The **Twin-in-the-loop** approach

for vehicle dynamics estimation and control

Mälardalen University, Västerås, Sweden
March 21, 2023

Simone Formentin

Outline

- Problem statement
- Twin-in-the-Loop control
- Twin-in-the-Loop estimation
- Conclusions

Outline

Problem statement

Twin-in-the-Loop control

Twin-in-the-Loop estimation

Conclusions

Problem statement

Vehicle dynamics systems - design process

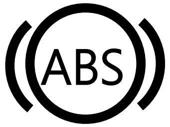
Traditional framework

- Vehicle dynamics systems development goes through **many steps**

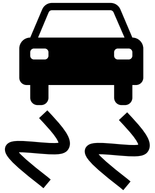
- **Many experiments** to achieve the final controller [1];

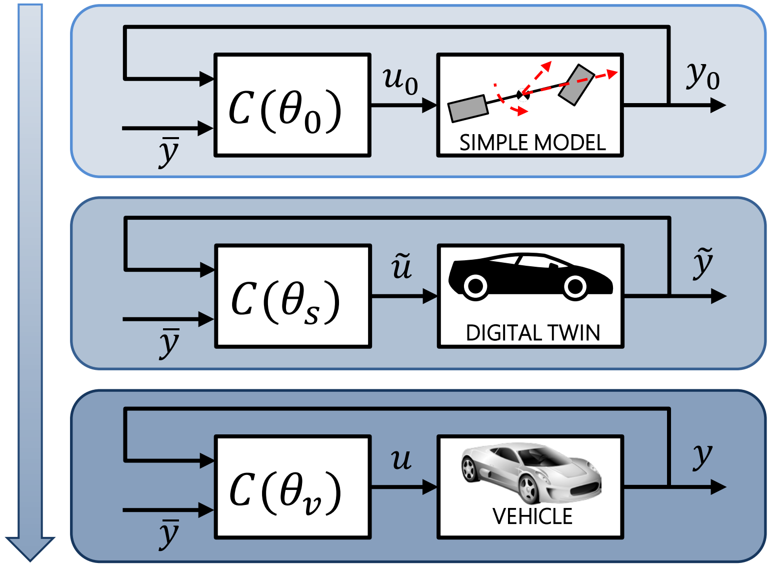
Vertical dynamics [2]

Longitudinal dynamics [3]



Lateral dynamics [4]





End-of-Line (EoL) tuning: most **time-consuming** task

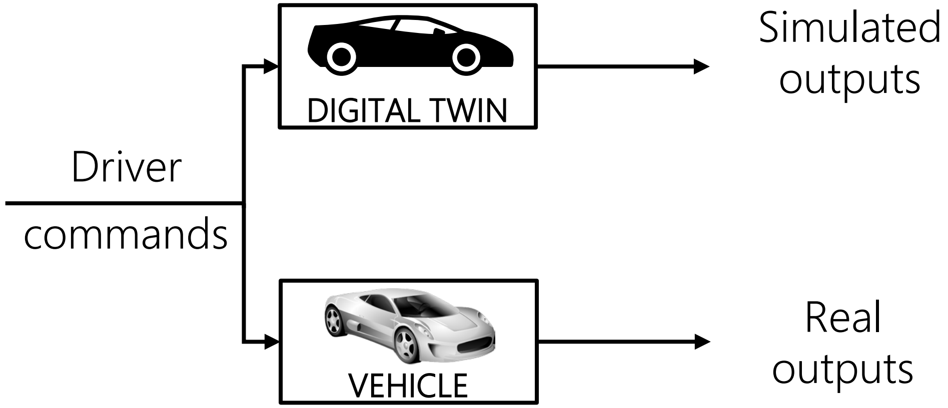
[1] M. Tanelli, G. Panzani, S.M. Savaresi and C. Pirola, **Transmission control for power-shift agricultural tractors: Design and end-of-line automatic tuning**, *Mechatronics*, 2011
[2] G. Savaia, Y. Sohn et al., **Experimental automatic calibration of a semi-active suspension controller via Bayesian Optimization**, *Control Engineering Practice*, 2021
[3] D. Tavernini, F. Vacca et al., **An explicit nonlinear model predictive ABS controller for electro-hydraulic braking systems**, *IEEE Transactions of Industrial Electronics*, 2020
[4] S. Xu and H. Peng, **Design, Analysis and Experiments of Preview Path Tracking Control for Autonomous Vehicles**, *IEEE Transactions on Intelligent Transportation Systems*, 2020

Framework and research goal

Problem statement – Twin-in-the-Loop control

Vehicle simulators currently used in development and prototyping stages [*].

- **Very accurate digital twins** (DT) of the vehicle;
- **Already available (and calibrated)** to car manufacturers.



Unexplored potential?

What if the digital twin can be used **on-board** and in **real-time**?

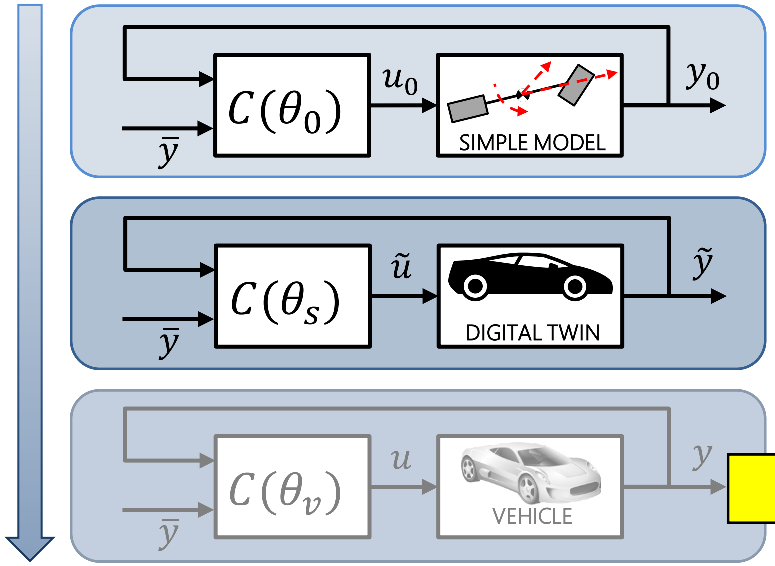
[*] E. Kutluay and H. Winner, **Validation of vehicle dynamics simulation models – a review**, *Vehicle System Dynamics*, 2014

Control design flow – shifting the paradigm

Problem statement – Twin-in-the-Loop control

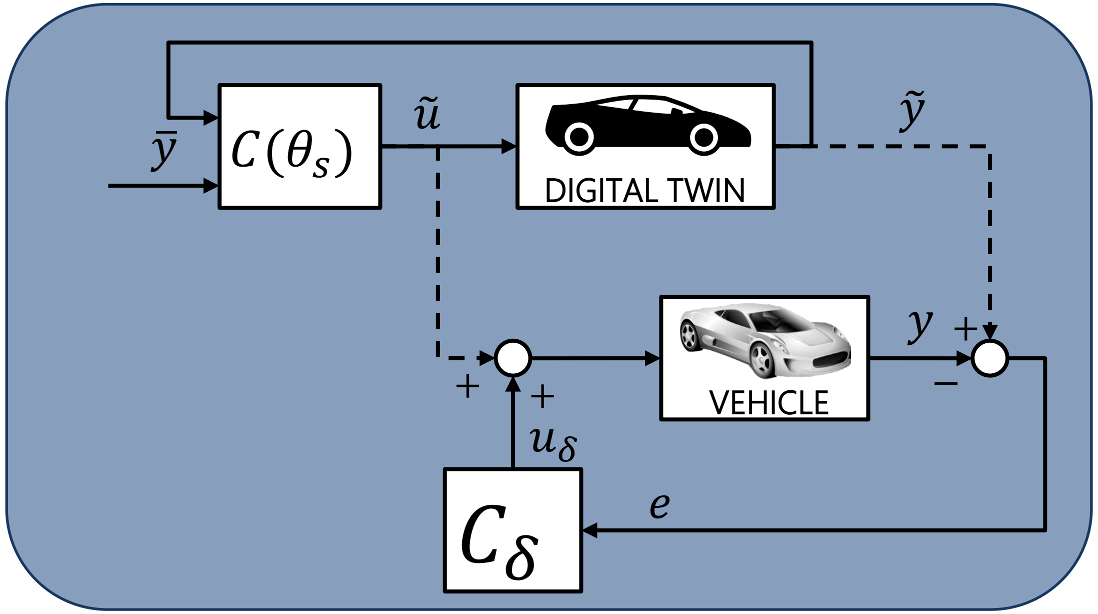
Traditional framework

- Vehicle dynamics systems development goes through **many steps**
- **Many experiments** to achieve end-of-line tuning [2]



Twin-in-the-Loop framework

- **Digital twin directly used on-board** to control the vehicle;
- **No need for controller fine tuning – simple compensator** added



Control design flow – shifting the paradigm

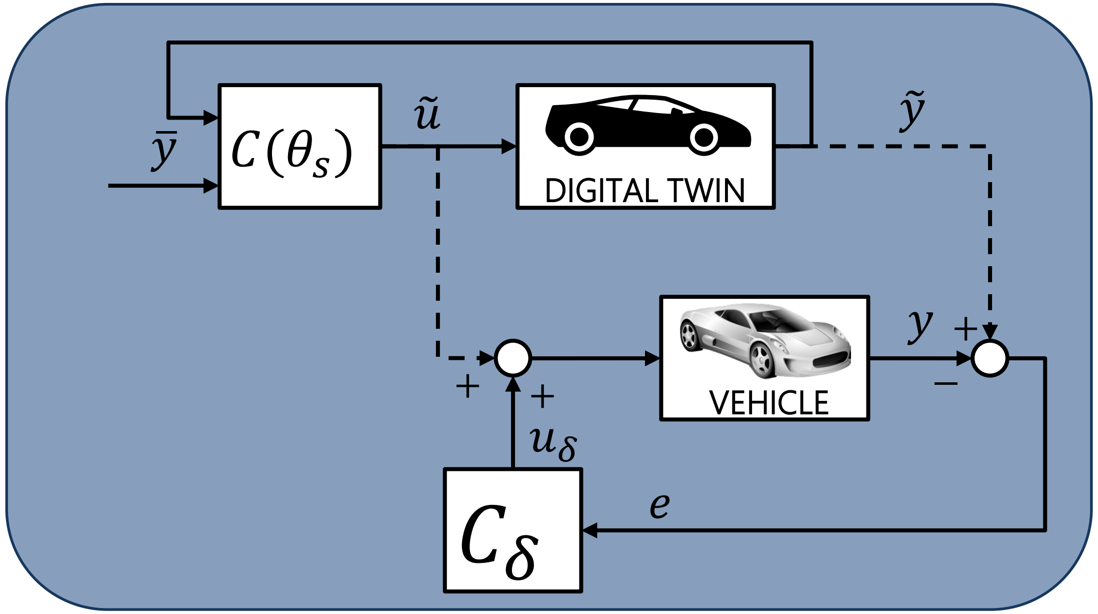
Problem statement – Twin-in-the-Loop control

Main features:

- C_δ possibly very **simple** structure;
- **High generalization capabilities**;
- **Control objectives separation** →
  - $\tilde{\boldsymbol{u}}$ nonlinear dynamics handling;
  - $\boldsymbol{u}_\delta$ noise compensation/ "small signal" control

Twin-in-the-Loop framework

- **Digital twin directly used on-board** to control the vehicle;
- **No need for controller fine tuning – simple compensator** added

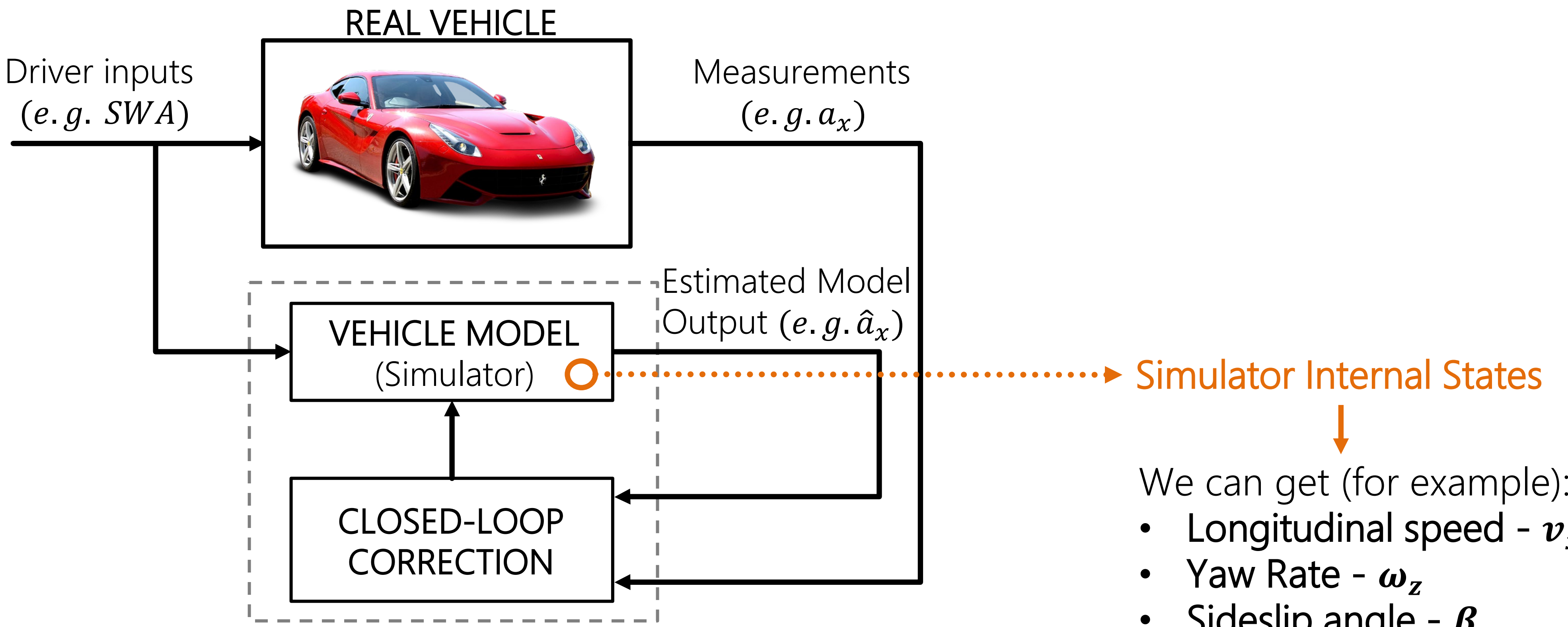


Filter design flow – shifting the paradigm

Problem statement – Twin-in-the-Loop estimation

Goal : development of a **full vehicle dynamics observer**. We are interested in:

- State Filtering
- Output Smoothing



We can get (for example):

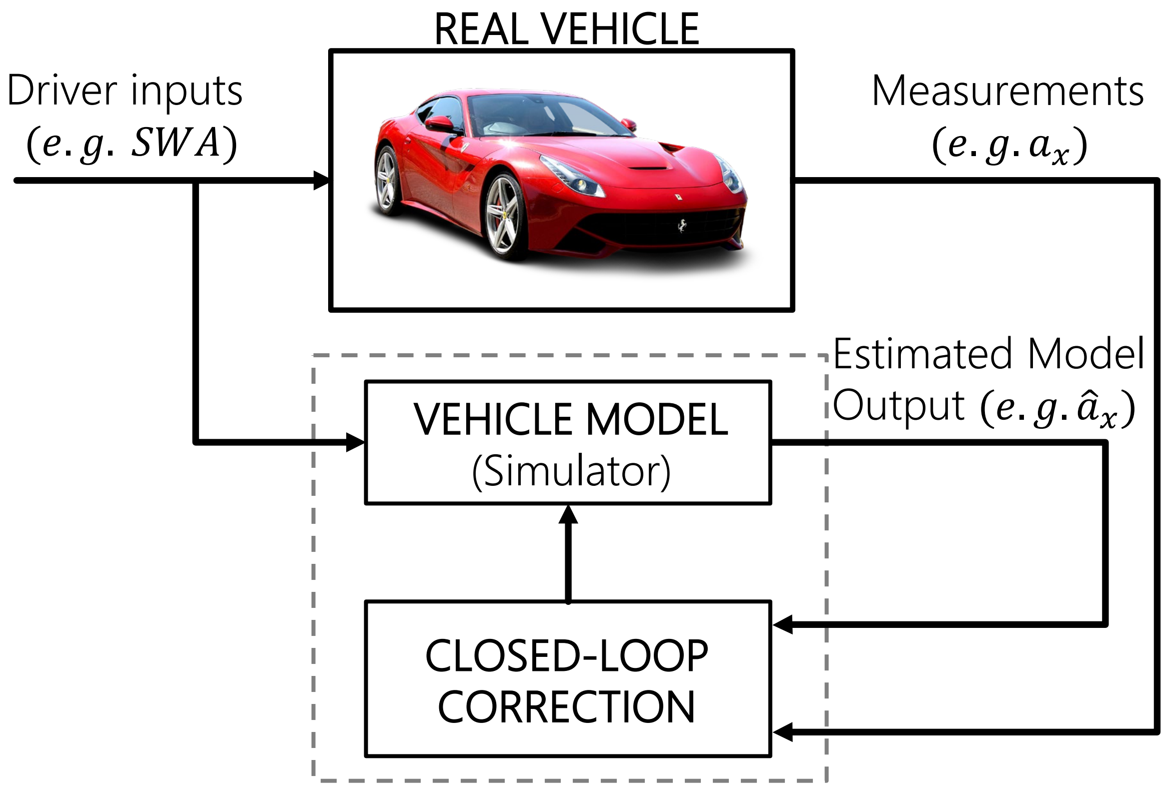
- Longitudinal speed - $\boldsymbol{v_x}$
- Yaw Rate - $\boldsymbol{\omega_z}$
- Sideslip angle - $\boldsymbol{\beta}$

Filter design flow – shifting the paradigm

Problem statement – Twin-in-the-Loop estimation

Goal : development of a **full vehicle dynamics observer**. We are interested in:

- State Filtering
- Output Smoothing



Main features:

- One model for several SW sensors
- **High generalization capabilities**;
- **Filtering objectives separation →**
  - nonlinear dynamics handling;
  - noise compensation/ "small signal" control

Outline

Problem statement

Twin-in-the-Loop control

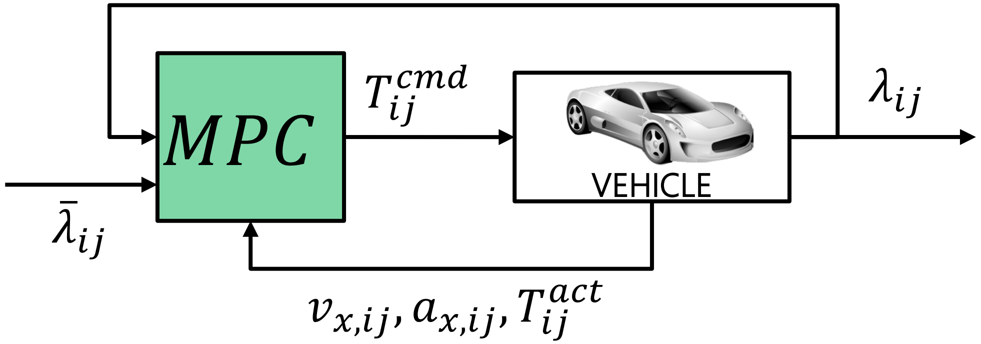
Twin-in-the-Loop estimation

Conclusions

Problem definition – model predictive controller

Benchmark controller:

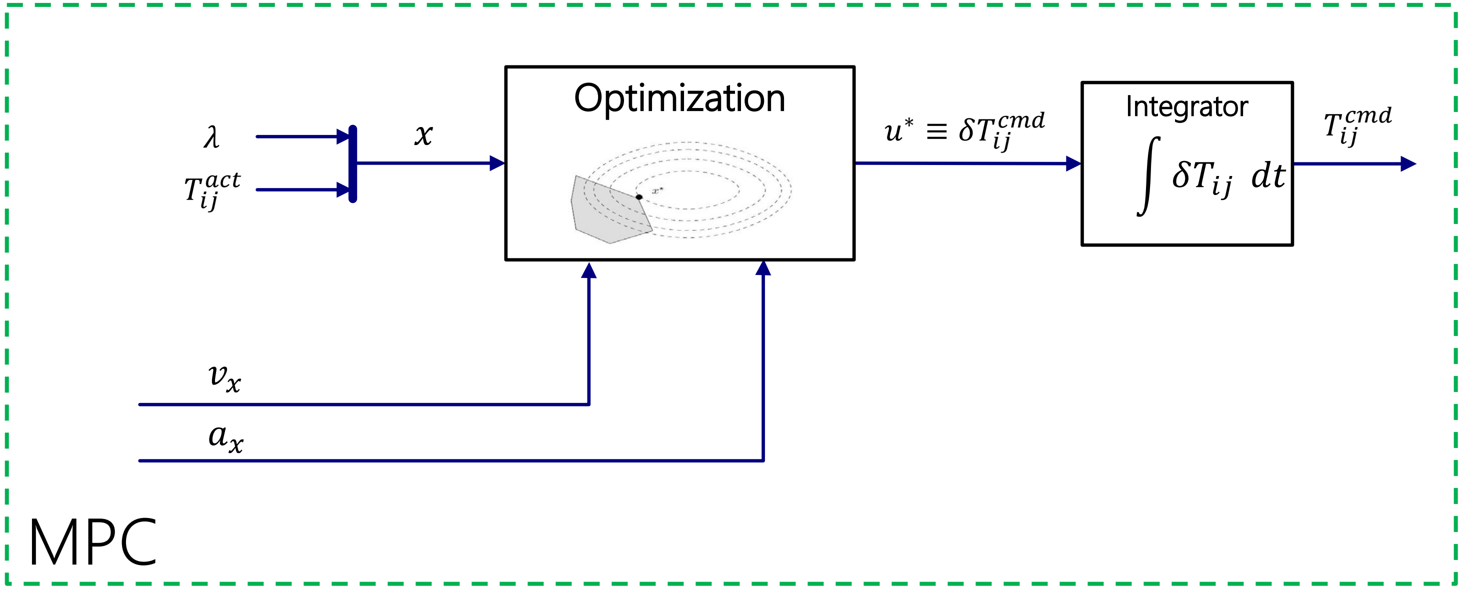
- Force-based **MPC controller**
- **Wheel-specific regulation** of the slip λ by acting on the braking torque;
- Velocity form MPC allows **integral action embedding**.



Prediction model (slip + actuator dynamics) at each wheel:

$$\dot{\lambda} = \frac{1-\lambda}{v_x} a_x + \frac{R_{wh}}{J_{wh} v_x} T_b^{act}$$

$$\dot{T}_b^{act} = -\frac{1}{\tau_{act}} \left(T_b^{act} - T_b^{cmd} \right)$$



Optimization problem (**N = 5**):

$$\min_{U} \frac{1}{2} U^T H U + f U$$

$$H = 2(\overline{W}_u + \Gamma_U^T \overline{W}_x \Gamma_U)$$

$$f = 2\left(\bar{x}_0^T \Gamma_0^T \overline{W}_x \Gamma_U - X^* \overline{W}_x \Gamma_U\right)$$

$$\underline{l} \leq A_I U \leq \overline{u}$$

$$\overline{u} = \begin{vmatrix} (\overline{T}_b - u_{-1}) I_{N\times 1} \\ \Delta T_B T_s^{mpc} + S_x \Gamma_0 \bar{x}_0 \end{vmatrix}$$

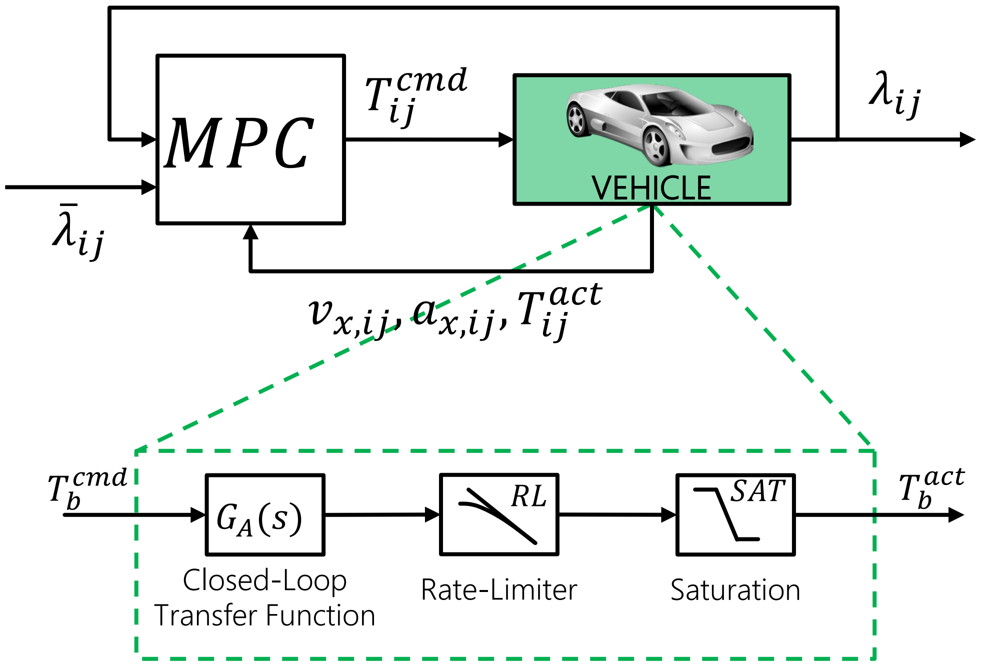
$$\underline{l} = \begin{vmatrix} -u_{-1} I_{N\times 1} \\ \Delta \underline{T_B} T_s^{mpc} - S_x \Gamma_0 \bar{x}_0 \end{vmatrix}$$

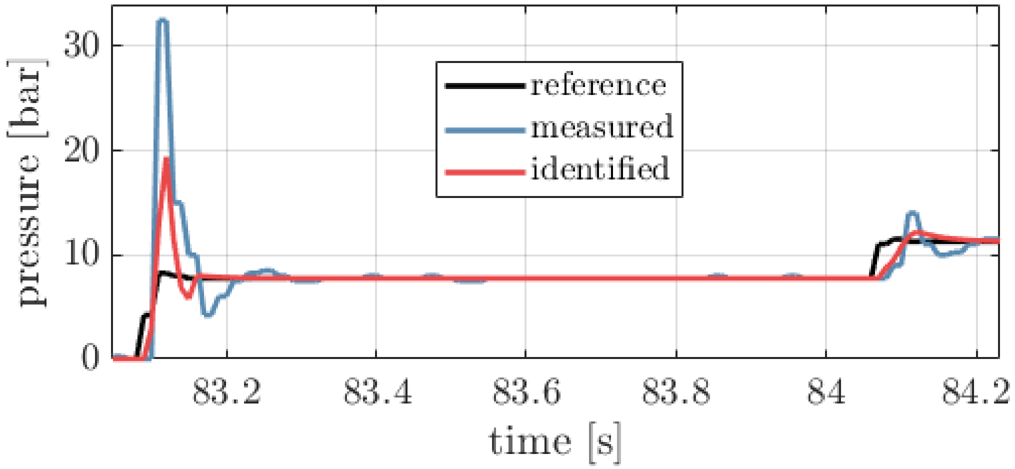
$$A_I = \begin{vmatrix} S_u \\ S_x \Gamma_u \end{vmatrix}$$

Problem definition – actuator model

Actuator model:

- Second order **transfer function** + **nonlinearities** (rate limiter and saturation);
- Identified from experimental data.

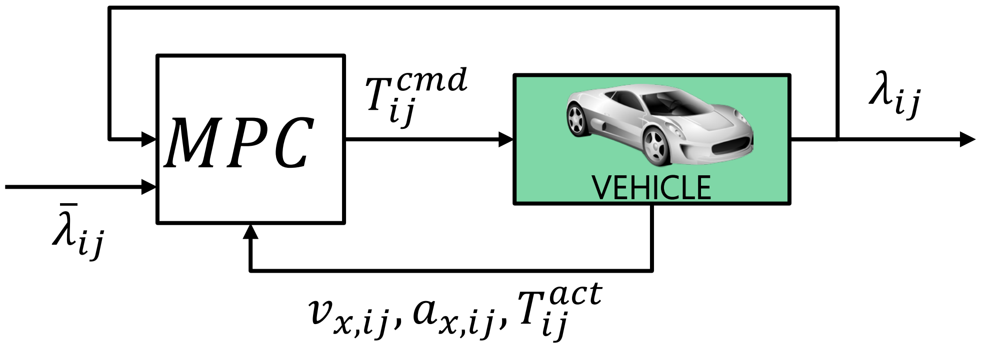


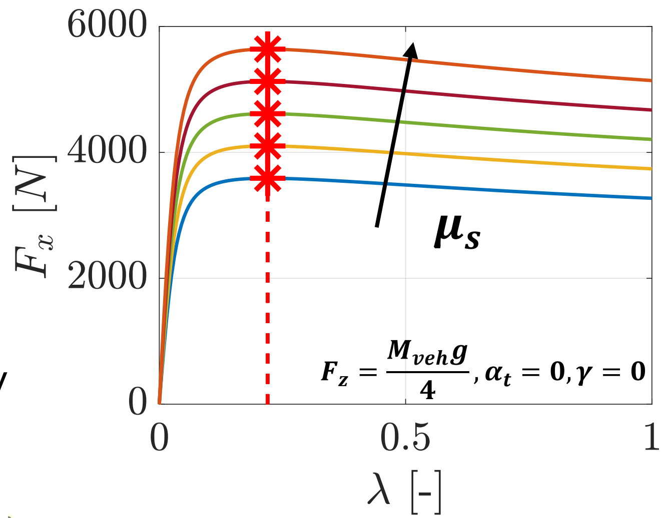


Problem definition – vehicle model

Vehicle model:

- **High-fidelity** Car-Real-Time model of Ferrari F171;
- We introduce model errors in the "real car".
  - **Friction** model perturbation;
  - Unmodeled **concentrated masses**.





$$F_x = \overbrace{\cos\left(C_{x\alpha_t} \arctan\left(B_{x\alpha_t}\alpha_t\right)\right) F_{x0}}^{\text{Pacejka model}} \cdot \overbrace{\boldsymbol{\mu_s}}^{\text{Uncertainty}}$$

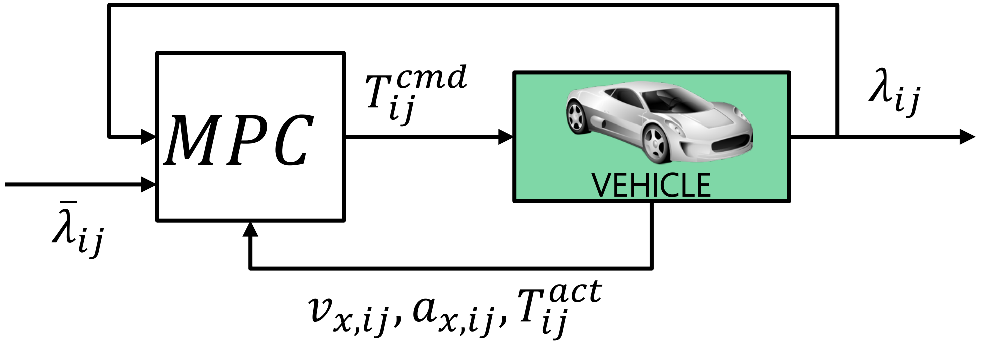
Peak friction scaling uncertainty ($\boldsymbol{\mu_s} \in [\mathbf{0.7}, \mathbf{1.1}]$)

- $\mu_s = 1 \rightarrow$ nominal curve;
- $\mu_s = 0.7 \rightarrow$ "pseudo" wet asphalt;
- $0.7 < \mu_s < 1 \rightarrow$ low grip asphalt;
- $\mu_s > 1 \rightarrow$ high grip asphalt.

Problem definition – vehicle model

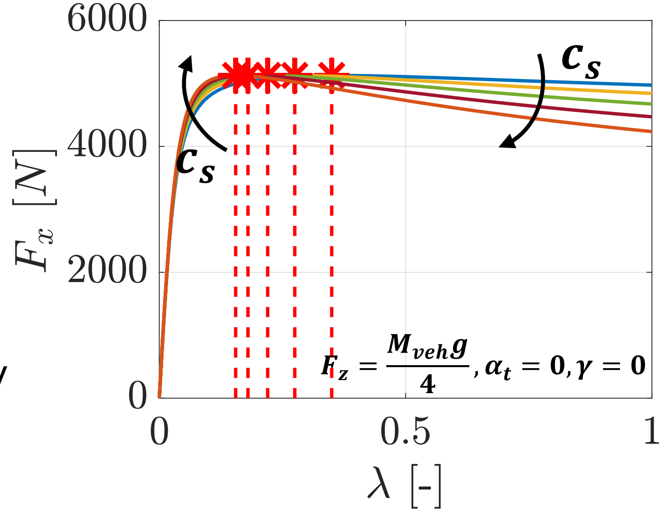
Vehicle model:

- **High-fidelity** Car-Real-Time model of Ferrari F171;
- We introduce model errors in the "real car".
  - **Friction** model perturbation;
  - Unmodeled **concentrated masses**.



$$\underbrace{\overbrace{F_x = \cos\left(C_{x\alpha_t} \arctan\left(B_{x\alpha_t}\alpha_t\right)\right) F_{x0}}^{\text{Pacejka model}} \cdot \overbrace{\boldsymbol{\mu_s}}^{\text{Uncertainty}}}$$

$$F_{x0} = D_x \sin[C_x \cdot \underbrace{\boldsymbol{c_s}}_{\text{Uncertainty}} \cdot \arctan(B_x\kappa - E_x(B_x\kappa - \arctan(B_x\kappa)))]$$



Shape factor scaling uncertainty $c_s \in [0.9, 1.1]$:

- Peak force abscissa is modified.

Problem definition – vehicle model

Vehicle model:

- **High-fidelity** Car-Real-Time model of Ferrari F171;
- We introduce model errors in the "real car".
  - **Friction** model perturbation;
  - Unmodeled **concentrated masses**.

Nominal model → driver ($\boldsymbol{m_d = 75\ kg}$)
- Total mass $\boldsymbol{M_{tot} = 1466\ kg}$

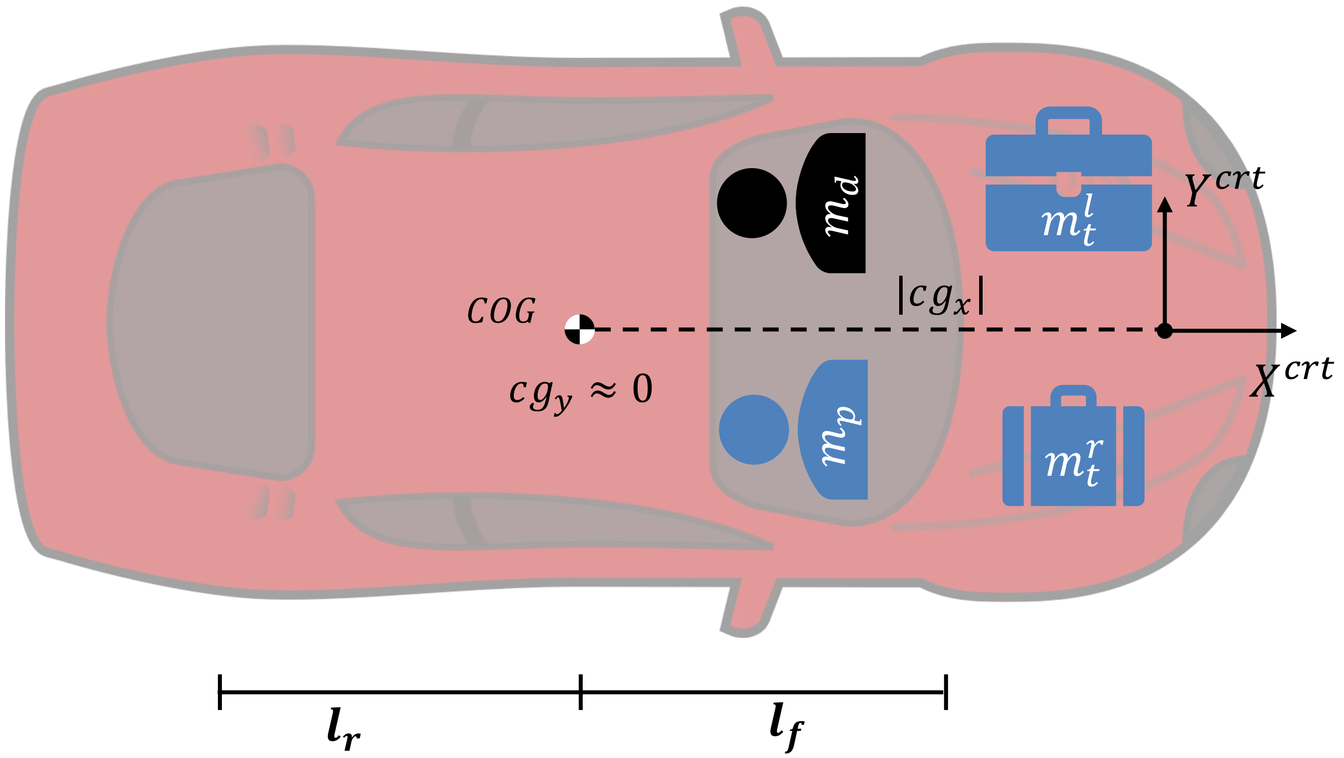
Perturbated model → passenger ($\boldsymbol{m_p = 75\ kg}$) + trunk load ($\boldsymbol{m_t^l = 90\ kg}$, $\boldsymbol{m_t^r = 30\ kg}$)
- Total mass $\boldsymbol{M_{tot} = 1661\ kg}$ $(+\boldsymbol{11}\%)$

Load transfer straight braking depends mostly on COG longitudinal position (cg_x) and vertical position (cg_z) →

Unmodeled masses in the real vehicle to **realistically** perturbate these parameters.

CRT directly accounts for inertia and COG position variations!

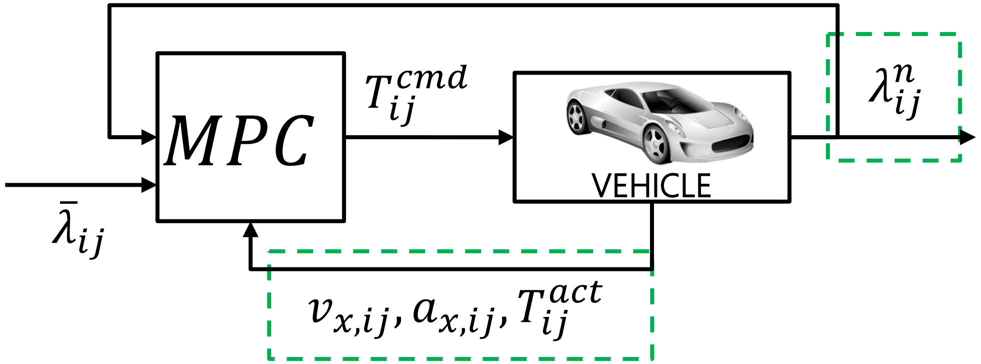
$$F_z^f = \frac{Mgl_r}{l_f + l_r} - \frac{Mh}{l_f + l_r}\dot{v}_x \qquad F_z^f = \frac{Mgl_r}{l_f + l_r} - \frac{Mh}{l_f + l_r}\dot{v}_x$$



Problem definition – sensor model and noise

Sensor/noise model:

- **MPC uses slips** and **longitudinal speed/acceleration** information;
- These signals are actually measured or estimated → disturbances introduction.



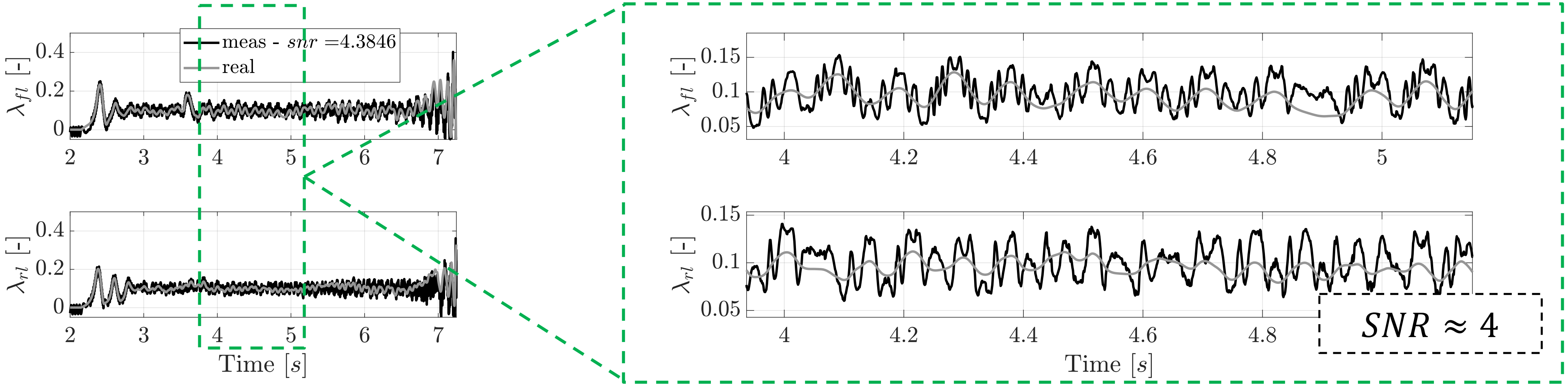
Noise characterization - slip computed from $\boldsymbol{\omega_{ij}}$ and $\boldsymbol{v_{x,ij}}$ →

- ω_{ij} measured through encoders – sinusoidal noise + random noise;
- $v_{x,ij}$ estimated through state observers – low frequency noise;

MPC also requires $\boldsymbol{a_x}$ measurements →

- a_x measured via inertial measurement units – high frequency noise.

Problem definition – sensor model and noise – mid noise



Measured quantities:

$$\omega_{ij}^n = \omega_{ij} + A_\omega(v_{x,ij}) \cdot \sin(\omega_{ij} t) + n_\omega, \qquad n_\omega \sim WN(0, \sigma_\omega^2);$$

$$v_{x,ij}^n = v_{x\,ij} + F_{lp}(n_v), \quad n_v \sim N(\mu_v, \sigma_v^2), \qquad F_{lp} = \frac{(2\pi f_1)(2\pi f_2)}{(s+2\pi f_1)(s+2\pi f_2)}$$

$$a_x^n = a_x + n_a, \qquad n_a \sim WN(0, \sigma_a^2).$$

Noise effect on slips:

$$\lambda_{ij}^n = \lambda_{ij} + n_\lambda^{ij}(v_{x,ij}^n, \omega_{ij}^n)$$

Noise characterization [+], slip computed from $\boldsymbol{\omega_{ij}}$ and $\boldsymbol{v_{x,ij}}$ →

- ω_{ij} measured through encoders – *sinusoidal noise + random noise*;
- $v_{x,ij}$ estimated through state observers – *low frequency noise*;

MPC also requires $\boldsymbol{a_x}$ measurements →

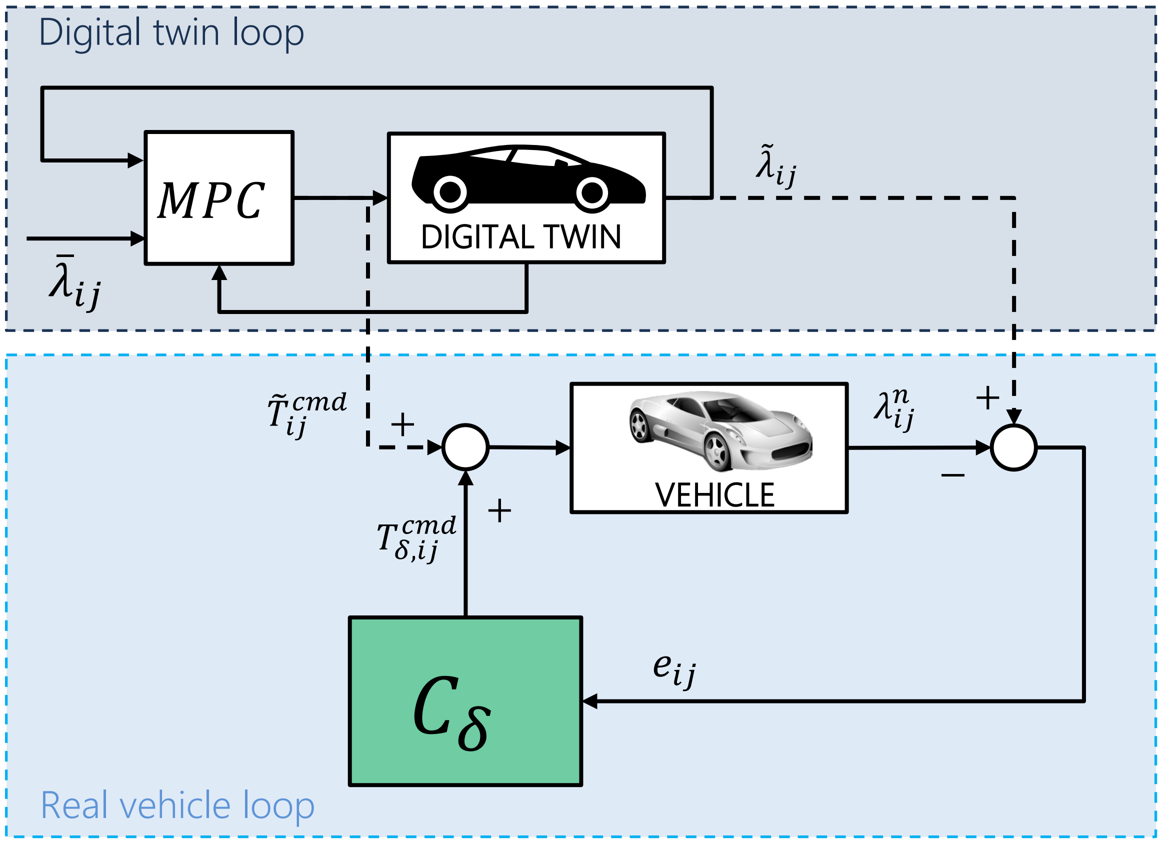
- a_x measured via inertial measurement units – *high frequency noise*.

[+] G. Panzani, M. Corno and S. M. Savaresi, "On the Periodic Noise Affecting wheel Speed Measurement", in 16th IFAC Symposium on System Identification, 2012

Active braking TiL-Control

TiL-C braking control:

- **Nonlinearity of slip dynamics** managed by **digital-twin loop;**
- **Proportional-Integral compensator $\boldsymbol{C_\delta}$** added to guarantee stability.



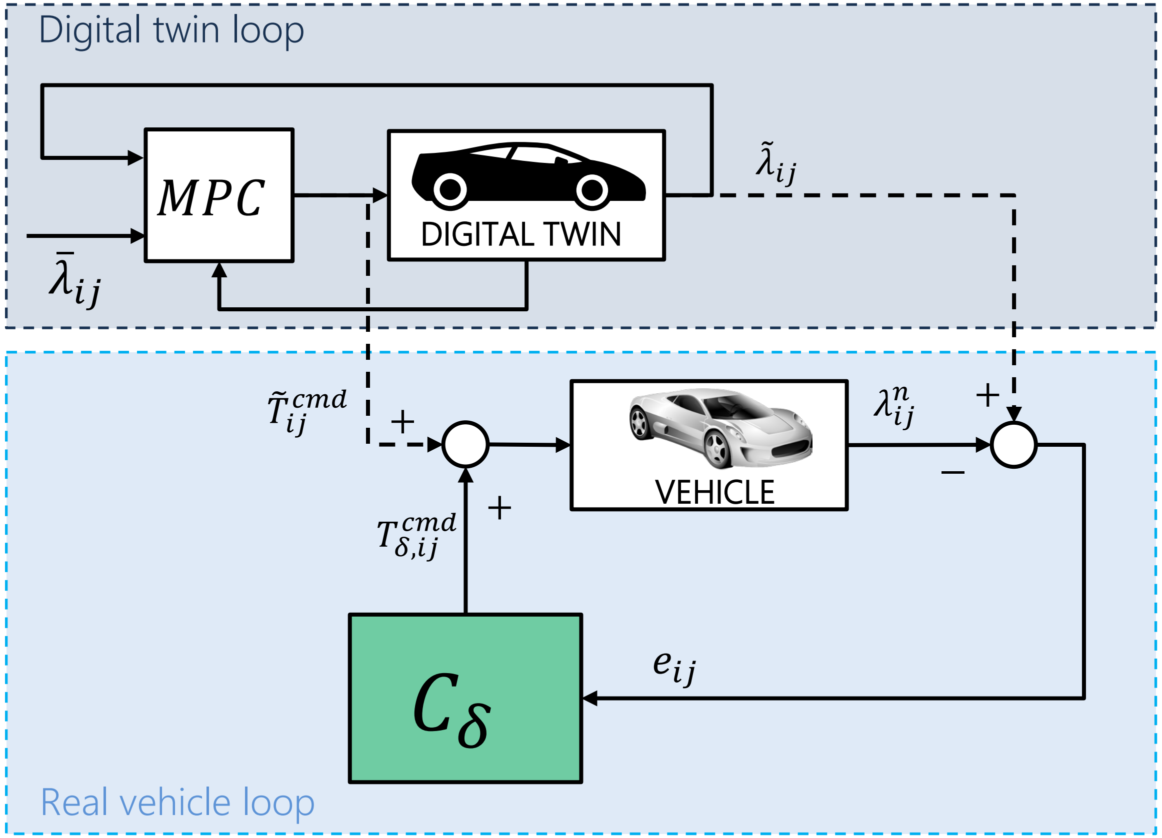
$\boldsymbol{C_\delta}$ control architecture → Scheduled PI + anti wind-up

Active braking TiL-Control

PROBLEM: $\boldsymbol{C_\delta}$ controls unknown residual between digital twin and real vehicle.
How can we tune it?

- Vehicle **tests are expensive** – need for an **efficient algorithm**;
- **Unknown underlying dynamics** – need for a **black-box algorithm.**

→ Bayesian Optimization.



$\boldsymbol{C_\delta}$ control architecture - · - · → Scheduled PI + anti wind-up

Bayesian optimization

Goal: to solve the optimization problem:

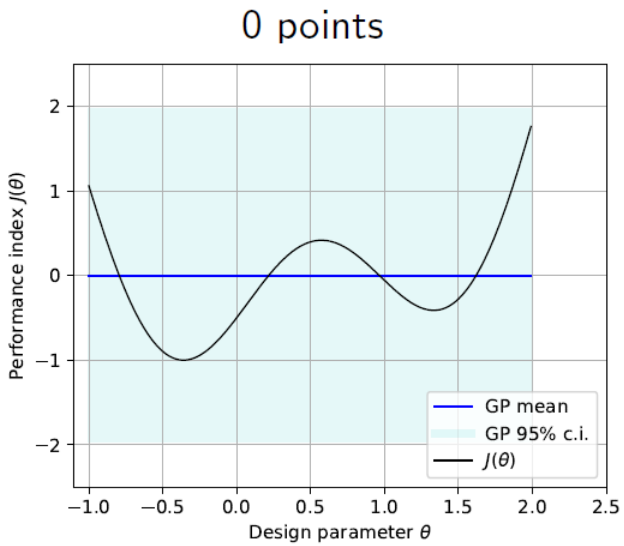
$$\min_{\theta} J(\theta)$$

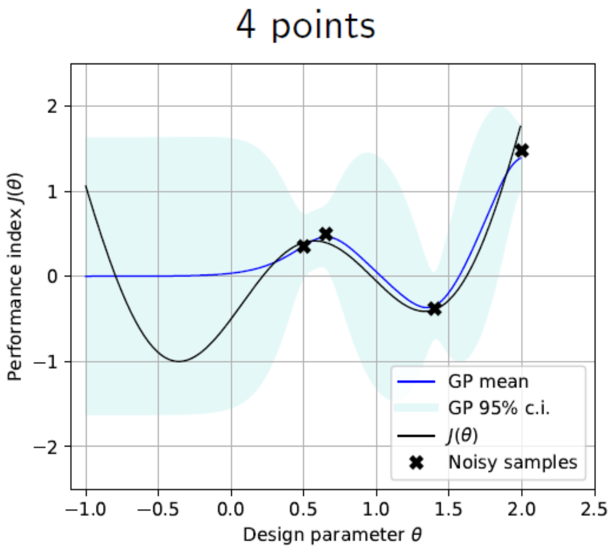
Setup: availability to collect samples of the unknown loss function J, i.e. $J(\theta_1), J(\theta_2), \ldots, J(\theta_N)$

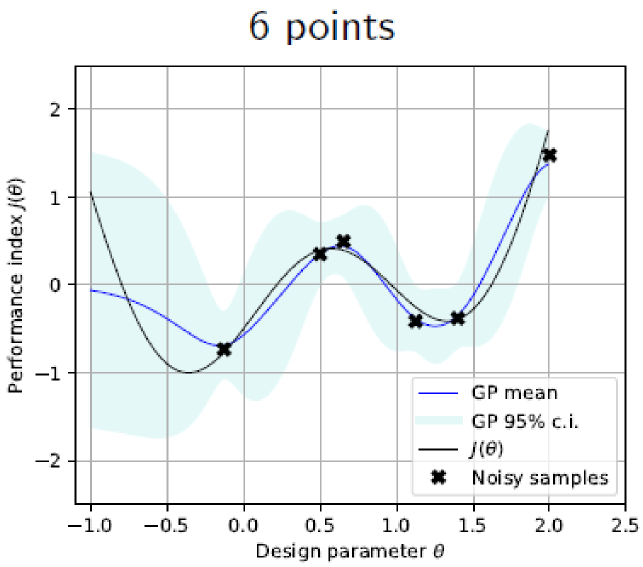
Bayesian Optimization (BO) is **not only** an **optimization algorithm**, **rather** it can be considered a **statistical learning** algorithm.

- BO iteratively updates a Bayesian surrogate model of $J(\theta)$ (Gaussian model with prior mean and covariance function)
- The measurements θ_i are **actively** selected to favor points with estimated good performance (exploitation) and/or high variance (exploration)

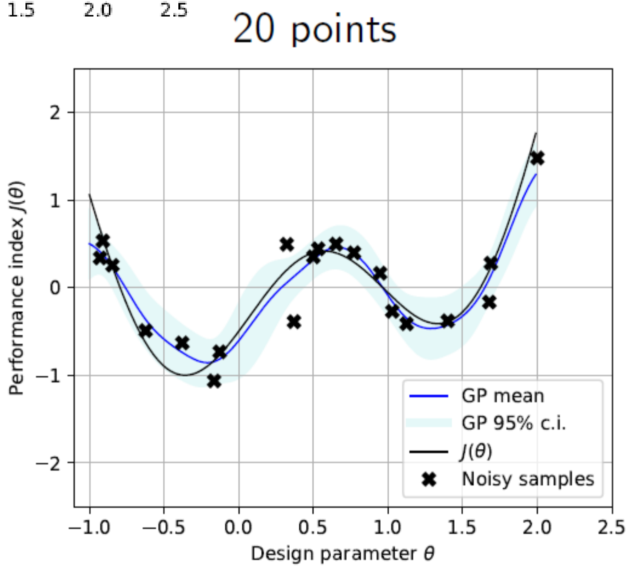
Bayesian optimization







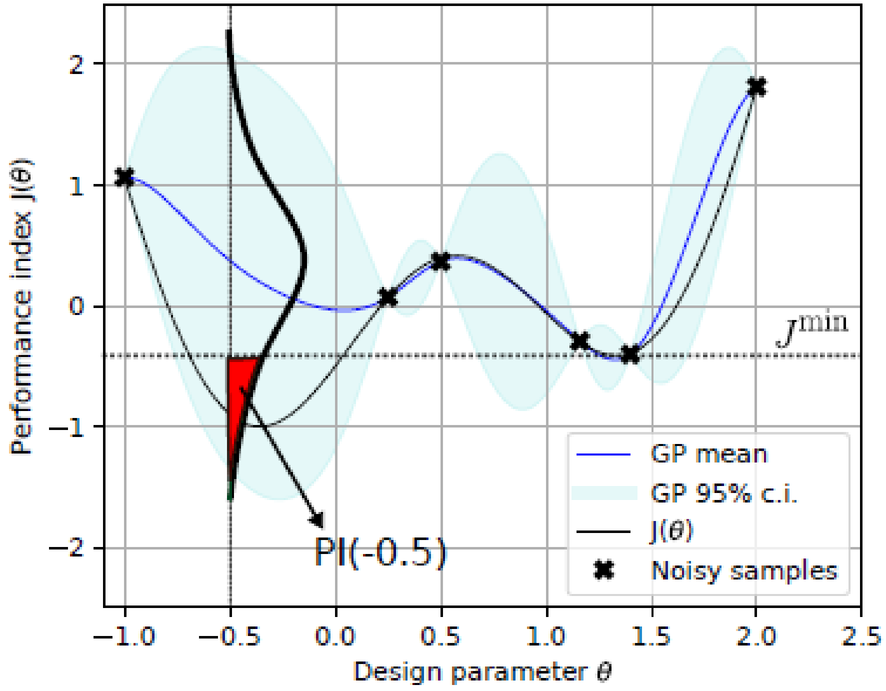
Increasing iterations…



Bayesian optimization

The GP provides the probability distribution of the function for each parameter. This probability is used to define an acquisition function, e.g.,

Probability of Improvement	Expected improvement
$A(\theta) = \mathrm{PI}(\theta) = p(J(\theta) \leq J^{\min})$	$A(\theta) = \mathrm{EI}(\theta) = \mathbb{E}[\max(0, J^{\min} - J(\theta))]$

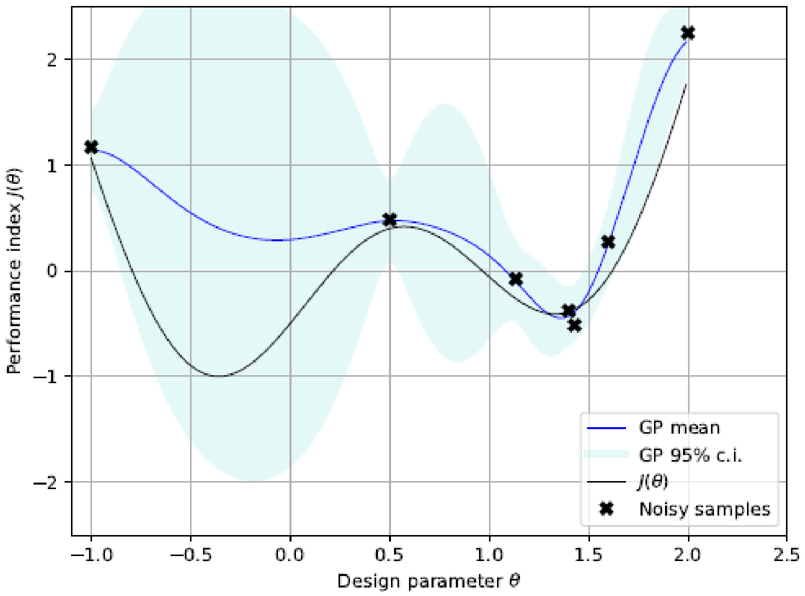


Bayesian optimization

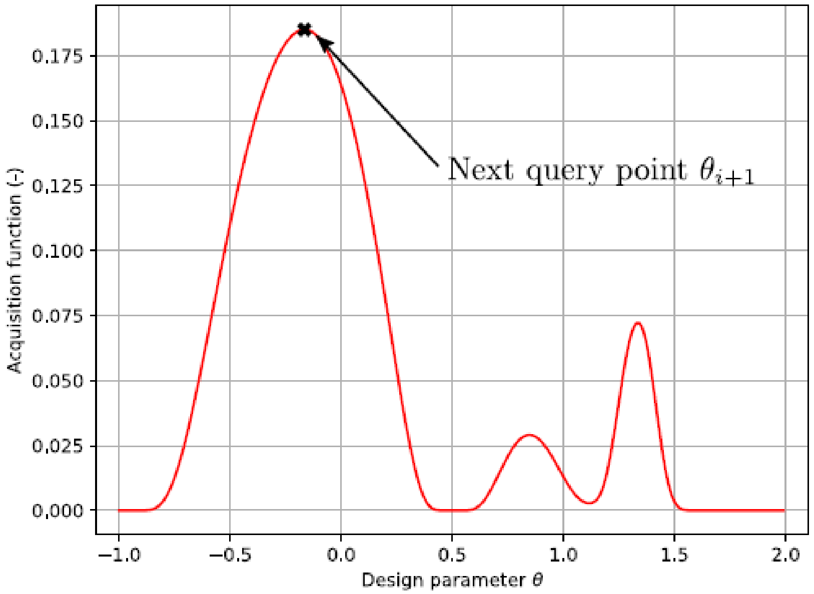
Steps of BO: for $i = 1, 2, \ldots i_{\max}$

1. **Execute** experiment with θ_i, measure $J_i = J(\theta_i) + e_i$
2. **Update** the GP model $\theta \rightarrow J(\theta)$ with (θ_i, J_i)
3. **Construct** acquisition function $A(\theta)$
4. **Maximize** $A(\theta)$ to obtain next query point θ_{i+1}

GP at iteration i



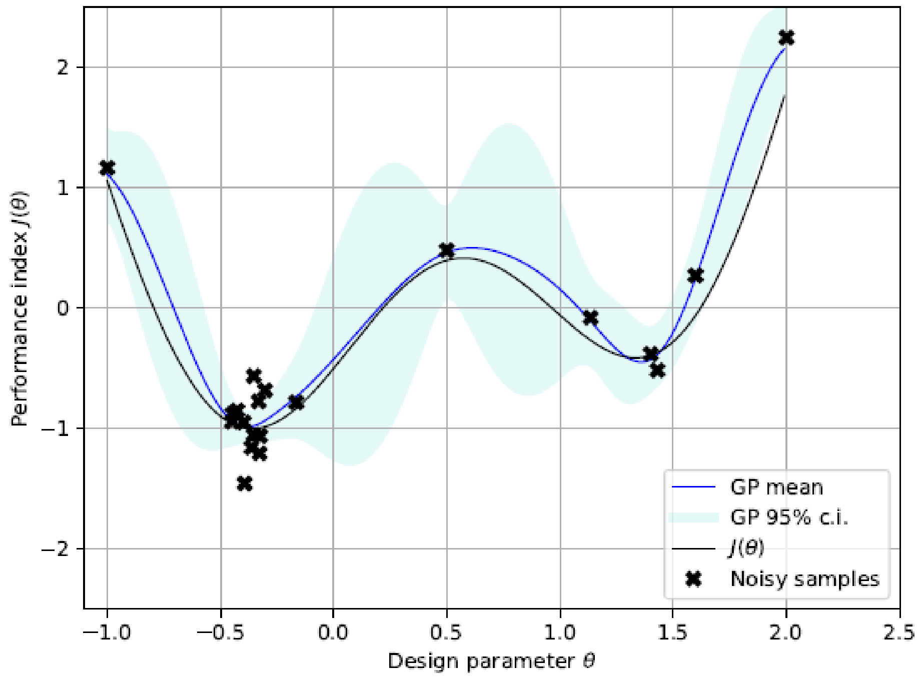
$A(\theta)$ at iteration i



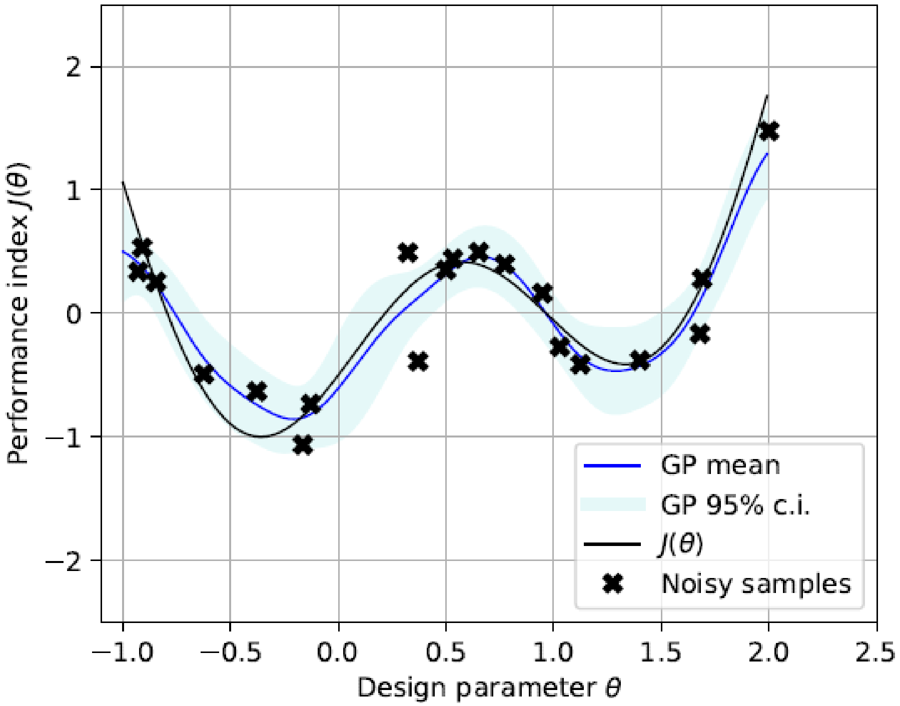
Bayesian optimization

iteration 20

Bayesian Optimization



Random sampling



Controller tuning setup

Simulator (ideal case):

- Noiseless signals;
- Perfect model knowledge.

Virtual closed-loop experiments

MPC tuning (ideal)

$\tilde{\lambda}$ - reference behaviour

Controller tuning setup

Simulator (ideal case):

- Noiseless signals;
- Perfect model knowledge.

Real vehicle:

- Noisy signals;
- Perturbated model (e.g. additional masses).

Virtual closed-loop experiments

MPC tuning (ideal)

$\tilde{\lambda}$ - reference behaviour

Controller tuning setup

Simulator (ideal case):

- Noiseless signals;
- Perfect model knowledge.

Real vehicle:

- Noisy signals;
- Perturbated model (e.g. additional masses).

Virtual closed-loop experiments

Real closed-loop experiments

MPC tuning (ideal)

$\tilde{\lambda}$ - **reference** behaviour → used in optimizer metrics

TiL-C tuning

MPC tuning (real)

MPC tuning

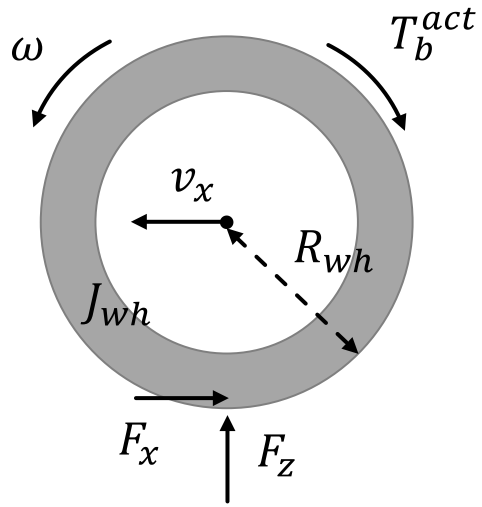
Closed-loop experiments on real vehicle → MPC prediction model fine tuning

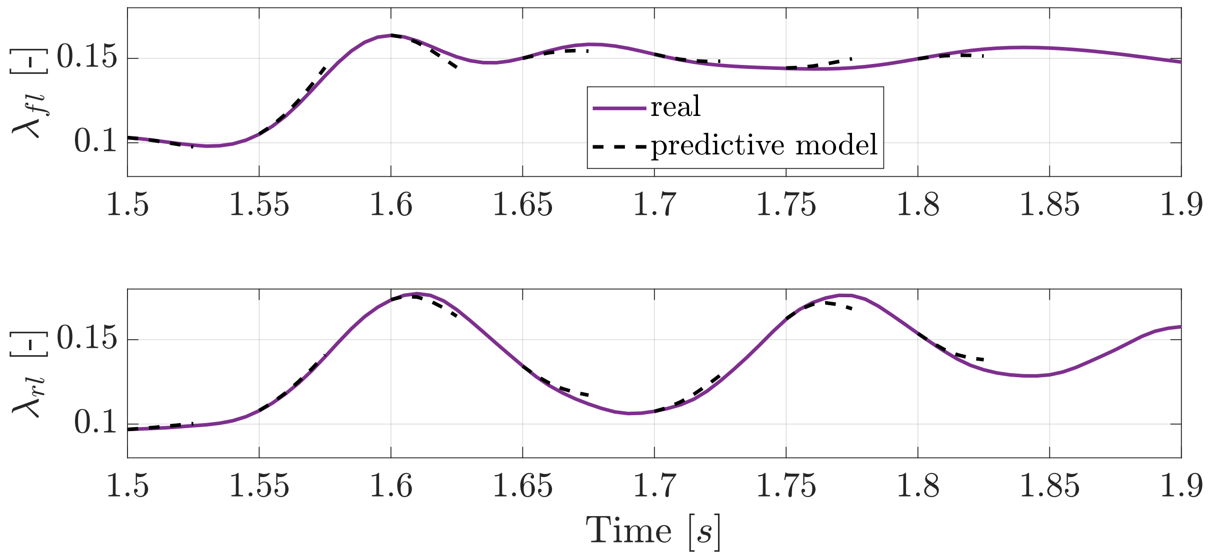
$$\dot{\lambda} = \frac{1-\lambda}{v_x} a_x + \frac{R_{wh}}{J_{wh} v_x} T_b^{act} \text{ (front/rear wheels)}$$

$$\theta_{mpc} = \begin{bmatrix} J_f & J_r & R_{wh}^f & R_{wh}^r \end{bmatrix}$$

Prediction error minimization:

$$\min_{\theta_{mpc}} \boldsymbol{J_{pred}} = \sum_{\boldsymbol{i=fl,fr,rl,rr}} \frac{\boldsymbol{rms}\left(\boldsymbol{\lambda_i^{mpc}} - \boldsymbol{\lambda_i}\right)}{\boldsymbol{4}}$$





TiL tuning

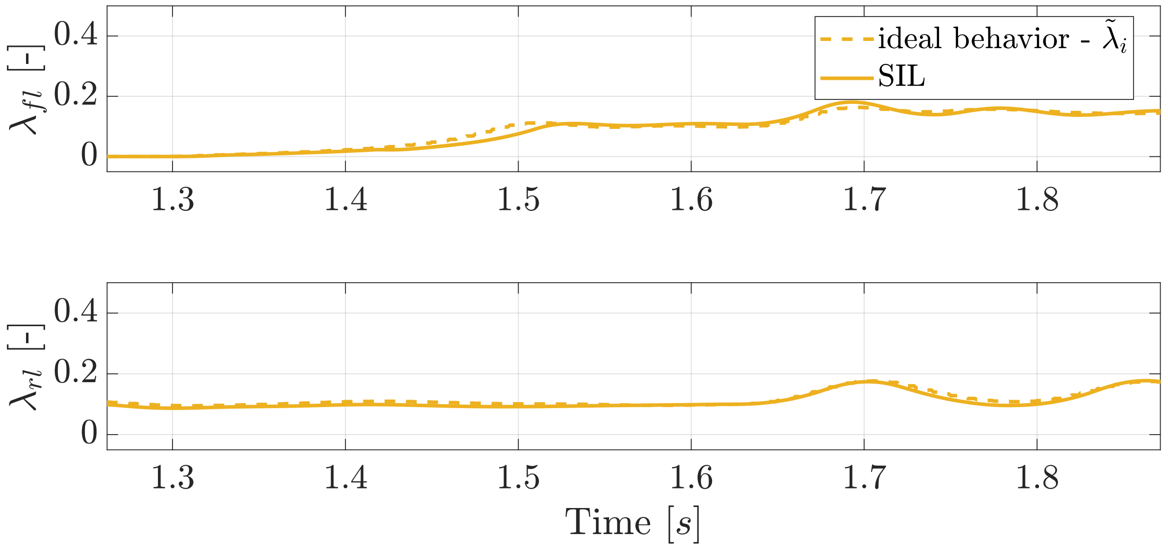
Closed-loop experiments on real vehicle → TiL-C parameters tuning

$$C_\delta(z) = \frac{k_p}{1+2T_i/T_s} \cdot \frac{z+1}{z+\frac{T_s-2T_i}{T_s+2T_i}}, \quad \text{(front/rear wheels)}$$

$$\theta_{\mathrm{TiL}} = \begin{bmatrix} k_p^f & k_p^r & T_i^f & T_i^r \end{bmatrix}$$

Reference tracking error minimization:

$$\min_{\theta_{TiL}} \boldsymbol{J_\lambda} = \sum_{\boldsymbol{i=fl,fr,rl,rr}} \frac{\boldsymbol{rms(\tilde{\lambda}_i - \lambda_i)}}{\boldsymbol{4}}$$



Controller tuning setup – nominal approach

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\mathbf{argmin}} \quad \boldsymbol{J}(\boldsymbol{\theta}, \boldsymbol{\Delta})$$

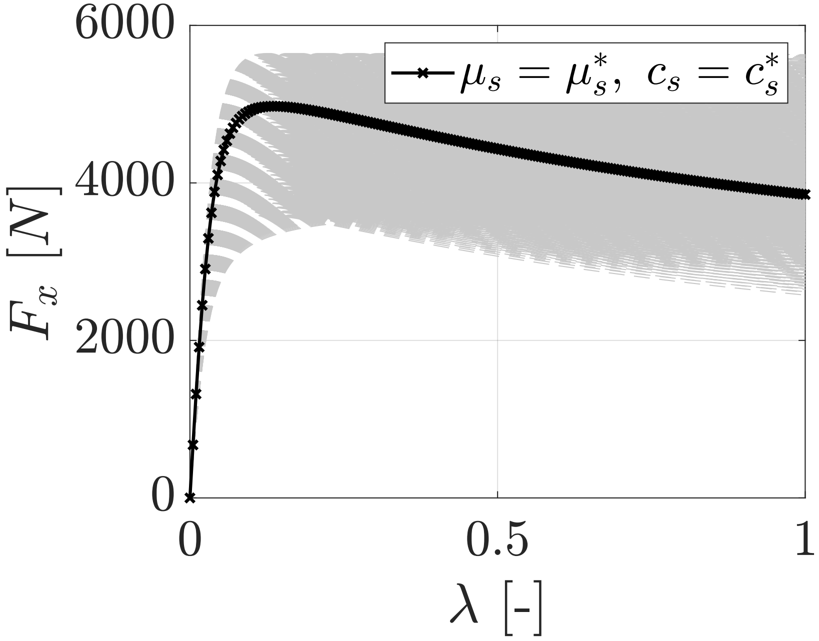
Nominal optimization/tuning approach:

- High performance on $\Delta = \Delta^*$ BUT lower robustness.

Tire model uncertainty $\Delta \rightarrow$

- $F_x = F_x(\lambda, F_z, \alpha_t, \gamma, \boldsymbol{\Delta})$;
- $\Delta = [\mu_s, c_s] \in \mathbb{R}^2$;
- $\mu_s \sim N(\mu_{\mu_s}, \sigma^2_{\mu_s}),\ c_s \sim N(\mu_{\mu_s}, \sigma^2_{\mu_s})$;
- $\Delta^{(1)}, \dots, \Delta^{(M)} \rightarrow$ uncertainty realizations.

Friction curve ensemble



Controller tuning setup – robust approach

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\mathbf{argmin}} \quad \boldsymbol{J}(\boldsymbol{\theta}, \boldsymbol{\Delta})$$

Nominal optimization/tuning approach:

- High performance on $\Delta = \Delta^*$ BUT lower robustness.

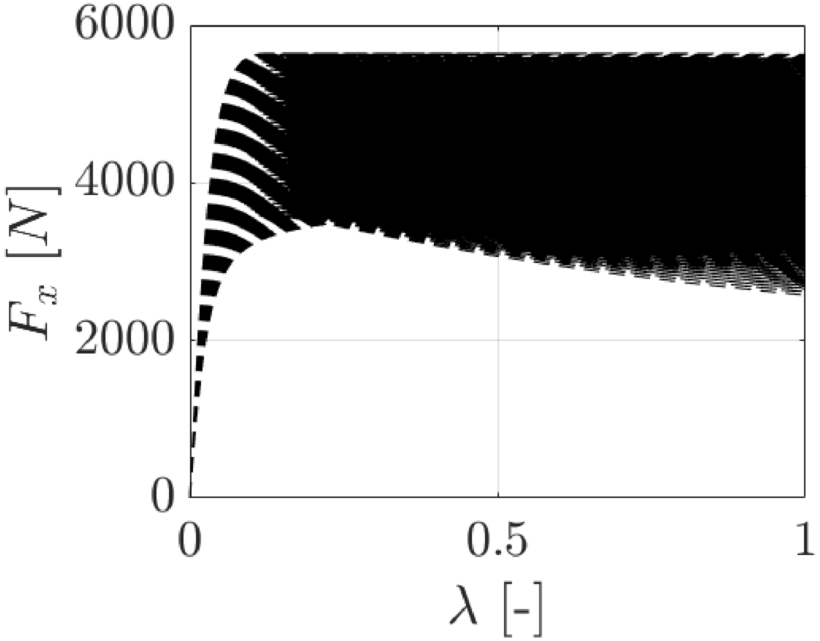
Robust optimization/tuning approach:

- More conservative on $\Delta = \Delta^*$ BUT higher robustness;
- Can we achieve probabilistic robustness guarantees?

Tire model uncertainty $\Delta \rightarrow$

- $F_x = F_x(\lambda, F_z, \alpha_t, \gamma, \boldsymbol{\Delta})$;
- $\Delta = [\mu_s, c_s] \in \mathbb{R}^2$;
- $\mu_s \sim N\left(\mu_{\mu_s}, \sigma^2_{\mu_s}\right)$, $c_s \sim N(\mu_{\mu_s}, \sigma^2_{\mu_s})$;
- $\Delta^{(1)}, \dots, \Delta^{(M)} \rightarrow$ uncertainty realizations.

Friction curve ensemble



Controller tuning setup – robust approach

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\mathbf{argmin}} \quad \boldsymbol{\gamma_{est}}(\boldsymbol{\theta}, \boldsymbol{\Delta}^{(\mathbf{1})}, \dots, \boldsymbol{\Delta}^{(\boldsymbol{M})})$$

subject to:

$\gamma_{est} \geq J\left(\theta, \Delta^{(j)}\right), \quad j = 1, \dots, M$ (chance constraints)

Tire model uncertainty $\Delta \rightarrow$

- $F_x = F_x(\lambda, F_z, \alpha_t, \gamma, \boldsymbol{\Delta})$;
- $\Delta = [\mu_s, c_s] \in \mathbb{R}^2$;
- $\mu_s \sim N\left(\mu_{\mu_s}, \sigma_{\mu_s}^2\right)$, $c_s \sim N(\mu_{\mu_s}, \sigma_{\mu_s}^2)$;
- $\Delta^{(1)}, \dots, \Delta^{(M)} \rightarrow$ uncertainty realizations.

Randomized analysis for probabilistic worst-case performance (RAWC) [#]:

Consider probability levels $\mathrm{p}^* \in (0,1)$ and $\delta \in (0,1) \rightarrow$

- $\Pr\{\Pr\{J(\Delta) \leq \gamma_{est}\} \geq p^*\} \geq 1 - \delta$;
- γ_{est} constructed on M realizations of Δ;
- A possibility is $\gamma_{est} = \max\limits_{i=1,\dots,M} J(\theta, \Delta^{(i)})$, and $M = \dfrac{\log(\delta^{-1})}{\log(p^{*-1})}$ (log-over-log bound).

[#] R. Tempo, E. W. Bai and F. Dabbene, Probabilistic robustness analysis: explicit bounds for the minimum number of samples, *Proceedings of 35th IEEE Conference on Decision and Control*, 1996.

Controller tuning setup – robust approach

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\mathbf{argmin}} \quad \boldsymbol{\gamma_{est}}(\boldsymbol{\theta}, \boldsymbol{\Delta}^{(1)}, \ldots, \boldsymbol{\Delta}^{(M)})$$

subject to:

$\gamma_{est} \geq J\left(\theta, \Delta^{(j)}\right), \quad j = 1, \ldots, M$ (chance constraints)

Tire model uncertainty $\Delta \rightarrow$

- $F_x = F_x(\lambda, F_z, \alpha_t, \gamma, \boldsymbol{\Delta})$;
- $\Delta = [\mu_s, c_s] \in \mathbb{R}^2$;
- $\mu_s \sim N\left(\mu_{\mu_s}, \sigma_{\mu_s}^2\right), \; c_s \sim N(\mu_{\mu_s}, \sigma_{\mu_s}^2)$;
- $\Delta^{(1)}, \ldots, \Delta^{(M)} \rightarrow$ uncertainty realizations.

Remarks:

- In practice, we are optimizing against the **probabilistically-guaranteed worst-case scenario**.
- Once $\theta_{\mathcal{B}}$ is found, we can **test its robustness** against a **new realization** of Δ.

Randomized analysis for probabilistic worst-case performance (RAWC) [#]:

Consider probability levels $\mathrm{p}^* \in (0,1)$ and $\delta \in (0,1) \rightarrow$

- $\Pr\{\Pr\{J(\Delta) \leq \gamma_{est}\} \geq p^*\} \geq 1 - \delta$;
- γ_{est} constructed on M realizations of Δ;
- A possibility is $\gamma_{est} = \max_{i=1,\ldots,M} J(\theta, \Delta^{(i)})$, and $M = \dfrac{\log(\delta^{-1})}{\log(p^{*-1})}$ (log-over-log bound).

[#] R. Tempo, E. W. Bai and F. Dabbene, **Probabilistic robustness analysis: explicit bounds for the minimum number of samples**, *Proceedings of 35th IEEE Conference on Decision and Control*, 1996.

Test definition

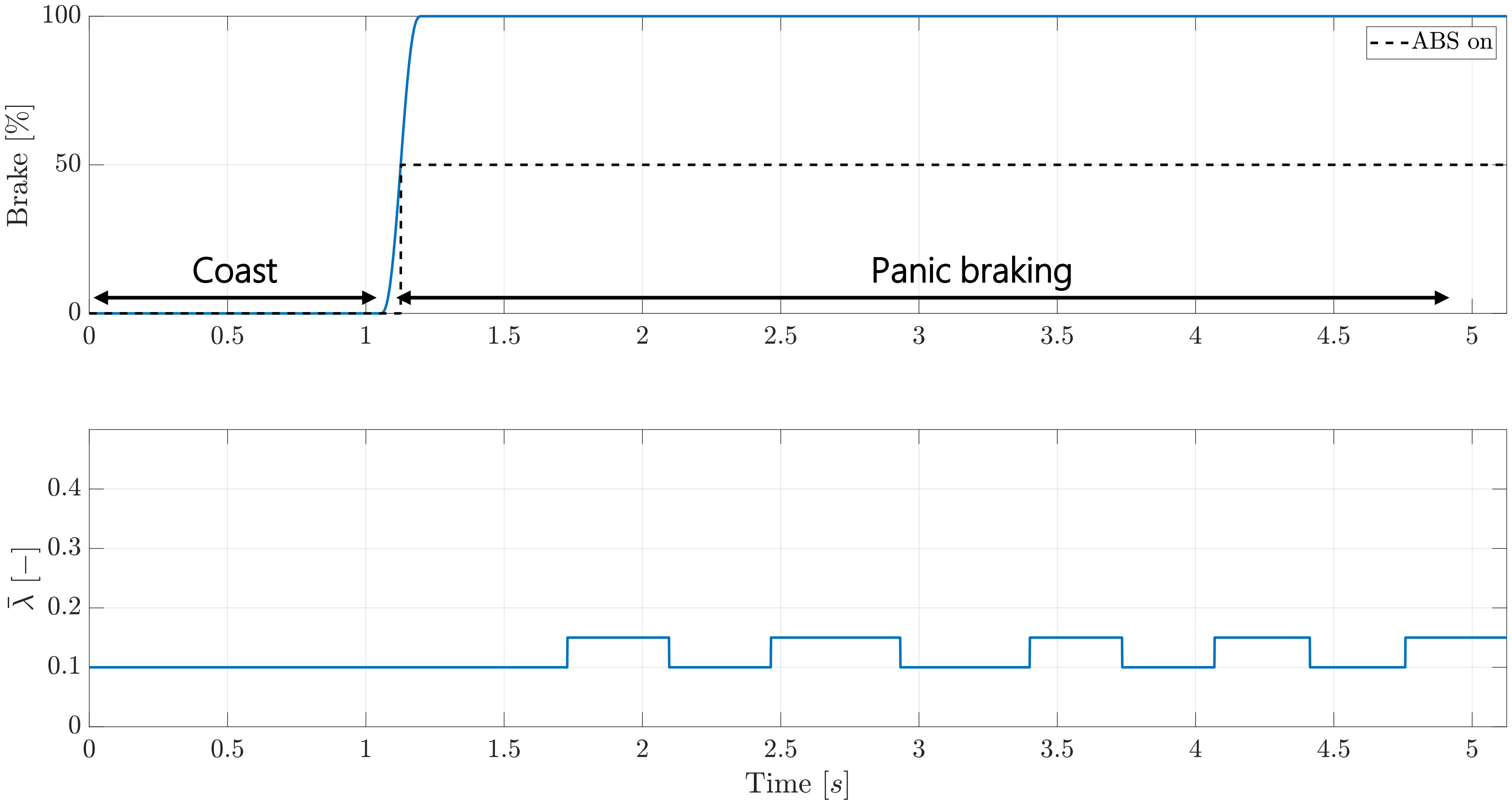
Optimization experiment:

- Straight braking maneuver ($SWA = 0\ deg$):
  1. Coasting down;
  2. Panic braking (full brake).

- Pulse wave reference – to better excite slip dynamics.

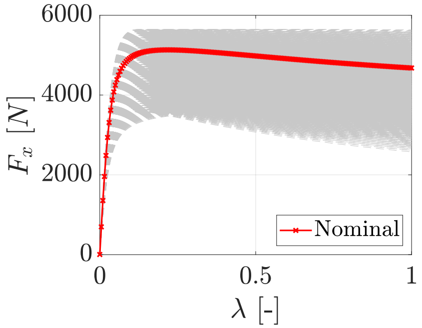
Test experiment:

- Constant slip reference is considered.

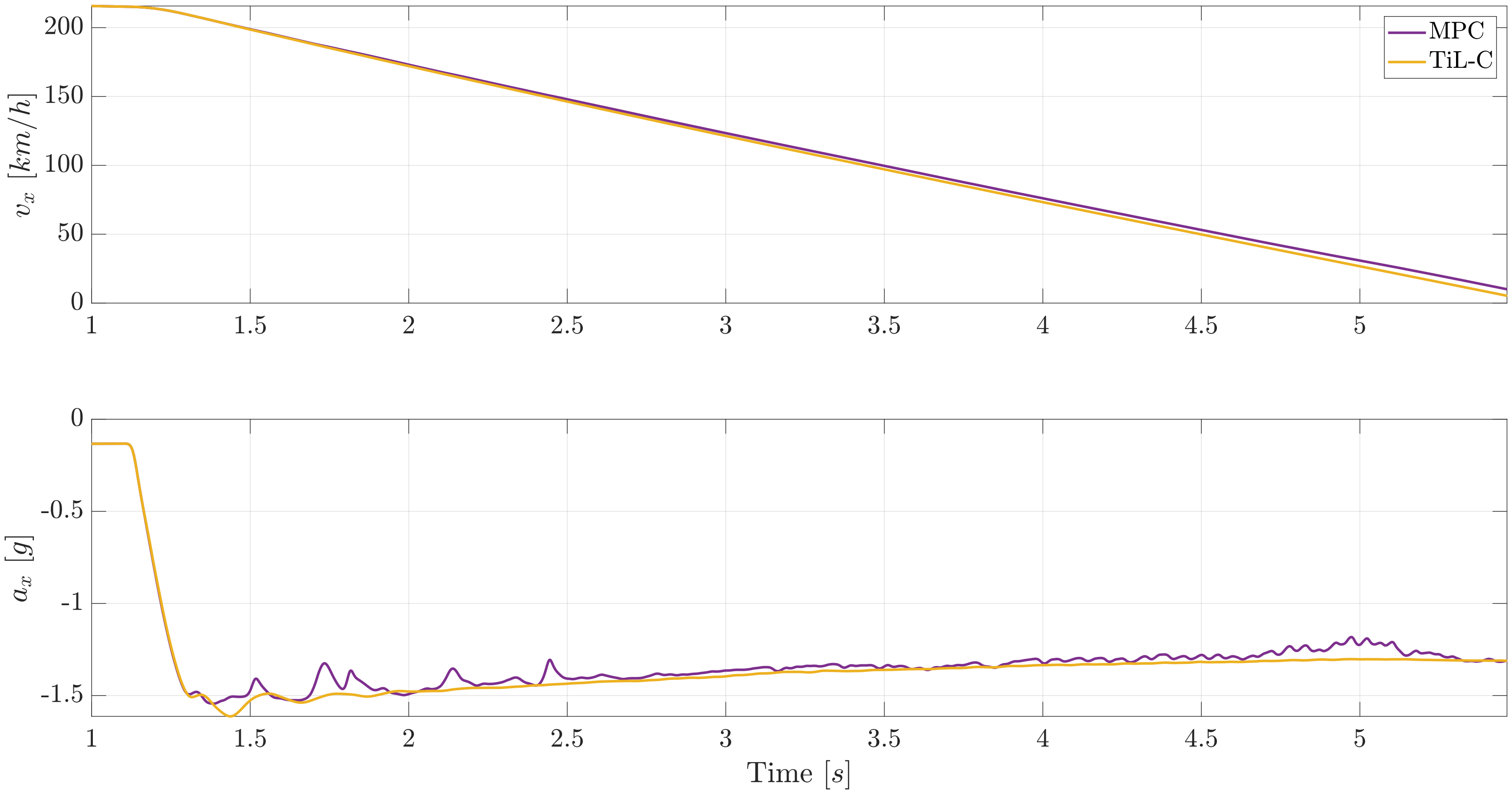


Nominal vehicle/noisy data – testing experiment

Speed/acceleration (TiL vs bench)



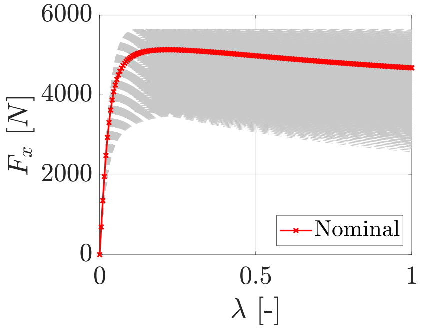
$\boldsymbol{SNR \approx 4}$



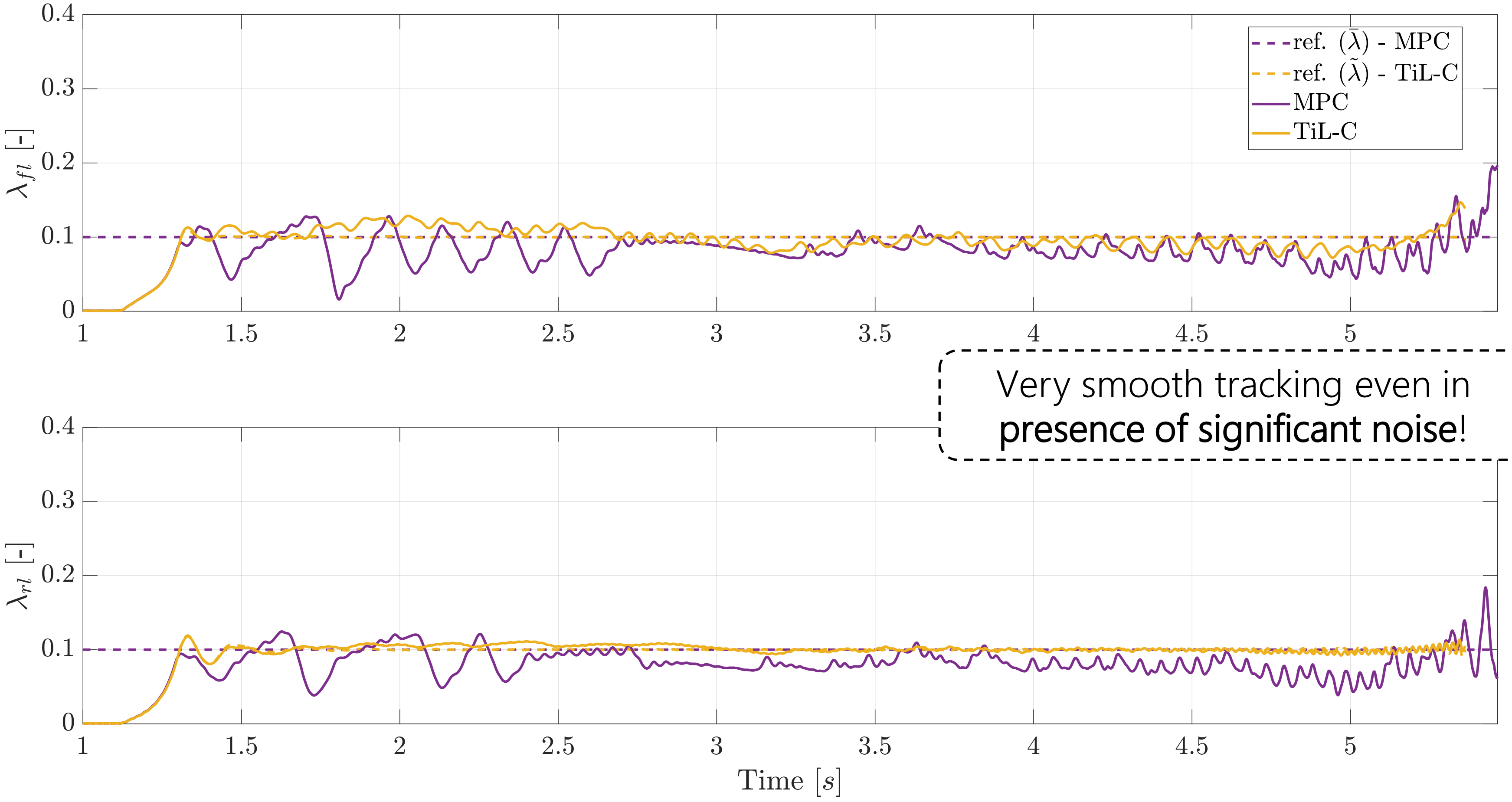
Nominal vehicle/noisy data – testing experiment

Slips (TiL vs bench)

	J_λ [%]
TiL	1.02
Bench	2.77
	+170%



$SNR \approx 4$

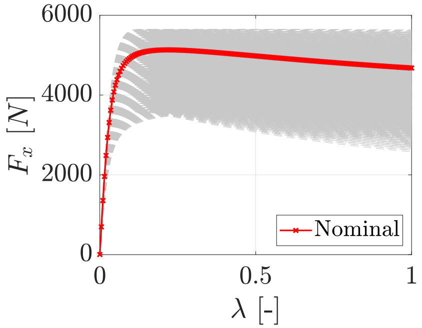


Very smooth tracking even in presence of significant noise!

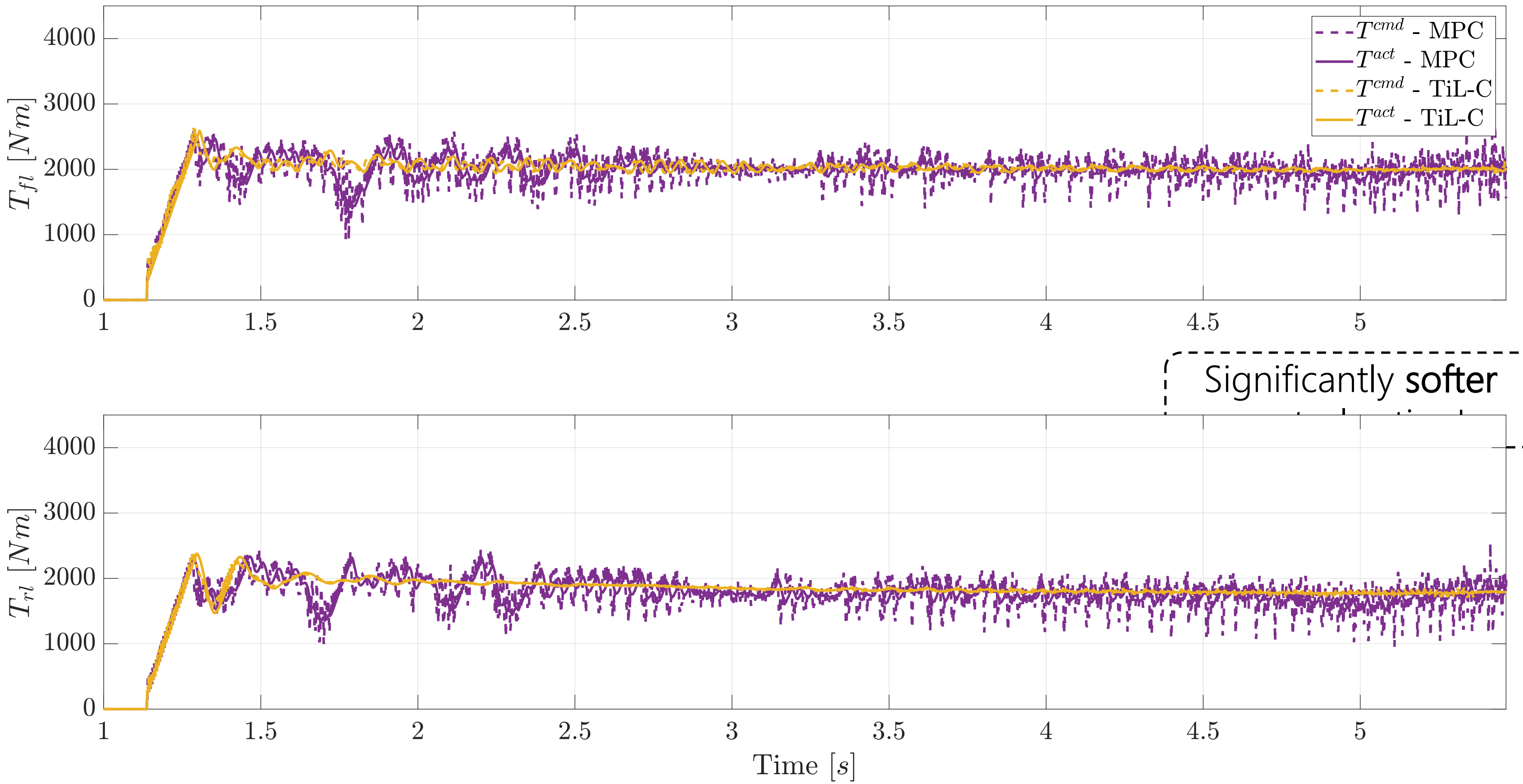
Nominal vehicle/noisy data – testing experiment

Commanded torques (TiL vs bench)

	J_λ [%]	$J_u \left[\frac{Nm}{s}\right]$
TiL	1.02	5.57
Bench	2.77	10.25
	+170%	+84%



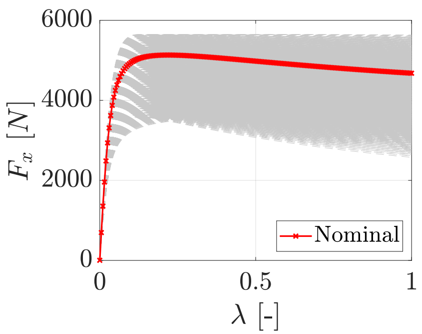
$SNR \approx 4$



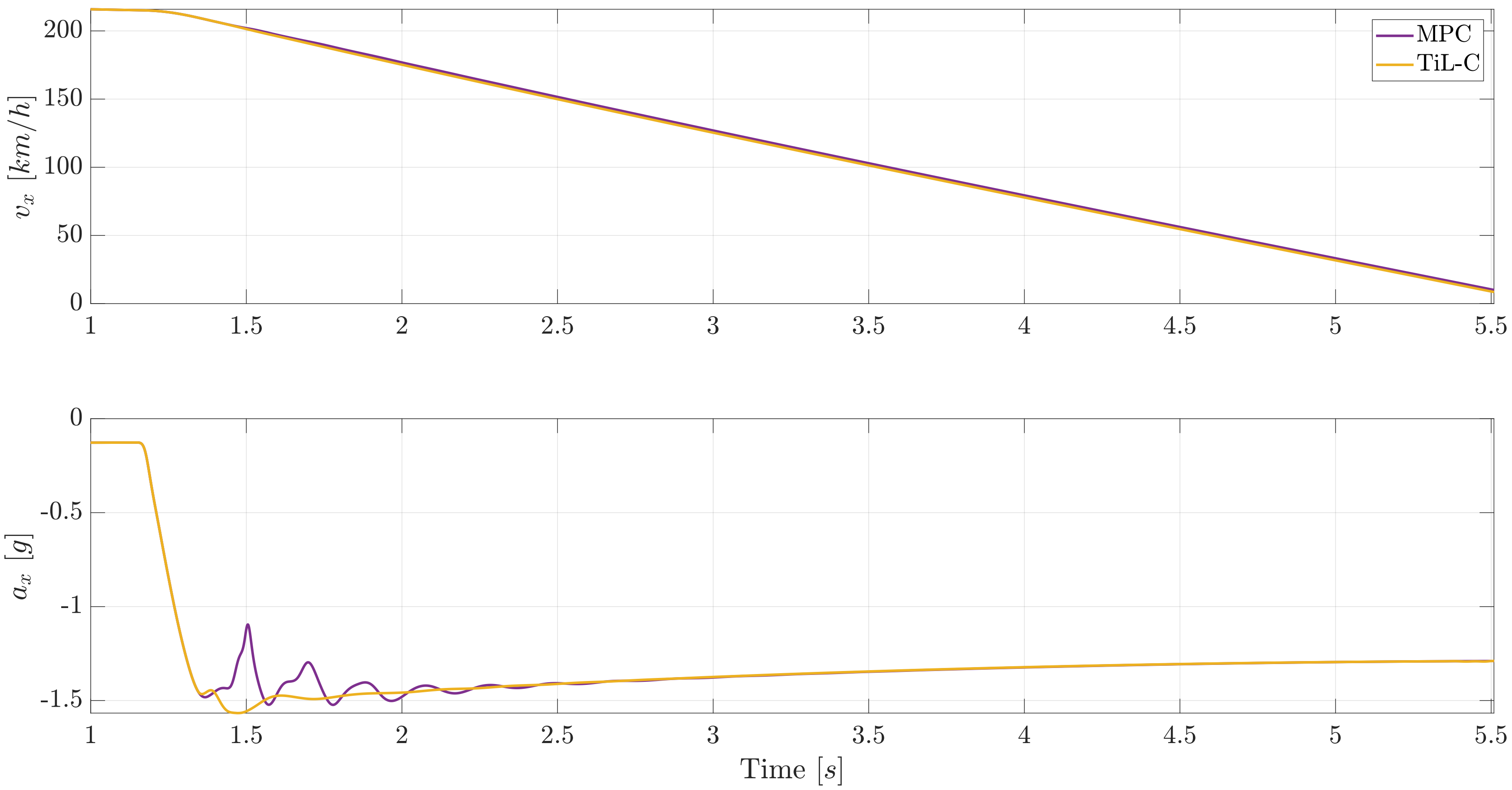
Significantly softer

Perturbated vehicle/noiseless data – testing experiment

Speed/acceleration (TiL vs bench)



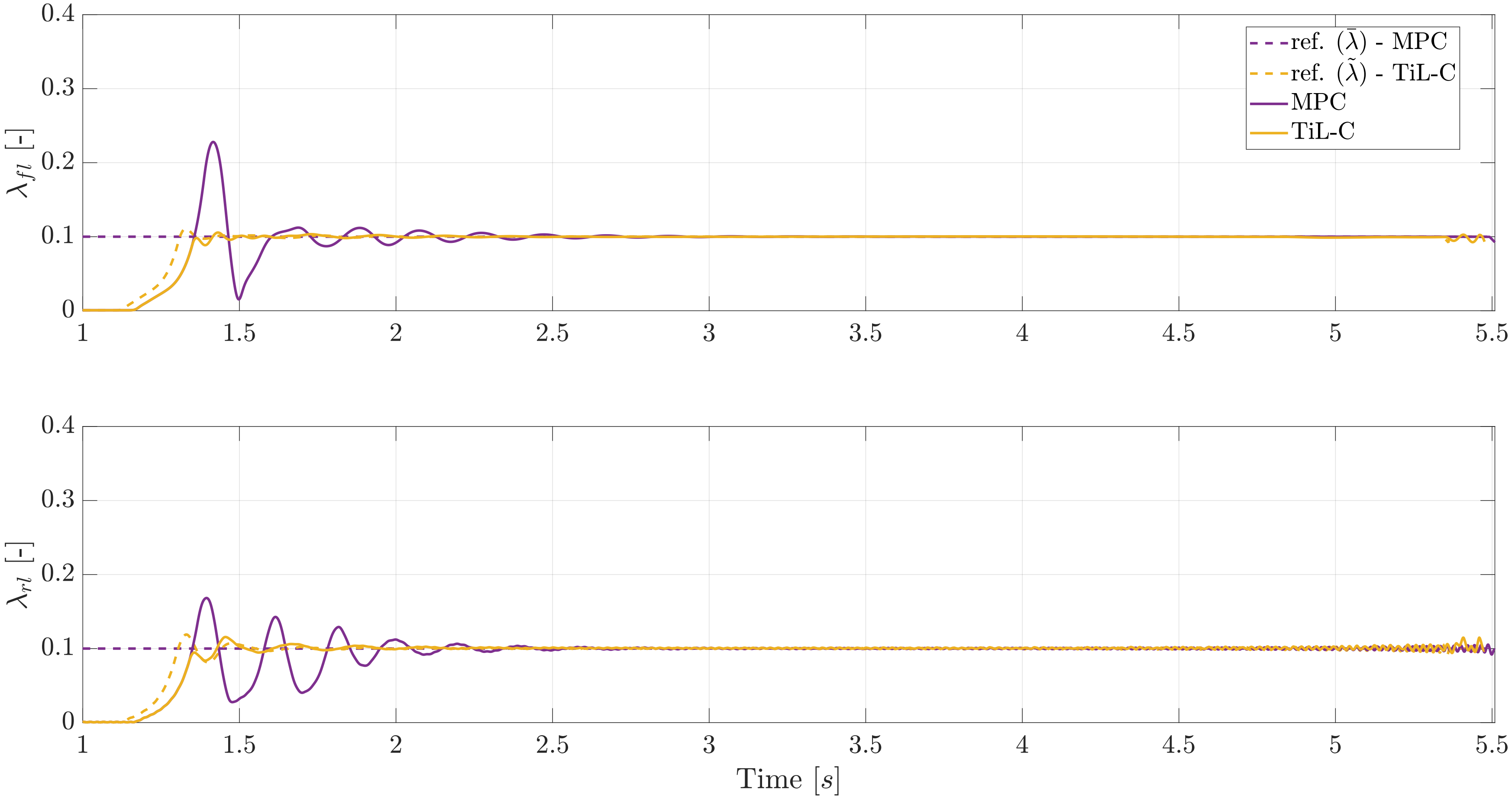
$SNR \approx \infty$

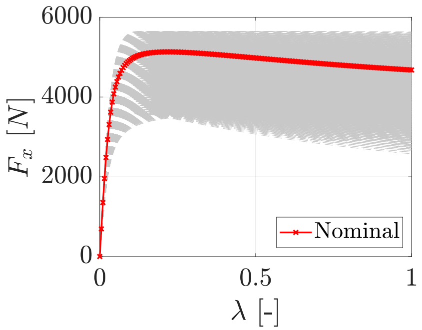


Perturbated vehicle/noiseless data – testing experiment

Slips (TiL vs bench)

	J_λ [%]
TiL	1.24
Bench	2.44
	+95%



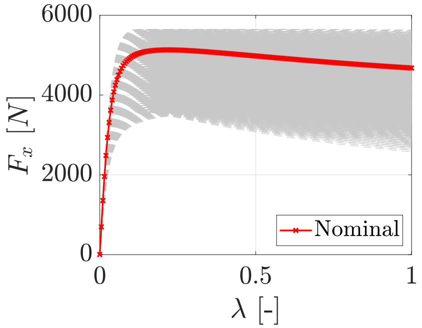


$\boldsymbol{SNR} \approx \infty$

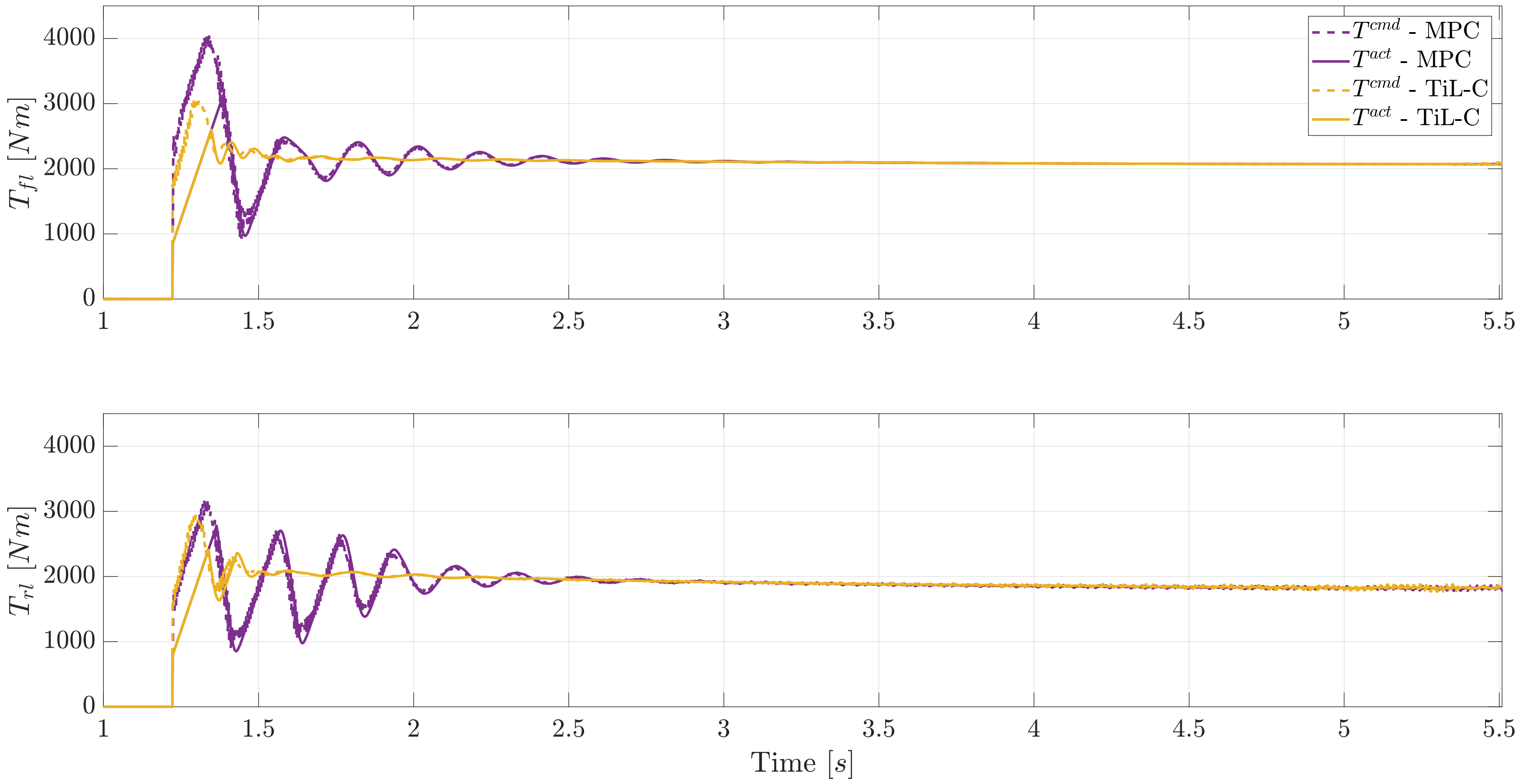
Perturbated vehicle/noiseless data – testing experiment

Slips (TiL vs bench)

	J_λ [%]	$J_u \left[\frac{Nm}{s}\right]$
TiL	1.24	4.64
Bench	2.44	7.67
	+95%	**+65%**



$SNR \approx \infty$



Perturbated vehicle/noisy data – testing experiment

m_p $[kg]$	m_t^l $[kg]$	m_t^r $[kg]$
0	0	0

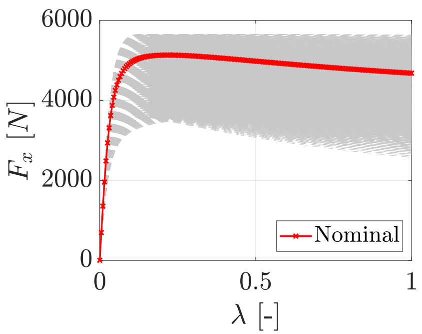
	J_λ [%]	J_u $\left[\frac{Nm}{s}\right]$
TiL	1.02	5.57
Bench	2.77	10.25
	+170%	**+84%**

m_p $[kg]$	m_t^l $[kg]$	m_t^r $[kg]$
50	0	0

	J_λ [%]	J_u $\left[\frac{Nm}{s}\right]$
TiL	1.11	5.59
Bench	2.79	9.78
	+151%	**+75%**

m_p $[kg]$	m_t^l $[kg]$	m_t^r $[kg]$
50	10	10

	J_λ [%]	J_u $\left[\frac{Nm}{s}\right]$
TiL	1.29	5.43
Bench	2.89	10.16
	+124%	**+87%**



$\boldsymbol{SNR \approx 4}$

What if we **test noisy-calibrated controllers with small unmodeled** masses?
Intrinsic robustness to masses.

Perturbated vehicle/noiseless data – testing experiment

$m_p\ [kg]$	$m_t^l\ [kg]$	$m_t^r\ [kg]$
0	0	0

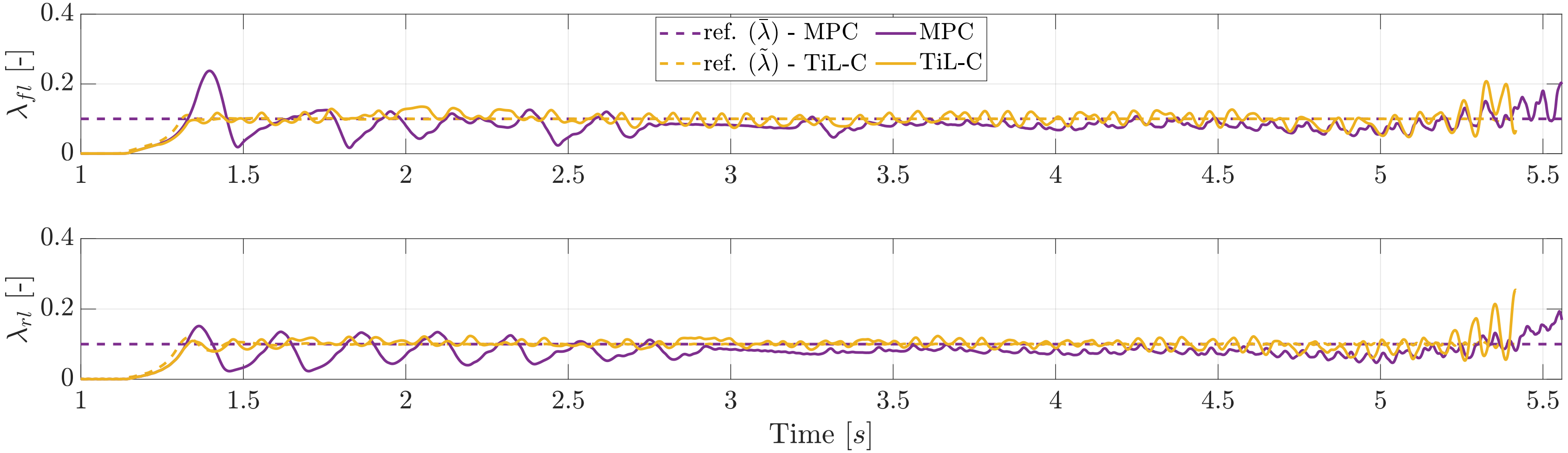
	J_λ [%]	$J_u \left[\frac{Nm}{s}\right]$
TiL	1.02	5.57
Bench	2.77	10.25
	+170%	**+84%**

$m_p\ [kg]$	$m_t^l\ [kg]$	$m_t^r\ [kg]$
50	0	0

	J_λ [%]	$J_u \left[\frac{Nm}{s}\right]$
TiL	1.11	5.59
Bench	2.79	9.78
	+151%	**+75%**

$m_p\ [kg]$	$m_t^l\ [kg]$	$m_t^r\ [kg]$
50	10	10

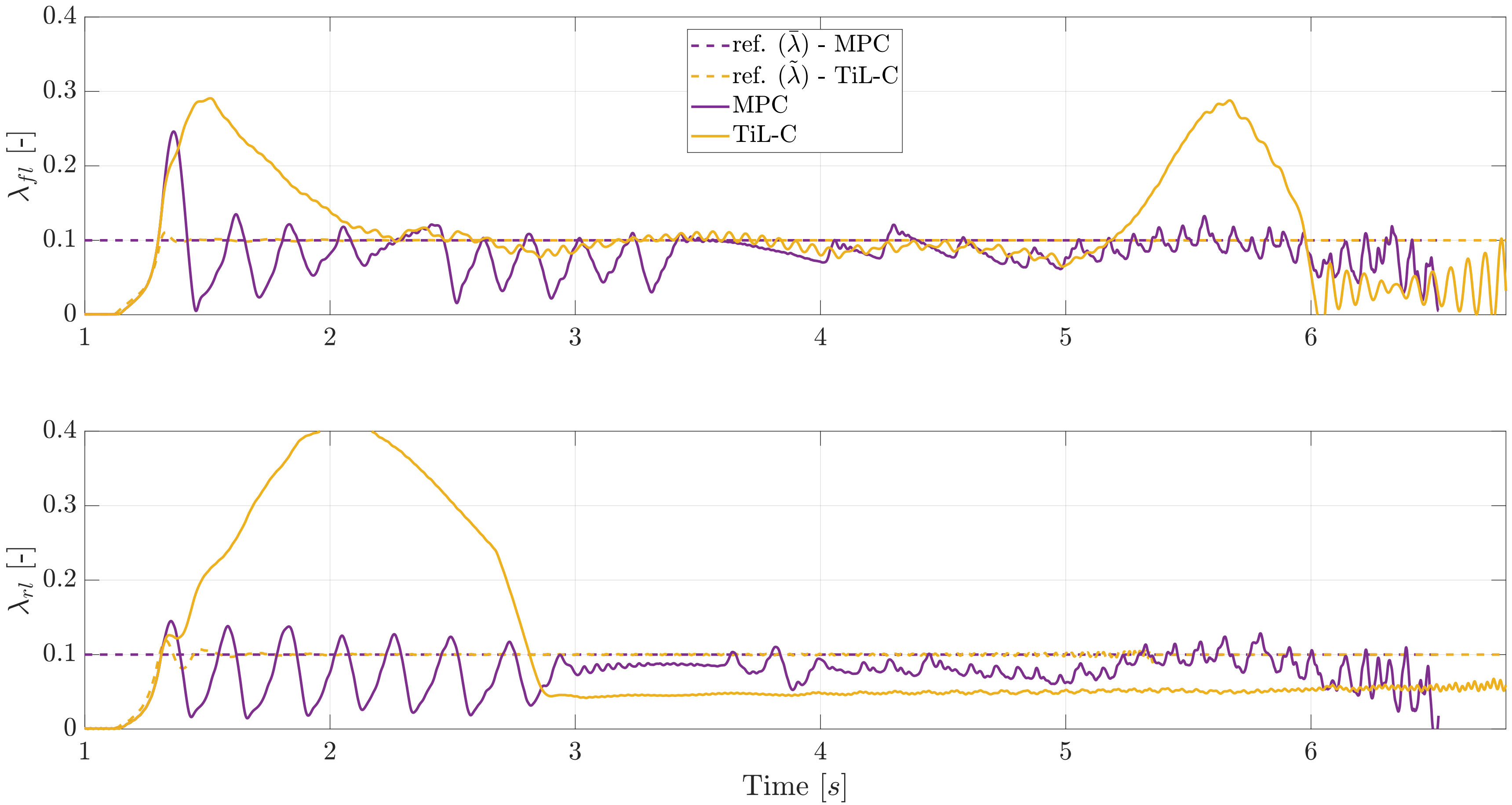
	J_λ [%]	$J_u \left[\frac{Nm}{s}\right]$
TiL	1.29	5.43
Bench	2.89	10.16
	+124%	**+87%**



Robustness to friction variation

What if we consider a **different friction model** when testing the noisy-calibrated controllers?

	J_λ [%]
TiL	12.30
Bench	2.89
	−76%



TiL-C not robust with respect to friction variations: necessity of robust tuning!

Results – TiL robust control tuning

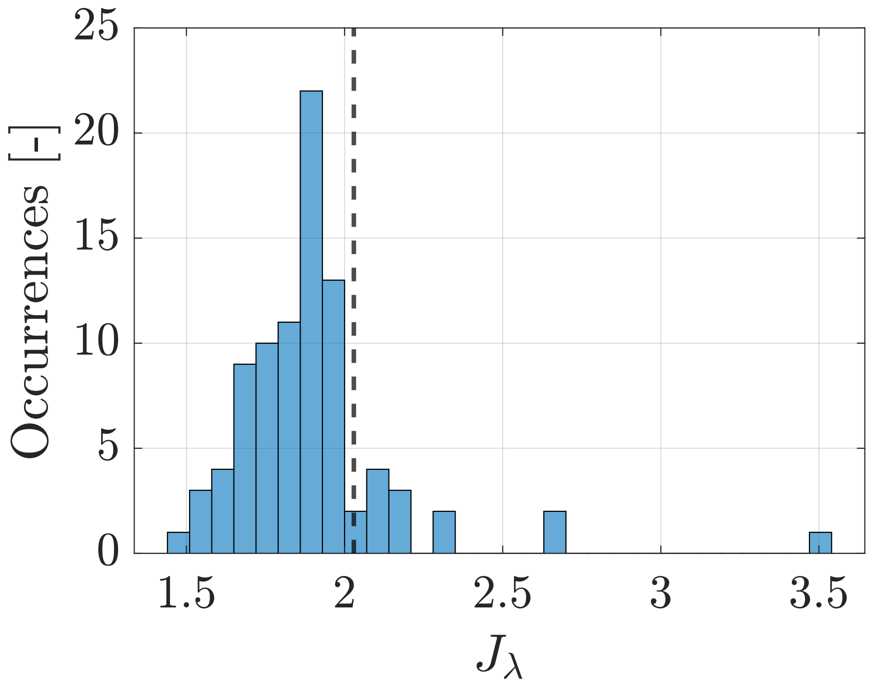
$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\mathbf{argmin}} \quad \boldsymbol{\gamma_{est}}(\boldsymbol{\theta}, \boldsymbol{\Delta}^{(1)}, \dots, \boldsymbol{\Delta}^{(M)})$$

subject to:

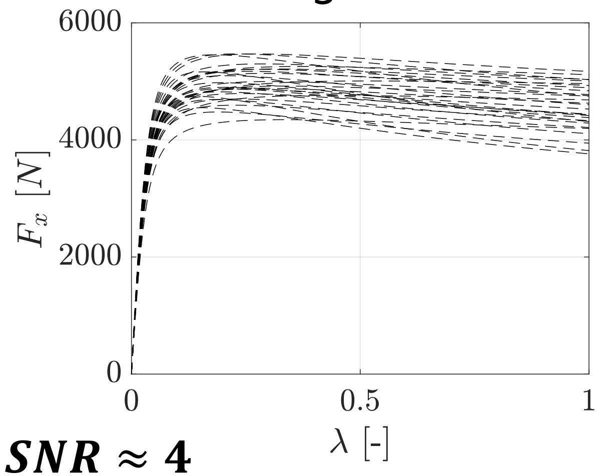
$$\gamma_{est} \geq J_\lambda(\theta, \Delta^{(j)}), \qquad j = 1, \dots, M$$

Parameters:

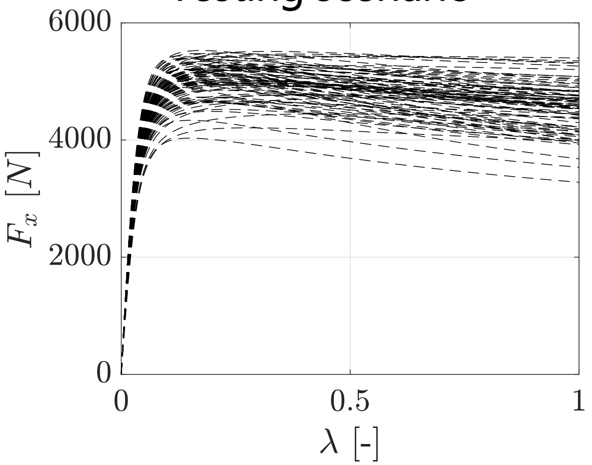
- $p^* = 0.85$;
- $1 - \delta = 0.99$;
- $M = 29$;
- $M_{test} = 3M = 87$.



Training scenarios



Testing scenario



$$\mathbf{Pr}\{\boldsymbol{J(\Delta)} \leq \boldsymbol{\gamma_{est}}\} = \boldsymbol{p_{est}} \approx \sum_{i=1}^{M_{test}} \frac{[\boldsymbol{J}(\boldsymbol{\Delta}^{(i)}) \leq \boldsymbol{\gamma_{est}}]}{\boldsymbol{M_{test}}} = \mathbf{0.8506} > \boldsymbol{p}^*$$

$\boldsymbol{SNR \approx 4}$

The found solutions is robust, satisfying the probabilistic guarantees out-of-sample.

Outline

Problem statement

Twin-in-the-Loop control

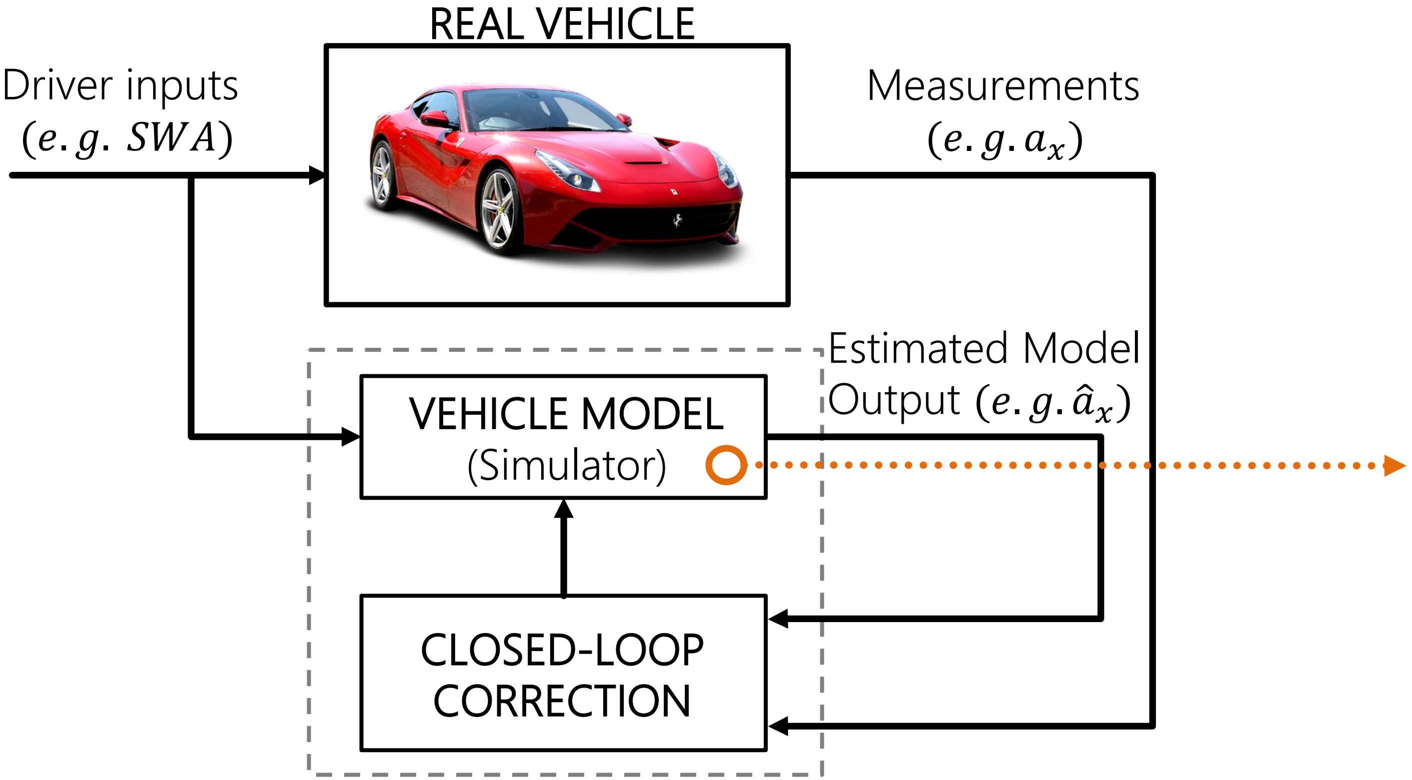
Twin-in-the-Loop estimation

Conclusions

Twin-in-the-Loop Filtering Paradigm - Block Diagram

Goal : development of a **full vehicle dynamics observer**. We are interested in:

- State Filtering
- Output Smoothing

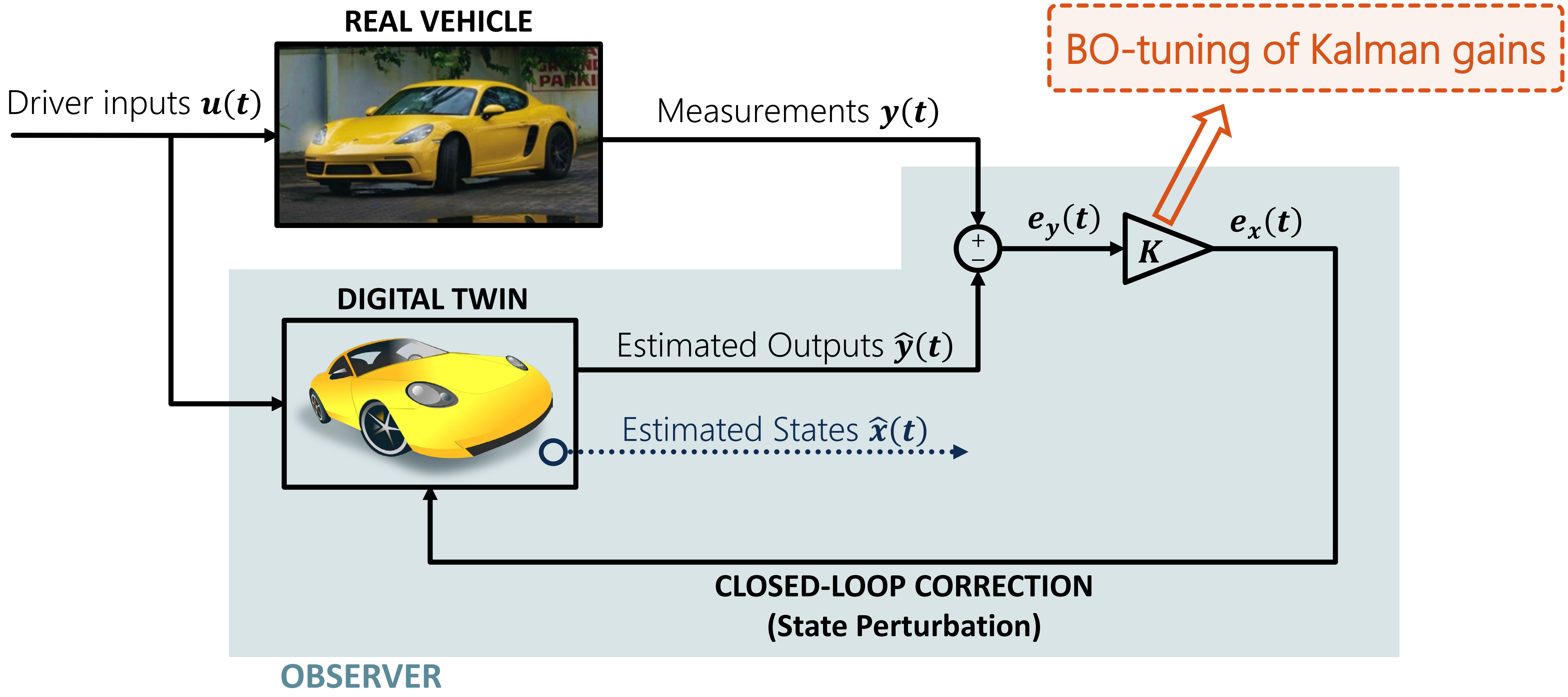


We can get (for example):

- Longitudinal speed - $\boldsymbol{v_x}$
- Yaw Rate - $\boldsymbol{\omega_z}$
- Sideslip angle - $\boldsymbol{\beta}$

Twin-in-the-Loop Filtering Paradigm - Block Diagram

We will use a **linear time-invariant correction law**:



TIL Filtering – Dimensionality Issue

In our setup we have:

- $n_x = 28$ states
- $n_y = 10$ measurements available

We would like to design an **output error correction** law:

$$\boldsymbol{\Delta x} = \boldsymbol{K \Delta y}$$

Where:

- Δx: state perturbation
- Δy: output error
- $K \in \mathbb{R}^{n_x \times n_y}$: correction matrix

$\Downarrow$

280 tuning parameters!

State variables (1)	
Chassis X-Position	p_x
Chassis Y-Position	p_y
Chassis Z-Position	p_z
Roll-Angle	θ
Pitch-Angle	ϕ
Yaw-Angle	ψ
Chassis Longitudinal Speed (front-ground)	v_x
Chassis Lateral Speed (front-ground)	v_y
Chassis Vertical Speed (front-ground)	v_z
X-Angular Speed	ω_x
Y-Angular Speed	ω_y
Z-Angular Speed	ω_z
FL Suspension Stroke	s_{fl}
FR Suspension Stroke	s_{fr}
RL Suspension Stroke	s_{rl}
RR Suspension Stroke	s_{rr}
FL Suspension Stroke Rate	$\dot{s}_{fl}$
FR Suspension Stroke Rate	$\dot{s}_{fr}$
RL Suspension Stroke Rate	$\dot{s}_{rl}$
RR Suspension Stroke Rate	$\dot{s}_{rr}$

State variables (2)	
FL Wheel Angle	Θ_{fl}
FR Wheel Angle	Θ_{fr}
RL Wheel Angle	Θ_{rl}
RR Wheel Angle	Θ_{rr}
FL Wheel Speed	ω_{fl}
FR Wheel Speed	ω_{fr}
RL Wheel Speed	ω_{rl}
RR Wheel Speed	ω_{rr}

Measured outputs	
X-Angular Speed	ω_x^m
Y-Angular Speed	ω_y^m
Z-Angular Speed	ω_z^m
FL Wheel Speed	ω_{fl}^m
FR Wheel Speed	ω_{fr}^m
RL Wheel Speed	ω_{rl}^m
RR Wheel Speed	ω_{rr}^m
Longitudinal Acceleration	a_x^m
Lateral Acceleration	a_y^m
Vertical Acceleration	a_z^m

Optimization Problem Statement

$$\min_{k_{i,j}} L(x_{meas}, \hat{x})$$ ⇦ Maximize filter performance

s.t.: $k_{i,j} \in [\underline{k}_{i,j}, \bar{k}_{i,j}], \quad \forall\, i = 1, \dots, n_x, \quad j = 1, \dots, n_y$ ⇦ Optimization variable ranges

Problem: **time consuming simulations** → gridding and gradient based methods would take years to converge, even with BO

PROBLEM! K might have **hundreds of parameters**, while BO can optimize only a handful

⇩

Dimensionality reduction needed

Simplified Architecture

Let us consider a simplified case study.

Corrected **states**:

$$\left.\begin{array}{l} \bullet\ \boldsymbol{v_x} \\ \bullet\ \boldsymbol{\omega_{fl}} \\ \bullet\ \boldsymbol{\omega_{rr}} \end{array}\right\} \textbf{Longitudinal}\text{ dynamics}$$

$$\left.\bullet\ \boldsymbol{\omega_z}\ \right\} \textbf{Lateral}\text{ dynamics}$$

Measured **outputs:**

- $\boldsymbol{\omega_{fl}}$
- $\boldsymbol{\omega_{rr}}$
- $\boldsymbol{\omega_z}$

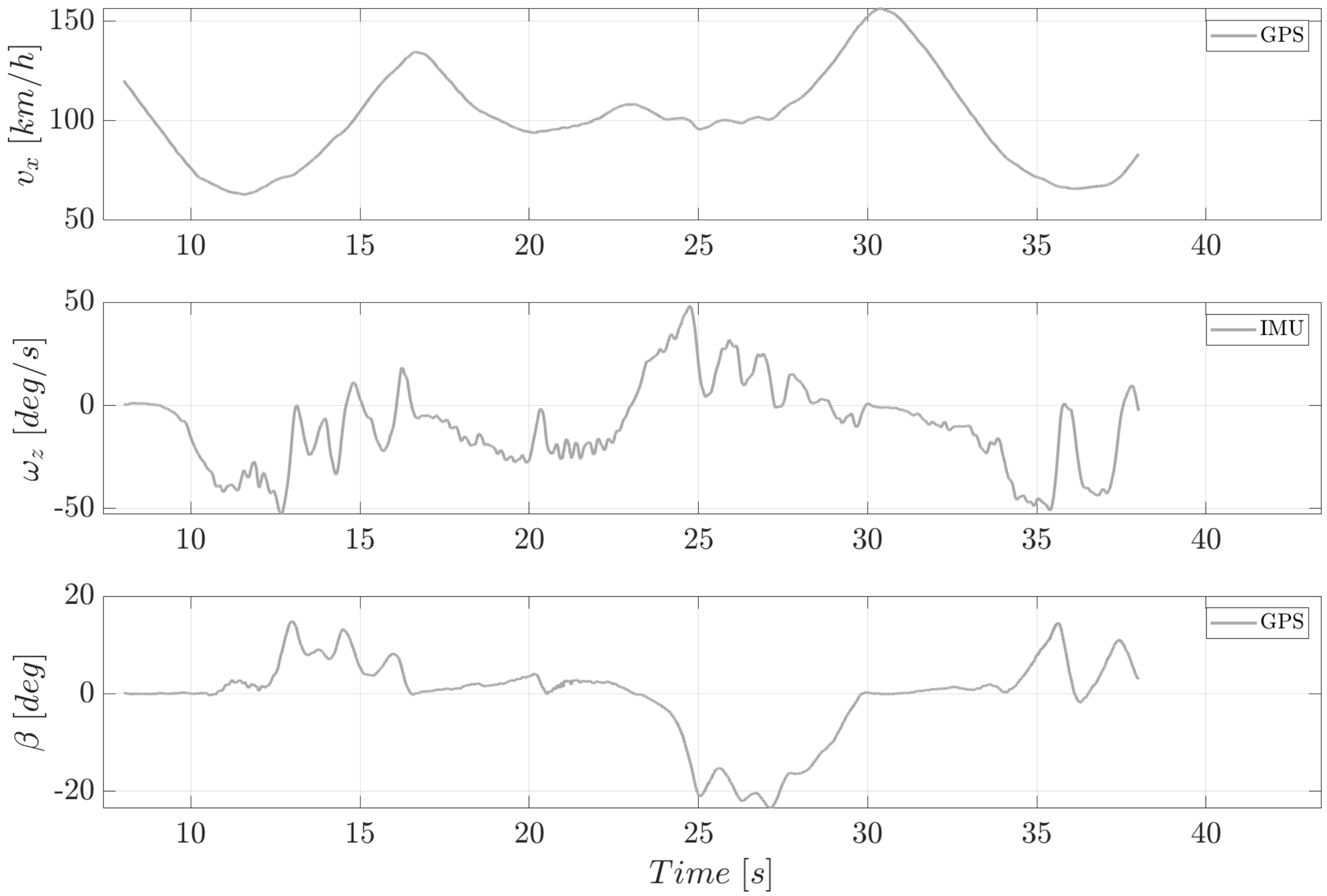
Only 12 parameters!

The **correction rule** becomes: $\begin{bmatrix} \Delta x_{v_x} \\ \Delta x_{\omega_z} \\ \Delta x_{\omega_{fl}} \\ \Delta x_{\omega_{rr}} \end{bmatrix} = \boldsymbol{K} \cdot \begin{bmatrix} \Delta y_{\omega_z} \\ \Delta y_{\omega_{fl}} \\ \Delta y_{\omega_{rr}} \end{bmatrix}$ where $K = \begin{bmatrix} k_{\omega_z v_x} & k_{\omega_{fl} v_x} & k_{\omega_{rr} v_x} \\ k_{\omega_z \omega_z} & k_{\omega_{fl} \omega_z} & k_{\omega_{rr} \omega_z} \\ k_{\omega_z \omega_{fl}} & k_{\omega_{fl} \omega_{fl}} & k_{\omega_{rr} \omega_{fl}} \\ k_{\omega_z \omega_{rr}} & k_{\omega_{fl} \omega_{rr}} & k_{\omega_{rr} \omega_{rr}} \end{bmatrix}$

Optimization Setup

Optimization setup:

- We choose a **dynamic training dataset** (both lateral and longitudinal dynamics are excited)
- Run **1000 iterations** (500 of which are random initial points)
- **Validate the result on a different lap**



Performance optimization problem

To tune the parameters in we will solve the following **optimization problem**:

Lateral dynamics tracking

Longitudinal dynamics tracking

$$\min_{\boldsymbol{k_{i,j}}} \ \boldsymbol{rms}(\boldsymbol{\omega_z} - \boldsymbol{\hat{\omega}_z}) + \boldsymbol{rms}(\boldsymbol{v_x} - \boldsymbol{\hat{v}_x})$$

$$\text{s.t.:} \quad k_{i,j} \in \left[\underline{k}_{i,j}, \bar{k}_{i,j}\right], \quad \forall\, i = 1, \dots, n_x, \quad j = 1, \dots, n_y$$

How many BO iterations? Optimizing all 12 parameters would take months to converge…

If we perform 1000 iterations → **impossible** to find a **converging solution** in a reasonable time!

⇩

12 parameters are still too many!
Need to reduce the dimensionality

Dimensionality reduction in Twin-in-the-loop estimation

We will discuss three possible solutions:

1. **Model-based** dimensionality reduction
2. Dimensionality reduction based on **supervised learning**
3. Dimensionality reduction based on **unsupervised learning**

Dimensionality reduction in Twin-in-the-loop estimation

We will discuss three possible solutions:

1. **Model-based** dimensionality reduction
2. Dimensionality reduction based on **supervised learning**
3. Dimensionality reduction based on **unsupervised learning**

Dimensionality Reduction

Model-based reduction: reduce the **number of parameters a-priori**, thanks to the physical knowledge on the system.

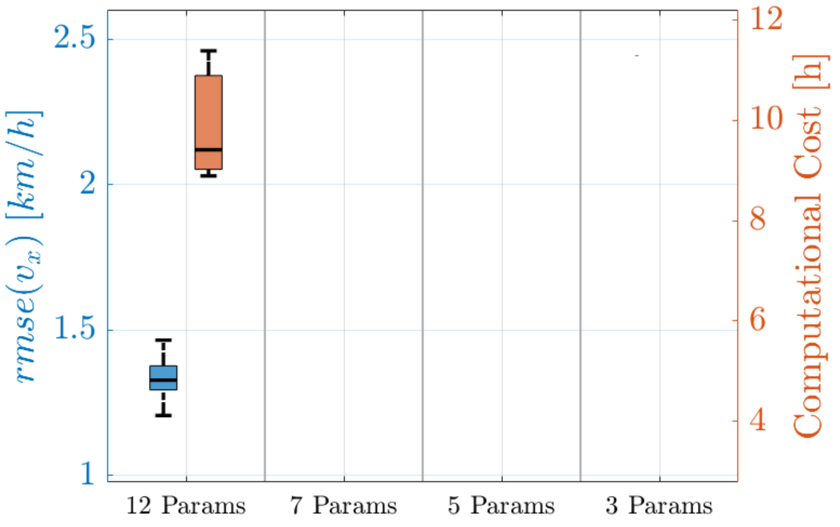
We will remove the following parameters:

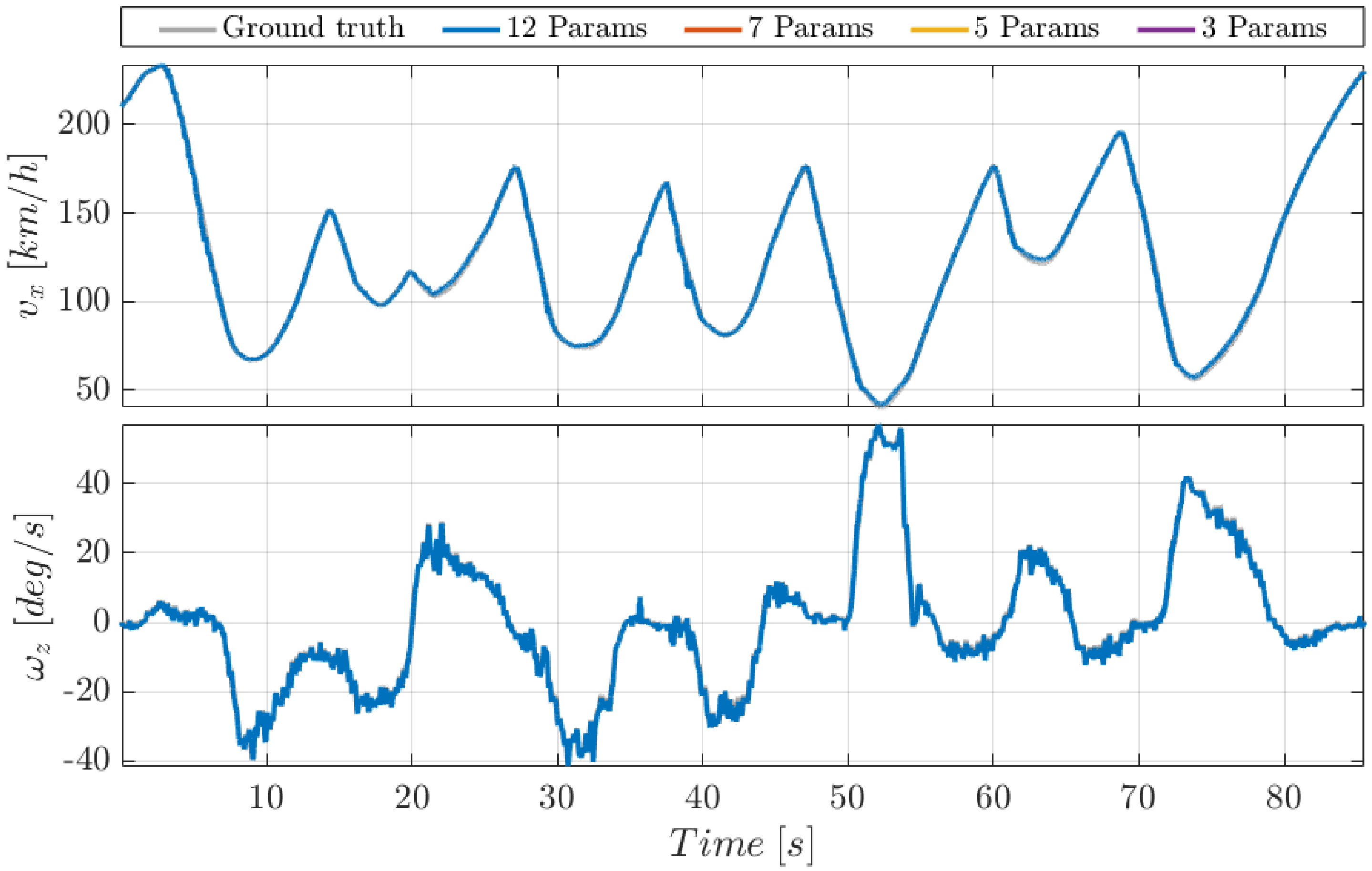
1) $k_{\omega_z \omega_{fl}}, k_{\omega_z \omega_{rr}}, k_{\omega_z v_x}, k_{\omega_{fl} \omega_z}, k_{\omega_{rr} \omega_z}$ → optimization with 7 parameters
2) $k_{\omega_{fl} \omega_{rr}}, k_{\omega_{rr} \omega_{fl}}$ → optimization with 5 parameters
3) $k_{\omega_{fl} v_x}, k_{\omega_{rr} v_x}$ → optimization with 3 parameters

Finally, we **optimize** the **remaining parameters**.

Validation #1 – 12 Parameters

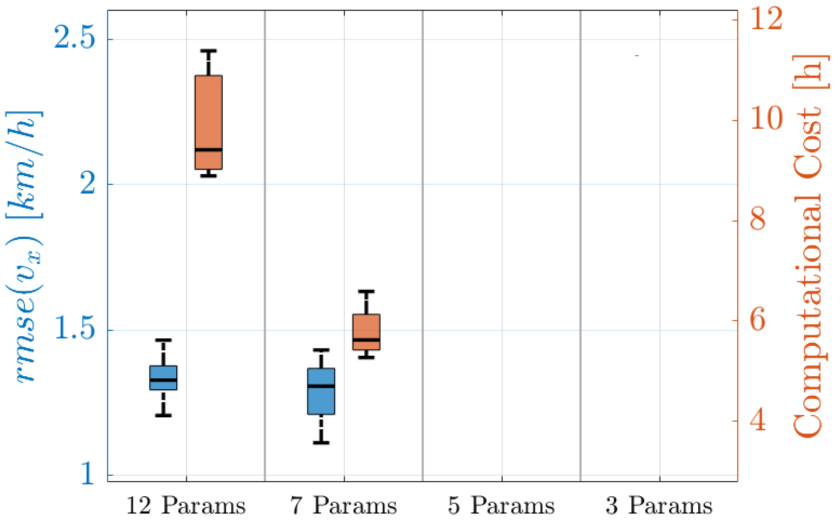
VALIDATION DATASET:
- 12 Params

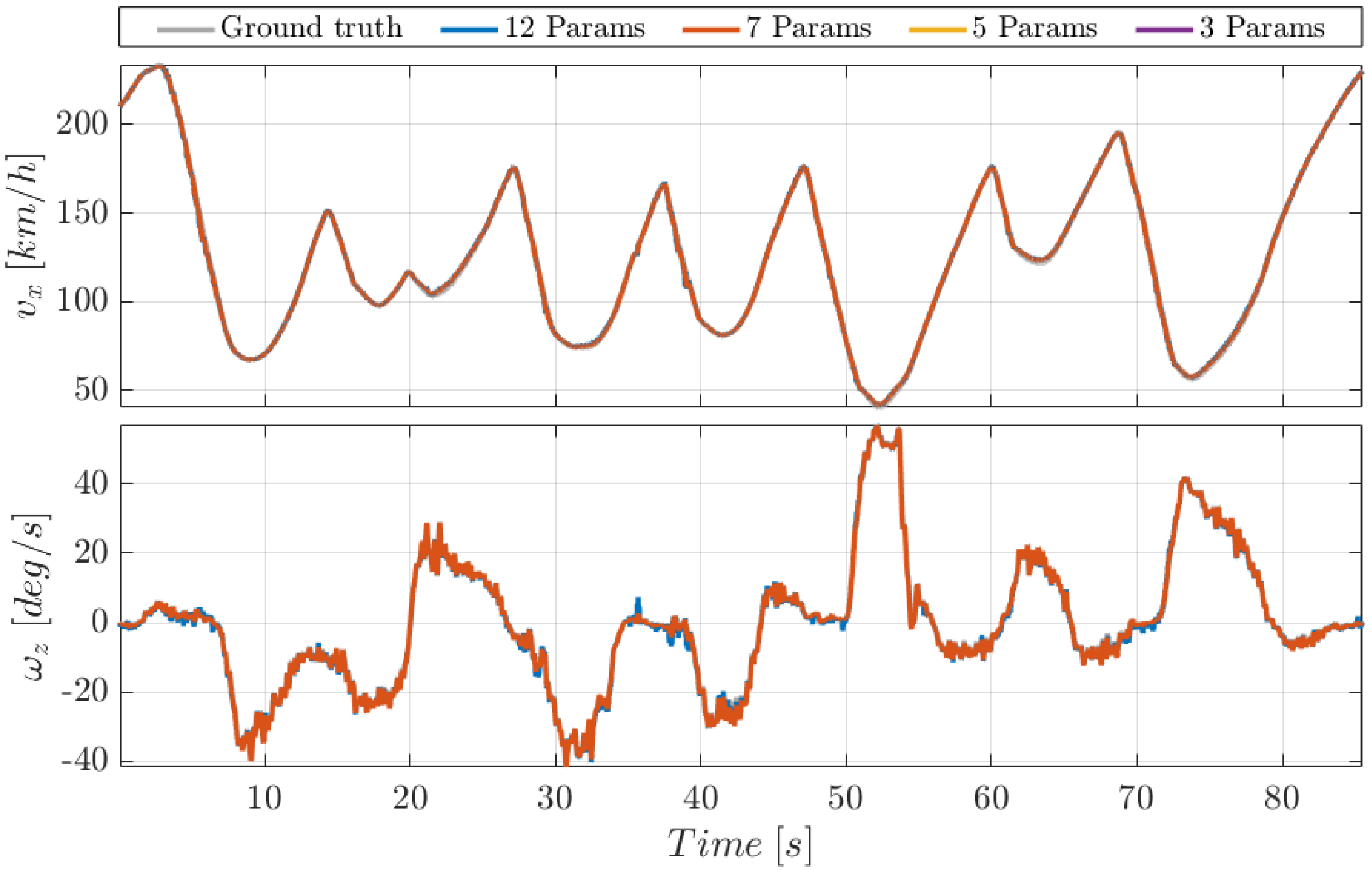




Validation #2 – 7 Parameters

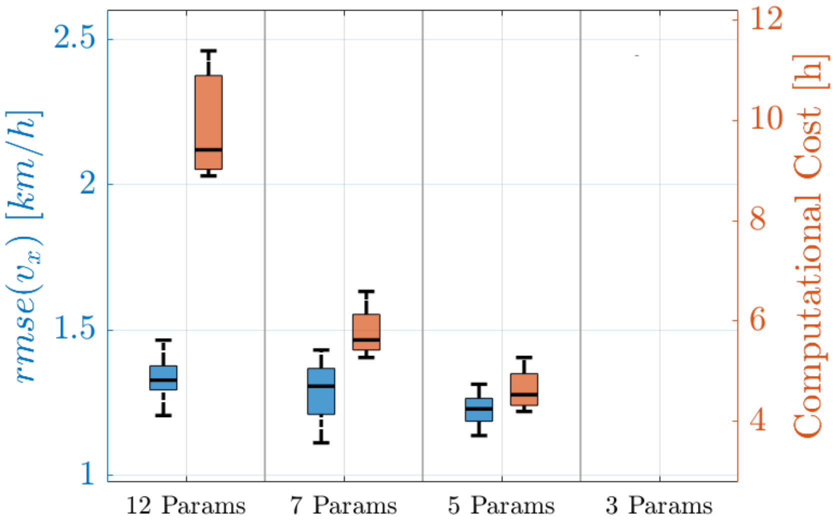
VALIDATION DATASET:
- 7 Params

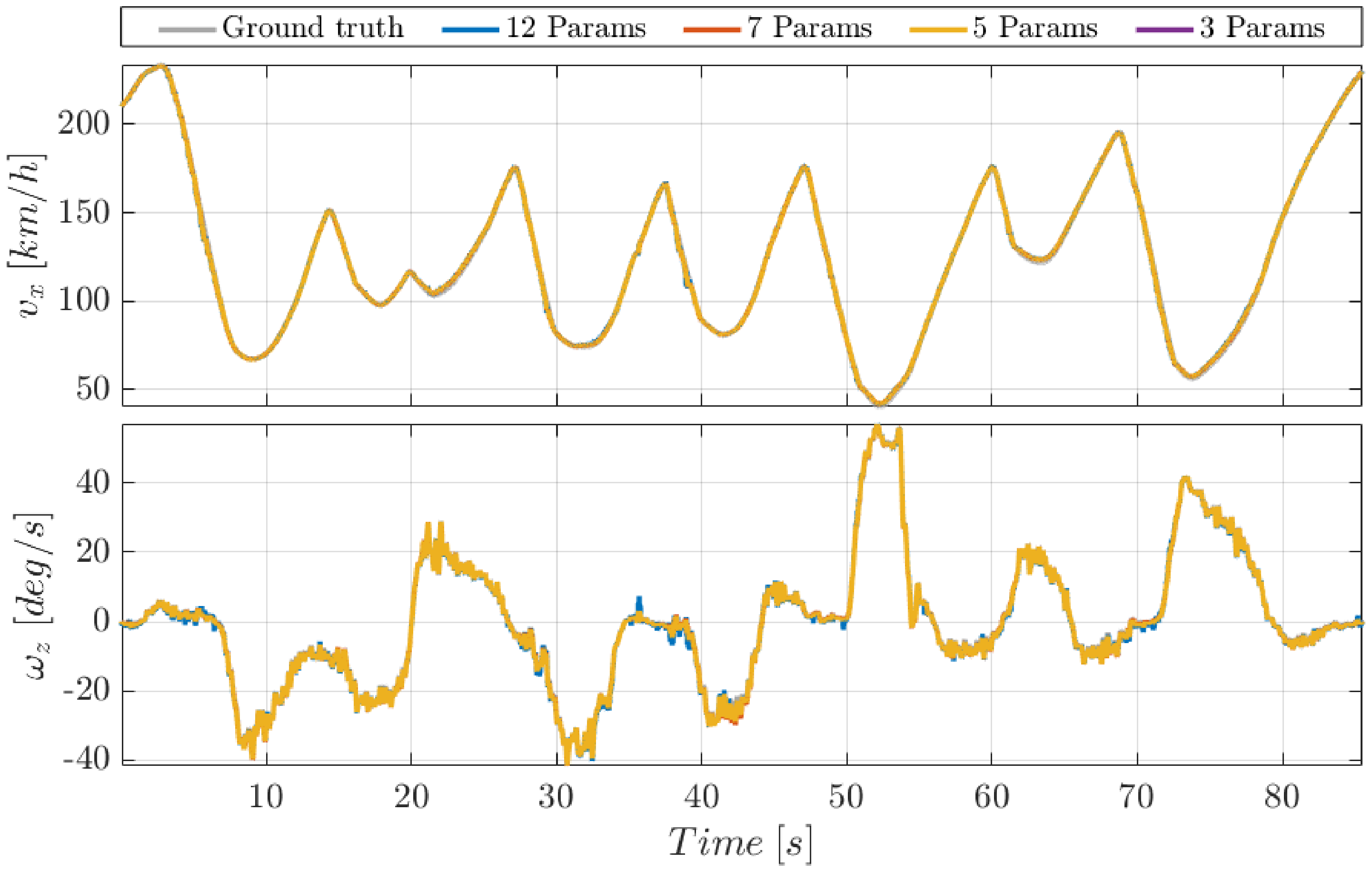




Validation #3 – 5 Parameters

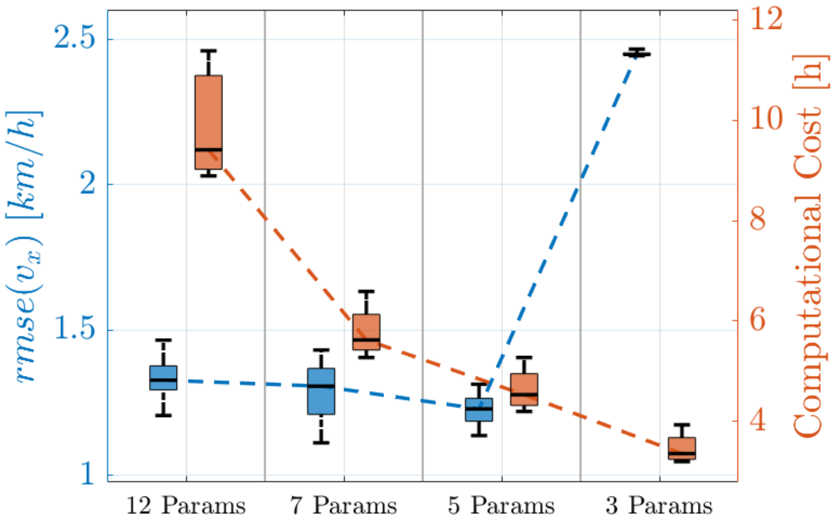
VALIDATION DATASET:
- 5 Params

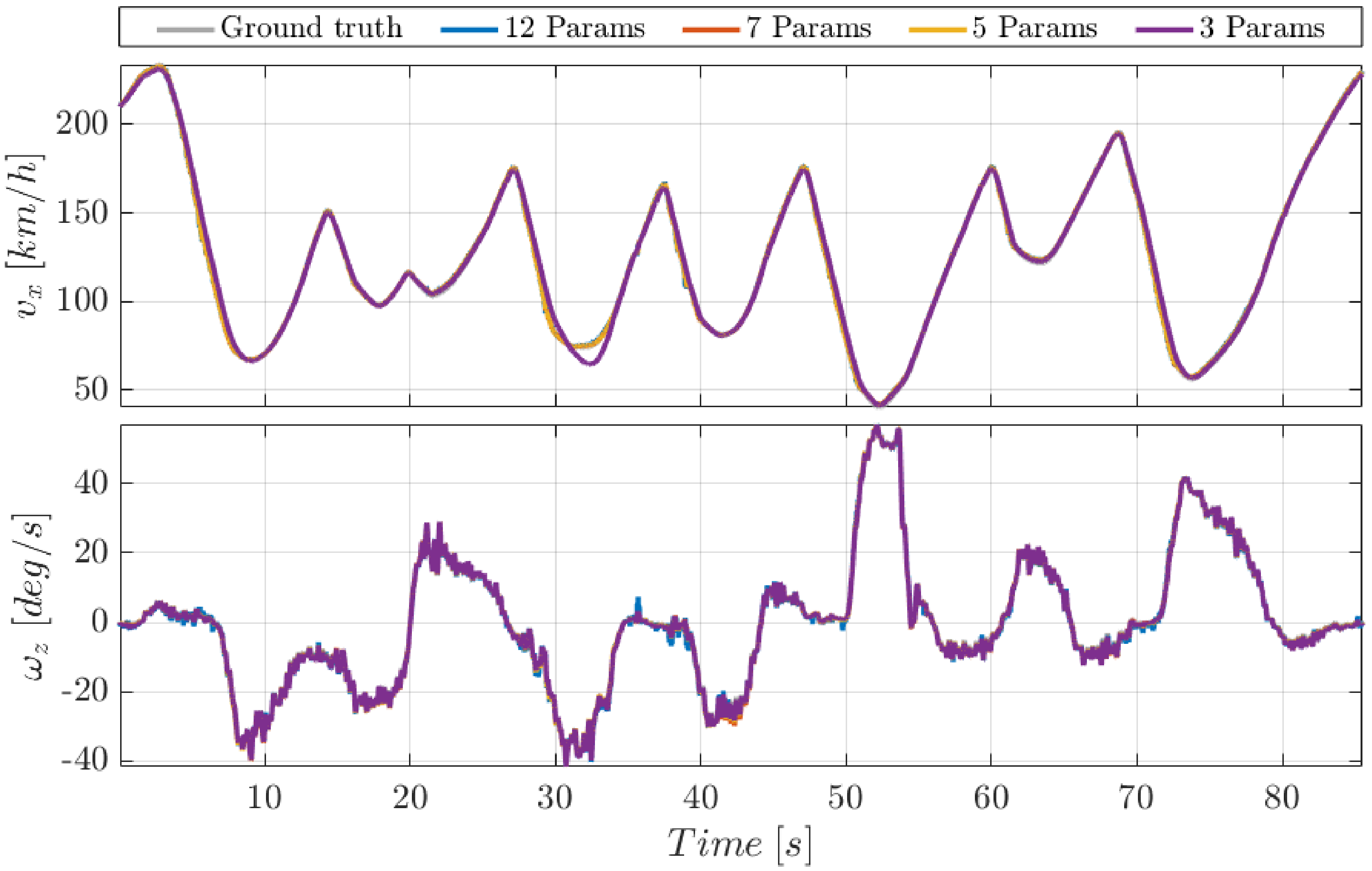




Validation #4 – 5 Parameters

VALIDATION DATASET:
- 5 Params

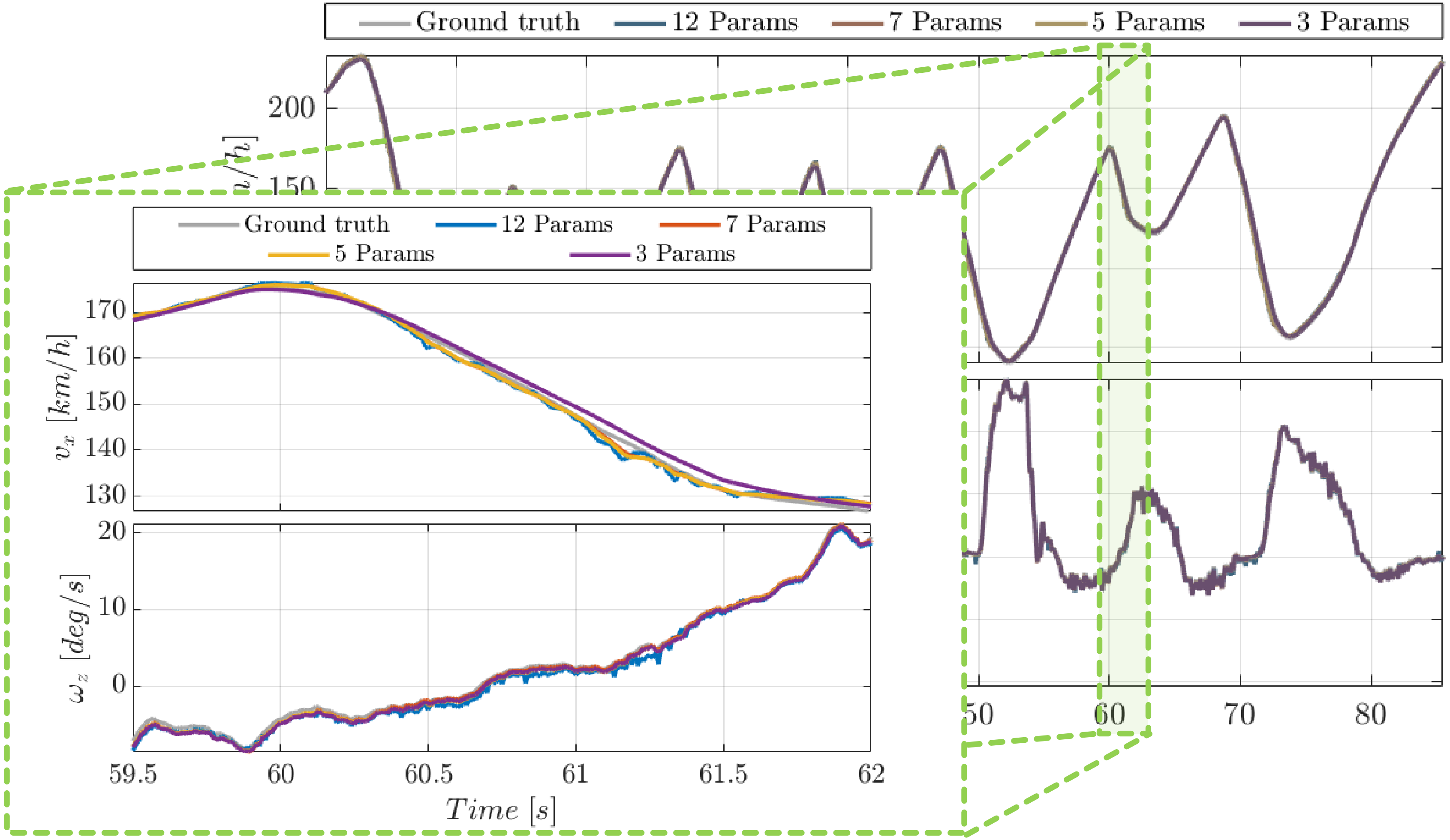




Validation comparison - Zoom

We can now compare a smaller section of the validation lap:

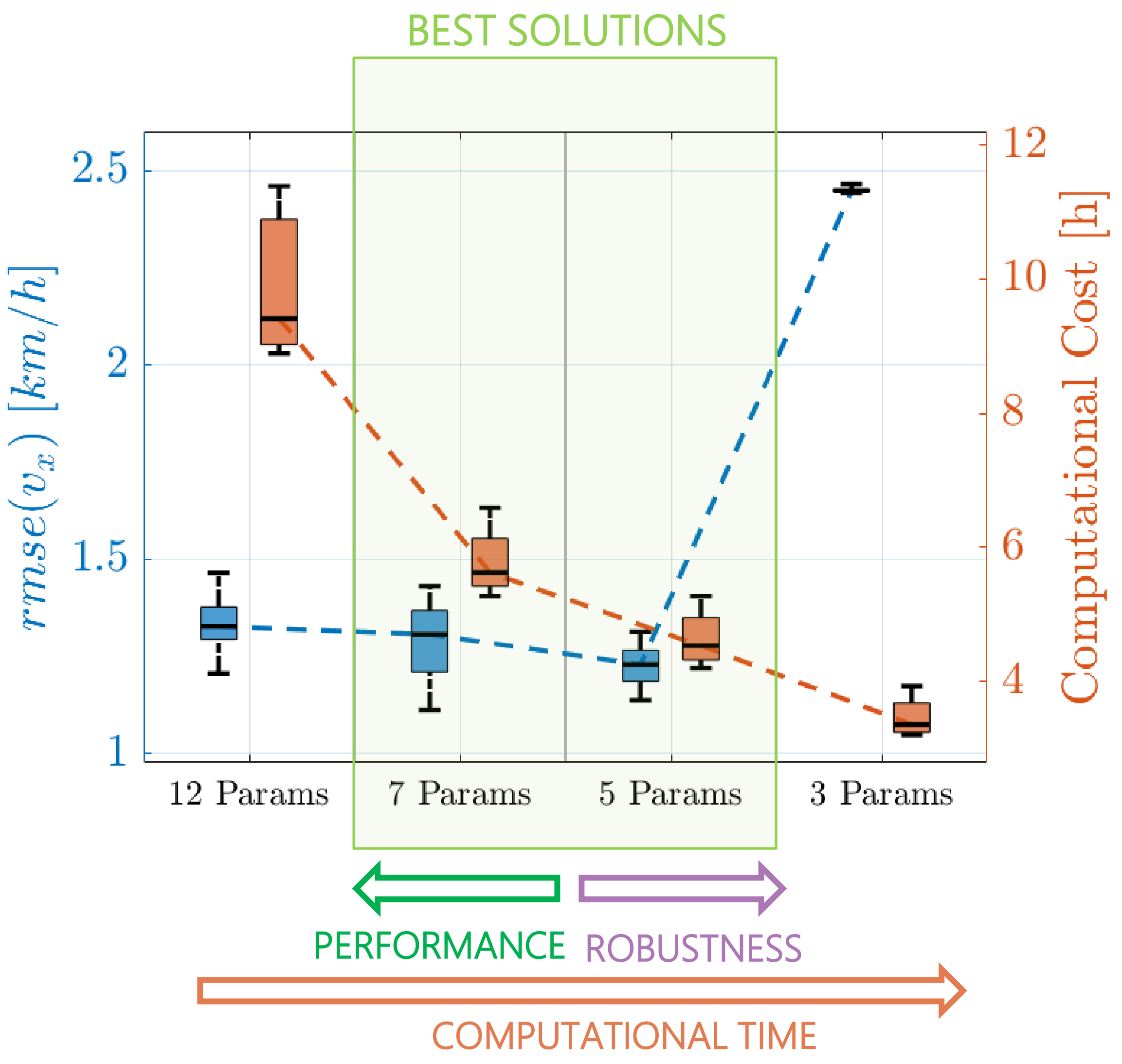
- 12 Parameters: very **noisy**
- 7 and 5 parameters: much **smoother**
- 3 parameters: smooth but **inaccurate**



Validation Comparison

Overall conclusions:

- If the number of **parameters** is too **high:**
    - No convergence
    - Noisy and unreliable solution
    - Computationally heavy
    - No robustness
- If the number of **parameters** is too **low:**
    - Solution is **too smooth**
    - **Inaccurate solution**



Dimensionality reduction in Twin-in-the-loop estimation

We will discuss three possible solutions:

1. **Model-based** dimensionality reduction
2. Dimensionality reduction based on **supervised learning**
3. Dimensionality reduction based on **unsupervised learning**

Automated Design of the Observer Structure

Three-step procedure:

1) We **sort** the parameters **by importance** by solving the following optimization problem:

Normalized parameters

$$\min_{k_{i,j}} \|\tilde{K}\|_1$$

Minimization of the ℓ_1-norm in order to **promote sparsity**

$$\text{s.t.:} \quad k_{i,j} \in [\underline{k}_{i,j}, \bar{k}_{i,j}], \quad \forall\, i = 1, \ldots, n_x, \quad j = 1, \ldots, n_y$$
$$\boldsymbol{rms(\omega_z - \hat{\omega}_z) < \mathrm{rms}_{\omega_{z\,threshold}}}$$
$$\boldsymbol{rms(v_x - \hat{v}_x) < \mathrm{rms}_{v_{x\,threshold}}}$$

Constraints on v_x and ω_z so that we can estimate both **longitudinal and lateral dynamics**.

2) Select a threshold δ to remove the least important parameters:

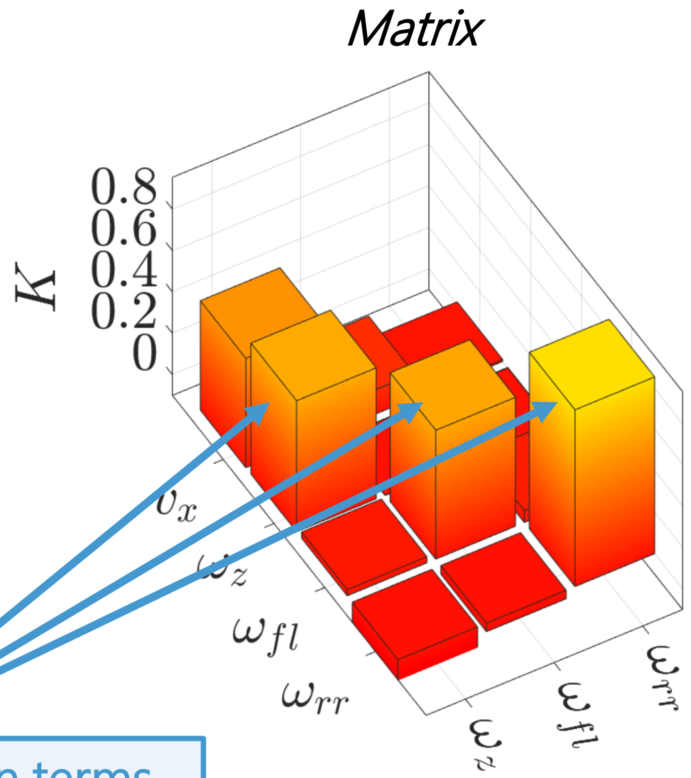
$$\begin{cases} \tilde{k}_{i,j} \geq \delta & \rightarrow \quad \text{keep parameter} \\ \boldsymbol{\tilde{k}_{i,j} < \delta} & \rightarrow \quad \textbf{remove parameter} \end{cases}$$

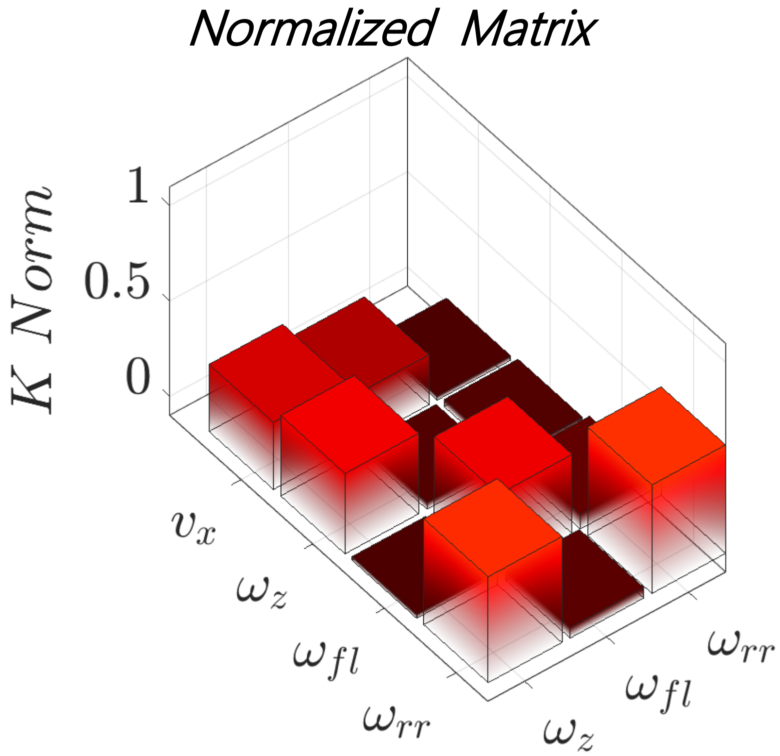
3) Run the **performance optimization** on the remaining parameters

Structure Optimization – 12 Parameters

We can now run the structure optimization and see what are the most important parameters:

OPTIMIZATION:
- 12 Params
  $\mathbf{rms}_{\omega_{z\,threshold}} = \mathbf{1.5}\ \boldsymbol{deg/s}$
  $\mathbf{rms}_{v_{x\,threshold}} = \mathbf{1.5}\ \boldsymbol{km/h}$
- 1000 Iterations





High auto-correlation terms

Automated Design Procedure

We will choose 3 values for δ and get 3 optimizations, with respectively 8, 6, 4 parameters:

$$\delta = 0.05$$

$$K = \begin{bmatrix} k_{\omega_z v_x} & k_{\omega_{fl} v_x} & \mathbf{k_{\omega_{rr} v_x}} \\ k_{\omega_z \omega_z} & \mathbf{k_{\omega_{fl} \omega_z}} & \mathbf{k_{\omega_{rr} \omega_z}} \\ \mathbf{k_{\omega_z \omega_{fl}}} & k_{\omega_{fl} \omega_{fl}} & k_{\omega_{rr} \omega_{fl}} \\ k_{\omega_z \omega_{rr}} & k_{\omega_{fl} \omega_{rr}} & k_{\omega_{rr} \omega_{rr}} \end{bmatrix}$$

- Almost all parameters involving ω_z are removed

$$\delta = 0.1$$

$$K = \begin{bmatrix} k_{\omega_z v_x} & k_{\omega_{fl} v_x} & \mathbf{k_{\omega_{rr} v_x}} \\ k_{\omega_z \omega_z} & \mathbf{k_{\omega_{fl} \omega_z}} & \mathbf{k_{\omega_{rr} \omega_z}} \\ \mathbf{k_{\omega_z \omega_{fl}}} & k_{\omega_{fl} \omega_{fl}} & \mathbf{k_{\omega_{rr} \omega_{fl}}} \\ k_{\omega_z \omega_{rr}} & \mathbf{k_{\omega_{fl} \omega_{rr}}} & k_{\omega_{rr} \omega_{rr}} \end{bmatrix}$$

- Almost all parameters involving ω_z are removed
- Wheel cross correlations are removed

$$\delta = 0.4$$

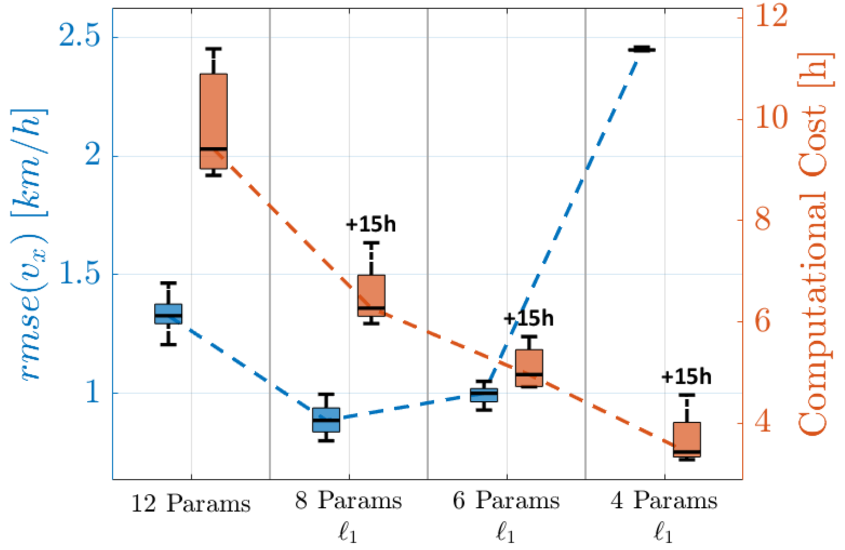
$$K = \begin{bmatrix} \mathbf{k_{\omega_z v_x}} & \mathbf{k_{\omega_{fl} v_x}} & \mathbf{k_{\omega_{rr} v_x}} \\ k_{\omega_z \omega_z} & \mathbf{k_{\omega_{fl} \omega_z}} & \mathbf{k_{\omega_{rr} \omega_z}} \\ \mathbf{k_{\omega_z \omega_{fl}}} & k_{\omega_{fl} \omega_{fl}} & \mathbf{k_{\omega_{rr} \omega_{fl}}} \\ k_{\omega_z \omega_{rr}} & \mathbf{k_{\omega_{fl} \omega_{rr}}} & k_{\omega_{rr} \omega_{rr}} \end{bmatrix}$$

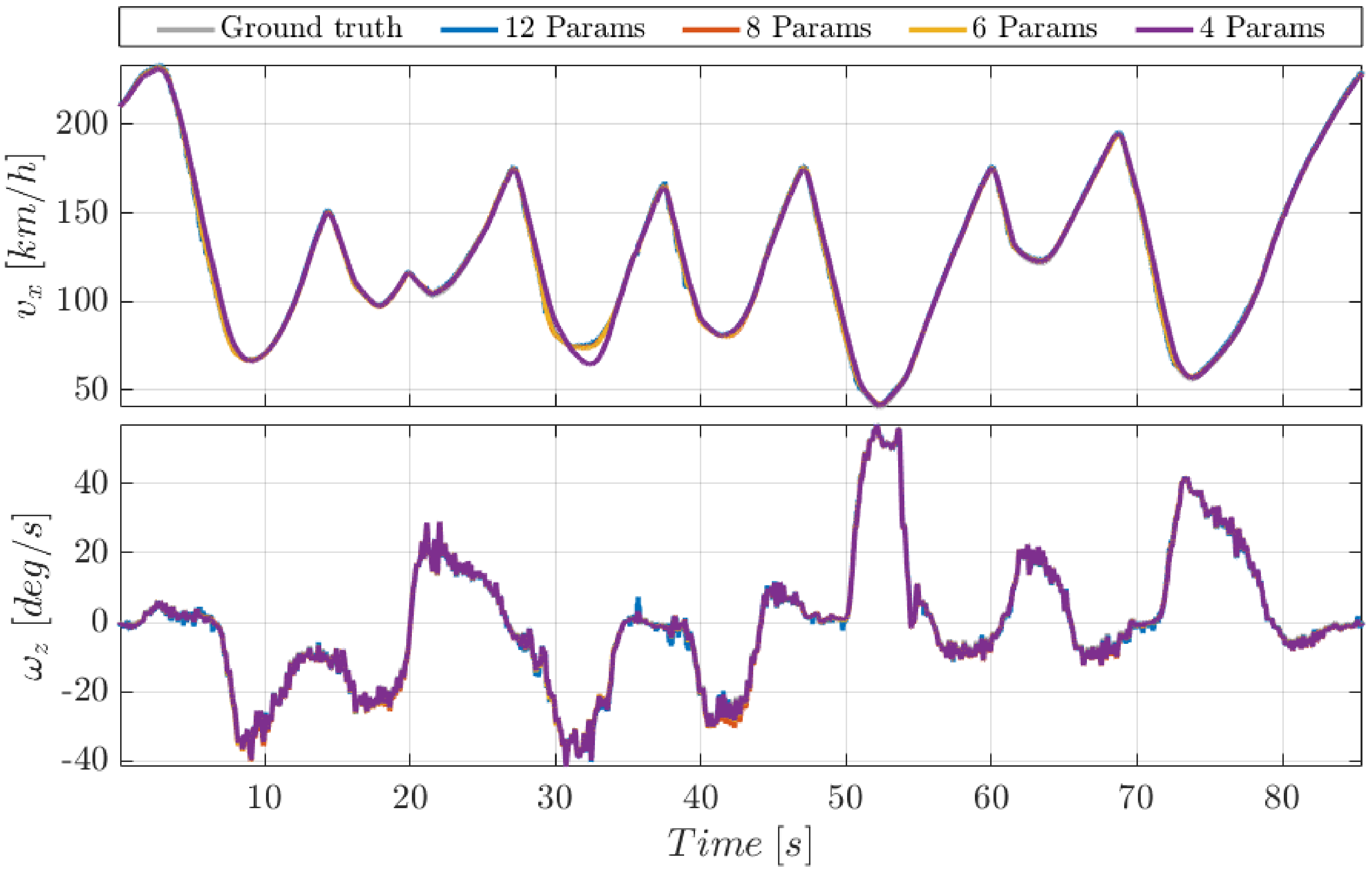
- Almost all parameters involving ω_z are removed
- Wheel cross correlations are removed
- Any correction for v_x is removed

Deleted parameters

All validations

VALIDATION DATASET:
- 12 Params
- 8 Params
- 6 Params
- 4 Params

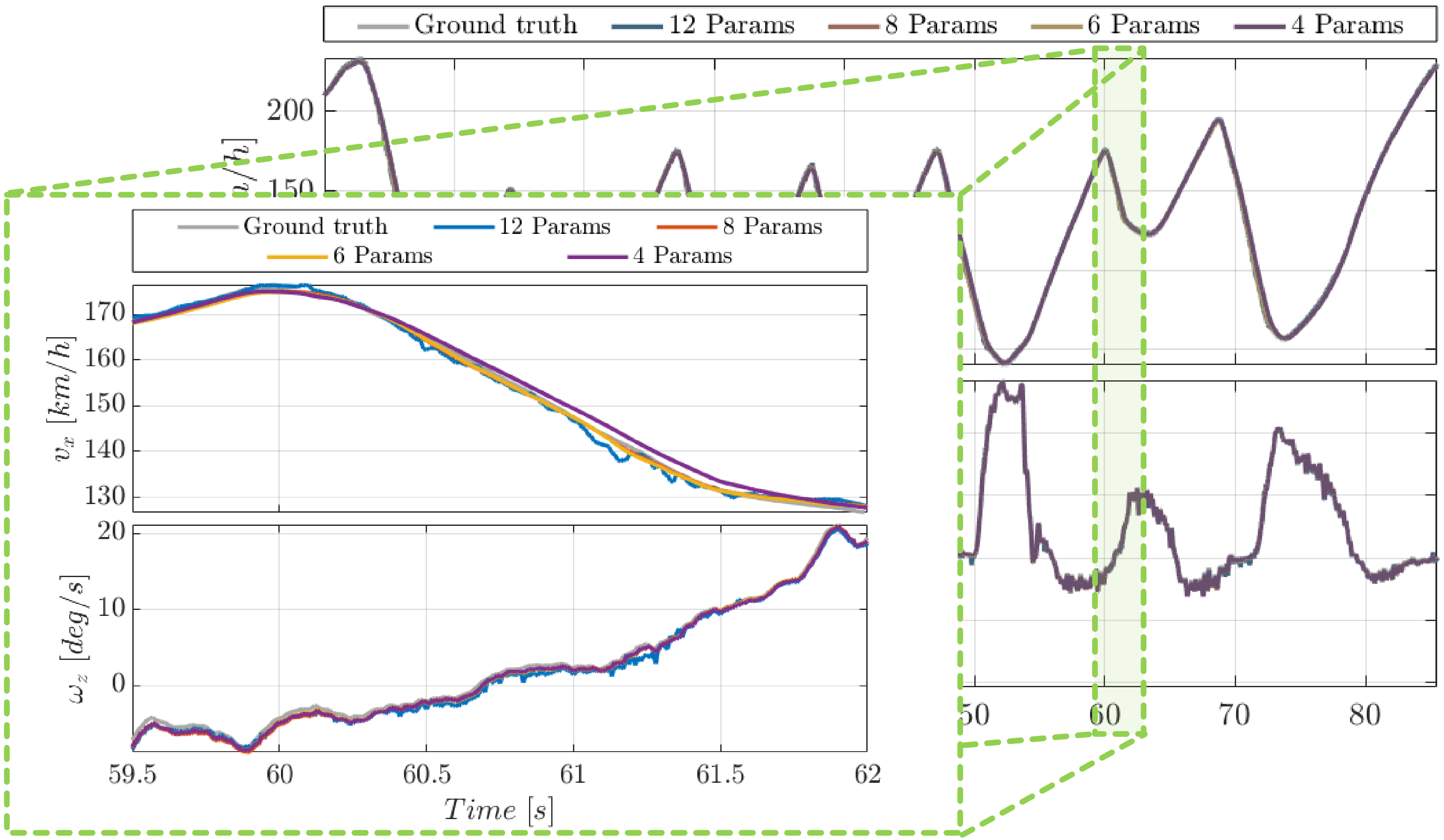




Validation comparison - Zoom

We can now compare a smaller section of the validation lap:

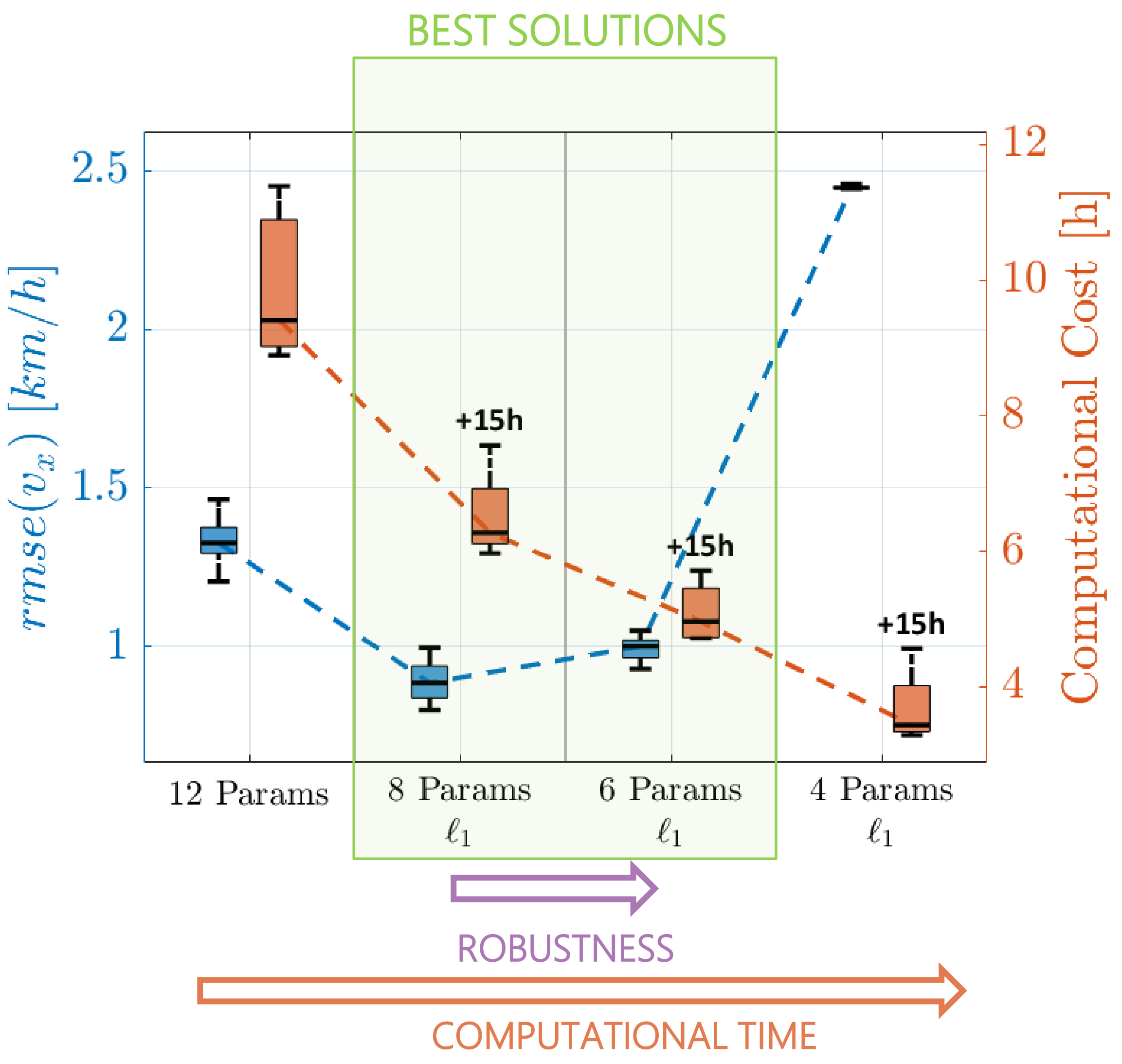
- 12 Parameters: very **noisy**
- 7 and 5 parameters: much **smoother**
- 3 parameters: smooth but **inaccurate**



Validation Comparison

The **same conclusions** of the previous case can be also drawn in this case:

- Increasing too much the number of parameters leads to noisy estimates and not converging solutions
- Decreasing too much the number of parameters leads to solutions which are too smooth and lose information
- The correct number of parameter is most probably 8-6



Principal Component Analysis

To improve the v_x estimate, we can **add the $\boldsymbol{a_x}$ measurement**. In this way:

Corrected **states**:

- v_x
- ω_{fl}
- ω_{rr}
- ω_z

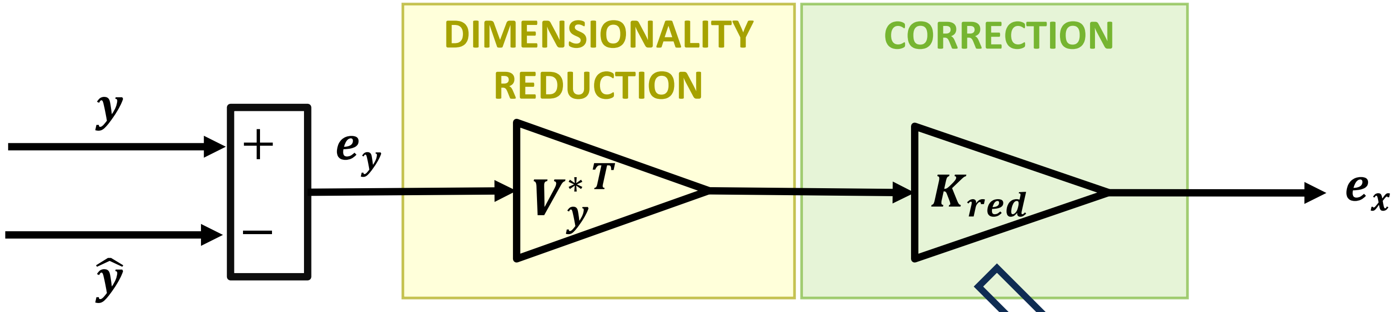
Used **outputs**:

- ω_{fl}
- ω_{rr}
- ω_z
- a_x

$\Rightarrow$ 16 parameters!
Numerically intractable with BO

$\Downarrow$

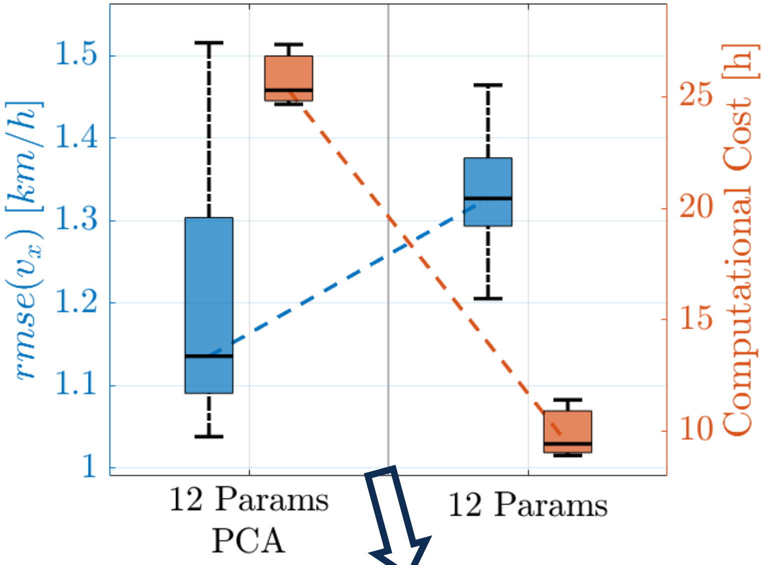
We can use Principal Component Analysis (PCA) to reduce the outputs from 4 to 3

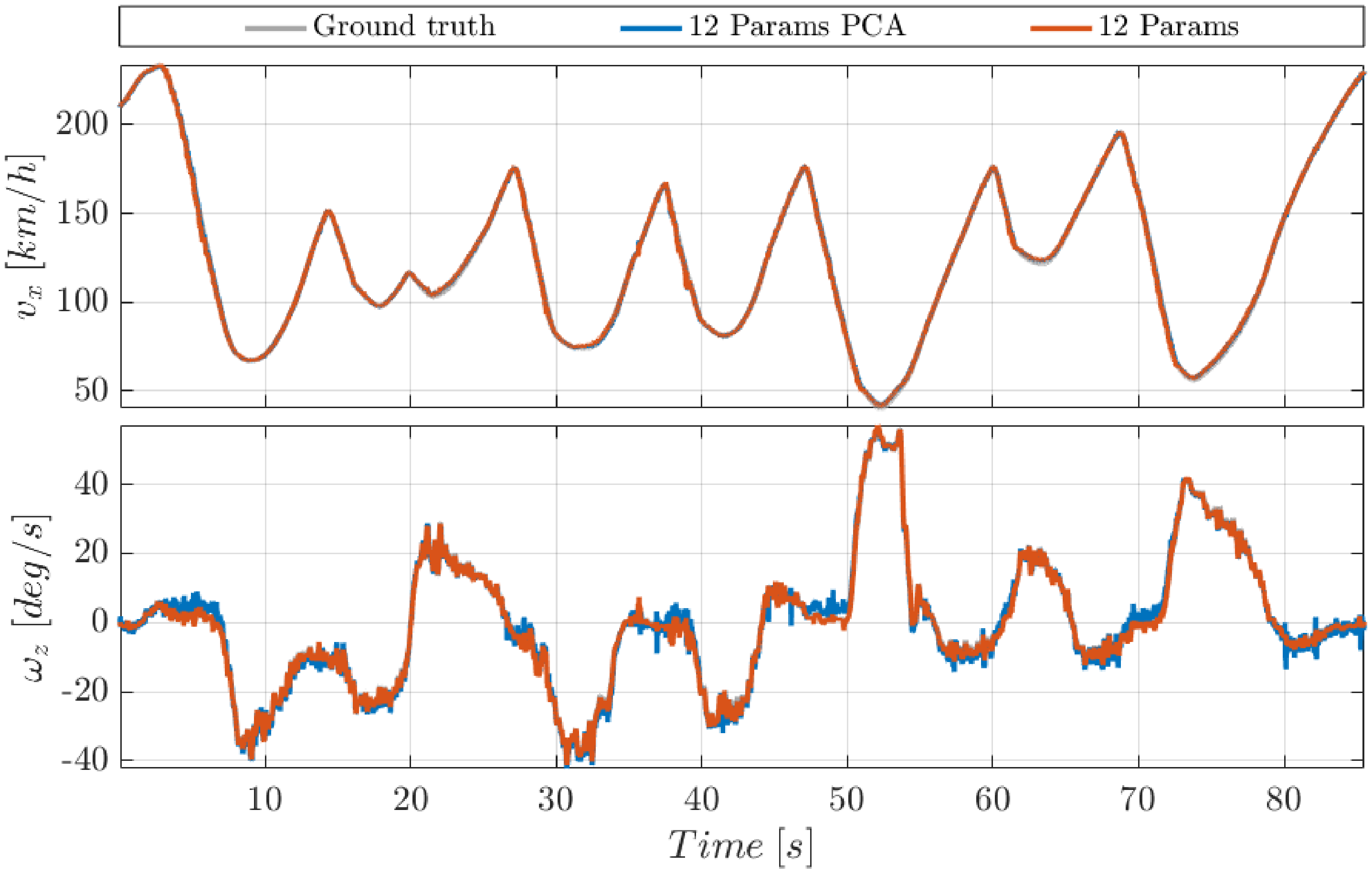


We optimize $\boldsymbol{K_{red}}$ and not $\boldsymbol{K}$

All validations

VALIDATION DATASET:
- 16→12 Params with PCA
- 12 Params (original)



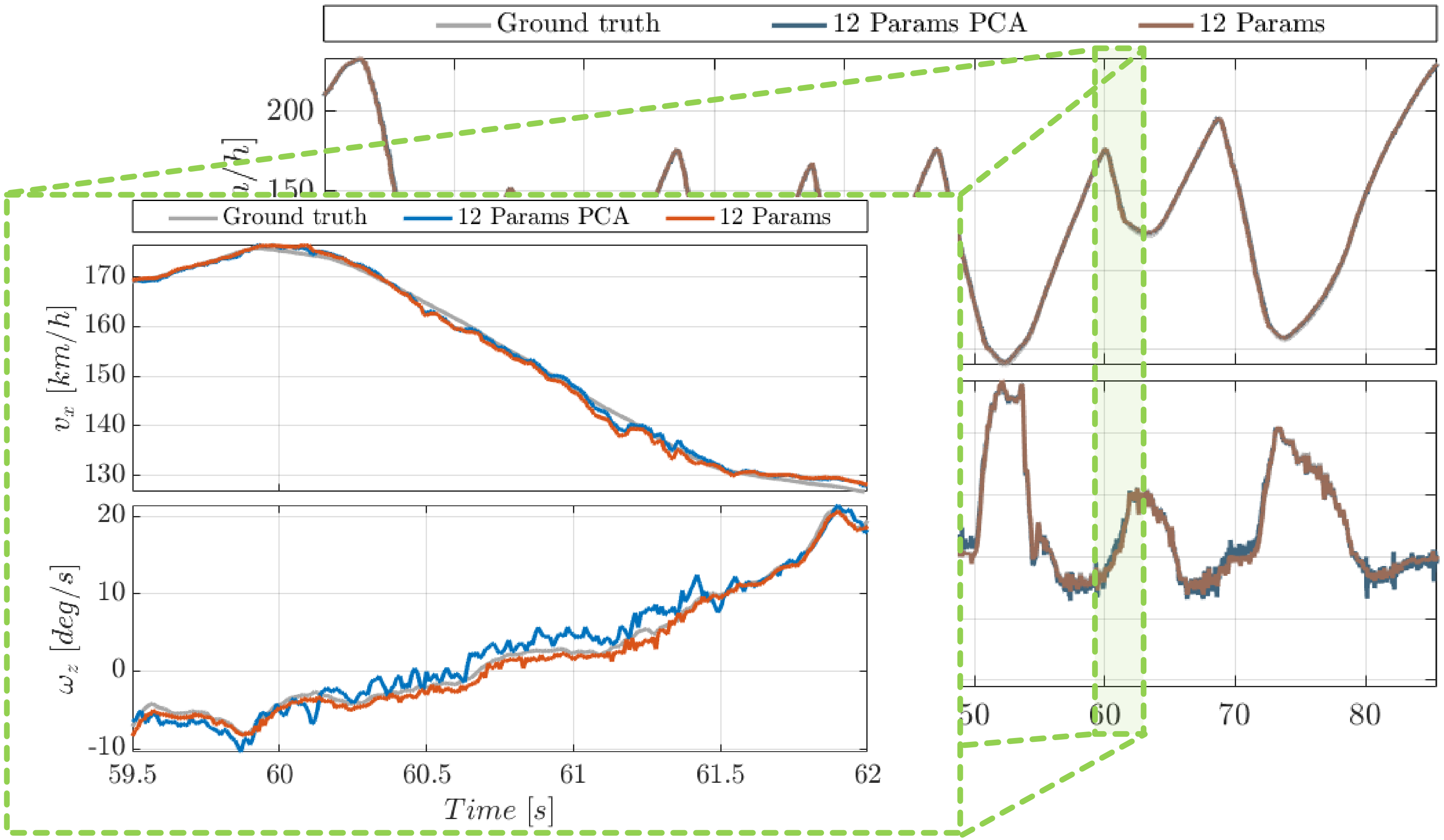


Unsupervised reduction:
- Better average result
- Larger variance
- Higher comp. cost
- More unpredictable

Validation comparison - Zoom

We can now compare a smaller section of the validation lap:

- in terms of v_x the datasets seem to be quite similar
- In terms of ω_z the PCA optimization performs worse



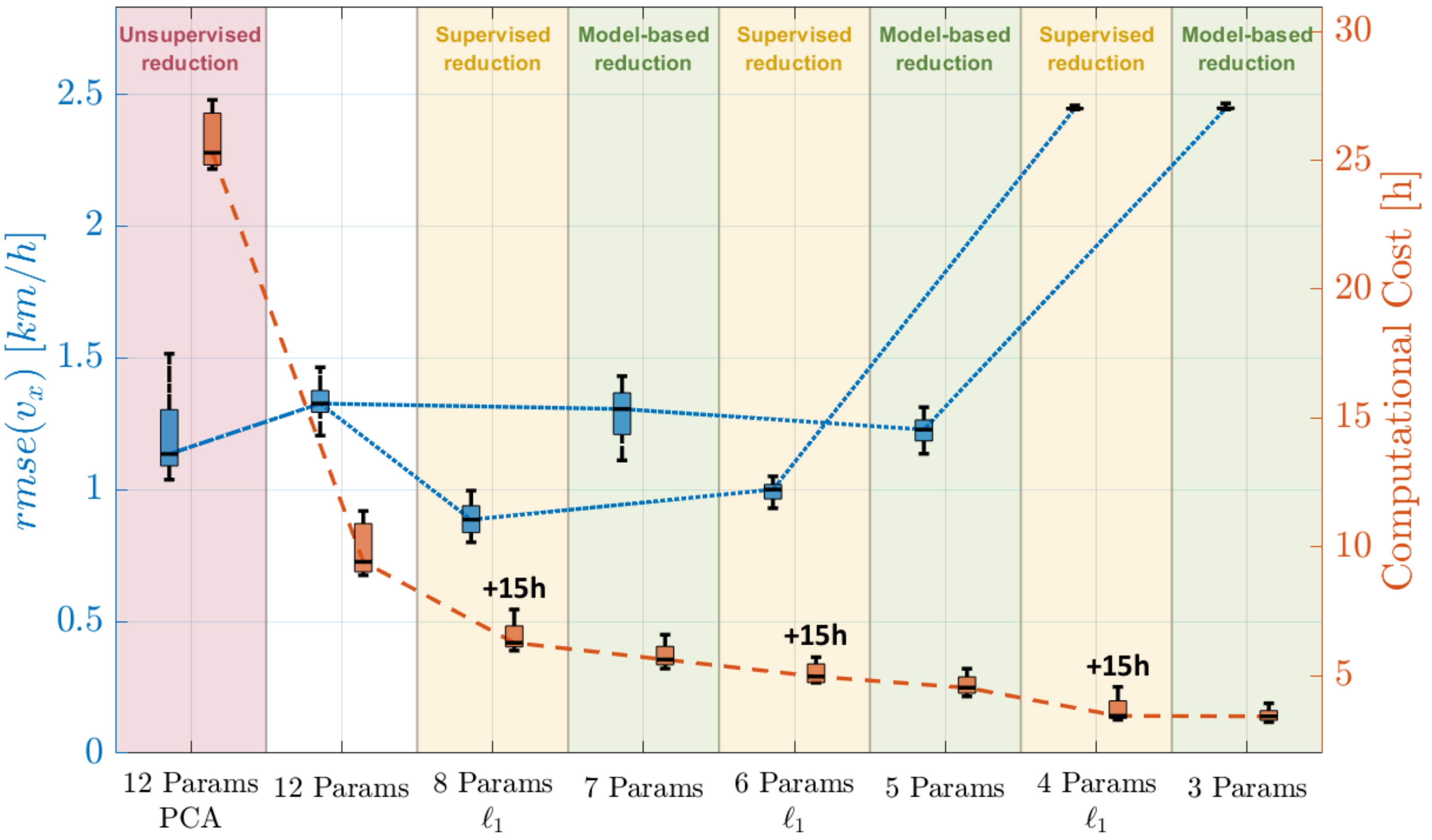
Model-based VS supervised reduction

1) Model-based Reduction
   - Requires **a-priori knowledge** on the system
   - We are dealing with a complex **black-box** simulator, hence some of the internal behaviors can be **behave differently** from the expected ones

2) Supervised Dimensionality Reduction
   - Fully Data-Driven approach, theoretically it **does not need any a priori knowledge**
   - The **results** are **very close if not better** than the model-based approach
   - Only one hyper-parameter to tune: δ which allows to select:
     - Filter complexity
     - Filter robustness
     - Computational complexity
     - Filter performance

Complete overview

- Supervised and Model-based reductions have the same trade-off between convergence and accuracy of results depending on the number of parameters
- Supervised reductions seems to have the best overall results
- Unsupervised reductions perform good but have the largest variance

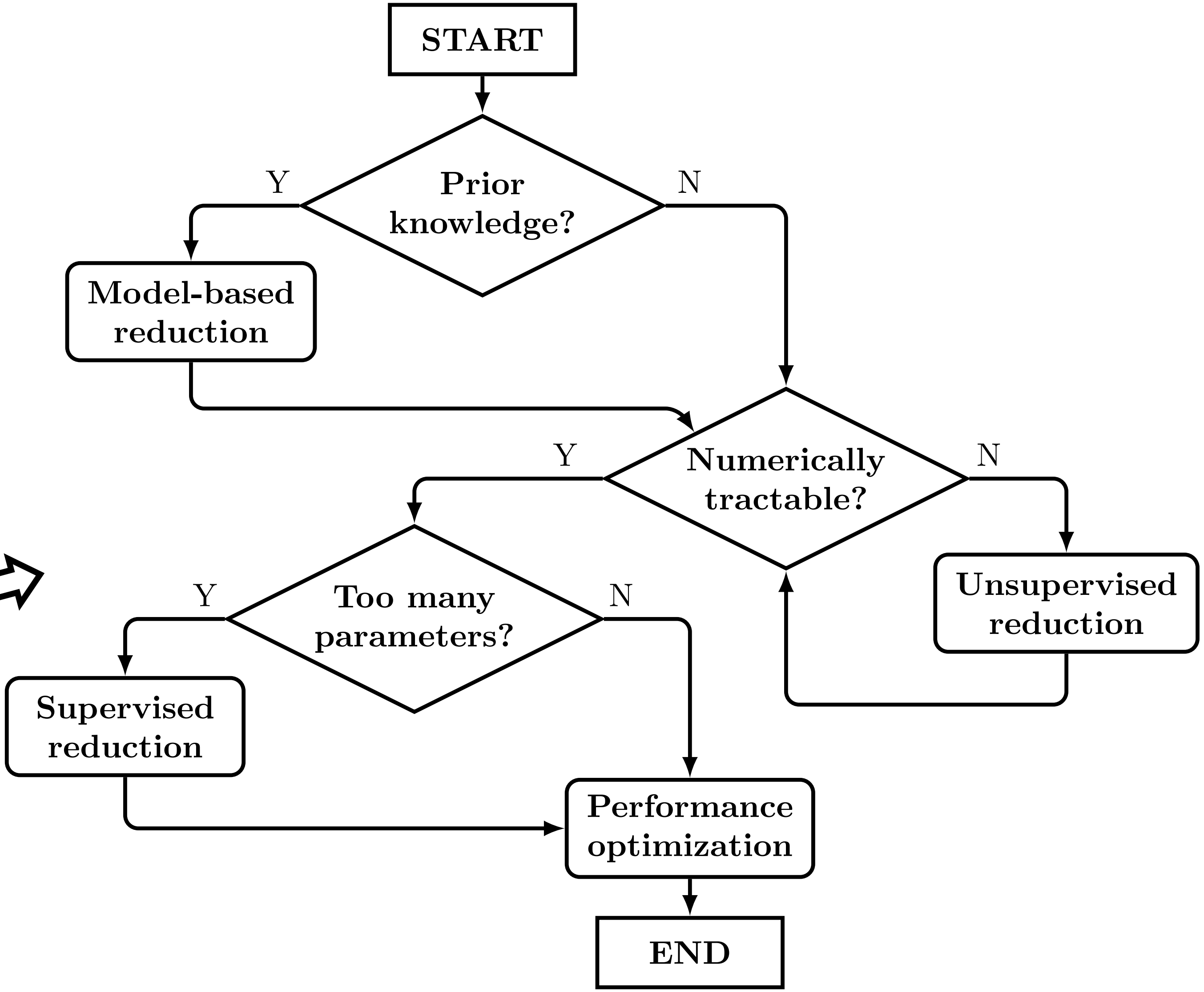


Flow diagram

We have found that:

- Unsupervised reduction will be used only if the problem is numerically intractable
- Supervised reduction performs very well

General procedure for dimensionality reduction of large-scale optimization problems in Twin-in-the-loop estimation



Outline

Problem statement

Twin-in-the-Loop control

Twin-in-the-Loop estimation

Conclusions

Conclusions

- The TiL approach significantly simplifies the End-of-Line tuning of filters and controllers:
  - ✓ Simpler system structures (with less parameters)
  - ✓ Few dedicated experiments via active learning
- More opportunities offered by the new framework:
  - ✓ A single estimator for many signals
  - ✓ Nonlinear state-feedback control even if states are not available
- Challenges:
  - ✓ The single estimator may require too many parameters
  - ✓ Robustness properties of TiL schemes still under investigation
  - ✓ Evaluation on different case studies
- Future works:
  - ✓ Robust solutions to the above challenges
  - ✓ Formal properties
  - ✓ Different optimization algorithms

Acknowledgements

POLITECNICO
MILANO 1863

THANK YOU!!!