Reinforcement Learning from Scarce Data

Marc Deisenroth

m.deisenroth@ucl.ac.uk

Centre for Artificial Intelligence
Department of Computer Science
University College London

@mpd37

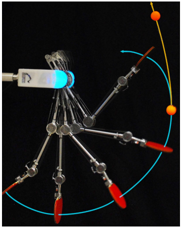
RIKEN Center for Advanced Intelligence Project

November 25, 2019

Autonomous Robots: Key Challenges

- Three key challenges in autonomous systems:

  Modeling. Predicting. Decision making.



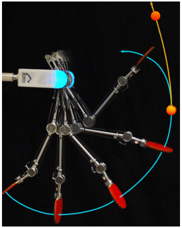
Robotics

Autonomous Robots: Key Challenges

- Three key challenges in autonomous systems:

  Modeling. Predicting. Decision making.

- No human in the loop ⏩ "Learn" from data
- Automatically extract information
- Data-efficient (fast) learning
- Uncertainty: sensor noise, unknown processes, limited knowledge, ...



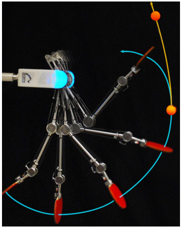
Robotics

Autonomous Robots: Key Challenges

- Three key challenges in autonomous systems:

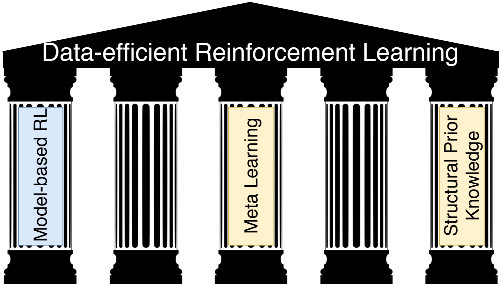
  Modeling. Predicting. Decision making.

- No human in the loop ⏩ "Learn" from data
- Automatically extract information
- Data-efficient (fast) learning
- Uncertainty: sensor noise, unknown processes, limited knowledge, ...



Robotics

⏩ **Reinforcement learning**
subject to data efficiency


Data-efficient Reinforcement Learning
Model-based RL
Meta Learning
Structural Prior Knowledge

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

Reinforcement Learning

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

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 $p(\boldsymbol{x}_1|\boldsymbol{\theta}), \dots, p(\boldsymbol{x}_T|\boldsymbol{\theta})$
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*

Fast Reinforcement Learning

Objective

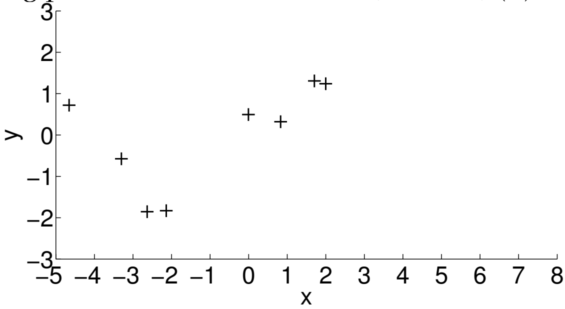
Minimize expected long-term cost $J(\boldsymbol{\theta}) = \sum_t \mathbb{E}[c(\boldsymbol{x}_t)|\boldsymbol{\theta}]$

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*

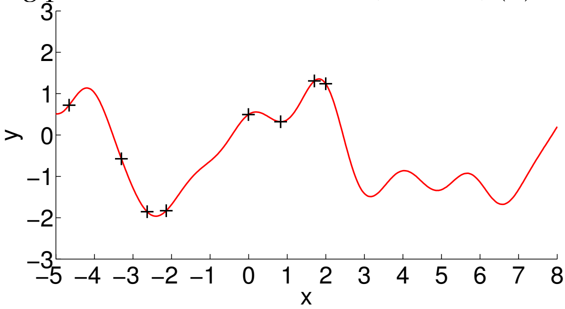
Model learning problem: Find a function $f : x \mapsto f(x) = y$



Observed function values

Model Learning (System Identification)

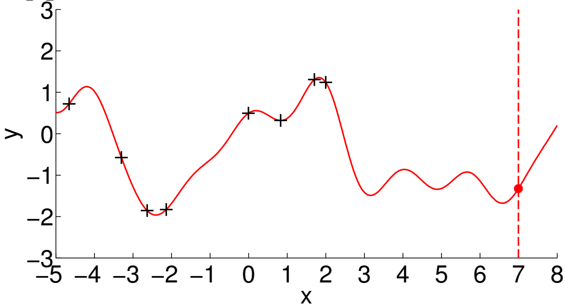
Model learning problem: Find a function $f : x \mapsto f(x) = y$



Plausible model

Model Learning (System Identification)

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

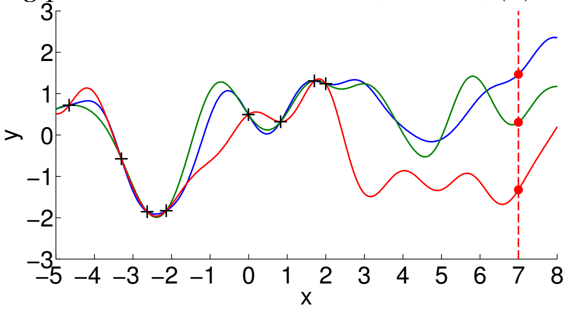


Plausible model

Predictions? Decision Making?

Model Learning (System Identification)

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

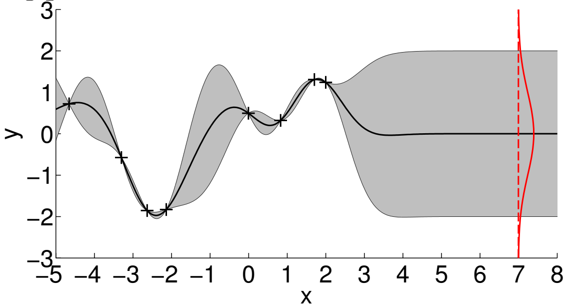


More plausible models

Predictions? Decision Making? Model Errors!

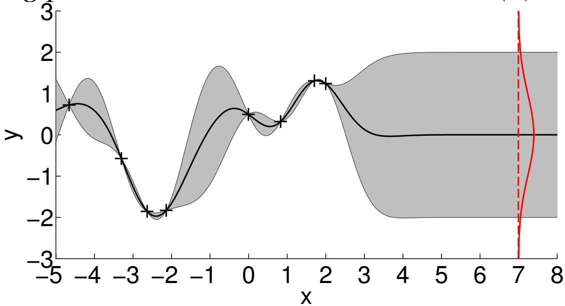
Model Learning (System Identification)

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



Distribution over plausible functions

Model learning 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)

Objective

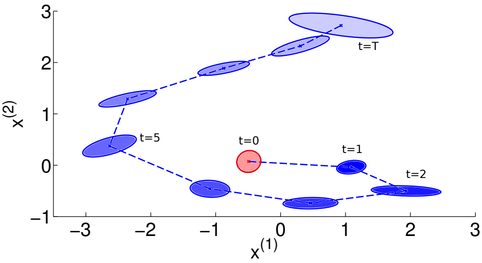
Minimize expected long-term cost $J(\boldsymbol{\theta}) = \sum_t \mathbb{E}[c(\boldsymbol{x}_t)|\boldsymbol{\theta}]$

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*

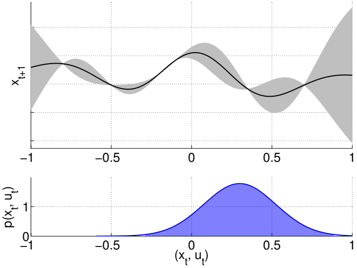
Long-Term Predictions



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

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

Long-Term Predictions

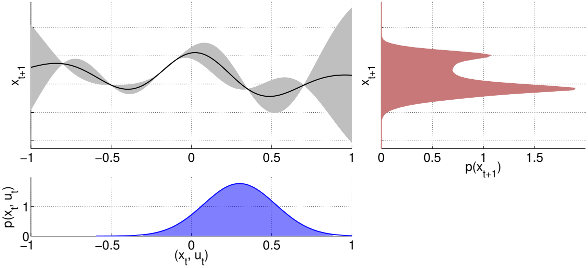


- Iteratively compute $p(\boldsymbol{x}_1|\boldsymbol{\theta}), \dots, p(\boldsymbol{x}_T|\boldsymbol{\theta})$

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

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

Long-Term Predictions

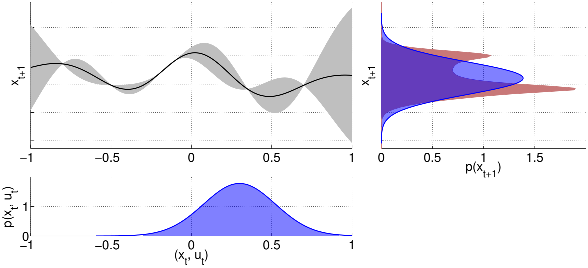


- Iteratively compute $p(\boldsymbol{x}_1|\boldsymbol{\theta}), \ldots, p(\boldsymbol{x}_T|\boldsymbol{\theta})$

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

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

Long-Term Predictions



- 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*

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*

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

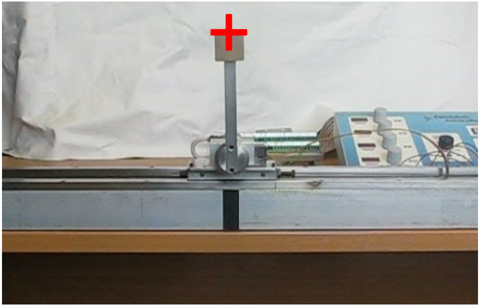
Minimize expected long-term cost $J(\boldsymbol{\theta}) = \sum_t \mathbb{E}[c(\boldsymbol{x}_t)|\boldsymbol{\theta}]$

PILCO Framework: High-Level Steps

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

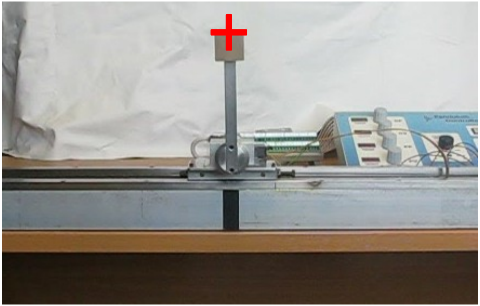


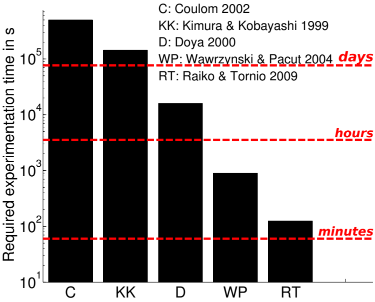
- Swing up and balance a freely swinging pendulum on a cart
- No knowledge about nonlinear dynamics ⏩ Learn from scratch
- Cost function $c(\boldsymbol{x}) = 1 - \exp(-\|\boldsymbol{x} - \boldsymbol{x}_{\text{target}}\|^2)$

- 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



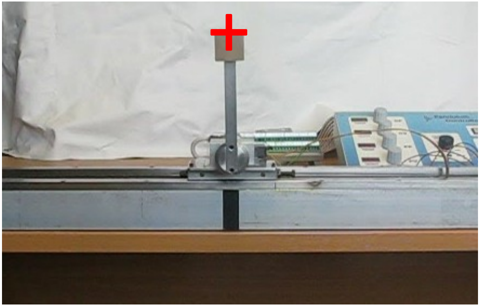


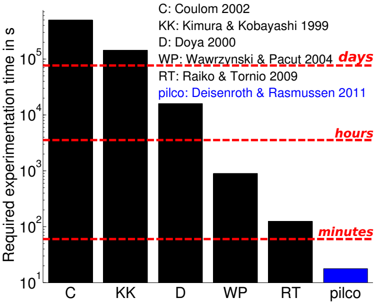
- Swing up and balance a freely swinging pendulum on a cart
- No knowledge about nonlinear dynamics ▸▸ Learn from scratch
- Cost function $c(\boldsymbol{x}) = 1 - \exp(-\|\boldsymbol{x} - \boldsymbol{x}_{\text{target}}\|^2)$

- 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

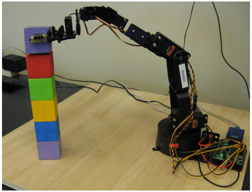




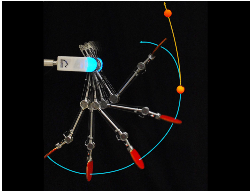
- Swing up and balance a freely swinging pendulum on a cart
- No knowledge about nonlinear dynamics ⏩ Learn from scratch
- Cost function $c(\boldsymbol{x}) = 1 - \exp(-\|\boldsymbol{x} - \boldsymbol{x}_{\text{target}}\|^2)$
- **Unprecedented learning speed** compared to state-of-the-art
- Code: https://github.com/ICL-SML/pilco-matlab

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

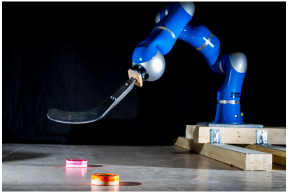
Wide Applicability



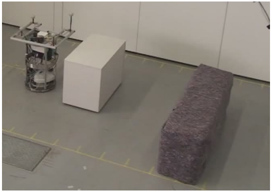
with D Fox



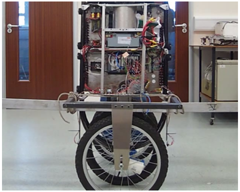
with P Englert, A Paraschos, J Peters



with A Kupcsik, J Peters, G Neumann



B Bischoff (Bosch), ESANN 2013

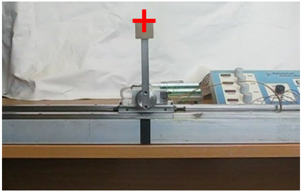
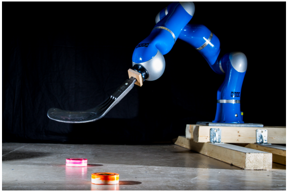
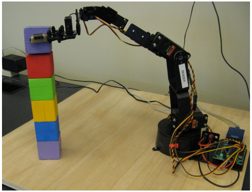


A McHutchon (U Cambridge)

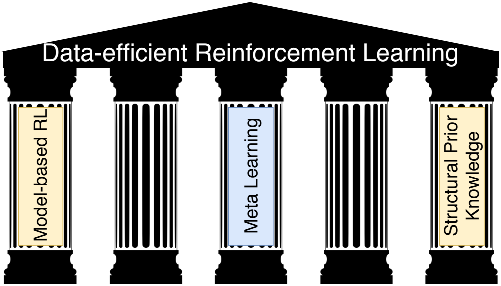
- Application to a wide range of robotic systems

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



Steindór Sæmundsson

Katja Hofmann

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

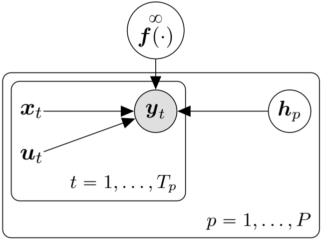
- Separate global and task-specific properties
- Shared global parameters describe general dynamics
- Describe task-specific (local) configurations with latent variable

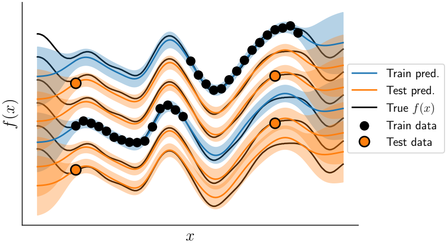
Approach

- 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

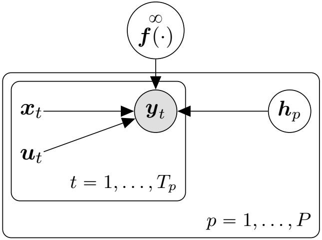
Meta Model Learning with Latent Variables

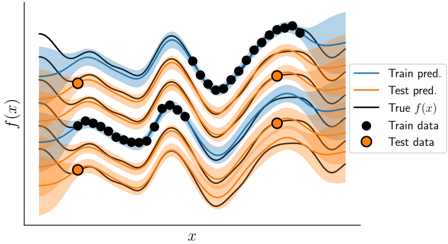




$$y_t = f(x_t, u_t, h_p)$$

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



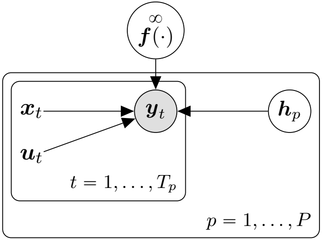


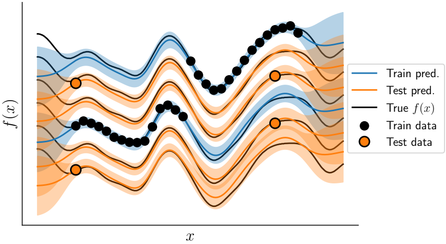
$$y_t = f(x_t, u_t, h_p)$$

- 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



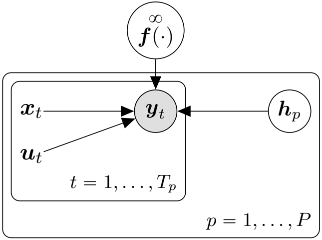


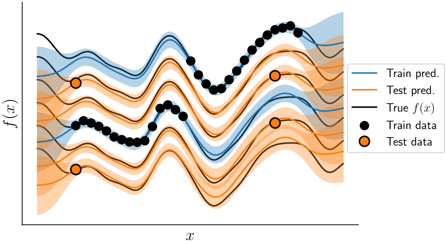
$$y_t = f(x_t, u_t, h_p)$$

- GP captures global properties of the dynamics
- Latent variable 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



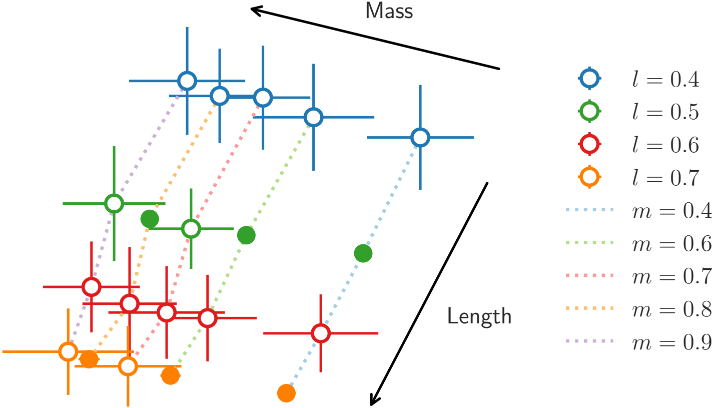


$$\boldsymbol{y}_t = \boldsymbol{f}(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{h}_p)$$

- 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*

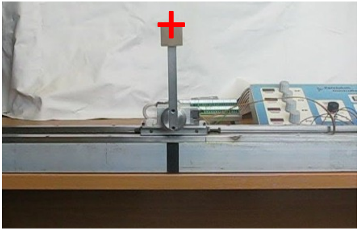
Latent Embeddings



- 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



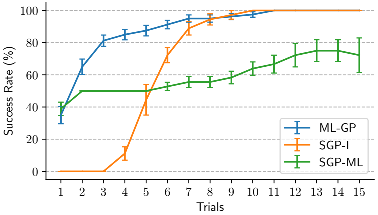
- Pre-trained on 6 training configurations until solved

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

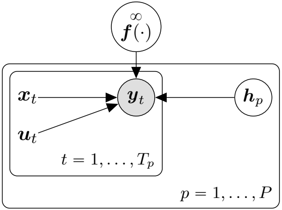


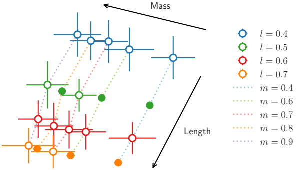
- Few-shot generalization on 4 unseen configurations
- Success: solve all 10 (6 training + 4 test) tasks
- Meta learning: blue
- Independent (GP-MPC): orange
- Aggregated experience model (no latents): green

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

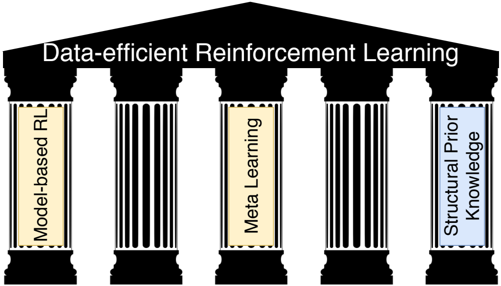
Sæmundsson et al. (UAI, 2018): *Meta Reinforcement Learning with Latent Variable Gaussian Processes*

Summary (2)





- 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



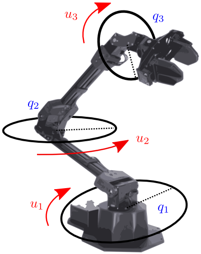
Steindór Sæmundsson

Alexander Terenin

Katja Hofmann

Structural Priors

High-level prior knowledge: e.g., laws of physics or configuration constraints



Equations of motion

$$u = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right) - \frac{\partial L}{\partial q}$$

▶▶ Improve data efficiency and generalization

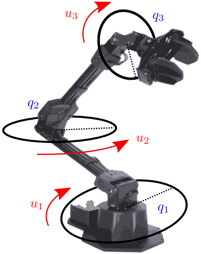
Variational Integrator Networks (VINs)

Network architectures with built-in physics and geometric structure

Outline:

- How we talk about physics
- How we think about neural networks
- How to encode physics and geometry into architecture

Physics: Lagrangian/Hamiltonian Mechanics



Equations of motion

$$u = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right) - \frac{\partial L}{\partial q}$$

- General framework:
  classical mechanics, quantum mechanics, relativity
- Global properties:
  conservation laws, configuration manifold, etc.
- Solve differential equations

Physics: Key Objects

- Configuration space:

$$q \in \mathcal{Q}$$

Physics: Key Objects

- Configuration space:

$$q \in \mathcal{Q}$$

- Lagrangian (specifies physics):

$$L(q(t), \dot{q}(t)) = K - U = \text{kinetic energy} - \text{potential energy}$$

Physics: Key Objects

- Configuration space:

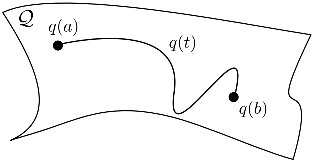
$$q \in \mathcal{Q}$$

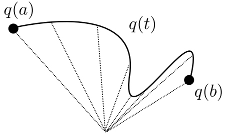
- Lagrangian (specifies physics):

$$L(q(t), \dot{q}(t)) = K - U = \text{kinetic energy} - \text{potential energy}$$

- Action (maps trajectories to real numbers)

$$A = \int_a^b L(q(t), \dot{q}(t))dt$$





Physics: Hamilton's Principle

Hamilton's Principle

Physical paths are stationary points of the action.

Hamilton's Principle

Physical paths are stationary points of the action.

Equations of motion (Euler-Lagrange equation):

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right) - \frac{\partial L}{\partial q} = 0$$

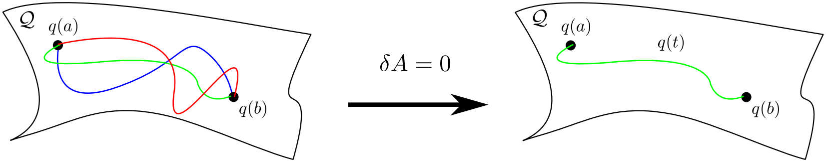
Hamilton's Principle

Physical paths are stationary points of the action.

Equations of motion (Euler-Lagrange equation):

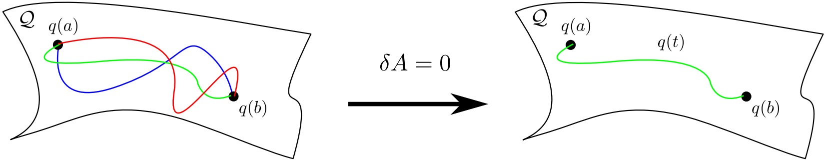
$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right) - \frac{\partial L}{\partial q} = 0$$

The solution $q(t)$ evolves according to the laws of physics.



Physics: Recap

- Lagrangian → Specifies the physics
- Hamilton's principle → Equations of motion
- Solution → Physical path



Neural ODE Perspective

- Residual networks = Learnable approximate ODE solvers

$$\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), t, \theta) \quad \longleftrightarrow \quad \boldsymbol{x}_{t+1} = \boldsymbol{x}_t + f(\boldsymbol{x}(t), \theta)$$

Neural ODE Perspective

- Residual networks = Learnable approximate ODE solvers

$$\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), t, \theta) \quad \longleftrightarrow \quad \boldsymbol{x}_{t+1} = \boldsymbol{x}_t + f(\boldsymbol{x}(t), \theta)$$

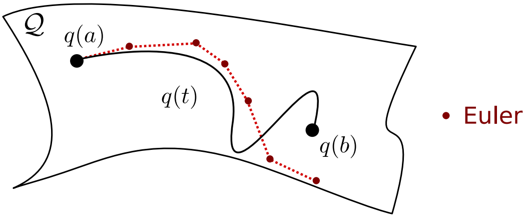
- **Intuition:** Physical networks = Learnable approximations to equations of motion

Neural ODE Perspective

- Residual networks = Learnable approximate ODE solvers

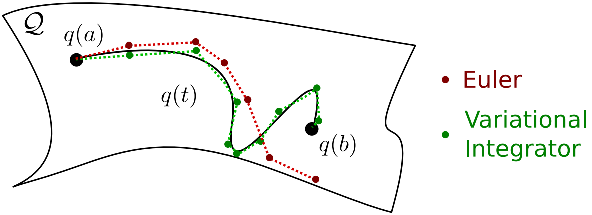
$$\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), t, \theta) \quad \longleftrightarrow \quad \boldsymbol{x}_{t+1} = \boldsymbol{x}_t + f(\boldsymbol{x}(t), \theta)$$

- **Intuition:** Physical networks = Learnable approximations to equations of motion
- **Problem:** Euler discretization leads to significant errors and physically implausible behavior



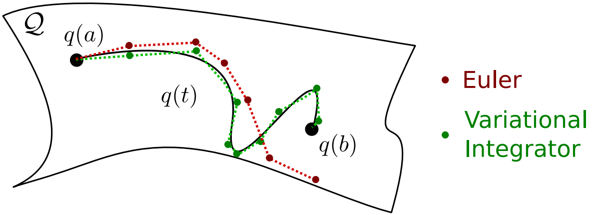
Variational Integrators

Geometric integrators that preserve global (physical) properties



Variational Integrators

Geometric integrators that preserve global (physical) properties



Properties:

- Symplectic (volume preserving)
- Momentum preserving
- Bounded energy behavior

Recipe for Variational Integrator Network

1. Write down parameterized Lagrangian:

$$L_\theta(q(t), \dot{q}(t))$$

Sæmundsson et al. (arXiv:1910.09349): *Variational Integrator Networks for Physically Meaningful Embeddings*

Recipe for Variational Integrator Network

1. Write down parameterized Lagrangian:

$$L_\theta(q(t), \dot{q}(t))$$

2. Derive **explicit** variational integrator:

$$\text{Lagrangian:} \quad q_{t+1} = f_\theta(q_t, q_{t-1})$$
$$\text{Hamiltonian:} \quad [q_{t+1}, \dot{q}_{t+1}] = f_\theta(q_t, \dot{q}_t)$$

Sæmundsson et al. (arXiv:1910.09349): *Variational Integrator Networks for Physically Meaningful Embeddings*

Recipe for Variational Integrator Network

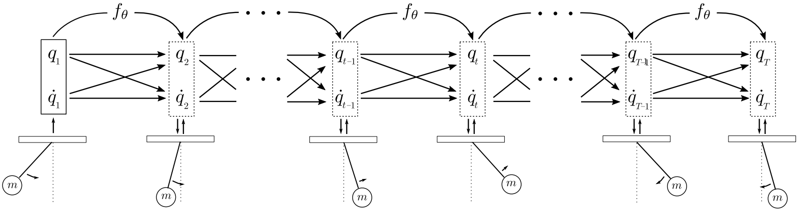
1. Write down parameterized Lagrangian:

$$L_\theta(q(t), \dot{q}(t))$$

2. Derive **explicit** variational integrator:

$$\text{Lagrangian:} \quad q_{t+1} = f_\theta(q_t, q_{t-1})$$
$$\text{Hamiltonian:} \quad [q_{t+1}, \dot{q}_{t+1}] = f_\theta(q_t, \dot{q}_t)$$

3. f_θ defines the network architecture



Sæmundsson et al. (arXiv:1910.09349): *Variational Integrator Networks for Physically Meaningful Embeddings*

Newtonian Potential System:

$$L_\theta(q(t), \dot{q}(t)) = K_\theta(\dot{q}(t)) - U_\theta(q(t))$$

- Newtonian network on $\mathbb{R}^D$

$$q_{t+1} = 2q_t - q_{t-1} - h^2 f_\theta(q_t)$$

VIN Examples

Newtonian Potential System:

$$L_\theta(q(t), \dot{q}(t)) = K_\theta(\dot{q}(t)) - U_\theta(q(t))$$

- Newtonian network on $\mathbb{R}^D$

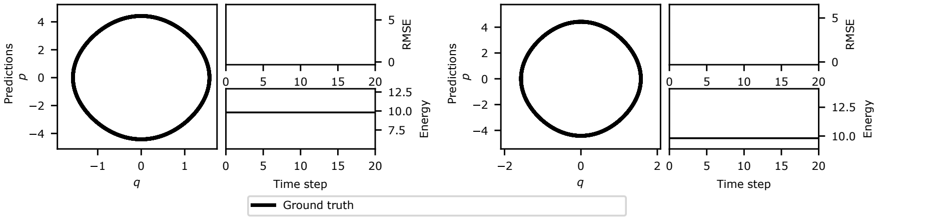
$$q_{t+1} = 2q_t - q_{t-1} - h^2 f_\theta(q_t)$$

- Newtonian network on $SO(2)$

$$\sin \Delta q_t = \sin \Delta q_{t-1} + h^2 r_\theta(q_t)$$
$$q_{t+1} = q_t + \Delta q_t$$

▶▶ Allows us to define dynamics on a manifold

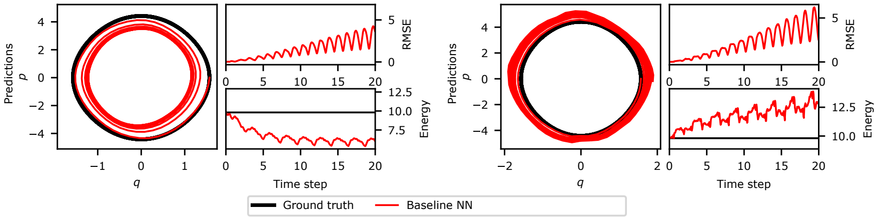
Learning from Noisy Data: Pendulum



Pendulum System. **Left:** 150 observations; **Right:** 750 observations.

Sæmundsson et al. (arXiv:1910.09349): *Variational Integrator Networks for Physically Meaningful Embeddings*

Learning from Noisy Data: Pendulum

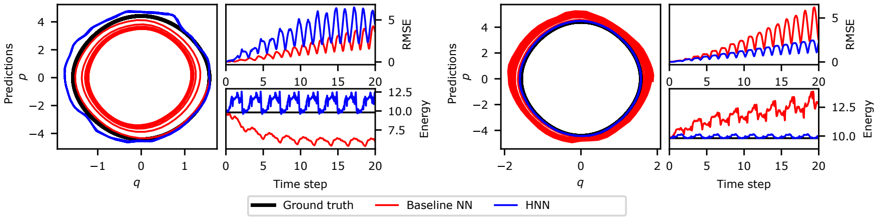


Pendulum System. **Left:** 150 observations; **Right:** 750 observations.

- Baseline neural network: Dissipates/adds energy for low and moderate data

Sæmundsson et al. (arXiv:1910.09349): *Variational Integrator Networks for Physically Meaningful Embeddings*

Learning from Noisy Data: Pendulum

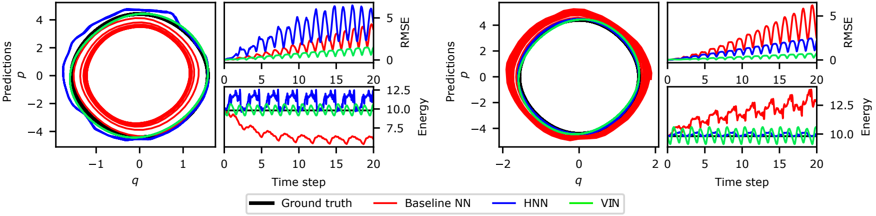


Pendulum System. **Left:** 150 observations; **Right:** 750 observations.

- Baseline neural network: Dissipates/adds energy for low and moderate data
- Hamiltonian neural network (Greydanus et al., 2019): Overfits in low-data regime

Sæmundsson et al. (arXiv:1910.09349): *Variational Integrator Networks for Physically Meaningful Embeddings*

Learning from Noisy Data: Pendulum

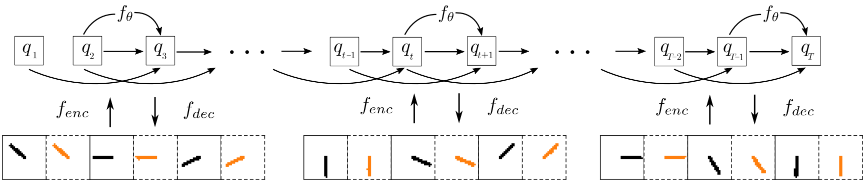


Pendulum System. **Left:** 150 observations; **Right:** 750 observations.

- Baseline neural network: Dissipates/adds energy for low and moderate data
- Hamiltonian neural network (Greydanus et al., 2019): Overfits in low-data regime
- Variational integrator network: Conserves energy and generalizes better in both regimes

Sæmundsson et al. (arXiv:1910.09349): *Variational Integrator Networks for Physically Meaningful Embeddings*

Learning from Pixel Data

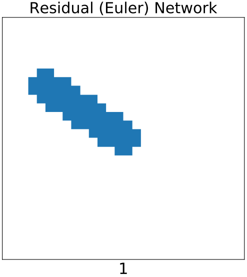


- VIN within variational auto-encoder (VAE) setup:
  - Learn physical system in lower-dimensional latent space
  - Use VIN for long-term forecasting

▶▶ Exploit geometry of the problem for system identification and forecasting

Sæmundsson et al. (arXiv:1910.09349): *Variational Integrator Networks for Physically Meaningful Embeddings*

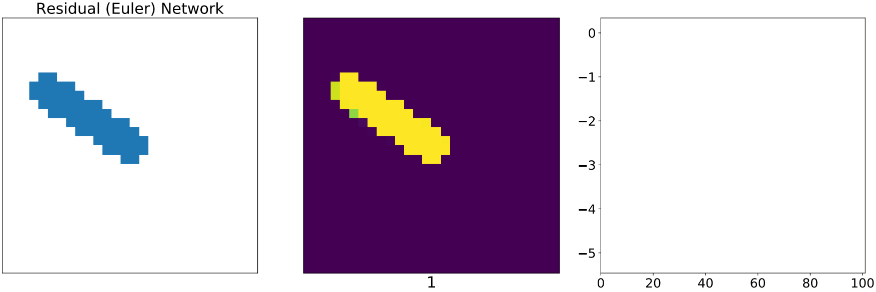
Learning from Pixel Data: Forecasting



- Observations: 28×28 pixel images of pendulum
- Training data: 40 images

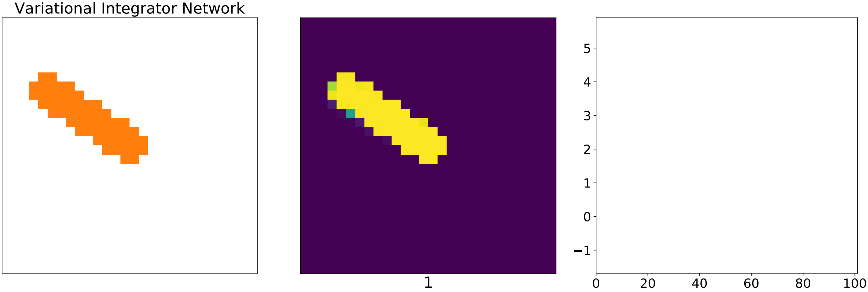
Sæmundsson et al. (arXiv:1910.09349): *Variational Integrator Networks for Physically Meaningful Embeddings*

Learning from Pixel Data: Forecasting



- Observations: 28×28 pixel images of pendulum
- Training data: 40 images
- Dynamic VAE: Forecasting is not meaningful

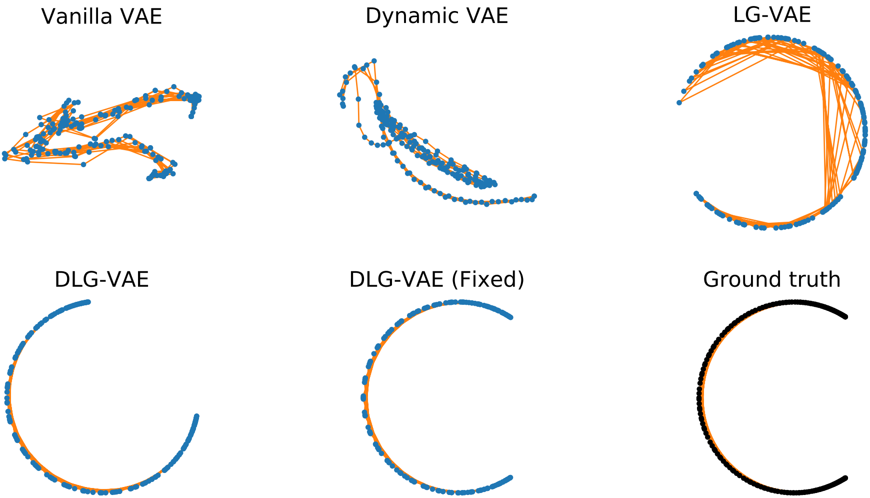
Sæmundsson et al. (arXiv:1910.09349): *Variational Integrator Networks for Physically Meaningful Embeddings*



- Observations: 28×28 pixel images of pendulum
- Training data: 40 images
- Dynamic VAE: Forecasting is not meaningful
- DLG-VAE: Physically meaningful long-term forecasts in latent and observation space

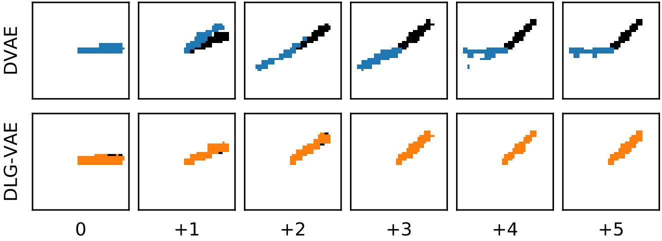
Sæmundsson et al. (arXiv:1910.09349): *Variational Integrator Networks for Physically Meaningful Embeddings*

Learning from Pixel Data: Latent Embeddings



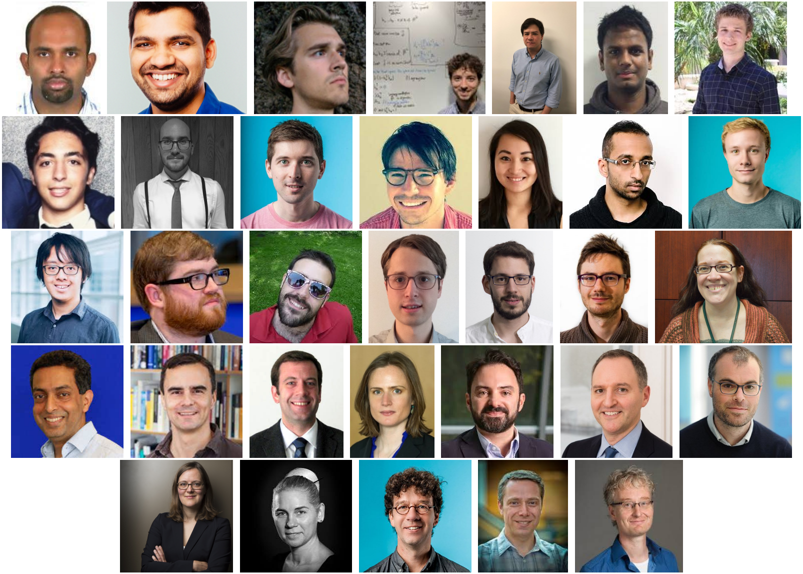
Sæmundsson et al. (arXiv:1910.09349): *Variational Integrator Networks for Physically Meaningful Embeddings*

Summary (3)

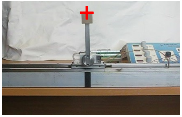


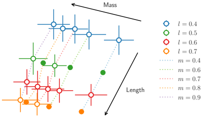
- Variational integrator networks to encode physics and geometric structure ⏩ Interpretability
- Data-efficient learning and physically meaningful long-term forecasts

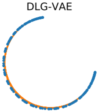
Team and Collaborators



Wrap-up

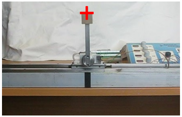


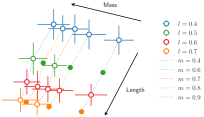


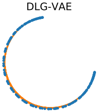


- **Data efficiency** is a practical challenge for autonomous robots
- Three pillars of data-efficient machine learning
  1. Model-based reinforcement learning with learned probabilistic models for fast learning from scratch
  2. Meta learning using latent variables to generalize knowledge to new situations
  3. Incorporation of structural priors for learning physically meaningful predictive models

Wrap-up







- **Data efficiency** is a practical challenge for autonomous robots
- Three pillars of data-efficient machine learning
  1. Model-based reinforcement learning with learned probabilistic models for fast learning from scratch
  2. Meta learning using latent variables to generalize knowledge to new situations
  3. Incorporation of structural priors for learning physically meaningful predictive models

ありがとうございました

References I

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

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

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

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

[10] 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.

[11] S. Greydanus, M. Dzamba, and J. Yosinski. Hamiltonian Neural Networks. In *Advances in Neural Information Processing Systems*, 2019.

[12] D. Jimenez Rezende, S. Mohamed, and D. Wierstra. Stochastic Backpropagation and Variational Inference in Deep Latent Gaussian Models. In *Proceedings of the International Conference on Machine Learning*, 2014.

References II

[13] D. P. Kingma and M. Welling. Auto-Encoding Variational Bayes. In *Proceedings of the International Conference on Learning Representations*, 2014.

[14] 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.

[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] S. Sæmundsson, A. Terenin, K. Hofmann, and M. P. Deisenroth. Variational Integrator Networks for Physically Meaningful Embeddings. In *arXiv:1910.09349*, 2019.

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

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

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

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

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

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

$$\boldsymbol{\beta} := (\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1}\boldsymbol{y}$$ ▶▶ 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