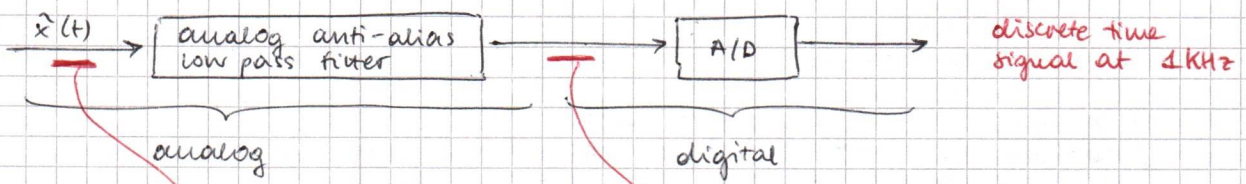
Final remark : another way of managing the choice of ΔT w.r.t. the ALIASING problem (signal-processing perspective)



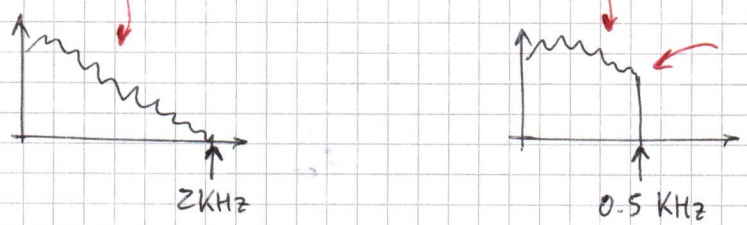
Spectrum of the signal:



Classical way to deal with aliasing is to use ANTI-ALIAS ANALOG FILTERS:



For example, if A/D is at 1 KHz ($\Delta T = 1$ ms):



Digital (full digital) approach without analog anti-alias filters: