# Performance-oriented model learning for data-driven MPC design

Dario Piga[1], Marco Forgione[1], Simone Formentin[2], and Alberto Bemporad[3]

[1]IDSIA Dalle Molle Institute for Artificial Intelligence SUPSI-USI, Lugano, Switzerland

[2]Dipartimento di Elettronica e Informazione, Politecnico di Milano, Italy

[3]IMT School for Advanced Studies Lucca, Lucca, Italy

Journées Nationales Automatique
Webinaire 25-27 Novembre 2020

# Dalle Molle Institute for Artificial Intelligence

A research institute focusing on AI (and more) in Lugano, Switzerland.

Lugano

Cynar liquor


Angelo Dalle Molle

Founded in 1988 by Italian entrepreneur & filantropist Angelo Dalle Molle.

Affiliated with

- USI - University of Lugano
- SUPSI - University of Applied Science (Fachhochschule)

# Dalle Molle Institute for Artificial Intelligence

A research institute focusing on AI (and more) in Lugano, Switzerland.

Lugano

Cynar liquor


Angelo Dalle Molle

Founded in 1988 by Italian entrepreneur & filantropist Angelo Dalle Molle.

Affiliated with

- USI - University of Lugano
- SUPSI - University of Applied Science (Fachhochschule)

# Dalle Molle Institute for Artificial Intelligence

Areas of expertise of the institute

- Machine Learning
- Artificial Vision
- Optimization
- Statistics
- Computational biophysics
- Systems & Control

Innosuisse
Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

European Commission

MANDATES

APPLIED RESEARCH

BASIC RESEARCH

erc
European Research Council
Established by the European Commission

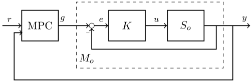
FET-OPEN

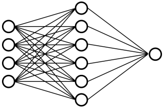
SILICON VALLEY community foundation

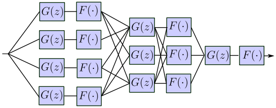
# Learning for Decision & Control

Our research focuses on:

- System Identification
- Model Predictive Control
- Distributed Optimization
- Robotics






At the interface of machine learning/control. Example (my research):

1. Deep learning tools for dynamical system models
2. Bayesian optimization for model learning and controller tuning

# Bayesian Optimization for Machine Calibration

Calibration problems are common in engineering practice:

- Several tuning knobs are available
- Optimal values to be determined
- Direct experimentation is possible

Typical approaches:

- Expert trial-and-error
- Model-free Design of Experiment:
  - Full factorial design
  - Fractional factorial design
  - ...

Drilling Machine


(Georg Fisher)

- Bayesian Optimization formalizes trial-and-error optimization!

# Bayesian Optimization for Machine Calibration

Calibration problems are common in engineering practice:

- Several tuning knobs are available
- Optimal values to be determined
- Direct experimentation is possible

Typical approaches:

- Expert trial-and-error
- Model-free Design of Experiment:
  - Full factorial design
  - Fractional factorial design
  - ...

Drilling Machine


(Georg Fisher)

- Bayesian Optimization formalizes trial-and-error optimization!

# Bayesian Optimization for Machine Calibration

The calibrator aims at minimizing a given performance index $J$ w.r.t. tuning parameters $\theta$:

$$\theta^{\mathrm{opt}} = \arg\min_{\theta \in \Theta} J(\theta)$$

The game has the following rules:

- An analytical expression of $J$ as a function of $\theta$ is not available
- One can run experiments of $J$ for different values of $\theta$ and measure the corresponding noisy observation $J$
- Each experiment can be costly and time-consuming

Goal: approach the global optimum of $J$ in a limited number of experiments

# Global Optimization Algorithms

## Overview

Different derivative-free global optimization may be used:

- Response surface methods
- Genetic algorithms
- Particle Swarm Optimization
- ...

Bayesian Optimization is efficient in terms of function evaluations

- Like response surface methods, BO fits a surrogate $\hat{J}(\theta)$ of the unknown objective $J(\theta)$
- However, the surrogate model is stochastic (describes uncertainty)
- Explicitly balances exploration and exploitation

# Global Optimization Algorithms

## Overview

Different derivative-free global optimization may be used:

- Response surface methods
- Genetic algorithms
- Particle Swarm Optimization
- ...

Bayesian Optimization is efficient in terms of function evaluations

- Like response surface methods, BO fits a surrogate $\hat{J}(\theta)$ of the unknown objective $J(\theta)$
- However, the surrogate model is stochastic (describes uncertainty)
- Explicitly balances exploration and exploitation

# Bayesian Optimization

- Iteratively updates a stochastic surrogate model $\hat{J}(\theta)$ of the unknown $J(\theta)$ via Bayesian inference. Typically, a Gaussian Process (GP)
- Balances exploitation and exploration by optimizing an acquisition function $A(\theta)$ instead of the surrogate model directly:

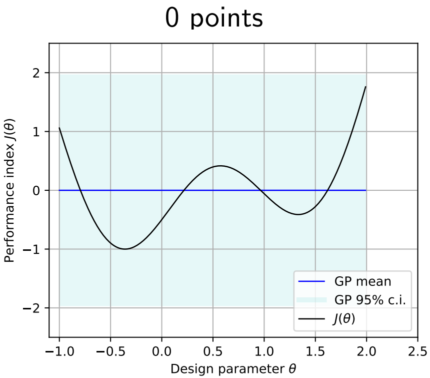
$$\theta_{i+1} = \arg\max_{\theta \in \Theta} A(\theta)$$

- The acquisition function $A(\theta)$ gives high value to points with expected good performance $\rightarrow$ exploitation and/or high variance $\rightarrow$ exploration

# Bayesian Optimization

## Gaussian Process

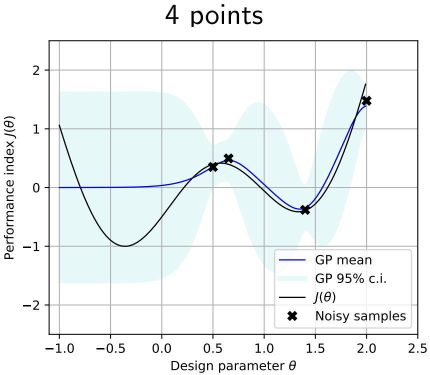
- The function $J(\theta)$ is assumed Gaussian with prior mean $E[J(\theta)] = \mu(\theta)$ and covariance $\text{cov}[J(\theta_1), J(\theta_2)] = \kappa(\theta_1, \theta_2)$.
- The posterior mean and covariance given a new observation $(\theta_i, J_i)$ is obtained in closed form



# Bayesian Optimization

## Gaussian Process

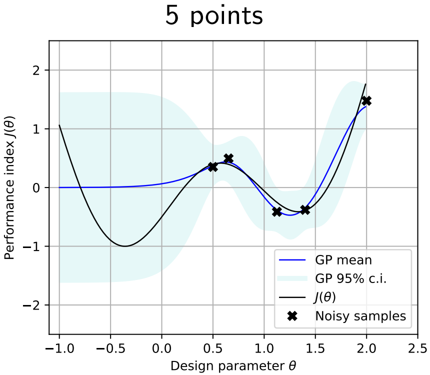
- The function $J(\theta)$ is assumed Gaussian with prior mean $E[J(\theta)] = \mu(\theta)$ and covariance $\text{cov}[J(\theta_1), J(\theta_2)] = \kappa(\theta_1, \theta_2)$.
- The posterior mean and covariance given a new observation $(\theta_i, J_i)$ is obtained in closed form



# Bayesian Optimization

## Gaussian Process

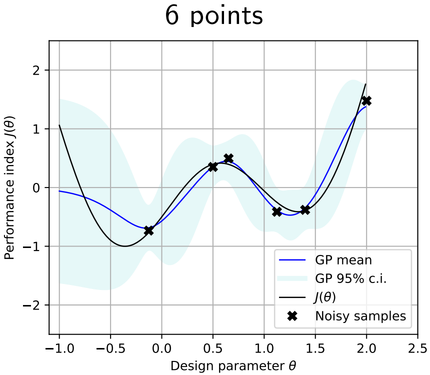
- The function $J(\theta)$ is assumed Gaussian with prior mean $E[J(\theta)] = \mu(\theta)$ and covariance $\text{cov}[J(\theta_1), J(\theta_2)] = \kappa(\theta_1, \theta_2)$.
- The posterior mean and covariance given a new observation $(\theta_i, J_i)$ is obtained in closed form



# Bayesian Optimization

## Gaussian Process

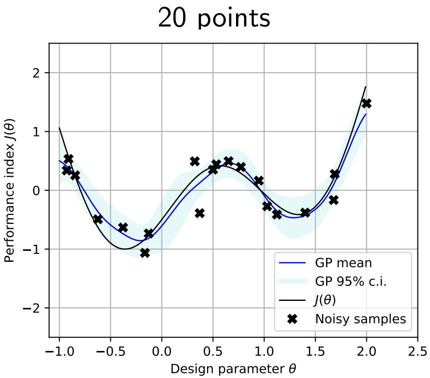
- The function $J(\theta)$ is assumed Gaussian with prior mean $E[J(\theta)] = \mu(\theta)$ and covariance $\text{cov}[J(\theta_1), J(\theta_2)] = \kappa(\theta_1, \theta_2)$.
- The posterior mean and covariance given a new observation $(\theta_i, J_i)$ is obtained in closed form



# Bayesian Optimization

## Gaussian Process

- The function $J(\theta)$ is assumed Gaussian with prior mean $E[J(\theta)] = \mu(\theta)$ and covariance $\text{cov}[J(\theta_1), J(\theta_2)] = \kappa(\theta_1, \theta_2)$.
- The posterior mean and covariance given a new observation $(\theta_i, J_i)$ is obtained in closed form



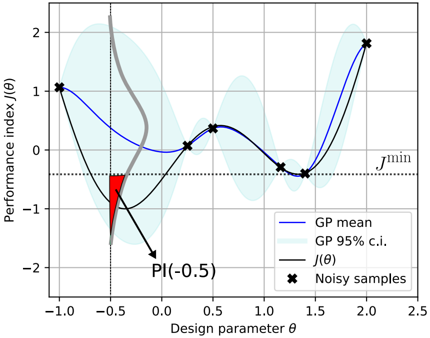
# Bayesian Optimization

## Acquisition function

The GP defines the probability distribution of $J(\theta)$ for each parameter $\theta$.
This probability is used to define an acquisition function $A(\theta)$.

The acquisition function *Probability of Improvement* is defined as:

$$A(\theta) = \mathrm{PI}(\theta) = \mathrm{Prob}(J(\theta) \leq J^{\mathrm{min}})$$



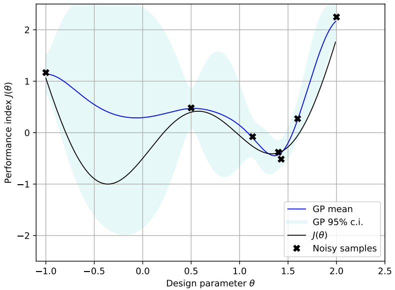
# Bayesian Optimization

## Algorithm

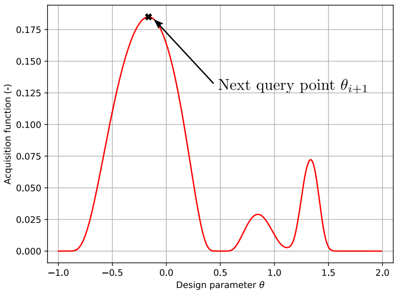
Steps of BO: for $i = 1, 2, \ldots n$

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

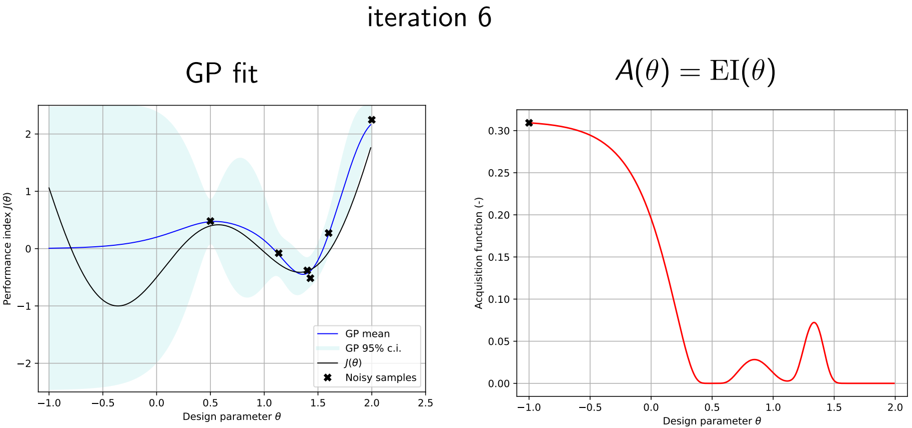
GP at iteration $i$




$A(\theta)$ at iteration $i$



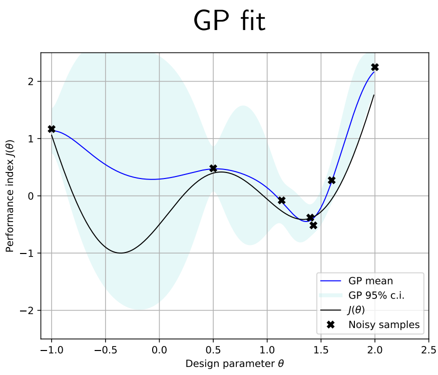
# Bayesian Optimization
## Example

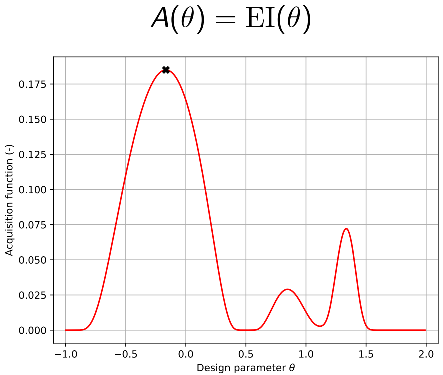


# Bayesian Optimization

## Example

iteration 7

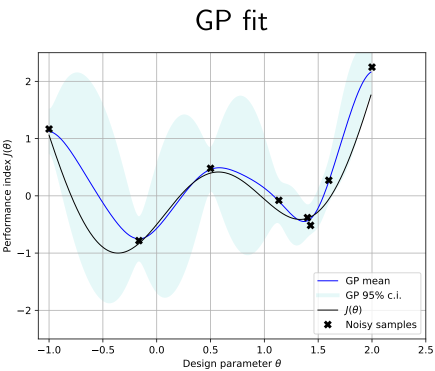


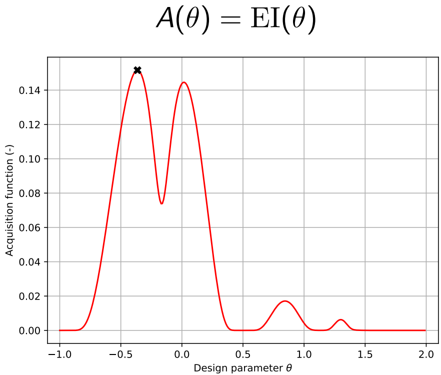



# Bayesian Optimization

## Example

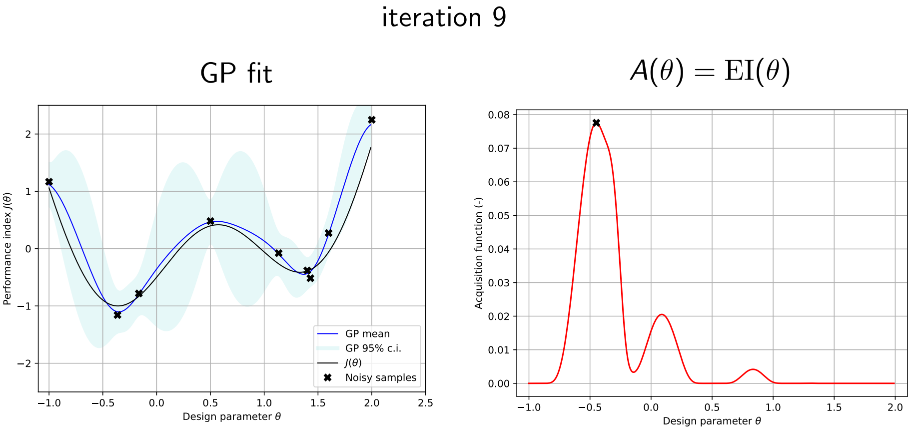
iteration 8



# Bayesian Optimization

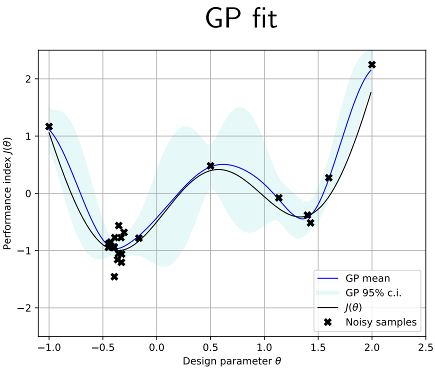
## Example

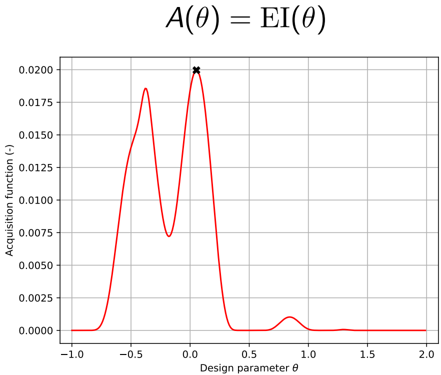


# Bayesian Optimization

## Example

iteration 20



# Bayesian Optimization for Systems & Control

We can't tell more of our industrial collaborations!

Some other (disclosable) applications

- Controller tuning for a 7-DoF robotic manipulator
- Tuning of sampling time, solver tolerance, etc. for embedded MPC
- Choice of the model for MPC

. . . and related publications


L. Roveda, M. Forgione, D. Piga. Robot control parameters auto-tuning in trajectory tracking applications.
*Control Engineering Practice, 101(2020), pp 72-78, 2020*

M. Forgione, D. Piga, A. Bemporad. Efficient Calibration of Embedded MPC.
*In Proc. of the 21st IFAC World Congress, 2020.*

D. Piga, M. Forgione, S. Formentin, A. Bemporad. Performance-oriented model learning for data-driven MPC design.
*IEEE Control Systems Letters, 3(3), pp 577-582, 2019.*

# MPC Model Calibration

## Motivation

Obtaining the predictive model for MPC is costly and time-consuming.

Typically, models are obtained through Physical modeling or Identification

- A trade-off emerges between accuracy and complexity

In this work:

- We consider the model as a design parameter and tune it on calibration experiments to optimize a user-defined performance index
- We specialize this framework for a hierarchical MPC architecture often encountered in industrial applications
- Can be seen as an extension of Identification for Control

# MPC Model Calibration

## Motivation

Obtaining the predictive model for MPC is costly and time-consuming.

Typically, models are obtained through Physical modeling or Identification

- A trade-off emerges between accuracy and complexity

In this work:

- We consider the model as a design parameter and tune it on calibration experiments to optimize a user-defined performance index
- We specialize this framework for a hierarchical MPC architecture often encountered in industrial applications
- Can be seen as an extension of Identification for Control

# MPC Model Calibration

## Motivation

Obtaining the predictive model for MPC is costly and time-consuming.

Typically, models are obtained through Physical modeling or Identification

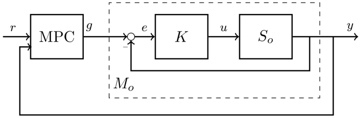
- A trade-off emerges between accuracy and complexity

In this work:

- We consider the model as a design parameter and tune it on calibration experiments to optimize a user-defined performance index
- We specialize this framework for a hierarchical MPC architecture often encountered in industrial applications
- Can be seen as an extension of Identification for Control

# Control architecture

We consider the Reference Governor architecture for system $S_o$




1. An inner controller $K$ handles fast dynamics
2. An outer MPC takes care of constraints and performance specs

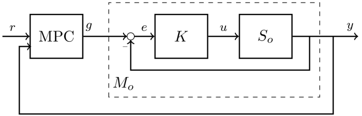
MPC requires a model $M$ of the inner loop $M_o$. Existing approaches:

- Build model $S$ for $S_o$, design $K \Rightarrow M = \texttt{feedback}(SK, I)$
- Direct identification of $K$ targeting a reference model $M$ (VRFT)

In our work, $M$ and $K$ are tuned simultaneously with a data-driven global optimization approach.

# Control architecture

We consider the Reference Governor architecture for system $S_o$




1. An inner controller $K$ handles fast dynamics
2. An outer MPC takes care of constraints and performance specs

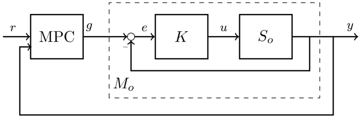
MPC requires a model $M$ of the inner loop $M_o$. Existing approaches:

- Build model $S$ for $S_o$, design $K \Rightarrow M = \texttt{feedback}(SK, I)$
- Direct identification of $K$ targeting a reference model $M$ (VRFT)

In our work, $M$ and $K$ are tuned simultaneously with a data-driven global optimization approach.

# Control architecture

We consider the Reference Governor architecture for system $S_o$




1. An inner controller $K$ handles fast dynamics
2. An outer MPC takes care of constraints and performance specs

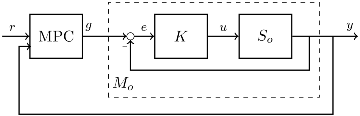
MPC requires a model $M$ of the inner loop $M_o$. Existing approaches:

- Build model $S$ for $S_o$, design $K \Rightarrow M = \texttt{feedback}(SK, I)$
- Direct identification of $K$ targeting a reference model $M$ (VRFT)

In our work, $M$ and $K$ are tuned simultaneously with a data-driven global optimization approach.

# Control architecture

## Inner Loop Controller




The inner controller $K$ generates the system input $u$.
It is designed to handle fast dynamics

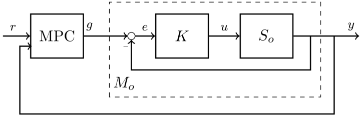
- Stabilize inner loop $M$
- Reject fast system disturbances

It is often as simple as a PID. . .

$$K(z,\theta) = \theta_P + \theta_I T_s \frac{1}{z-1} + \theta_D \frac{N_d}{1 + N_d T_s \frac{1}{z-1}}$$

# Control architecture

## Model Predictive Controller




The outer MPC generates the reference $g$ for the inner loop $M_o$ using a model $M(\theta) : g \to \left[\begin{smallmatrix} y \\ u \end{smallmatrix}\right]$

$$\xi_{t+1} = A_M \xi_t + B_M g_t$$
$$\begin{bmatrix} y_t \\ u_t \end{bmatrix} = C_M \xi_t + D_M g_t,$$

to handle constraints and enhance performance, according to

$$\min_{\{g_{t+k|t}\}_{k=1}^{N_p}} \sum_{k=1}^{N_p} \left\| y_{t+k|t} - r_{t+k} \right\|_{Q_y}^2 + \left\| u_{t+k|k} - u_{t+k-1|t} \right\|_{Q_{\Delta u}}^2$$

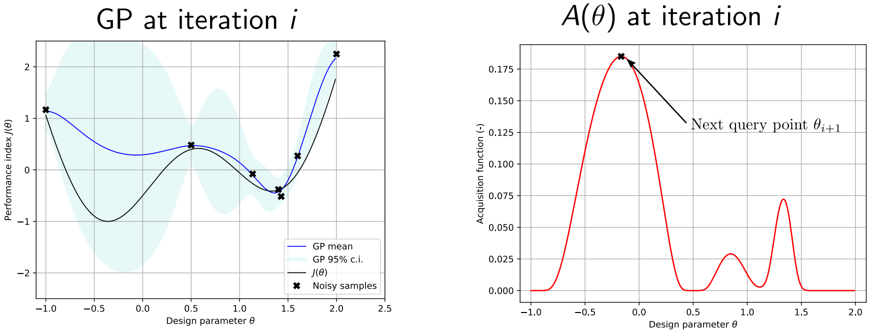
s.t. model equations, constraints on $g$, $y$, $u$, $\Delta u$

# Bayesian Optimization

## Algorithm

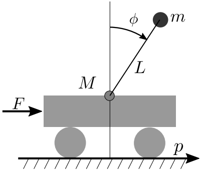
Steps of BO: for $i = 1, 2, \ldots n$

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



# Simulation Example

Cart-pole system



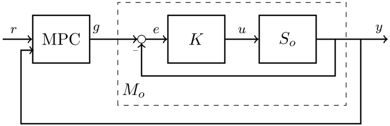

- State $x = [p \; \dot{p} \; \phi \; \dot{\phi}]^\top$
- Output $y = [p \; \phi]^\top$ corrupted by white measurement noise
- Input $u = F$ with fast additive disturbance (10 rad/sec)
- Control structure: inner PID on $\phi$, outer MPC as Reference Governor

Objective: starting at $p_0 = 0$, $\phi_0 = 15^o$

1. stabilize pendulum in the upright unstable equilibrium $\phi = 0$
2. keep cart position $p$ in $[-1 \;\; 1]$ m

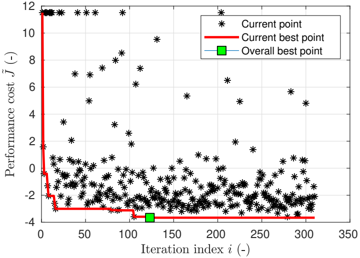
$$J(\theta) = \log\left[\frac{1}{T}\sum_{t=1}^{T}\left(\frac{1}{10}|p_t| + \frac{9}{10}|\phi_t|\right)\right] + \sum_{t=1}^{T}\ell(|p_t| - 1)$$

- Design parameters: PID gains, model $M$, prediction horizon $N_p$
- Calibration experiments of 10 s
- $T_s^{\mathrm{PID}} = 5$ ms, $T_s^{\mathrm{MPC}} = 50$ ms

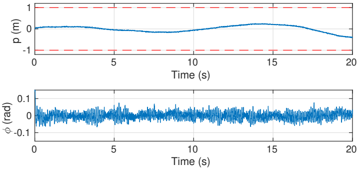


# Simulation Example

Performance index vs. iteration



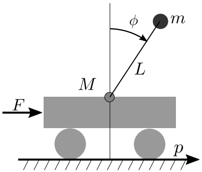

Optimal trajectory




- For increasing iteration $i$, more and more points have "low" cost
- Optimal trajectory satisfies constraints $p \in [-1 \;\; 1]$ m
- Achieved performance is better than our manual tuning

# MPC Calibration Example

Cart-pole system




- Input: force $F$ with fast disturbance (5 rad/sec)
- Output: noisy position $p$ and angle $\phi$
- Objective: follow trajectory for $p$, keep $\phi$ small.

Optimization-based calibration of

1. MPC sample time $T_s^{\mathrm{MPC}}$
2. MPC weights $Q_y$ and $Q_{\Delta u}$
3. ...
4. QP solver tolerances

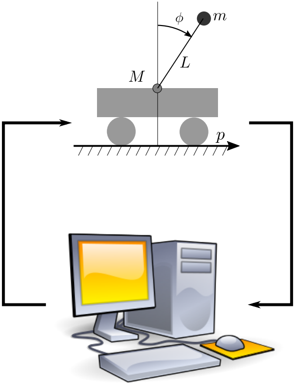
$$J(\theta) = \int_0^{T_{\mathrm{exp}}} |\overline{p}(t) - p(t)| + 30|\phi(t)|\, dt$$

subject to:

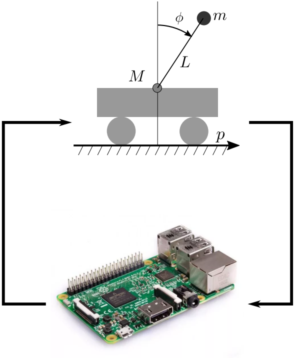
$$T_{\mathrm{calc}}^{\mathrm{MPC}}(\theta) < T_s^{\mathrm{MPC}}$$

to guarantee real-time implementability.

# MPC Calibration Example



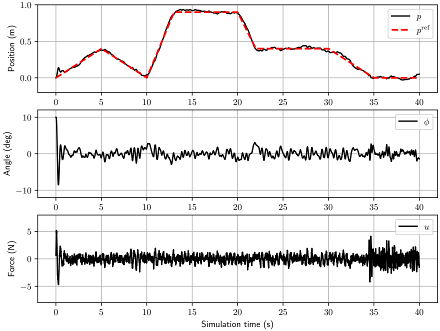

vs.




- An Intel i5 PC (left) vs. an ARM-based Raspberry PI 3 (right)
- PI is about 10 time slower than the PC for MPC computations

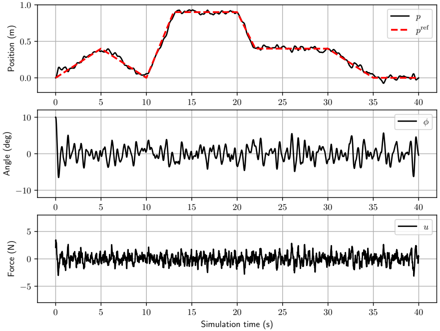
# MPC Calibration Example

Optimized MPC on the PC




$T_s^{\mathrm{MPC}} = 6$ ms

Optimized MPC on the Raspberry PI




$T_s^{\mathrm{MPC}} = 22$ ms

- Position and angle control tighter on the PC
- Faster loop update on the PC $\Rightarrow$ more effective disturbance rejection
- Calibration squeezes max performance out of the hardware!

# Conclusions

Bayesian Optimization is a powerful global optimization tool. Applications:

- Machine calibration
- Robot control tuning
- Embedded MPC design
- MPC model learning

Questions:

- How does a controller generalizes to unseen trajectories?
- How to ensure safety during experimentations?

Extension: preference-based learning. Performance index not available, human expert gives his/her preferences to binary comparisions.

# Conclusions

Bayesian Optimization is a powerful global optimization tool. Applications:

- Machine calibration
- Robot control tuning
- Embedded MPC design
- MPC model learning

Questions:

- How does a controller generalizes to unseen trajectories?
- How to ensure safety during experimentations?

Extension: preference-based learning. Performance index not available, human expert gives his/her preferences to binary comparisions.

# Conclusions

Bayesian Optimization is a powerful global optimization tool. Applications:

- Machine calibration
- Robot control tuning
- Embedded MPC design
- MPC model learning

Questions:

- How does a controller generalizes to unseen trajectories?
- How to ensure safety during experimentations?

Extension: preference-based learning. Performance index not available, human expert gives his/her preferences to binary comparisions.

# Thank you.
# Questions?

marco.forgione@idsia.ch