The Role of Uncertainty in Model-based Reinforcement Learning

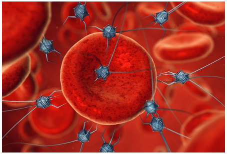
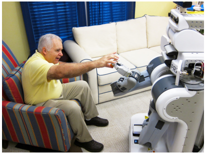
Marc Deisenroth
Centre for Artificial Intelligence
Department of Computer Science
University College London

@mpd37
m.deisenroth@ucl.ac.uk
https://deisenroth.cc

Workshop on Uncertainty Propagation in Composite Models, Munich

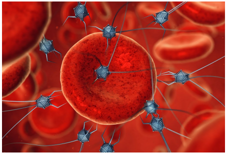
October 10, 2019

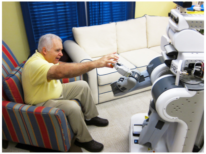
Autonomous Robots

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

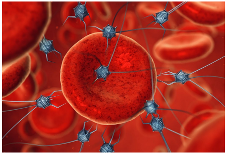
Autonomous Robots

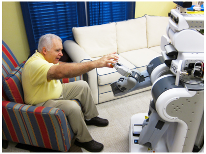





- Vision: Autonomous robots support humans in everyday activities ⏩ Fast learning and automatic adaptation
- Currently: Data-hungry learning or human guidance

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

Central Problem

Data-Efficient Reinforcement Learning

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

▸▸ **Model-based 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

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

Transition function

Control

Policy

Policy parameters

Objective (Controller Learning)

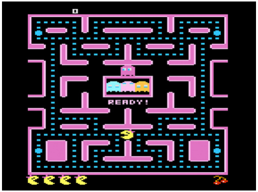
Find policy parameters $\boldsymbol{\theta}^*$ that minimize the expected long-term cost

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

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

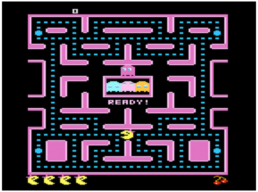
- Typical objective in optimal control and reinforcement learning (Bertsekas, 2005; Sutton & Barto, 1998)

Model-based RL



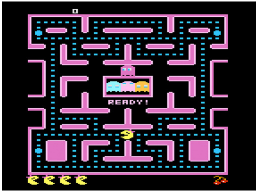
- Insight: If we had a realistic simulator of the world, we would not need to run experiments in the real world, but in the cheaper simulator (e.g., chess, Go, pacman)

Model-based RL

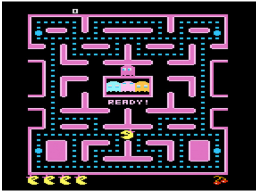


- Insight: If we had a realistic simulator of the world, we would not need to run experiments in the real world, but in the cheaper simulator (e.g., chess, Go, pacman)
- Idea: Build simulator based on observed trajectories

Model-based RL

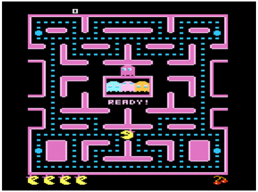


- Insight: If we had a realistic simulator of the world, we would not need to run experiments in the real world, but in the cheaper simulator (e.g., chess, Go, pacman)
- Idea: Build simulator based on observed trajectories
- Issue: Model errors can lead to catastrophic failures



- Insight: If we had a realistic simulator of the world, we would not need to run experiments in the real world, but in the cheaper simulator (e.g., chess, Go, pacman)
- Idea: Build simulator based on observed trajectories
- Issue: Model errors can lead to catastrophic failures
- Policy is learned only based on simulator data

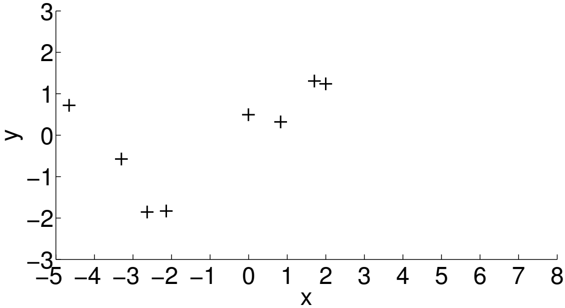
Model-based RL



- Insight: If we had a realistic simulator of the world, we would not need to run experiments in the real world, but in the cheaper simulator (e.g., chess, Go, pacman)
- Idea: Build simulator based on observed trajectories
- Issue: Model errors can lead to catastrophic failures
- Policy is learned only based on simulator data
- How can we build better models? ▶▶ **Probabilistic models**

Model Learning (System Identification)

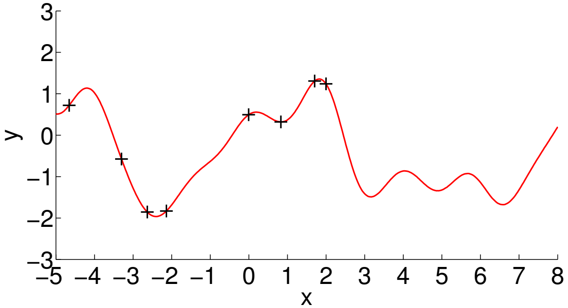
Model learning problem: Find a function $f : x \mapsto f(x) = y$



Observed function values

Model Learning (System Identification)

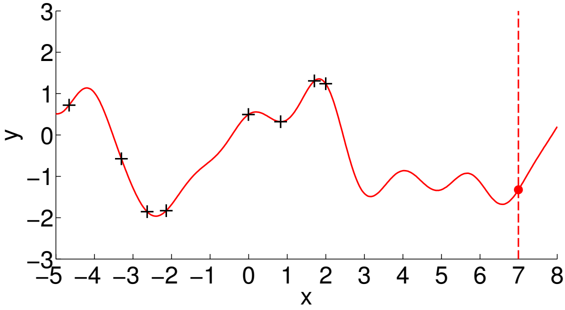
Model learning problem: Find a function $f : x \mapsto f(x) = y$



Plausible model

Model Learning (System Identification)

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

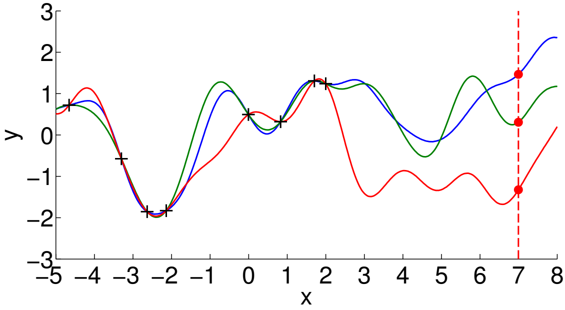


Plausible model

Predictions? Decision Making?

Model Learning (System Identification)

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

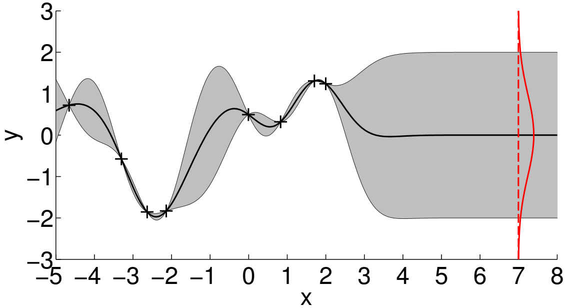


More plausible models

Predictions? Decision Making? Model Errors!

Model Learning (System Identification)

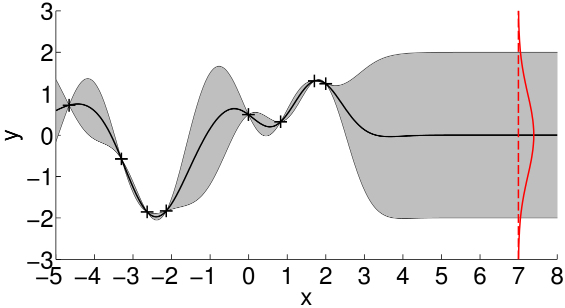
Model learning problem: Find a function $f : x \mapsto f(x) = y$



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 processes, Bayesian linear regression, ensembles for model 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

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 $p(\boldsymbol{x}_1|\boldsymbol{\theta}), \dots, p(\boldsymbol{x}_T|\boldsymbol{\theta})$
3. Policy improvement
4. Apply controller

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

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*

Objective

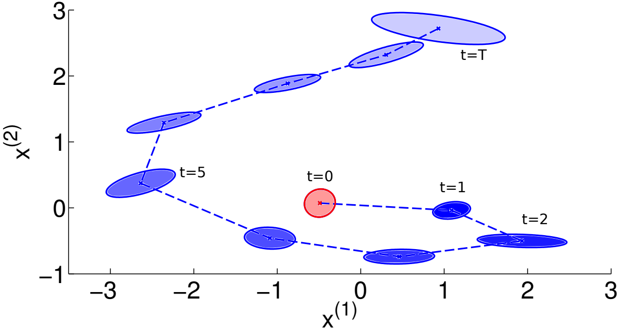
Minimize expected long-term cost $J(\boldsymbol{\theta}) = \sum_t \mathbb{E}[c(\boldsymbol{x}_t)|\boldsymbol{\theta}]$

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

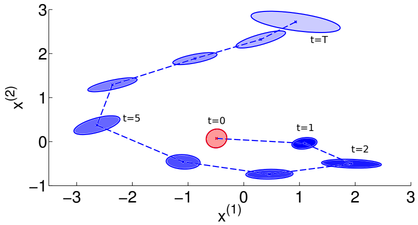
Deisenroth et al. (IEEE-TPAMI, 2015): *Gaussian Processes for Data-Efficient Learning in Robotics and Control*

Long-Term Predictions



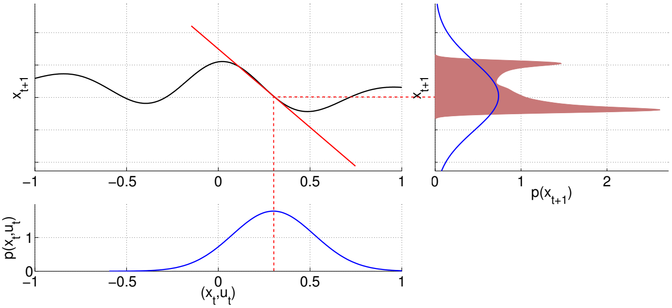
- Uncertainty propagation: Iteratively compute state distributions $p(\boldsymbol{x}_t)$, $t = 1, \ldots, T$.

Uncertainty Propagation



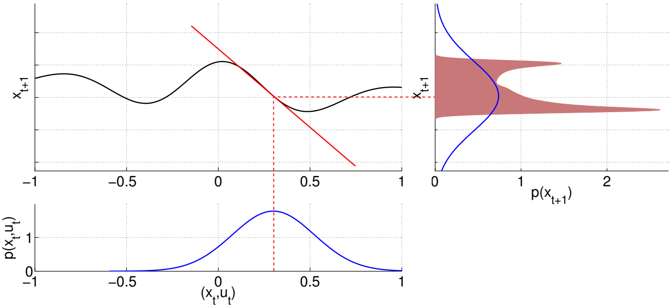
- Deterministic approximate inference (implicit local linearization)
  - Exact moment matching
  - Approximate moment matching (e.g., linearization, unscented transformation)
- Stochastic approximate inference
  - Trajectory sampling
    Important: Compute the GP posterior after every fantasized sample (augment dataset with fantasy data)

Linearization

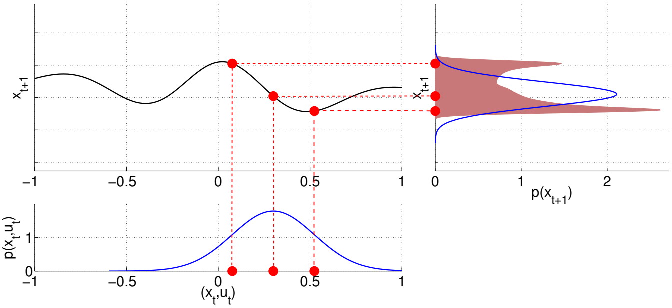


- Approximate nonlinear function at the mean of the distribution with a linear function (1st-order Taylor series)
- Push Gaussian through this linear function: Closed-form mean and covariance of predictive distribution

Linearization

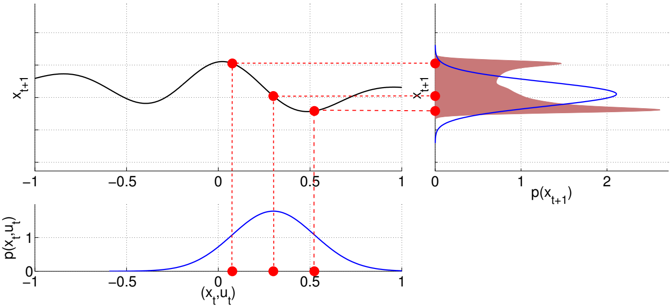


- Approximate nonlinear function at the mean of the distribution with a linear function (1st-order Taylor series)
- Push Gaussian through this linear function: Closed-form mean and covariance of predictive distribution
- Requires gradients of nonlinear function
- Can get the true moments catastrophically wrong



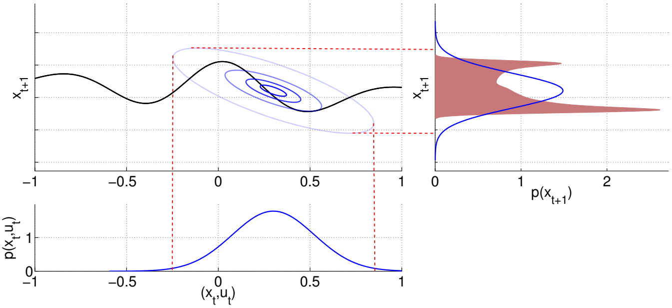
- Approximate distribution with a small number of sigma points
- Evaluate nonlinear function at those points
- Compute sample statistics (mean, covariance) of predictive distribution

Unscented Transformation

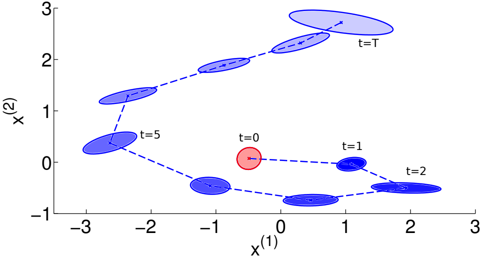


- Approximate distribution with a small number of sigma points
- Evaluate nonlinear function at those points
- Compute sample statistics (mean, covariance) of predictive distribution
- Can get the true moments catastrophically wrong

Moment Matching



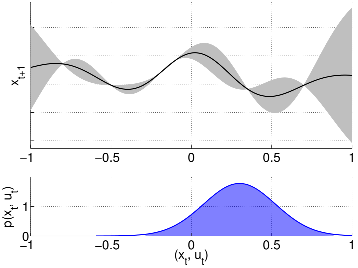
- Compute mean and covariance of predictive distribution
- Approximate predictive distribution with a Gaussian that possesses the correct moments
- Higher-order moments ignored
- Exact moments can only be computed in special cases



- 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*

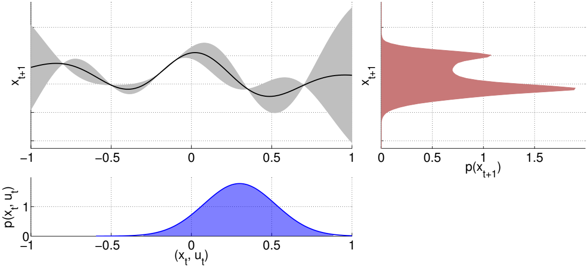
Long-Term Predictions with GPs



- 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*

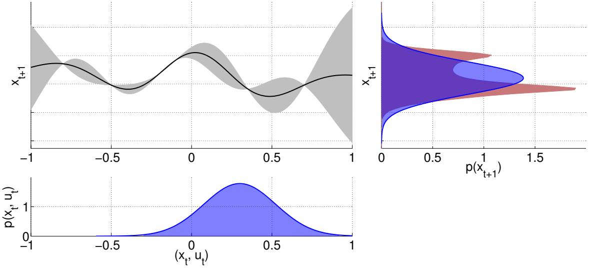


- Iteratively compute $p(\boldsymbol{x}_1|\boldsymbol{\theta}), \ldots, p(\boldsymbol{x}_T|\boldsymbol{\theta})$

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

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

Long-Term Predictions with GPs



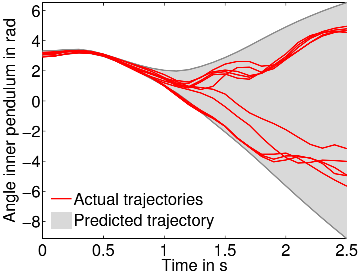
- Iteratively compute $p(\boldsymbol{x}_1|\boldsymbol{\theta}), \ldots, p(\boldsymbol{x}_T|\boldsymbol{\theta})$

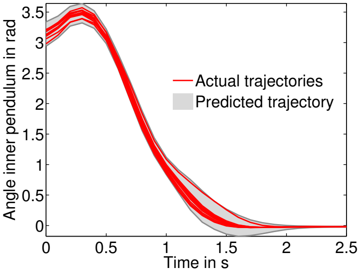
$$p(\boldsymbol{x}_{t+1}|\boldsymbol{\theta}) = \iiint \underbrace{p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{u}_t)}_{\text{GP prediction}} \underbrace{p(\boldsymbol{x}_t, \boldsymbol{u}_t|\boldsymbol{\theta})}_{\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})} df\, d\boldsymbol{x}_t\, d\boldsymbol{u}_t$$

▶▶ 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*

Moment Matching: Long-Term Predictions





- Left: Early stages of learning (model not confident)
- Right: More confident model

▶▶ Predictive error bars seem reasonable

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 $J(\boldsymbol{\theta}) = \sum_t \mathbb{E}[c(\boldsymbol{x}_t)|\boldsymbol{\theta}]$

- Know how to predict $p(\boldsymbol{x}_1|\boldsymbol{\theta}), \ldots, p(\boldsymbol{x}_T|\boldsymbol{\theta})$

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

Policy Improvement

Objective

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

- Know how to predict $p(\boldsymbol{x}_1|\boldsymbol{\theta}), \ldots, p(\boldsymbol{x}_T|\boldsymbol{\theta})$
- Compute

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

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 $J(\boldsymbol{\theta}) = \sum_t \mathbb{E}[c(\boldsymbol{x}_t)|\boldsymbol{\theta}]$

- Know how to predict $p(\boldsymbol{x}_1|\boldsymbol{\theta}), \dots, p(\boldsymbol{x}_T|\boldsymbol{\theta})$
- 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*

Objective

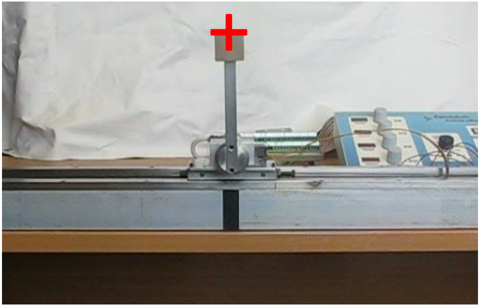
Minimize expected long-term cost $J(\boldsymbol{\theta}) = \sum_t \mathbb{E}[c(\boldsymbol{x}_t)|\boldsymbol{\theta}]$

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*

Standard Benchmark: Cart-Pole Swing-up

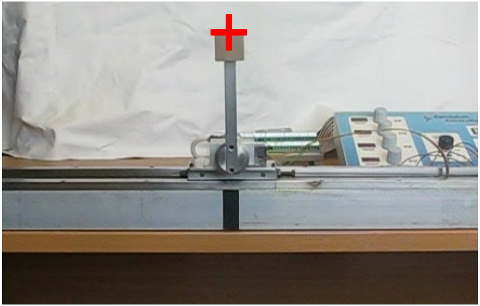


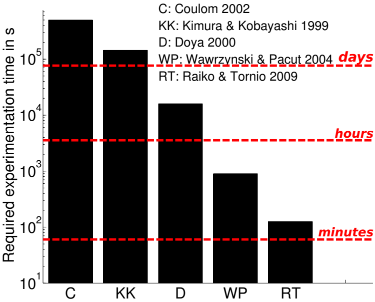
- Swing up and balance a freely swinging pendulum on a cart
- No knowledge about nonlinear dynamics ⏩ Learn from scratch
- Cost function $c(\boldsymbol{x}) = 1 - \exp(-\|\boldsymbol{x} - \boldsymbol{x}_{\text{target}}\|^2)$

- 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



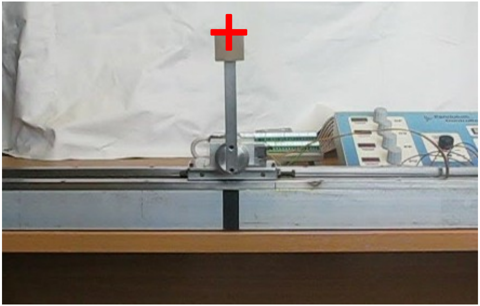


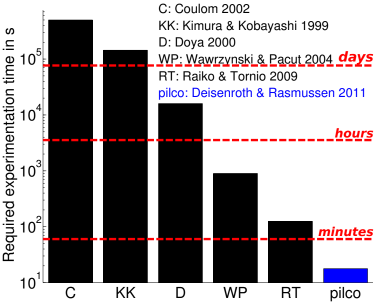
- Swing up and balance a freely swinging pendulum on a cart
- No knowledge about nonlinear dynamics ⏩ Learn from scratch
- Cost function $c(\boldsymbol{x}) = 1 - \exp(-\|\boldsymbol{x} - \boldsymbol{x}_{\text{target}}\|^2)$

- 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

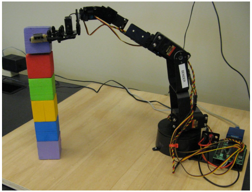




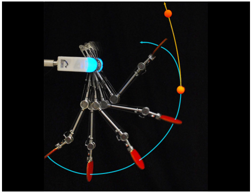
- Swing up and balance a freely swinging pendulum on a cart
- No knowledge about nonlinear dynamics ⏩ Learn from scratch
- Cost function $c(\boldsymbol{x}) = 1 - \exp(-\|\boldsymbol{x} - \boldsymbol{x}_{\text{target}}\|^2)$
- **Unprecedented learning speed** compared to state-of-the-art
- Code: `https://github.com/ICL-SML/pilco-matlab`

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

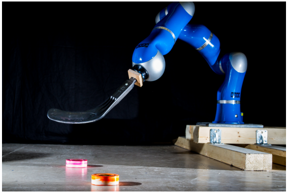
Wide Applicability



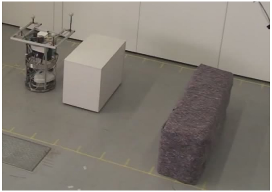
with D Fox



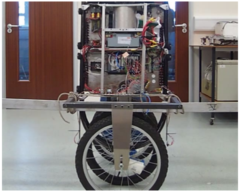
with P Englert, A Paraschos, J Peters



with A Kupcsik, J Peters, G Neumann



B Bischoff (Bosch), ESANN 2013



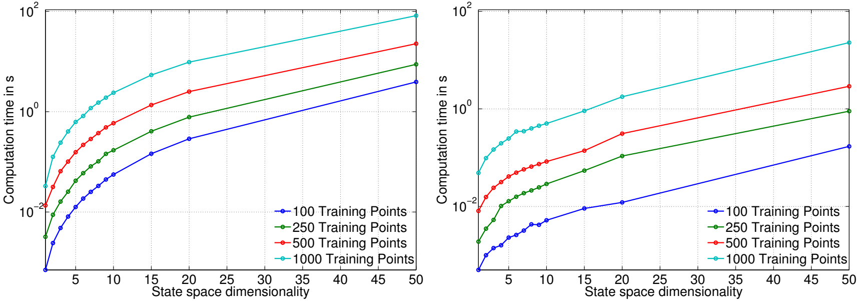
A McHutchon (U Cambridge)

- 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*

Some More Details

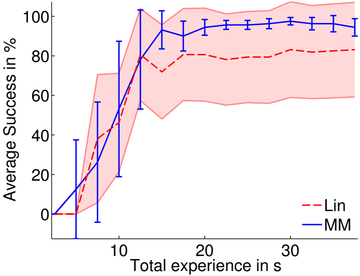
Computational Demand (Inference)



- Graphs: Time required to compute gradient (one time step; 2013 laptop; single CPU)
- Left: Moment matching
- Right: Linearization

▶▶ Linearization is significantly faster than moment matching

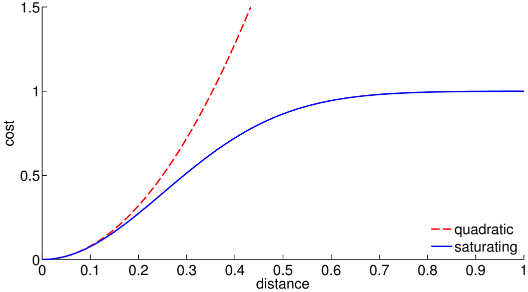
Linearization vs Moment Matching



- These are **NOT** learning curves on training data.
  - Success evaluated on unseen test data
- Linearization is faster (compute), but performs worse than moment matching

Meaningful error bars are useful in RL

Loss Function



- Typical loss function: quadratic
- We use a saturating cost function instead. Why is that?

Quadratic Loss

Assume $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and $c(\boldsymbol{x}) = \boldsymbol{x}^\top \boldsymbol{x}$. Then

$$\mathbb{E}[c(\boldsymbol{x})] = \mathbb{E}_{\boldsymbol{x}}[\boldsymbol{x}^\top \boldsymbol{x}] = \operatorname{tr}(\boldsymbol{\Sigma}) + \boldsymbol{\mu}^\top \boldsymbol{\mu}$$

- Scales quadratically in the length of $\boldsymbol{\mu}$ (mean deviation from target)
- Scales linearly in the marginal uncertainty of $\boldsymbol{x}$

Quadratic Loss

Assume $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and $c(\boldsymbol{x}) = \boldsymbol{x}^\top \boldsymbol{x}$. Then

$$\mathbb{E}[c(\boldsymbol{x})] = \mathbb{E}_{\boldsymbol{x}}[\boldsymbol{x}^\top \boldsymbol{x}] = \mathrm{tr}(\boldsymbol{\Sigma}) + \boldsymbol{\mu}^\top \boldsymbol{\mu}$$

- Scales quadratically in the length of $\boldsymbol{\mu}$ (mean deviation from target)
- Scales linearly in the marginal uncertainty of $\boldsymbol{x}$
- If the $p(\boldsymbol{x})$ is either centered far from the origin (target state) or the marginal uncertainties are large, we incur high cost
- In the early stages of learning, our models are uncertain, and we are far from the target

Quadratic Loss

Assume $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and $c(\boldsymbol{x}) = \boldsymbol{x}^\top \boldsymbol{x}$. Then

$$\mathbb{E}[c(\boldsymbol{x})] = \mathbb{E}_{\boldsymbol{x}}[\boldsymbol{x}^\top \boldsymbol{x}] = \text{tr}(\boldsymbol{\Sigma}) + \boldsymbol{\mu}^\top \boldsymbol{\mu}$$

- Scales quadratically in the length of $\boldsymbol{\mu}$ (mean deviation from target)
- Scales linearly in the marginal uncertainty of $\boldsymbol{x}$
- If the $p(\boldsymbol{x})$ is either centered far from the origin (target state) or the marginal uncertainties are large, we incur high cost
- In the early stages of learning, our models are uncertain, and we are far from the target

▶▶ Finding a path to the target may be very costly and even suboptimal (if the planning horizon is not sufficiently large)

Saturating Cost

Assume $\boldsymbol{x} \sim \mathcal{N}\big(\boldsymbol{\mu}, \boldsymbol{\Sigma}\big)$ and $c(\boldsymbol{x}) = 1 - \exp(-\frac{1}{2}\boldsymbol{x}^\top \boldsymbol{x})$. Then

$$\mathbb{E}[c(\boldsymbol{x})] = 1 - \underbrace{|\boldsymbol{I} + \boldsymbol{\Sigma}|^{-\frac{1}{2}}}_{\in[0,1]} \underbrace{\exp(-\tfrac{1}{2}\boldsymbol{\mu}^\top (\boldsymbol{I} + \boldsymbol{\Sigma})^{-1}\boldsymbol{\mu})}_{\in[0,1]} \in [0, 1]$$

- For large $\boldsymbol{\mu}$ (far away from the target) the cost tends to 1

Saturating Cost

Assume $\boldsymbol{x} \sim \mathcal{N}\big(\boldsymbol{\mu}, \boldsymbol{\Sigma}\big)$ and $c(\boldsymbol{x}) = 1 - \exp(-\frac{1}{2}\boldsymbol{x}^\top\boldsymbol{x})$. Then

$$\mathbb{E}[c(\boldsymbol{x})] = 1 - \underbrace{|\boldsymbol{I} + \boldsymbol{\Sigma}|^{-\frac{1}{2}}}_{\in[0,1]} \underbrace{\exp(-\tfrac{1}{2}\boldsymbol{\mu}^\top(\boldsymbol{I} + \boldsymbol{\Sigma})^{-1}\boldsymbol{\mu})}_{\in[0,1]} \in [0, 1]$$

- For large $\boldsymbol{\mu}$ (far away from the target) the cost tends to 1
- The cost is only 0 if the mean prediction hits the target ($\boldsymbol{\mu} \approx \mathbf{0}$) and the prediction is certain ($\boldsymbol{\Sigma} \approx \mathbf{0}$)

Saturating Cost

Assume $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and $c(\boldsymbol{x}) = 1 - \exp(-\frac{1}{2}\boldsymbol{x}^\top\boldsymbol{x})$. Then

$$\mathbb{E}[c(\boldsymbol{x})] = 1 - \underbrace{|\boldsymbol{I} + \boldsymbol{\Sigma}|^{-\frac{1}{2}}}_{\in[0,1]} \underbrace{\exp(-\tfrac{1}{2}\boldsymbol{\mu}^\top(\boldsymbol{I} + \boldsymbol{\Sigma})^{-1}\boldsymbol{\mu})}_{\in[0,1]} \in [0, 1]$$

- For large $\boldsymbol{\mu}$ (far away from the target) the cost tends to 1
- The cost is only 0 if the mean prediction hits the target ($\boldsymbol{\mu} \approx \boldsymbol{0}$) and the prediction is certain ($\boldsymbol{\Sigma} \approx \boldsymbol{0}$)
- Large $\boldsymbol{\Sigma}$ can be good (especially if $\boldsymbol{\mu}$ is also large)

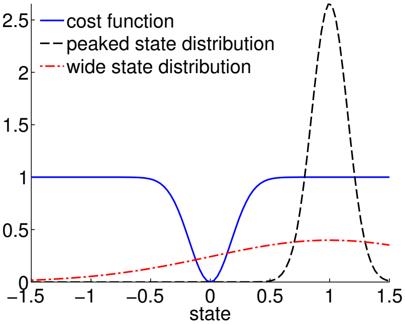
Saturating Cost

Assume $\boldsymbol{x} \sim \mathcal{N}\big(\boldsymbol{\mu}, \boldsymbol{\Sigma}\big)$ and $c(\boldsymbol{x}) = 1 - \exp(-\frac{1}{2}\boldsymbol{x}^\top \boldsymbol{x})$. Then

$$\mathbb{E}[c(\boldsymbol{x})] = 1 - \underbrace{|\boldsymbol{I} + \boldsymbol{\Sigma}|^{-\frac{1}{2}}}_{\in[0,1]} \underbrace{\exp(-\tfrac{1}{2}\boldsymbol{\mu}^\top (\boldsymbol{I} + \boldsymbol{\Sigma})^{-1}\boldsymbol{\mu})}_{\in[0,1]} \in [0, 1]$$

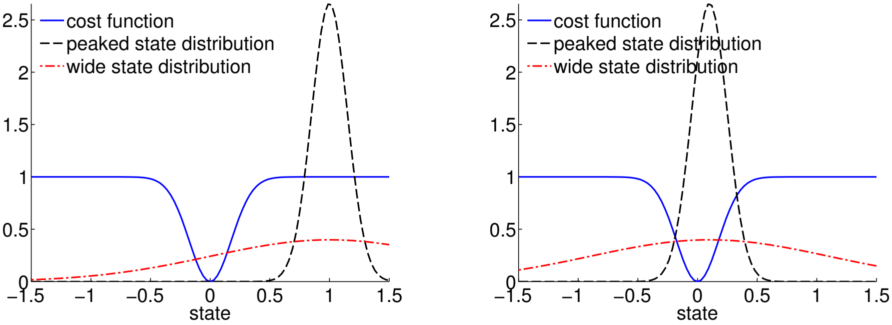
- For large $\boldsymbol{\mu}$ (far away from the target) the cost tends to 1
- The cost is only 0 if the mean prediction hits the target ($\boldsymbol{\mu} \approx \mathbf{0}$) and the prediction is certain ($\boldsymbol{\Sigma} \approx \mathbf{0}$)
- Large $\boldsymbol{\Sigma}$ can be good (especially if $\boldsymbol{\mu}$ is also large)
- If $\boldsymbol{\mu} \approx \mathbf{0}$ then large $\boldsymbol{\Sigma}$ is not good

Automatic Exploration



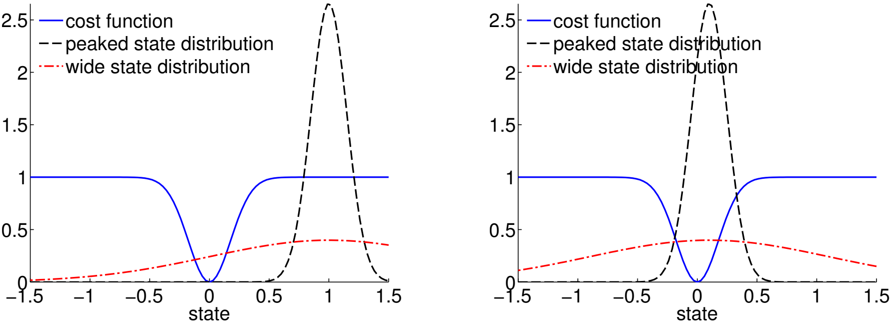
- Far away from the target, uncertainty is favored

Automatic Exploration



- Far away from the target, uncertainty is favored
- Close to the target, uncertainty is discouraged

Automatic Exploration



- Far away from the target, uncertainty is favored
- Close to the target, uncertainty is discouraged

▶▶ Saturating cost automatically deals with the exploration/exploitation trade-off (in some sense)

Probabilistic Model Essential?

DEMO

- Probabilistic model: GP
- Deterministic model: Mean function of GP (still nonparametric)

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

Probabilistic Model Essential?

DEMO

- Probabilistic model: GP
- Deterministic model: Mean function of GP (still nonparametric)

Table: Average learning success with non-parametric transition models

	GP	"Deterministic" GP
Learning success	**94.52%**	**0%**

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

Probabilistic Model Essential?

DEMO

- Probabilistic model: GP
- Deterministic model: Mean function of GP (still nonparametric)

Table: Average learning success with non-parametric transition models

	GP	"Deterministic" GP
Learning success	**94.52%**	**0%**

Reasons for failure of deterministic model:

- Model errors: Long-term predictions make absolutely no sense, and the predicted states are nowhere near the target
  ⏩ No gradient signal
- No automatic exploration (model and policy are deterministic)
  ⏩ Stochastic policy fixes this to some degree

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

Other Topics

- Predictive error bars are useful for **debugging** (code, model, RL algorithm, ...)
- Learning with **sparse rewards** (get reward only in a tiny area around the goal state) can work, even in continuous state spaces (Deisenroth & Rasmussen, 2011)
- Use predictive uncertainty for **safe exploration** (e.g., Sui et al., 2015; Berkenkamp et al., 2017; Kamthe & Deisenroth, 2018)
- Use BO-type acquisition functions for **exploration incentives** (McAllister 2017)
- **Meta reinforcement learning** (Sæmundsson et al., 2018)
- ...

Wrap-up

- Data-efficient RL is often critical when working with real-world systems
- Model-based RL is one way to accelerate learning
- Key: **Probabilistic models reduce the effect of model errors**
- "Faithful" uncertainty propagation leads to faster learning
- Classical quadratic losses are not good when working with uncertainty
- Saturating loss allows for automatic exploration/exploitation

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] K. Chua, R. Calandra, R. McAllister, and S. Levine. Deep Reinforcement Learning in a Handful of Trials Using Probabilistic Dynamics Models. In *Advances in Neural Information Processing Systems*, 2018.

[6] M. P. Deisenroth. *Efficient Reinforcement Learning using Gaussian Processes*, volume 9 of *Karlsruhe Series on Intelligent Sensor-Actuator-Systems*. KIT Scientific Publishing, Nov. 2010. ISBN 978-3-86644-569-7.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

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

References II

[13] 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.

[14] 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.

[15] 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.

[16] R. McAllister. *Bayesian Learning for Data-Efficient Control*. PhD thesis, 2017.

[17] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. Human-Level Control through Deep Reinforcement Learning. *Nature*, 518:529–533, 2015.

[18] 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.

[19] 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.

[20] 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.

[21] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the Game of Go with Deep Neural Networks and Tree Search. *Nature*, 529(7587):484, 2016.

[22] 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.