Data-Enabled Predictive Control of Autonomous Energy Systems

Florian Dörfler

Automatic Control Laboratory, ETH Zürich
Acknowledgements

Brain-storming: P.M. Esfahani, B. Recht, R. Smith, B. Bamieh, I. Markovsky, and M. Morari

Jeremy Coulson

Linbin Huang

John Lygeros
Big, deep, intelligent and so on

- **unprecedented availability** of computation, storage, and data
- **theoretical advances** in optimization, statistics, and machine learning
- ...and **big-data** frenzy

→ increasing importance of **data-centric methods** in all of science/engineering

Make up your own opinion, but machine learning works too well to be ignored.
Feedback – our central paradigm

actuation

“making a difference to the world”

automation and control

sensing

“making sense of the world”

inference and data science

physical world

information technology
Control in a data-rich world

• ever-growing trend in CS and robotics: \textit{data-driven control} by-passing models
• canonical problem: \textit{black/gray-box system control} based on I/O samples

\textbf{Q:} Why give up physical modeling and reliable model-based algorithms?

Data-driven control is \textit{viable alternative} when

• models are too complex to be useful (e.g., fluids, wind farms, & building automation)
• first-principle models are not conceivable (e.g., human-in-the-loop, biology, & perception)
• modeling & system ID is too cumbersome (e.g., robotics & converter applications)

\textit{Central promise:} It is often easier to learn control policies directly from data, rather than learning a model.

\textit{Example:} PID
Snippets from the literature

1. **reinforcement learning** / dual control / stochastic adaptive control / approximate dynamic programming

with key **mathematical challenges**
- approximate (or neuro) **DP**
- (stochastic) **function approximation**
- **exploration-exploitation** trade-offs

and **practical limitations**
- **inefficiency**: computation & samples
- **complex and fragile** algorithms
- **safe real-time** exploration

Ø suitable for physical control systems with real-time & safety constraints?
2. gray-box **safe learning & control**
- **robust** → conservative & complex control
- **adaptive** → hard & asymptotic performance
- **contemporary learning** algorithms  
  (e.g., MPC + Gaussian processes / RL)
→ non-conservative, optimal, & safe
Ø limited applicability: need a-priori safety

3. Sequential **system ID + control**
- ID with uncertainty quantification followed by robust control design
→ recent finite-sample & end-to-end ID  
  + control pipelines out-performing RL
Ø ID seeks best but not most useful model
→ “easier to learn policies than models”
Key take-aways

• claim: *easier to learn controllers* from data rather than models
• data-driven approach is *no silver bullet* (see previous Ø)
• *predictive models are preferable over data* (even approximate)
  → models are tidied-up, compressed, & de-noised representations
  → model-based methods vastly out-perform model-agnostic ones

Ø  deadlock ?

• a useful ML insight: *non-parametric methods* are often preferable over parametric ones (e.g., basis functions vs. kernels)
  → build a predictive & non-parametric model directly from raw data?
If you had the **impulse response** of a LTI system, then . . .

- can **identify model** (e.g., transfer function or Kalman-Ho realization)
- . . . but can also build **predictive model directly from raw data**:

\[
y_{\text{future}}(t) = \begin{bmatrix} y_1 & y_2 & y_3 & \cdots \end{bmatrix} \cdot \begin{bmatrix} u_{\text{future}}(t) \\ u_{\text{future}}(t - 1) \\ u_{\text{future}}(t - 2) \\ \vdots \end{bmatrix}
\]

- **model predictive control** from data: dynamic matrix control (DMC)
- **today**: can we do so with arbitrary, finite, and corrupted I/O samples?
Contents

I. Data-Enabled Predictive Control (DeePC): Basic Idea


II. From Heuristics & Numerical Promises to Theorems


III. Application: End-to-End Automation in Energy Systems

complex 2-area power system: large \((n \approx 10^2)\), few measurements \((5)\), nonlinear, noisy, stiff, & with input constraints

control objective: damping of inter-area oscillations via HVDC but without model

seek a method that works reliably, can be efficiently implemented, & certifiable

→ automating ourselves
Behavioral view on LTI systems

**Definition:** A discrete-time *dynamical system* is a 3-tuple \((\mathbb{Z}_{\geq 0}, \mathcal{W}, \mathcal{B})\) where

(i) \(\mathbb{Z}_{\geq 0}\) is the discrete-time axis,

(ii) \(\mathcal{W}\) is a signal space, and

(iii) \(\mathcal{B} \subseteq \mathcal{W}^{\mathbb{Z}_{\geq 0}}\) is the behavior.

**Definition:** The dynamical system \((\mathbb{Z}_{\geq 0}, \mathcal{W}, \mathcal{B})\) is

(i) *linear* if \(\mathcal{W}\) is a vector space & \(\mathcal{B}\) is a subspace of \(\mathcal{W}^{\mathbb{Z}_{\geq 0}}\),

(ii) *time-invariant* if \(\mathcal{B} \subseteq \sigma \mathcal{B}\), where \(\sigma w_t = w_{t+1}\), and

(iii) *complete* if \(\mathcal{B}\) is closed \(\iff\) \(\mathcal{W}\) is finite dimensional.

\(\mathcal{B}\) = *set of trajectories* & \(\mathcal{B}_T\) is *restriction* to \(t \in [0, T]\)
LTI systems and matrix time series

foundation of state-space subspace system ID & signal recovery algorithms

\[(u(t), y(t))\] satisfy recursive difference equation

\[b_0 u_t + b_1 u_{t+1} + \ldots + b_n u_{t+n} + a_0 y_t + a_1 y_{t+1} + \ldots + a_n y_{t+n} = 0\]

(ARMA / kernel representation)

\[
\begin{bmatrix}
  b_0 & a_0 & b_1 & a_1 & \ldots & b_n & a_n
\end{bmatrix}
\]

spans left nullspace of \textit{Hankel matrix} (collected from data)

\[
\mathcal{H}_L \left( \begin{array}{c}
u \\ y
\end{array} \right) =
\begin{bmatrix}
(u_1) & (u_2) & (u_3) & \ldots & (u_{T-L+1}) \\
y_1 & y_2 & y_3 & \ldots & y_{T-L+1}
\end{array}
\begin{array}{c}
(u_2) & (u_3) & (u_4) & \ldots & \\
y_2 & y_3 & y_4 & \ldots & \\
\vdots & \vdots & \vdots & \ddots & \\
(u_L) & \ldots & \ldots & \ldots & (u_T)
\end{array}
\end{bmatrix}
\]
The Fundamental Lemma

**Definition:** The signal \( u = \text{col}(u_1, \ldots, u_T) \in \mathbb{R}^{mT} \) is **persistently exciting of order** \( L \) if \( \mathcal{H}_L(u) = \begin{bmatrix} u_1 & \cdots & u_{T-L+1} \\ \vdots & \ddots & \vdots \\ u_L & \cdots & u_T \end{bmatrix} \) is of full row rank, i.e., if the signal is **sufficiently rich and long** \((T - L + 1 \geq mL)\).

**Fundamental lemma** [Willems et al, ’05]: Let \( T, t \in \mathbb{Z}_{>0}, \) Consider

- a **controllable** LTI system \((\mathbb{Z}_{\geq 0}, \mathbb{R}^{m+p}, \mathcal{B})\), and
- a \( T \)-sample long **trajectory** \( \text{col}(u^d, y^d) \in \mathcal{B}_T \), where
- \( u \) is **persistently exciting** of order \( t + n \) (prediction span + # states).

Then \( \text{colspan} \left( \mathcal{H}_t \left( \begin{bmatrix} u \\ y \end{bmatrix} \right) \right) = \mathcal{B}_t \).
Cartoon of Fundamental Lemma

Persistently exciting \( \rightarrow \) controllable LTI \( \rightarrow \) sufficiently many samples

\[
x_{k+1} = Ax_k + Bu_k \\
y_k = Cx_k + Du_k
\]

Parametric state-space model \( \iff \) \( \text{colspan} \left[ \begin{array}{c} u_1 \\ y_1 \\ u_2 \\ y_2 \\ u_3 \\ y_3 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \end{array} \right] \)

Non-parametric model from raw data

All trajectories constructible from finitely many previous trajectories
Data-driven simulation [Markovsky & Rapisarda ’08]

**Problem**: predict future output $y \in \mathbb{R}^{p \cdot T_{\text{future}}}$ based on

- input signal $u \in \mathbb{R}^{m \cdot T_{\text{future}}}$ → to predict forward
- past data $\text{col}(u^d, y^d) \in \mathcal{B}_{T_{\text{data}}}$ → to form Hankel matrix

**Assume**: $\mathcal{B}$ controllable & $u^d$ persistently exciting of order $T_{\text{future}} + n$

**Solution**: given $(u_1, \ldots, u_{T_{\text{future}}}) \rightarrow$ compute $g$ & $(y_1, \ldots, y_{T_{\text{future}}})$ from

\[
\begin{bmatrix}
  u_1^d & u_2^d & \cdots & u_{T-N+1}^d \\
  \vdots & \vdots & \ddots & \vdots \\
  u_{T_{\text{future}}}^d & u_{T_{\text{future}}+1}^d & \cdots & u_T^d \\
  y_1^d & y_2^d & \cdots & y_{T-N+1}^d \\
  \vdots & \vdots & \ddots & \vdots \\
  y_{T_{\text{future}}}^d & y_{T_{\text{future}}+1}^d & \cdots & y_T^d
\end{bmatrix}
\]

\[ g = \begin{bmatrix}
  u_1 \\
  \vdots \\
  u_{T_{\text{future}}} \\
  y_1 \\
  \vdots \\
  y_{T_{\text{future}}}
\end{bmatrix} \]

**Issue**: predicted output is not unique $\rightarrow$ need to set initial conditions!
**Refined problem**: predict future output $y \in \mathbb{R}^{p \cdot T_{\text{future}}}$ based on

- initial trajectory $\text{col}(u_{\text{ini}}, y_{\text{ini}}) \in \mathbb{R}^{(m+p)T_{\text{ini}}} \rightarrow$ to estimate initial $x_{\text{ini}}$
- input signal $u \in \mathbb{R}^{m \cdot T_{\text{future}}} \rightarrow$ to predict forward
- past data $\text{col}(u^d, y^d) \in \mathcal{B}_{T_{\text{data}}} \rightarrow$ to form Hankel matrix

**Assume**: $\mathcal{B}$ controllable & $u^d$ persist. exciting of order $T_{\text{ini}} + T_{\text{future}} + n$

**Solution**: given $(u_1, \ldots, u_{T_{\text{future}}})$ & $\text{col}(u_{\text{ini}}, y_{\text{ini}})$

→ compute $g$ & $(y_1, \ldots, y_{T_{\text{future}}})$ from

⇒ if $T_{\text{ini}} \geq$ lag of system, then $y$ is unique

$$
\begin{bmatrix}
U_p \\
Y_p \\
U_f \\
Y_f
\end{bmatrix} \triangleq
\begin{bmatrix}
u_1^d & \cdots & u_{T-T_{\text{future}}-T_{\text{ini}}+1}^d \\
\vdots & \ddots & \vdots \\
u_{T_{\text{ini}}}^d & \cdots & u_{T-T_{\text{future}}+1}^d \\
u_{T_{\text{ini}}+1}^d & \cdots & u_T^d
\end{bmatrix}
$$

$$
\begin{bmatrix}
y_1^d & \cdots & y_{T-T_{\text{future}}-T_{\text{ini}}+1}^d \\
\vdots & \ddots & \vdots \\
y_{T_{\text{ini}}}^d & \cdots & y_{T-T_{\text{future}}+1}^d \\
y_{T_{\text{ini}}+1}^d & \cdots & y_T^d
\end{bmatrix} \triangleq
\begin{bmatrix}
y_1 \\
\vdots \\
y_{T_{\text{ini}}} \\
y_{T_{\text{ini}}+1} \\
\vdots \\
y_T
\end{bmatrix}
$$
Output Model Predictive Control

The canonical receding-horizon MPC optimization problem:

\[
\begin{align*}
\text{minimize} & \quad \sum_{k=0}^{T_{\text{future}}-1} \|y_k - r_{t+k}\|^2_Q + \|u_k\|^2_R \\
\text{subject to} & \quad x_{k+1} = Ax_k + Bu_k, \quad \forall k \in \{0, \ldots, T_{\text{future}} - 1\}, \\
& \quad y_k = Cx_k + Du_k, \quad \forall k \in \{0, \ldots, T_{\text{future}} - 1\}, \\
& \quad x_{k+1} = Ax_k + Bu_k, \quad \forall k \in \{-T_{\text{ini}} - 1, \ldots, -1\}, \\
& \quad y_k = Cx_k + Du_k, \quad \forall k \in \{-T_{\text{ini}} - 1, \ldots, -1\}, \\
& \quad u_k \in U, \quad \forall k \in \{0, \ldots, T_{\text{future}} - 1\}, \\
& \quad y_k \in Y, \quad \forall k \in \{0, \ldots, T_{\text{future}} - 1\}.
\end{align*}
\]

For a deterministic LTI plant and an exact model of the plant, MPC is the gold standard of control: safe, optimal, tracking, \ldots

- **quadratic cost** with $R > 0$, $Q \succeq 0$ & ref. $r$
- **model for prediction** over $k \in [0, T_{\text{future}} - 1]$
- **model for estimation** (many variations)
- **hard operational or safety constraints**
**Data-Enabled Predictive Control**

*DeePC* uses non-parametric and data-based Hankel matrix time series as prediction/estimation model inside MPC optimization problem:

$$\begin{align*}
\text{minimize} & \quad g, u, y \\
\text{subject to} & \quad T_{\text{future}} - 1 \\
& \quad \sum_{k=0}^{T_{\text{future}}-1} \|y_k - r_{t+k}\|_Q^2 + \|u_k\|_R^2 \\
& \quad \begin{bmatrix}
U_P \\
Y_P \\
U_f \\
Y_f
\end{bmatrix} g = \begin{bmatrix}
u_{\text{ini}} \\
y_{\text{ini}} \\
u \\
y
\end{bmatrix}, \\
u_k \in \mathcal{U}, \quad \forall k \in \{0, \ldots, T_{\text{future}} - 1\}, \\
y_k \in \mathcal{Y}, \quad \forall k \in \{0, \ldots, T_{\text{future}} - 1\}
\end{align*}$$

- Quadratic cost with $R \succ 0$, $Q \succeq 0$ & ref. $r$
- Non-parametric model for prediction and estimation
- Hard operational or safety constraints

- Hankel matrix with $T_{\text{ini}} + T_{\text{future}}$ rows from past data collected offline
  
  $$\begin{bmatrix}
U_P \\
U_f
\end{bmatrix} = \mathcal{H}_{T_{\text{ini}}+T_{\text{future}}} (u^d) \quad \text{and} \quad \begin{bmatrix}
Y_P \\
Y_f
\end{bmatrix} = \mathcal{H}_{T_{\text{ini}}+T_{\text{future}}} (y^d)$$

- Past $T_{\text{ini}} \geq$ lag samples $(u_{\text{ini}}, y_{\text{ini}})$ for $x_{\text{ini}}$ estimation updated online
Correctness for LTI Systems

**Theorem:** Consider a *controllable LTI system* and the DeePC & MPC optimization problems with *persistently exciting* data of order $T_{\text{ini}} + T_{\text{future}} + n$. Then the *feasible sets of DeePC & MPC coincide*.

**Corollary:** If $\mathcal{U}, \mathcal{Y}$ are *convex*, then also the *trajectories coincide*.

*Aerial robotics case study:*
Thus, *MPC carries over to DeePC*  
...at least in the *nominal case*. 

Beyond LTI, what about measurement noise, corrupted past data, and nonlinearities?
Noisy real-time measurements

\[ \text{minimize} \quad g, u, y \quad \sum_{k=0}^{T_{\text{future}} - 1} \|y_k - r_{t+k}\|_Q^2 + \|u_k\|_R^2 + \lambda y \|\sigma_y\|_1 \]

subject to

\[
\begin{bmatrix}
U_p \\
Y_p \\
U_f \\
Y_f
\end{bmatrix}
\begin{bmatrix}
u_{\text{ini}} \\
y_{\text{ini}} \\
u \\
y
\end{bmatrix}
+ \begin{bmatrix} 0 \\ \sigma_y \\ 0 \\ 0 \end{bmatrix},
\]

\[ u_k \in \mathcal{U}, \quad \forall k \in \{0, \ldots, T_{\text{future}} - 1\}, \]
\[ y_k \in \mathcal{Y}, \quad \forall k \in \{0, \ldots, T_{\text{future}} - 1\} \]

**Solution:** add slack to ensure feasibility with \( \ell_1 \)-penalty

\[ \Rightarrow \text{for } \lambda_y \text{ sufficiently large } \sigma_y \neq 0 \text{ only if constraint infeasible} \]

c.f. *sensitivity analysis* over randomized sims
Hankel matrix corrupted by noise

\[
\begin{align*}
\text{minimize} & \quad g, u, y \\
\text{subject to} & \quad \sum_{k=0}^{T_{\text{future}}-1} \|y_k - r_{t+k}\|_Q^2 + \|u_k\|_R^2 + \lambda_g \|g\|_1 \\
\end{align*}
\]

\[
\begin{bmatrix}
U_p \\
Y_p \\
U_f \\
Y_f
\end{bmatrix}
\begin{bmatrix}
u_{\text{ini}} \\
y_{\text{ini}} \\
u \\
y
\end{bmatrix} = g,
\]

\[
u_k \in \mathcal{U}, \quad \forall k \in \{0, \ldots, T_{\text{future}} - 1\},
\]

\[
y_k \in \mathcal{Y}, \quad \forall k \in \{0, \ldots, T_{\text{future}} - 1\},
\]

\textbf{Solution}: add a \( \ell_1 \)-penalty on \( g \)

\textbf{intuition}: \( \ell_1 \) sparsely selects

\{Hankel matrix columns\}  
= \{past trajectories\}  
= \{motion primitives\}

c.f. \textit{sensitivity analysis}  
over randomized sims

---

Average cost

Average constraint violations

\[
\lambda_g
\]

\[
\lambda_g
\]
Towards nonlinear systems . . .

**Idea**: lift nonlinear system to large/∞-dimensional bi-/linear system
→ Carleman, Volterra, Fliess, Koopman, Sturm-Liouville methods
→ nonlinear dynamics can be approximated LTI on finite horizons

→ **exploit size rather than nonlinearity** and find features in data
→ **regularization** singles out relevant features / basis functions

**case study**: regularization for \( g \) and \( \sigma_y \)
recall the central promise: it is easier to learn control policies directly from data, rather than learning a model.
Comparison to system ID + MPC

**Setup**: nonlinear stochastic quadcopter model with full state info

**DeePC** + $\ell_1$-regularization for $g$ and $\sigma_y$

**MPC**: system ID via prediction error method + nominal MPC

![Graphs showing comparison between DeePC and System ID + MPC](image)

- **Single fig-8 run**
- **Random sims**

![Cost vs Number of simulations](image)

![Constraint Violations](image)
from heuristics & numerical promises to \textit{theorems}
Robust problem formulation

1. the *nominal problem* (without $g$-regularization)

$$\begin{align*}
\text{minimize} & \quad \sum_{k=0}^{T_{\text{future}}-1} \|y_k - r_{t+k}\|_Q^2 + \|u_k\|_R^2 + \lambda y \|\sigma y\|_1 \\
\text{subject to} & \quad [\hat{U}_p \hat{Y}_p \hat{U}_f \hat{Y}_f] g = \begin{bmatrix} u_{\text{ini}} \\ \hat{y}_{\text{ini}} \\ u \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \sigma y \\ 0 \end{bmatrix}, \\
& \quad u_k \in \mathcal{U}, \quad \forall k \in \{0, \ldots, T_{\text{future}} - 1\}
\end{align*}$$

where $\hat{\cdot}$ denotes measured & thus possibly corrupted data

2. an *abstraction* of this problem

$$\begin{align*}
\text{minimize} & \quad f \left( \hat{U}_f g, \hat{Y}_f g \right) + \lambda y \|\hat{Y}_p g - \hat{y}_{\text{ini}}\|_1 \\
\text{where} & \quad G = \{ g : \hat{U}_p g = u_{\text{ini}} \& \hat{U}_f g \in \mathcal{U} \} \quad \text{(23/33)}
\end{align*}$$
3. a further abstraction

\[
\begin{align*}
\text{minimize } & \ c \left( \hat{\xi}, g \right) = \text{minimize } \ E_{\hat{\mathbb{P}}}[c(\xi, g)] \\
\text{with } & \ G = \left\{ g : \hat{U}_pg = u_{\text{ini}} \ & \hat{U}_fg \in \mathcal{U} \right\}, \ \text{measured } \hat{\xi} = \left( \hat{U}_p, \hat{U}_f, \hat{Y}_p, \hat{Y}_f, \hat{y}_{\text{ini}} \right), \\
& \ \& \hat{\mathbb{P}} = \delta_{\hat{\xi}} \ \text{denotes the empirical distribution from which we obtained } \hat{\xi}
\end{align*}
\]

4. the solution \( g^* \) of the above problem gives poor out-of-sample performance for the problem we really want to solve:

\[ E_{\mathbb{P}}[c(\xi, g^*)] \]

where \( \mathbb{P} \) is the unknown probability distribution of \( \xi \)

5. distributionally robust formulation

\[
\begin{align*}
\inf & \ \sup_{g \in G, Q \in B_{\epsilon}(\hat{\mathbb{P}})} E_Q[c(\xi, g)] \\
\text{where the ambiguity set } & \mathbb{B}_{\epsilon}(\hat{\mathbb{P}}) \text{ is an } \epsilon-\text{Wasserstein ball centered at } \hat{\mathbb{P}}:
\end{align*}
\]

\[
\mathbb{B}_{\epsilon}(\hat{\mathbb{P}}) = \left\{ P : \inf_{\Pi} \int \|\xi - \xi'\|_W \ d\Pi \leq \epsilon \right\} \ \text{where } \Pi \text{ has marginals } \hat{\mathbb{P}} \ \text{and } P
\]
5. **distributionally robust** formulation

\[
\inf_{g \in G} \sup_{Q \in B_\epsilon(\hat{P})} \mathbb{E}_Q [c(\xi, g)]
\]

where the **ambiguity set** \(B_\epsilon(\hat{P})\) is an \(\epsilon\)-Wasserstein ball centered at \(\hat{P}\):

\[
B_\epsilon(\hat{P}) = \left\{ P : \inf_{\Pi} \int \| \xi - \xi' \|_W d\Pi \leq \epsilon \right\}
\]

where \(\Pi\) has marginals \(\hat{P}\) and \(P\)

---

**Theorem**: Under minor technical conditions:

\[
\inf_{g \in G} \sup_{Q \in B_\epsilon(\hat{P})} \mathbb{E}_Q [c(\xi, g)] = \min_{g \in G} c\left(\hat{\xi}, g\right) + \epsilon \lambda y \|g\|_W^\star
\]

**Cor**: \(\ell_\infty\)-robustness in trajectory space \(\Leftrightarrow\) \(\ell_1\)-regularization of DeePC

---

**Proof** uses methods by Kuhn & Esfahani: semi-infinite problem becomes finite after marginalization & for discrete worst case
Relation to system ID & MPC

1. **regularized DeePC** problem

   \[
   \begin{align*}
   \text{minimize} & \quad f(u, y) + \lambda g \|g\|_2^2 \\
   \text{subject to} & \quad \begin{bmatrix} U_p \\ Y_p \\ U_f \\ Y_f \end{bmatrix} g = \begin{bmatrix} u_{\text{ini}} \\ y_{\text{ini}} \\ u \\ y \end{bmatrix}
   \end{align*}
   \]

2. standard model-based **MPC** (ARMA parameterization)

   \[
   \begin{align*}
   \text{minimize} & \quad f(u, y) \\
   \text{subject to} & \quad y = K \begin{bmatrix} u_{\text{ini}} \\ y_{\text{ini}} \end{bmatrix}
   \end{align*}
   \]

3. **subspace ID** \( y = Y_f g^* \)

   where \( g^* = g^*(u_{\text{ini}}, y_{\text{ini}}, u) \) solves

   \[
   \begin{align*}
   \arg\min & \quad \|g\|_2^2 \\
   \text{subject to} & \quad \begin{bmatrix} U_p \\ Y_p \\ U_f \end{bmatrix} g = \begin{bmatrix} u_{\text{ini}} \\ y_{\text{ini}} \\ u \end{bmatrix}
   \end{align*}
   \]

4. equivalent **prediction error ID**

   \[
   \begin{align*}
   \text{minimize} & \quad \sum_j \left\| y_j^d - K \begin{bmatrix} u_{\text{ini},j} \\ y_{\text{ini},j} \\ u_j \end{bmatrix} \right\|_2^2 \\
   \rightarrow & \quad y = K \begin{bmatrix} u_{\text{ini}} \\ y_{\text{ini}} \\ u \end{bmatrix} = Y_f g^*
   \end{align*}
   \]
\[
\text{minimize } f(u, y) \quad u \in U, y \in Y \\
\text{subject to } y = K \begin{bmatrix} u_{\text{ini}} \\ y_{\text{ini}} \\ u \end{bmatrix}
\]
where \( K \) solves
\[
\text{arg min } \sum_j \left\| y_j - K \begin{bmatrix} u_{\text{ini},j} \\ y_{\text{ini},j} \\ u_j \end{bmatrix} \right\|^2
\]

**regularized DeePC**

\[
\text{minimize } f(u, y) + \lambda g \left\| g \right\|_2^2 \\
\text{subject to } g = \begin{bmatrix} U_p \\ Y_p \\ U_f \\ Y_f \end{bmatrix} g = \begin{bmatrix} u_{\text{ini}} \\ y_{\text{ini}} \\ u \\ y \end{bmatrix}
\]

\[
\Rightarrow \text{feasible set of ID & MPC } \subseteq \text{feasible set for DeePC}
\]

\[
\Rightarrow \text{DeePC } \leq \text{MPC } + \lambda g \cdot \text{ID}
\]

"easier to learn control policies from data rather than models"
application: **end-to-end automation** in energy systems
Grid-connected converter control

**Task:** control converter (nonlinear, noisy, & constrained) without a model of the grid, line, passives, or inner loops

*DeePC* tracking constant $dq$-frame references subject to constraints
Effect of regularizations

DeePC time-domain cost

\[ \text{DeePC time-domain cost} = \sum_k \| y_k - r_k \|_Q^2 + \| u_k \|_R^2 \]
(closed-loop measurements)

Optimization cost

\[ \text{Optimization cost} = \sum_k \| y_k - r_k \|_Q^2 + \| u_k \|_R^2 + \lambda_g \| g \|_2^2 \]
(closed-loop measurements)
Data length

\[
T_{\text{ini}} = 40, \quad T_{\text{future}} = 30
\]

Sys ID + MPC

DeePC (\(T = 500\))

DeePC (\(T = 330\))

\(I_d^{\text{ref}} = 1, \quad I_q^{\text{ref}} = 0\) (open loop)

works like a charm for \(T\) large, \textit{but}

\[
\text{card}(g) = T - T_{\text{ini}} - T_{\text{future}} + 1
\]

\(\rightarrow\) (possibly?) prohibitive on \(\mu\)DSP
Power system case study

**extrapolation** from previous case study: const. voltage → grid

**complex** 2-area power system: large ($n \approx 10^2$), few measurements (5), nonlinear, noisy, stiff, & with input constraints

**control objective:** damping of inter-area oscillations via HVDC

**real-time** closed-loop MPC & DeePC become prohibitive (on laptop)

→ choose $T$, $T_{\text{ini}}$, and $T_{\text{future}}$ wisely
Choice of time constants

→ choose $T$ sufficiently large
→ short horizon $T_{\text{future}} \approx 10$
→ $T_{\text{ini}} \geq 10$ estimates sufficiently rich model complexity

Time-domain cost

$$\text{time-domain cost} = \sum_{k} \| y_k - r_k \|_Q^2 + \| u_k \|_R^2$$
(closed-loop measurements)
Summary & conclusions

- fundamental lemma from behavioral systems
- matrix time series serves as predictive model
- data-enabled predictive control (DeePC)

✓ certificates for deterministic LTI systems
✓ distributional robustness via regularizations
✓ outperforms ID + MPC in optimization metric

→ certificates for nonlinear & stochastic setup
→ adaptive extensions, explicit policies, ...
→ applications to building automation, bio, etc.

Why have these powerful ideas not been mixed long before?

Willems '07: “[MPC] has perhaps too little system theory and too much brute force computation in it.”

The other side often proclaims “behavioral systems theory is beautiful but did not prove utterly useful”