Comparison: direct vs. indirect control

indirect* ID-based *data-driven control

minimize control cost (u, y)

subject to (u, y) satisfy parametric model

where model $\in$ argmin id cost (u^d, y^d)

subject to model $\in$ LTI(n, ℓ) class

ID

ID projects data on the set of LTI models

- with parameters (n, ℓ)
- removes noise & thus lowers variance error
- suffers bias error if plant is not LTI(n, ℓ)

direct* regularized *data-driven control

minimize control cost (u, y) + $\lambda \cdot$ regularizer

subject to (u, y) consistent with (u^d, y^d) data

- ***regularization robustifies***
  $\rightarrow$ choosing λ makes it work
- ***no projection*** on LTI(n, ℓ)
  $\rightarrow$ no de-noising & no bias

hypothesis: ID wins in stochastic (variance) & DeePC in nonlinear (bias) case

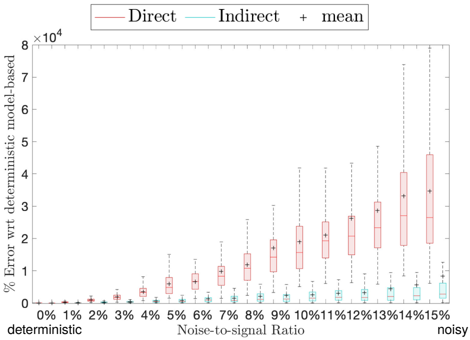
Case study: direct vs. indirect control

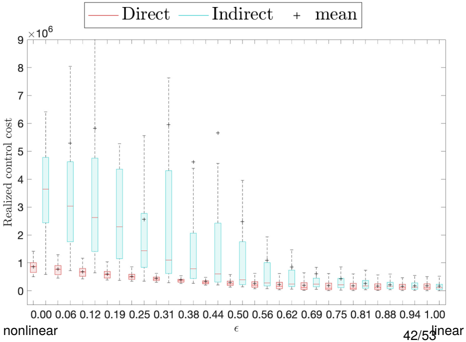
stochastic LTI case $\rightarrow$ indirect ID wins

- LQR control of 5th order LTI system
- Gaussian noise with varying noise to signal ratio (100 rollouts each case)
- ℓ_1-regularized DeePC, SysID via N4SID, & judicious hyper-parameters

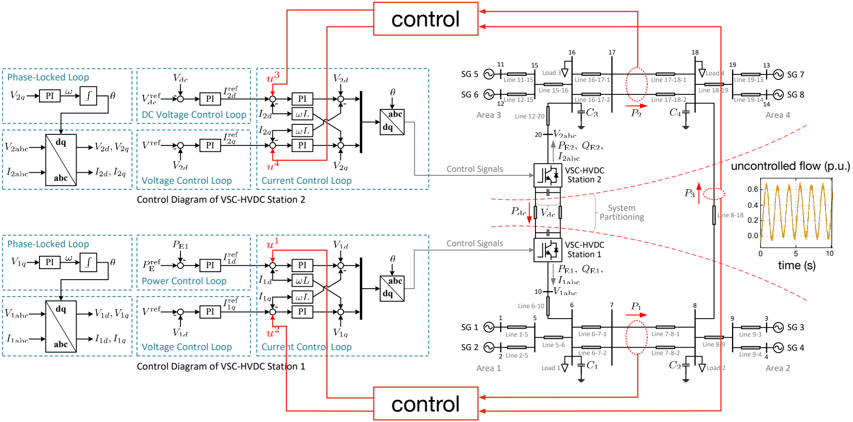
nonlinear case $\rightarrow$ direct DeePC wins

- Lotka-Volterra + control: $x^+ = f(x,u)$
- interpolated system $x^+ = \epsilon \cdot f_{\text{linearized}}(x,u) + (1-\epsilon)\cdot f(x,u)$
- same ID & DeePC as on the left & 100 initial x_0 rollouts for each ϵ



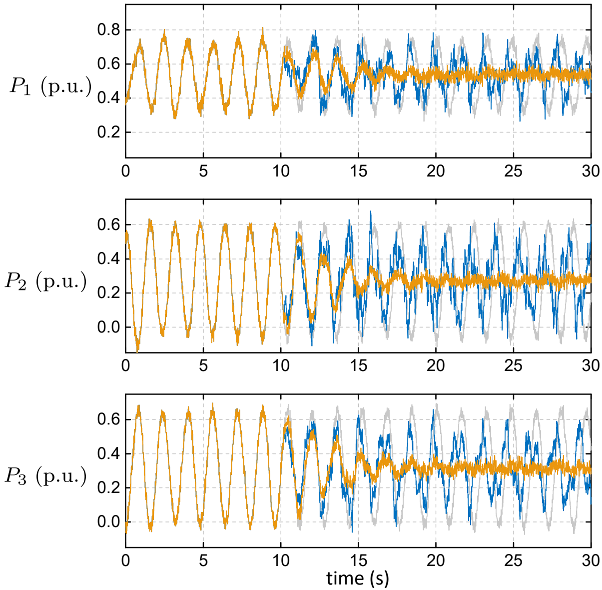


Power system case study revisited



- ***complex*** 4-area power ***system***: large ($n = 208$), few measurements (8), nonlinear, noisy, stiff, input constraints, & decentralized control
- ***control objective:*** damping of inter-area oscillations via HVDC link
- ***real-time*** MPC & DeePC prohibitive $\rightarrow$ choose T, T_{ini}, & T_{future} wisely

Centralized control



DeePC

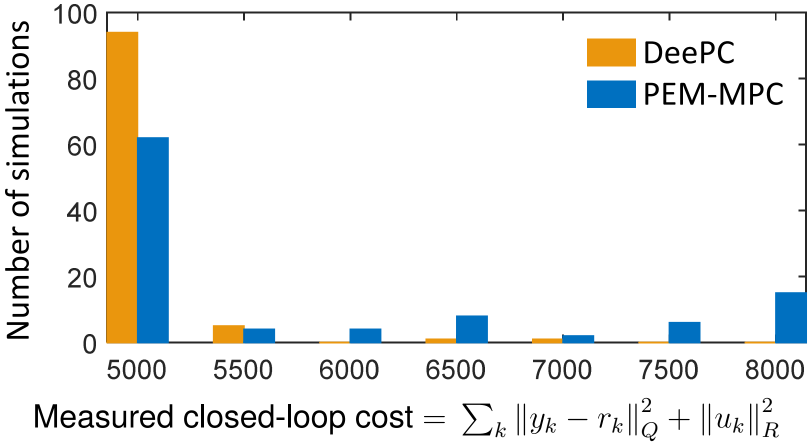
PEM-MPC

= Prediction Error Method (PEM) System ID + MPC

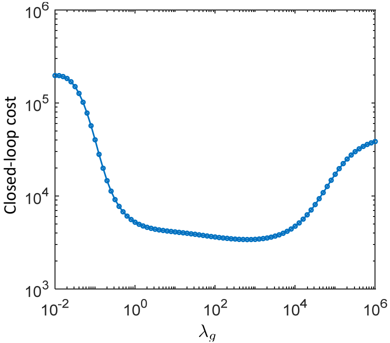
$t < 10\,$s: open loop data collection with white noise excitat.

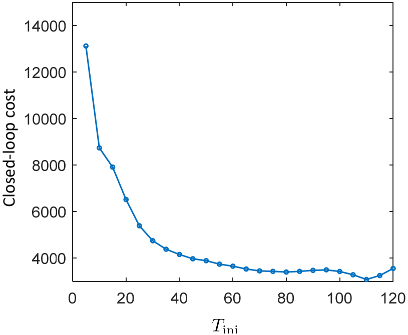
$t > 10\,$s: control

Performance: DeePC wins (clearly!)



DeePC hyper-parameter tuning





regularizer λ_g

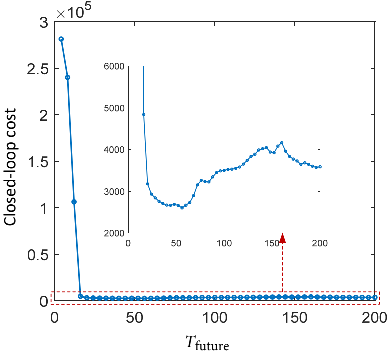
- for distributional robustness $\approx$ radius of Wasserstein ball
- wide range of sweet spots

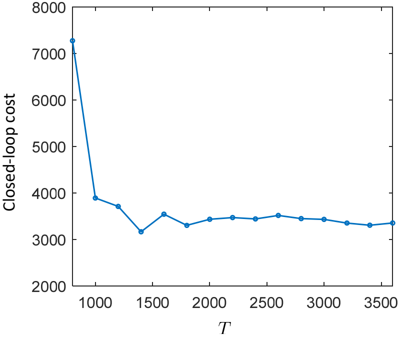
$\rightarrow$ choose $\lambda_g = 20$

estimation horizon T_{ini}

- for model complexity $\approx$ lag
- $T_{\text{ini}} \geq 50$ is sufficient & low computational complexity

$\rightarrow$ choose $T_{\text{ini}} = 60$





prediction horizon T_{future}

- nominal MPC is stable if horizon T_{future} long enough

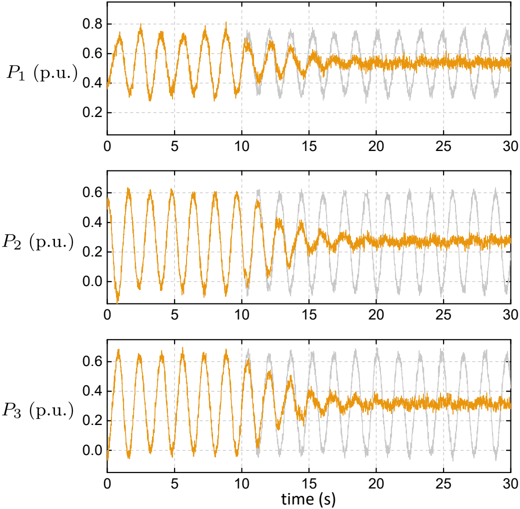
  $\rightarrow$ choose $T_{\text{future}} = 120$ & apply first 60 input steps

data length T

- long enough for low-rank condition but card(g) grows

  $\rightarrow$ choose $T = 1500$ (data matrix $\approx$ square)

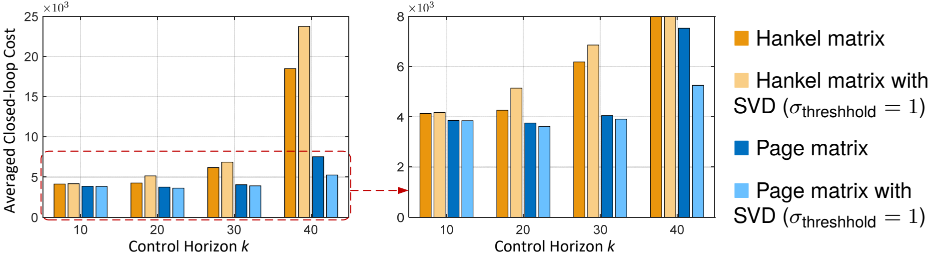
Computational cost



- $T = 1500$
- $\lambda_g = 20$
- $T_{\text{ini}} = 60$
- $T_{\text{future}} = 120$ & apply first 60 input steps
- sampling time = 0.02 s
- solver (OSQP) time = 1 s (on Intel Core i5 7200U)

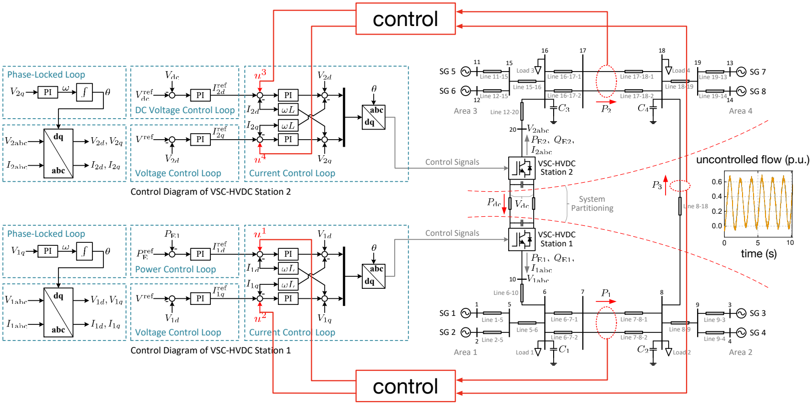
$\Rightarrow$ **implementable**

Comparison: Hankel & Page matrix



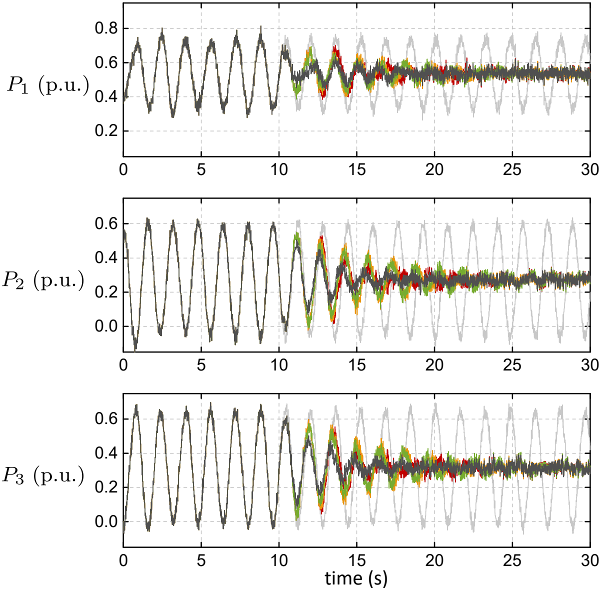
- comparison baseline: Hankel and Page matrices of ***same size***
- ***perfomance*** : Page consistency beats Hankel matrix predictors
- offline ***denoising via SVD threshholding*** works wonderfully for Page though obviously not for Hankel (entries are constrained)
- effects very pronounced for ***longer horizon*** (= open-loop time)
- ***price-to-be-paid*** : Page matrix predictor requires more data

Decentralized implementation



- ***plug'n'play MPC:*** treat interconnection P_3 as disturbance variable w with past disturbance w_{ini} measurable & future $w_{\text{future}} \in \mathcal{W}$ uncertain
- for each controller ***augment trajectory matrix*** with disturbance data w
- decentralized ***robust min-max DeePC:*** $\min_{g,u,y} \max_{w \in \mathcal{W}}$

Decentralized control performance



- colors correspond to different hyper-parameter settings (not discernible)
- ambiguity set $\mathcal{W}$ is ∞-ball (box)
- for computational efficiency $\mathcal{W}$ is downsampled (piece-wise linear)
- solver time ≈ 2.6 s

$\Rightarrow$ **implementable**

Conclusions

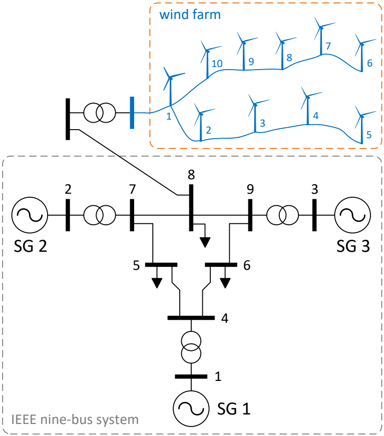
main take-aways

- matrix time series as predictive model
- robustness & side-info by regularization
- method that works in theory & practice
- focus is robust prediction not predictor ID

ongoing work

→ certificates for adaptive & nonlinear cases

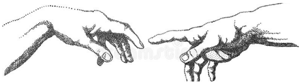
→ applications with a true “business case”, push TRL scale, & industry collaborations

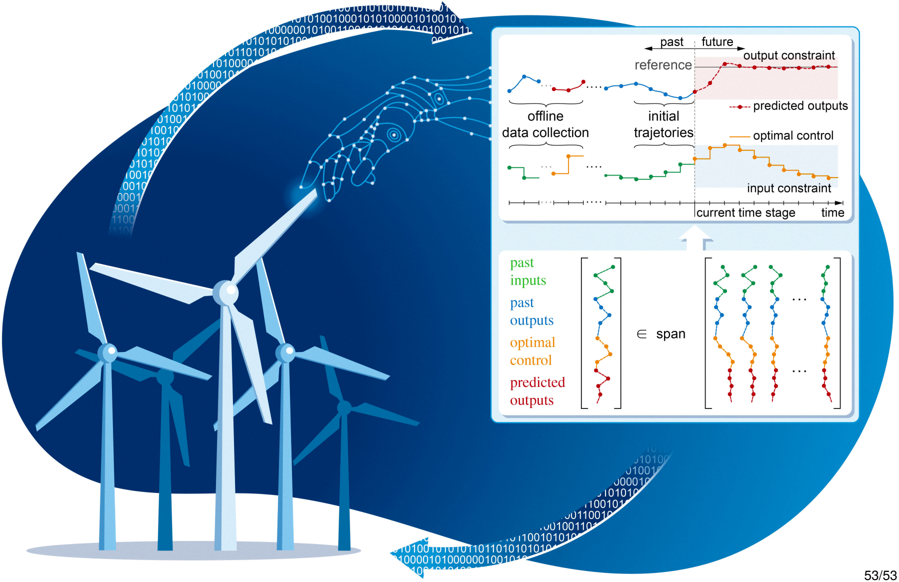


questions we should discuss

- catch? violate no-free-lunch theorem ? → more real-time computation
- when does direct beat indirect ? → Id4Control & bias/variance issues ?

Florian's version of





Thanks !

Florian Dörfler

mail: dorfler@ethz.ch

[link] to homepage

[link] to related publications