Gaussian Process based Model Predictive Control

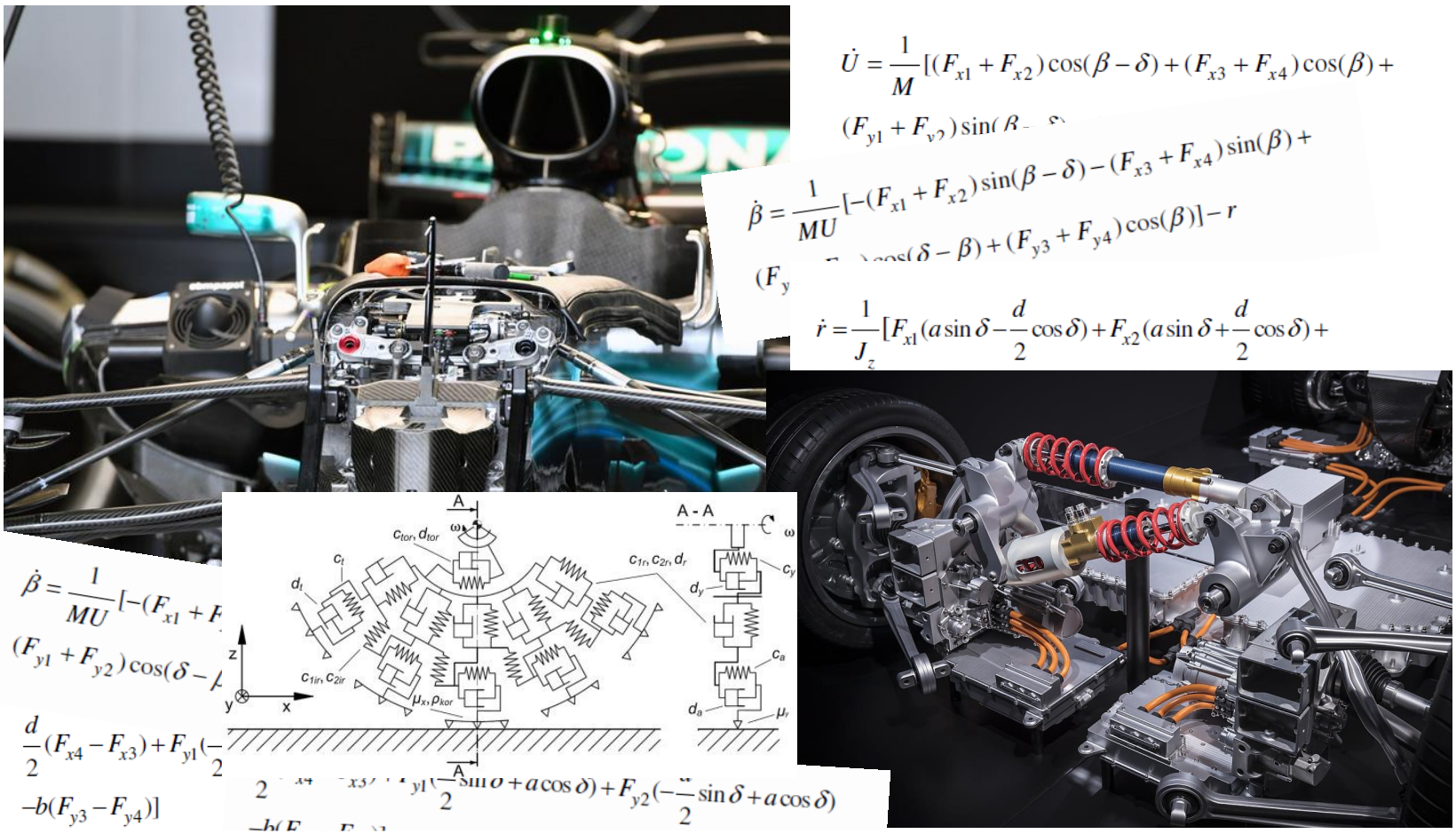
Statistical Learning and Stochastic Control

Lucas Rath Luzia Knödler Dimitrios Gkoutzos

Institute for Systems Theory and Automatic Control
University of Stuttgart

February 4, 2020

• Motivation



[1], [2], [3], [7]

• Table of Contents

• Overview Gaussian Process-based Model Predictive Control

Model Predictive Control (MPC)

Model of Plant Dynamics
(Identification is challenging due to complex dynamics, unknown parameters)

Learning the **full plant dynamics** using Gaussian process (GP) regression

Generation of **local GPs**:

for each subspace of the GP input space a different GP is identified

simple and fixed **nominal model**
+
Learning **disturbance model** using GP regression

The disturbance model represents the error between the true behaviour of the plant and the nominal model

For more information see [8], [6], [5], respectively.

• Overview Gaussian Process-based Model Predictive Control

Model Predictive Control (MPC)

Model of Plant Dynamics
(Identification is challenging due to complex dynamics, unknown parameters)

Learning the **full plant dynamics** using Gaussian process (GP) regression

Generation of **local GPs**:

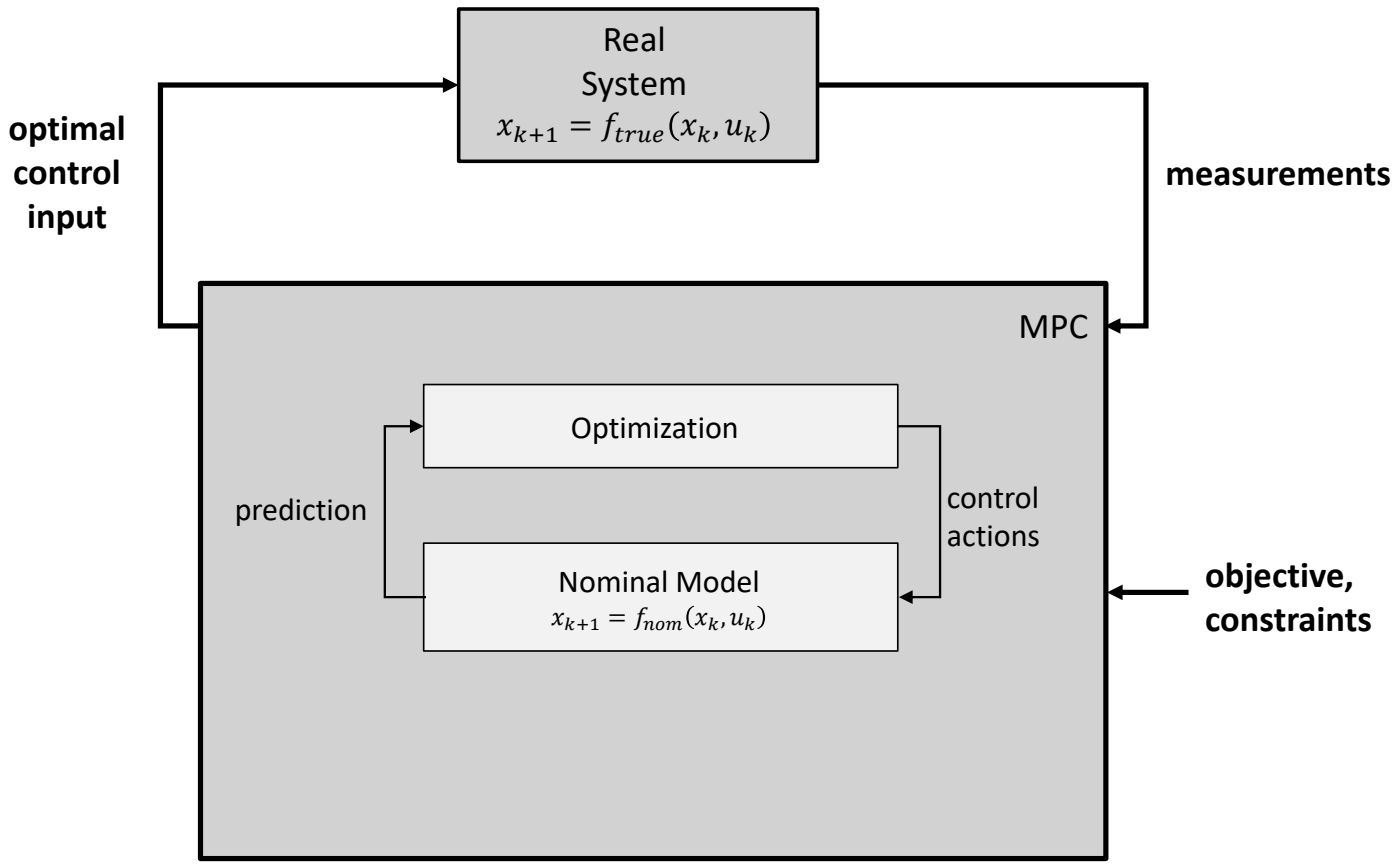
for each subspace of the GP input space a different GP is identified

simple and fixed **nominal model**
+
Learning **disturbance model** using GP regression

The disturbance model represents the error between the true behaviour of the plant and the nominal model

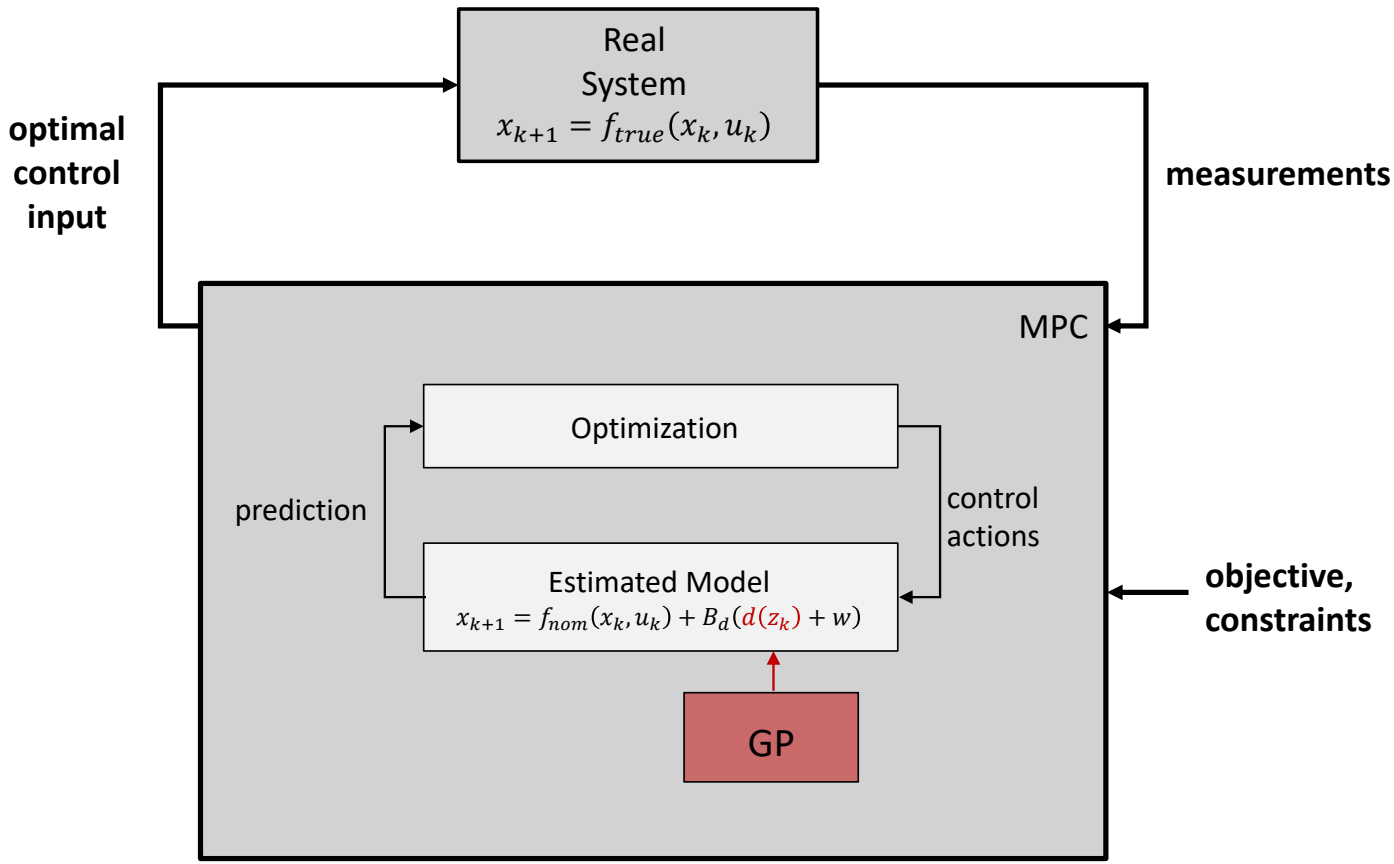
For more information see [8], [6], [5], respectively.

• Structure MPC



Based on [4]

• Structure GP-based MPC



Based on [4]

• Standard Nonlinear MPC (NMPC)

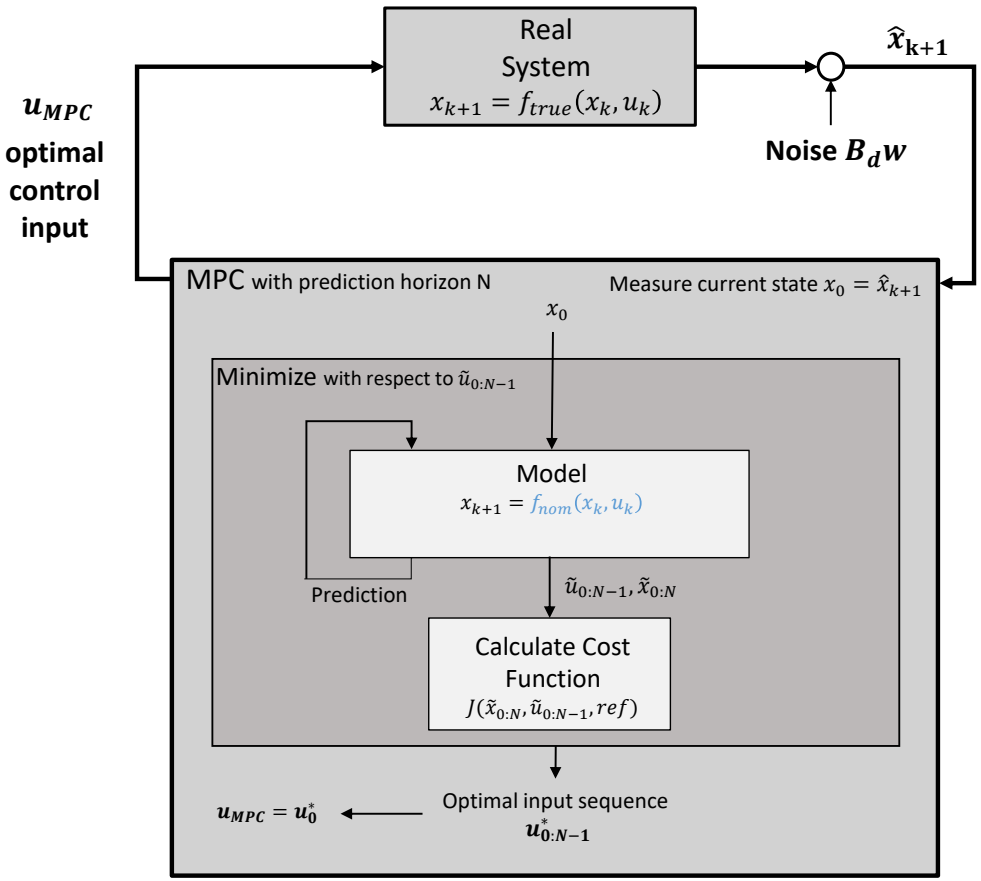
NMPC

$$
\begin{aligned}
u^*_{0:N-1}, x^*_{0:N} = \underset{x_{0:N}, u_{0:N-1}}{\arg\min} \quad & \sum_{k=0}^{N-1} f_o(x_k, u_k) + \phi(X_N) \\
s.t. \quad & x_{k+1} = f_{nom}(x_k, u_k) \\
& x_0 = \bar{x} \\
& x_k \in \mathcal{X}(x_k) \\
& x_N \in \mathcal{X}_f \\
& u_k \in \mathcal{U}(x_k)
\end{aligned} \tag{1}
$$

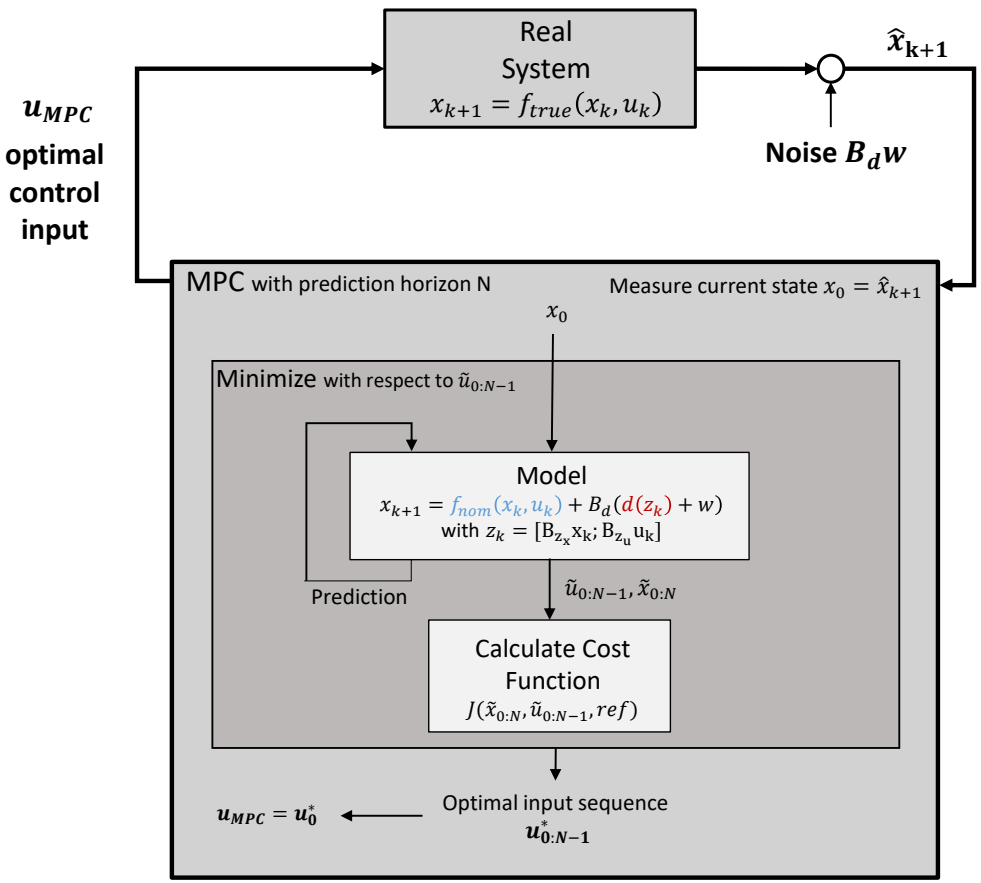
NMPC Algorithm:

1) Measure (estimate) current state $\bar{x}$
2) Solve the open-loop discrete-time finite-horizon optimal control problem (1)
3) Implement the first portion of the optimal input $u_{NMPC}(\bar{x}) = u^*_0$
4) Go to 1)

• Overview Standard NMPC



Overview GP-based NMPC



• Learning-based Prediction

Idea: Can we use the uncertainty knowledge from GP and propagate uncertainties over predictions?

$$x_{k+1}(x_k, u_k) = f_{nom}(x_k, u_k) + B_d\ (d(x_k, u_k) + w) \tag{2}$$

Problem: Transformation $x_{k+1}(x_k, u_k)$ is nonlinear and also depends on a Gaussian process.

Solution: Extended Kalman Filter and GP marginalization

$$x_{k+1}(x_k, d, w) \approx x_{k+1}(\mu_{x_k}, \mu_d, \mu_w) + \nabla x_{k+1}(\mu_{x_k}, \mu_d, \mu_w)^{\mathsf{T}} \begin{bmatrix} x_k - \mu_{x_k} \\ d - \mu_d \\ w - \mu_w \end{bmatrix}$$

which implies that:

$$\mathbb{E}[x_{k+1}] \approx f_{nom}(\mu_{x_k}, \mu_d, \mu_w) + B_d\, \mu_d(\mu_x)$$

$$\mathbb{V}ar[x_{k+1}] \approx \begin{bmatrix} \nabla_{x_k} f_{nom}(\mu_{x_k}, \mu_d, \mu_w) \\ B_d^{\mathsf{T}} \\ B_d^{\mathsf{T}} \end{bmatrix}^{\mathsf{T}} \mathbb{V}ar \begin{bmatrix} x_k \\ d \\ w \end{bmatrix} \begin{bmatrix} \nabla_{x_k} f_{nom}(\mu_{x_k}, \mu_d, \mu_w) \\ B_d^{\mathsf{T}} \\ B_d^{\mathsf{T}} \end{bmatrix}$$

• Learning-based NMPC Formulation

Learning-based NMPC

$$
\begin{aligned}
u^*_{0:N-1}, x^*_{0:N} = \underset{x_{0:N}, u_{0:N-1}}{\arg\min} \quad & \sum_{k=0}^{N-1} f_o(x_k, u_k) + \phi(X_N) \\
s.t. \quad & x_{k+1} = f_{nom}(x_k, u_k) + B_d\ (d(z_k) + w) \\
& x_0 = \bar{x} \\
& x_k \in \mathcal{X}(x_k) \\
& x_N \in \mathcal{X}_f \\
& u_k \in \mathcal{U}(x_k)
\end{aligned}
\tag{3}
$$

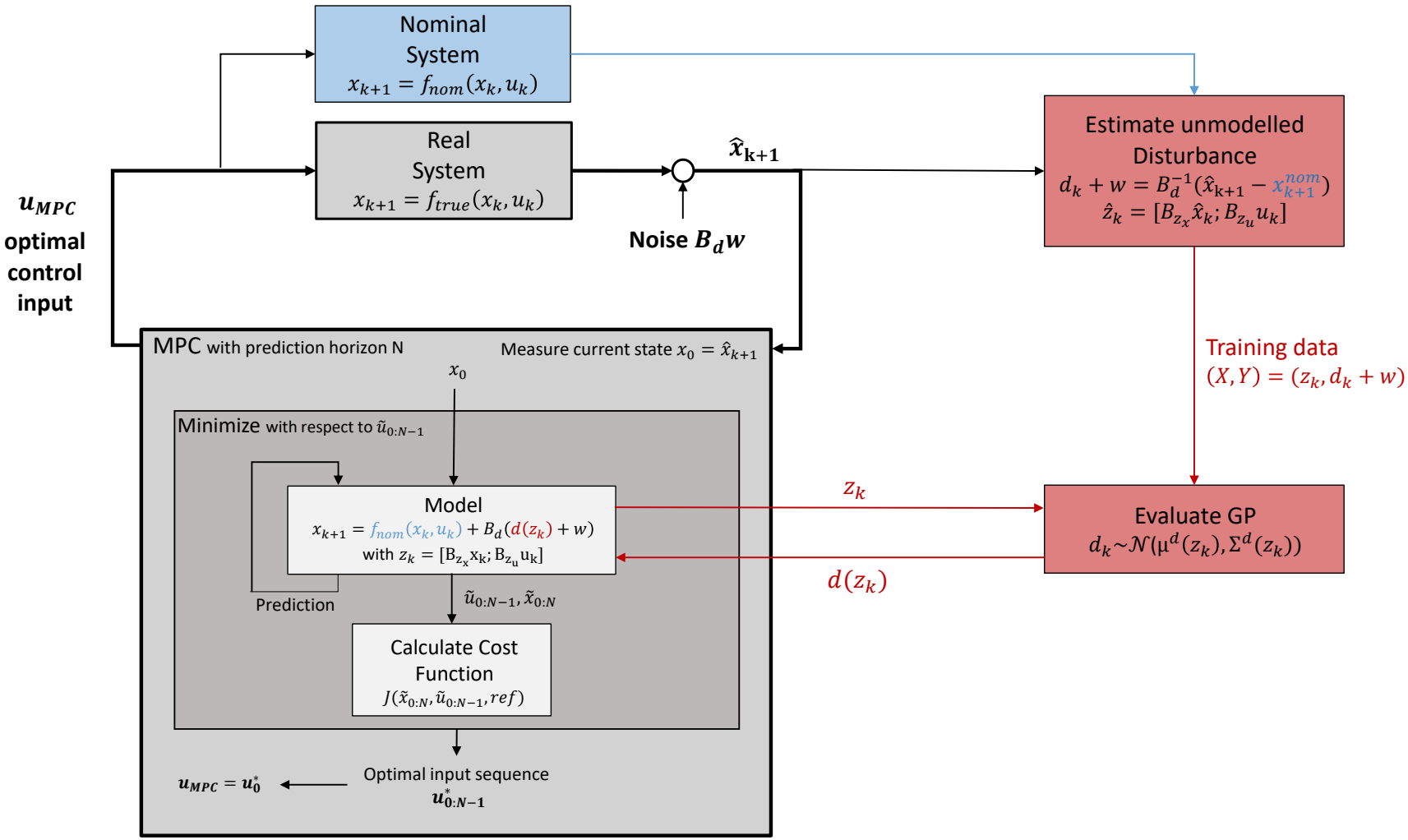
with

$$
\begin{aligned}
d(z_k) &\sim GP(m(z_k), k(z_k, z'_k)) \\
z_k &= [B_{z_x} x_k; B_{z_u} u_k] \in \mathbb{R}^{n_z} \\
w &\sim \mathcal{N}(0, \sigma_n^2) \text{ process noise}
\end{aligned}
$$

The GP function $d(z_k)$ will be responsible for capturing the unmodeled dynamics

Learning-based NMPC Algorithm as before

• Learning-based NMPC Overview



Generating GP Training Data

Given the adaptive process model

$$x_{k+1} = f_d(x_k, u_k) + B_d\ (d(z_k) + w) \tag{4}$$

Assume that the state observation $\hat{x}_{k+1}$ at time k+1 can be perfectly measured.

This means that the training data (Z, Y) can be generated to be the difference between the true measured and the nominal simulated states:

$$\begin{aligned} Z &= z_k \\ Y &= B_d^{\dagger}\,(x_{k+1} - f_{nom}(x_k, u_k)) \;=\; d_{true} + w \end{aligned} \tag{5}$$

GP Enhancements

GP Model Enhancements:

- Efficient implementation - faster model predictions
- Hyper-parameter optimization
- Sparse training data (Sparse GP)

• Efficient GP Implementation

Problem: The computational complexity of a GP regression strongly depends on the number of data points N. The most computational demanding task is the matrix inversion: $(K(X,X)+I\sigma_n^2)^{-1}$.

$$f^*|Z^*,Y,Z \sim \mathcal{N}\left(\begin{array}{c} m(Z^*)+K(Z^*,Z)\left(K(Z,Z)+I\sigma_n^2\right)^{-1}(Y-m(Z)) \\ K(Z^*,Z^*)-K(Z^*,Z)\left(K(Z,Z)+I\sigma_n^2\right)^{-1}K(Z,Z^*) \end{array}\right) \quad (6)$$

Solution: Use *Cholesky* decomposition: $(x=A^{-1}b \Leftrightarrow x=L^{\mathsf{T}}\backslash(L\backslash b), A=LL^{\mathsf{T}})$

$$\begin{aligned} L &= cholesky(K(Z,Z)+\sigma_n^2 I) \\ \alpha &= L^{\mathsf{T}}\backslash(L\backslash(Y-m(Z))) \\ v &= L\backslash K(Z,Z^*) \end{aligned}$$

$$f^*|Z^*,Y,Z \sim \mathcal{N}\left(\begin{array}{c} m(Z^*)+K(Z^*,Z)\alpha \\ K(Z,Z)-v^{\mathsf{T}}v \end{array}\right) \quad (7)$$

Obs: each output dimension is treated as a different GP (with possibly different hyper-parameters as well).

• Hyper-parameter Optimization

The GP Likelihood is Gaussian and in the case of the zero mean function, is given by:

$$Y|Z,\theta \sim \mathcal{N}(0, \underbrace{K(Z,Z) + \sigma_n^2 I}_{K_y}) \tag{8}$$

where $\theta = [\{M\}, \sigma_f^2, \sigma_n^2]$ is a vector containing all hyper-parameters.

Among many possible choices, we chose to parameterize the length-scale covariance matrix M as diagonal positive-semidefinite:

$$M = \begin{bmatrix} l_1 & & 0 \\ & \ddots & \\ 0 & & l_n \end{bmatrix} \tag{9}$$

with $l_i \geq 0, \forall i \in 0, \ldots, n$

such that the hyperparameter vector becomes $\theta = [l_1, \ldots, l_n, \sigma_f^2, \sigma_n^2]$

• Hyper-parameter Optimization

One may optimize the GP hyper-parameters by maximizing the Log Likelihood (LL):

$$log\, p(Y|Z,\theta) = -\frac{1}{2} y^T K_y^{-1} y - \frac{1}{2}\log|K_y| - \frac{n}{2}\log(2\pi)$$

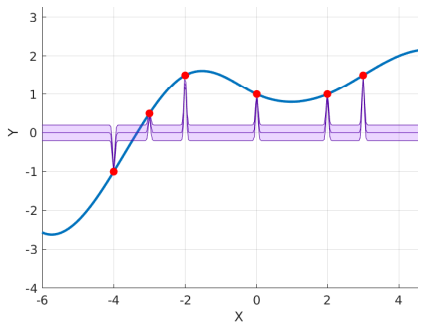
$$\theta = \arg\max_{\theta}\ log\, p(Y|Z,\theta) \tag{10}$$

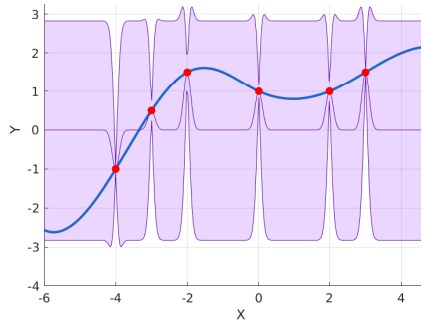
which allows local optimization of the hyper-parameters, since (10) is nonconvex (one optimization problem for each output dimension).

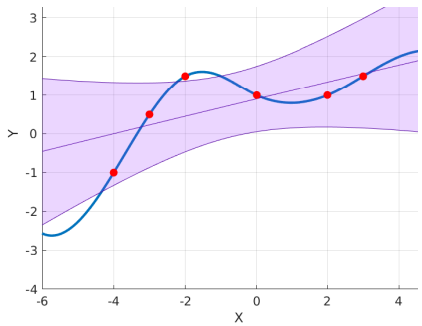
Obs: Even gradient-free tools like `fmincon` from Matlab, showed to be efficient. Nevertheless, the gradient of (10) can be easily derived, as shown in [?]

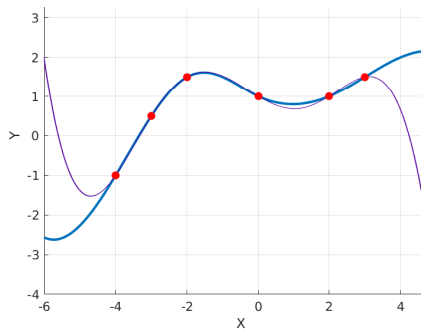
• Hyper-parameter Optimization

Different hyper-parameters might lead to completely different Gaussian processes

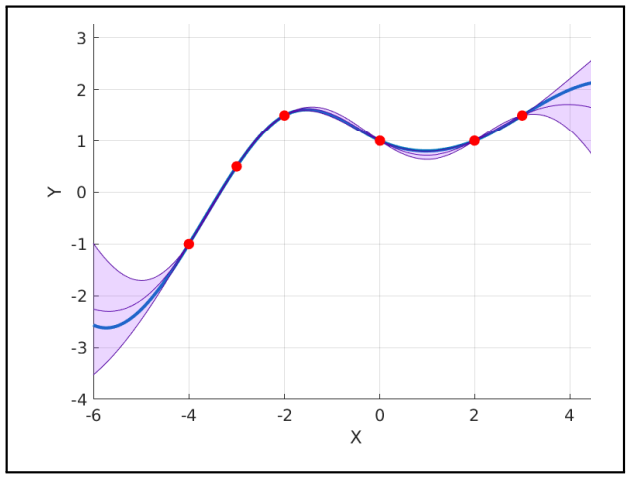








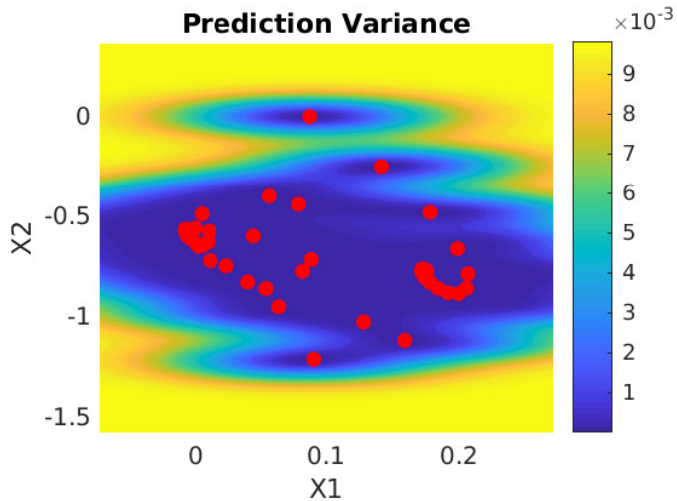
After hyper-parameter optimization



• Sparse Gaussian Process Regression

Problem: The computational complexity of a GP regression strongly depends on the number of data points N. How to keep N low, while ensuring good prediction performance?

Solution: (i) Set a size limit for the dictionary. (ii) If dictionary is full, make a selection of the so called, *inducing points*.



Strategy A) Choose the closest point in the Euclidean space to the new point to be replaced.

Strategy B) Add new points, update model and remove the points with the lowest covariances.

• Dynamical Models

Cart-pole Dynamics

Continuous time dynamics:

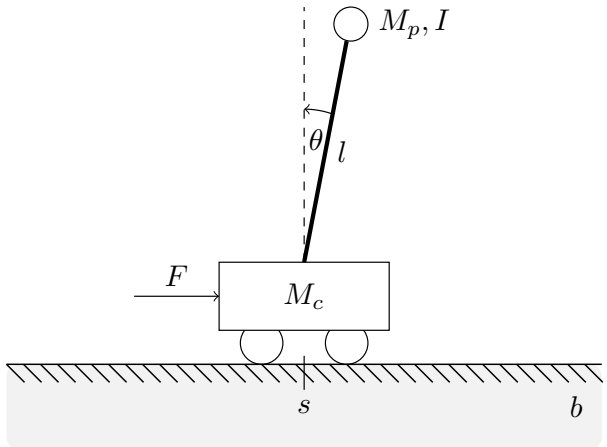
$$(M_c + M_p)\ddot{x} + b\dot{x} + \frac{1}{2}M_p l\ddot{\theta}\cos\theta - \frac{1}{2}M_p l\dot{\theta}^2\sin\theta = F$$

$$\left(I + M_p\left(\frac{l}{2}\right)^2\right)\ddot{\theta} - \frac{1}{2}M_p gl\sin\theta + M_p l\ddot{x}\cos\theta = 0.$$

After making explicit and RK4 discretization:

$$x_{k+1} = f_{nom}(x_k, u_k)$$

$$x = [s, \dot{s}, \theta, \dot{\theta}]^T, \quad u = F$$



True Dynamics

$$x_{k+1} = f_{nom}(x_k, u_k) + B_d\left(d_{true}(z_k) + w\right)$$

$$d_{true}(z_k) = d(\theta, \dot{\theta})$$

$$B_d = [0 \quad 0 \quad 1 \quad 0]$$

Estimated Dynamics

$$x_{k+1} = f_{nom}(x_k, u_k) + B_d(\, d_{GP}(z_k) + w)$$

$$d_{GP}(z_k) \sim \mathcal{GP}(m(x), k(x, x'))$$

$$B_d = [0 \quad 0 \quad 1 \quad 0]$$

Estimated Dynamics and Cost function

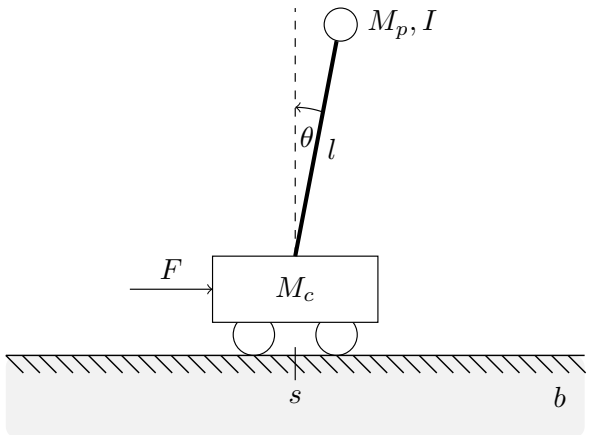
Cost Function

$C \in \mathbb{R}^{3\times 4}$, $Q \in \mathbb{R}^{3\times 3}$ and $R \in \mathbb{R}$

$$f_o(\mu_k^x, u_k) = (C\mu_k^x - r)^T Q(C\mu_k^x - r) + Ru^2$$
$$\phi(x_N) = (C\mu_N^x - r)^T Q(C\mu_N^x - r)$$

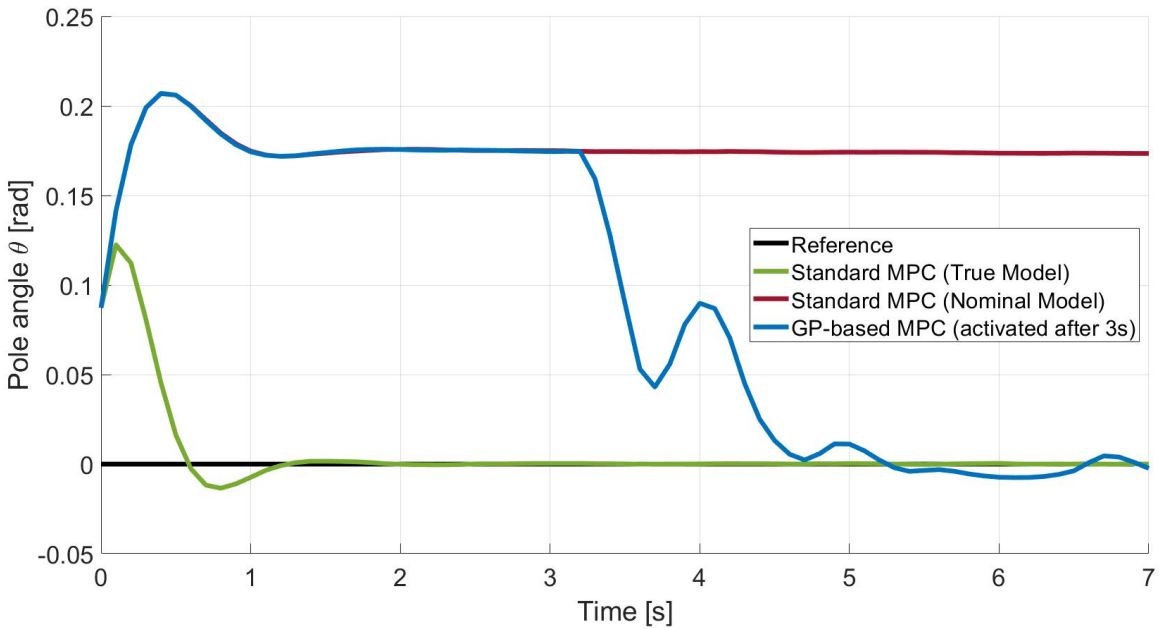
with

$$r = [\dot{s} \;\; \theta \;\; \dot{\theta}]^{\mathsf{T}} = [0 \;\; 0 \;\; 0] \tag{11}$$



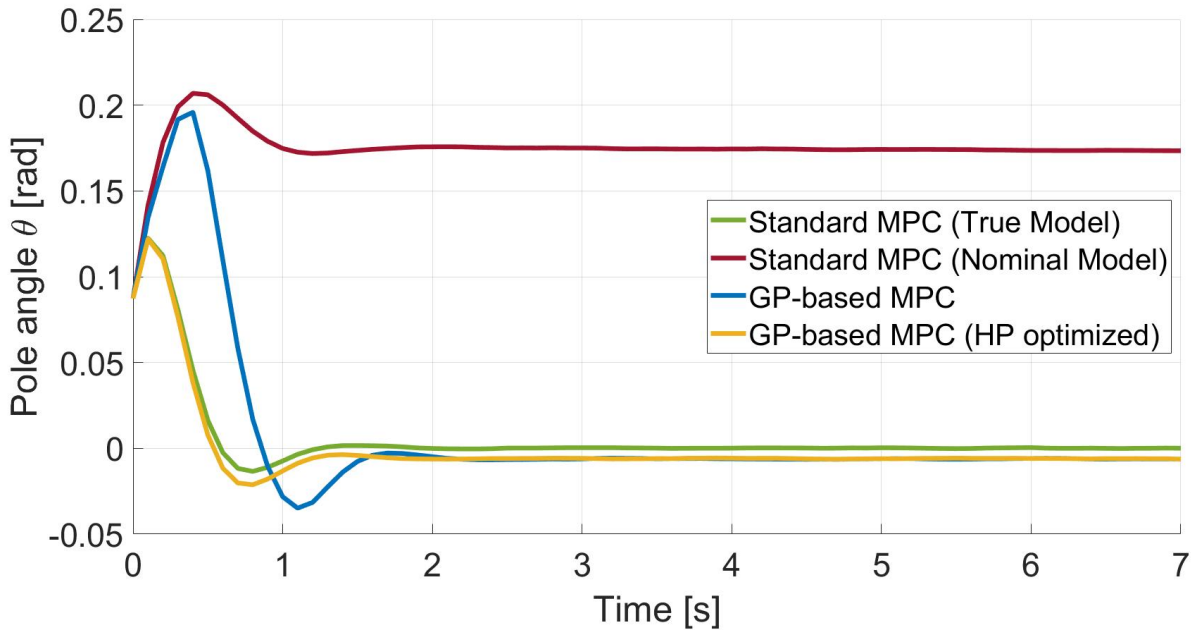
• Experiment 1: Simulation Results

Starting with $\theta = 5°$. GP accumulates data to the dictionary. At t=3s, we activate predictions with GP.



• Experiment 2: Simulation Results

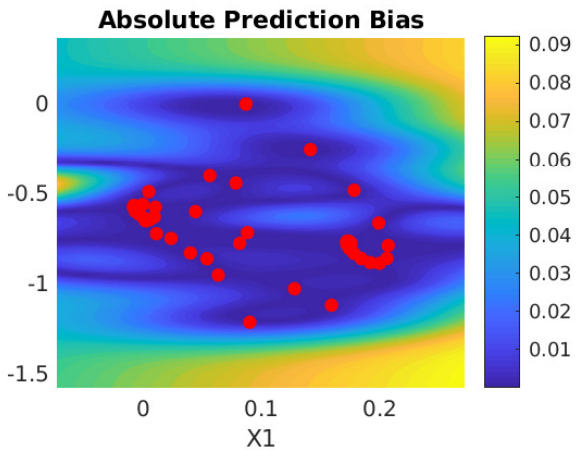
Starting with $\theta = 5°$. GP uses offline generated data. GP is activated from the beginning.

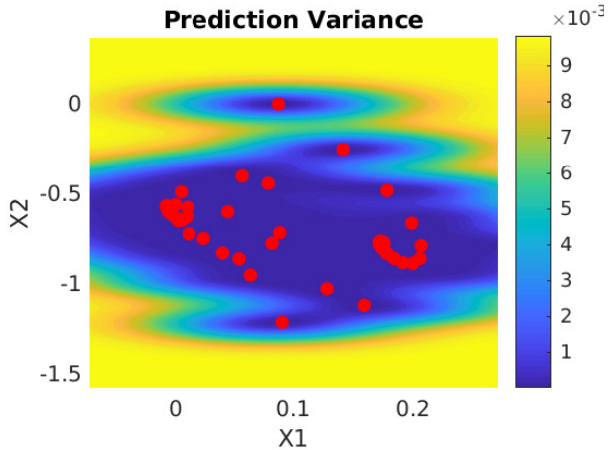


⇒ Hyper-parameter optimization improves overall performance

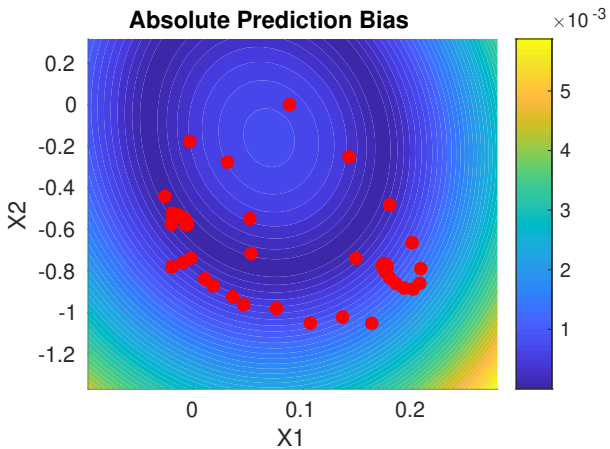
• Comparison with and without hyper-parameter optimization

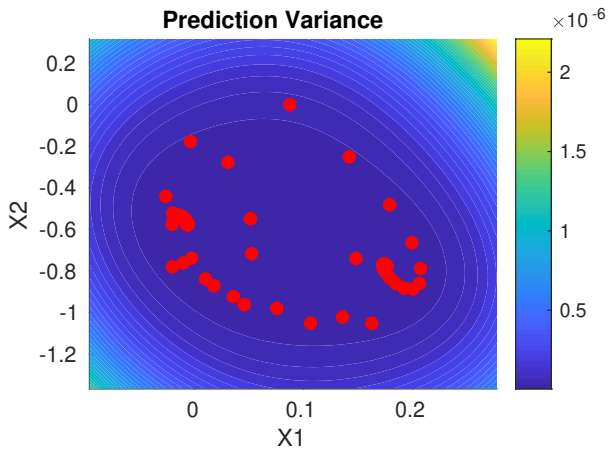
Before Hyper-parameter optimization





After Hyper-parameter optimization

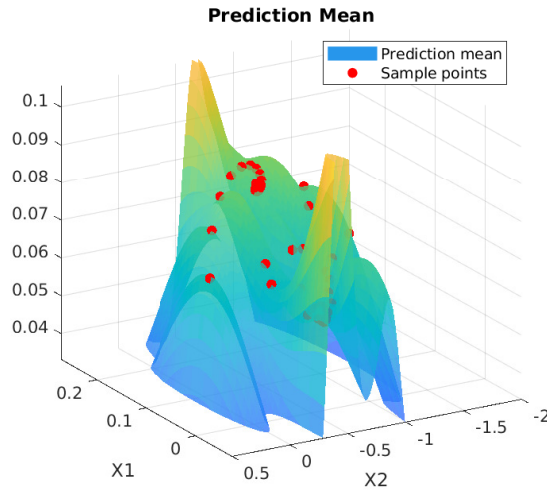




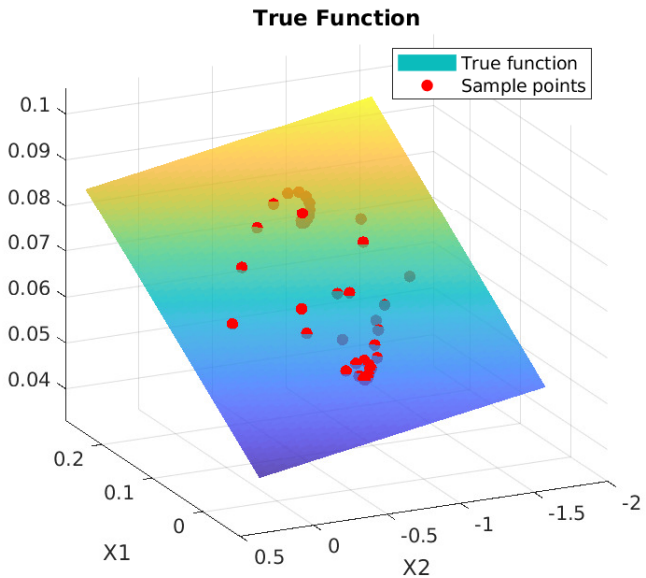
⇒ Hyper-parameter optimization improves overall performance

Training

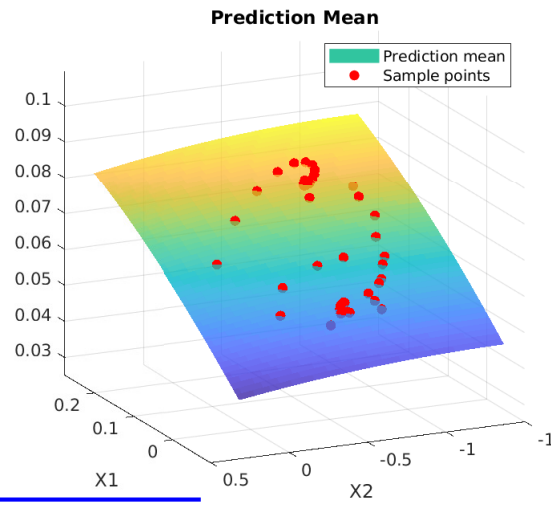
Before Hyper-parameter optimization



True disturbance function

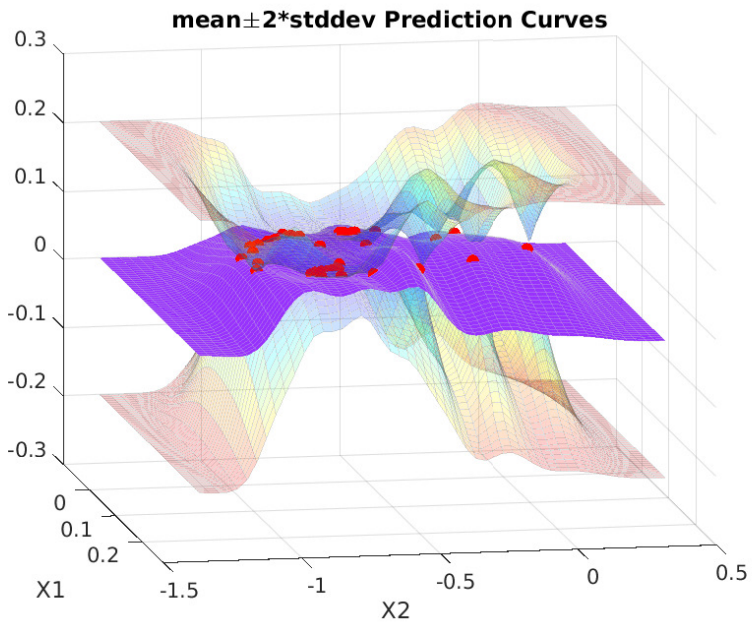


After Hyper-parameter optimization

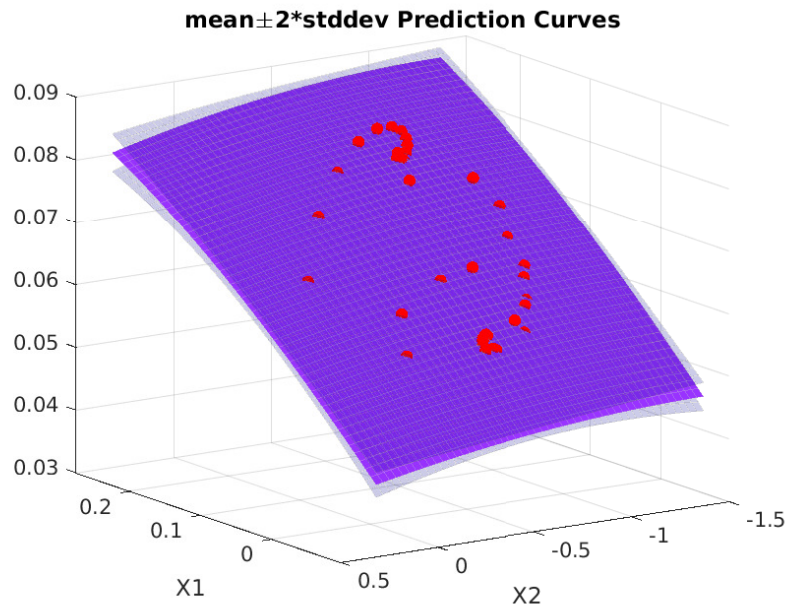


• Training

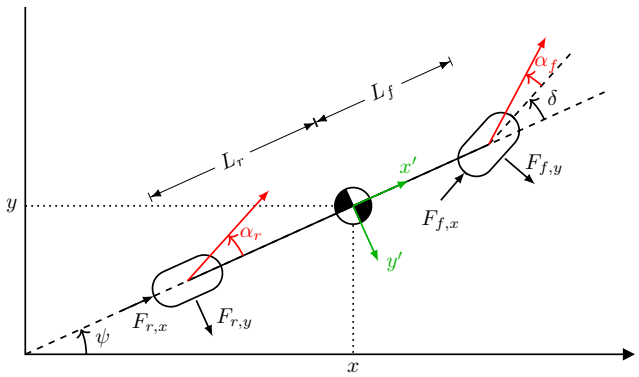
Before Hyper-parameter optimization

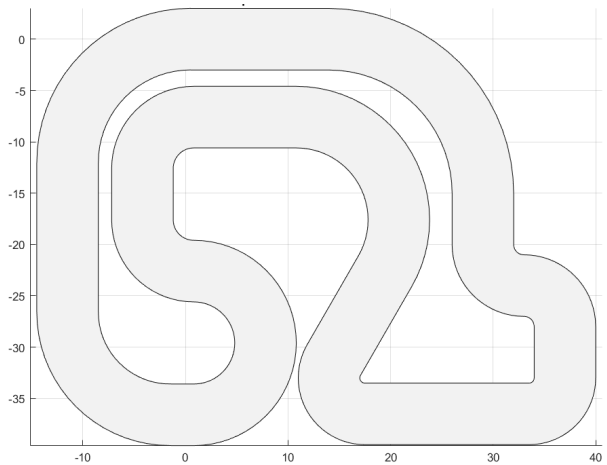


After Hyper-parameter optimization



• Racing Car Problem





- center line coordinates $[x_c, y_c]$
- center line orientation ψ_c
- travelled distance of the vehicle d

States:

$$x = \begin{bmatrix} I_x \\ I_y \\ \psi \\ V_x \\ V_y \\ \dot{\psi} \\ d_{track} \end{bmatrix}$$

x position in global coordinates

y position in global coordinates

yaw angle

longitudinal velocity

lateral velocity

yaw rate

distance travelled along the track center line

Inputs:

$$u = \begin{bmatrix} \delta \\ T \\ v_{track} \end{bmatrix}$$

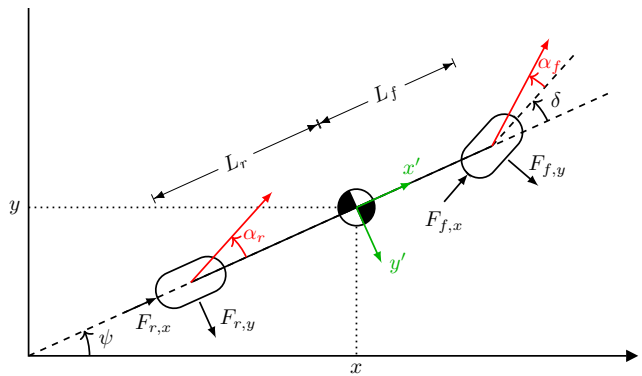
steering angle

wheel torque gain

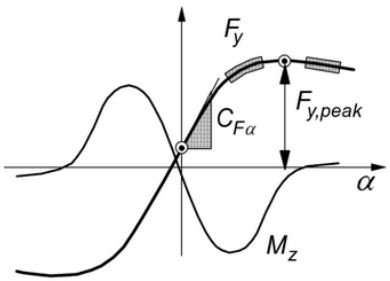
track center line velocity

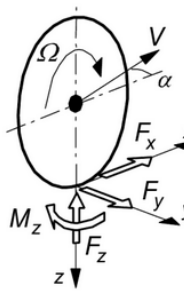
• Basic Vehicle Dynamics

$$\dot{x} = \begin{bmatrix} V_x' \cos(\psi) - V_y' \sin(\psi) \\ V_x' \sin(\psi) + V_y' \cos(\psi) \\ \dot{\psi} \\ \frac{1}{M}(F_{r,x'} + F_{f,x'} * \cos(\delta) - F_{f,y'} \sin(\delta) + V_{y'}\dot{\psi}) \\ \frac{1}{M}(F_{r,y'} + F_{f,x'} \sin(\delta) + F_{f,y'} \cos(\delta) - V_{y'}\dot{\psi}) \\ \frac{1}{I}(F_{f,y'} L_f \cos(\delta) + F_{f,x'} L_f \sin(\delta) - F_{r,y'} L_r) \\ v_{track} \end{bmatrix}$$

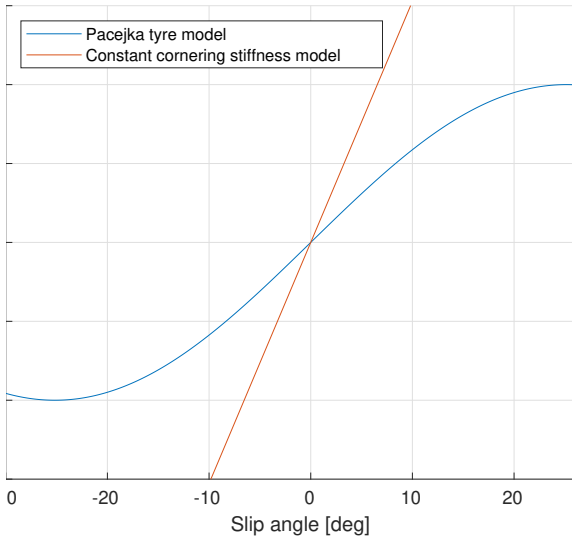


• Differences between True and Nominal Dynamics





Tyre forces. Pacejka, Hans. Tire and vehicle dynamics. Elsevier, 2005.



True Dynamics (Pacejka Tyre Model for Front and Rear Tyre)

$$F_y = D \sin\Big[C \arctan\Big(B\alpha - E\big(B\alpha - \arctan(B\alpha)\big)\Big)\Big]$$

with stiffness factor B, peak factor D and shape factors C and E as well as slip angle α

Nominal Dynamics (Linear Tyre Model)

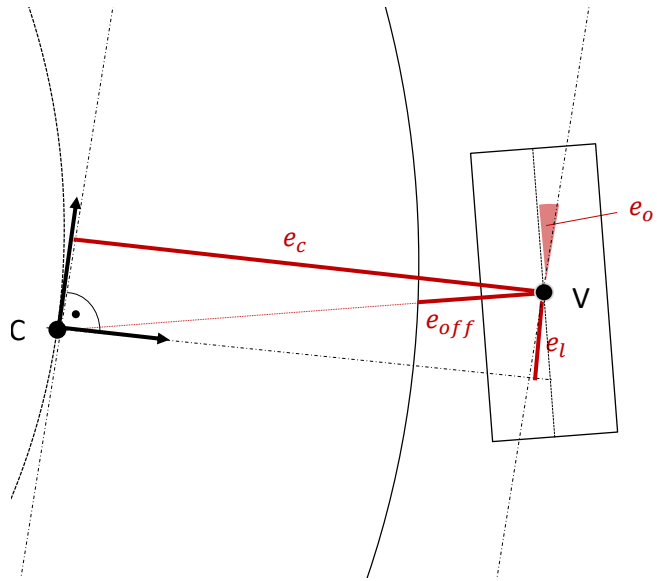
$$F_y = c\alpha$$

with cornering stiffness c and slip angle α

Cost Function

Components of Cost Function

$$
\begin{aligned}
f_{o_{contour}} &= q_c \cdot \Delta^2_{contour} \\
f_{o_{lag}} &= q_l \cdot \Delta^2_{lag} \\
f_{o_{orientation}} &= q_o \cdot \Delta^2_{orientation} \\
f_{o_{outside}} &= \cdots \text{Applying smooth barrier function to offroad error} \\
f_{o_{\dot{\psi}}} &= q_{\dot{\psi}} \cdot \dot{\psi}^2 = q_{\dot{\psi}} \cdot \mu^x_k(6)^2 \\
f_{o_{dist}} &= q_d \cdot v_{track} = q_d \cdot u(3)
\end{aligned}
$$



Cost Function

$$f_o(\mu^x_k, u_k) = f_{o_{contour}} + f_{o_{lag}} + f_{o_{orientation}} + f_{o_{outside}} + f_{o_{\dot{\psi}}} \tag{12}$$

$$J(\bar{x}, u^*_{0:N-1}) = \sum_{k=0}^{N-1} \min_{u_{0:N-1}} \max_{v_{track,0:N-1}} f_o(\mu^x_k, u_k) + f_{o_{dist}} \tag{13}$$

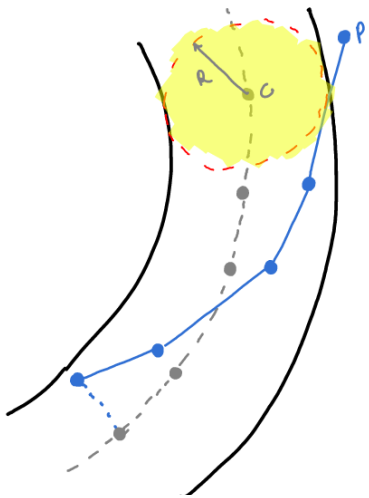
• Constraints

Soft-Constraints

We try to ensure that the vehicle is always inside the track by modelling the soft-constraints

$$|p - c(\theta)| \leq R(\theta) \tag{14}$$

as **relaxed-barrier functions** into the cost function.



Hard-Constraints

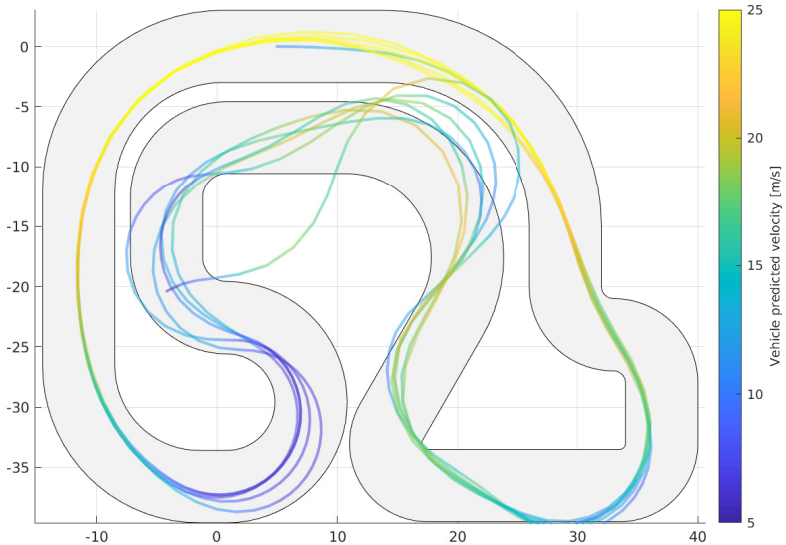
Input are restricted by hard-constraints due to actuation limits

$$\begin{aligned} -20^\circ < &\delta < 20^\circ, \\ -1 < &T < 1 \\ 5 < &v_{track} < 30 \end{aligned}$$

and modelled as **barrier-functions** in the cost function

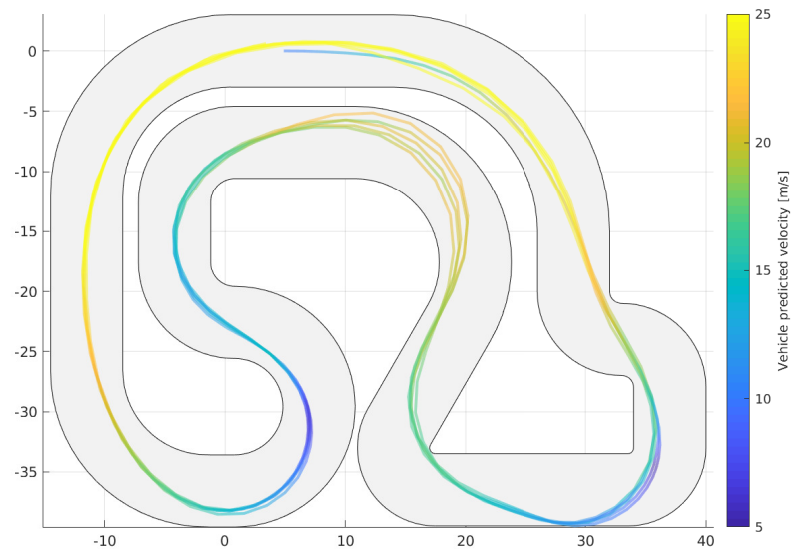
• Results

Traditional NMPC



Lap	Lap time [s]
1	15.30
2	15.75
3	14.25
4	15.15
avg.	**15.11**

Learning-based NMPC



Lap	Lap time [s]
1	12.45
2	11.85
3	12.75
4	12.15
avg.	**12.3**

$\Rightarrow$ Using learning-based NMPC decreases the mean lap time by 18.6%

Outlook and Conclusion

- GP might introduce high nonlinearities in the prediction, making more difficult for the optimizer to a find a good local optimum.
- Replacing the (soft) inequality constraints by (relaxed) barrier functions increase significantly the computational performance (while always ensuring feasibility).
- Hyper-parameter optimization of GP plays an important role in the final controller performance
- No stability guarantee known

Bibliography I


Hohenheim tyre model.
https://reifenmodell.uni-hohenheim.de/en/model.
Accessed: 2020-02-03.


Race car suspension.
https://de.motorsport.com/f1/photos/mercedes-benz-f1-w08-vorderrad-aufhangung-13550764/35144245/.
Accessed: 2020-02-03.


Race car suspension.
https://www.autocar.co.uk/car-news/mercedes-amg-hypercar.
Accessed: 2020-02-03.


Arnold, M., Negenborn, R. R., Andersson, G., and De Schutter, B.
Multi-area predictive control for combined electricity and natural gas systems.
In *2009 European Control Conference (ECC)* (2009), IEEE, pp. 1408–1413.


Kabzan, J., Hewing, L., Liniger, A., and Zeilinger, M. N.
Learning-based model predictive control for autonomous racing.
IEEE Robotics and Automation Letters 4, 4 (2019), 3363–3370.


Nguyen-Tuong, D., Peters, J. R., and Seeger, M.
Local gaussian process regression for real time online model learning.
In *Advances in Neural Information Processing Systems* (2009), pp. 1193–1200.

Bibliography II

Solmaz, S., et al.
Construction of a rational tire model for high fidelity vehicle dynamics simulation under extreme driving and environmental conditions.
In *ASME 2010 10th Biennial Conference on Engineering Systems Design and Analysis* (2010), American Society of Mechanical Engineers Digital Collection, pp. 131–137.

Van Niekerk, B., Damianou, A., and Rosman, B. S.
Online constrained model-based reinforcement learning.