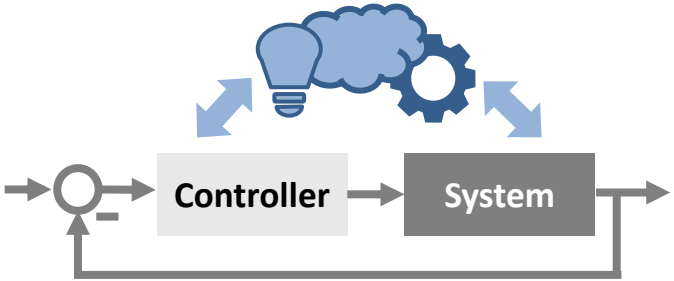
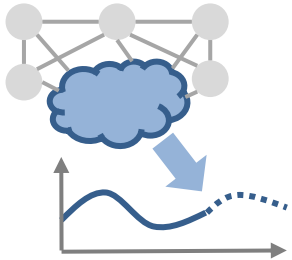
TECHNISCHE UNIVERSITÄT DARMSTADT

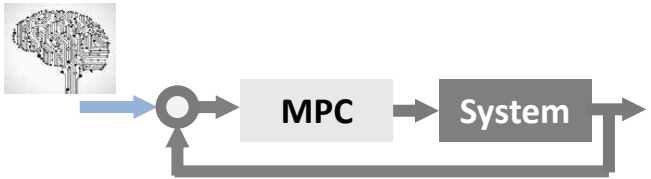
Integrating Physics in Learning for Model-Based Control

Rolf Findeisen, Maik Pfefferkorn, Sebastian Hirt, Hoang Hai Nguyen, Andreas Höhl

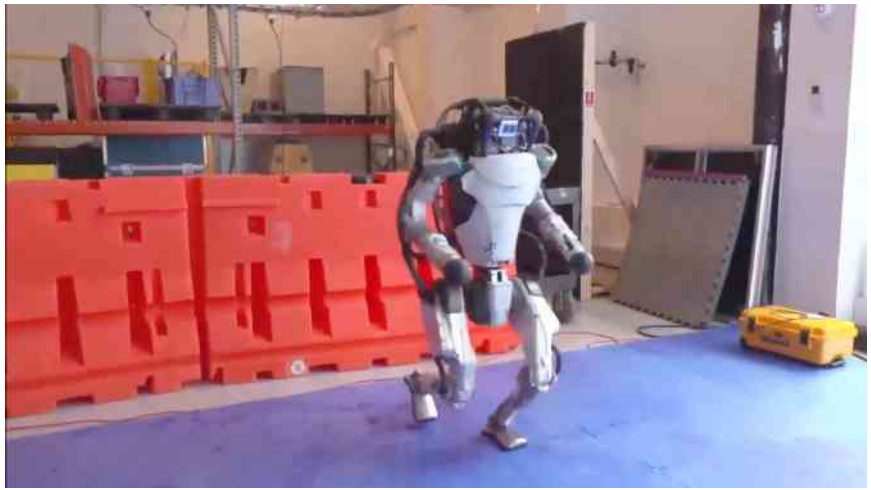
Control and Cyber-Physical Systems Laboratory
Technical University of Darmstadt, Germany







Motivation



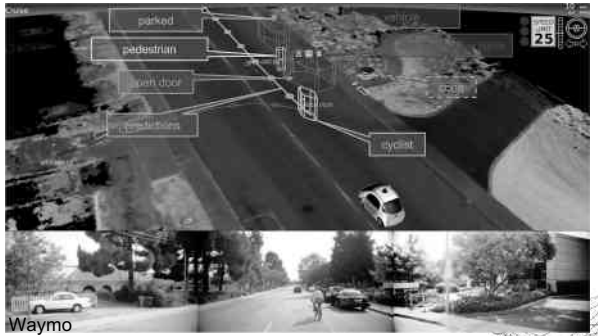
Autonomous systems should be cable of

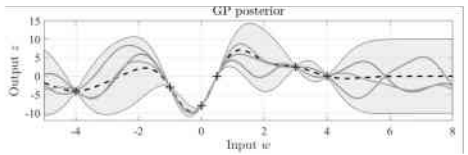
- operating safely and reliably
- reacting to disturbances & adapt to changes in the environment, operation mode or task objectives

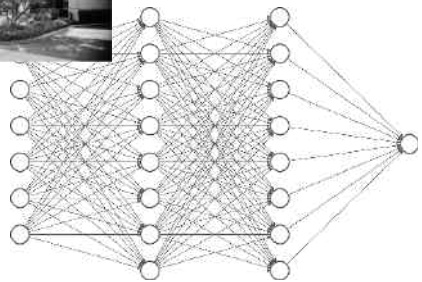
Reality: autonomous systems often fail to adapt ► loss of safety and reliability

Fuse control and machine learning to restore adaptability and achieve autonomy

Motivation







Machine Learning

- Significant advancements in recent years
- Resembles human learning by improving from experience / acquired data
- Naturally adapts to changing conditions

Mainly driven by

- Availability of data for training due to digitalization
- (Cloud) computing resources for learning
- Open-source software (PyTorch, …)

Fuse control and machine learning to restore adaptability and achieve autonomy

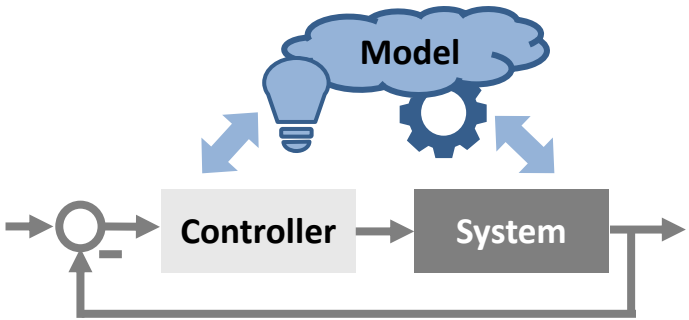
- Safety and reliability ►constraint satisfaction (quality, ….)?
- **Exploit prior knowledge (models, physical insight, …)?**
  - ► **purely data-based models / controllers do not necessarily reflect physical properties**
  - ► **decreased control performance**
  - ► **reduced explainability**

Model-based Control

- Many control methods rely on **models of the system, objectives, constraints, and disturbances**
- Models are traditionally obtained from first principles (physical laws), or via identification

Common model-based approaches

- Inversion-based control/feedback linearization
- Optimization based control (LQR, Model Predictive Control, …)
- Model-based Reinforcement learning
- …



Advantages:

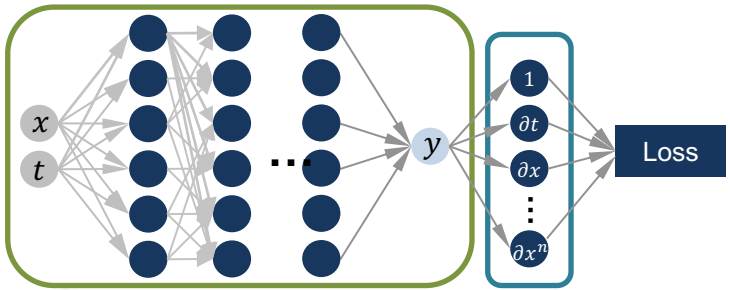
- Rich literature, sound theory, guarantees by design and validation possible
- Leverages physical insights, interpretability
- Widely used and accepted

Challenges:

- Time consuming modelling, identification
- Difficult to obtain sufficiently good model if the system is complex, changes, …

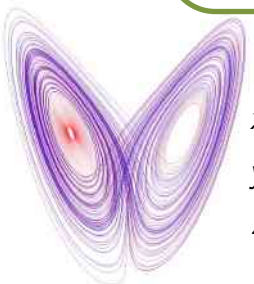
Challenge: Machine learning does not guarantee that the model takes physical insights into account

Motivation

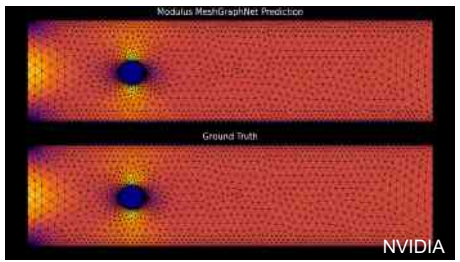


Physics-informed Machine Learning

- Physical dynamics models are readily available
  ► exploit prior (expert) knowledge
- Accelerate simulations/predictions especially for **high-dimensional** and **complex dynamics**



$$\dot{x} = \sigma(x - y)$$
$$\dot{y} = x(\rho - z)$$
$$\dot{z} = xy - \beta z$$



Approaches

- Provide physical insights to ML models via
  - Cost function during training
  - Structurally via model architecture

Physics-informed ML models provide

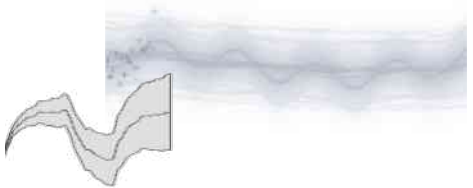
- Improved interpretability and explainability
- Enhanced extrapolation, robustness, and generalization (e.g., preventing overfitting)
- Better sample-efficiency due to reduced search space

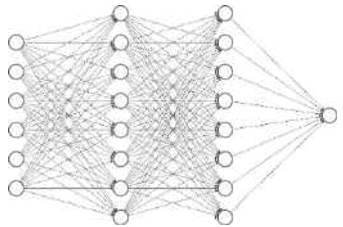
► **Exploit the strengths of existing physics knowledge and machine learning for model-based control**

Physics-Informed Learning for Model-based Control

Motivation

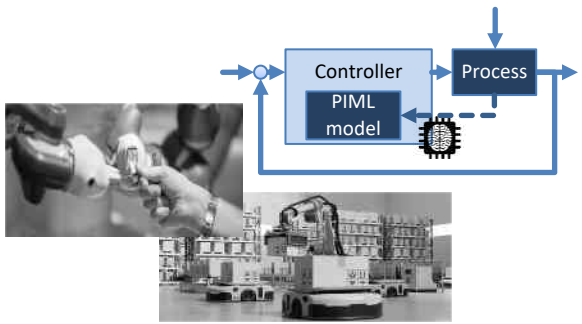
- Fusing control and machine learning to achieve autonomy of systems
- Physics-informed ML: integrate known parts, infer unknown parts from data





Physics-informed machine learning

- Integrating physics in Gaussian process models
- Integrating physics in neural network models



Physics-informed learning for model-based control

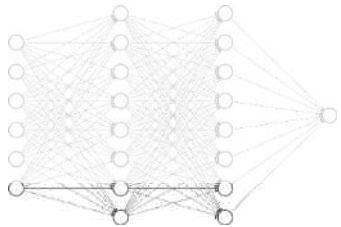
- Physics-informed model predictive control

Physics-informed learning: build reliable model integrating known physics and data

Physics-Informed Learning for Model-based Control

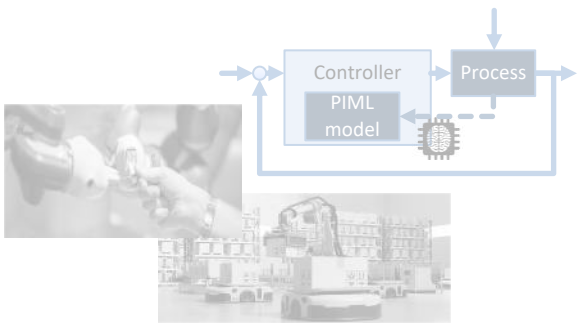
Motivation

- Fusing control and machine learning to achieve autonomy of systems
- Physics-informed ML: integrate known parts, infer unknown parts from data



Physics-informed machine learning

- Integrating physics in Gaussian process models
- Integrating physics in neural network models

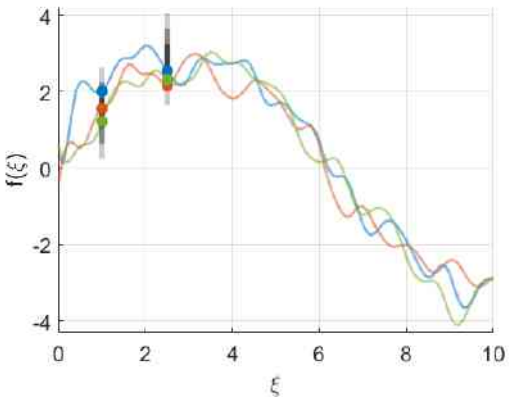


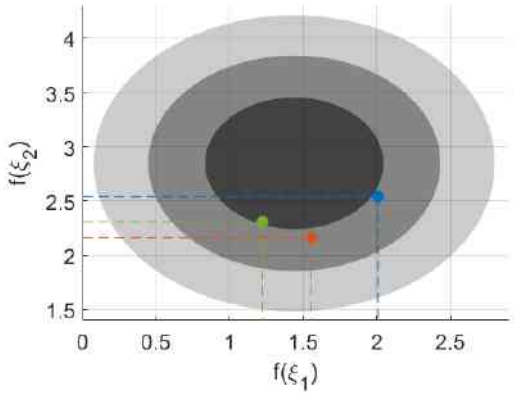
Physics-informed learning for model-based control

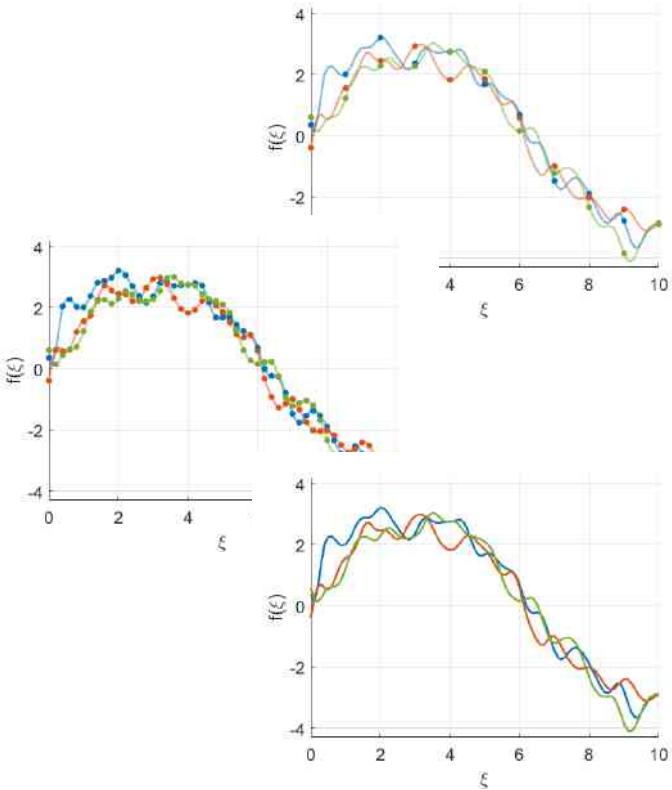
- Physics-informed model predictive control

An Intuition about Gaussian Processes

Core idea: Modeling of realizations of uncertain functions as samples from normal distributions







Definition (Gaussian process)

Be T an index set and be Ω a sample space. A Gaussian process is a collection

$$X = \{ X_t(\omega) \mid t \in T, \omega \in \Omega \}$$

of random variables $X_t(\omega)$ any finite subset of which has a joint and consistent Gaussian distribution.

Gaussian Process Priors

A Gaussian process $\mathrm{f}(\xi) \sim \mathcal{GP}\big(m(\xi), k(\xi, \xi')\big)$ is a Gaussian probability distribution over functions

Objective:

- Exploit knowledge provided by **training data** $\big\{(\xi_i, \gamma_i)_{i=1}^n | \gamma_i = f(\xi_i) + \varepsilon, \varepsilon \sim \mathcal{N}(0, \sigma_n^2)\big\}$
- ▶ Infer unobserved function values (**test targets**) $f_* = f(\xi_*)$ at **test inputs** ξ_*

Basic assumption (joint prior distribution)

Any finite number of function values have a joint Gaussian probability distribution, especially

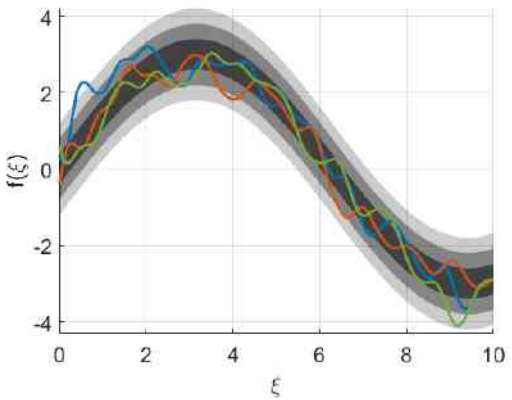
$$\begin{bmatrix} g(\Xi) \\ g(\Xi_*) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m(\Xi) \\ m(\Xi_*) \end{bmatrix}, \begin{bmatrix} k(\Xi, \Xi) + \sigma_n^2 \mathbb{I} & k(\Xi, \Xi_*) \\ k(\Xi_*, \Xi) & k(\Xi_*, \Xi_*) \end{bmatrix}\right)$$

where $\Xi = [\xi^{(1)}, \ldots, \xi^{(n)}]^\mathsf{T} \in \mathbb{R}^{n \times d}, \Xi_* = [\xi_*^{(1)}, \ldots, \xi_*^{(n_*)}] \in \mathbb{R}^{n_* \times d}$

Prior GP model of the unknown function

$$g(\xi_*) \sim \mathcal{GP}(m(\xi_*), k(\xi_*, \xi_*'))$$

- ▶ Encodes beliefs about the unknown function



Prior Model - Mean Functions

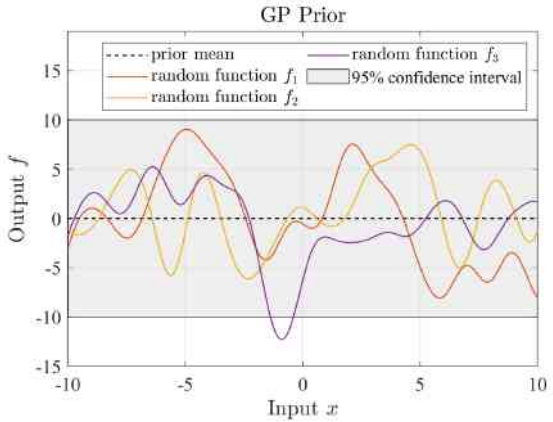
Prior mean function $m : \mathbb{R}^d \to \mathbb{R}, \xi \mapsto \mathrm{E}[f(\xi)]$

$$g(\xi) \sim \mathcal{GP}\left(m(\xi), k(\xi, \xi')\right)$$

Used to encode prior knowledge about the underlying function

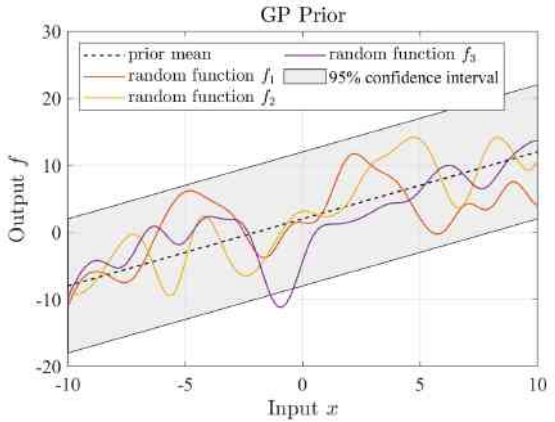
Constant mean function:

$$m(\xi) = c, c \in \mathbb{R}$$



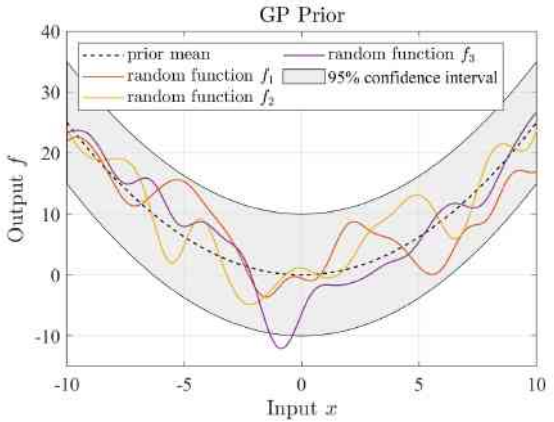
Affine mean function:

$$m(\xi) = a\xi + b, a, b \in \mathbb{R}$$



Quadratic mean function:

$$m(\xi) = a\xi^2, a \in \mathbb{R}$$



The **prior mean functions specifies an overall trend** but not the particular shape of the functions

ontrol

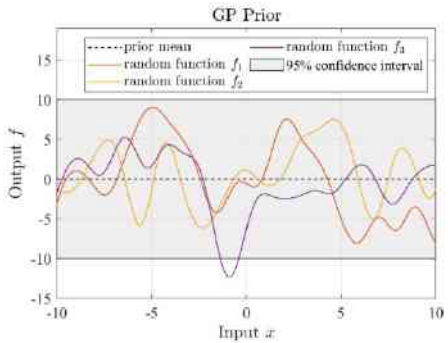
Prior Model: Covariance Functions

Prior covariance (kernel) function $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}, (\xi, \xi') \mapsto \mathrm{Cov}[f(\xi), f(\xi')]$

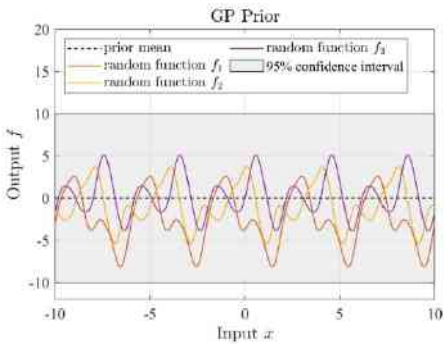
$$f(\xi) \sim \mathcal{GP}\left(m(\xi), k(\xi, \xi')\right)$$

Used to encode prior knowledge about the underlying function

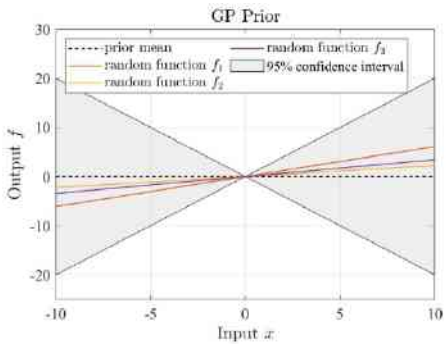
Squared exponential kernel:
- Cont. differentiable fcn's



Periodic kernel
- Periodic functions



Linear kernel:
- Linear functions



The **prior covariance function allows to induce properties such as smoothness, periodicity**, … of the sample functions and defines the function class to which they belong

Gaussian Process Posteriors

A Gaussian process $\mathrm{f}(\xi) \sim \mathcal{GP}\big(m(\xi), k(\xi, \xi')\big)$ is a Gaussian probability distribution over functions

Incorporation of training data:

- **Conditioning of the prior on the training data**
  - ▶ posterior model / predictive distribution

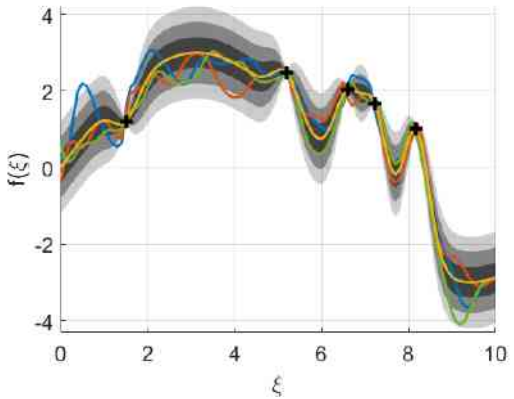
$$g(\Xi_*) \mid \Xi_*, \Xi, \gamma \sim \mathcal{N}\left(m^+(\Xi_*), k^+(\Xi_*, \Xi_*)\right)$$

Posterior GP model of the unknown function

$$g(\xi_*) \sim \mathcal{GP}\big(m^+(\xi_*), k^+(\xi_*, \xi_*')\big)$$

- ▶ **Rejection of prior functions that are inconsistent with the training observations**

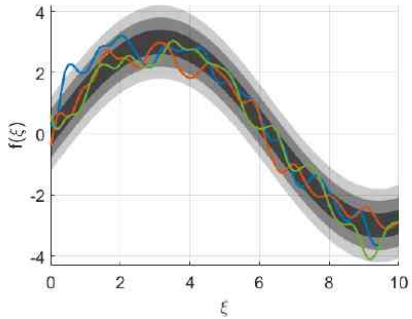
- $m^+(\xi_*) = m(\xi_*) + k(\xi_*, \Xi)\Sigma(\gamma - m(\Xi))$
  - ▶ Estimate for the unknown function values
- $k^+(\xi_*, \xi_*') = k(\xi_*, \xi_*') - k(\xi_*, \Xi)\Sigma k(\Xi, \xi_*') + \sigma_n^2$
  - ▶ $k^+(\xi_*, \xi_*)$ quantifies prediction uncertainty
- $\Sigma = \left[k(\Xi, \Xi) + \sigma_n^2 \mathbb{I}\right]^{-1}$

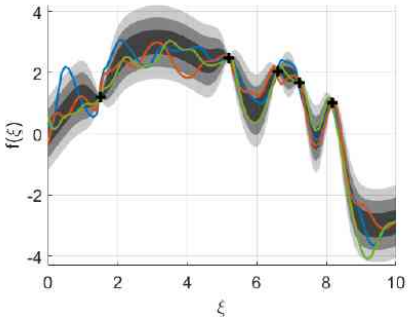


Gaussian processes model unknown functions probabilistically based on previous observations

Hyperparameter Learning

- Prior mean $m(\cdot;\theta)$ and covariance $k(\cdot,\cdot;\theta)$ define GP model
- Dependent on **hyperparameters** θ





► **Meaningful prediction model requires suitable hyperparameters**

Learning optimal hyperparameters by maximizing data evidence (logarithmic marginal likelihood):

$$\theta^* = \arg\max_{\theta} \left\{ \underbrace{\frac{1}{2}(\gamma - m(\Xi))^T [k(\Xi,\Xi) + \sigma_n^2 \mathbb{I}]^{-1} (\gamma - m(\Xi)) + \frac{1}{2} \log\left(\det(k(\Xi,\Xi) + \sigma_n^2 \mathbb{I})\right) + \frac{n}{2} \log(2\pi)}_{\log(p(\gamma|\Xi,\theta))} \right\}$$

- Corresponds to maximizing the GP's capability of explaining the training data
- Global optimum not guaranteed ► local optima correspond to different interpretations of the data

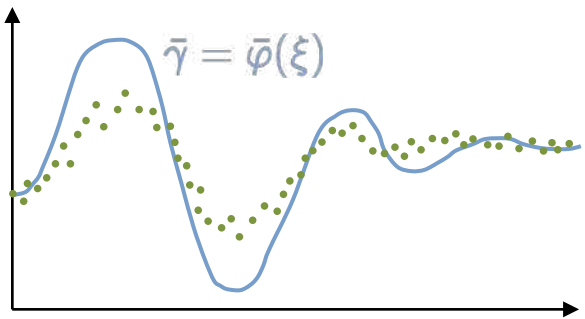
- **GP learning is an optimization problem**
- **Hyperparameter learning adapts the GP model to the underlying problem**

Physics-Informed Gaussian Processes

How to embed physical prior knowledge in Gaussian process models?

Systems are often partially unknown

- Some **physics-based model** $\bar{\gamma} = \bar{\varphi}(\xi)$ of the unknown function $\gamma = \varphi(\xi)$ **is available**
- This **model is uncertain/imperfect** (e.g., unknown parameters, unmodeled parts, …)
- The **missing information should be provided by data**



Relying purely on the physics-based model results in an insufficient prediction quality

Relying on a GP model based on only the data may lead to models that are inconsistent with the known physics ▶ unrealisitic predictions

We want to design a GP model that

- **complies with the known physics**, i.e., incorporates the known model
- **corrects/infers the uncertain parts of the model using the data**
- **provides uncertainty information about the model**

Physics-Informed Gaussian Processes

How to embed physical prior knowledge in Gaussian process models?

Physical knowledge may be available in various forms

- Bound constraints
- Monotonicity constraints
- Differential equation constraints
- Boundary condition constraints
- …

Strategies to bound constraints

- Transforming the GP output
- Employing non-Gaussian likelihoods
- Truncation of the Gaussian output distribution
- **Constrained MLE optimization (training)**

Strategies to monotonicity constraints

- Constrained likelihood including derivative information
- Truncated Gaussian distributions

Strategies to differential equation constraints

- Block covariance kernels to enforce compliance with DEs at collocation points
- Specialized kernel construction (restriction of the GP sample space)
- Empirical mean and covariance

Strategies to boundary condition constraints

- Spectral expansion of kernels

In the following: focus on **differential equation constraints**

Physics-Informed Gaussian Processes

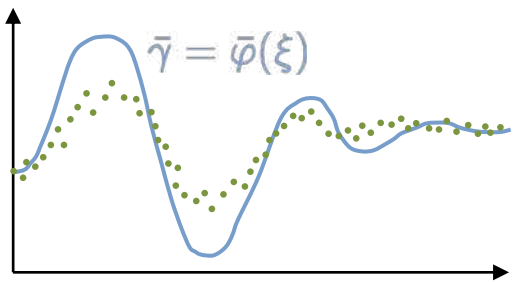
How to embed physical prior knowledge in Gaussian process models?

Reminder: GP posterior mean

$$m^+(\xi_*) = m(\xi_*) + \sum_{i=1}^{n} \alpha_i k(\xi, \xi_*)$$

- linear **combination of prior covariance functions "shifted" by prior mean function**

Use a dedicated prior to incorporate physics



Physics-informed prior mean

- Use $m(\xi_*) = \bar{\varphi}(\xi_*)$
- The GP learns the difference between $\bar{\varphi}(\xi_*)$ and $\varphi(\xi_*)$ from the data
- The functional structure of the posterior mean and the sample functions does not match that of the true function

Physics-informed prior covariance

- If the functional structure of φ is known, construct $k(\cdot,\cdot)$ such that $m^+(\cdot)$ and sample functions are restricted to the same class
- Any sample complies with physics, i.e., has an exact real-world realization

How to construct physics-informed GP priors?

Physics-Informed Gaussian Processes

Purely data-driven priors

[X. Yang, D. Barajas-Solano, G. Tartakovsky, A. M. Tartakovsky, 2018]

Stochastic physics-based models:
include expert knowledge (known physics) and random parameters or random fields (imperfect system knowledge)

- stochastic physics-based model $\bar{\varphi}(\xi;\omega)$
- M realizations $\{\Phi^m(\xi)\}_{m=1}^M$ thereof (**Monte-Carlo simulations/sampling**)

▶ $\Phi(\xi) \sim \mathcal{GP}(m_{\mathrm{MC}}(\xi), k_{\mathrm{MC}}(\xi,\xi'))$

Construction of a purely data-based prior:

$$m_{\mathrm{MC}}(\xi) = \frac{1}{M}\sum_{m=1}^{M}\Phi^m(\xi)$$

$$k_{\mathrm{MC}}(\xi,\xi') = \frac{1}{M-1}\sum_{m=1}^{M}(\Phi^m(\xi) - m_{\mathrm{MC}}(\xi)) \cdot (\Phi^m(\xi') - m_{\mathrm{MC}}(\xi'))$$

GP posterior

$$m^+(\xi_*) = m_{\mathrm{MC}}(\xi_*) + k_{\mathrm{MC}}(\xi_*,\Xi)k_{\mathrm{MC}}(\Xi,\Xi)^{-1}(\gamma - m_{\mathrm{MC}}(\Xi))$$

$$k^+(\xi_*,\xi_*') = k_{\mathrm{MC}}(\xi_*,\xi_*') - k_{\mathrm{MC}}(\xi_*,\Xi)k_{\mathrm{MC}}(\Xi,\Xi)^{-1}k_{\mathrm{MC}}(\Xi,\xi_*')$$

▶ All covariances and means are computed empirically from the ensemble

- No need to assume specific structural knowledge or a specific, parametrized prior
- No restriction ro certain function classes
- The model fulfills linear physical constraints

The data needs to contain all relevant information

Physics-Informed Gaussian Processes

Embedding differential equations (DE)

[Long et al., 2022]

Example:

$$\partial_t \varphi - \nu \partial_\xi^2 \varphi + \beta \varphi(\varphi^2 - 1) + g(\xi) = 0$$

with target function $\varphi(\xi(t))$

Construct GP priors over φ and g:

$$\varphi(\xi) \sim \mathcal{GP}(m_\varphi(\xi), k_\varphi(\xi, \xi'))$$
$$g(\xi) \sim \mathcal{GP}(m_g(\xi), k_g(\xi, \xi'))$$

Learned functions are supposed to not only fit observations $\{\Xi, \varphi(\Xi)\}, \{\Xi, g(\Xi)\}$ but also to comply with the known differential eq.

- Idea: **enforce additionally satisfaction of the L.H.S. of the DE at collocation points** $\hat{\Xi}$

1. **Embedding the DE requires derivatives of φ**
   - GPs are closed under differentiation: derivative models are again GPs
2. Joint Gaussian prior $p(f) = \mathcal{N}(f \mid \mu, \Sigma)$ over $f = [\varphi; \hat{\varphi}; \hat{\varphi}_t; \hat{\varphi}_{\xi\xi}; \hat{g}]$, $\hat{\varphi}_\cdot = \varphi_\cdot(\hat{\Xi})$
3. Noise model: $p(\gamma \mid f) = \mathcal{N}(\gamma \mid f, \sigma_n^2 \mathbb{I})$
4. Virtual second likelihood to integrate DE: $p(0 \mid f) = \mathcal{N}(0 \mid \hat{\varphi}_t - \nu \hat{\varphi}_{\xi\xi} + \beta \hat{\varphi} \circ (\hat{\varphi} \circ \hat{\varphi} - 1) + g, \lambda \mathbb{I})$

- Joint prior model:

$$p(f, \gamma, 0) = \mathcal{N}(f \mid \mu, \Sigma)\mathcal{N}(\gamma \mid f, \sigma_n^2 \mathbb{I}) \cdot \mathcal{N}(0 \mid \hat{\varphi}_t - \nu \hat{\varphi}_{\xi\xi} + \beta \hat{\varphi} \circ (\hat{\varphi} \circ \hat{\varphi} - 1) + g, \lambda \mathbb{I})$$

Tune λ to enforce conformity to physics

For dedicated functional structures, DE embeddings can be done by design instead of a penalty

Physics-Informed Gaussian Processes

Physically consistent GP models of conservative Lagrangian systems [Evangelisti, Hirche, 2022]

Euler-Lagrange Dynamic Systems

$$\frac{\mathrm{d}}{\mathrm{d}t}(\nabla_{\dot{q}} L) - \nabla_q L = \tau$$

with Lagrangian $L = T(q, \dot{q}) - V(q)$

- Kinetic energy: $T(q, \dot{q}) = \dot{q}^\top M(q)\dot{q}$
- Potential energy: $V(q) = U(q) + G(q)$
- Elastic energy: $U(q) = q^\top S(q) q$
- Gravitational energy: $G(0) = 0, \ \nabla_q G(0) = 0$

► Alternatively represented by

$$\tau = M(q)\ddot{q} + C(q, \dot{q})\dot{q} + g(q)$$

Lagrange differential operator

$$\mathcal{L}_q = \frac{\mathrm{d}}{\mathrm{d}t}\nabla_{\dot{q}} - \nabla_q = (\nabla_{\dot{q}}^\top \ddot{q} + \nabla_q^\top \dot{q})\nabla_{\dot{q}} - \nabla_q$$

Model kinetic, elastic and gravitational energies by Gaussian processes

$$T(q, \dot{q}) \sim \mathcal{GP}(m_T(q, \dot{q}), k_T(q, q', \dot{q}, \dot{q}'))$$
$$U(q) \sim \mathcal{GP}(m_U(q), k_U(q, q'))$$
$$G(q) \sim \mathcal{GP}(m_G(q), k_G(q, q'))$$

GPs are closed under linear operators, yielding

$$\tau \sim \mathcal{GP}(m_\tau(q, \dot{q}, \ddot{q}), k_\tau(q, q', \dot{q}, \dot{q}', \ddot{q}, \ddot{q}'))$$

$$m_\tau = \mathcal{L}_q m_L, \ \ k_\tau = \mathcal{L}_q \mathcal{L}_{q'}^\top k_L$$

$$m_L = m_T - m_V, \ \ k_L = k_T + k_V$$

$$m_V = m_U + m_G, \ \ k_V = k_U + k_G$$

Physics-Informed Gaussian Processes

Physically consistent GP models of conservative Lagrangian systems [Evangelisti, Hirche, 2022]

Properties of Euler-Lagrange Dynamic Systems

1. Kinetic energy: $T(q,\dot{q}) = \dot{q}^\top M(q)\dot{q}$
   $M(q) = M^\top(q),\ M(q) \succ 0$
2. Elastic energy: $U(q) = q^\top S(q) q$
   $S(q) = S^\top(q),\ S(q) \succ 0$
3. Gravitational energy: $G(0) = 0,\ \nabla_q G(0) = 0$

► How to enforce compliance of the GP model with the known system properties?

Define GP prior over gravitational energy such that $m_G(0) = 0,\ \nabla_q m_G(0) = 0$

► Compliance with the equilibrium conditions on the gravitational energy

► can be easily ensured

Prior mean for kinetic and elastic energies:

$$m_T(q,\dot{q}) = \frac{1}{2}\dot{q}^\top M_0(q)\dot{q},\ m_U(q) = \frac{1}{2} q^\top S_0(q) q$$

Prior covariances for kinetic and elastic energies

$$k_T(q,q',\dot{q},\dot{q}') = \frac{1}{4}(\dot{q}\circ\dot{q})^\top \Theta_T(q,q')(\dot{q}\circ\dot{q}')$$

$$k_U(q,q') = \frac{1}{4}(q\circ q)^\top \Theta_U(q,q')(q\circ q')$$

$$\Theta_\cdot(q,q') = R_\cdot^\top(q,q') R_\cdot(q,q')$$

- $R_\cdot(q,q')$ is an upper triangular matrix of kernels

$$r(q,q') = \sigma_f^2 \exp\left(-\frac{1}{2}(q-q')^\top \Lambda^{-1}(q-q')\right)$$

The physically consistent GP possesses the properties of EL systems (with high probability)

Physics-Informed Gaussian Processes

Physically consistent GP models of port-Hamiltonian systems [Beckers et al., 2023]

Port-Hamiltonian sytems:

$$\dot{x} = [J(x) - R(x)]\nabla_x H(x) + G(x)u$$
$$y = G(x)^\top \nabla_x H(x)$$

- Hamiltonian $H(x)$ represents total energy
- Interconnection matrix $J(x)$
- Dissipation matrix $R(x)$
- $G(x)$ specifies interactions with environment

► Port-Hamiltonian formalism for energy-based modeling of multi-physics systems with energy exchange among different domains

The specific structure of port-Hamiltonian systems can be integrated in GP-based models

Port-Hamiltonian GP prior

- Place a GP prior over the Hamiltonian
  $$\hat{H} \sim \mathcal{GP}(0, k(x, x'))$$
  with SE kernel $k(x, x') = \sigma_f^2 \exp(\|x - x'\|^2_{\Lambda^{-1}})$
- Systems dynamics are an affine transformation of the Hamiltonian:
  $$\dot{\hat{x}} \sim \mathcal{GP}(G(x)u, k_{\text{PHS}}(x, x'))$$
  $$k_{\text{PHS}}(x, x') = \sigma_f^2 [J(x) - R(x)] \cdot \nabla_x^2 k(x, x') [J(x) - R(x)]^\top$$

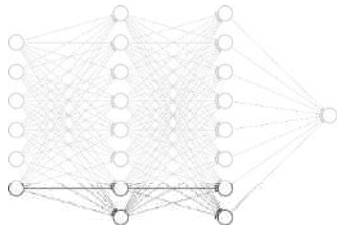
This restricts the sample space of the GP to port-Hamiltonian systems ► any realization drawn from the GP is guaranteed to represent a PHS

The physics-informed GP model is guaranteed to have the same system properties almost surely

Physics-Informed Learning for Model-based Control

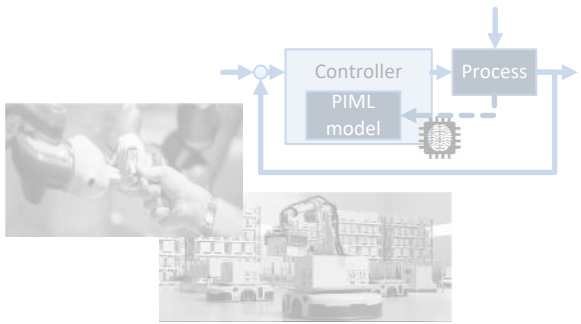
Motivation

- Fusing control and machine learning to achieve autonomy of systems
- Physics-informed ML: integrate known parts, infer unknown parts from data



Physics-informed machine learning

- Integrating physics in Gaussian process models
- Integrating physics in neural network models

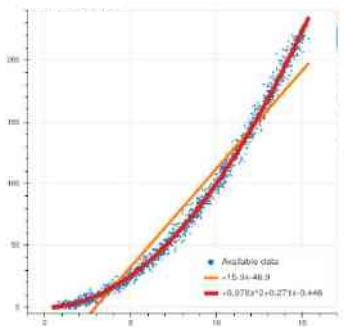


Physics-informed learning for model-based control

- Why model-based control and why physics-informed models?
- Physics-informed model predictive control

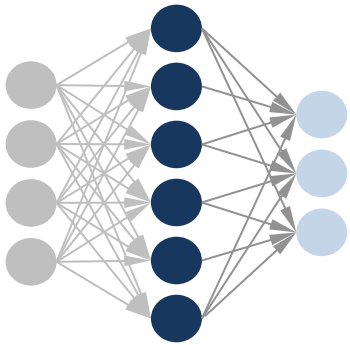
Neural Networks Refresher

Linear and nonlinear least-squares regression



- Linear least-squares regression given a set of data: $\gamma = a\xi + b$
- Extension of this set-up to nonlinear functions: $\gamma = \sum_{i=0}^{n} c_i \phi_i(\xi)$
- **Problem:** The correct parametric model must be available

Neural networks



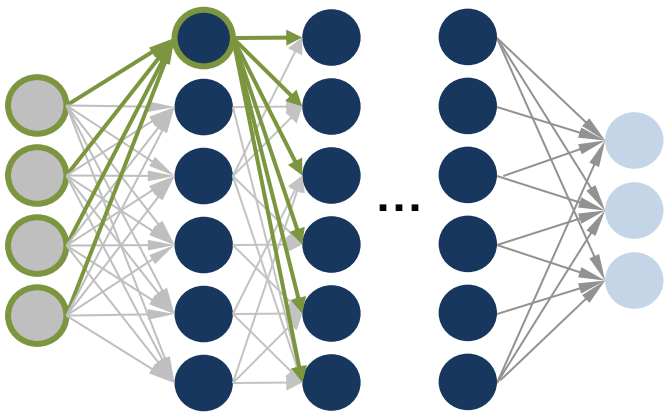
- Idea: **Use repeated regression to introduce required nonlinearities**
- Input processing by passing it subsequently through several individual regression problems
- Application in Robotics, Automation, …
- Bio-inspired model architecture

Neural networks are nonparametric, universal function approximators

Neural Networks: Basic Idea

General architecture of (feedforward) neural networks

Input layer ξ	Hidden layers z	Output layer γ



Feedforward NNs: only forward connections between neurons

- Input layer: incoming values $z^{(0)} = \xi$
- Every **hidden layer neuron** in the network is of the form

$$z_j^{(i)} = \sigma\left(A_{j:}^{(i)} z^{(i-1)} + B_j^{(i)}\right)$$

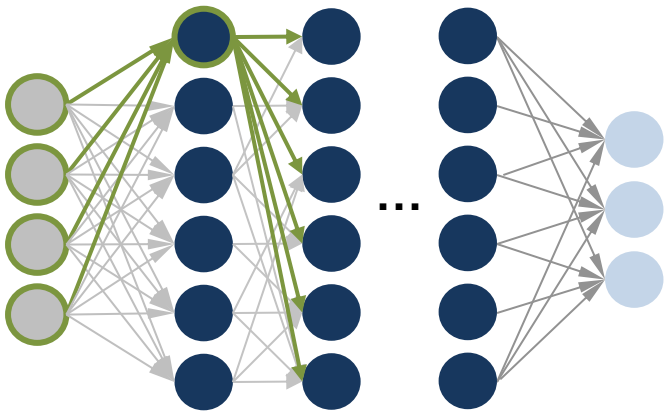
i : Number of the hidden layer
j : Number of the neuron
$A^{(i)}$: Parameter matrix of layer i
$B^{(i)}$: Bias vector of layer i

- Activation function $\sigma(\cdot)$ has often some switch-like characteristics
- Characterization of different architectures by types of connections and number of layers

Neural Networks: Basic Idea

General architecture of (feedforward) neural networks

Input layer ξ	Hidden layers z	Output layer γ



Feedforward NNs: only forward connections between neurons

Objective:

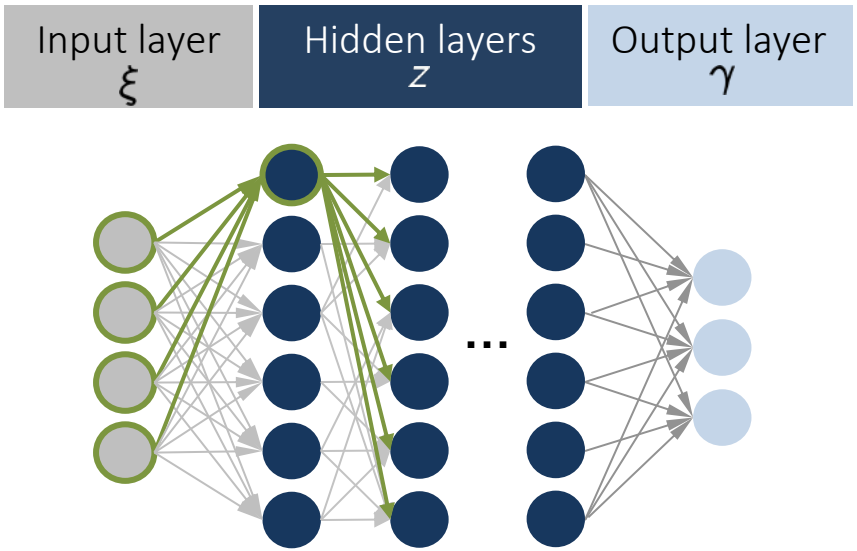
- Employ the neural network $\hat{\gamma} = \mathrm{NN}(\xi)$ to learn a representation of an unknown function
$$\gamma = \varphi(\xi)$$
- ▶ Infer unobserved function values $\gamma_* = \varphi(\xi_*)$ at test inputs ξ_*

- **The NN outputs depend on the parameters**, i.e., $\hat{\gamma} = \mathrm{NN}(\xi; \theta)$ with parameters
$$\theta = \{(A^{(i)}, B^{(i)}) \mid i = 1, \ldots, n_l\}$$
- ▶ The parameters need to be adapted to match the underlying problem
- ▶ Train the network on available data

How to train the parameters of the network such that it learns to represent the underlying function?

Neural Networks: Training

How can we determine suitable network parameters from training data?



Given training data $\{(\xi_i, \gamma_i) \mid i = 1, \ldots, n\}$ and Network architecture $\mathrm{NN}(\cdot; \theta)$, iterate:

i. **Propagation** of the training inputs through the network: $\hat{\gamma}_i = \mathrm{NN}(\xi_i; \theta), \; i = 1, \ldots, n$

ii. **Compute the loss, i.e., a measure for the NN's error**, e.g.: $L = \frac{1}{n} \sum_{i=1}^{n} \|\gamma_i - \hat{\gamma}_i\|_2^2$

iii. **Backpropagation** of the loss through the network layer by layer to compute the gradients $\nabla_\theta L$

iv. **Parameter update** by stochastic gradient descent algorithm exploiting $\nabla_\theta L$

Training a neural network is an optimization problem. Similarly to GPs, one can

- **embed physical knowledge in the loss function / optimization criterion**
- **enforce function classes via network structure**
- …

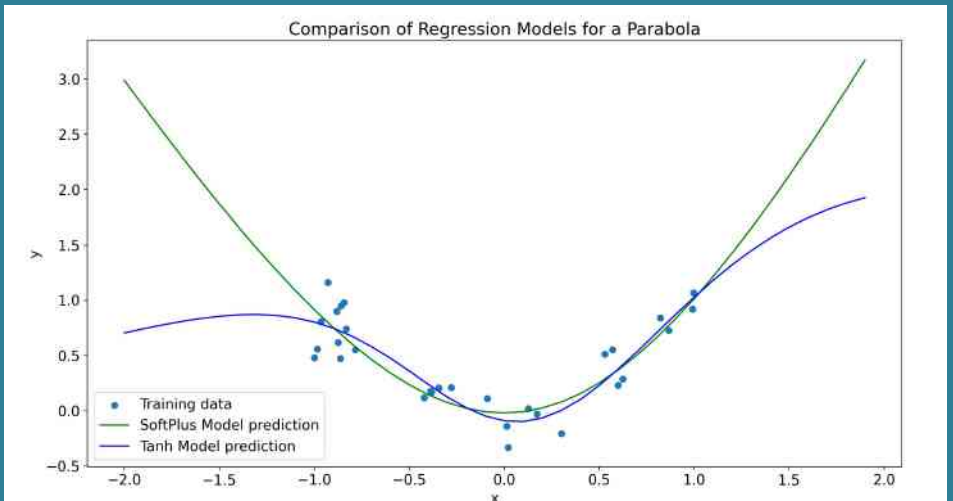
Constrained Neural Networks

- **Restrict parameters or architecture** such that the NN is, e.g.,
  - Convex
  - Lipschitz continuous
  - Monotonous
  - Differentiable
- **Enforce via**
  - Hard-constraints
  - Via **cost function** / regularization
  - Structure of the network itself

- Allows to incorporate prior knowledge
- Can be used to enforce system properties

Example: Convex NN

Fit $y(x) = x^2 + \epsilon, \quad \epsilon \sim \mathcal{N}(\mu, \sigma^2)$



Physics-informed Neural Networks (PINNs)

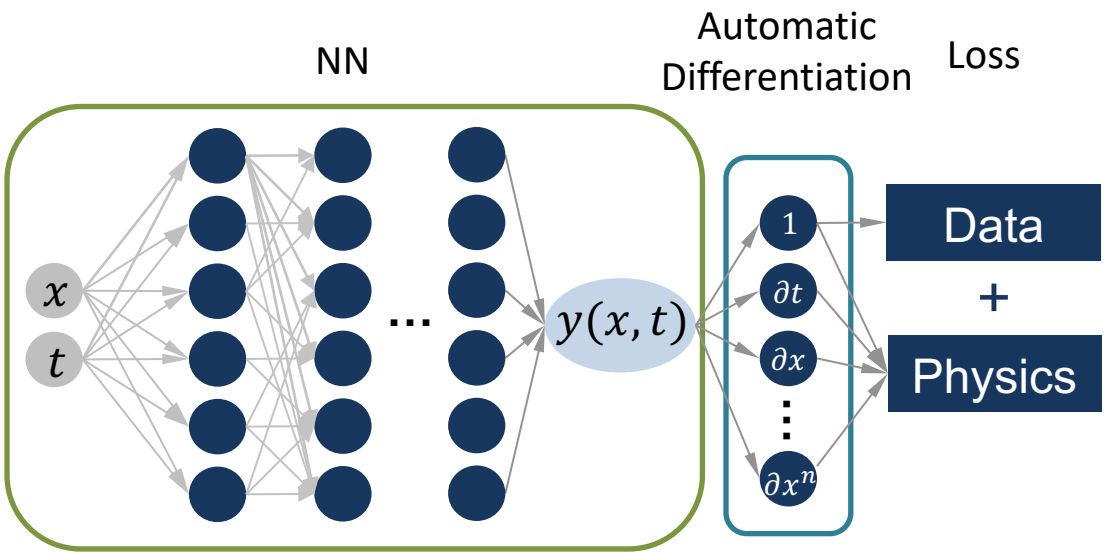
Model needs to satisfy differential operator (physical insight)

$$\mathcal{F}(y(x,t), \frac{\partial y}{\partial t}, \frac{\partial y}{\partial x}, \cdots, \frac{\partial^n y}{\partial x^n}) = 0$$

Penalize loss using system model

$$\mathcal{L}(\theta, \mathcal{D}) = \sum_i \left(y_i - \mathrm{NN}\left(x_i, t_i; \theta\right)\right)^2 + \lambda \sum_i \mathcal{F}(\cdot)^2\big|_{x_i, t_i}$$

- Improves sample efficiency by regularization
- Predictions generally do not strictly comply with physics



Restricted to autonomous systems ► How to extend to controlled systems?

Physics-informed Neural Networks for Control (PINCs)

"PINNs do not allow control inputs, neither can they simulate for variable long-range intervals without serious degradation in their predictions"

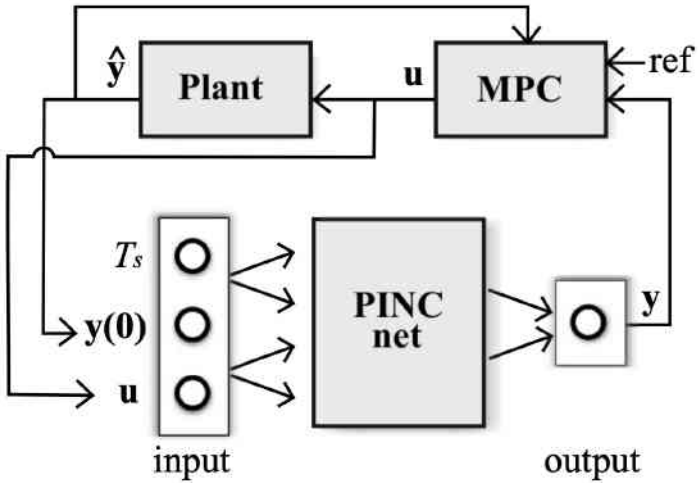
- Same basic principle as PINNs
- Introduce control input u
- Train the network for a sampling period T_s

► NN learns the solution of the system dynamics

- Multi-step predictions by repeated evaluations

Related to / motivated by direct multiple shooting

- Can be combined with model-based control
- Allows for long time prediction



[Antonelo et al. 2022]

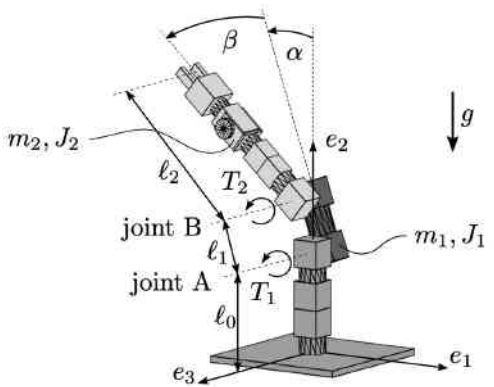
Example: PINCs for Control of Multi-Link Manipulators

Use PINC to learn forward dynamics of multilink manipulator.

Given the generalized coordinates $q = [\alpha, \beta]^T$ and the input currents u, the system can be described by:

$$0 = \begin{bmatrix} I & 0 \\ 0 & M(q) \end{bmatrix} \begin{bmatrix} \dot{q} \\ \ddot{q} \end{bmatrix} - \begin{bmatrix} \dot{q} \\ h(q, \dot{q}) - k(q, \dot{q}) \end{bmatrix} - \begin{bmatrix} 0 \\ Bu \end{bmatrix}$$

- With $x = \begin{bmatrix} q & \dot{q} \end{bmatrix}^\top$, we denote $0 = \mathcal{F}(x, \dot{x}, u)$
- zero-order hold for u and sampling time T_s
- ▶ Solution: $x_{k+1} = \phi(T_s, u_k, x_k)$



[Nicodemus et al. 2022]

Learn DNN to approximate $\hat{x}(t) = \hat{\phi}_\theta(t, u_k, x_0) \approx \phi(t, u_k, x_0)$
Sample data $x_i = \phi(t_i, u_i, x_{0,i})$ and select weights θ to minimize

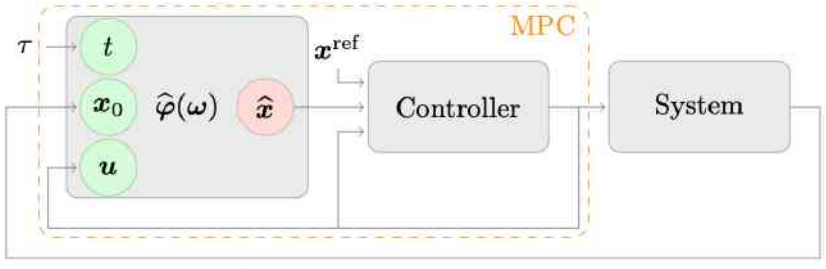
$$\sum_i \left\| \hat{\phi}_\theta(t_i, u_i, x_{0,i}) - x_i \right\|^2 + \lambda \left\| \mathcal{F}(\hat{\phi}_\theta(t_i, u_i, x_{0,i}), \frac{\partial}{\partial t} \hat{\phi}_\theta(t_i, u_i, x_{0,i}), u_i) \right\|^2$$

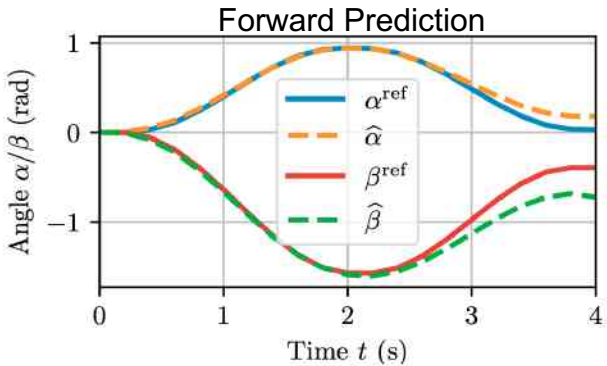
PINCs can achieve good data fit and conformity with physics, traded off by λ

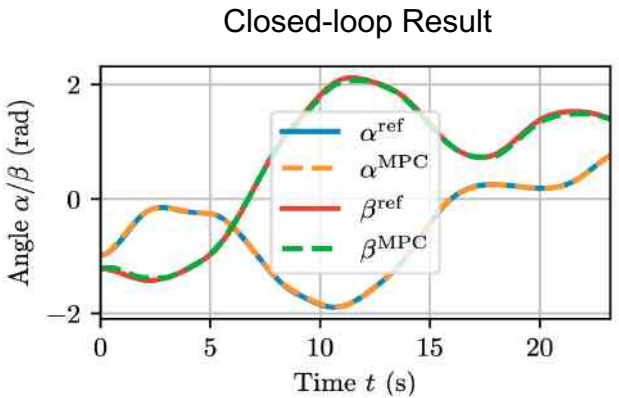
Example: PINCs for Control of Multi-Link Manipulators

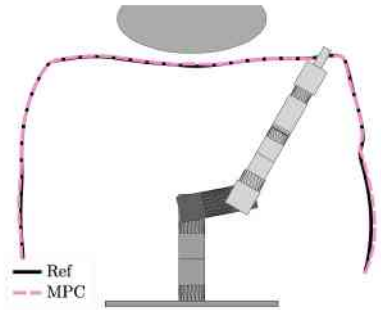
Evaluate the model and combine with MPC to track a reference trajectory.

- ▶ Evaluation of PINN is faster than solving the dynamics
- ▶ Enables for example real time MPC









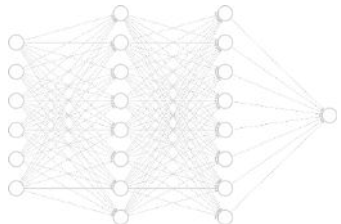
[Nicodemus et al. 2022]

PINCs enable the efficient combination of prior system knowledge and data for control tasks

Physics-Informed Learning for Model-based Control

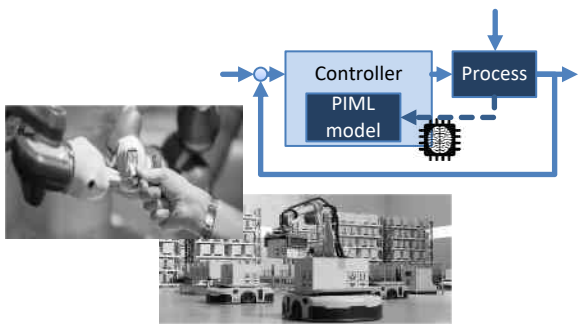
Motivation

- Fusing control and machine learning to achieve autonomy of systems
- Physics-informed ML: integrate known parts, infer unknown parts from data



Physics-informed machine learning

- Integrating physics in Gaussian process models
- Integrating physics in neural network models

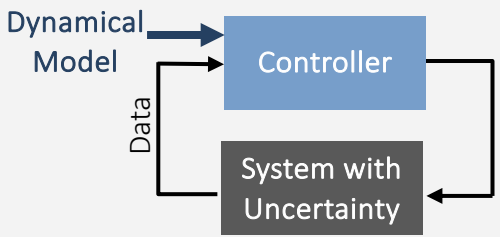


Physics-informed learning for model-based control

- Physics-informed model predictive control (same examples)

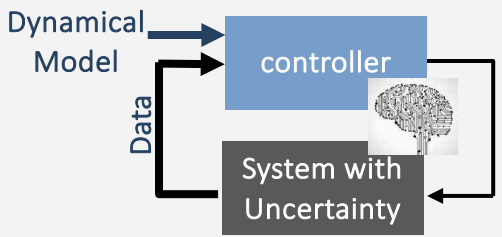
Fusing Learning and Model-based Control

Model-based control



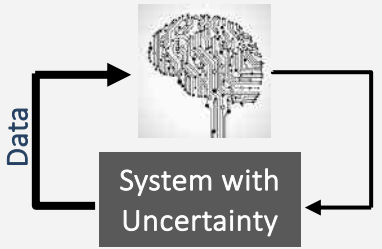
+ Exploits physical knowledge
+ Provides physical insight
+ Exploits model and data
+ Guarantees possible
- difficult to handle large changes

Learning-supported control



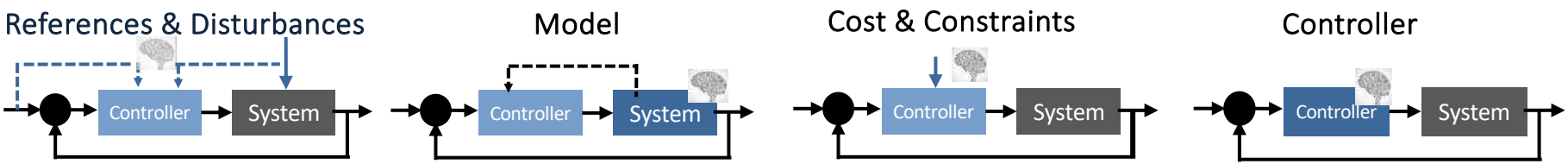
+ Exploits physical knowledge
+ Provides physical insight
+ Exploits model and data
+ Guarantees possible
+ Allows adaptation

Learning-based control



+ Exploits available data
+ No physical model needed
- Lots of data needed
- Physical insight, guarantees?
- Retraining required

The performance of controllers can be significantly improved if physics-informed models are employed



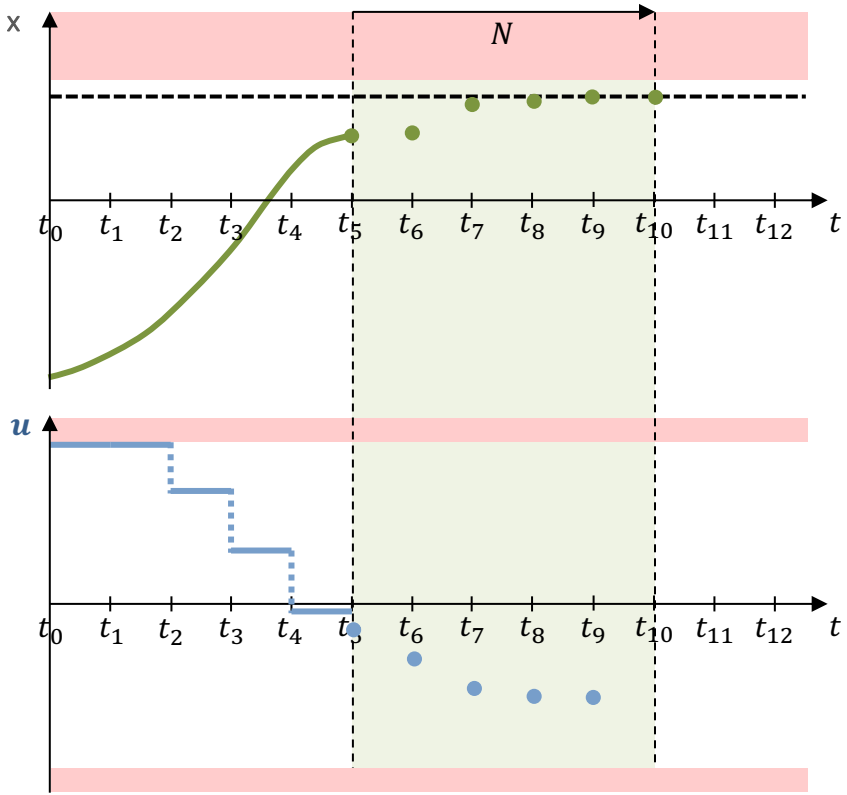
Model-based Control: Model Predictive Control

MPC repeatedly solves a finite-horizon optimal control problem:

$$\min_{u_k} \left\{ \sum_{i=0}^{N-1} F(x_{i|k}, u_{i|k}) + V_f(x_{N|k}) \right\}$$

$$\text{s. t. } \forall i \in \mathcal{I}_{0:N-1} : x_{i+1|k} = f(x_{i|k}, u_{i|k}),\ x_{0|k} = x_k$$
$$\forall i \in \mathcal{I}_{0:N-1} : x_{i|k} \in \mathcal{X},\ u_{i|k} \in \mathcal{U}$$
$$x_{N|k} \in \mathcal{E}$$

1) Obtain x_k & solve OCP at time point $\mathbf{t}_k$



Model-based Control: Model Predictive Control

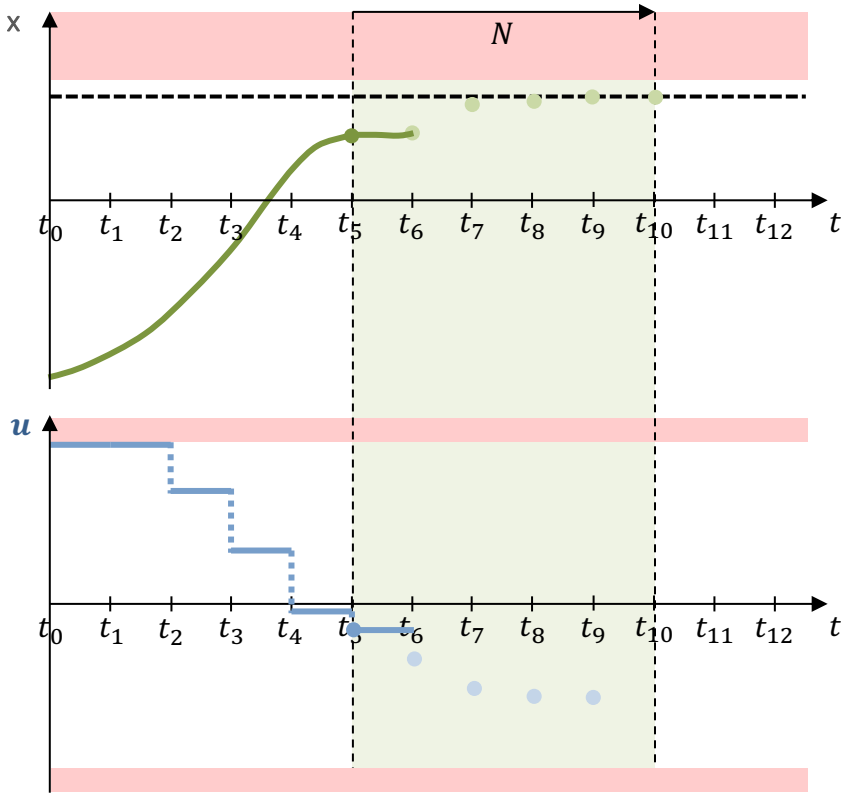
MPC means to repeatedly solve a finite-horizon optimal control problem:

$$\min_{u_k} \left\{ \sum_{i=0}^{N-1} F(x_{i|k}, u_{i|k}) + V_f(x_{N|k}) \right\}$$

$$\text{s. t. } \forall i \in \mathcal{I}_{0:N-1} : x_{i+1|k} = f(x_{i|k}, u_{i|k}),\ x_{0|k} = x_k$$
$$\forall i \in \mathcal{I}_{0:N-1} : x_{i|k} \in \mathcal{X},\ u_{i|k} \in \mathcal{U}$$
$$x_{N|k} \in \mathcal{E}$$

1) Obtain x_k & solve OCP at time point $\mathbf{t}_k$

2) Employ $u(t) = u_{0|k}$ for $t \in [t_k, t_{k+1}]$



Model-based Control: Model Predictive Control

MPC means to repeatedly solve a finite-horizon optimal control problem:

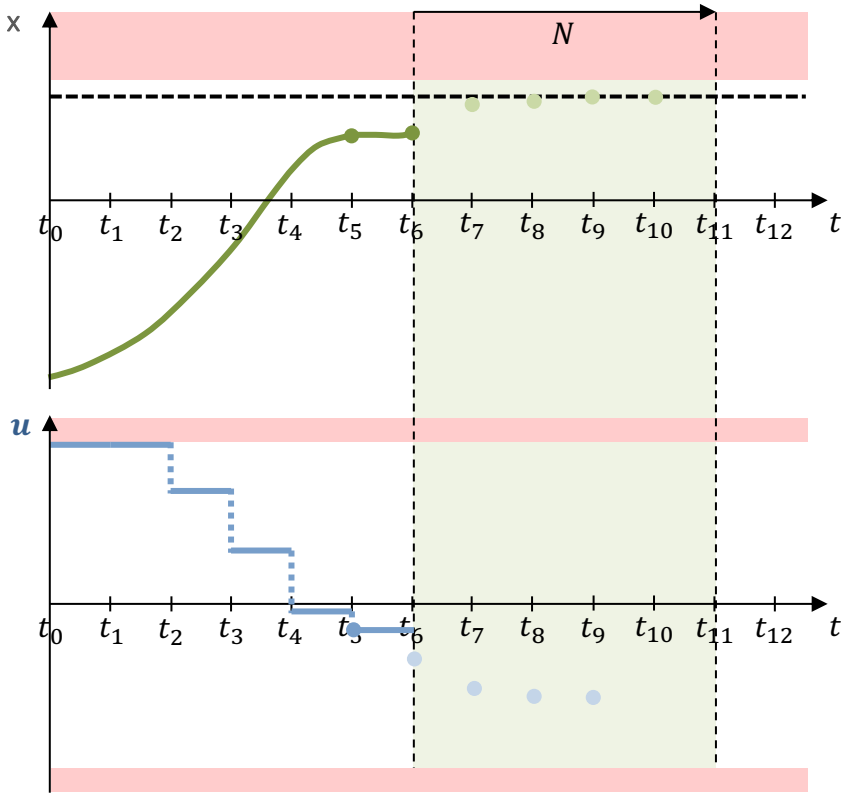
$$\min_{u_k} \left\{ \sum_{i=0}^{N-1} F(x_{i|k}, u_{i|k}) + V_f(x_{N|k}) \right\}$$

$$\text{s. t. } \forall i \in \mathcal{I}_{0:N-1} : x_{i+1|k} = f(x_{i|k}, u_{i|k}),\ x_{0|k} = x_k$$
$$\forall i \in \mathcal{I}_{0:N-1} : x_{i|k} \in \mathcal{X},\ u_{i|k} \in \mathcal{U}$$
$$x_{N|k} \in \mathcal{E}$$

1) Obtain x_k & solve OCP at time point t_k

2) Employ $u(t) = u_{0|k}$ for $t \in [t_k, t_{k+1}]$

3) Shift horizon by one step, i.e., $\mathrm{t}_k \leftarrow t_{k+1}$



Model-based Control: Model Predictive Control

MPC means to repeatedly solve a finite-horizon optimal control problem:

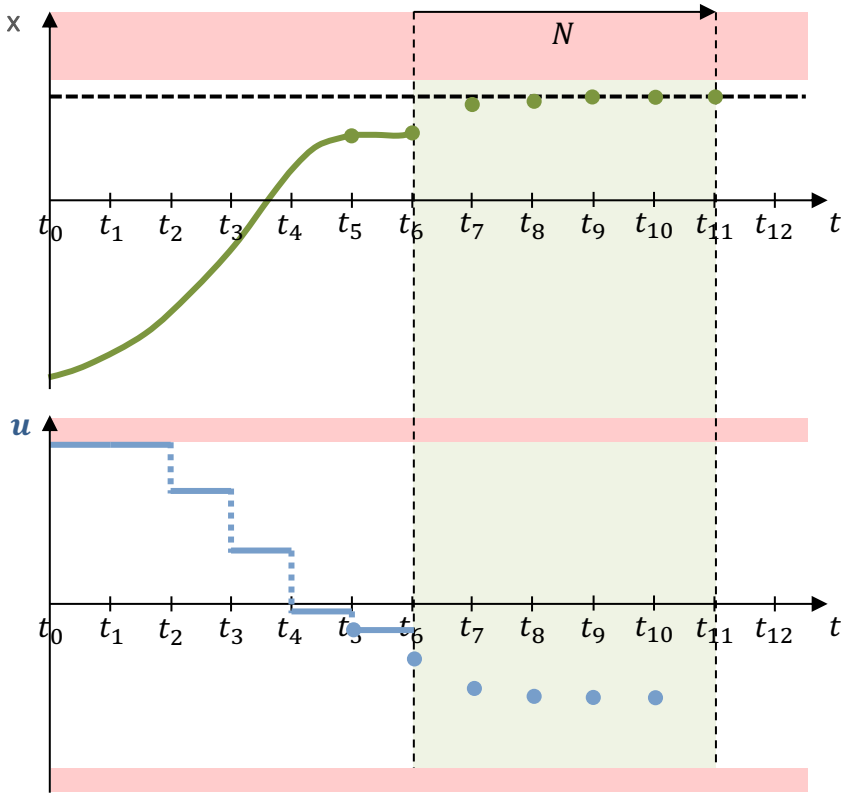
$$\min_{u_k} \left\{ \sum_{i=0}^{N-1} F(x_{i|k}, u_{i|k}) + V_f(x_{N|k}) \right\}$$

$$\text{s. t. } \forall i \in \mathcal{I}_{0:N-1} : x_{i+1|k} = f(x_{i|k}, u_{i|k}),\ x_{0|k} = x_k$$
$$\forall i \in \mathcal{I}_{0:N-1} : x_{i|k} \in \mathcal{X},\ u_{i|k} \in \mathcal{U}$$
$$x_{N|k} \in \mathcal{E}$$

1) Obtain x_k & solve OCP at time point t_k

2) Employ $u(t) = u_{0|k}$ for $t \in [t_k, t_{k+1}]$

3) Shift horizon by one step, i.e., $t_k \leftarrow t_{k+1}$



Model-based Control: Model Predictive Control

1. Obtain current state state $x_k = x(t_k)$
2. Predict & optimize system evolution
   $\min_{\bar{u}} \{J(x_k, \bar{u})\}$
   s. t. Dynamics
   Constraints
3. Apply optimal input until t_{k+1}

$$\min_{\bar{u}_k} \left\{ \int_{t_k}^{t_k+T} F(\tau, \bar{x}(\tau), \bar{u}(\tau)) + E(\bar{x}(\tau+T)) \right\}$$

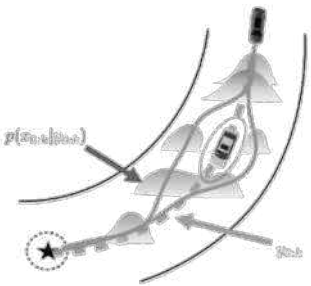
s. t. $\forall \tau \in [t_k, t_k + T]$:

$$\dot{\bar{x}}(\tau) = f(\bar{x}(\tau), \bar{u}(\tau)) \ , \ \bar{x}(t_k) = x(t_k)$$

$$\bar{y}(\tau) = h(\bar{x}(\tau))$$

$$\bar{u}(\tau) \in \mathcal{U}, \bar{x}(\tau) \in \mathcal{X}$$

$$\bar{x}(t_k + T) \in \Omega$$



Advantages of MPC

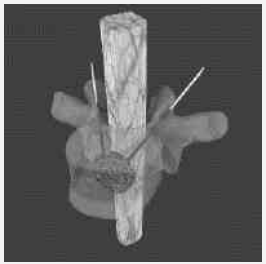
- Direct consideration of constraints ▶ safety
- Nonlinear, multiple inputs/output systems ▶ flexibility
- Use of preview information and adaptation of model/parameters ▶ adaptiveness
- Possibility to “optimize” a cost ▶ performance
- Many stability & robustness results available ▶ stability & robustness
- Efficient embedded optimization strategies

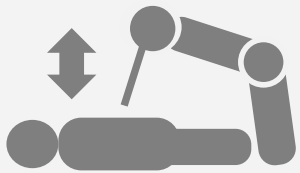
Perfect knowledge of system & environment required ▶ use machine learning to adapt to the unknown

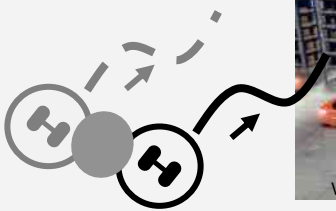
Example: “Physics”-Informed MPC

Objective: increased autonomy by learning from interaction
► Learn **reference signal or disturbance model** to support desired interaction with environment

Examples







Control Task (Reference Trajectory Tracking)

Given output reference $r_{tt} : [0, \infty) \to \mathbb{R}^{n_y}, t \mapsto y^r(t)$ and system $\dot{x}(t) = f(x(t), u(t)), \; y(t) = h(x(t))$, design controller that achieves

i. Convergence to reference:

$$\lim_{t \to \infty} \|y(t) - r_{tt}(t)\| = 0$$

ii. Constraint satisfaction:

$$\forall t \geq 0 : x(t) \in \mathcal{X}, u(t) \in \mathcal{U}, y(t) \in \mathcal{Y}$$

The reference is not always known

Support controller with a learned reference to ensure safe and desired interaction with environment

Example: Physics-Informed MPC

Learning Task: Reference Generator for Trajectory Tracking

Learn a reference generator $g : [t_k, t_k + T] \to \mathcal{Y} \subset \mathbb{R}^{n_y}$ from data $\mathcal{D}$, which provides the reference $y^r(\tau) = g(\tau; \mathcal{D}), \forall \tau \in [t_k, t_k + T]$ that is used in the MPC, such that

i. **Trackability:** The reference is trackable under output, state and input constraints, i.e.,

$$\forall \tau \in [t_k, t_k+T] : \exists u^r(\tau) \in \mathcal{U} : y^r(\tau) = h(x^r(\tau)) \in \mathcal{Y}, x^r(\tau) \in \mathcal{X}$$

ii. **Data fit:** The reference is data-consistent (and preferably of low complexity)

$$\forall i = 1, \ldots, N : y^r(t_i) \approx r_i, \text{ where } (t_i, r_i) \in \mathcal{D}$$

Data fit: Use GP as reference generator

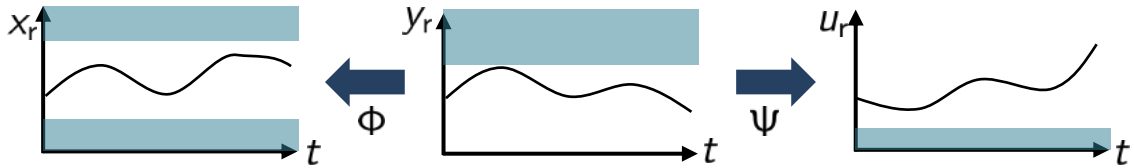
Trackability & constraint satisfaction: Constrain learning process exploiting system properties

Data fit is trivially achieved; How to ensure trackability / safety of the learned reference?

Example: Physics-Informed MPC

Assumption: the system under consideration is differentially flat

$$\exists \Phi, \Psi : x_r = \Phi\left(y_r, \dot{y}_r, \ldots, y_r^{(\beta-1)}\right)$$
$$u^r = \Psi\left(y_r, \dot{y}_r, \ldots, y_r^{(\beta)}\right)$$



Integrating physics by constrained hyperparameter optimization

$$\theta^* = \arg\max_{\theta} \{\log(p(\gamma \mid \chi, \theta))\}$$

s. t. $\forall \tau \in [\,0\,, \bar{t}\,]$:

$$m^+(\tau; \mathcal{D}, \theta) \in \mathcal{Y}$$

$$\Phi\left(m^+(\tau; \mathcal{D}, \theta), \dot{m}^+(\tau; \mathcal{D}, \theta), \ldots, m^{+(\beta-1)}(\tau; \mathcal{D}, \theta)\right) \in \mathcal{X}$$

$$\Psi\left(m^+(\tau; \mathcal{D}, \theta), \dot{m}^+(\tau; \mathcal{D}, \theta), \ldots, m^{+(\beta)}(\tau; \mathcal{D}, \theta)\right) \in \mathcal{U}$$

Standard training

System dynamics

Constraints

Theorem [Matschek *et al.*, 2021]:
Feasibility of hyperparameter optimization guarantees trackability and constraint satisfaction for $[0, T]$

Example: Physics-Informed MPC

Can we obtain a safe reference trajectory (constraint satisfaction) all times?

Example periodic references ▶ use stationary, periodic kernel $k(t, t') = k(t, t' + nT_p)$

Constrained hyperparameter optimization

$$\theta^* = \arg\max_{\theta} \{\log(p(\gamma \mid \chi, \theta))\}$$

$$\text{s. t. } \forall \tau \in [\,0\,, \bar{t}\,]:$$

$$m^+(\tau; \mathcal{D}, \theta) \in \mathcal{Y}$$

$$\Phi\left(m^+(\tau; \mathcal{D}, \theta), \dot{m}^+(\tau; \mathcal{D}, \theta), \ldots, m^{+(\beta-1)}(\tau; \mathcal{D}, \theta)\right) \in \mathcal{X}$$

$$\Psi\left(m^+(\tau; \mathcal{D}, \theta), \dot{m}^+(\tau; \mathcal{D}, \theta), \ldots, m^{+(\beta)}(\tau; \mathcal{D}, \theta)\right) \in \mathcal{U}$$

$$T_p \leq \bar{t}$$

Coverage constraint

- Ensure safety & trackability for at least one period
- ▶ Safety & trackability for all periods

Trackability always ▶ recursive feasibility & stability certificates of MPC via standard approaches [Matschek *et al.*, 2021]

Example: Physics-Informed MPC for Mobile Robots

Control Task

Synchronize motion of ego robot to that of target robot, which is unknown to ego

- ► Learn motion of ego robot as reference

Model

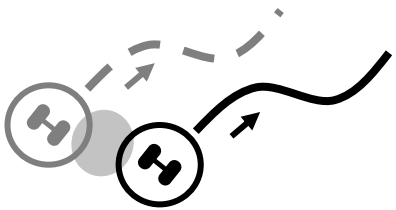
$$\dot{x}_1 = u_1 \sin(x_3) \qquad x_r = \Phi(y_r, \dot{y}_r) = \left(y_{r,1}, y_{r,2}, \operatorname{atan}\left(\frac{\dot{y}_{r,2}}{\dot{y}_{r,1}}\right) + i\pi\right)^T$$

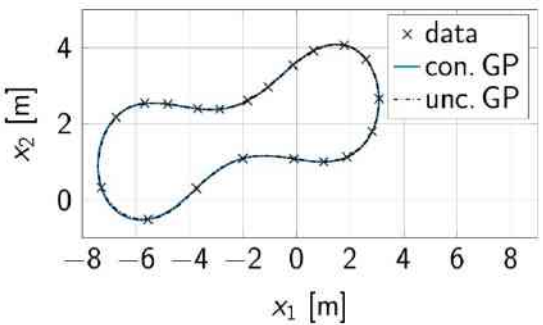
$$\dot{x}_2 = u_1 \cos(x_3)$$

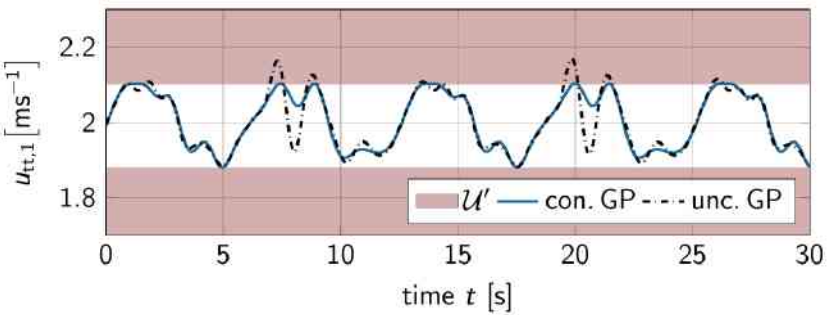
$$\dot{x}_3 = u_2 \qquad u_r = \Psi(y_r, \dot{y}_r, \ddot{y}_r) = \left(\|\dot{y}_r\|_2^2, \frac{\ddot{y}_{r,2}\dot{y}_{r,1} - \dot{y}_{r,2}\ddot{y}_{r,1}}{\|\dot{y}_r\|_2^2}\right)^T$$

$$y = (x_1, x_2)$$

Learned reference trajectories – Outputs and inputs



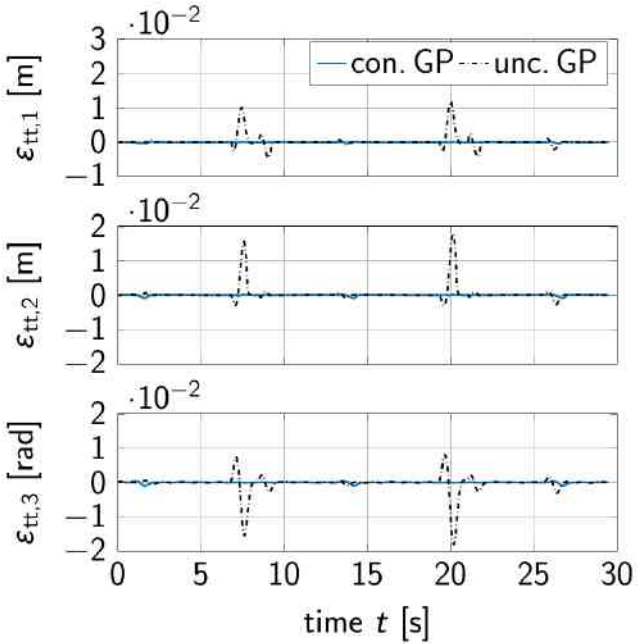




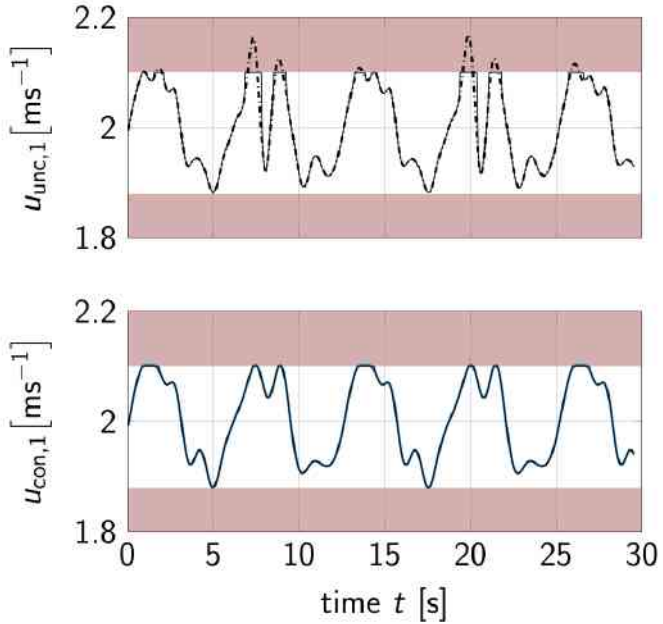
Unconstrained GP learning yields an unsafe reference trajectory that violates constraints

Example: Physics-Informed MPC for Mobile Robots

State tracking errors



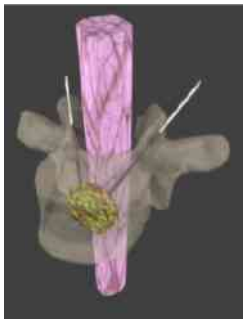
Applied control inputs

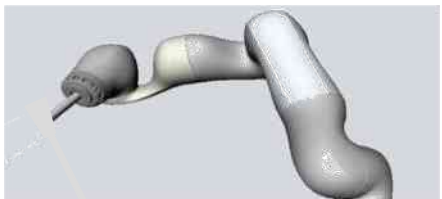
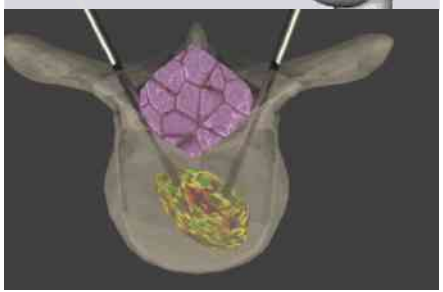
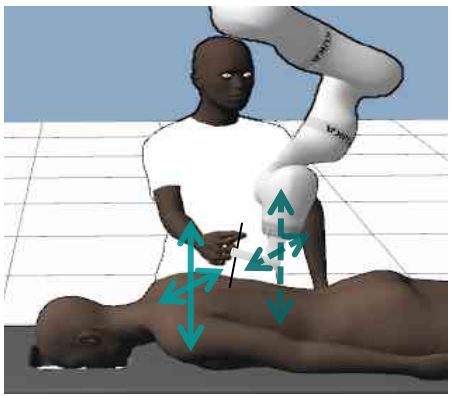


Exploiting physically constrained learning for reference generation allows to significantly improve control performance

Example: Physics-Informed MPC for Medical Robotics

Medical Task: Robot-assisted radio-frequency ablation of spinal cord tumors (i.e., heat ablation)

Challenges

- Complete ablation of tumor despite uncertainties
- Avoid thermal overheating of spinal cord and damage to non-tumor-tissue
- Satisfaction of constraints
- “Humans in the loop“

Requirements

- Precise positioning and stabilization of needles for ablation in the spinal cord
- Precise control of applied forces
- Adaptation to changing conditions
- Disturbance rejection/attenuation

Robot supported intervention

Example: Physics-Informed MPC for Medical Robotics

Many subtasks have to be solved

- **Planning** optimal and collision-free paths
- **Selecting/Planning** optimal robot configurations
- **Tracking** of desired needle positions
- **Measuring** needle positions, registration of events
- **Human-Robot-Interface**
- **Controlling** contact forces

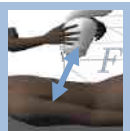
Therapy planning

Exp. Design Optimal Control

Path-following under constraints

Constrained path-following control

Force limitation & compensation



Force control

The patient is breathing

► Motion compensation (unknown reference)

Example: Physics-Informed MPC for Medical Robotics

Learning patient-specific references for breathing motion compensation using GPs

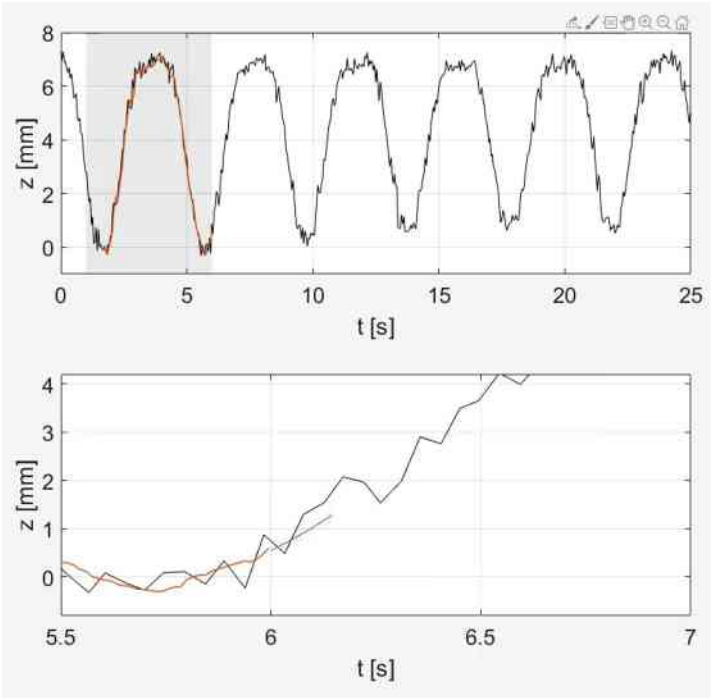
Clinical data

- Time-invariance of hyperparameters
- ► Patient-specific offline estimation

Breathing motion modeling

- Individual GP with **periodic kernel** for each quantity; coupling via constraints
- **Smooth** predictions/reference signals
- Extrapolation with decent quality due to selected prior

Improved performance and guaranteed trackability using constrained GP



Use patient-specific hyperparameters and safe data updates for online reference predictions

Example: Physics-Informed MPC for Medical Robotics

Results on motion compensation using GP-supported MPC

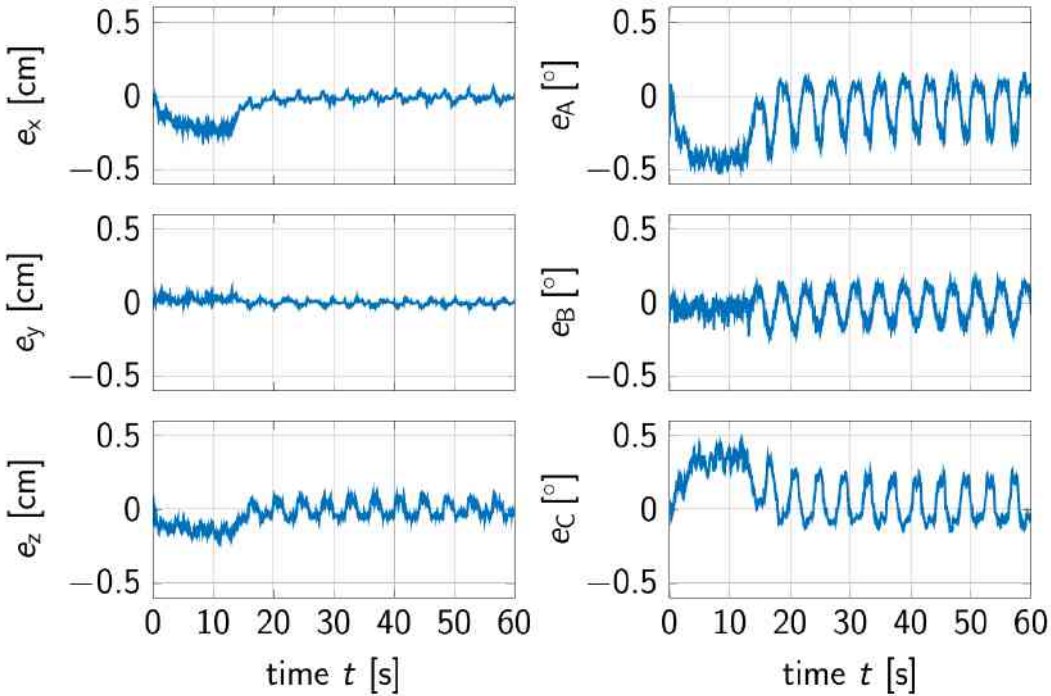
MPC

- Horizon length: 30-time steps
- Sampling time: 4 ms
- Reference: GP mean predictions; variances not (directly) exploited

Average control errors

$$e_x = 0.97\,\text{mm},\ e_A = 0.23\,^\circ$$
$$e_y = 0.20\,\text{mm},\ e_B = 0.10\,^\circ$$
$$e_z = 0.78\,\text{mm},\ e_C = 0.19\,^\circ$$

Low tracking errors and constraint satisfaction at all times



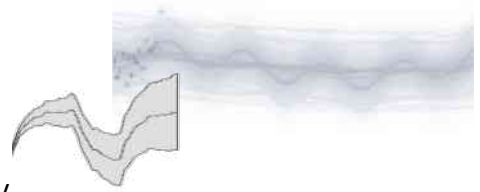
Physics-informed learning-supported MPC for motion compensation provides very promising results

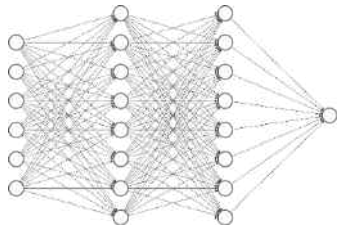
Conclusions and Outlook

Physics-informed learning

- Models incorporate physical knowledge to better represent real-world systems
- GPs and neural networks can be enforced to comply with physical knowledge by design or by penalty

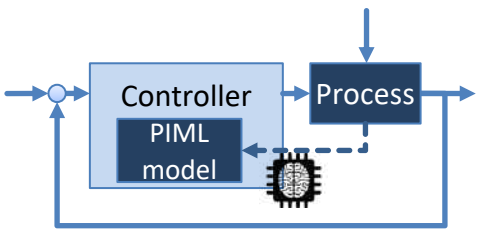
► Interpretable high-quality models

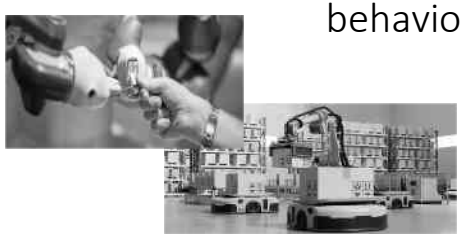




Model-based control using physics-informed ML models

- Adaptiveness and uncertainty quantification (ML part) while explainable and reasonable behavior (physics)
- Improved control performance: desired system behavior is physically realizable





Outlook

- Physics-informed learning of closed-loop models (controller + model + constraints + …) with desirable properties

TECHNISCHE UNIVERSITÄT DARMSTADT

Integrating Physics in Learning for Model-Based Control

Rolf Findeisen, Maik Pfefferkorn, Sebastian Hirt, Hoang Hai Nguyen, Andreas Höhl

Control and Cyber-Physical Systems Laboratory
Technical University of Darmstadt, Germany