Toward model-free predictive control Possible connections with AI

Michel FLIESS

with

Cédric JOIN, Emmanuel DELALEAU

Workshop on data-driven control and analysis of dynamical systems

September 30th - October 1st, 2025 -- ENSEEIHT, Toulouse, France

Model-Predictive Control (MPC)

Jacques RICHALET: Model predictive control (MPC) (*Automatica*, 1978)
Interesting reference: *La commande prédictive a gagné la partie*
(Mesures, avril 2005)

MPC: Popular control schemes where **MODELS** are used for predicting the future behavior of the system over a *short* time window, the *horizon*.

Key rôles of various **OPTIMAL CONTROL** techniques: LQR, Dynamic Programming, Pontryagin's maximum principle, ...

Reset after a short time lapse $\neq$ traditional optimal control

Predictive Control without a Model

1. Models via Artificial Neural Networks (ANNs) $\Rightarrow$ Machine-Learning
2. Various theoretical frameworks (*data-driven*) – Popular in the Western academic world:
   J. Berberich, F. Allgöwer, *Annual Rev. Contr. Robot. Autonom. Syst*. 2025.
3. Via the *ultra-local* model associated to *model-free control*. Large number of publications, especially in China, often with concrete applications.
   A most recent example:
   G. Wu, P. Guan, F. Huang, Z. Long and J. He, Robust Speed-Stator Flux Tolerant Predictive Control for PMSM Drives With Parameters Disturbance Faults Diagnosis, IEEE Sensors J., 25, 34840-34849, **Sept.15, 2025**.

MFC *ultra-local model*

Ultra-local model, only valid during a short time lapse,

$$\dot{y} = F + \alpha u \tag{1}$$

for most input-output systems, under weak assumptions:

- α: constant parameter such that the 3 terms are of same magnitude $\Rightarrow \alpha$ does NOT need to be precisely estimated
- F: the whole information on the unknown system (including the external **perturbations**)
- **Data-driven** estimate:

$$F_{\text{est}}(t) = -\frac{6}{\tau^3} \int_{t-\tau}^{t} \left[(t - 2\sigma) y(\sigma) + \alpha \sigma (t - \sigma) \Delta u(\sigma) \right] d\sigma$$

where τ is small. In practice: digital filter.

MFC: *Intelligent* P controller

Intelligent P controller (iP):

$$u = -\frac{F_{\text{est}} - \dot{y}^* + K_P e}{\alpha}$$

- y^*: output reference trajectory
- $e = y - y^*$: tracking error
- K_P: tuning gain

$$\Downarrow$$

$$\dot{e} + K_P e = F - F_{\text{est}} \approx 0$$

$$\Downarrow$$

SIMPLE GAIN TUNING for local stability.

MFPC: Model-free predictive control

Short time lapse $\Rightarrow$ Eq. (1) becomes $\dot{y} = a + \alpha u$: elementary **flat** system, where $F \approx a$ approx. const., y **flat** output.
Lagrangian (cost function):

$$\mathcal{L} = (y - y_{\text{setpoint}})^2 + u^2 = (y - y_{\text{setpoint}})^2 + \left(\frac{\dot{y} - a}{\alpha}\right)^2$$

Criterion: $J = \int_{t_i}^{t_f} \mathcal{L} dt$
$\Rightarrow$Euler-Lagrange equation: $\frac{\partial \mathcal{L}}{\partial y} - \frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{y}} = 0$
$\Rightarrow$ non-homogeneous linear ODE (independent of a): $\ddot{y} - \alpha^2(y - y_{\text{setpoint}}) = 0$

Optimal solution: $y^\star(t) = y_{\text{setpoint}} + c_1 \exp(\alpha t) + c_2 \exp(-\alpha t)$
Boundary conditions: $y(t_i) = y_i$, $y(t_f) = y_{\text{setpoint}}$

$$\Downarrow$$

$$c_1 = \frac{y_i \exp(-\alpha t_f) - y_{\text{setpoint}} \exp(-\alpha t_f)}{\exp(\alpha t_i)\exp(-\alpha t_f) - \exp(-\alpha t_i)\exp(\alpha t_f)}$$

$$c_2 = -\frac{\exp(\alpha t_f)(y_i - y_{\text{setpoint}})}{\exp(\alpha t_i)\exp(-\alpha t_f) - \exp(-\alpha t_i)\exp(\alpha t_f)}$$

Subdivide the time interval $[t_0, t_N[$, $t_0 < \cdots < t_k < t_{k+1} < \cdots < t_N$. On each time interval $[t_k, t_{k+1}[$, repeat the above procedure.

Three examples

From the MPC literature

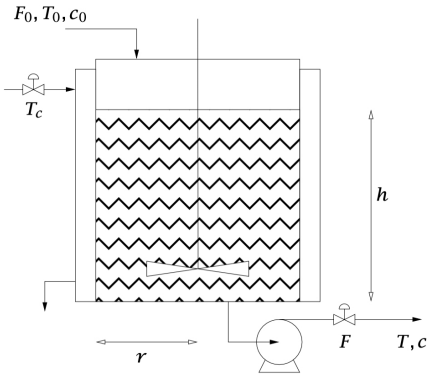
1. Chemical reactor (AIChE 2003 (chemical engineering)): linear approx. around operating point.
2. Two tanks (Europ. J. Contr. 2024): neural models (machine learning) & reinforcement learning.
3. Avoidance of unexpected obstacles (IFAC PapersOnLine 2021): neural models (machine learning) & reinforcement learning.

What is *reinforcement learning* (*RL*)?

Reinforcement Learning (RL) is an area of machine learning where algorithms learn to make decisions by trial and error, ultimately striving to maximize some cumulative reward. This technique draws inspiration from how humans and animals learn from feedback, adapting to complex, dynamic environments over time. Reinforcement Learning stands out in its capability to tackle problems where decision-making is sequential, and the consequences of actions unfold over time.

RL which requires ridiculously large numbers of trials to learn any new task
(Yann LeCun)

A chemical reactor

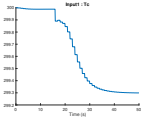


(1) Well-stirred reactor

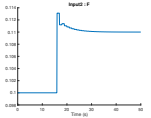
$$\begin{cases} \dot{c} = \frac{F_0(c_0 - c)}{\pi r^2 h} - k_0 \exp\left(-\frac{E}{RT}\right) c \\ \dot{T} = \frac{F_0(T_0 - T)}{\pi r^2 h} - \frac{\Delta H}{\rho C_p} k_0 \exp\left(-\frac{E}{RT}\right) c + \frac{2U}{r \rho C_p}(T_c - T) \\ \dot{h} = \frac{F_0 - F}{\pi r^2} \end{cases} \tag{2}$$

c (resp. h): fluid concentration (resp. height). Both measured. Control variables: F and T_c.

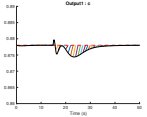
Chemical reactor: MFPC



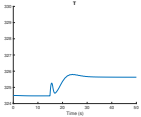
(2) T_c



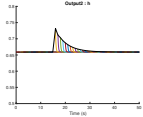
(3) F



(4) c (– black) and its set-point (- - red)

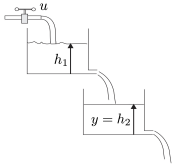


(5) T



(6) h (– black) and its set-point (- - red)

Two tanks

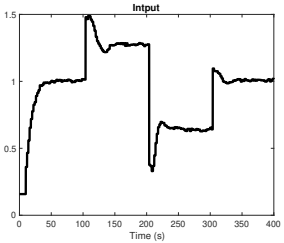


(7) Two tanks

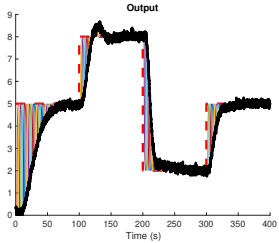
$$\begin{cases} s_1 \dot{h}_1 = u - k_1 \sqrt{h_1} \\ s_2 \dot{h}_2 = k_1 \sqrt{h_1} - k_2 \sqrt{h_2} \end{cases} \tag{3}$$

$u \geq 0$: control variable, h_ι, $\iota = 1, 2$, $0 \leq h_\iota \leq 10$: water level, k_ι and s_ι: constant parameters.

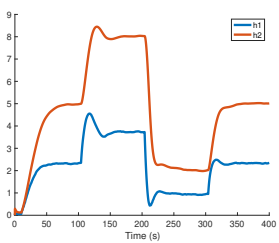
Two tanks: MFPC



(8) u (–)

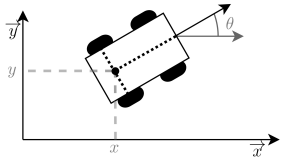


(9) Setpoint (- -) and y (–)



(10) Tank levels h_1 and h_2

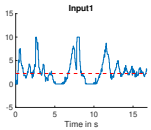
Autonomous robot: Dubins'car



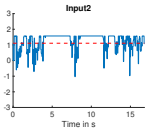
$$\begin{cases} \dot{x} = u_1 \cos(u_2) \\ \dot{y} = u_1(1+p)\sin(u_2) \end{cases} \tag{4}$$

- x, y: Cartesian coordinates of the middle of the rear axle;
- Control variables: u_1 (linear velocity), $\theta = u_2$ (angle with x axis);
- p ($-0.5 \leq p \leq +0.5$): uniformly distributed piecewise-constant perturbation.

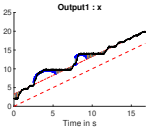
Autonomous robot: MFPC



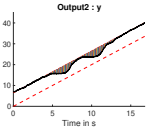
(11) u_1



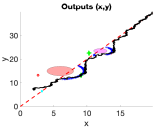
(12) u_2



(13) $x(t)$

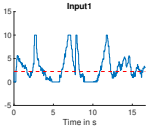


(14) $y(t)$

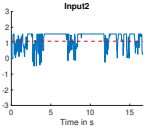


(15) (x, y)

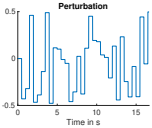
Perturbed autonomous robot: MFPC



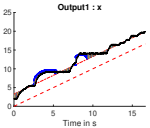
(16) u_1



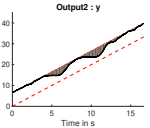
(17) u_2



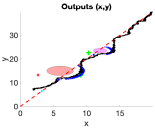
(18) p



(19) $x(t)$



(20) $y(t)$



(21) (x, y)

Today's AI & LeCun's program

Remarkable advances in modern AI (e.g. Generative AI) often rely on

- models striving for comprehensiveness (e.g. LLMs: Large Language Models),
- their associated machine learning mechanisms.

Yann LeCun' **program**:

- **Machine learning sucks**
- **I do favor MPC over RL. I've been making that point since at least 2016**

Analogy with program in math. research – Ex. Langlands program

Today's AI & LeCun's program

Our techniques comply with those program = requirements:

- Supremacy of predictive control.
- Reduced rôle of machine learning.
- Abandon of reinforcement learning in favor of model predictive control.

Energy& Infrastructure

Brad Smith, Vice Chair & President of Microsoft, September 2024:

The capital spending needed for AI infrastructure and the new energy to power it goes beyond what any single company or government can finance.

Our more subtle techniques should be less greedy.

Today's references

- C. Join, E. Delaleau, M. Fliess, Model-Free Predictive Control: Introductory Algebraic Calculations, and a Comparison with HEOL and ANNs. *Joint IFAC Conf.: SSSC, TDS, COSY*, Gif-sur-Vette, France, 30 June-2 July 2025. arXiv:2502.00443
- C. Join, M. Fliess, Avoidance of an unexpected obstacle without reinforcement learning: Why not using advanced control-theoretic tools? IEEE 2025 - 13th Int. Conf. Syst. Contr. (ICSC), Marrakesh, Morocco, October 22-24, 2025. arXiv:2509.03721