MPC

INTRODUCTION

- SOFT CONSTRAINT
- INTEGRAL ACTION
- DESIGN CHOICES

Advanced and Multivariable Control

Model Predictive Control – Part 1

↳ lot of applications in physical system!

⇓

$\left\{ \begin{array}{l} \text{relevant control} \\ \text{technique nowday} \end{array} \right\}$

Riccardo Scattolini

General characteristics of MPC

not only for large syst

- By far the ***most popular advanced control method*** in industry (see the following slides) and in embedded applications. All the main automation companies (ABB, Siemens, …) have SW tools for its implementation

  → chemical companyes → problem unsolvable with classic approach

- Developed in the late ’70 and early ’80 in the process industry to cope with large scale systems, ***constraints*** on the process variables, time varying reference signals (known in advance)

  ↳ constrain to deal with! a priori ↳ tracking of traj of mobile elements! New theory needed

- The basic idea is ***to transform the control synthesis problem into an optimization one***. A finite horizon control problem is stated and solved

  (with suitable SW tools) (optimization problem!) → to optimize, giving a sequence of future control action, and solve iteratively

  → identification procedure

- ***Empirical models*** (impulse/step response) can be used to reduce the time required by the design phase

  with MPC design phase you use this model

  ↳ In large scale system, you have complex dym., a phisical model is hard to obtain ↳ easyer to build by experiments

- ***Nonlinear models*** obtained through physical modeling can be used

  (you can linearize … OR include NONLIN model on MPC) by efficent SW

- With respect to finite horizon optimal control previously studied, a ***time invariant control law*** is obtained by means of the ***Receding Horizon principle***, see next slides,

  optimal control ⇒ but it lead to ⇑ a defention of time variant finite orizon implementation of MPC survey

S.J. Qin, T.A. Badgwell / Control Engineering Practice 11 (2003) 733–764 745

many company realize/implement MPC...

different field of usage of MPC

[Many application on real plant]

Table 6
Summary of linear MPC applications by areas (estimates based on vendor survey; estimates do not include applications by companies who have licensed vendor technology)[a]

Area	Aspen Technology	Honeywell Hi-Spec	Adersa[b]	Invensys	SGS[c]	Total
Refining	1200	480	280	25		1985
Petrochemicals	450	80	—	20		550
Chemicals	100	20	3	21		144
Pulp and paper	18	50	—	—		68
Air & Gas	—	10	—	—		10
Utility	—	10	—	4		14
Mining/Metallurgy	8	6	7	16		37
Food Processing	—	—	41	10		51
Polymer	17	—	—	—		17
Furnaces	—	—	42	3		45
Aerospace/Defense	—	—	13	—		13
Automotive	—	—	7	—		7
Unclassified	40	40	1045	26	450	1601
Total	1833	696	1438	125	450	4542
First App.	DMC:1985 IDCOM-M:1987 OPC:1987	PCT:1984 RMPCT:1991	IDCOM:1973 HIECON:1986	1984	1985	
Largest App.	603 × 283	225 × 85	—	31 × 12	—	

big number of variables to deal with!

[a] The numbers reflect a snapshot survey conducted in mid-1999 and should not be read as static. A recent update by one vendor showed 80% increase in the number of applications.
[b] Adersa applications through January 1, 1996 are reported here. Since there are many embedded Adersa applications, it is difficult to accurately report their number or distribution. Adersa's product literature indicates over 1000 applications of PFC alone by January 1, 1996.
[c] The number of applications of SMOC includes in-house applications by Shell, which are unclassified. Therefore, only a total number is estimated here.

lot industrial field where slow time constant.. in fact with MPC you iterate optimiz., that require time! → now BIG computing power → used in fast applications today *some data from*

Economic assessment of advanced process control – a survey and framework

Margret Bauer, Ian K. Craig

Journal of Process Control 18 (2008) pp. 2–18

(most recent)

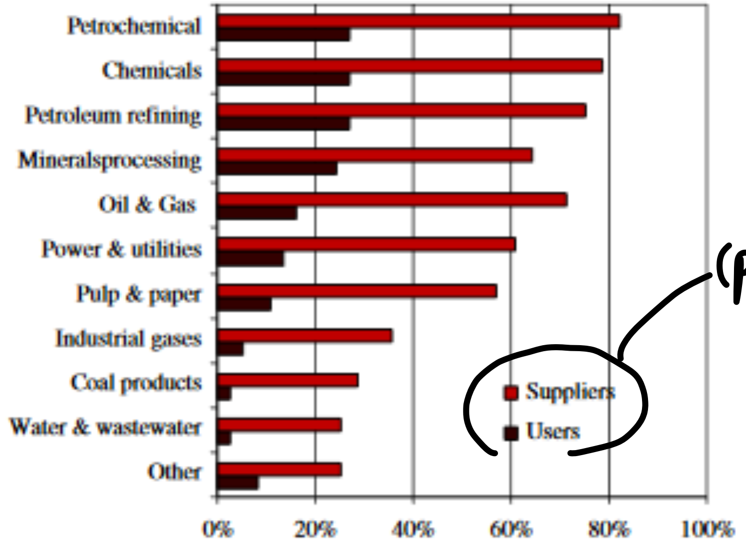
• how things work in industry ?

technology improvement:
- when unable to solve
- when allow gain
⇒ necessity to improve

A key objective of industrial advanced process control (APC) projects is to stabilize the process operation. In order to justify the cost associated with the introduction of new APC technologies to a process, ***the benefits have to be quantified in economic terms***. In the past, economic assessment methods have been developed that link the variation of key controlled process variables to economic performance quantities. This paper reviews these methods and incorporates them in a framework for the economic evaluation of APC projects. A web-based survey on the economic assessment of process control has been completed by over 60 industrial APC experts. The results give information about the state-of-the-art assessment of economic benefits of advanced process control.

(Bauer, Craig, 2008)

- Economic assessment of Advanced Process Control (APC)

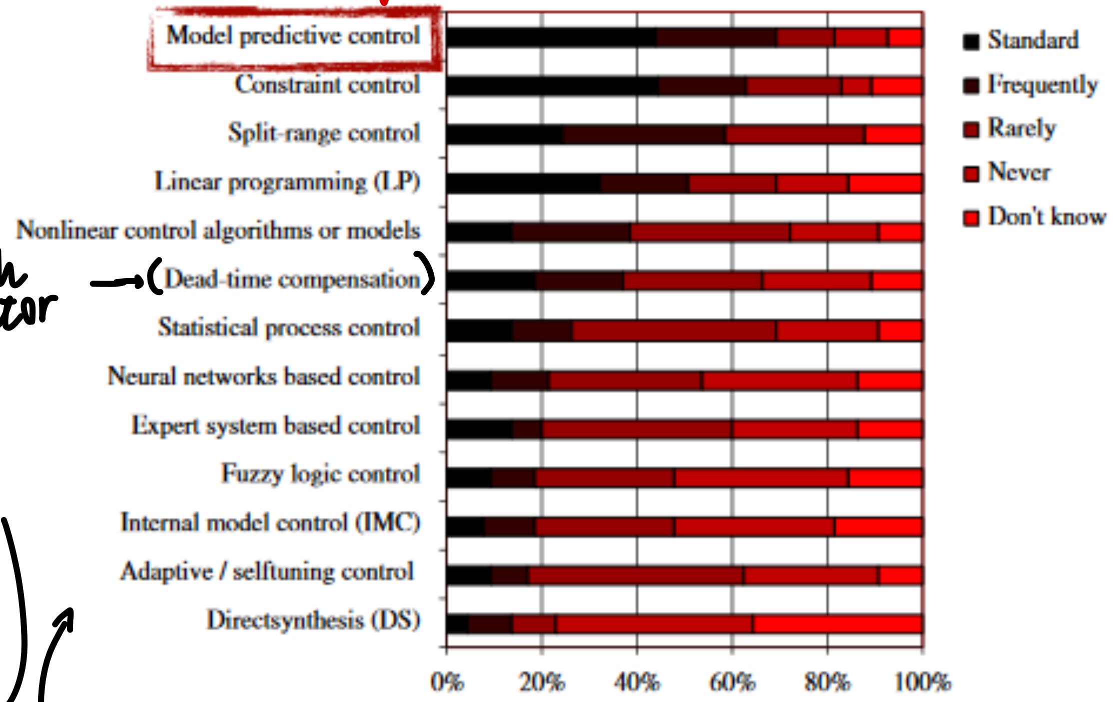


(partitioned)

participants of APC survey by industry (worldwide)

(most popular technique of control!)

smith predictor



(divided into specific family)

Industrial use of APC methods: survey results survey

(Samad, IEEE CS Magazine, 2017)

- Impact of advanced control technologies in industry

TABLE 1 A list of the survey results in order of industry impact as perceived by the committee members.

most PID popular...

nowday most used controller

Rank and Technology	High-Impact Ratings	Low- or No-Impact Ratings
PID control	100%	0%
Model predictive control	78%	9%
System identification	61%	9%
Process data analytics	61%	17%
Soft sensing	52%	22%
Fault detection and identification	50%	18%
Decentralized and/or coordinated control	48%	30%
Intelligent control	35%	30%
Discrete-event systems	23%	32%
Nonlinear control	22%	35%
Adaptive control	17%	43%
Robust control	13%	43%
Hybrid dynamical systems	13%	43%

NOT so popular on that application

basic idea behind MPC:
(it is the only Advanced control techn. in practical problem) → [courses to train operator to use MPC]

↓

How is it explained in industrial courses?

⇒ process operator

Let's have a look at it, then we will place it in the context of this course main idea

MPC is based on the knowledge of a **dynamic model of the system** in order to compute the **future evolution of the controlled variables** as function of **the future evolution of the control inputs**. The input sequence is computed minimizing a **cost function under state, input, and output constraints**

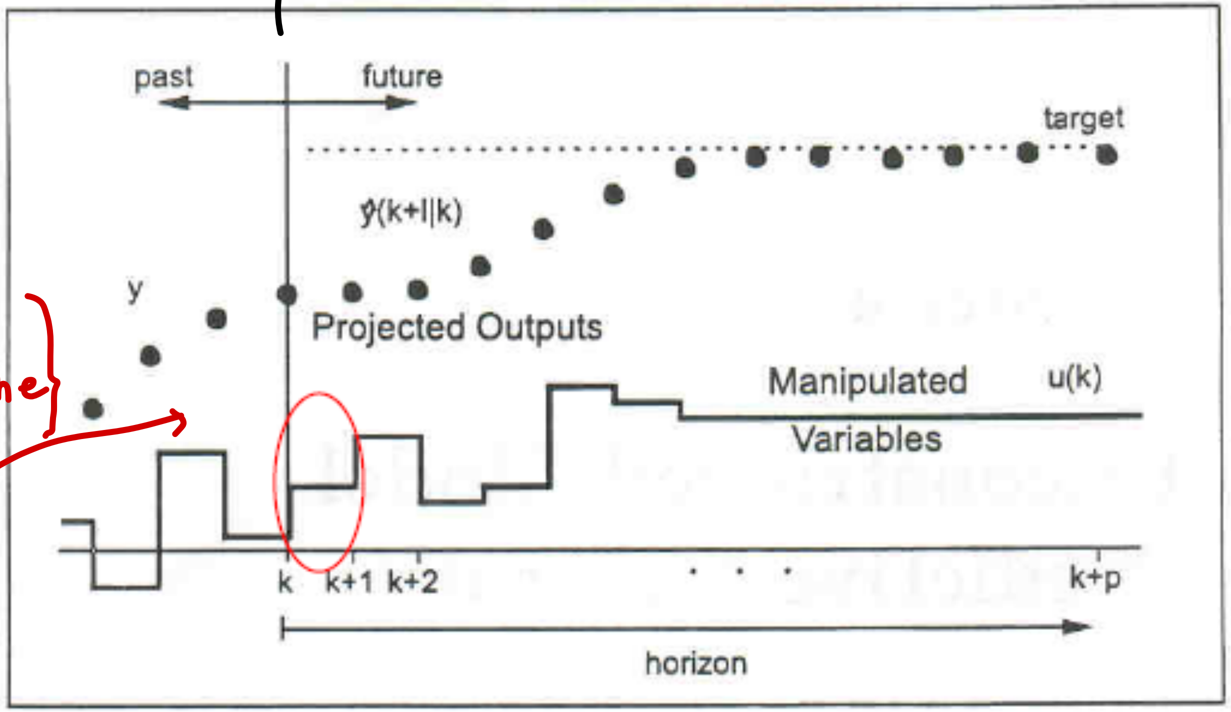
@ k optimize future evolution of output from k to k+p

⇐ being @ time k, having the states

time inv. control law by apply just the first control output

⇓

OPTIMIZATION: under future state/input/outp constraints



obtain control sequence solving a suitable optimiz. problem

↳ providing best output sequence

At time *k* the future sequence of control variables is computed, but only its first value is used

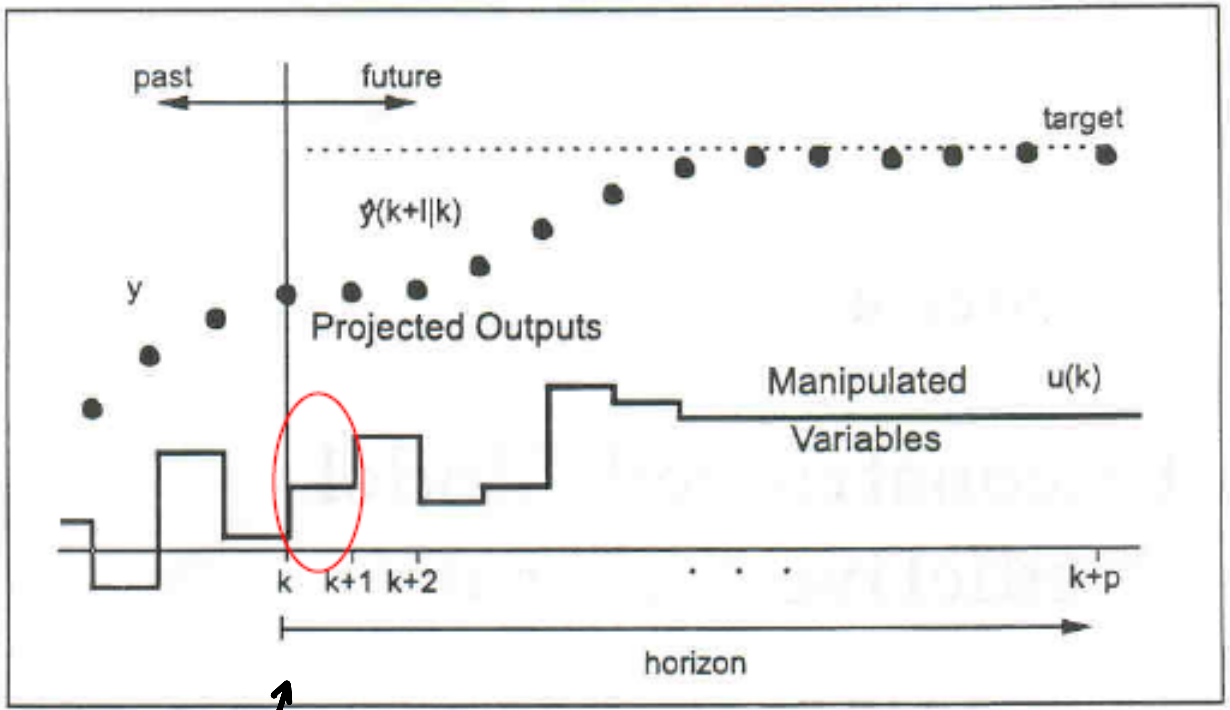
At time *k+1* the optimization procedure is repeated with the same prediction horizon

repeat overall optimiz. problem → mooving the prediction horizon one step in advance

↳ strategy named

- ***Receding Horizon***
- ***Rolling Horizon***
- ***Moving Horizon***

approach used in different field



repeat optimiz. NOT taking previous u(k+1) computed

CHESS game

you predict in advance moove, apply the FIRST and then after the action you re-plan the action

DRIVING

↑ look foreward to decide what to do, than reupdate decision

In this way, a time invariant control law is obtained

respect instant k where you are... for same conditions, same sequence obtained if reoptimize

time inv. C.L

↓

you compute $u(k)$ sequence, but @ K depends on state @ time K. BUT at different K you are on same problem... repeated time invariant problem →

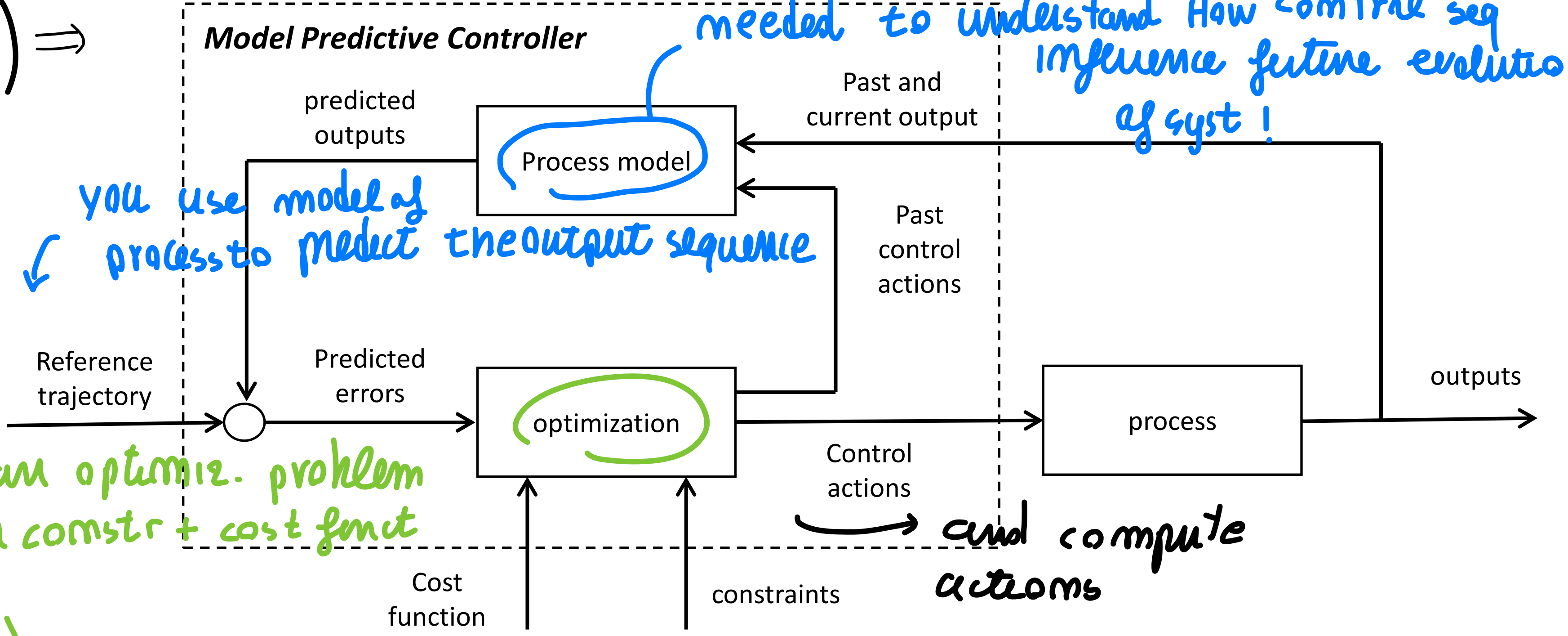
solution @ K depends on values @ K

(Basic structure Idea) ⇒

needed to understand how control seq influence future evolution of syst!

you use model of process to predict the output sequence

based on future ref traj + output



you solve an optimiz. problem with constr + cost funct

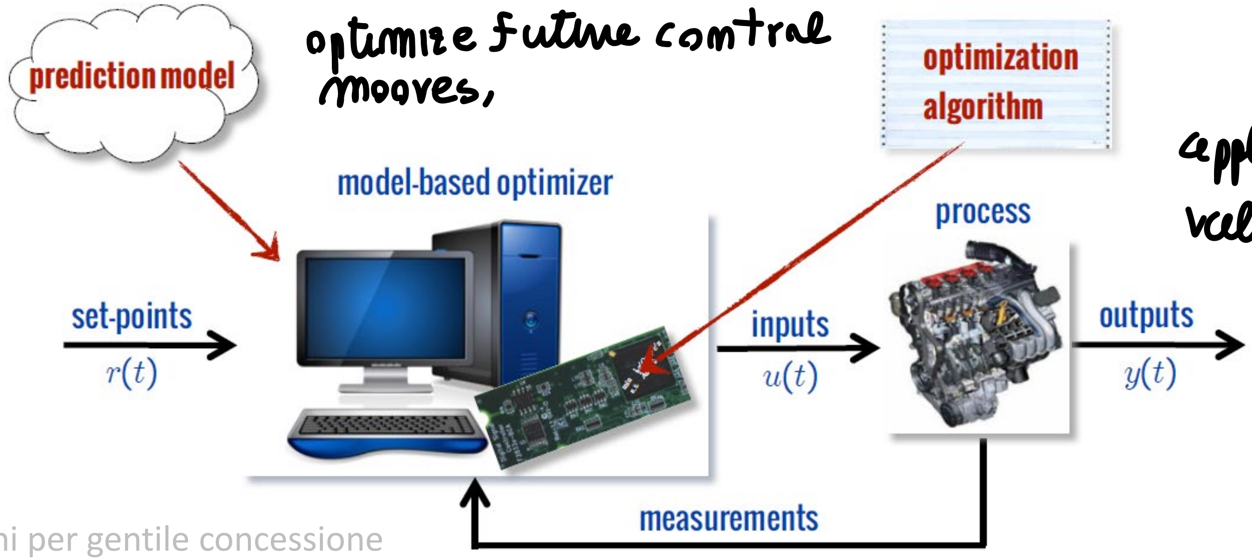
and compute actions

(OPTIMIZATION needed)

optimize future control mooves,

[Example]

Engine MPC:



apply first u(t) values and repeat optimiz.

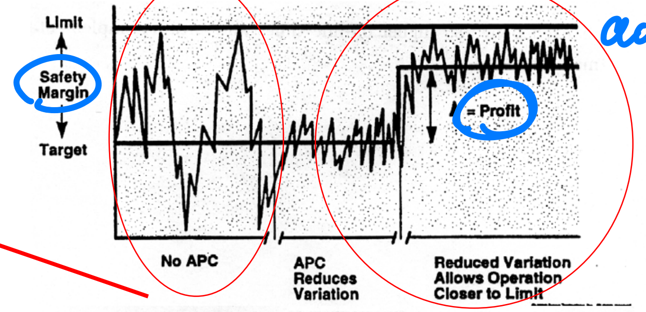
Why so succesfull ? for big systems! classical tech hard to use

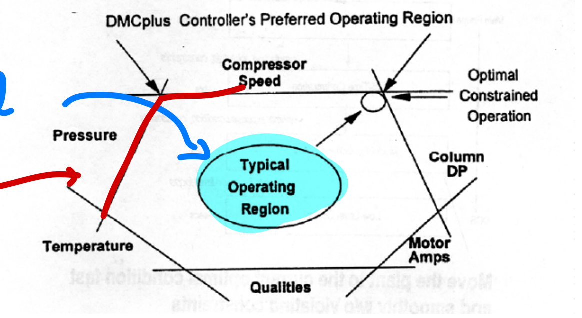
- centralized MPC control structure is better to use gaining power control

- More efficiency and tighter control with respect to traditional PID control schemes. This means that the reference signals can be set to values near the operating constraints, with economic advantages
- It can easily consider the knowledge of future external disturbances to improve the control action

(MPC based on future prediction! given control action)

Better control action !





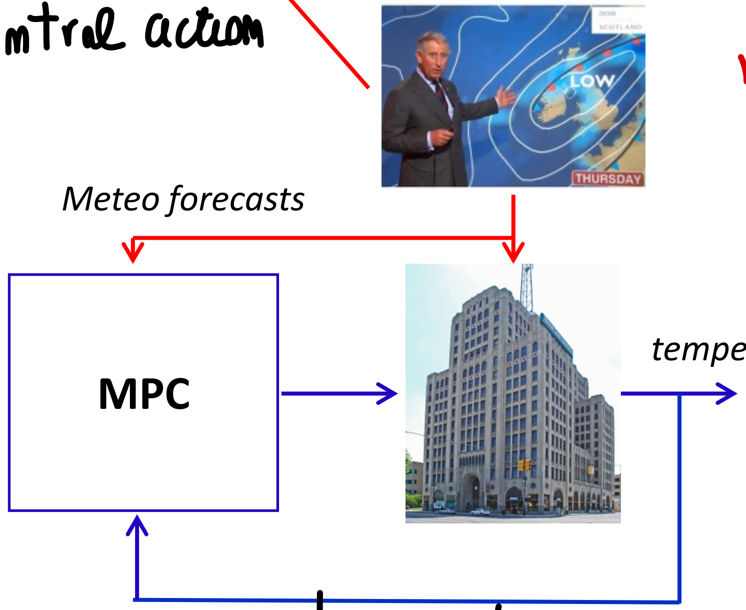
limiting the control

safety reason, you reduce the bound

using PID emulation performance! wide limit to guarantee

with limit margin, you must take better control action with good economic revenew, Better results!

great advantages! to control building Temperature big inertia of build. so low τ time... IF you predict what happen outside, you use knowledge to anticipate control action → predict in advance environment outside



In the overall control scheme, where to use MPC?

every control technique has its own application field..

carefully considery the problem request

each problem require different control technique

each block provide set point for lower level! ⇒ MPC has meanings if MORE actuators to control

plan-schedule future syst production

classical control pyramid

Supply chain optimization, planning, scheduling (1-7 giorni)

⇕

Real Time Optimization RTO (1-2 h)

→ given external condition, What is best operating point? static optimize, compute set point

⇕ ⇕

Model Predictive Control MPC (10-120 s)

at higher level centralized control

⇕ ⇕ ⇕

Distributed control systems PID (10-100 ms)

controlled by PI(D) with short τ

⇕ ⇕ ⇕ ⇕ ⇕

Actuators and sensors

cascade control scheme...

but now, let's proceed smoothly according to the course layout …

How to state MPC sol. and optimization problem

… and start from linear models, state feedback, state and control weighting in the cost function

(state available)

we see main idea! main implication

Anyway remember … MPC is a very large family of control algorithms, the goal in this course is to transmit the main ideas behind them. For the whole analysis of all the developments and applications of MPC 10 CFU would not be enough

overall approach to continue with different formulation

Finite Horizon LQ control

(link of MPC with what we already saw)

System

linear syst, known states measurable $x(k)$

$$
\begin{cases}
x(k+1) = Ax(k) + Bu(k) \\
y(k) = Cx(k)
\end{cases}
\qquad x \text{ measurable}
$$

Optimization problem

@ k you optimize respect sequence of future control move a cost fcnt

$$
\min_{u(k),\, u(k+1),\, \ldots,\, u(k+N-1)} J(x(k), u(\cdot), k) = \sum_{i=0}^{N-1} \left(\|x(k+i)\|_Q^2 + \|u(k+i)\|_R^2 \right) + \|x(k+N)\|_S^2
$$

prediction horizon

«classical» Finite Horizon optimal control problem already studied problem.. → solution based on HJB eq.

repeat same optimiz problem

$-K(0)$

Closed-loop solution (Riccati equation)

ITERATING

@ k+1 you have $u^o(k+1) = -K(0)x(k+1)$

[time inv. control law !]

$$u^o(k+i) = -K(i)x(k+i), \quad i = 0, 1, \ldots, N-1$$

gain over all prediction horiz. ⇒

$$K(i) = \left(R + B'P(i+1)B\right)^{-1} B'P(i+1)A$$

Gain given by that seq.

iterate backward ↳ P values

$$P(i) = Q + A'P(i+1)A - A'P(i+1)B\left(R + B'P(i+1)B\right)^{-1} B'P(i+1)A$$

$$P(N) = S$$

← initialize to final weight

we define K(i) time varying... ↴ How time invariant and ∞ horiz

Defined over a finite horizon and time varying. How to obtain a time invariant control law?

⇓

Receding Horizon (RH) principle

(time invariant)

time inv. c.l

because @ NEXT k time, you repeat seq and reuse this u(k) first

$$u^{MPC}(k) = -K(0)x(k)$$

apply only the first value obtained by optimiz.

No constraints on state and inputs *can be included*

With this kind of solution! neglect constraints! NOT included by HJB

→ how to include CONSTRAINTS ?!

 on cost funct J

another solution including constrain.. whe wanna min $\|X(k)\|^2$ which does NOT depend on u_k and min $\|u(k+i)\|$...

Open-loop solution

alternative idea..

Lagrange equation

$$x(k+i) = A^i x(k) + \sum_{j=0}^{i-1} A^{i-j-1} B u(k+j), \quad i > 0$$

state var. seq.

quadratic form in future x, u

consider the seq of future state var over predict horiz..

(depend on future control var)

Define

unkn. to compute

$$X(k) = \begin{bmatrix} x(k+1) \\ x(k+2) \\ \vdots \\ x(k+N-1) \\ x(k+N) \end{bmatrix}, \mathcal{A} = \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^{N-1} \\ A^N \end{bmatrix}, U(k) = \begin{bmatrix} u(k) \\ u(k+1) \\ \vdots \\ u(k+N-2) \\ u(k+N-1) \end{bmatrix}$$

state vector..

$$\mathcal{B} = \begin{bmatrix} B & 0 & 0 & \cdots & 0 & 0 \\ AB & B & 0 & \cdots & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ A^{N-2}B & A^{N-3}B & A^{N-4}B & \cdots & B & 0 \\ A^{N-1}B & A^{N-2}B & A^{N-3}B & \cdots & AB & B \end{bmatrix}$$

future state depends on initial state and future control var. over predict. horiz

so write the future state funct of that $\mathcal{A}, \mathcal{B}$ MATRIX

future state var

$$X(k) = \mathcal{A}x(k) + \mathcal{B}U(k)$$

state evolution along the prediction horizon as function of the initial state and future inputs

Infact looking

$$x(k+2) = A\,x(k+1) + Bu(k+1) = A^2 x(k) + ABu(k) + Bu(k+1)$$

$$x(k+1) = Ax(k) + Bu(k)$$

$$\underbrace{X}_{\text{future state vector}}(k) = \mathcal{A}\,x(k) + \mathcal{B}U(k) \Leftarrow$$

evolution of $x(k+i)$ equation in compact matrix form

Define the matrices, with N blocks on the diagonal

properly define Q, R matrix

$$\mathcal{Q} = \begin{bmatrix} Q & 0 & \cdots & 0 & 0 \\ 0 & Q & \cdots & 0 & 0 \\ \cdots & \cdots & \ddots & \cdots & \cdots \\ 0 & 0 & \cdots & Q & 0 \\ 0 & 0 & \cdots & 0 & S \end{bmatrix} \geq 0, \quad \mathcal{R} = \begin{bmatrix} R & 0 & \cdots & 0 & 0 \\ 0 & R & \cdots & 0 & 0 \\ \cdots & \cdots & \ddots & \cdots & \cdots \\ 0 & 0 & \cdots & R & 0 \\ 0 & 0 & \cdots & 0 & R \end{bmatrix} > 0$$

to write MATRIX..

Define

cost funct as:

(respect our matrix definition) we can write cost funct as follows...

$$\bar{J}(x(k), U(k), k) = X'(k)\mathcal{Q}X(k) + U'(k)\mathcal{R}U(k)$$

compact form definition

different way to express

Note that

Neglect term NOT influenced by optimiz. u

$$J(x(k), U(k), k) = \bar{J}(x(k), U(k), k) + \cancel{x'(k)Qx(k)}$$

const term @ k, doesn't modify by our choice! NOT influence, we can remove it

term independent on ***u(k+i), i≥0***

⇒ Therefore at the end

NOT considering first term, NOT import. to us

$$arg(\min_{u(\cdot)} J(x(k),u(\cdot),k)) = arg(\min_{U(k)} \bar{J}(x(k),U(k),k))$$

simplified J

and we can minimize with respect to ***U(k)***

$X(k) = \mathcal{A}x + \mathcal{B}U$

>0 nucleus

$$\bar{J}(x(k),U(k),k) = (\mathcal{A}x(k) + \mathcal{B}U(k))' \mathcal{Q} (\mathcal{A}x(k) + \mathcal{B}U(k)) + U'(k)\mathcal{R}U(k)$$

$$= x'(k)\mathcal{A}'\mathcal{Q}\mathcal{A}x(k) + 2x'(k)\mathcal{A}'\mathcal{Q}\mathcal{B}U(k) + U'(k)(\mathcal{B}'\mathcal{Q}\mathcal{B} + \mathcal{R})U(k)$$

U(k) is the unknown

↳ quadratic expression in U(k)

*If there are no constraints, one can simply derive with respect to **U(k)** and set the derivative to zero (the quadratic function in **U(k)** is positive definite)*

simplest sol. without constr.

obtaining optimal value ⇒

$$\boxed{U^o(k) = -(\mathcal{B}'\mathcal{Q}\mathcal{B} + \mathcal{R})^{-1}\mathcal{B}'\mathcal{Q}\mathcal{A}x(k)}$$

depend on current state $x(k)$

our solution unconstrained

matrix, partitioned in $\mathcal{K}(i)$...

$$U^o(k) = -(\mathcal{B}'\mathcal{Q}\mathcal{B}+\mathcal{R})^{-1}\mathcal{B}'\mathcal{Q}\mathcal{A}x(k) = \begin{bmatrix} \mathcal{K}(0) \\ \mathcal{K}(1) \\ \vdots \\ \mathcal{K}(N-1) \end{bmatrix} x(k), \quad \mathcal{K}(i) \in R^{m,n}$$

rows of overall "gain" matrix

$$u^o(k+i) = -\mathcal{K}(i)x(k), \quad i = 0, 1, \dots, N-1$$

same sol by R.E OR open loop sol

The numerical values of the optimal control sequence are the same obtained with the closed-loop strategy

different way to compute BUT same solution!

Closed-loop (Riccati equation)

$$u^o(k+i) = -K(i)x(k+i), \quad i = 0, 1, \dots, N-1$$

same time instant

"**Open-loop**"

$$u^o(k+i) = -\mathcal{K}(i)x(k), \quad i = 0, 1, \dots, N-1$$

Future control depends on current only!

formally different BUT must lead to same sol

Receding Horizon

$$u^{MPC}(k) = -K(0)x(k) = -\mathcal{K}(0)x(k)$$

must be equal @ 0

@ k we have same solution ($i = 0$)

once computed $K(0)$ someway (solved the problem..)

↓

can we guarantee STABILITY of closed loop syst ? about this gain (constant)

↓

NO

↳ as we can see by examples... ⟹

(problem) ⟹ not slimeness of stabilizing control action! ↴ stability depend on prediction horizon taken

(Notice.. By open loop sol we can include constraints, while not with Ricati eq)

If the state is not measurable?

A state observer must be used (pole placement, KF,…)

Is the receding horizon strategy stabilizing?

$x(h+1) = 3x(h) - u(h)$ (unstable)

$J = \sum_{i=0}^{N-1} \left(x^2(h+i) + 2u^2(h+i) \right)$ to minimize $\begin{cases} Q=1 \\ R=2 \end{cases}$

$\downarrow$ unconstrained! we can use ∞ horiz. approach solving steady state R.E

$\boxed{N \to \infty}$ $\rightsquigarrow$ LQ_∞ $\boxed{\text{solution}}$ $(A=3, B=-1, Q=1, R=2, S=0)$

$$\bar{P} = 9\bar{P} + 1 - \frac{9\bar{P}^2}{\bar{P}+2} \longrightarrow \bar{P} = 17.12,\ \bar{K} = -2.69,\ A-B\bar{K} = 0.31$$

(OK, inside stability)

$\boxed{\text{RH with different horizons}}$ different approach...

no constraint

Using receding horizon approach with short prediction horizon, don't stabilize!

$P_N = 0 \rightarrow K_{N-1} = 0$, $A - BK_{N-1} = 3$ unstable!

one iteration...

$P_{N-1} = 1 \rightarrow K_{N-2} = -1$, $A - BK_{N-2} = 2$ unstable!

(N = 3)

$P_{N-2} = 7 \rightarrow K_{N-3} = -\frac{21}{9}$, $A - BK_{N-3} = \frac{2}{3} < 1$ $\boxed{\text{STABLE}}$

We N = 3 we stabilize! We need large N!

receding horiz good for time inv. control law

The minimum prediction horizon required to have closed-loop stability is N = 3

The issue of the stabilizing properties of MPC will be discussed later on ← conditions about stability

↳ BUT don't guarantee stability → industrially works well anyway

intuitively: optimal sol iterating R.E equal to prediction horiz.

$P \rightsquigarrow P_\infty$ iterating, stabilizing LQ_∞ solution, taking large prediction horizon means steady state LQ_∞ control in industrial control syst

How to consider constraints?

(actuators saturate, force state / outputs inside range (like a manipulator which moove between obstacles, so constrained motion)

consider it a-posteriori by ANTI WINDUP is suitable for control var. NOT for state and output !

limitations!

$$
\begin{cases}
u_m \le u(k+i) \le u_M, & i = 0, \dots, N-1 \\
x_m \le x(k+i) \le x_M, & i = 1, \dots, N \\
y_m \le y(k+i) \le y_M, & i = 1, \dots, N
\end{cases}
$$

(elementwise)

Define Bounds limit in Matrix form!

$$
U_m = \begin{bmatrix} u_m \\ u_m \\ \vdots \\ u_m \end{bmatrix}, \quad U_M = \begin{bmatrix} u_M \\ u_M \\ \vdots \\ u_M \end{bmatrix}, \quad X_m = \begin{bmatrix} x_m \\ x_m \\ \vdots \\ x_m \end{bmatrix}, \quad X_M = \begin{bmatrix} x_M \\ x_M \\ \vdots \\ x_M \end{bmatrix}
$$

$$
Y_m = \begin{bmatrix} y_m \\ y_m \\ \vdots \\ y_m \end{bmatrix}, \quad Y_M = \begin{bmatrix} y_M \\ y_M \\ \vdots \\ y_M \end{bmatrix}, \quad Y(k) = \begin{bmatrix} y(k+1) \\ y(k+2) \\ \vdots \\ y(k+N-1) \\ y(k+N) \end{bmatrix} \implies
$$

↓ reformulate the problem {forget about control prob ↳ transform it into optimiz.}

Constrained optimization problem

QP problem definition: ↳

Quadratic cost funct

$$\min_{U(k)} \bar{J}(x(k), U(k), k) = (\mathcal{A}x(k) + \mathcal{B}U(k))' \mathcal{Q} (\mathcal{A}x(k) + \mathcal{B}U(k)) + U'(k)\mathcal{R}U(k)$$

$$\left.\begin{aligned} X_m &\leq X(k) = \mathcal{A}x(k) + \mathcal{B}U(k) \leq X_M \\ U_m &\leq U(k) \leq U_M \\ Y_m &\leq Y(k) \leq Y_M \end{aligned}\right\} \text{constr. on state, in, out}$$

⟹ extendable approach to NONLIN syst and NON Quadratic cost funct! → extension

This problem **does not have** an **explicit solution**, which can be computed with suitable **optimization algorithms**.

(in most cases, NOT explicit sol.!) → standard optimiz. problem! quadratic programming + linear constr. problem

- In this case however, the problem is a ***Quadratic Programming* (*QP*) one**, which can be solved with very efficient algorithms

↑ {run an optimiz alg. providing you numerical solution} → reoptimize ∀k that optimiz problem, so in the past require many time and was usable for slow systems!

Nowday fast solution Alg.

Excursus: Quadratic Programming problems in Matlab

Quadratic term   linear term

cost funct: $min_{\varphi} 0.5\varphi' H\varphi + f'\varphi$ on optimiz. variable φ

constraints
$$
\begin{cases}
A\varphi & \leq & b \\
A_{eq}\varphi & = & b_{eq} \\
low_b & \leq & \varphi \\
\varphi & \leq & up_b
\end{cases}
$$

Matlab sw solution

$$\varphi = quadprog(H, f, A, b, A_{eq}, b_{eq}, low_b, up_b)$$

{assign all parameters, give you overall solution}

(in our case)

$$\min_{U(k)} \bar{J}(x(k), U(k), k) = \cancel{x'(k)\mathcal{A}'\mathcal{Q}\mathcal{A}x(k)} + \underbrace{2x'(k)\mathcal{A}'\mathcal{Q}\mathcal{B}}_{f'}\underbrace{U(k)}_{\varphi} + U'(k)\underbrace{(\mathcal{B}'\mathcal{Q}\mathcal{B}+\mathcal{R})}_{0.5H}U(k))$$

don't depend on U(k)

to solve on matlab

Try by yourself to write the state, input, and output constraints in the form required by *quadprog*

in matlab good optimiz. toolbox → many optimiz. toolbox / sw environments nowdays

(many fields of usage of optimization)

another advantage of MPC respect LQ control

Tracking of reference signals and disturbances

(time varying)

over the prediction horizon... $y(k) \simeq y^o(k)$

must well follow y^o KNOWN signal!

(follow a given path!)

System

$$
\begin{cases}
x(k+1) = Ax(k) + Bu(k) + Md(k) \\
y(k) = Cx(k) + d(k)
\end{cases}
$$

$\Downarrow$ IF y^o is output of a system (step/ramps ..) → you can define an explicit sol → while if general y^o, LQ is problematic!

Cost function

easy to reformulate in MPC for general y^o

penalize $(y^o(k+i) - y(k+i))^T Q (y^o(k+i) - y(k+i))$

$$
J(x(k), u(\cdot), k) = \sum_{i=0}^{N-1} \left(\|y^o(k+i) - y(k+i)\|_Q^2 + \|u(k+i)\|_R^2 \right) + \|y^o(k+N) - y(k+N)\|_S^2
$$

Include this quadratic term

also @ end of prediction horizon

$\left\{ Q \in \mathbb{R}^{p \times p} \text{ matrix of weight over tracking error} \right\}$

⇒ problem "easy" to consider → iterating the model equation to compute future $y(k)$ values!

The future evolution of the output along the prediction horizon is:

predicting elements

$$Y(k) = \mathcal{A}_c x(k) + \mathcal{B}_c U(k) + \mathcal{M}_c D(k)$$

(function of current state + future input + future disturbance)

where

disturbance acting on state eq.

$$\mathcal{A}_c = \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^{N-1} \\ CA^N \end{bmatrix}, \quad D(k) = \begin{bmatrix} d(k) \\ d(k+1) \\ \vdots \\ d(k+N-2) \\ d(k+N-1) \\ d(k+N) \end{bmatrix} \quad \mathcal{M}_c = \begin{bmatrix} CM & I & 0 & \cdots & 0 & 0 & 0 \\ CAM & CM & I & \cdots & 0 & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ CA^{N-2}M & CA^{N-3}M & CA^{N-4}M & \cdots & CM & I & 0 \\ CA^{N-1}M & CA^{N-2}M & CA^{N-3}M & \cdots & CAM & CM & I \end{bmatrix}$$

$$\mathcal{B}_c = \begin{bmatrix} CB & 0 & 0 & \cdots & 0 & 0 \\ CAB & CB & 0 & \cdots & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ CA^{N-2}B & CA^{N-3}B & CA^{N-4}B & \cdots & CB & 0 \\ CA^{N-1}B & CA^{N-2}B & CA^{N-3}B & \cdots & CAB & CB \end{bmatrix}$$

same consideration of before, iterate x, y!

Letting

$$Y^o(k) = \begin{bmatrix} y^o(k+1) \\ y^o(k+2) \\ \vdots \\ y^o(k+N-1) \\ y^o(k+N) \end{bmatrix}$$ Ref signals

free to choose future traj!

by properly defining the matrices $\mathcal{Q}$ e $\mathcal{R}$ one obtains :

here Q, R matrix defined properly dimension

$$arg(\min_{U(k)} J(x(k), u(\cdot), k)) = arg(\min_{U(k)} \bar{J}(x(k), U(k), k))$$

and the cost function to be minimized becomes

(where removing x(k), y°(k) of current instant, that cannot be changed)

$$\bar{J}(x(k), U(k), k) = (Y^o(k) - Y(k))' \mathcal{Q} (Y^o(k) - Y(k)) + U'(k)\mathcal{R}U(k)$$

possibly ***under state, control, and output constraints***

Note the cost function (and the solution of the optimization problem)

IF we know y^o, we weight future error, having anticipative action

$$\bar{J}(x(k), U(k), k) = (Y^o(k) - Y(k))' \mathcal{Q} (Y^o(k) - Y(k)) + U'(k)\mathcal{R}U(k)$$

to optimize respect U(k)

BUT, there are ISSUES

- depend on future reference signals and disturbances

sometimes ref signal know in advance, other times not!

$$Y(k) = \mathcal{A}_c x(k) + \mathcal{B}_c U(k) + \mathcal{M}_c D(k)$$

Knowing in advance @k the evolution of $y^o(k)$, the MPC anticipate the instruction in advance respect y^o change!

sometimes you know how disturbances will change in the future, so we act to compensate anticipating

This allows one to consider the problem of time varying reference signals

If these quantities are known in advance, MPC produces an anticipative response

If *d* and y^o are unknown in advance, it is customary to set

set as constant current value

$$y^o(k+i) = y^o(k) \quad , \quad d(k+i) = d(k)$$

to properly solve the problem. simplest approach consideration about optimization problem!

Soft and hard constraints

An optimization problem is solved on-line. It must be guaranteed that a solution is always computed by the optimization algorithm

at any time sol. should exist → you have to look to the constraints ⇒

3 kinds of constraints possible...

$$
\begin{cases}
U_m \leq U(k) \leq U_M \\
X_m \leq X(k) \leq X_M \\
Y_m \leq Y(k) \leq Y_M
\end{cases}
$$

These constraints can always be satisfied because the control variables are our design choices (the results of the optimization procedure). They are implemented as hard constraints

always can be satisfied, you decide the input!
↳ able to satisfy in any case!

(real issue!) you have unknown disturbances that can influence your system → you must guarantee that in any case, EVEN IF unable to satisfy: your alg. must give a solution

@ any k must provide sol. IF you have this state, output constr, the solution cannot exist sometimes

These constraints can not (temporarilly or permanently) be satisfied, due for example to the effects of disturbances. They are implemented as soft constraints by means of suitable slack variables for Example IF controlling room temp.

⇓

having a
lower bound : $T > T_{min}$ IF disturbance happen, and
low external temp,

then recover ← output $< T_{min}$!

unable to solve
this optimiz ←
temporary

x, y constr
sometimes
Impossible to satisfy

↳ so in some cases we need to RELAX
this constraints by slack variables !

↑

essential optimiz tech ⇒

IF too much weight on ε ! optimiz tends to set ε≃0, so you are on original problem

← [ρ: weight taken sufficiently large!]

MPC with soft constraints

(reformulating problem)

instead if unable to solve the problem.. you set an ε>0 enlarging constraint for feasible sol.

{ still NOT guarantee asymp. follow of const. reference } ←

$$\min_{U(k),\varepsilon} \; (Y^o(k)-Y(k))'\,Q\,(Y^o(k)-Y(k))+U'(k)\mathcal{R}U(k)+\rho\varepsilon$$

[to be weighted inside the cost function!]

⇓

[MPC with integral action ?!]

⇒ NEXT.. ZERO error reg

subject to

I use here one ε for all, but you can weight with different ε_i on constr! →

$$\begin{cases} U_m \le U(k) \le U_M \\ Y_m - \varepsilon I \le Y(k) \le Y_M + \varepsilon I \\ \varepsilon \ge 0 \end{cases}$$

output var constraint

you "enlarge" the range by ε (slack var.)

where $I = \left[\begin{array}{cccc} 1 & 1 & \ldots & 1 \end{array}\right]'$ and ρ must be selected sufficiently high to guarantee that the optimum is $\varepsilon = 0$ when a feasible solution exists

for states/outputs suitable implement soft constr.

Constant reference signals and integral action { reformulate MPC to deal with the steady state & error regulation }

for long K const.

to solve integrator request

Assume that $y^o(k) = y^o$ and $d = 0$ (for simplicity) . If $\boxed{p \le m}$ and the system does not have transmission zeros in 1, it is possible to compute the equilibrium $(\bar{x}, \bar{u})$ such that

$$
\begin{cases} \bar{x} = A\bar{x} + B\bar{u} \\ y^o = C\bar{x} \end{cases}
$$

define by $\bar{x}, \bar{u}$ asymp values such $y = y^o$

once computed $\bar{x}, \bar{u}$

and the cost function to be minimized can be written as (plus additional weight on the states)

not real integral action!

with small errors you reach bad equilibrium

cost funct will be used for stability analysis →

$$
J(x(k), u(\cdot), k) = \sum_{i=0}^{N-1} \left(\underbrace{\|y^o - y(k+i)\|_Q^2}_{0 \text{ when } y = y^o} + \underbrace{\|u(k+i) - \bar{u}\|_R^2}_{@\text{ steady state all } 0} \right) + \|y^o - y(k+N)\|_S^2
$$

(steady state)

you have to weight $u - \bar{u}$ ⇒ must reach the value $\bar{u}$: weight the variation respect asym stable value

Pros: $J = 0$ for $y(k+i) = y^o$ and $u(k+i) = \bar{u}$

Cons: in case of unknown disturbances and/or modeling errors the computation of the steady state is not correct, and steady state errors occur

↳ good only if correct $\bar{x}, \bar{u}$, in case of disturb/model errors.. you get steady state errors! → how to deal with this problem?!

↳ 2 ways:

How to include the equivalence of integral action in problem formulation

Integral action in MP – method 1 : DISTURBANCE ESTIMATION

using an old trick... Basic idea

include this equivalent disturbance enlarging this system
(external dist. + modelling err)

Assume that the system is affected by an *unknown*, *constant* disturbance

define equivalent disturbance affecting both output + state : (idea)

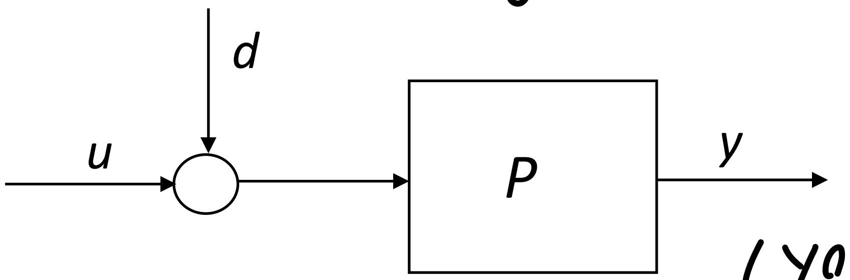
$$\begin{cases} x(k+1) = Ax(k) + Bu(k) + Md(k) \\ d(k+1) = d(k) \quad \text{constant disturb} \\ y(k) = Cx(k) + Fd(k) \end{cases} \longleftrightarrow \begin{cases} \begin{bmatrix} x(k+1) \\ d(k+1) \end{bmatrix} = \begin{bmatrix} A & M \\ 0 & I \end{bmatrix} \begin{bmatrix} x(k) \\ d(k) \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u(k) \\ y(k) = \begin{bmatrix} C & F \end{bmatrix} \begin{bmatrix} x(k) \\ d(k) \end{bmatrix} \end{cases}$$

rearrange the enlarged syst.

integral dynamic of the disturbance

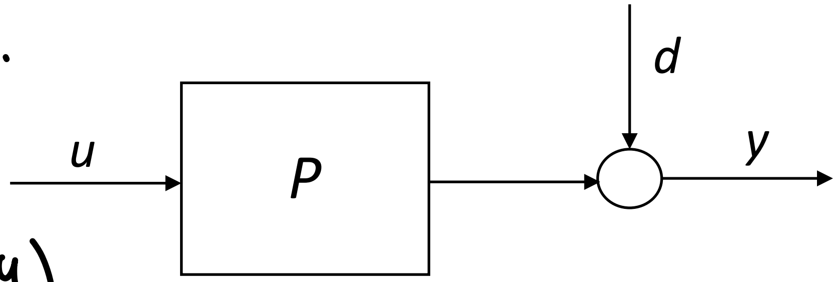
⇓

Possible choices {other disturbance modelling choice! } only state var. affected by disturbance... | or disturbance mainly acting on OUTPUT...



M=B, F=0

(you have to specify M, F value!)



M=0, F=I

For the enlarged system

considering disturbance

$$\begin{cases} \begin{bmatrix} x(k+1) \\ d(k+1) \end{bmatrix} = \begin{bmatrix} A & M \\ 0 & I \end{bmatrix} \begin{bmatrix} x(k) \\ d(k) \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u(k) \\ y(k) = \begin{bmatrix} C & F \end{bmatrix} \begin{bmatrix} x(k) \\ d(k) \end{bmatrix} \end{cases}$$

you can use it to compute steady state value

you need to estimate the disturbance... with new matrix $\bar{A}, \bar{C}$

The following result holds (proof by yourselves)

must be OBS pair

to estimate d, $(\bar{A}, \bar{C})$ OBS

↳ valid if (A, C) OBS!

The pair

$$\bar{A} = \begin{bmatrix} A & M \\ 0 & I \end{bmatrix} \quad , \quad \bar{C} = \begin{bmatrix} C & F \end{bmatrix}$$

with result..

is observable if and only if the pair (A, C) is observable and

(properties:)

$$\underbrace{rank \begin{bmatrix} A-I & M \\ C & F \end{bmatrix}}_{\text{+ you don't have invariant zeros...}} = n + r \longleftrightarrow r \le p, d \in R^r$$

If previous assumptions satisfied

use the theory of OBSERVER to estimate the state (extended)

Under the previous conditions, it is possible to build an ovbserver to estimate, at any time instant $\begin{bmatrix} \hat{x}(k) \\ \hat{d}(k) \end{bmatrix}$

If known, we use reduced ORD estimator

once estimated the state $(\hat{x}, \hat{d})$

(assume constant d @ estim. value $\hat{d}$)

Then, assuming that the future disturbance is constant, $(d(k+i) = \hat{d}(k),\ i \geq 0)$, one can compute the equilibrium pair $(\bar{x}, \bar{u})$ such that

repeating for enlarged state the steady state values such that ($y = y^o$)

$$\begin{bmatrix} I-A & -B \\ C & 0 \end{bmatrix} \begin{bmatrix} \bar{x} \\ \bar{u} \end{bmatrix} = \begin{bmatrix} M & 0 \\ -F & I \end{bmatrix} \begin{bmatrix} \hat{d}(k) \\ y^o \end{bmatrix}$$

use fictitious d and redefine steady state using estimated $\hat{d}$

so you get $\bar{u}$ on previous formulation...

and use these values in the cost function (possibly with a weight on the states)

in this way you are including integrator on the system! (Proof)

$$J(x(k), u(\cdot), k) = \sum_{i=0}^{N-1} \left(\|y^o - y(k+i)\|_Q^2 + \|u(k+i) - \bar{u}\|_R^2 \right) + \|y^o - y(k+N)\|_S^2$$

Interesting/popular method → extendible for NONLIN syst.

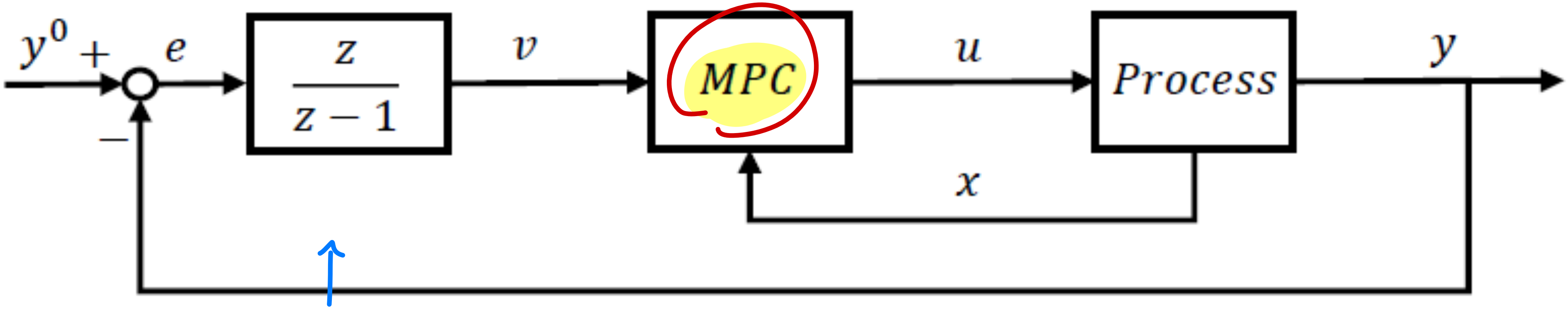
(less popular)

Integral action in MP – method 2 : EFFECTIVE INTEGRAL ACTION

The idea is to resort to the scheme

Include it for real on the system scheme..

HOW TO DESIGN IT?



IF overall syst asymp. stable, all signals tend to an asymp. stable value

analogous to the schemes previously studied

[e = const only IF $e = 0$ OR $\to \infty$ integrating!]

0 steady state error guarantee by $\frac{z}{z-1}$

{we must compute MPC made by process and integrator enlarged sys} in discrete time.

system equation

Plant + integrators

$$\begin{cases} x(k+1) &= Ax(k)+Bu(k) \\ v(k+1) &= v(k)+e(k+1) \\ e(k)=y^o-y(k) \end{cases}$$

state eq

Integrator eq. TF: $\frac{z}{z-1}$

error definition

we can rewrite the II eq.

$e(k+1)$

$$\begin{cases} x(k+1) &= Ax(k)+Bu(k) \\ v(k+1) &= v(k)+y^o-Cx(k+1)=v(k)+y^o-CAx(k)-CBu(k) \end{cases}$$

$\Downarrow$

define $\begin{cases} \delta x(k)=x(k)-x(k-1) \\ \delta u(k)=u(k)-u(k-1) \end{cases}$

difference variable at different sampling time

subtract the same equations at *k-1*

recall that $v(k+1)-v(k)=e(k+1)$

$$\begin{cases} x(k+1)-x(k)=\delta x(k+1) \\ v(k+1)-v(k)=\delta v(k+1)=e(k+1) \end{cases}$$

$$\begin{bmatrix} \delta x(k+1) \\ e(k+1) \end{bmatrix} = \begin{bmatrix} A & 0 \\ -CA & I \end{bmatrix} \begin{bmatrix} \delta x(k) \\ e(k) \end{bmatrix} + \begin{bmatrix} B \\ -CB \end{bmatrix} \delta u(k)$$

on $\delta x, e, \delta u$ variables..

system in $\boxed{\text{velocity form}}$

(enlarged syst in δ variables)

$$\begin{bmatrix} \delta x(k+1) \\ e(k+1) \end{bmatrix} = \begin{bmatrix} A & 0 \\ -CA & I \end{bmatrix} \begin{bmatrix} \delta x(k) \\ e(k) \end{bmatrix} + \begin{bmatrix} B \\ -CB \end{bmatrix} \delta u(k)$$

reachability condition necessary

reachable if (A,B) reachable and the system under control does not have transmission zeros at z=1

↓ reformulating MPC for this new system..
(we have redefined process + integrator in velocity form)

An interesting cost function

sequence of future δu..

new variable

$$\min_{\delta u(k+i),i=0,\dots,N-1} J(\delta x(k), \delta u(\cdot), k) = \underbrace{\sum_{i=0}^{N-1}}_{\text{sum over predict. horiz.}} \left(\|e(k+i)\|_Q^2 + \|\underbrace{\delta u(k+i)}_{\text{increment}}\|_R^2 \right) + \overbrace{\|e(k+N)\|_S^2}^{\text{terminal cost}}$$

interesting cost function:

J=0 for ***e=0***, ***δu=0***, no computation of the steady state values of states and inputs is required

$u(k) = u(k-1) + \delta u(k)$! like having integrator ⇒ $\delta u = 0$:⇒ $u(k) = u(k-1)$ steady state

weight and constraints on ***δu*** have physical meaning, and are related to the «speed of variation» of the inputs (think to a valve). In addition, they can be used to speed up or slow down the control action by means of a proper selection of the bounds on ***δu***

(using only first δu value:.)

Nce the optimal sequence of future control variations $\delta u^o(k+i),\ i=0,\dots,N-1$ is computed, one sets

(method 1) we enlarge syst with fictitious d(k).. recompute the steady state values... while with this (method 2) → You don't need to compute $\bar{u}, \bar{x}$

→ not needed!

$$u(k) = u(k-1) + \delta u^o(k)$$

Notice: you can always include into your problem constraints for other approaches

↳ on control increment

$\| \Delta u(k) = u(k) - u(k-1)$ limitation! $\|$

↳ constraints not only on $u_{MIN} \div u_{MAX}$

but also on velocity variation of $u(k)$

↓

limiting speed of variation between 2 sequent instant... { typical of actuators that usually has variation constraints }

limit control variation → DON'T allow too fast changes!

1st method → can be extended for non lin. syst, widely used

2nd method → cannot be extended for non lin syst!

the extension and subtract previous value does NOT work! → the velocity form cannot be found if N.L system!

$$x(k+1) = f(x(k), u(k))$$

$$x(k+1) - x(k) = f(x(k), u(k)) - f(x(k-1), u(k-1)) \quad ?!$$

NOT in $\delta x, \delta u$

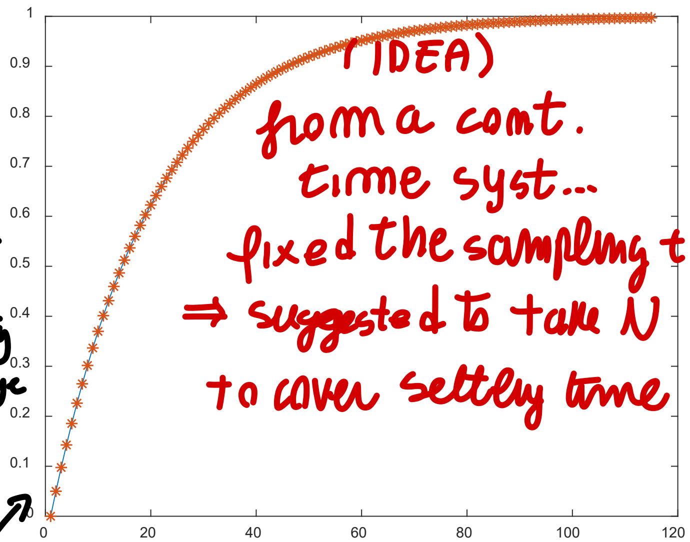
[still done using linear models] ← { BUT IN many cases anyway we use linear model } practical consideration to be done to implement MPC control

- **Choice of the prediction horizon N** ?

IF sufficent large $N \to$ in unconstrained case is equal to solve more R.E ! taking N long $\to \infty$ horiz. optimiz. $\downarrow$ (stability advantage

In general, it is suggested to set N to cover the settling time of the process (in case of asymptotically stable systems) . It obviously depends on the adopted sampling period

assuming asymp. stable syst.



N

(# points) depend on sampling time

Empirical motivation: in the case of unconstrained systems (closed-loop solution), the Riccati equation converges and the RH-MPC control law is stabilizing

{consider prediction horizon T and divide by time intervals t_k to get N} $\to$ empirically considering settling time you take large N so you solve approx ∞ horiz LQ control

Problem: **N** is also the number of control inputs (multiplied by m) to be computed as the solution of the optimization problem, which can become very large and time consuming

$\downarrow$ you reach ∞ horiz solution appro

(DRAWBACK) $\uparrow$ you can have $N \simeq 100$! and you have to find the u variables unknown $u(k), u(k+1)$.. each of m dimension overall optimize respect $N \cdot m$ var $\to$ large optimiz. problem $\to$ demanding computation how to deal with computational drawback (n·m variables!)

how to select sampling period ?
t_k choosen by: digital signal theory

Choice of the sampling period

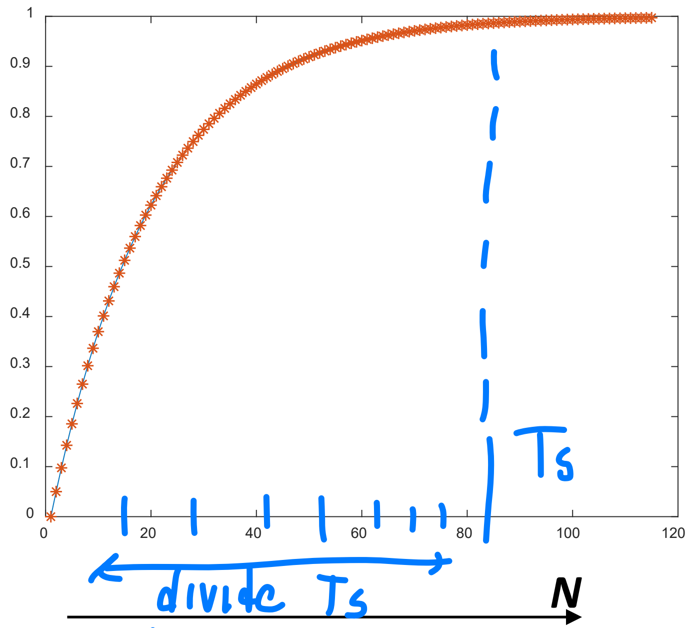
It should be based on the Shannon theorem and on the (required) crossover bandwith in closed-loop

practical rule

(• IF short sampling time → N bigger ! ⇒ (trade-off!)
• IF large sampling → issue on shannonth

Rule of thumb

Define by *Ts* the settling time at 99% of the open loop step response



in 15÷50 part to get each ΔT (not too fast)

Select the sampling time *ΔT* so as to have 15-50 samples in the settling time ← in industrial applications

with this rule of thumb

For first order systems, this roughly corresponds to consider a Nyquist frequency guaranteeing an attenuation of 20-30dB with respect to the frequency of the open-loop time constant

?

↳ from I ORD syst. you use a Nyquist freq with 20÷30dB attenuation on Bode diag respect low freq

3rd problem... IF N big ... ⇒ large amount of var to solve optimiz problem

industrial trick:
(to reduce order number, simpler optimiz. problem)

Control horizon N_u

Only the first N_u control variables are optimized, with

$$0<N_u<N$$

then it is assumed that the control is constant over the remaining part of the prediction horizon, i.e.

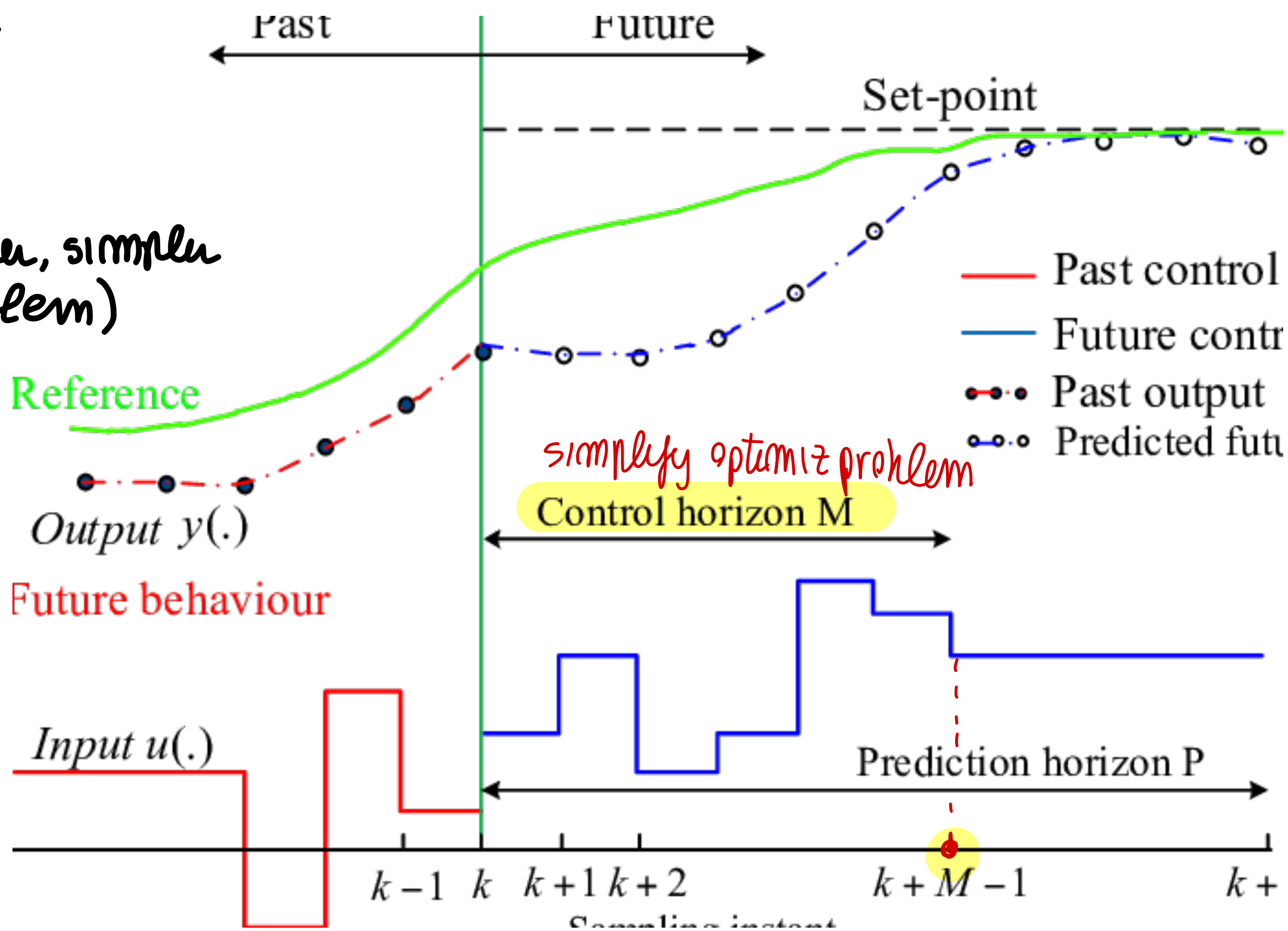
maintain constant after Nu

$$u(k+i)=u(k+i-1), \quad i=N_u,\ldots,N-1$$

or

$$\delta u(k+i)=0, \quad i=N_u,\ldots,N-1$$

no changes in u after Nu



When N is too big → better to reduce to Nu variation of the input allowed!

Here $M = N_u$

Basic idea

taking (N) you have to optimize respect a certain overall sequence
we can have $u(k+i+..) \simeq$ constant

↓

In practice @ K you solve optimiz,
BUT you don't optimize respect all K
while const from a given point

so you reduce the number
of optimiz variable $M < N$
$(= N_u)$

⇓ **PRO:**

- reduce amount of control var used
- taking small N_u you reduce responce speed..

Cost function with control horizon N_u

Using a control horizon different from the prediction horizon implies to slightly modify the cost function (here we use the incremental one for systems in velocity form)

modify J, (reduce control var)

consider control

penalyze all

$$J(x(k),u(\cdot),k) = \sum_{i=0}^{N-1} \|y^o(k+i)-y(k+i)\|_Q^2 + \sum_{i=0}^{N_u-1} \|\delta u(k+i)\|_R^2 + \|y^o(k+N)-y(k+N)\|_S^2$$

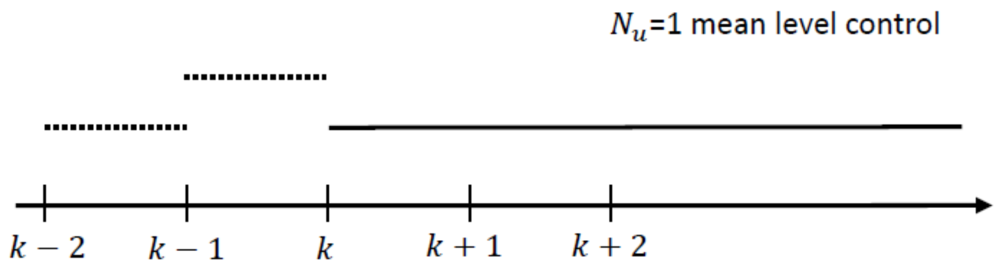
future error

variables until (k+Nu-1)

A particular case, usually named ***Mean Level Control***, is the one with ***N_u=1.*** Only one control move is allowed, but at every *k* the control value is changed due to the RH principle, see the figure

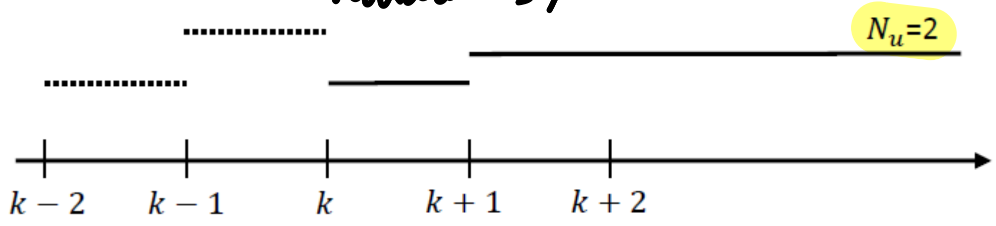
(you reduce the control horiz obtaining slower syst → modify speed response weighting control var, imposing δu limit δu max)

In general, short control horizons slow down the controlled system



(a)

(allow two variations)



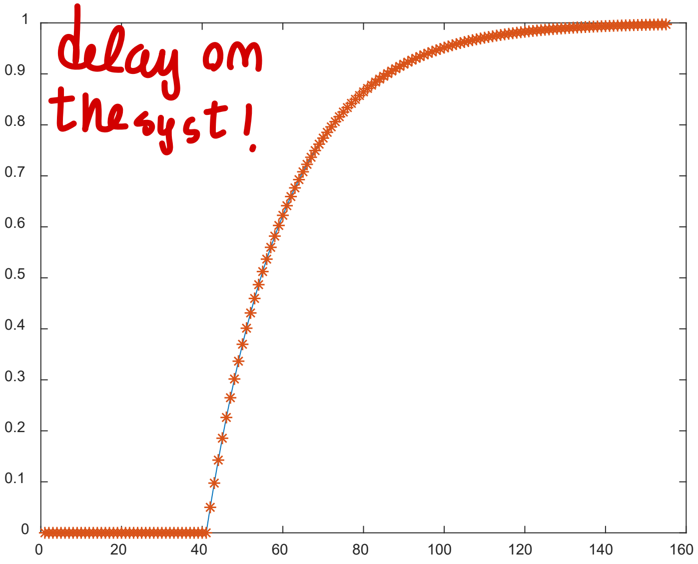
another industrial tricks

Minimum prediction horizon

For nonminimum phase systems, with **delay** or **inverse response** due to unstable zeros, it is convenient not to penalize the future values of the output (or of the error), which would force the system to move in a wrong direction (the standard problem of non minimum phase systems which require a slow control action) ⇒ usefull trick! → issue of initial strange behaviour on N_1..

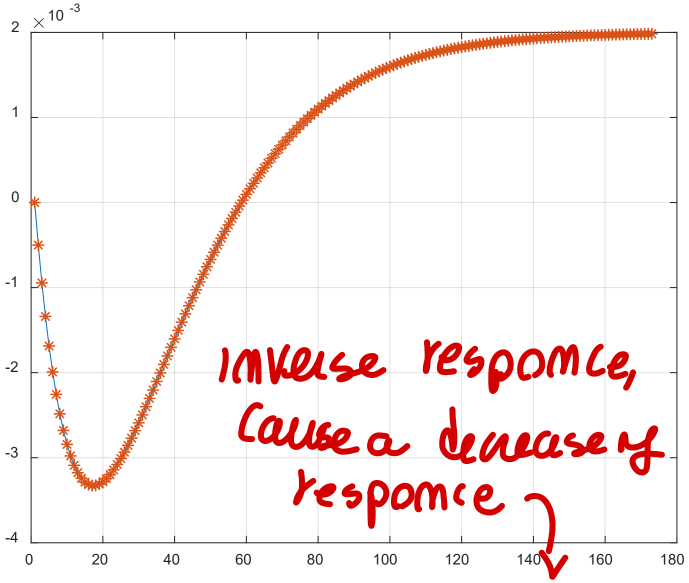
NOT include on cost funct u, x relative to delay / inverse N_1

IF $N_u < N_1$... Nothing WORKS! Destabilize system



delay on the syst!

N_1 before effect of step variation



inverse responce, cause a decrease of responce due to unstable zeros

N_1

using limited horizon

$$J(x(k),u(\cdot),k)=\sum_{i=N_1}^{N-1}\|y^o(k+i)-y(k+i)\|_Q^2+\sum_{i=0}^{N_u-1}\|\delta u(k+i)\|_R^2+\|y^o(k+N)-y(k+N)\|_S^2$$

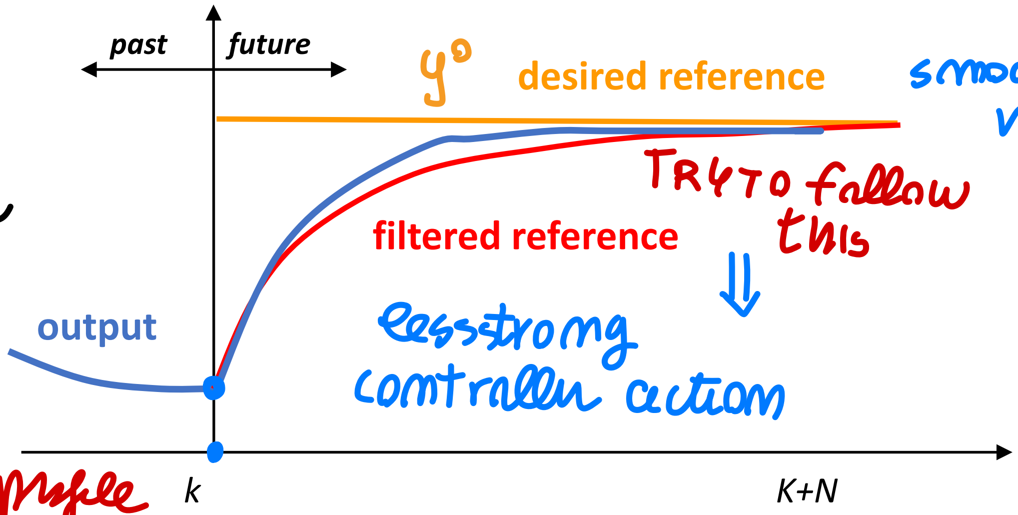
we start from the time delay

Filtering the reference signal

looking the standard implementation

To reduce the control effort, step variations of the reference signal are avoided, and a filtered reference is computed at any time starting from the current output

after transient @k we wanna reach a certain reference → BUT the difference will be large ⇒ STRONG controller action! to smooth, we consider fictitious traj which reach reference with smooth profile



smoothing control response

IDCOM - Adersa

company that implement MPC industrially

A possible signal generator is

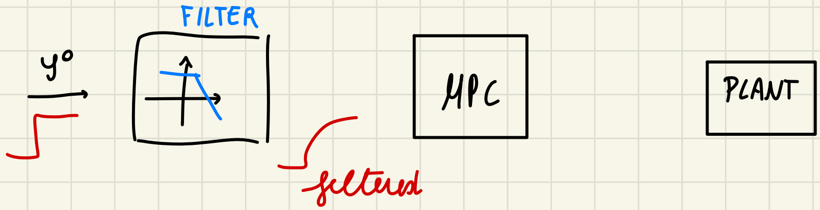
@any k you compute future ref in this way

widely used BUT hard to design!

$$y^o(k+i) = \left(1-\alpha^i\right)\bar{y}^o + \alpha^i y(k) \quad , 0 \leq \alpha < 1$$

steady state Ref

Note that this causes an additional (external) loop which should be considered in the stability analysis on the implementation

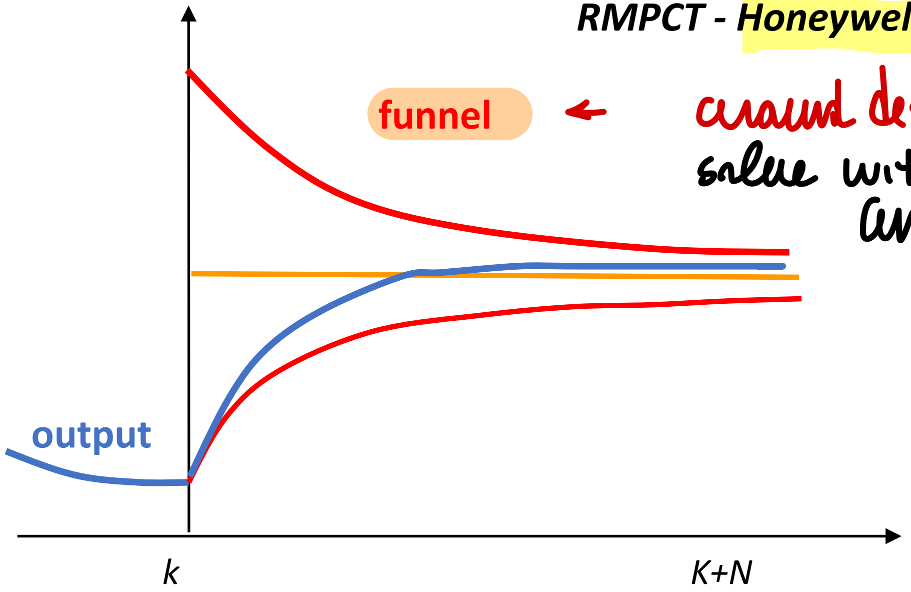


you close an external loop on the system itself! hard to analyze stability

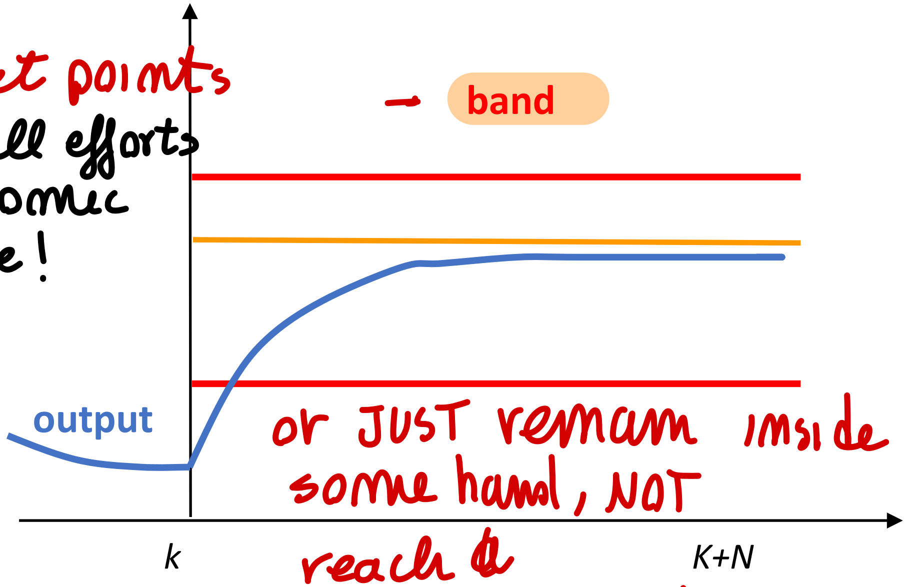
↳ another approach introduced by Honeywell ⇒

to reach the reference NOT immediately, I allow to reach it forcing the output in a given funnel → smooth action

RMPCT - Honeywell



around desired set points value with small efforts and economic save!



or JUST remain inside some band, NOT reach a specific requested y^o

You ask to stay inside prescribed limit

In many cases it is not mandatory to reach a steady state value, but to remain into prescribed limits. Then, instead of penalizing the error, it is possible to include in the optimization problem some hard (or soft) constraints. Recall that hard constraints on the output are difficult to be satisfied in view of the possible presence of disturbances.

hard to satisfy... due to disturbance!
you have to consider slack variables!