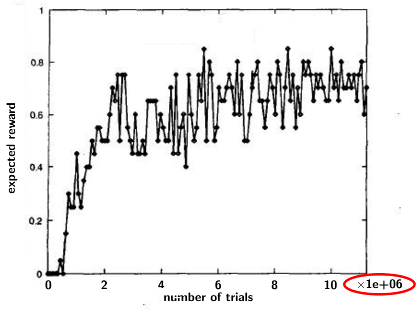
PILCO: A Model-Based and Data-Efficient Approach to Policy Search

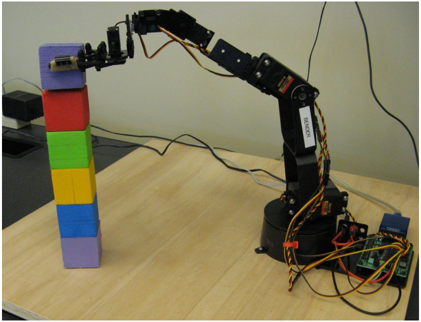
Marc Peter Deisenroth and Carl Edward Rasmussen

Talk at
International Conference on Machine Learning
Bellevue, WA, USA
July 1, 2011

Motivation



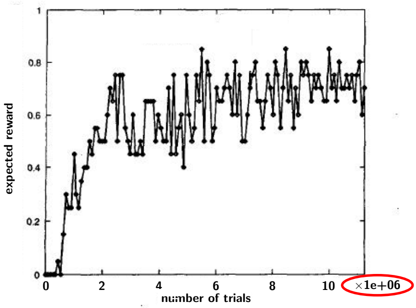
(a) Typical learning curve for cart-pole balancing.



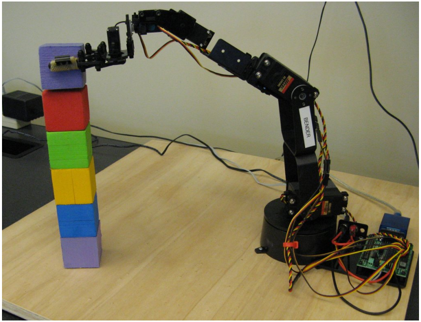
(b) Lynxmotion robotic arm.

- RL often data inefficient if we learn from scratch: needs too many trials
  ➡ largely inapplicable to mechanical systems
- Make RL **more data efficient** (get away with fewer trials)
  - More informative prior knowledge (e.g., demonstrations, system equations)
  - **Extract more valuable information from data**

Motivation



(c) Typical learning curve for cart-pole balancing.



(d) Lynxmotion robotic arm.

- RL often data inefficient if we learn from scratch: needs too many trials
  ➡ largely inapplicable to mechanical systems
- Make RL **more data efficient** (get away with fewer trials)
  - ~~More informative prior knowledge (e.g., demonstrations, system equations)~~
  - **Extract more valuable information from data**

Problem Formulation

Objective

Learn a **policy** π^* that yields minimal **expected long-term cost** $J^\pi(\boldsymbol{\theta})$

$$J^\pi(\boldsymbol{\theta}) = \sum_{t=0}^{T} \mathbb{E}_{\mathbf{x}_t}[c(\mathbf{x}_t)|\pi]$$

Follow π for T steps starting from $p(\mathbf{x}_0)$

- **Policy parameters** $\boldsymbol{\theta}$
- **Cost** $c(\mathbf{x}_t)$ of being in state $\mathbf{x}_t$. We choose

$$c(\mathbf{x}_t) = 1 - \exp(-\tfrac{1}{2}\|\mathbf{x}_t - \mathbf{x}_{\text{target}}\|^2/\sigma_c^2) \quad \in [0, 1]$$

Problem Formulation

Objective

Learn a **policy** π^* that yields minimal **expected long-term cost** $J^\pi(\boldsymbol{\theta})$

$$J^\pi(\boldsymbol{\theta}) = \sum_{t=0}^{T} \mathbb{E}_{\mathbf{x}_t}[c(\mathbf{x}_t)|\pi]$$

Follow π for T steps starting from $p(\mathbf{x}_0)$

- **Policy parameters** $\boldsymbol{\theta}$
- **Cost** $c(\mathbf{x}_t)$ of being in state $\mathbf{x}_t$. We choose

$$c(\mathbf{x}_t) = 1 - \exp(-\tfrac{1}{2}\|\mathbf{x}_t - \mathbf{x}_{\text{target}}\|^2/\sigma_c^2) \quad \in [0, 1]$$

Challenges:

- **Data-efficient** solution (few trials)
- **Unknown transition dynamics** $f : (\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) \mapsto \mathbf{x}_t$
- **No expert knowledge/demonstrations** available ➡ learn from scratch

Making RL Efficient

Model-based RL

- Learn model of transition dynamics f
- Use model for internal simulation ➡ certainty equivalence assumption (Schneider, NIPS 1997; Bagnell and Schneider, ICRA 2001)
- Learn policy based on these simulations
- Hope: few interactions with system
  ➡ suffers from **model errors**, but can be **data efficient**

Making RL Efficient

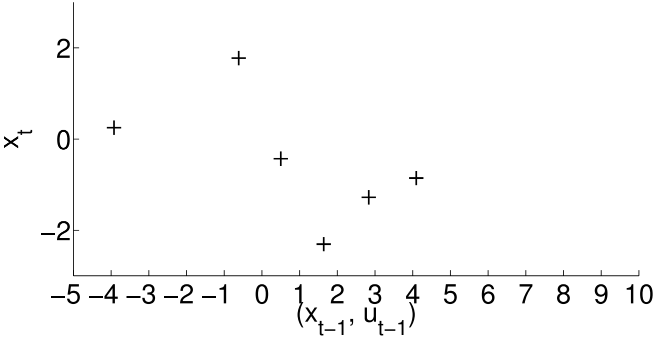
Model-based RL

- Learn model of transition dynamics f
- Use model for internal simulation ➡ certainty equivalence assumption (Schneider, NIPS 1997; Bagnell and Schneider, ICRA 2001)
- Learn policy based on these simulations
- Hope: few interactions with system
  ➡ suffers from **model errors**, but can be **data efficient**

➡ Being efficient (often) requires **dealing with model errors** (Atkeson and Santamaría, ICML 1997)

Dealing with Model Errors

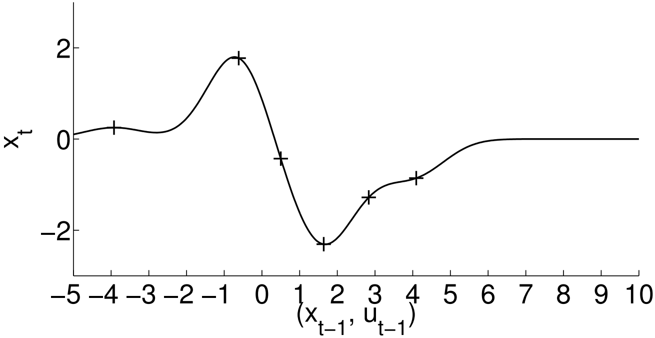
Task: find a (transition) function $f : (\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) \mapsto \mathbf{x}_t$



Training set for model learning

Dealing with Model Errors

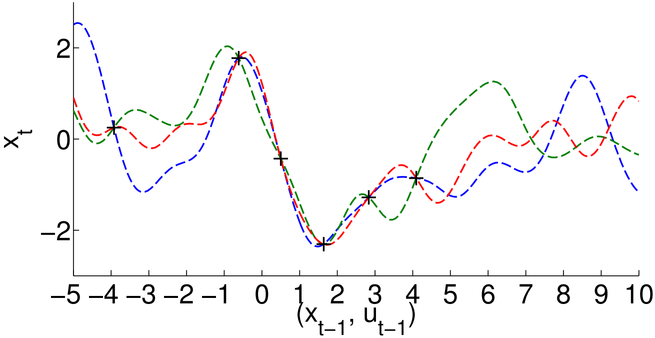
Task: find a (transition) function $f : (\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) \mapsto \mathbf{x}_t$



Deterministic (MAP) function approximator

Dealing with Model Errors

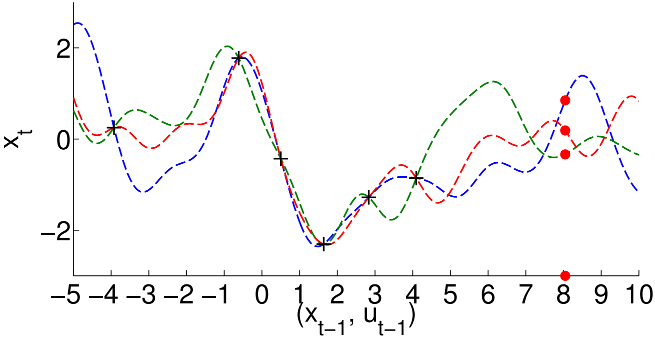
Task: find a (transition) function $f : (\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) \mapsto \mathbf{x}_t$



Other plausible function approximators

Dealing with Model Errors

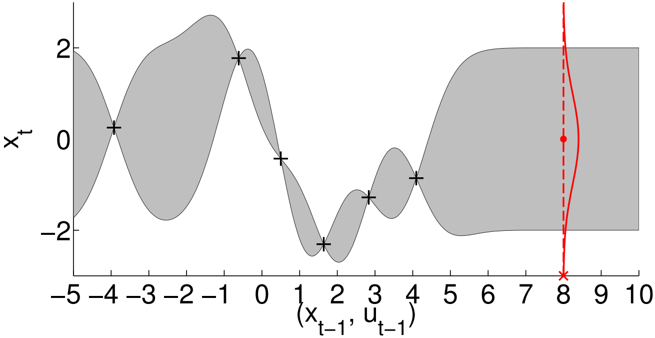
Task: find a (transition) function $f : (\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) \mapsto \mathbf{x}_t$



Other plausible function approximators

Dealing with Model Errors

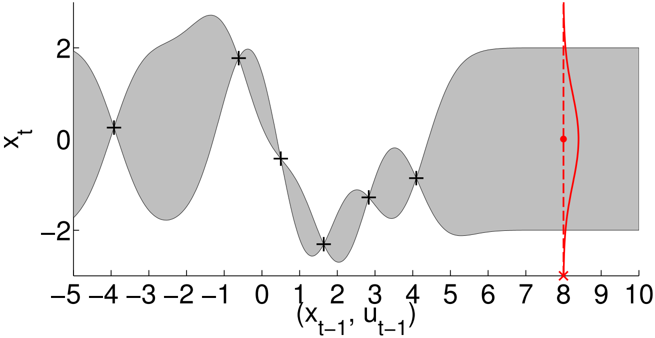
Task: find a (transition) function $f : (\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) \mapsto \mathbf{x}_t$



Probabilistic function approximator: distribution over plausible functions

Dealing with Model Errors

Task: find a (transition) function $f : (\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) \mapsto \mathbf{x}_t$



Probabilistic function approximator: distribution over plausible functions

- Express **model uncertainty** about the function at unobserved locations
- **Must** use a **probabilistic** function approximator
- **Pilco** framework (Nonparametric Gaussian processes for dynamics model)

PILCO

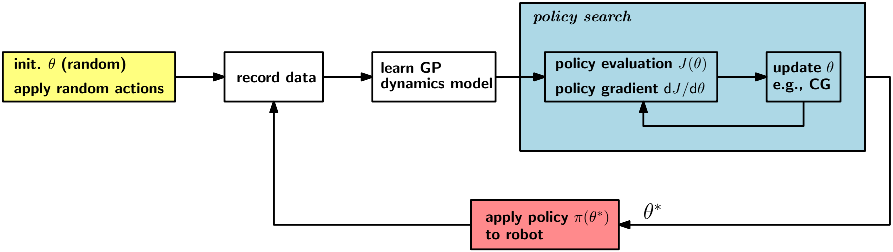
- Probabilistic inference for learning control
- Model-based **policy search** method with **analytic policy gradients**
  ➡ find good policy parameters $\boldsymbol{\theta}^*$
- **Gaussian processes** for probabilistic dynamics model
  zero prior mean, SE covariance function

PILCO

- Probabilistic inference for learning control
- Model-based **policy search** method with **analytic policy gradients**
  ➡ find good policy parameters $\boldsymbol{\theta}^*$
- **Gaussian processes** for probabilistic dynamics model
  zero prior mean, SE covariance function
- Explicitly describe **model uncertainties**
  ➡ Take them into account during planning
  ➡ Reduce effect of model errors
  ➡ Allows for learning from scratch (episodic tasks)

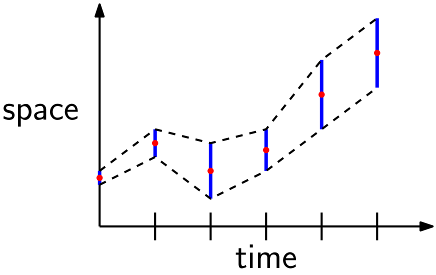
PILCO

- Probabilistic inference for learning control
- Model-based **policy search** method with **analytic policy gradients**
  ➡ find good policy parameters $\boldsymbol{\theta}^*$
- **Gaussian processes** for probabilistic dynamics model
  zero prior mean, SE covariance function
- Explicitly describe **model uncertainties**
  ➡ Take them into account during planning
  ➡ Reduce effect of model errors
  ➡ Allows for learning from scratch (episodic tasks)



Approximate Inference for Policy Evaluation

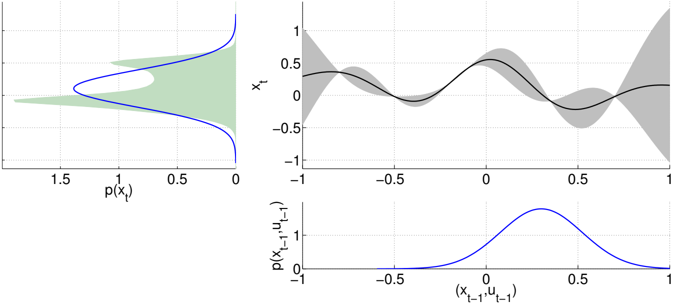
- Want to compute $J^{\pi}(\boldsymbol{\theta}) = \sum_t \mathbb{E}[c(\mathbf{x}_t)]$
- Obtain one-step transition probabilities $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t-1})$ from GP dynamics model
- Idea: **cascade** predictions to get $p(\mathbf{x}_1), \dots, p(\mathbf{x}_T)$



➡ $J^{\pi}(\boldsymbol{\theta})$ can be evaluated (assuming $\mathbb{E}_{\mathbf{x}}[c(\mathbf{x})]$ can be computed)

Approximate Inference for Policy Evaluation (2)

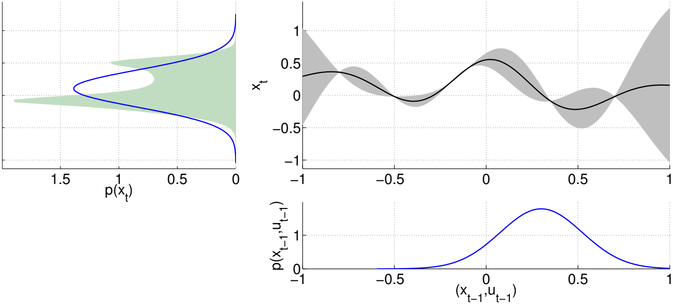
- **Problem:** predictions $p(\mathbf{x}_1), \ldots, p(\mathbf{x}_T)$ cannot be computed exactly.
- Approximate inference required
  - Robust **moment matching** approximation of predictive distribution (Quiñonero-Candela et al., ICASSP 2003; Deisenroth et al., ICML 2009)



➡ Get approximate Gaussian state distributions $p(\mathbf{x}_1), \ldots, p(\mathbf{x}_T)$

Approximate Inference for Policy Evaluation (2)

- **Problem:** predictions $p(\mathbf{x}_1), \dots, p(\mathbf{x}_T)$ cannot be computed exactly.
- Approximate inference required
  - Robust **moment matching** approximation of predictive distribution (Quiñonero-Candela et al., ICASSP 2003; Deisenroth et al., ICML 2009)

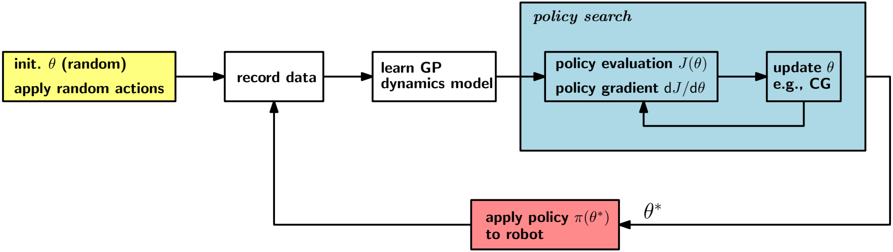


➡ Get approximate Gaussian state distributions $p(\mathbf{x}_1), \dots, p(\mathbf{x}_T)$

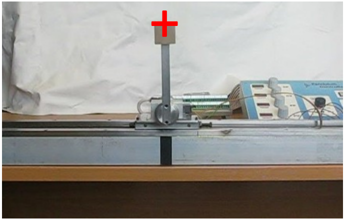
➡ **Analytic policy evaluation and policy gradients d$J^\pi(\boldsymbol{\theta})$/d$\boldsymbol{\theta}$**

Results

- Hardware applicability
- High-dimensional problems
- Data efficiency

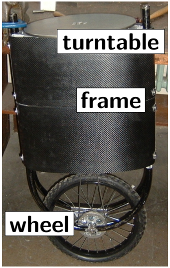


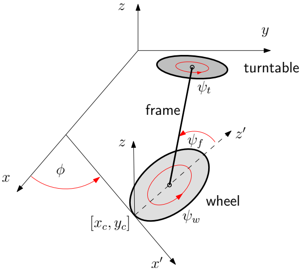
Standard Benchmark Problem

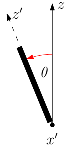


- State space: $\mathbf{x} \in \mathbb{R}^4$
- Policy parameters: $\boldsymbol{\theta} \in \mathbb{R}^{300}$
- Control frequency: 10 Hz
- $<$ 10 trials
- $\approx$ 20 seconds of interaction time

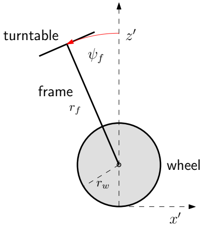
Scaling to Higher Dimensions: Unicycling







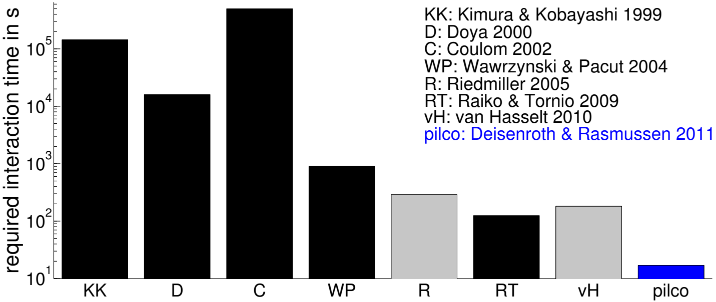
sideways tilt of wheel



tilt of the frame

- State space: $\mathbf{x} \in \mathbb{R}^{12}$, $\boldsymbol{\theta} \in \mathbb{R}^{26}$
- 2-dimensional controls (wheel torque and flywheel torque)
- Control frequency: 6.66 Hz
- $\approx$ 15–20 trials (including 5 random trials)
- $\approx$ 30 seconds interaction time

Data Efficiency



Cart-pole task (results from literature)

- Only "**learning from scratch**" (no demonstrations etc.)
- Gray bars: balancing
- Black bars: swing up and balancing
- Slightly different setups (masses, rewards, discretization)
- About one **order of magnitude less interaction** time than best other method

Wrap-up

- PILCO: Data-efficient model-based policy search method
- **No expert knowledge/demonstrations** required
- Key point: **reduce model errors** by using probabilistic dynamics models
- **Unprecedented speed of learning**
- Hardware applicability, scaling to high dimensions

Wrap-up

- PILCO: Data-efficient model-based policy search method
- **No expert knowledge/demonstrations** required
- Key point: **reduce model errors** by using probabilistic dynamics models
- **Unprecedented speed of learning**
- Hardware applicability, scaling to high dimensions

- Use probabilistic models to express what you don't know for sure

Wrap-up

- PILCO: Data-efficient model-based policy search method
- **No expert knowledge/demonstrations** required
- Key point: **reduce model errors** by using probabilistic dynamics models
- **Unprecedented speed of learning**
- Hardware applicability, scaling to high dimensions

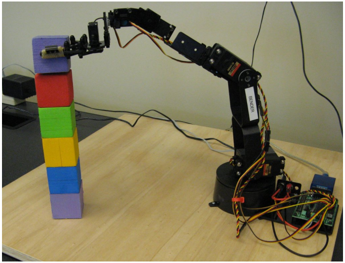
- Use probabilistic models to express what you don't know for sure

`http://mlg.eng.cam.ac.uk/carl/pilco`

`http://www.cs.washington.edu/homes/marc/pilco`

`marc@cs.washington.edu`

Controlling a Really Noisy Robot



- Low-cost robotic manipulator
- Kinect-style depth camera only sensor
- Learn to stack blocks (from scratch)

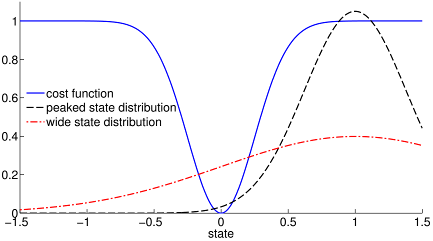
(Deisenroth et al., R:SS 2011)

Parameters to be set

- number of basis functions (policy)
- general system properties (e.g. length of pendulum)
- cost function
- control frequency (Δ_t)
- length of control/prediction horizon T

Exploration/Exploitation

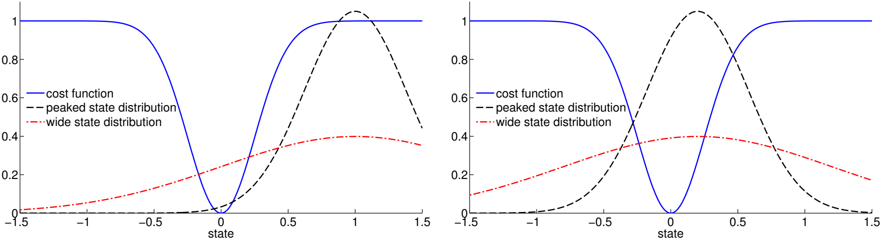
- Compute $\mathbb{E}[c(\mathbf{x}_t)]$
- We choose $c(\mathbf{x}) = 1 - \exp(-\frac{1}{2}\|\mathbf{x} - \mathbf{x}_{\text{target}}\|^2/\sigma_c^2)$



- Far away from the target, uncertainty (this comes from averaging out model uncertainty!) is favorable ➡ **explore**

Exploration/Exploitation

- Compute $\mathbb{E}[c(\mathbf{x}_t)]$
- We choose $c(\mathbf{x}) = 1 - \exp(-\frac{1}{2}\|\mathbf{x} - \mathbf{x}_{\text{target}}\|^2/\sigma_c^2)$



- Far away from the target, uncertainty (this comes from averaging out model uncertainty!) is favorable ➡ **explore**
- Close to the target, we want to be certain ➡ **exploit**

Computational complexity

- training dynamics models

$$\mathcal{O}(dn^3)$$

- predictions (policy evaluation)

$$\mathcal{O}(d^3n^2)$$

➡ sparse approximations speed up (factor n)

Policy improvement

- policy: parameterized function (parameters $\boldsymbol{\theta}$)
- $\mathbf{x}_t$ is a function of $\boldsymbol{\theta}$ through

$$\begin{aligned} \mathbf{x}_t &= f(\mathbf{x}_{t-1}, \mathbf{u}_{t-1})\,, \\ \mathbf{u}_{t-1} &= \pi(\mathbf{x}_{t-1}, \boldsymbol{\theta}) \end{aligned}$$

Policy improvement

- policy: parameterized function (parameters $\boldsymbol{\theta}$)
- $\mathbf{x}_t$ is a function of $\boldsymbol{\theta}$ through

$$\mathbf{x}_t = f(\mathbf{x}_{t-1}, \mathbf{u}_{t-1}),$$
$$\mathbf{u}_{t-1} = \pi(\mathbf{x}_{t-1}, \boldsymbol{\theta})$$

- policy evaluation can be done analytically (with approximations)
  ➡ **analytic gradients** (chain rule) are available:

$$\frac{\mathrm{d}J^{\pi}(\boldsymbol{\theta})}{\mathrm{d}\boldsymbol{\theta}}$$

Policy improvement

- policy: parameterized function (parameters $\boldsymbol{\theta}$)
- $\mathbf{x}_t$ is a function of $\boldsymbol{\theta}$ through

$$\mathbf{x}_t = f(\mathbf{x}_{t-1}, \mathbf{u}_{t-1}),$$
$$\mathbf{u}_{t-1} = \pi(\mathbf{x}_{t-1}, \boldsymbol{\theta})$$

- policy evaluation can be done analytically (with approximations)
  ➡ **analytic gradients** (chain rule) are available:

$$\frac{\mathrm{d}J^{\pi}(\boldsymbol{\theta})}{\mathrm{d}\boldsymbol{\theta}}$$

➡ use your favorite toolbox for nonconvex optimization to get $\boldsymbol{\theta}^*$

➡ **no value function model** required

Policy parametrization

$$\pi(\mathbf{x}) = \sum_{i=1}^{n} w_i \phi_i(\mathbf{x}) = \sum_{i=1}^{n} w_i \exp\left(-\tfrac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^\top \boldsymbol{\Lambda}^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)\right)$$

$$\boldsymbol{\Lambda} = \mathrm{diag}(\ell_1^2, \ldots, \ell_d^2)\,, \quad d = \mathsf{dim}(\mathbf{x})$$

Policy parametrization

$$\pi(\mathbf{x}) = \sum_{i=1}^{n} w_i \phi_i(\mathbf{x}) = \sum_{i=1}^{n} w_i \exp\left(-\tfrac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^\top \boldsymbol{\Lambda}^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)\right)$$

$$\boldsymbol{\Lambda} = \mathrm{diag}(\ell_1^2, \dots, \ell_d^2)\,, \quad d = \mathsf{dim}(\mathbf{x})$$

policy parameters $\boldsymbol{\theta}$

- n weights w_i (per control dimension)
- d length-scales $\ell_1, \dots, \ell_d$ (per control dimension)
- n centers $\boldsymbol{\mu}_i \in \mathbb{R}^d$ of basis functions (shared across control dimensions)

Policy parametrization

$$\pi(\mathbf{x}) = \sum_{i=1}^{n} w_i \phi_i(\mathbf{x}) = \sum_{i=1}^{n} w_i \exp\left(-\tfrac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_i)^\top \boldsymbol{\Lambda}^{-1}(\mathbf{x} - \boldsymbol{\mu}_i)\right)$$

$$\boldsymbol{\Lambda} = \mathrm{diag}(\ell_1^2, \ldots, \ell_d^2)\,, \quad d = \mathsf{dim}(\mathbf{x})$$

policy parameters $\boldsymbol{\theta}$

- n weights w_i (per control dimension)
- d length-scales $\ell_1, \ldots, \ell_d$ (per control dimension)
- n centers $\boldsymbol{\mu}_i \in \mathbb{R}^d$ of basis functions (shared across control dimensions)
- $(d+1)n + d$ parameters
  example: $n = 50, d = 6, \mathsf{dim}(\mathbf{u}) = 2 \longrightarrow |\boldsymbol{\theta}| \approx 400$

[1] Rémi Coulom. Reinforcement Learning Using Neural Networks, with Applications to Motor Control. PhD thesis, Institut National Polytechnique de Grenoble, 2002.

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

[3] Marc P. Deisenroth, Marco F. Huber, and Uwe D. Hanebeck. Analytic Moment-based Gaussian Process Filtering. In L. Bouttou and M. L. Littman, editors, Proceedings of the 26th International Conference on Machine Learning, pages 225–232, Montreal, QC, Canada, June 2009. Omnipress.

[4] Marc P. Deisenroth and Carl E. Rasmussen. PILCO: A Model-Based and Data-Efficient Approach to Policy Search. In Proceedings of the International Conference on Machine Learning, Bellevue, USA, June 2011.

[5] Marc P. Deisenroth, Carl E. Rasmussen, and Dieter Fox. Learning to Control a Low-Cost Manipulator using Data-Efficient Reinforcement Learning. In Proceedings of the International Conference on Robotics: Science and Systems, 2011.

[6] Kenji Doya. Reinforcement Learning in Continuous Time and Space. Neural Computation, 12(1):219–245, January 2000.

[7] Hajime Kimura and Shigenobu Kobayashi. Efficient Non-Linear Control by Combining Q-learning with Local Linear Controllers. In Proceedings of the 16th International Conference on Machine Learning, pages 210–219, 1999.

[8] Joaquin Quiñonero-Candela, Agathe Girard, Jan Larsen, and Carl 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, April 2003.

[9] Tapani Raiko and Matti Tornio. Variational Bayesian Learning of Nonlinear Hidden State-Space Models for Model Predictive Control. Neurocomputing, 72(16–18):3702–3712, 2009.

[10] Carl E. Rasmussen and Marc P. Deisenroth. Recent Advances in Reinforcement Learning, volume 5323 of Lecture Notes in Computer Science, chapter Probabilistic Inference for Fast Learning in Control, pages 229–242. Springer-Verlag, November 2008.

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

[12] Martin Riedmiller. Neural Fitted Q Iteration—First Experiences with a Data Efficient Neural Reinforcement Learning Method. In Proceedings of the 16th European Conference on Machine Learning, Porto, Portugal, 2005.

[13] Hado van Hasselt. Insights in Reinforcement Learning: Formal Analysis and Empirical Evaluation of Temporal-Difference Learning Algorithms. Wöhrmann Print Service, 2010. ISBN 978-90-39354964.

[14] Paweł Wawrzynski and Andrzej Pacut. Model-free off-policy Reinforcement Learning in Continuous Environment. In Proceedings of the INNS-IEEE International Joint Conference on Neural Networks, pages 1091–1096, 2004.