Computational Control

Data-Driven Predictive Control Part I

Saverio Bolognani and Keith Moffat

Automatic Control Laboratory (IfA)
ETH Zurich

System response

$$y_t = \underbrace{CA^t x_0}_{\text{free response}} + \underbrace{\sum_{j=0}^{t-1} CA^{t-1-j} B u_j}_{\text{convolution}}$$

Simplifying assumption (SISO)

For now, assume a scalar input $u \in \mathbb{R}$ and a scalar output $y \in \mathbb{R}$.

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{L-1} \end{bmatrix} = \underbrace{\begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{L-1} \end{bmatrix}}_{\mathcal{O}_L} x_0 + \underbrace{\begin{bmatrix} 0 & 0 & \dots & & \\ CB & 0 & \dots & & \\ CAB & CB & 0 & \dots & \\ \vdots & & & & \\ CA^{L-2}B & CA^{L-3}B & \dots & CAB & CB & 0 \end{bmatrix}}_{\mathcal{G}_L} \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ \vdots \\ u_{L-1} \end{bmatrix}$$

- $\mathcal{O}_L \in \mathbb{R}^{L \times n}$ (tall) is the **extended observability matrix**
- $\mathcal{G}_L \in \mathbb{R}^{L \times L}$ (square) is the **convolution matrix**

Free response

$$\underbrace{\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{L-1} \end{bmatrix}}_{\mathbf{y}_L} = \underbrace{\begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{L-1} \end{bmatrix}}_{\mathcal{O}_L} x_0$$

Assumption: Observable system (SISO)

$\mathcal{O}_n$, with n the state dimension, is full rank.

Last week we focused on whether we can reconstruct x_0 by looking at the output sequence.

- If $L < n$, then x_0 cannot be reconstructed (not enough information)
- If $L = n$, then x_0 can be reconstructed: $x_0 = \mathcal{O}_n^{-1}\mathbf{y}_n$
- If $L > n$, then the question is overdetermined:
  - x_0 can be reconstructed from a valid output sequence
  - not all y sequences are valid!

Example, $n = 1$

$$x_{t+1} = \frac{1}{2} x_t + u_t$$
$$y_t = x_t$$

$$\mathcal{O}_L = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{L-1} \end{bmatrix} = \begin{bmatrix} 1 \\ 1/2 \\ 1/4 \\ \vdots \\ 1/2^{L-1} \end{bmatrix}$$

Assuming no input, can you reconstruct x_0 from

- $y_0 = 1$?
- $y_0 = 2, y_1 = 1$?
- $y_0 = 1, y_1 = 1$?

What are all the output sequences produced by the free response of this system?

Exercise: Try with an undamped oscillator, $n = 2$.

Suppose that the input sequence $u_0, \ldots, u_L$ is non zero, but **known**.

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{L-1} \end{bmatrix} = \underbrace{\begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{L-1} \end{bmatrix}}_{\mathcal{O}_L} x_0 + \underbrace{\begin{bmatrix} 0 & 0 & \ldots & & & \\ CB & 0 & \ldots & & & \\ CAB & CB & 0 & \ldots & & \\ \vdots & & & & & \\ CA^{L-2}B & CA^{L-1}B & \ldots & CAB & CB & 0 \end{bmatrix}}_{\mathcal{G}_L} \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ \vdots \\ u_{L-1} \end{bmatrix}$$

Can you determine x_0?

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{L-1} \end{bmatrix} - \underbrace{\begin{bmatrix} 0 & 0 & \ldots & & & \\ CB & 0 & \ldots & & & \\ CAB & CB & 0 & \ldots & & \\ \vdots & & & & & \\ CA^{L-2}B & CA^{L-1}B & \ldots & CAB & CB & 0 \end{bmatrix}}_{\mathcal{G}_L} \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ \vdots \\ u_{L-1} \end{bmatrix} = \underbrace{\begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{L-1} \end{bmatrix}}_{\mathcal{O}_L} x_0$$

$$\mathbf{y}_L - \mathcal{G}_L \mathbf{u}_L = \mathcal{O}_L x_0$$

Similar to before, the problem can be solved if

$$\mathbf{y}_L - \mathcal{G}_L \mathbf{u}_L$$

is in the column image of $\mathcal{O}_L$.

For scalar systems

- if $L < n$, underdetermined
- if $L = n$, then $x_0 = \mathcal{O}_n^{-1} (\mathbf{y}_n - \mathcal{G}_n \mathbf{u}_n)$
- if $L > n$, overdetermined

Admissible input-output sequences

Interesting consequence: for $L > n$, $\mathbf{y}_L - \mathcal{G}_L \mathbf{u}_L$ needs to belong to a subspace!

All input-output trajectories of length N of the system need to satisfy

$$\begin{bmatrix} \mathbf{u}_L \\ \mathbf{y}_L \end{bmatrix} = \underbrace{\begin{bmatrix} I_L & 0 \\ \mathcal{G}_L & \mathcal{O}_L \end{bmatrix}}_{\Lambda_L} \begin{bmatrix} \mathbf{u}_L \\ x_0 \end{bmatrix}$$

- $\Lambda_L \in \mathbb{R}^{(L+L)\times(L+n)}$
- if $L \geq n$, Λ_L has rank $L + n$

Behavioral representation

Trajectories $\mathbf{u}_L, \mathbf{y}_L$ of the system need to belong to the **column image** of Λ_L.

- Λ_L is a full description of the system, exactly like A, B, C
- If you are given the matrix Λ_L, you can tell whether an input/output sequence is compatible with that plant
  - compatible = "there exists an initial state x_0 such that..."

"Behavioral System Theory" by Jan C. Willems, from approx the 80s.
System theory without classifying signals as inputs and outputs!

$L = n$ (observability problem)

$$\begin{bmatrix} \mathbf{u}_n \\ \mathbf{y}_n \end{bmatrix} = \underbrace{\begin{bmatrix} I_n & 0 \\ \mathcal{G}_n & \mathcal{O}_n \end{bmatrix}}_{\Lambda_n} \begin{bmatrix} \mathbf{u}_n \\ x_0 \end{bmatrix}$$

- Λ_n has dimension $2n \times 2n$
- Λ_n has rank $2n$

$\rightarrow$ invertible matrix

Any input/output sequence of length n can be "explained" by an initial condition x_0.

Toy example:

$$x_{t+1} = x_t + u_t$$
$$y_t = x_t$$

$L = n + 1$

$$\begin{bmatrix} u_0 \\ \vdots \\ u_{n-1} \\ u_n \\ y_0 \\ \vdots \\ y_{n-1} \\ y_n \end{bmatrix} = \underbrace{\begin{bmatrix} I_{n+1} & 0 \\ \mathcal{G}_{n+1} & \mathcal{O}_{n+1} \end{bmatrix}}_{\Lambda_{n+1}} \begin{bmatrix} u_0 \\ \vdots \\ u_{n-1} \\ u_n \\ x_0 \end{bmatrix}$$

- Λ_{n+1} has dimension $2(n+1) \times 2n+1$
- Λ_{n+1} has rank $L + n = 2n + 1$
- the input/output sequence is a vector in $\mathbb{R}^{2(n+1)}$

$\rightarrow$ the input/output sequence belongs to a subspace of dimension $2n + 1$

Toy example:

$L = n + 1$ (prediction problem)

$$\begin{bmatrix} u_0 \\ \vdots \\ u_{n-1} \\ u_n \\ y_0 \\ \vdots \\ y_{n-1} \\ y_n \end{bmatrix} = \underbrace{\begin{bmatrix} I_{n+1} & 0 \\ \mathcal{G}_{n+1} & \mathcal{O}_{n+1} \end{bmatrix}}_{\Lambda_{n+1}} \begin{bmatrix} u_0 \\ \vdots \\ u_{n-1} \\ u_n \\ x_0 \end{bmatrix}$$

Prediction problem

Suppose that y_n is missing. Can you reconstruct it?

- Intuitively:
  - We already know that we can reconstruct x_0 from the first n input/output samples
  - Once we have x_0 we can "simulate" the output until y_n.

$L = n + K$ (prediction problem)

$$\begin{bmatrix} u_0 \\ \vdots \\ u_{n-1} \\ u_n \\ \vdots \\ u_{n+K-1} \\ y_0 \\ \vdots \\ y_{n-1} \\ y_n \\ \vdots \\ y_{n+K-1} \end{bmatrix} = \underbrace{\begin{bmatrix} I_{n+K} & 0 \\ \mathcal{G}_{n+K} & \mathcal{O}_{n+K} \end{bmatrix}}_{\Lambda_{n+K}} \begin{bmatrix} u_0 \\ \vdots \\ u_{n-1} \\ u_n \\ \vdots \\ u_{n+K-1} \\ x_0 \end{bmatrix}$$

Prediction problem

Suppose that the sequence $y_n \dots y_{n+K-1}$ is missing. Can you reconstruct it?

“Linear-algebra” interpretation

Any I/O sequence of length $L > n$ belongs to a subspace of dimension $L + n$

$$\begin{bmatrix} u_0 \\ \vdots \\ u_{n-1} \\ u_n \\ \vdots \\ u_{n+K-1} \\ y_0 \\ \vdots \\ y_{n-1} \\ y_n \\ \vdots \\ y_{n+K-1} \end{bmatrix} = Hg$$

Image representation

- columns of $H \in \mathbb{R}^{2L \times (L+n)}$ are a basis of the subspace
- $g \in \mathbb{R}^{L+n}$ is a vector of coefficients

- A basis for this subspace is given by the columns on Λ_L
  - We need A, B, C to build Λ_L
- Can you find another basis?

  Hint: m linearly independent vectors from a subspace of dim. m form a basis.

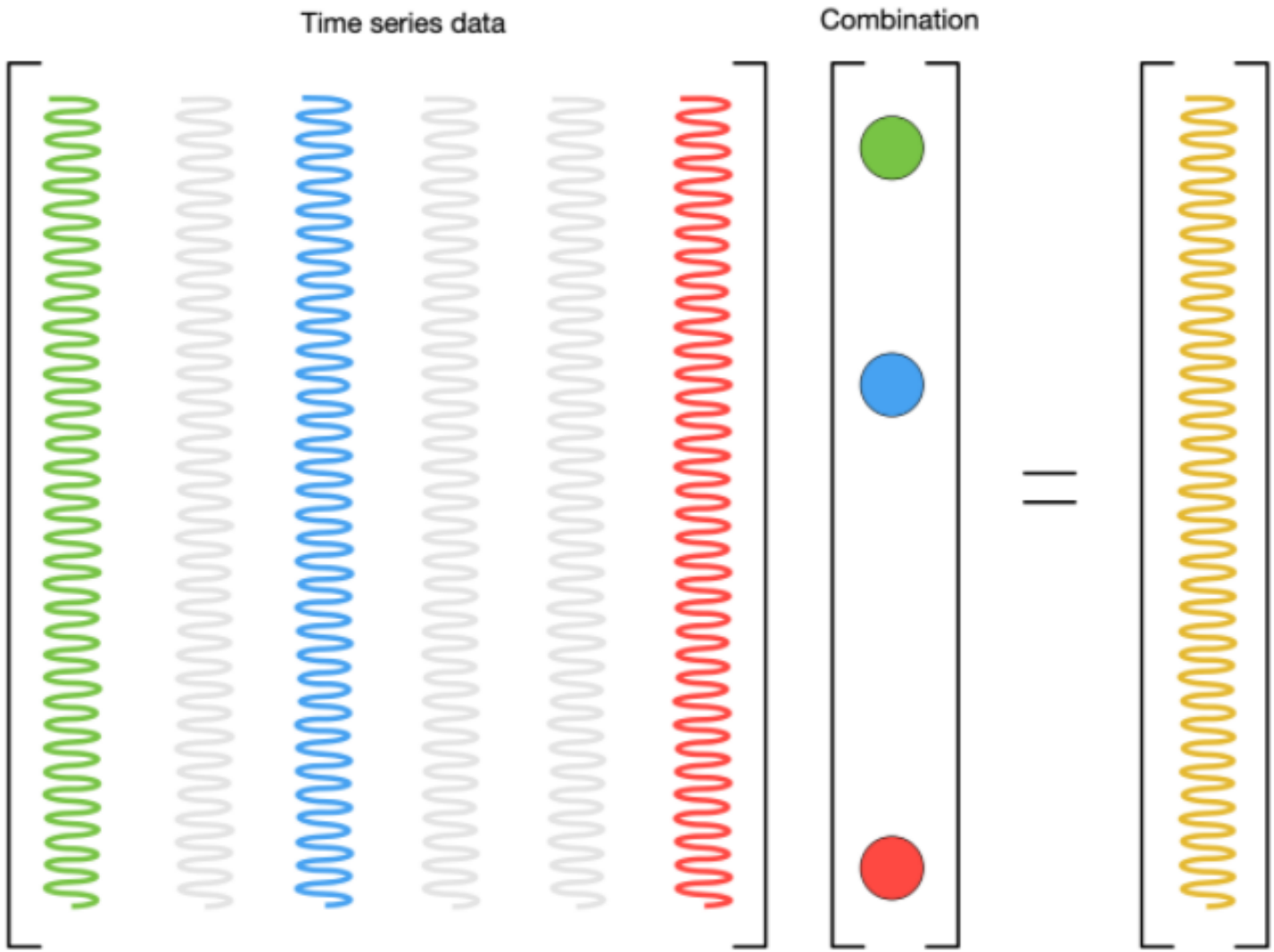
  How would you generate columns from this subspace? Data!

$$\begin{bmatrix} \mathbf{u}_L^{(1)} \\ \mathbf{y}_L^{(1)} \end{bmatrix} \begin{bmatrix} \mathbf{u}_L^{(2)} \\ \mathbf{y}_L^{(2)} \end{bmatrix} \begin{bmatrix} \mathbf{u}_L^{(3)} \\ \mathbf{y}_L^{(3)} \end{bmatrix} \cdots \begin{bmatrix} \mathbf{u}_L^{(L+n)} \\ \mathbf{y}_L^{(L+n)} \end{bmatrix}$$

- I/O sequences must belong to the subspace of valid signals
- **Assuming linear independence**, $L+n$ of them will form a basis.
- **All** and **only** the valid I/O sequences are of the form

$$\begin{bmatrix} \mathbf{u}_L \\ \mathbf{y}_L \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{u}_L^{(1)} & \mathbf{u}_L^{(2)} & \mathbf{u}_L^{(3)} & \cdots & \mathbf{u}_L^{(L+n)} \\ \mathbf{y}_L^{(1)} & \mathbf{y}_L^{(2)} & \mathbf{y}_L^{(3)} & \cdots & \mathbf{y}_L^{(L+n)} \end{bmatrix}}_{H} g$$

  with $g \in \mathbb{R}^{L+n}$.


Time series data
Combination

Exercise

- Consider the scalar ($n = 1$) SISO system

$$x_{t+1} = x_t + u_t$$
$$y_t = x_t$$

- Consider input/output trajectories of length $L = 3$.
- Generate trajectories of length L (random initial conditions, random inputs)
- Check the rank of the matrix H for an increasing number of columns
- What is the maximum rank?

Data matrix

Data is expensive and hard to obtain.

- to predict K steps in the future for a system of dimension n we need trajectories of length $L = K + n$
- we need at least $L + n = K + 2n$ of them

A very efficient use of your data points:

$$\ldots \quad u_0 \quad u_1 \quad u_2 \quad u_3 \quad u_4 \quad u_5 \quad u_6 \quad u_7 \quad \ldots$$

rearranged as

$$\mathcal{H}_L(\mathbf{u}) = \begin{bmatrix} u_0 & u_1 & u_2 & \ldots & u_{L+n-1} \\ u_1 & u_2 & u_3 & & \\ u_2 & u_3 & u_4 & & \\ \ldots & & & & \\ u_{L-1} & & & & u_{2L+n-2} \end{bmatrix}$$

Have you met this matrix before?

$$\mathcal{H}_L(\mathbf{u}) = \begin{bmatrix} u_0 & u_1 & u_2 & \dots & u_{L+n-1} \\ u_1 & u_2 & u_3 & & \\ u_2 & u_3 & u_4 & & \\ \dots & & & & \\ u_{L-1} & & & & u_{2L+n-2} \end{bmatrix}$$

$$\mathcal{H}_L(\mathbf{y}) = \begin{bmatrix} y_0 & y_1 & y_2 & \dots & y_{L+n-1} \\ y_1 & y_2 & y_3 & & \\ y_2 & y_3 & y_4 & & \\ \dots & & & & \\ y_{L-1} & & & & y_{2L+n-2} \end{bmatrix}$$

All system trajectories of length L are generated as

$$\begin{bmatrix} \mathcal{H}_L(\mathbf{u}) \\ \mathcal{H}_L(\mathbf{y}) \end{bmatrix} g$$

- $\mathbf{u}$ and $\mathbf{y}$ sequences of length $2L + n - 1$
- $g \in \mathbb{R}^{L+n}$ free vector of parameters

K-step predictor ($L = n + K$)

Given the **past input**, **past output**, and **future input**, compute **future output**.

$$\begin{bmatrix} \mathcal{H}_L(\mathbf{u}_{\text{data}}) \\ \mathcal{H}_L(\mathbf{y}_{\text{data}}) \end{bmatrix} g = \begin{bmatrix} \mathbf{u}_{\text{past}} \\ \mathbf{u}_{\text{future}} \\ \mathbf{y}_{\text{past}} \\ \mathbf{y}_{\text{future}} \end{bmatrix}$$

- $g \in \mathbb{R}^{2n+K}$ and $\mathbf{y}_{\text{future}} \in \mathbb{R}^K$ unknowns $\rightarrow$ total $2(n+K)$
- $2(n+K)$ equations

Many uses:

- Filtering
- Recover missing data
- Prediction / simulation
- **Control**

Model-based Predictive Control

Consider an **output tracking MPC** problem (SISO, for now).

At each time, the state x is measured and the following optimization problem is solved:

$$\begin{aligned} \min_{\mathbf{u},\mathbf{x},\mathbf{y}} \quad & y^T Q y + u^T R u \\ \text{subject to} \quad & x_{k+1} = A x_k + B u_k \\ & y_k = C x_k \\ & x_0 = x \\ & u_k \in \mathcal{U}_k \ \forall k \\ & y_k \in \mathcal{Y}_k \ \forall k \end{aligned}$$

The model equations enforce that **u** and **y** are trajectories of the plant for the next K steps.

Data-enabled Predictive Control

Replace the state-space model with our new behavioral model

$$\min_{\mathbf{u},\mathbf{y},g} \quad y^T Q y + u^T R u$$

$$\text{subject to} \quad \begin{bmatrix} \mathcal{H}_L(\mathbf{u}_{\text{data}}) \\ \mathcal{H}_L(\mathbf{y}_{\text{data}}) \end{bmatrix} g = \begin{bmatrix} \mathbf{u}_{\text{past}} \\ \mathbf{u} \\ \mathbf{y}_{\text{past}} \\ \mathbf{y} \end{bmatrix}$$

$$u_k \in \mathcal{U}_k \ \forall k$$

$$y_k \in \mathcal{Y}_k \ \forall k$$

where

- a new decision variable g has been introduced
- $\mathbf{u}_{\text{past}}$ and $\mathbf{y}_{\text{past}}$ are the last n inputs and outputs of the system, and play the role of the **initial condition**

$$\begin{aligned}
\min_{\mathbf{u},\mathbf{x},\mathbf{y}} \quad & y^T Q y + u^T R u \\
\text{subject to} \quad & x_{k+1} = A x_k + B u_k \\
& y_k = C x_k \\
& x_0 = x \\
& u_k \in \mathcal{U}_k \ \forall k \\
& y_k \in \mathcal{Y}_k \ \forall k
\end{aligned}$$

$$\begin{aligned}
\min_{\mathbf{u},\mathbf{y},g} \quad & y^T Q y + u^T R u \\
\text{subject to} \quad & \begin{bmatrix} \mathcal{H}_L(\mathbf{u}_{\text{data}}) \\ \mathcal{H}_L(\mathbf{y}_{\text{data}}) \end{bmatrix} g = \begin{bmatrix} \mathbf{u}_{\text{past}} \\ \mathbf{u} \\ \mathbf{y}_{\text{past}} \\ \mathbf{y} \end{bmatrix} \\
& u_k \in \mathcal{U}_k \ \forall k \\
& y_k \in \mathcal{Y}_k \ \forall k
\end{aligned}$$

- Computationally more complex ($2n + K$ additional decision variables)
- Larger memory footprint (**data** instead of **parametric model** A, B, C)
- No additional **state estimator** needed.

A few remarks

Assumption

We assumed **persistence of excitation** in the collected data

- technically, to ensure that the data matrix has full rank
- intuitively, because we want the data to represent all dynamics of the system

This is a standard requirement in **identification**, and it is typically resolved by

- noisy/random identification input sequences
- relying on ambient/process noise (for some applications)
- using historical data + a posteriori linear independence test



MIMO case

The theory extends very naturally to **multiple inputs** and **multiple outputs**.

Lag of a system

The **lag** of a system is the smallest ℓ such that

$$\mathcal{O}_\ell = \begin{bmatrix} C \\ CA \\ \dots \\ CA^{\ell-1} \end{bmatrix}$$

has rank n.

- A past trajectory of length $\rho \geq \ell$ is needed as "initial conditions".
- For MIMO systems
  - input $u \in \mathbb{R}^m$
  - output $y \in \mathbb{R}^p$
  - state $x \in \mathbb{R}^n$

  an input/output sequences of at length $T \geq (m+1)(\rho+K)+n-1$ is needed
- For SISO systems, $\ell = n$.

Lag and system size estimation

$\rightarrow$ a past trajectory of length $\rho \geq \ell$ is needed as "initial conditions"

$\rightarrow$ input/output sequences of at length $T \geq (m+1)(\rho+K)+n-1$ is needed

As $\ell \leq n$, in practice we need an a-priori estimate of the minimal realization size n.

- First principles
  - for example, electro-mechanical systems
- A posteriori verification
  - Build increasingly bigger Hankel matrices and verify the rank

Once the order is estimated, $\rho = n$ is a valid solution, although in some cases side information can allow to choose a smaller ρ.

K is determined by the control specifications (similar to MPC).

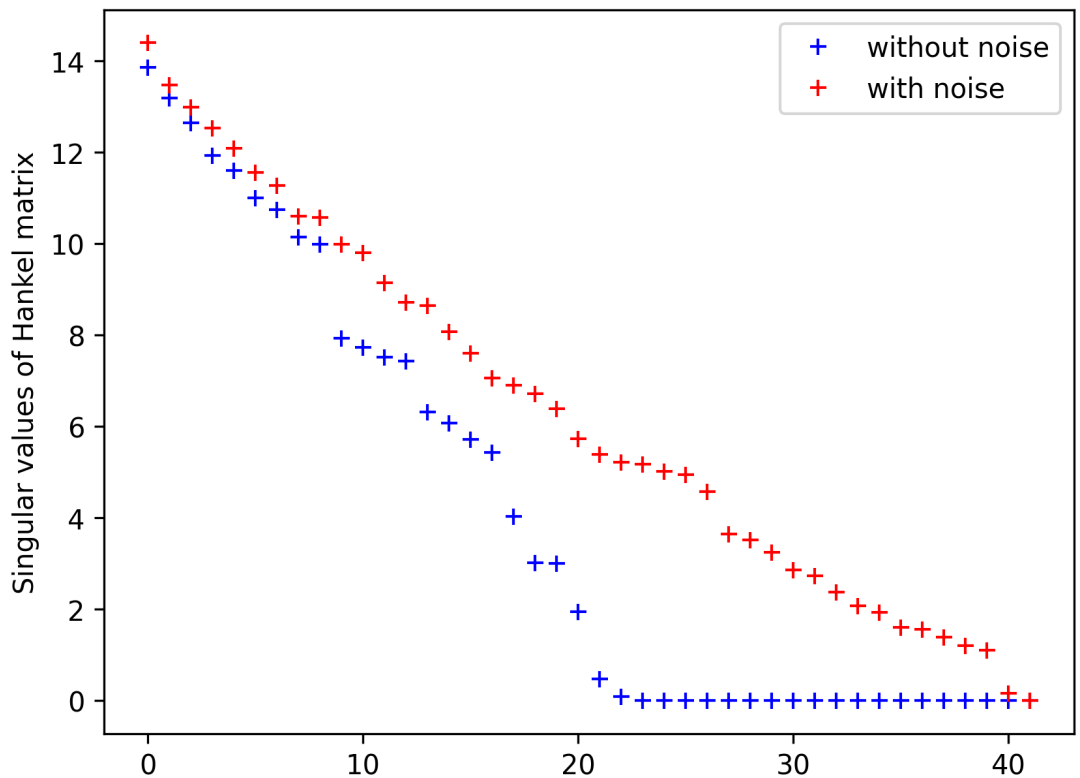
The effect of noise

Consequences of noise in the data

Without noise, $\text{rank}(H) \leq mL + n$, regardless of how many columns/data H has.

$$\underbrace{\begin{bmatrix} \mathbf{u}_L^{(1)} & \mathbf{u}_L^{(2)} & \mathbf{u}_L^{(3)} & \cdots \\ \mathbf{y}_L^{(1)} & \mathbf{y}_L^{(2)} & \mathbf{y}_L^{(3)} & \cdots \end{bmatrix}}_{H} g$$

With **noisy data**, the rank of H increases past $mL + n$:



The additional singular values/data rank does *not* describe true system behavior.

Consequences of noise in the data

Consider the (pathological) case in which one of the outputs is identically zero.

$$y_k = Cx_k = \begin{bmatrix} \star & \star & \star \\ 0 & 0 & 0 \end{bmatrix} x_k$$

All output sequences should be of the form

$$y_k = \begin{bmatrix} \star \\ 0 \end{bmatrix}$$

However, if the data contains noise, some columns in the data matrix may present a non-zero second coordinate in the output

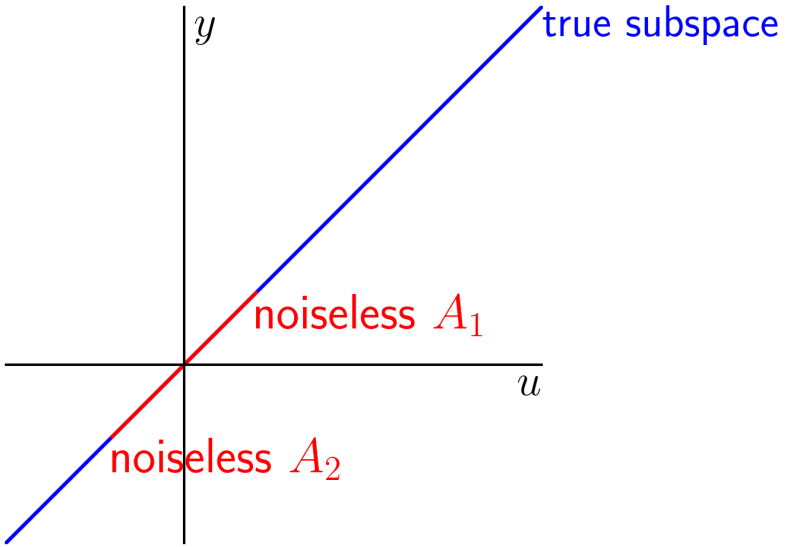
$$\underbrace{\begin{bmatrix} \mathbf{u}_L^{(1)} & \mathbf{u}_L^{(2)} & \mathbf{u}_L^{(3)} & \cdots \\ \mathbf{y}_L^{(1)} & \mathbf{y}_L^{(2)} & \mathbf{y}_L^{(3)} + \epsilon & \cdots \end{bmatrix}}_{H} g$$

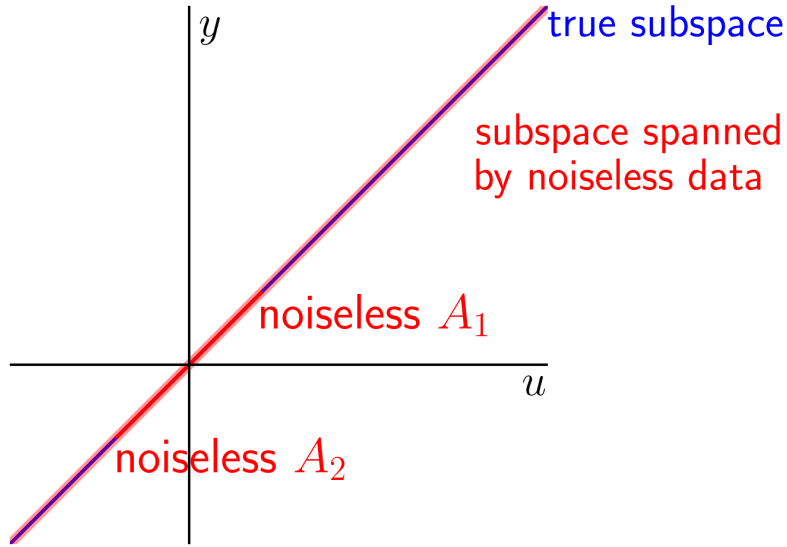
The controller is led to

- "assume" that the input sequence $\mathbf{u}_L^{(3)}$ is capable of affecting the last coordinate of the output!
- use a large g ($\rightarrow$ large input!) to try to affect the uncontrollable output

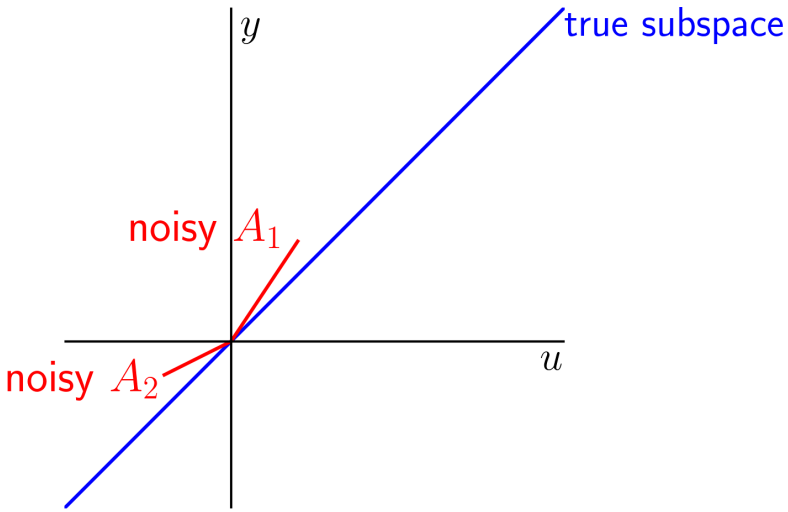
Consequences of noise in the data

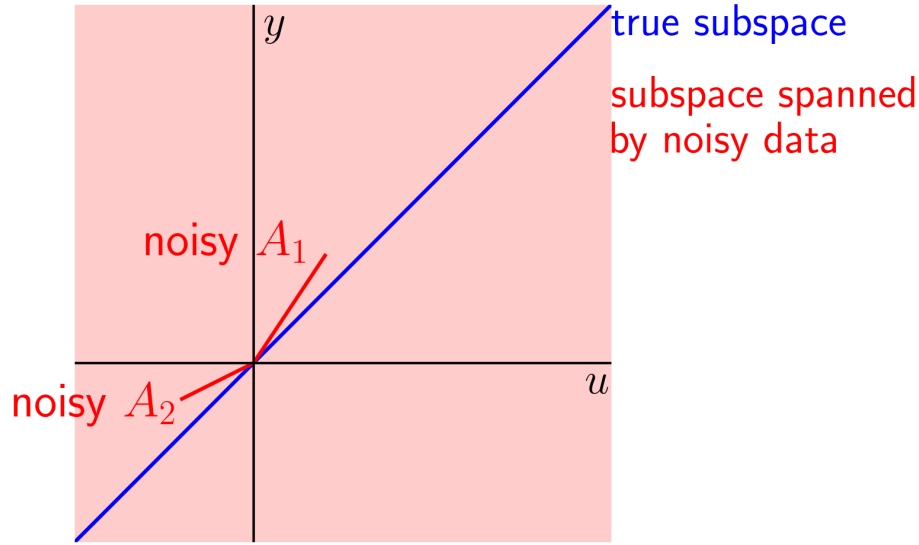
Now consider the simplified/projected case in which the true subspace is $y = u$





With noiseless data, the subspace spanned by the data is the line $y = u$.

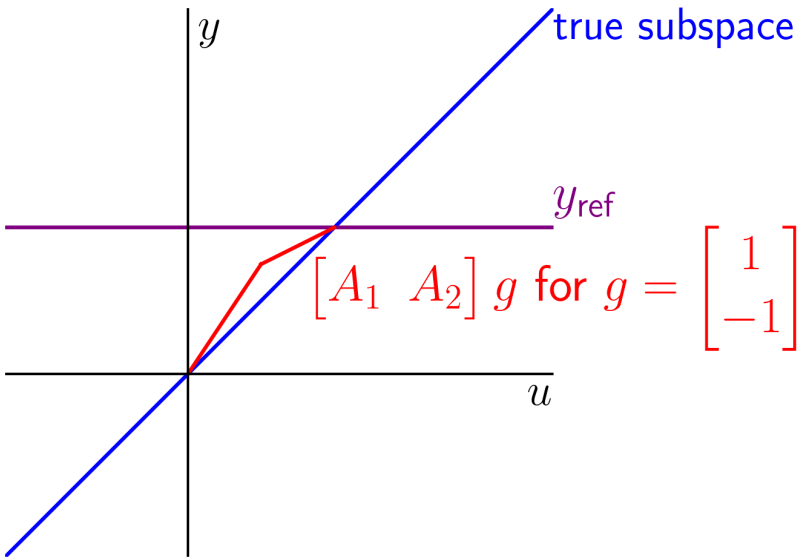


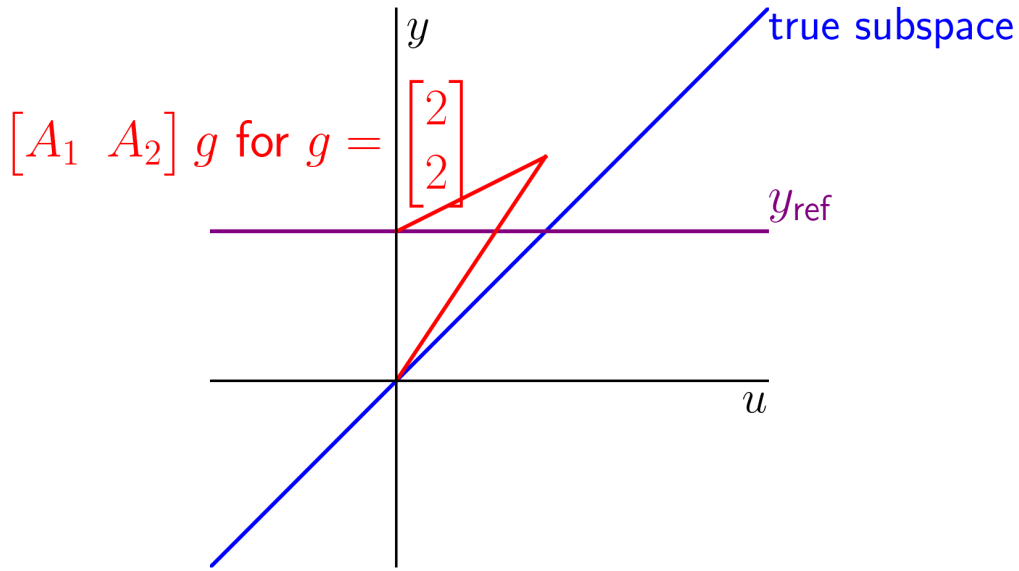


Whereas with noisy data, the subspace spanned by the data is all (u, y) pairs.

The relationship between noisy data and $\|g\|$

Using the noisy data to determine a u such that $y = y_{\text{ref}}$, there are many viable gs:





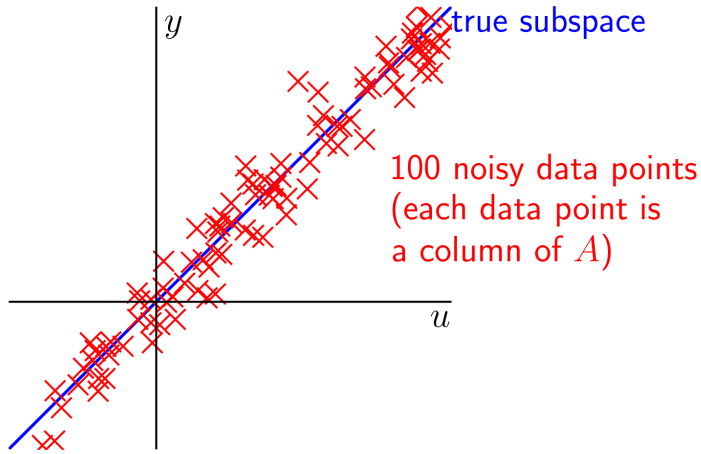
$g = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$ gives a valid (u, y) pair,

whereas $g = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$ predicts $y = y_{\text{ref}}$ for $u = 0$.

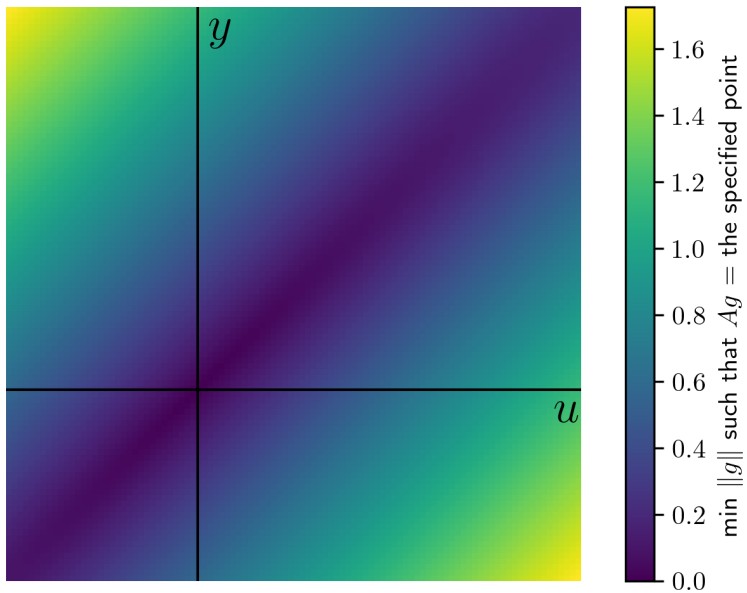
Thus, it would appear that “smaller” gs lead to more accurate predictions.

The relationship between noisy data and $\|g\|$

Consider a larger data set, with 100 noisy data points:



The heatmap below plots the minimum $\|g\|$ necessary for Ag to equal the given point, with A defined by the data above.



Indeed, "smaller" gs lead to more accurate predictions.

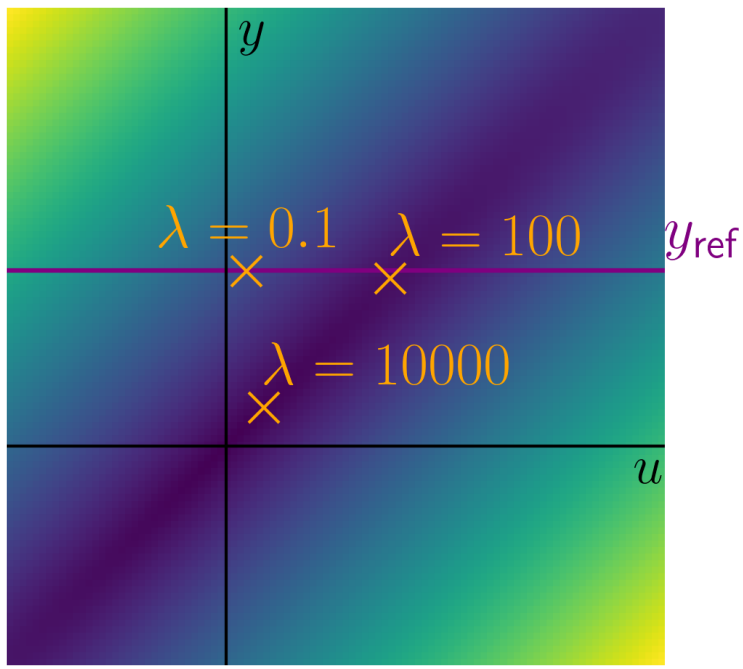
Addressing noise in the data by penalizing $\|g\|$

Consider the following optimization:

$$\min_{\mathbf{u},\mathbf{y},g} \quad (y - y_{\text{ref}})^2 + 0.01u^2 + \lambda\|g\|$$

$$\text{subject to} \quad Ag = \begin{bmatrix} u \\ y \end{bmatrix}$$

For $\lambda = 0.1$,100, and 10000, we get the following points:



- $\lambda = 0.1$ penalizes $\|g\|$ too little, allowing an incorrect (u, y) relationship.
- $\lambda = 100$ penalizes $\|g\|$ appropriately, and y is close to y_{ref}.
- $\lambda = 10000$ penalizes $\|g\|$ too much, pulling y far from y_{ref}.

Regularized Data-enabled Predictive Control

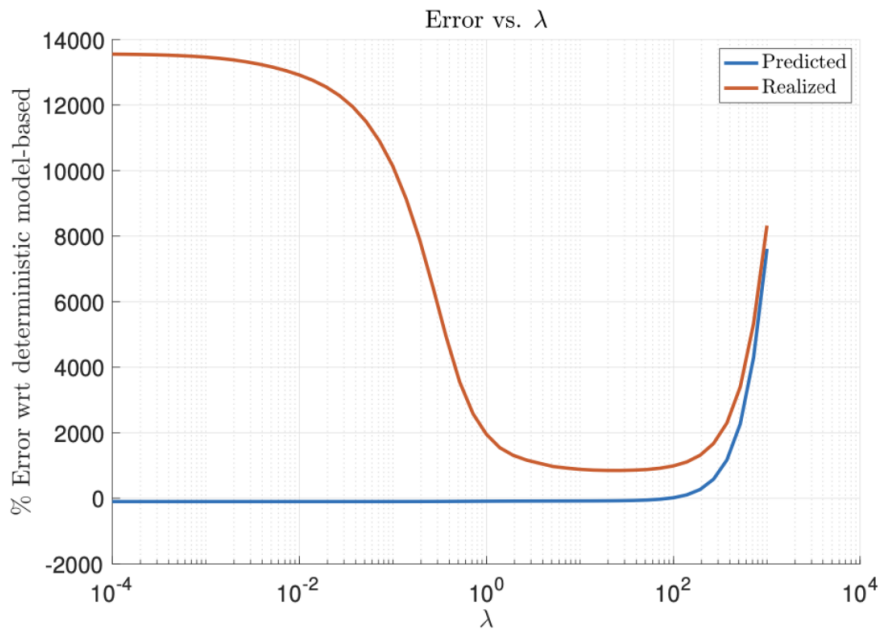
$$\min_{\mathbf{u},\mathbf{y},g} \quad y^T Q y + u^T R u + \lambda \|g\|$$

$$\text{subject to} \quad \begin{bmatrix} \mathcal{H}_L(\mathbf{u}_{\text{data}}) \\ \mathcal{H}_L(\mathbf{y}_{\text{data}}) \end{bmatrix} g = \begin{bmatrix} \mathbf{u}_{\text{past}} \\ \mathbf{u} \\ \mathbf{y}_{\text{past}} \\ \mathbf{y} \end{bmatrix}$$

$$u_k \in \mathcal{U}_k \ \forall k$$

$$y_k \in \mathcal{Y}_k \ \forall k$$

Larger λ_g increases robustness to noise, but (negatively) affects optimality:



Dörfler, Florian, Jeremy Coulson, and Ivan Markovsky. "Bridging direct and indirect data-driven control formulations via regularizations and relaxations." IEEE Transactions on Automatic Control 68.2 (2022): 883-897.

1-norm-regularized Data-enabled Predictive Control

Only $L + n$ elements of g are needed (for the noiseless SISO case)

$\rightarrow$ Penalizing the 1-norm of g rather than the 2-norm prioritizes a sparser g

$$\min_{\mathbf{u},\mathbf{y},g} \quad y^T Q y + u^T R u + \lambda_1 \|g\|_1$$

$$\text{subject to} \quad \begin{bmatrix} \mathcal{H}_L(\mathbf{u}_{\text{data}}) \\ \mathcal{H}_L(\mathbf{y}_{\text{data}}) \end{bmatrix} g = \begin{bmatrix} \mathbf{u}_{\text{past}} \\ \mathbf{u} \\ \mathbf{y}_{\text{past}} \\ \mathbf{y} \end{bmatrix}$$

$$u_k \in \mathcal{U}_k \ \forall k$$

$$y_k \in \mathcal{Y}_k \ \forall k$$

The 1-norm can work better than the 2-norm in practice. (Why?)

As with the 2-norm, larger λ_1 increases robustness to noise, but (negatively) affects optimality.

Noise in the real-time measurements

Noise may also be present in the "initial conditions" $\mathbf{u}_{\text{past}}, \mathbf{y}_{\text{past}}$.

Thus, it is possible, though rare, that $\mathbf{y}_{\text{past}}$ is not in span$(\mathcal{H}_L(\mathbf{y}_{\text{data}}))$.

In this case, the infeasibility can be resolved by adding a slack variable σ:

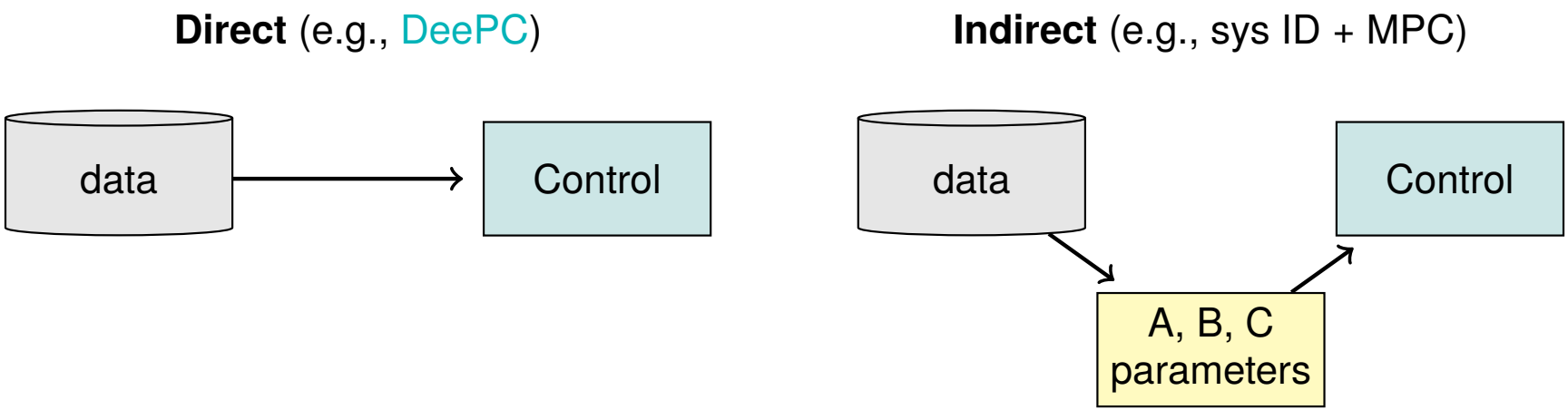
$$\min_{\mathbf{u},\mathbf{y},g,\sigma} \quad y^T Q y + u^T R u + \lambda \|g\| + \lambda_\sigma \|\sigma\|$$

$$\text{subject to} \quad \begin{bmatrix} \mathcal{H}_L(\mathbf{u}_{\text{data}}) \\ \mathcal{H}_L(\mathbf{y}_{\text{data}}) \end{bmatrix} g = \begin{bmatrix} \mathbf{u}_{\text{past}} \\ \mathbf{u} \\ \mathbf{y}_{\text{past}} + \sigma \\ \mathbf{y} \end{bmatrix}$$

$$u_k \in \mathcal{U}_k \ \forall k$$

$$y_k \in \mathcal{Y}_k \ \forall k$$

Noisy data: comparing DeePC and "indirect" methods



In addition to the noiseless benefits (no state needed), DeePC may be better-suited to deal with noise (less conservative) than **indirect methods**.

Evaluating and quantifying this claim is the subject of **active research**.

What is hard to do with data-driven control

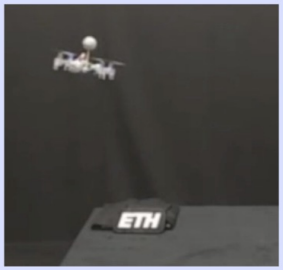
- **State constraint**
  If a state is known and its value needs to be controlled, then a model-based approach may work better.

- **Prior/side information**
  If some system information is available (range of parameters, nominal values, structure of the problem) then sysID can typically incorporate this information.

- **Nonlinearity**
  The theory we have seen today strongly relies on LTI system theory. Extensions to nonlinear systems are complicated.

- **Computational complexity**
  A data representation is less compact than state-space models (what is a model, by the way?). This requires memory and computation time.

https://control.ee.ethz.ch/research/theory/data-enabled-predictive-control.html



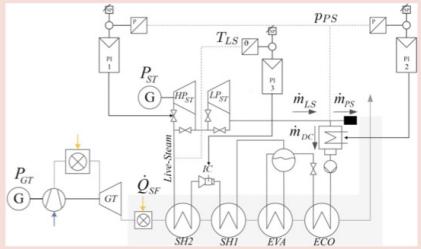
quad coptor fig-8 tracking



quadruped (by Fawcett, Afsari Amers, & Hamed)



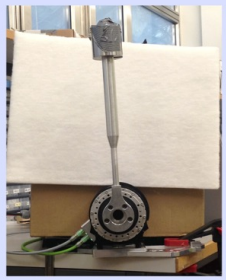
greenhouse automation (by Automatoes)



combined cycle power plant (by P Mahdavipour et. al)



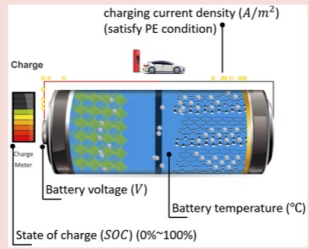
robotic excavator



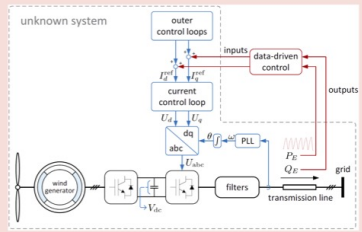
pendulum swing up



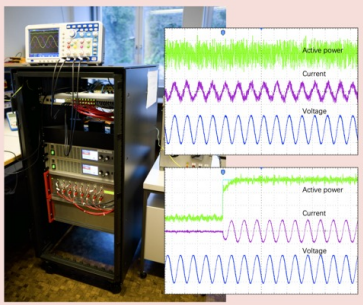
traffic coordination (by J. Wang et al.)



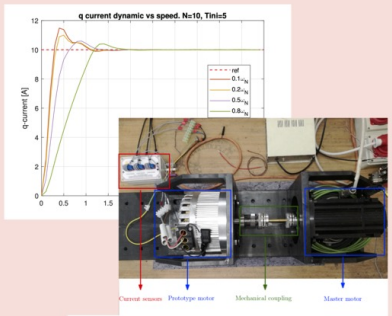
battery charging (by K. Chen et al.)



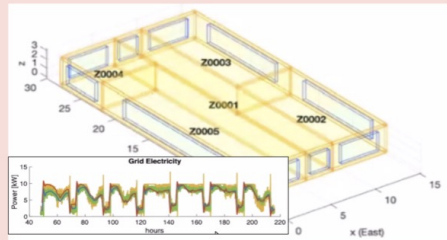
wind turbine control



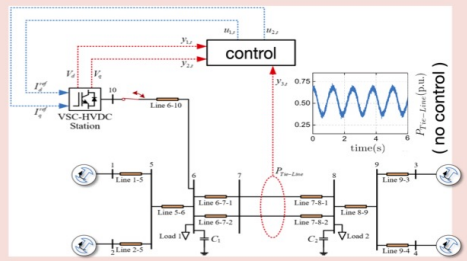
grid-connected converter



synchronous motor drive



energy hub & building automation



power system oscillation damping

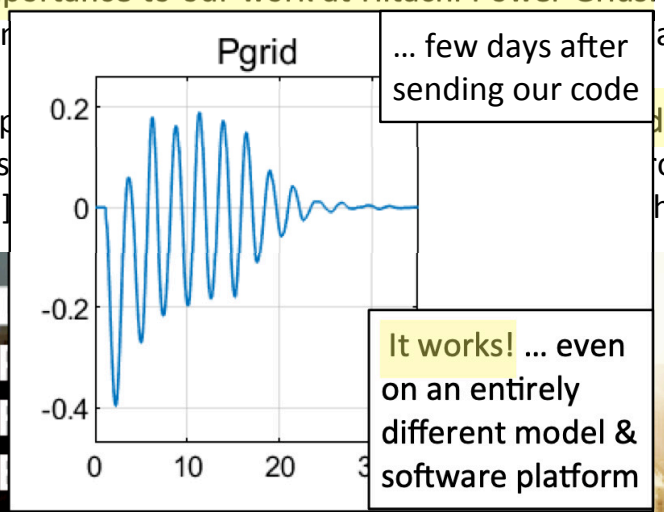
Surely, nobody would put apply such a **shaky data-driven method**

- on the **world's most complex engineered system** (the electric grid),
- using the **world's biggest actuators** (Gigawatt-sized HVDC links),
- and subject to **real-time, safety, & stability constraints** . . . right?

Dear Linbin and Florian,

I just submitted a very favourable review of your paper […] which I believe could be of importance to our work at Hitachi Power Grids. We do have […] require off-line tuning that […] cor an adaptive approach would be very interesting.

If p d DeePC approach with our more detailed HVDC sys oblem. Could so some code be made available […] her to do such a demonstration ? […]

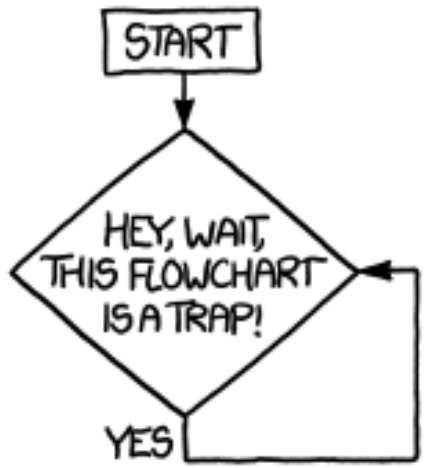


… few days after sending our code

It works! … even on an entirely different model & software platform



The control engineer flowchart





This work is licensed under a
Creative Commons Attribution-ShareAlike 4.0 International License

https://bsaver.io/COCO