Data-Efficient Reinforcement Learning for Autonomous Robots

Marc Deisenroth

Centre for Artificial Intelligence
Department of Computer Science
University College London

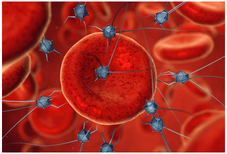
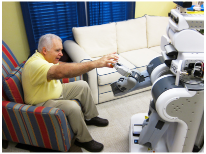
m.deisenroth@ucl.ac.uk

@mpd37

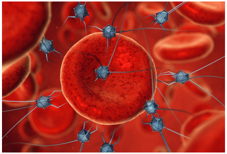
IBIS 2019, Nagoya

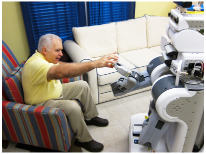
November 20, 2019

Autonomous Robots

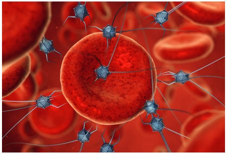
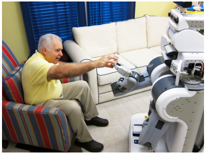
- Vision: Autonomous robots support humans in everyday activities ⏩ Fast learning and automatic adaptation






- Vision: Autonomous robots support humans in everyday activities ⏩ Fast learning and automatic adaptation
- Currently: Data-hungry learning or human guidance

Autonomous Robots

- Vision: Autonomous robots support humans in everyday activities ⏩ Fast learning and automatic adaptation
- Currently: Data-hungry learning or human guidance

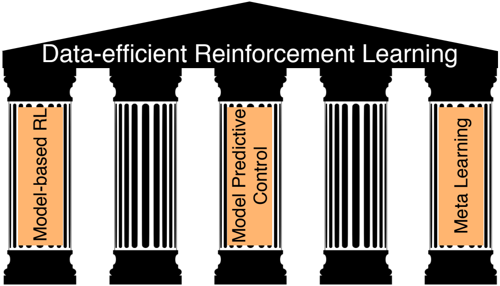
Fully **autonomous learning and decision making with little data** in real-life situations

Central Problem

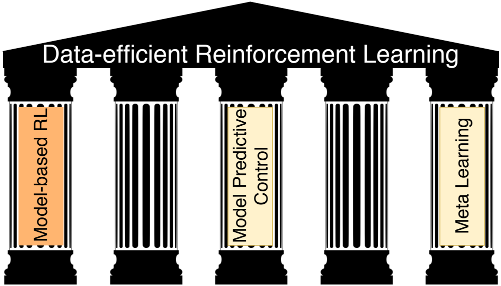
Data-Efficient Reinforcement Learning

Ability to learn and make decisions in complex domains without requiring large quantities of data

Data-Efficient RL for Autonomous Robots



1. **Model-based RL**
   - Data-efficient decision making
2. **Model predictive RL**
   - Speed up learning further by online planning
3. **Meta learning**
   - Generalization of knowledge to new situations


Data-efficient Reinforcement Learning
Model-based RL
Model Predictive Control
Meta Learning

Reinforcement Learning

$$\underbrace{\boldsymbol{x}_{t+1}}_{\text{State}} = \underbrace{f}_{\text{Transition function}}(\boldsymbol{x}_t, \underbrace{\boldsymbol{u}_t}_{\text{Control}}) + \boldsymbol{w}\,, \qquad \boldsymbol{u}_t = \underbrace{\pi}_{\text{Policy}}(\boldsymbol{x}_t, \underbrace{\boldsymbol{\theta}}_{\text{Policy parameters}})$$

Reinforcement Learning

$$\boldsymbol{x}_{t+1} = f(\boldsymbol{x}_t, \boldsymbol{u}_t) + \boldsymbol{w}\,, \qquad \boldsymbol{u}_t = \pi(\boldsymbol{x}_t, \boldsymbol{\theta})$$

State　Control　Policy　Policy parameters

Transition function

Objective (Controller Learning)

Find policy parameters $\boldsymbol{\theta}^*$ that minimize the expected long-term cost

$$J(\boldsymbol{\theta}) = \sum\nolimits_{t=1}^{T} \mathbb{E}[c(\boldsymbol{x}_t)|\boldsymbol{\theta}]\,, \qquad p(\boldsymbol{x}_0) = \mathcal{N}\big(\boldsymbol{\mu}_0,\, \boldsymbol{\Sigma}_0\big)\,.$$

Instantaneous cost $c(\boldsymbol{x}_t)$, e.g., $\|\boldsymbol{x}_t - \boldsymbol{x}_{\text{target}}\|^2$

- Typical objective in optimal control and reinforcement learning (Bertsekas, 2005; Sutton & Barto, 1998)

Objective

Minimize expected long-term cost $J(\boldsymbol{\theta}) = \sum_t \mathbb{E}[c(\boldsymbol{x}_t)|\boldsymbol{\theta}]$

PILCO Framework: High-Level Steps

1. Probabilistic model for transition function f
   - ▶▶ System identification

Deisenroth et al. (IEEE-TPAMI, 2015): *Gaussian Processes for Data-Efficient Learning in Robotics and Control*

Objective

Minimize expected long-term cost $J(\boldsymbol{\theta}) = \sum_t \mathbb{E}[c(\boldsymbol{x}_t)|\boldsymbol{\theta}]$

PILCO Framework: High-Level Steps

1. Probabilistic model for transition function f
   ▶▶ System identification
2. Compute long-term predictions $p(\boldsymbol{x}_1|\boldsymbol{\theta}), \ldots, p(\boldsymbol{x}_T|\boldsymbol{\theta})$

Deisenroth et al. (IEEE-TPAMI, 2015): *Gaussian Processes for Data-Efficient Learning in Robotics and Control*

Objective

Minimize expected long-term cost $J(\boldsymbol{\theta}) = \sum_t \mathbb{E}[c(\boldsymbol{x}_t)|\boldsymbol{\theta}]$

PILCO Framework: High-Level Steps

1. Probabilistic model for transition function f
   ▶▶ System identification
2. Compute long-term predictions $p(\boldsymbol{x}_1|\boldsymbol{\theta}), \ldots, p(\boldsymbol{x}_T|\boldsymbol{\theta})$
3. Policy improvement

Deisenroth et al. (IEEE-TPAMI, 2015): *Gaussian Processes for Data-Efficient Learning in Robotics and Control*

Objective

Minimize expected long-term cost $J(\boldsymbol{\theta}) = \sum_t \mathbb{E}[c(\boldsymbol{x}_t)|\boldsymbol{\theta}]$

PILCO Framework: High-Level Steps

1. Probabilistic model for transition function f
   ▸▸ System identification
2. Compute long-term predictions $p(\boldsymbol{x}_1|\boldsymbol{\theta}), \ldots, p(\boldsymbol{x}_T|\boldsymbol{\theta})$
3. Policy improvement
4. Apply controller

Deisenroth et al. (IEEE-TPAMI, 2015): *Gaussian Processes for Data-Efficient Learning in Robotics and Control*

Objective

Minimize expected long-term cost $J(\boldsymbol{\theta}) = \sum_t \mathbb{E}[c(\boldsymbol{x}_t)|\boldsymbol{\theta}]$

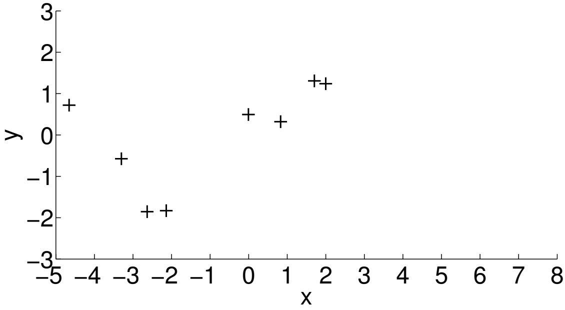
PILCO Framework: High-Level Steps

1. **Probabilistic model for transition function f**
   ▸▸ **System identification**
2. Compute long-term predictions $p(\boldsymbol{x}_1|\boldsymbol{\theta}), \ldots, p(\boldsymbol{x}_T|\boldsymbol{\theta})$
3. Policy improvement
4. Apply controller

Deisenroth et al. (IEEE-TPAMI, 2015): *Gaussian Processes for Data-Efficient Learning in Robotics and Control*

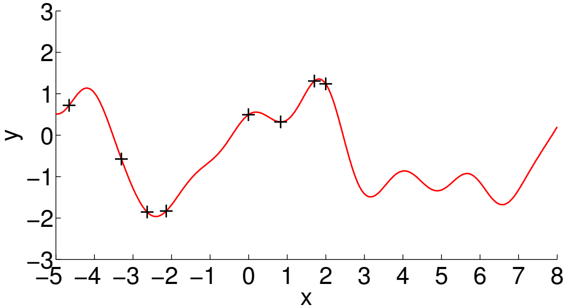
Model Learning (System Identification)

Model learning problem: Find a function $f : x \mapsto f(x) = y$



Observed function values

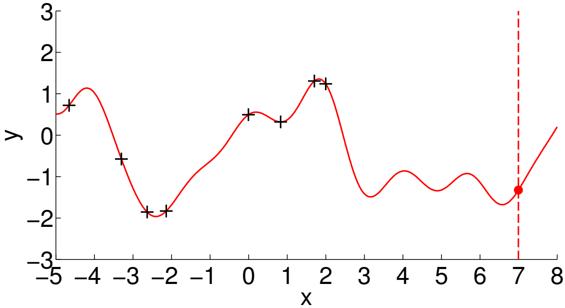
Model learning problem: Find a function $f : x \mapsto f(x) = y$



Plausible model

Model Learning (System Identification)

Model learning problem: Find a function $f : x \mapsto f(x) = y$

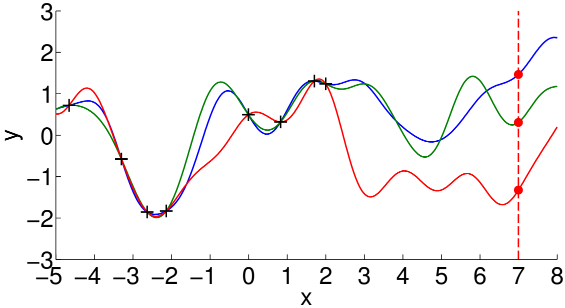


Plausible model

Predictions? Decision Making?

Model Learning (System Identification)

Model learning problem: Find a function $f : x \mapsto f(x) = y$

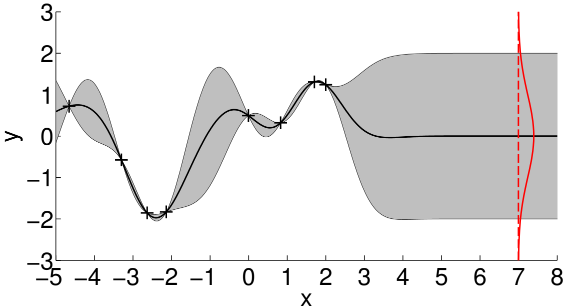


More plausible models

Predictions? Decision Making? Model Errors!

Model Learning (System Identification)

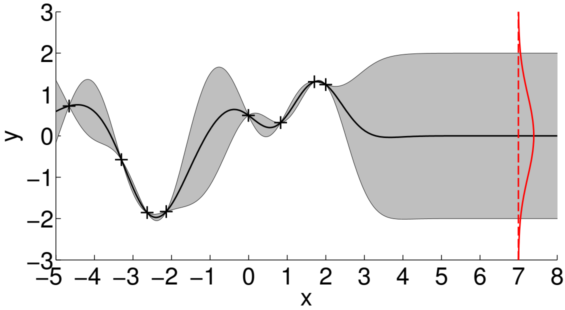
Model learning problem: Find a function $f : x \mapsto f(x) = y$



Distribution over plausible functions

Model Learning (System Identification)

Model learning problem: Find a function $f : x \mapsto f(x) = y$



Distribution over plausible functions

- Express uncertainty about the underlying function to be robust to model errors
- Gaussian process for model learning (Rasmussen & Williams, 2006)

Fast Reinforcement Learning

Objective

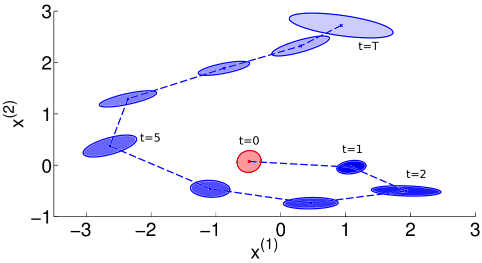
Minimize expected long-term cost $J(\boldsymbol{\theta}) = \sum_t \mathbb{E}[c(\boldsymbol{x}_t)|\boldsymbol{\theta}]$

PILCO Framework: High-Level Steps

1. Probabilistic model for transition function f
   ▶▶ System identification
2. **Compute long-term predictions** $p(\boldsymbol{x}_1|\boldsymbol{\theta}), \dots, p(\boldsymbol{x}_T|\boldsymbol{\theta})$
3. Policy improvement
4. Apply controller

Deisenroth et al. (IEEE-TPAMI, 2015): *Gaussian Processes for Data-Efficient Learning in Robotics and Control*

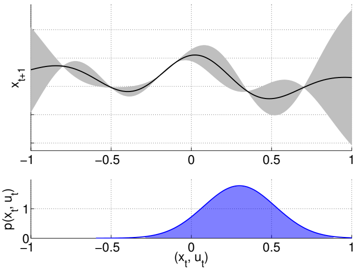
Long-Term Predictions



- Iteratively compute $p(\boldsymbol{x}_1|\boldsymbol{\theta}), \ldots, p(\boldsymbol{x}_T|\boldsymbol{\theta})$

Deisenroth et al. (IEEE-TPAMI, 2015): *Gaussian Processes for Data-Efficient Learning in Robotics and Control*

Long-Term Predictions

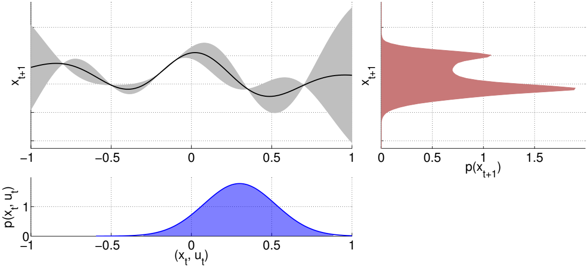


- Iteratively compute $p(\boldsymbol{x}_1|\boldsymbol{\theta}), \ldots, p(\boldsymbol{x}_T|\boldsymbol{\theta})$

$$\underbrace{p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{u}_t)}_{\text{GP prediction}} \underbrace{p(\boldsymbol{x}_t, \boldsymbol{u}_t|\boldsymbol{\theta})}_{\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})}$$

Deisenroth et al. (IEEE-TPAMI, 2015): *Gaussian Processes for Data-Efficient Learning in Robotics and Control*

Long-Term Predictions

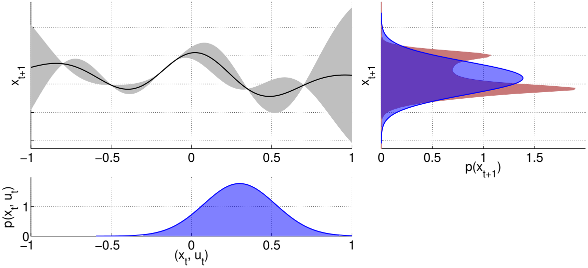


- Iteratively compute $p(\boldsymbol{x}_1|\boldsymbol{\theta}), \ldots, p(\boldsymbol{x}_T|\boldsymbol{\theta})$

$$\boxed{p(\boldsymbol{x}_{t+1}|\boldsymbol{\theta})} = \iiint \underbrace{\boxed{p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{u}_t)}}_{\text{GP prediction}} \underbrace{\boxed{p(\boldsymbol{x}_t, \boldsymbol{u}_t|\boldsymbol{\theta})}}_{\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})} df\, d\boldsymbol{x}_t\, d\boldsymbol{u}_t$$

Deisenroth et al. (IEEE-TPAMI, 2015): *Gaussian Processes for Data-Efficient Learning in Robotics and Control*

Long-Term Predictions



- Iteratively compute $p(\boldsymbol{x}_1|\boldsymbol{\theta}), \ldots, p(\boldsymbol{x}_T|\boldsymbol{\theta})$

$$\boxed{p(\boldsymbol{x}_{t+1}|\boldsymbol{\theta})} = \iiint \underbrace{\boxed{p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{u}_t)}}_{\text{GP prediction}} \underbrace{\boxed{p(\boldsymbol{x}_t, \boldsymbol{u}_t|\boldsymbol{\theta})}}_{\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})} df \, d\boldsymbol{x}_t \, d\boldsymbol{u}_t$$

▶▶ GP moment matching
(Girard et al., 2002; Quiñonero-Candela et al., 2003)

Deisenroth et al. (IEEE-TPAMI, 2015): *Gaussian Processes for Data-Efficient Learning in Robotics and Control*

Objective

Minimize expected long-term cost $J(\boldsymbol{\theta}) = \sum_t \mathbb{E}[c(\boldsymbol{x}_t)|\boldsymbol{\theta}]$

PILCO Framework: High-Level Steps

1. Probabilistic model for transition function f
   ⏩ System identification
2. Compute long-term predictions $p(\boldsymbol{x}_1|\boldsymbol{\theta}), \ldots, p(\boldsymbol{x}_T|\boldsymbol{\theta})$
3. **Policy improvement**
   - Compute expected long-term cost $J(\boldsymbol{\theta})$
   - Find parameters $\boldsymbol{\theta}$ that minimize $J(\boldsymbol{\theta})$
4. Apply controller

Deisenroth et al. (IEEE-TPAMI, 2015): *Gaussian Processes for Data-Efficient Learning in Robotics and Control*

Policy Improvement

Objective

Minimize expected long-term cost $J(\boldsymbol{\theta}) = \sum_t \mathbb{E}[c(\boldsymbol{x}_t)|\boldsymbol{\theta}]$

- (Gaussian) state distributions $p(\boldsymbol{x}_1|\boldsymbol{\theta}), \ldots, p(\boldsymbol{x}_T|\boldsymbol{\theta})$

Deisenroth et al. (IEEE-TPAMI, 2015): *Gaussian Processes for Data-Efficient Learning in Robotics and Control*

Policy Improvement

Objective

Minimize expected long-term cost $J(\boldsymbol{\theta}) = \sum_t \mathbb{E}[c(\boldsymbol{x}_t)|\boldsymbol{\theta}]$

- (Gaussian) state distributions $p(\boldsymbol{x}_1|\boldsymbol{\theta}), \dots, p(\boldsymbol{x}_T|\boldsymbol{\theta})$
- Compute

$$\mathbb{E}[c(\boldsymbol{x}_t)|\boldsymbol{\theta}] = \int c(\boldsymbol{x}_t)\mathcal{N}\big(\boldsymbol{x}_t \,|\, \boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t\big) d\boldsymbol{x}_t \,, \quad t = 1, \dots, T \,,$$

and sum them up to obtain $J(\boldsymbol{\theta})$

Deisenroth et al. (IEEE-TPAMI, 2015): *Gaussian Processes for Data-Efficient Learning in Robotics and Control*

Policy Improvement

Objective

Minimize expected long-term cost $J(\boldsymbol{\theta}) = \sum_t \mathbb{E}[c(\boldsymbol{x}_t)|\boldsymbol{\theta}]$

- (Gaussian) state distributions $p(\boldsymbol{x}_1|\boldsymbol{\theta}), \dots, p(\boldsymbol{x}_T|\boldsymbol{\theta})$
- Compute

$$\mathbb{E}[c(\boldsymbol{x}_t)|\boldsymbol{\theta}] = \int c(\boldsymbol{x}_t)\mathcal{N}\big(\boldsymbol{x}_t \,|\, \boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t\big) d\boldsymbol{x}_t \,, \quad t = 1, \dots, T \,,$$

  and sum them up to obtain $J(\boldsymbol{\theta})$
- Analytically compute gradient $\mathrm{d}J(\boldsymbol{\theta})/\mathrm{d}\boldsymbol{\theta}$
- Standard gradient-based optimizer (e.g., BFGS) to find $\boldsymbol{\theta}^*$

Deisenroth et al. (IEEE-TPAMI, 2015): *Gaussian Processes for Data-Efficient Learning in Robotics and Control*

Fast Reinforcement Learning

Objective

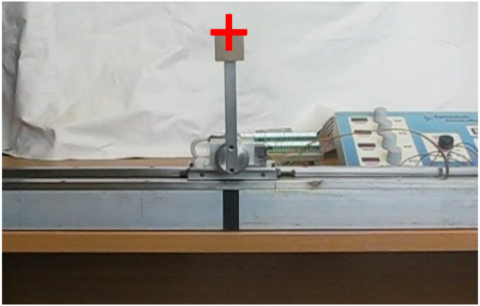
Minimize expected long-term cost $J(\boldsymbol{\theta}) = \sum_t \mathbb{E}[c(\boldsymbol{x}_t)|\boldsymbol{\theta}]$

PILCO Framework: High-Level Steps

1. Probabilistic model for transition function f
   ▶▶ System identification
2. Compute long-term predictions $p(\boldsymbol{x}_1|\boldsymbol{\theta}), \ldots, p(\boldsymbol{x}_T|\boldsymbol{\theta})$
3. Policy improvement
4. **Apply controller**

Deisenroth et al. (IEEE-TPAMI, 2015): *Gaussian Processes for Data-Efficient Learning in Robotics and Control*

Standard Benchmark: Cart-Pole Swing-up

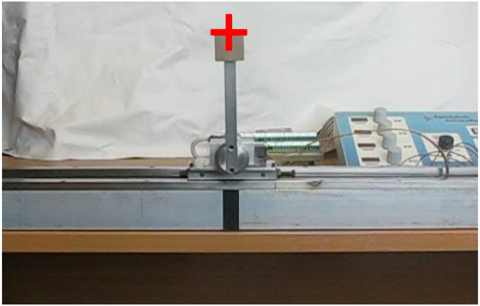


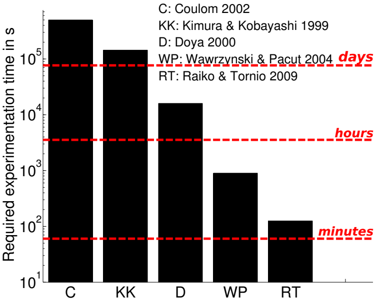
- Swing up and balance a freely swinging pendulum on a cart
- No knowledge about nonlinear dynamics ⏩ Learn from scratch
- Cost function $c(\boldsymbol{x}) = 1 - \exp(-\|\boldsymbol{x} - \boldsymbol{x}_{\text{target}}\|^2)$

- Code: https://github.com/ICL-SML/pilco-matlab

Deisenroth & Rasmussen (ICML, 2011): *PILCO: A Model-based and Data-efficient Approach to Policy Search*

Standard Benchmark: Cart-Pole Swing-up



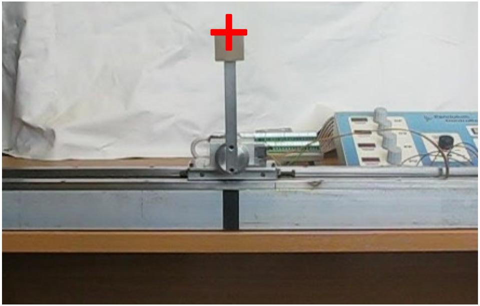


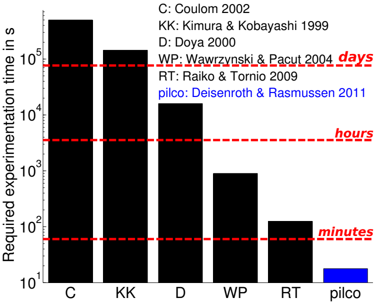
- Swing up and balance a freely swinging pendulum on a cart
- No knowledge about nonlinear dynamics ⏩ Learn from scratch
- Cost function $c(\boldsymbol{x}) = 1 - \exp(-\|\boldsymbol{x} - \boldsymbol{x}_{\text{target}}\|^2)$

- Code: https://github.com/ICL-SML/pilco-matlab

Deisenroth & Rasmussen (ICML, 2011): *PILCO: A Model-based and Data-efficient Approach to Policy Search*

Standard Benchmark: Cart-Pole Swing-up

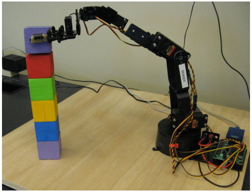




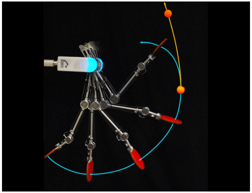
- Swing up and balance a freely swinging pendulum on a cart
- No knowledge about nonlinear dynamics ⏩ Learn from scratch
- Cost function $c(\boldsymbol{x}) = 1 - \exp(-\|\boldsymbol{x} - \boldsymbol{x}_{\text{target}}\|^2)$
- **Unprecedented learning speed** compared to state-of-the-art
- Code: `https://github.com/ICL-SML/pilco-matlab`

Deisenroth & Rasmussen (ICML, 2011): *PILCO: A Model-based and Data-efficient Approach to Policy Search*

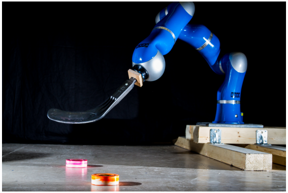
Wide Applicability



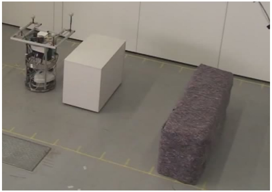
with D Fox



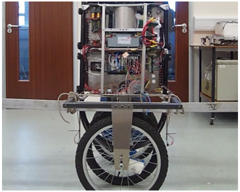
with P Englert, A Paraschos, J Peters



with A Kupcsik, J Peters, G Neumann



B Bischoff (Bosch), ESANN 2013



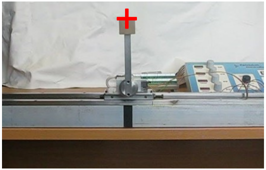
A McHutchon (U Cambridge)

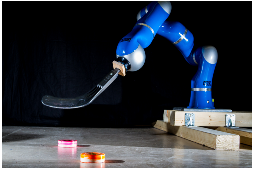
B Bischoff (Bosch), ECML 2013

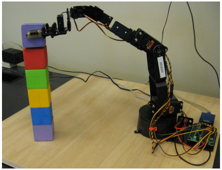
- Application to a wide range of robotic systems

Deisenroth et al. (RSS, 2011): *Learning to Control a Low-Cost Manipulator using Data-efficient Reinforcement Learning*
Englert et al. (ICRA, 2013): *Model-based Imitation Learning by Probabilistic Trajectory Matching*
Deisenroth et al. (ICRA, 2014): *Multi-Task Policy Search for Robotics*
Kupcsik et al. (AIJ, 2017): *Model-based Contextual Policy Search for Data-Efficient Generalization of Robot Skills*

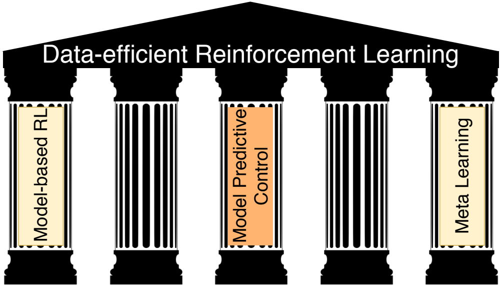
Summary (1)



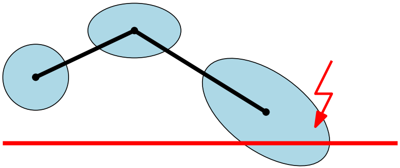




- In robotics, data-efficient learning is critical
- Probabilistic, model-based RL approach
  - Reduce model bias
  - Unprecedented learning speed
  - Wide applicability

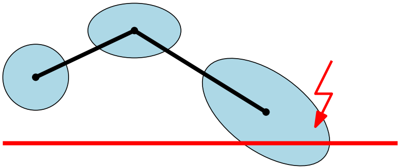

Data-efficient Reinforcement Learning
Model-based RL
Model Predictive Control
Meta Learning

Safe Exploration



- Deal with real-world safety constraints (states/controls)
- Use probabilistic model to predict whether state constraints are violated (e.g., Sui et al., 2015; Berkenkamp et al., 2017)
- Adjust policy if necessary (during policy learning)

Safe Exploration



- Deal with real-world safety constraints (states/controls)
- Use probabilistic model to predict whether state constraints are violated (e.g., Sui et al., 2015; Berkenkamp et al., 2017)
- Adjust policy if necessary (during policy learning)

▸▸ Safe exploration within an MPC-based RL setting

▸▸ Optimize control signals $\boldsymbol{u}_t$ directly (no policy parameters)

Approach

- Idea: Optimize control signals directly (instead of policy parameters)
- Few parameters to optimize ⏩ Low-dimensional search space
- Open-loop control
  ⏩ No chance of success (with minor model inaccuracies)

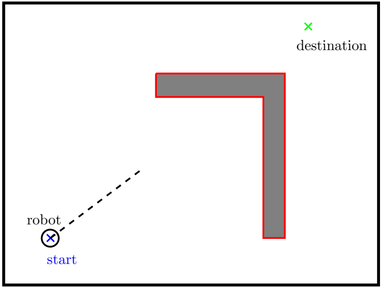
Approach

- Idea: Optimize control signals directly (instead of policy parameters)
- Few parameters to optimize ⏩ Low-dimensional search space
- Open-loop control
  ⏩ No chance of success (with minor model inaccuracies)
- Model Predictive Control (MPC) turns this into a closed-loop control approach

Approach

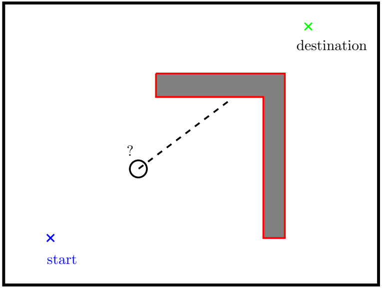
- Idea: Optimize control signals directly (instead of policy parameters)
- Few parameters to optimize ⏩ Low-dimensional search space
- Open-loop control
  ⏩ No chance of success (with minor model inaccuracies)
- Model Predictive Control (MPC) turns this into a closed-loop control approach
- Positive side-effect: Increase robustness to model errors (online approach) ⏩ Increase data efficiency

Model Predictive Control



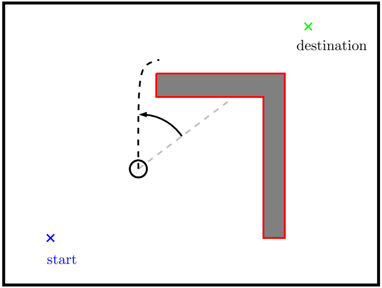
- Given a state $\boldsymbol{x}_t$, plan (open loop) over a short horizon of length H to get an open-loop control sequence $\boldsymbol{u}^*_{t+0}, \dots, \boldsymbol{u}^*_{t+H-1}$
- After transitioning into a new state $\boldsymbol{x}_{t+1}$, re-plan (as previously): Get $\boldsymbol{u}^*_{t+1+0}, \dots, \boldsymbol{u}^*_{t+1+H-1}$ ▶▶ closed-loop/feedback control

Model Predictive Control



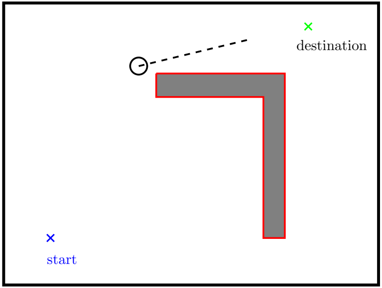
- Given a state $\boldsymbol{x}_t$, plan (open loop) over a short horizon of length H to get an open-loop control sequence $\boldsymbol{u}^*_{t+0}, \dots, \boldsymbol{u}^*_{t+H-1}$
- After transitioning into a new state $\boldsymbol{x}_{t+1}$, re-plan (as previously): Get $\boldsymbol{u}^*_{t+1+0}, \dots, \boldsymbol{u}^*_{t+1+H-1}$ ▶▶ closed-loop/feedback control

Model Predictive Control



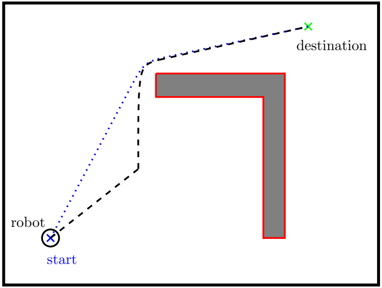
- Given a state x_t, plan (open loop) over a short horizon of length H to get an open-loop control sequence $\boldsymbol{u}^*_{t+0}, \ldots, \boldsymbol{u}^*_{t+H-1}$
- After transitioning into a new state x_{t+1}, re-plan (as previously): Get $\boldsymbol{u}^*_{t+1+0}, \ldots, \boldsymbol{u}^*_{t+1+H-1}$ ⏩ closed-loop/feedback control

Model Predictive Control



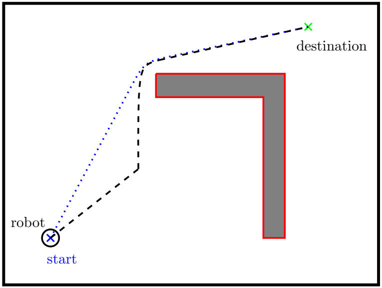
- Given a state $\boldsymbol{x}_t$, plan (open loop) over a short horizon of length H to get an open-loop control sequence $\boldsymbol{u}^*_{t+0}, \dots, \boldsymbol{u}^*_{t+H-1}$
- After transitioning into a new state $\boldsymbol{x}_{t+1}$, re-plan (as previously): Get $\boldsymbol{u}^*_{t+1+0}, \dots, \boldsymbol{u}^*_{t+1+H-1}$ ⏩ closed-loop/feedback control

Model Predictive Control



- Given a state x_t, plan (open loop) over a short horizon of length H to get an open-loop control sequence $\boldsymbol{u}^*_{t+0}, \ldots, \boldsymbol{u}^*_{t+H-1}$
- After transitioning into a new state x_{t+1}, re-plan (as previously): Get $\boldsymbol{u}^*_{t+1+0}, \ldots, \boldsymbol{u}^*_{t+1+H-1}$ ▶▶ closed-loop/feedback control

Model Predictive Control



- Given a state x_t, plan (open loop) over a short horizon of length H to get an open-loop control sequence $\boldsymbol{u}^*_{t+0}, \ldots, \boldsymbol{u}^*_{t+H-1}$
- After transitioning into a new state x_{t+1}, re-plan (as previously): Get $\boldsymbol{u}^*_{t+1+0}, \ldots, \boldsymbol{u}^*_{t+1+H-1}$ ▶▶ closed-loop/feedback control
- Use this within a trial-and-error model-based RL setting

Probabilistic MPC in RL

- Learn GP model for transition dynamics
- Repeat (while executing the policy):
  1. In current state $\boldsymbol{x}_t$, determine optimal control sequence $\boldsymbol{u}_0^*, \ldots, \boldsymbol{u}_{H-1}^*$
  2. Apply first control $\boldsymbol{u}_0^*$ in state $\boldsymbol{x}_t$
  3. Transition to next state $\boldsymbol{x}_{t+1}$
  4. Update GP transition model with observed transition $\big((\boldsymbol{x}_t, \boldsymbol{u}_0^*), \boldsymbol{x}_{t+1}\big)$

Kamthe & Deisenroth (AISTATS, 2018): *Data-Efficient Reinforcement Learning with Probabilistic Model Predictive Control*

Theoretical Results

- Uncertainty propagation is deterministic (GP moment matching)
  - Re-formulate system dynamics:

$$\begin{aligned} \boldsymbol{z}_{t+1} &= f_{MM}(\boldsymbol{z}_t, \boldsymbol{u}_t) \\ \boldsymbol{z}_t &= \{\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t\} \end{aligned}$$

  - Collects moments

Theoretical Results

- Uncertainty propagation is deterministic (GP moment matching)
  - Re-formulate system dynamics:

$$\begin{aligned} \boldsymbol{z}_{t+1} &= f_{MM}(\boldsymbol{z}_t, \boldsymbol{u}_t) \\ \boldsymbol{z}_t &= \{\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t\} \end{aligned}$$

  - Collects moments

- Deterministic system function that propagates moments
- Lipschitz continuity (under mild assumptions) implies that we can apply Pontryagin's minimum principle
  - Principled treatment of constraints on controls

Theoretical Results

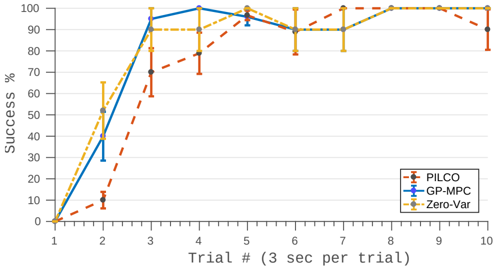
- Uncertainty propagation is deterministic (GP moment matching)
  ⏩ Re-formulate system dynamics:

$$\begin{aligned} \boldsymbol{z}_{t+1} &= f_{MM}(\boldsymbol{z}_t, \boldsymbol{u}_t) \\ \boldsymbol{z}_t &= \{\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t\} \end{aligned}$$

⏩ Collects moments

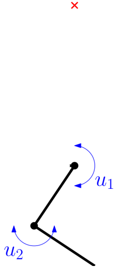
- Deterministic system function that propagates moments
- Lipschitz continuity (under mild assumptions) implies that we can apply Pontryagin's minimum principle
  ⏩ Principled treatment of constraints on controls
- Use predictive uncertainty to check violation of state constraints

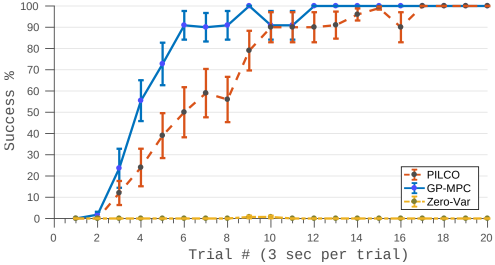
Learning Speed (Cart Pole)



- Zero-Var: Only use the mean of the GP, discard variances for long-term predictions
- MPC: Increased data efficiency (40% less experience required than PILCO)
  ▶▶ MPC more robust to model inaccuracies than a parametrized feedback controller

Kamthe & Deisenroth (AISTATS, 2018): *Data-Efficient Reinforcement Learning with Probabilistic Model Predictive Control*

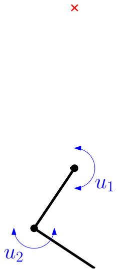


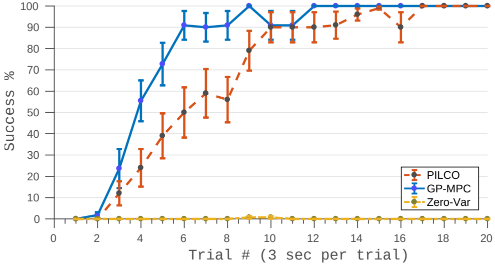


- GP-MPC maintains the same improvement in data efficiency
- Zero-Var fails:
  - Gets stuck in local optimum near start state
  - Insufficient exploration due to lack of uncertainty propagation

Kamthe & Deisenroth (AISTATS, 2018): *Data-Efficient Reinforcement Learning with Probabilistic Model Predictive Control*

Learning Speed (Double Pendulum)

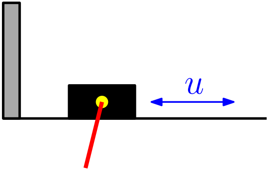


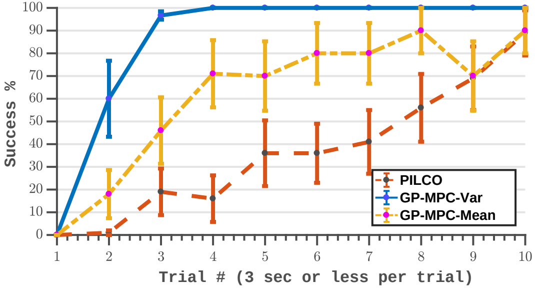


- GP-MPC maintains the same improvement in data efficiency
- Zero-Var fails:
  - Gets stuck in local optimum near start state
  - Insufficient exploration due to lack of uncertainty propagation
- Although MPC is fairly robust to model inaccuracies we cannot get away without uncertainty propagation

Kamthe & Deisenroth (AISTATS, 2018): *Data-Efficient Reinforcement Learning with Probabilistic Model Predictive Control*

Safety Constraints (Cart Pole)



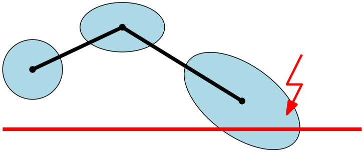


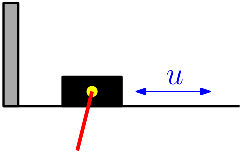
PILCO	16/100	constraint violations
GP-MPC-Mean	21/100	constraint violations
GP-MPC-Var	3/100	constraint violations

▸▸ Propagating model uncertainty important for safety

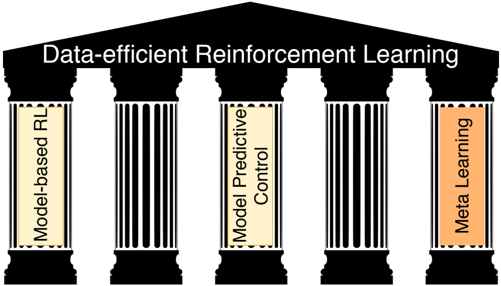
Kamthe & Deisenroth (AISTATS, 2018): *Data-Efficient Reinforcement Learning with Probabilistic Model Predictive Control*

Summary (2)





- Probabilistic prediction models for safe exploration
- Uncertainty propagation can be used to reduce violation of safety constraints
- MPC framework increases robustness to model errors
  ▶▶ Increased data efficiency


Data-efficient Reinforcement Learning
Model-based RL
Model Predictive Control
Meta Learning

Meta Learning

Meta Learning

Generalize knowledge from known tasks to new (related) tasks

Meta Learning

Generalize knowledge from known tasks to new (related) tasks

- Different robot configurations (link lengths, weights, ...)
- Re-use experience gathered so far generalize learning to new dynamics that are similar
  ▶▶ Accelerated learning

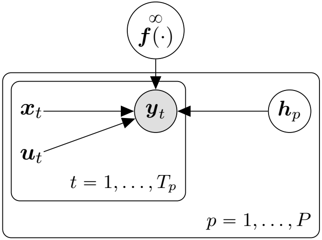
Approach

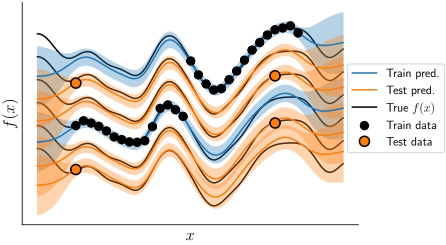
- Separate global and task-specific properties
- Shared global parameters describe general dynamics
- Describe task-specific (local) configurations with latent variable

Approach

- Separate global and task-specific properties
- Shared global parameters describe general dynamics
- Describe task-specific (local) configurations with latent variable
- Online variational inference of (unseen) configurations
- Few-shot model-based RL

Meta Model Learning with Latent Variables

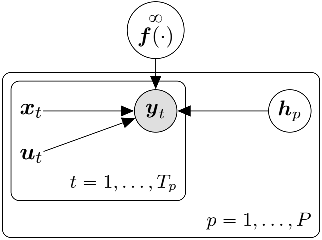


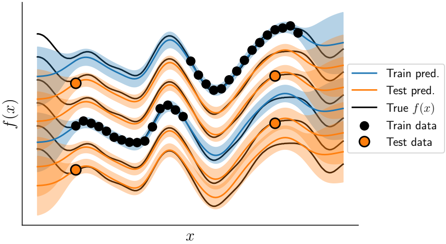


$$y_t = f(x_t, u_t, h_p)$$

Sæmundsson et al. (UAI, 2018): *Meta Reinforcement Learning with Latent Variable Gaussian Processes*

Meta Model Learning with Latent Variables



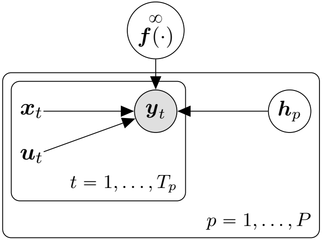


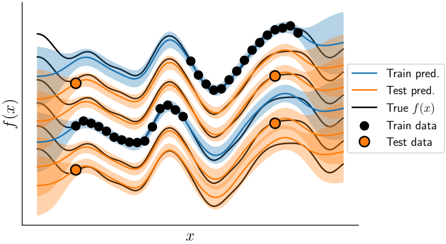
$$\boldsymbol{y}_t = \boldsymbol{f}(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{h}_p)$$

- GP captures global properties of the dynamics

Sæmundsson et al. (UAI, 2018): *Meta Reinforcement Learning with Latent Variable Gaussian Processes*

Meta Model Learning with Latent Variables



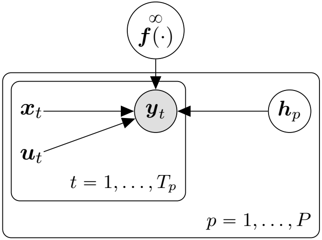


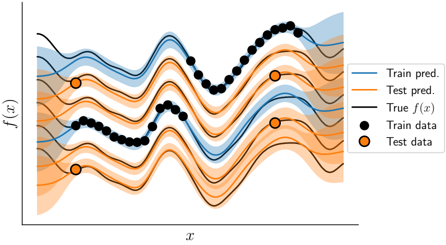
$$\boldsymbol{y}_t = \boldsymbol{f}(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{h}_p)$$

- GP captures global properties of the dynamics
- Latent variable $\boldsymbol{h}_p$ describes local configuration
  ▶▶ Variational inference to find a posterior on latent configuration

Sæmundsson et al. (UAI, 2018): *Meta Reinforcement Learning with Latent Variable Gaussian Processes*

Meta Model Learning with Latent Variables



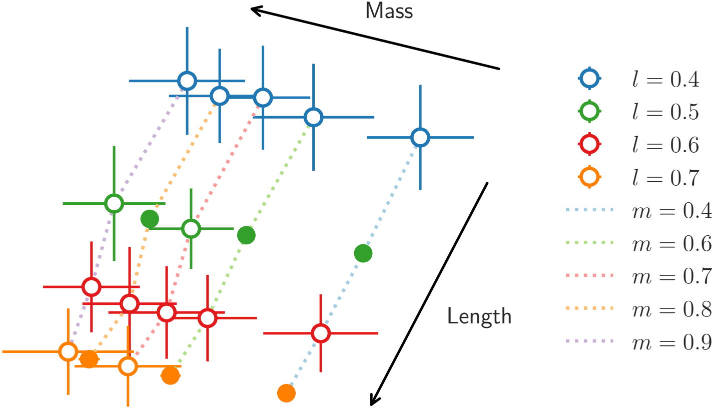


$$\boldsymbol{y}_t = \boldsymbol{f}(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{h}_p)$$

- GP captures global properties of the dynamics
- Latent variable $\boldsymbol{h}_p$ describes local configuration
  ▶▶ Variational inference to find a posterior on latent configuration
- Fast online inference of new configurations (no model re-training required)

Sæmundsson et al. (UAI, 2018): *Meta Reinforcement Learning with Latent Variable Gaussian Processes*

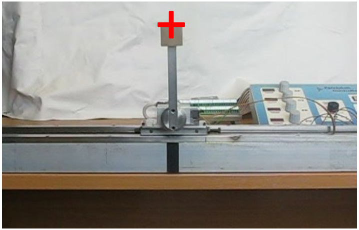
Latent Embeddings



- Latent variable $\boldsymbol{h}$ encodes length l and mass m of the cart pole
- 6 training tasks, 14 held-out test tasks

Sæmundsson et al. (UAI, 2018): *Meta Reinforcement Learning with Latent Variable Gaussian Processes*

Meta-RL (Cart Pole): Training



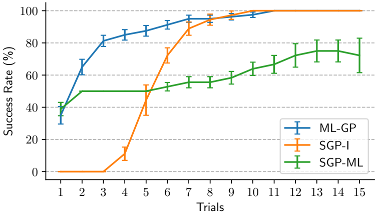
- Pre-trained on 6 training configurations until solved

Model	Training (s)	Description
Independent	16.1 ± 0.4	Independent GP-MPC
Aggregated	23.7 ± 1.4	Aggregated experience (no latents)
Meta learning	$\mathbf{15.1 \pm 0.5}$	Aggregated experience (with latents)

▸▸ Meta learning can help speeding up RL

Sæmundsson et al. (UAI, 2018): *Meta Reinforcement Learning with Latent Variable Gaussian Processes*

Meta-RL (Cart Pole): Few-Shot Generalization

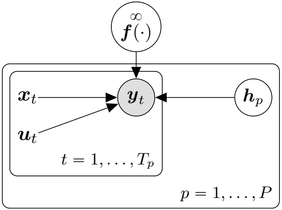


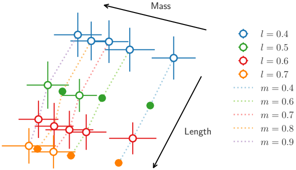
- Few-shot generalization on 4 unseen configurations
- Success: solve all 10 (6 training + 4 test) tasks
- Meta learning: blue
- Independent (GP-MPC): orange
- Aggregated experience model (no latents): green

▸▸ **Meta RL generalizes well to unseen tasks**

Sæmundsson et al. (UAI, 2018): *Meta Reinforcement Learning with Latent Variable Gaussian Processes*

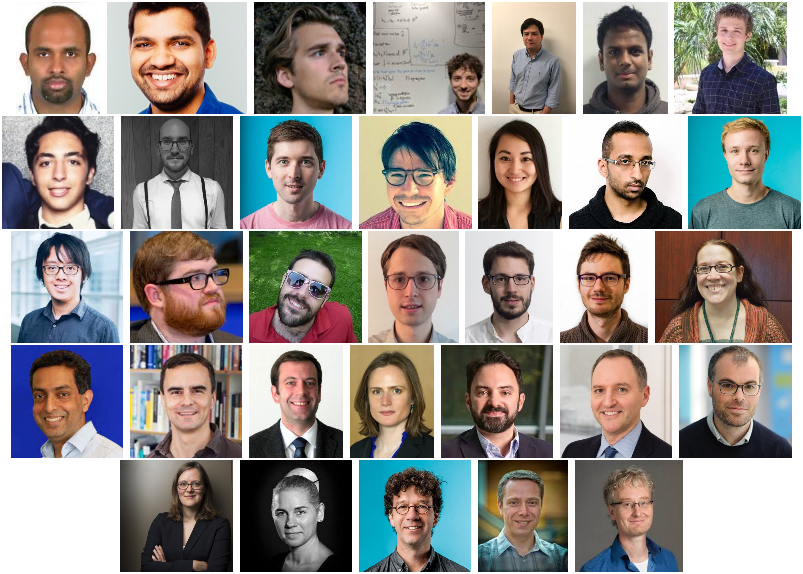
Summary (3)

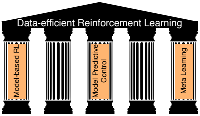


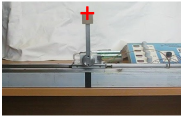


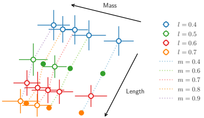
- Generalize knowledge from known situations to unseen ones
  ▶▶ **Few-shot learning**
- Latent variable can be used to infer task similarities
- Significant speed-up in model learning and model-based RL

Team and Collaborators



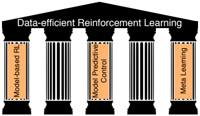


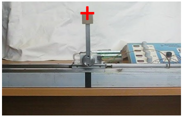


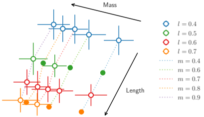


- **Data efficiency** is a practical challenge for autonomous robots
- Three pillars of data-efficient reinforcement learning for autonomous robots
  1. Model-based reinforcement learning with learned probabilistic models for fast learning from scratch
  2. Model predictive control with learned dynamics models accelerate learning and allow for safe exploration
  3. Meta learning using latent variables to generalize knowledge to new situations
- **Key to success:** Probabilistic modeling and Bayesian inference

Wrap-Up







- **Data efficiency** is a practical challenge for autonomous robots
- Three pillars of data-efficient reinforcement learning for autonomous robots
  1. Model-based reinforcement learning with learned probabilistic models for fast learning from scratch
  2. Model predictive control with learned dynamics models accelerate learning and allow for safe exploration
  3. Meta learning using latent variables to generalize knowledge to new situations
- **Key to success:** Probabilistic modeling and Bayesian inference

Thank you for your attention

ありがとうございました

References I

[1] F. Berkenkamp, M. Turchetta, A. P. Schoellig, and A. Krause. Safe Model-based Reinforcement Learning with Stability Guarantees. In *Advances in Neural Information Processing Systems*, 2017.

[2] D. P. Bertsekas. *Dynamic Programming and Optimal Control*, volume 1 of *Optimization and Computation Series*. Athena Scientific, Belmont, MA, USA, 3rd edition, 2005.

[3] D. P. Bertsekas. *Dynamic Programming and Optimal Control*, volume 2 of *Optimization and Computation Series*. Athena Scientific, Belmont, MA, USA, 3rd edition, 2007.

[4] B. Bischoff, D. Nguyen-Tuong, T. Koller, H. Markert, and A. Knoll. Learning Throttle Valve Control Using Policy Search. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases*, 2013.

[5] M. P. Deisenroth, P. Englert, J. Peters, and D. Fox. Multi-Task Policy Search for Robotics. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2014.

[6] M. P. Deisenroth, D. Fox, and C. E. Rasmussen. Gaussian Processes for Data-Efficient Learning in Robotics and Control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(2):408–423, 2015.

[7] M. P. Deisenroth and C. E. Rasmussen. PILCO: A Model-Based and Data-Efficient Approach to Policy Search. In *Proceedings of the International Conference on Machine Learning*, 2011.

[8] M. P. Deisenroth, C. E. Rasmussen, and D. Fox. Learning to Control a Low-Cost Manipulator using Data-Efficient Reinforcement Learning. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2011.

[9] P. Englert, A. Paraschos, J. Peters, and M. P. Deisenroth. Model-based Imitation Learning by Probabilistic Trajectory Matching. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2013.

[10] P. Englert, A. Paraschos, J. Peters, and M. P. Deisenroth. Probabilistic Model-based Imitation Learning. *Adaptive Behavior*, 21:388–403, 2013.

[11] A. Girard, C. E. Rasmussen, and R. Murray-Smith. Gaussian Process Priors with Uncertain Inputs: Multiple-Step Ahead Prediction. Technical Report TR-2002-119, University of Glasgow, 2002.

[12] S. Kamthe and M. P. Deisenroth. Data-Efficient Reinforcement Learning with Probabilistic Model Predictive Control. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2018.

References II

[13] A. Kupcsik, M. P. Deisenroth, J. Peters, L. A. Poha, P. Vadakkepata, and G. Neumann. Model-based Contextual Policy Search for Data-Efficient Generalization of Robot Skills. *Artificial Intelligence*, 2017.

[14] T. X. Nghiem and C. N. Jones. Data-driven Demand Response Modeling and Control of Buildings with Gaussian Processes. In *Proceedings of the American Control Conference*, 2017.

[15] J. Quiñonero-Candela, A. Girard, J. Larsen, and C. E. Rasmussen. Propagation of Uncertainty in Bayesian Kernel Models—Application to Multiple-Step Ahead Forecasting. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 2, pages 701–704, Apr. 2003.

[16] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning. The MIT Press, Cambridge, MA, USA, 2006.

[17] S. Sæmundsson, K. Hofmann, and M. P. Deisenroth. Meta Reinforcement Learning with Latent Variable Gaussian Processes. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2018.

[18] Y. Sui, A. Gotovos, J. W. Burdick, and A. Krause. Safe Exploration for Optimization with Gaussian Processes. In *Proceedings of the International Conference on Machine Learning*, 2015.

[19] M. K. Titsias. Variational Learning of Inducing Variables in Sparse Gaussian Processes. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2009.

Controller Parametrization

- Controller:

$$\tilde{\pi}(\boldsymbol{x}, \boldsymbol{\theta}) = \sum_{k=1}^{K} w_k \exp\big(-\tfrac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_k)^\top \boldsymbol{\Lambda}(\boldsymbol{x} - \boldsymbol{\mu}_k)\big)$$

$$u = \pi(\boldsymbol{x}, \boldsymbol{\theta}) = u_{\max}\sigma(\tilde{\pi}(\boldsymbol{x}, \boldsymbol{\theta})) \in [-u_{\max}, u_{\max}]\,,$$

where σ is a squashing function.

Controller Parametrization

- Controller:

$$\tilde{\pi}(\boldsymbol{x}, \boldsymbol{\theta}) = \sum_{k=1}^{K} w_k \exp\left(-\tfrac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_k)^\top \boldsymbol{\Lambda}(\boldsymbol{x}-\boldsymbol{\mu}_k)\right)$$

$$u = \pi(\boldsymbol{x}, \boldsymbol{\theta}) = u_{\max}\sigma(\tilde{\pi}(\boldsymbol{x}, \boldsymbol{\theta})) \in [-u_{\max}, u_{\max}]\,,$$

  where σ is a squashing function.

- Parameters:

$$\boldsymbol{\theta} := \{w_k, \boldsymbol{\mu}_k, \boldsymbol{\Lambda}\}$$

Controller Parametrization

- Controller:

$$\tilde{\pi}(\boldsymbol{x}, \boldsymbol{\theta}) = \sum_{k=1}^{K} w_k \exp\big(-\tfrac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_k)^\top \boldsymbol{\Lambda}(\boldsymbol{x}-\boldsymbol{\mu}_k)\big)$$
$$u = \pi(\boldsymbol{x}, \boldsymbol{\theta}) = u_{\max}\sigma(\tilde{\pi}(\boldsymbol{x}, \boldsymbol{\theta})) \in [-u_{\max}, u_{\max}],$$

where σ is a squashing function.

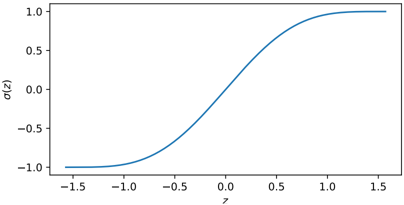
- Parameters:

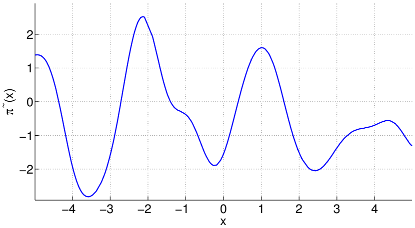
$$\boldsymbol{\theta} := \{w_k, \boldsymbol{\mu}_k, \boldsymbol{\Lambda}\}$$

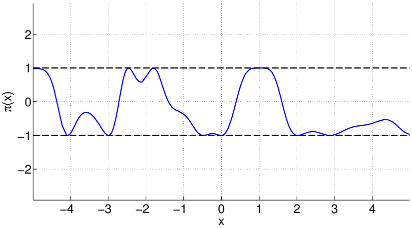
- Squashing function:

$$\sigma(z) = \frac{9}{8}\sin(z) + \frac{1}{8}\sin(3z)$$

Squashing Function

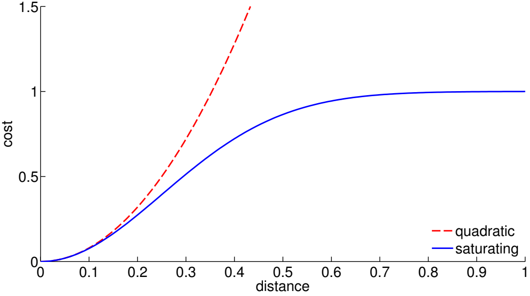






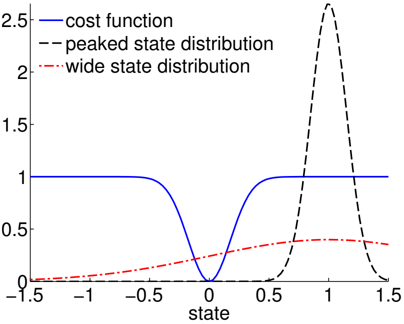
Cost Functions

- Quadratic cost $c(\boldsymbol{x}) = (\boldsymbol{x} - \boldsymbol{x}_{\text{target}})^\top \boldsymbol{W}(\boldsymbol{x} - \boldsymbol{x}_{\text{target}})$
- Saturating cost $c(\boldsymbol{x}) = 1 - \exp\big(-(\boldsymbol{x} - \boldsymbol{x}_{\text{target}})^\top \boldsymbol{W}(\boldsymbol{x} - \boldsymbol{x}_{\text{target}})\big)$



- Quadratic cost pays a lot of attention to states "far away"
  - Bad idea for exploration

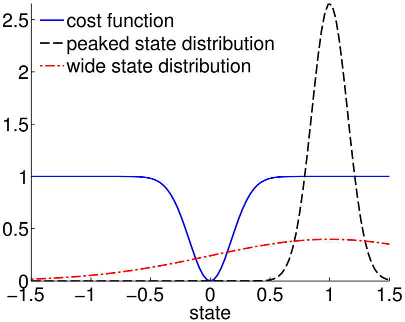
Task: Minimize $\mathbb{E}[c(\boldsymbol{x}_t)]$

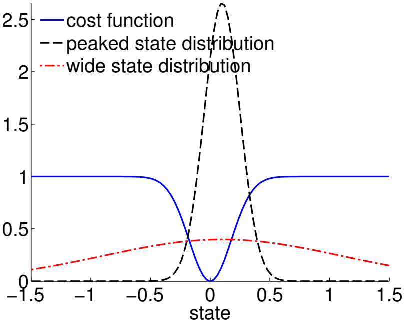


- In the early stages of learning, state predictions are expected to be far away from the target

Natural Exploration with the Saturating Cost

Task: Minimize $\mathbb{E}[c(\boldsymbol{x}_t)]$

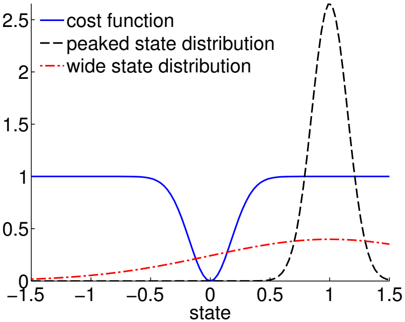


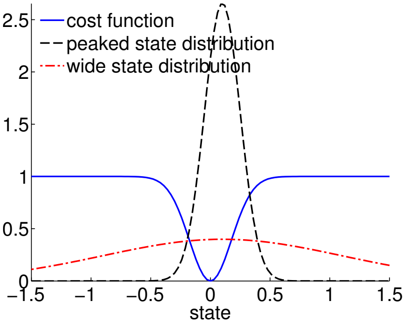


- In the early stages of learning, state predictions are expected to be far away from the target ▶▶ Exploration favored

Natural Exploration with the Saturating Cost

Task: Minimize $\mathbb{E}[c(\boldsymbol{x}_t)]$

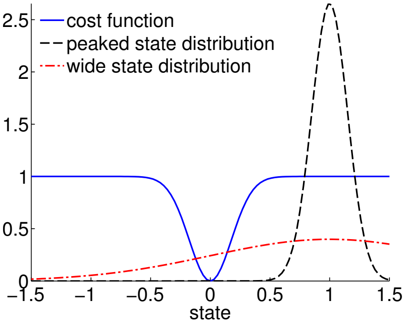


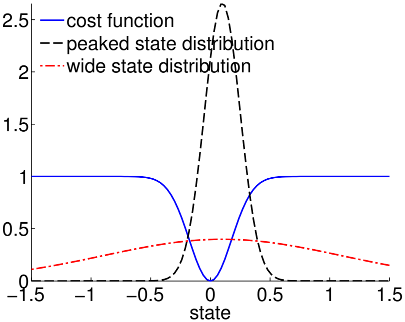


- In the early stages of learning, state predictions are expected to be far away from the target ▶▶ Exploration favored
- In the final stages of learning, state predictions are expected to be close to the target

Natural Exploration with the Saturating Cost

Task: Minimize $\mathbb{E}[c(\boldsymbol{x}_t)]$

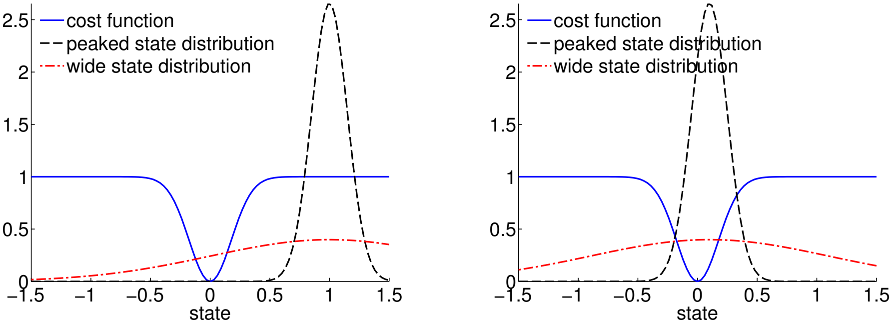




- In the early stages of learning, state predictions are expected to be far away from the target ▸▸ Exploration favored
- In the final stages of learning, state predictions are expected to be close to the target ▸▸ Exploitation favored

Natural Exploration with the Saturating Cost

Task: Minimize $\mathbb{E}[c(\boldsymbol{x}_t)]$



- In the early stages of learning, state predictions are expected to be far away from the target ⏩ Exploration favored
- In the final stages of learning, state predictions are expected to be close to the target ⏩ Exploitation favored

⏩ Bayesian treatment: Natural exploration/exploitation trade-off

GP Moment Matching: Some Details

$$f \sim GP(0, k)\,, \quad \text{Training data: } \boldsymbol{X}, \boldsymbol{y}$$
$$\boldsymbol{x}_* \sim \mathcal{N}\big(\boldsymbol{\mu},\, \boldsymbol{\Sigma}\big)$$

- Compute $\mathbb{E}[f(\boldsymbol{x}_*)]$

GP Moment Matching: Some Details

$$f \sim GP(0, k)\,, \quad \text{Training data: } \boldsymbol{X}, \boldsymbol{y}$$

$$\boldsymbol{x}_* \sim \mathcal{N}\big(\boldsymbol{\mu}, \boldsymbol{\Sigma}\big)$$

- Compute $\mathbb{E}[f(\boldsymbol{x}_*)]$

$$\mathbb{E}_{f,\boldsymbol{x}_*}[f(\boldsymbol{x}_*)] = \mathbb{E}_{\boldsymbol{x}}\big[\mathbb{E}_f[f(\boldsymbol{x}_*)|\boldsymbol{x}_*]\big] = \mathbb{E}_{\boldsymbol{x}_*}[m_f(\boldsymbol{x}_*)]$$

GP Moment Matching: Some Details

$$f \sim GP(0, k)\,, \quad \text{Training data: } \boldsymbol{X}, \boldsymbol{y}$$
$$\boldsymbol{x}_* \sim \mathcal{N}\big(\boldsymbol{\mu}, \boldsymbol{\Sigma}\big)$$

- Compute $\mathbb{E}[f(\boldsymbol{x}_*)]$

$$\mathbb{E}_{f,\boldsymbol{x}_*}[f(\boldsymbol{x}_*)] = \mathbb{E}_{\boldsymbol{x}}\big[\mathbb{E}_f[f(\boldsymbol{x}_*)|\boldsymbol{x}_*]\big] = \mathbb{E}_{\boldsymbol{x}_*}\big[m_f(\boldsymbol{x}_*)\big]$$
$$= \mathbb{E}_{\boldsymbol{x}_*}\big[k(\boldsymbol{x}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2\boldsymbol{I})^{-1}\boldsymbol{y}\big]$$

GP Moment Matching: Some Details

$$f \sim GP(0, k)\,, \quad \text{Training data: } \boldsymbol{X}, \boldsymbol{y}$$
$$\boldsymbol{x}_* \sim \mathcal{N}\big(\boldsymbol{\mu},\, \boldsymbol{\Sigma}\big)$$

- Compute $\mathbb{E}[f(\boldsymbol{x}_*)]$

$$\mathbb{E}_{f,\boldsymbol{x}_*}[f(\boldsymbol{x}_*)] = \mathbb{E}_{\boldsymbol{x}}\big[\,\mathbb{E}_f[f(\boldsymbol{x}_*)|\boldsymbol{x}_*]\,\big] = \mathbb{E}_{\boldsymbol{x}_*}\big[\,m_f(\boldsymbol{x}_*)\,\big]$$
$$= \mathbb{E}_{\boldsymbol{x}_*}\big[\,k(\boldsymbol{x}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2\boldsymbol{I})^{-1}\boldsymbol{y}\,\big]$$
$$= \boldsymbol{\beta}^\top \int k(\boldsymbol{X}, \boldsymbol{x}_*)\mathcal{N}\big(\boldsymbol{x}_* \,|\, \boldsymbol{\mu},\, \boldsymbol{\Sigma}\big) d\boldsymbol{x}_*$$
$$\boldsymbol{\beta} := (\boldsymbol{K} + \sigma_n^2\boldsymbol{I})^{-1}\boldsymbol{y}$$

▶▶ independent of $\boldsymbol{x}_*$

GP Moment Matching: Some Details

$$f \sim GP(0, k)\,, \quad \text{Training data: } \boldsymbol{X}, \boldsymbol{y}$$
$$\boldsymbol{x}_* \sim \mathcal{N}\big(\boldsymbol{\mu},\, \boldsymbol{\Sigma}\big)$$

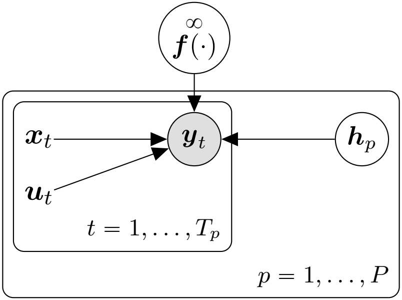
- Compute $\mathbb{E}[f(\boldsymbol{x}_*)]$

$$\begin{aligned}
\mathbb{E}_{f,\boldsymbol{x}_*}[f(\boldsymbol{x}_*)] &= \mathbb{E}_{\boldsymbol{x}}\big[\,\mathbb{E}_f[f(\boldsymbol{x}_*)|\boldsymbol{x}_*]\,\big] = \mathbb{E}_{\boldsymbol{x}_*}\big[\,m_f(\boldsymbol{x}_*)\,\big] \\
&= \mathbb{E}_{\boldsymbol{x}_*}\Big[\,k(\boldsymbol{x}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2\boldsymbol{I})^{-1}\boldsymbol{y}\,\Big] \\
&= \boldsymbol{\beta}^\top \int k(\boldsymbol{X}, \boldsymbol{x}_*)\mathcal{N}\big(\boldsymbol{x}_* \,|\, \boldsymbol{\mu},\, \boldsymbol{\Sigma}\big)d\boldsymbol{x}_*
\end{aligned}$$

$$\boldsymbol{\beta} := (\boldsymbol{K} + \sigma_n^2\boldsymbol{I})^{-1}\boldsymbol{y} \quad \blacktriangleright\!\!\blacktriangleright \text{ independent of } \boldsymbol{x}_*$$

- If k is a Gaussian (squared exponential) kernel, this integral can be solved analytically
- Variance of $f(\boldsymbol{x}_*)$ can be computed similarly

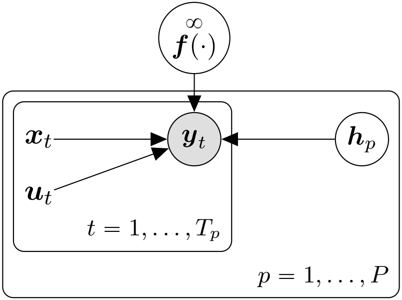
Meta Learning Model



$$\boldsymbol{f}(\cdot) \sim GP$$

$$p(\boldsymbol{H}) = \prod_p p(\boldsymbol{h}_p)\,, \quad p(\boldsymbol{h}_p) = \mathcal{N}\big(\boldsymbol{0},\, \boldsymbol{I}\big)$$

Meta Learning Model



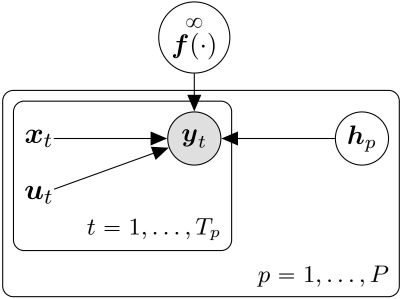
$$\boldsymbol{f}(\cdot) \sim GP$$

$$p(\boldsymbol{H}) = \prod_{p} p(\boldsymbol{h}_p)\,, \quad p(\boldsymbol{h}_p) = \mathcal{N}\big(\boldsymbol{0},\, \boldsymbol{I}\big)$$

$$p(\boldsymbol{Y}, \boldsymbol{H}, \boldsymbol{f}(\cdot)|\boldsymbol{X}, \boldsymbol{U}) = \prod\nolimits_{p=1}^{P} p(\boldsymbol{h}_p) \prod\nolimits_{t=1}^{T_p} p(\boldsymbol{y}_t|\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{h}_p, \boldsymbol{f}(\cdot))p(\boldsymbol{f}(\cdot))$$

$$\boldsymbol{y}_t = \boldsymbol{x}_{t+1} - \boldsymbol{x}_t$$

Variational Inference in Meta Learning Model



Mean-field variational family:

$$q(\boldsymbol{f}(\cdot), \boldsymbol{H}) = q(\boldsymbol{f}(\cdot))q(\boldsymbol{H})$$

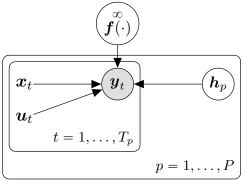
$$q(\boldsymbol{H}) = \prod\nolimits_{p=1}^{P} \mathcal{N}(\boldsymbol{h}_p | \boldsymbol{n}_p, \boldsymbol{T}_p)\,,$$

$$q(\boldsymbol{f}(\cdot)) = \int p(\boldsymbol{f}(\cdot)|\boldsymbol{f}_Z)q(\boldsymbol{f}_Z)d\boldsymbol{f}_Z$$

▶▶ SV-GP (Titsias, 2009)

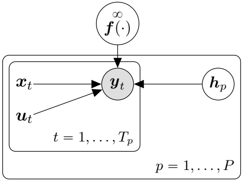
Evidence Lower Bound

$$ELBO = \mathbb{E}_{q(\boldsymbol{f}(\cdot),\boldsymbol{H})}\Big[\log \frac{p(\boldsymbol{Y},\boldsymbol{H},\boldsymbol{f}(\cdot)|\boldsymbol{X},\boldsymbol{U})}{q(\boldsymbol{f}(\cdot),\boldsymbol{H})}\Big]$$



Evidence Lower Bound

$$
\begin{aligned}
ELBO &= \mathbb{E}_{q(\boldsymbol{f}(\cdot),\boldsymbol{H})}\Big[\log \frac{p(\boldsymbol{Y},\boldsymbol{H},\boldsymbol{f}(\cdot)|\boldsymbol{X},\boldsymbol{U})}{q(\boldsymbol{f}(\cdot),\boldsymbol{H})}\Big] \\
&= \sum\nolimits_{p=1}^{P}\sum\nolimits_{t=1}^{T_p}\mathbb{E}_{q(\boldsymbol{f}_t|\boldsymbol{x}_t,\boldsymbol{u}_t,\boldsymbol{h}_p)q(\boldsymbol{h}_p)}\big[\log p(\boldsymbol{y}_t|\boldsymbol{f}_t)\big] \\
&\quad - \mathrm{KL}(q(\boldsymbol{H})||p(\boldsymbol{H})) - \mathrm{KL}(q(\boldsymbol{f}(\cdot))||p(\boldsymbol{f}(\cdot)))
\end{aligned}
$$



Evidence Lower Bound

$$
\begin{aligned}
ELBO &= \mathbb{E}_{q(\boldsymbol{f}(\cdot),\boldsymbol{H})}\Big[\log \frac{p(\boldsymbol{Y},\boldsymbol{H},\boldsymbol{f}(\cdot)|\boldsymbol{X},\boldsymbol{U})}{q(\boldsymbol{f}(\cdot),\boldsymbol{H})}\Big] \\
&= \sum_{p=1}^{P}\sum_{t=1}^{T_p} \mathbb{E}_{q(\boldsymbol{f}_t|\boldsymbol{x}_t,\boldsymbol{u}_t,\boldsymbol{h}_p)q(\boldsymbol{h}_p)}\big[\log p(\boldsymbol{y}_t|\boldsymbol{f}_t)\big] \\
&\quad - \mathrm{KL}(q(\boldsymbol{H})||p(\boldsymbol{H})) - \mathrm{KL}(q(\boldsymbol{f}(\cdot))||p(\boldsymbol{f}(\cdot))) \\
&= \sum_{p=1}^{P}\sum_{t=1}^{T_p} \overbrace{\mathbb{E}_{q(\boldsymbol{f}_t|\boldsymbol{x}_t,\boldsymbol{u}_t,\boldsymbol{h}_p)q(\boldsymbol{h}_p)}\big[\log p(\boldsymbol{y}_t|\boldsymbol{f}_t)\big]}^{\text{Monte Carlo estimate}} \\
&\quad \underbrace{-\mathrm{KL}(q(\boldsymbol{H})||p(\boldsymbol{H})) - \mathrm{KL}(q(\boldsymbol{F}_Z)||p(\boldsymbol{F}_Z))}_{\text{closed-form solution}}
\end{aligned}
$$