Towards physically structured probabilistic reinforcement learning

Alexander Terenin
Imperial College London

Joint work with Steindór Sæmundsson, James T. Wilson, Viacheslav Borovitskiy, Peter Mostowsky, Katja Hoffmann, and Marc Deisenroth

Talk for University College London

April 29th, 2020

HTTPS://AVT.IM/ · @AVT_IM

Imperial College London

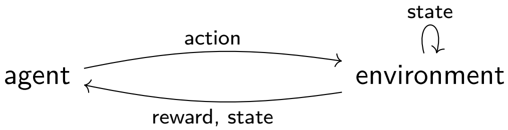
UCL

Reinforcement learning

An agent interacts with an environment in discrete time

At each time step

- Agent chooses action
- Environment changes to a new state
- Agent receives reward from environment



Goal: maximize reward

Continuous control

Continuous-time: controlled differential equations

$$\dot{x}(t) = f(x(t), u(x))$$

Goal: find control map $u : X \to U$ minimizing the path integral

$$\int_0^T c(x(t), u(t)) \mathrm{d}t + c(x(T))$$

f: unknown

Model-based RL: learn a model of f and act accordingly

(possibly taking uncertainty into account via Bayesian methods)

Geometric control and reinforcement learning

Robots are mechanical systems satisfying the laws of physics

Hence, letting $x = (q, p)$, the CDE

$$\dot{x}(t) = f(x(t), u(x))$$

carries the structure of Hamilton's equations

$$\dot{q}(t) = \frac{\partial H}{\partial p} \qquad \dot{p}(t) = -\frac{\partial H}{\partial q} + F$$

Our program: use this structure to make RL more data-efficient

- Formulate theory in the language of *geometry and mechanics*
- Motivates Bayesian theory on *Riemannian manifolds*

Variational integrator networks for physically structured embeddings

Steindór Sæmundsson, Alexander Terenin,
Katja Hoffmann, and Marc Deisenroth

Neural ODEs for embedding dynamical systems

Goal: learn an embedding of a dynamical system observed as pixels

Neural ODEs: view RNNs and ResNets as discretizations of ODEs

$$\dot{x}(t) = f(x(t), t)$$

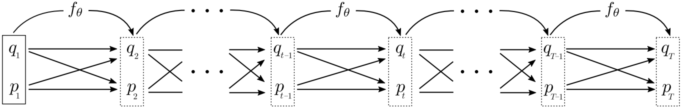
Idea: rather than a free-form ODE, work with Hamilton's equations

$$\dot{q}(t) = \frac{\partial H}{\partial p} \qquad\qquad \dot{p}(t) = -\frac{\partial H}{\partial q}$$

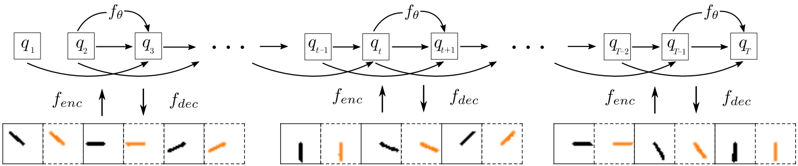
Variational integrator networks

Discretize the variational principle

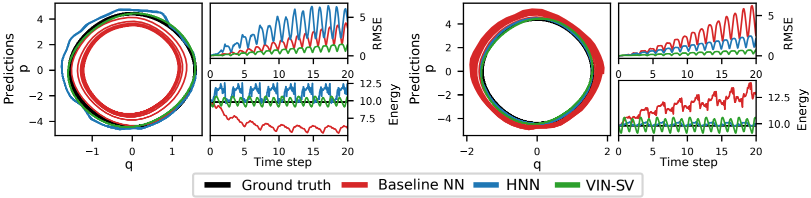
$\implies$ new network architectures!



Learn from pixels using a VAE

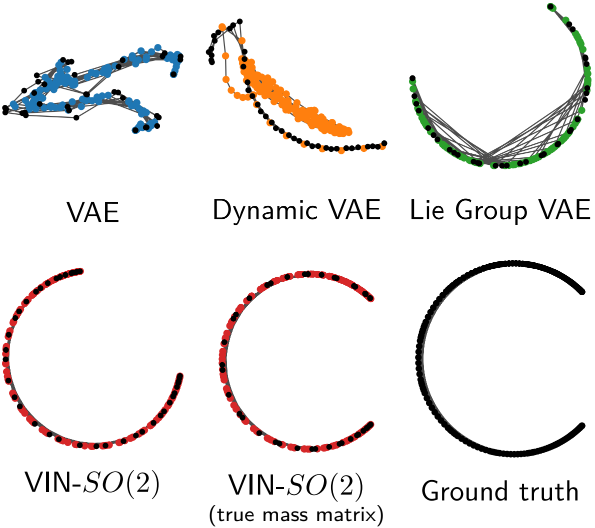


Conservation laws



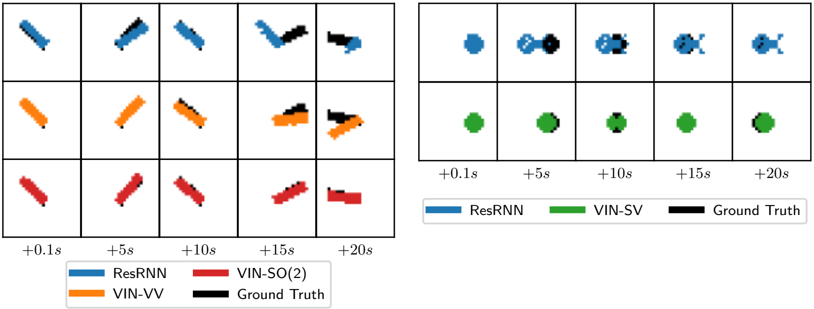
VINs conserve phase volume and momentum *exactly*
and conserve energy much better than free-form RNNs

Latent state space



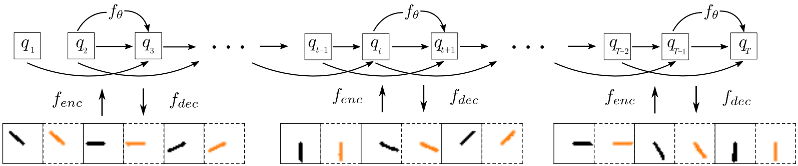
Continuous latent state space in space and time

Long-term forecasting



Better long-term accuracy in small-data settings

A physically structured architecture



✓ data-efficiency
✓ interpretability
✓ latent space behavior
✓ long-term generalization
✓ well-understood mathematics

☒ can be slightly less expressive
☒ external forces are tricky (working on this now)

Efficiently sampling functions from Gaussian process posteriors

James T. Wilson,[*] Viacheslav Borovitskiy,[*] Alexander Terenin,[*]
Peter Mostowsky,[*] and Marc Deisenroth




[*]Equal contribution

Probabilistic models for reinforcement learning

Controlled differential equations

$$\dot{x}(t) = f(x(t), u(x))$$

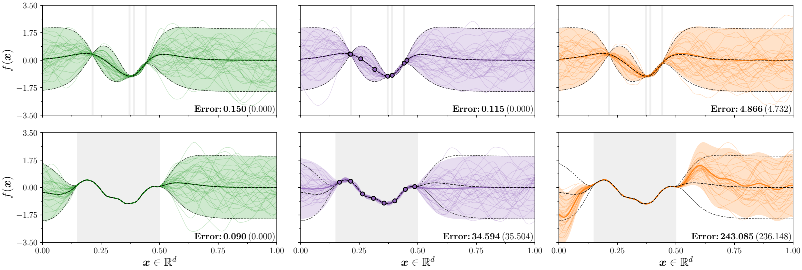
Time-discretization $\implies$ rollouts

$$x_{t+1} = x_t + \Delta f(x_t, u(x_t))$$
$$x_{t+2} = x_{t+1} + \Delta f(x_{t+1}, u(x_{t+1}))$$

✓ Gaussian processes: excellent data-efficiency (PILCO)

☒ Gaussian process rollouts: $O(T^3)$

This work: address this without sacrificing accuracy


Error: 0.150 (0.000)
Error: 0.115 (0.000)
Error: 4.866 (4.732)
Error: 0.090 (0.000)
Error: 34.594 (35.504)
Error: 243.085 (236.148)
$f(\boldsymbol{x})$
$\boldsymbol{x} \in \mathbb{R}^d$

Key idea: Matheron's update rule

$$(f \mid \boldsymbol{y})(\cdot) \overset{\mathrm{d}}{=} f(\cdot) + \mathbf{K}_{(\cdot)\boldsymbol{x}}\mathbf{K}_{\boldsymbol{x}\boldsymbol{x}}^{-1}(\boldsymbol{y} - \boldsymbol{f}_{\boldsymbol{x}})$$

Why? For $\begin{bmatrix}\boldsymbol{x}_1 \\ \boldsymbol{x}_2\end{bmatrix} \sim \mathrm{N}\left(\begin{bmatrix}\boldsymbol{\mu}_2 \\ \boldsymbol{\mu}_2\end{bmatrix}, \begin{bmatrix}\mathbf{\Sigma}_{11} & \mathbf{\Sigma}_{12} \\ \mathbf{\Sigma}_{21} & \mathbf{\Sigma}_{22}\end{bmatrix}\right)$ we have

$$(\boldsymbol{x}_1 \mid \boldsymbol{x}_2 = \boldsymbol{u}) \overset{\mathrm{d}}{=} \boldsymbol{x}_1 + \mathbf{\Sigma}_{12}\mathbf{\Sigma}_{22}^{-1}(\boldsymbol{u} - \boldsymbol{x}_2)$$

Path-wise sampling with sparse GPs

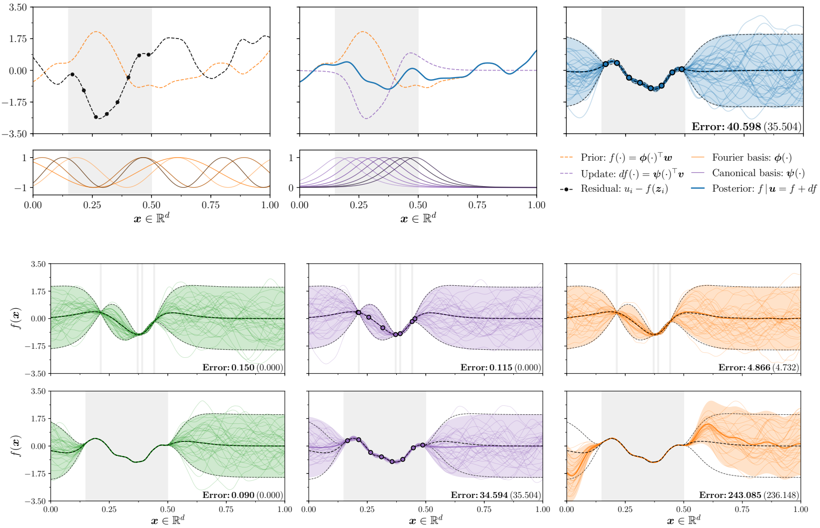
$$\underbrace{(f \mid \boldsymbol{y})(\cdot)}_{\text{posterior}} \overset{\mathrm{d}}{=} \underbrace{f(\cdot)}_{\text{prior}} + \underbrace{\mathbf{K}_{(\cdot)\boldsymbol{x}}\mathbf{K}_{\boldsymbol{x}\boldsymbol{x}}^{-1}(\boldsymbol{y} - \boldsymbol{f}_{\boldsymbol{x}})}_{\text{update}}$$

Prior term: discretize with random Fourier features

Data term: approximate with sparse GPs

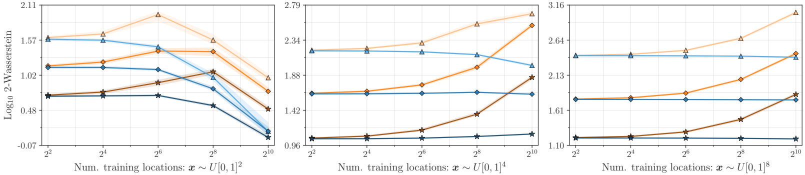
$$\underbrace{(f \mid \boldsymbol{y})(\cdot)}_{\substack{\text{approximate}\\\text{posterior}}} \overset{\mathrm{d}}{\approx} \underbrace{\sum_{i=1}^{\ell} w_i \phi_i(\cdot)}_{\substack{\text{RFF basis for}\\\text{stationary prior}}} + \underbrace{\sum_{i=1}^{m} v_i k(\cdot, z_i)}_{\substack{\text{canonical basis}\\\text{for sparse update}}} \qquad \boldsymbol{v} = \mathbf{K}_{\boldsymbol{z}\boldsymbol{z}}^{-1}(\boldsymbol{u} - \boldsymbol{\Phi}^\top \boldsymbol{w})$$

Visualizing decoupled sample paths



Error analysis

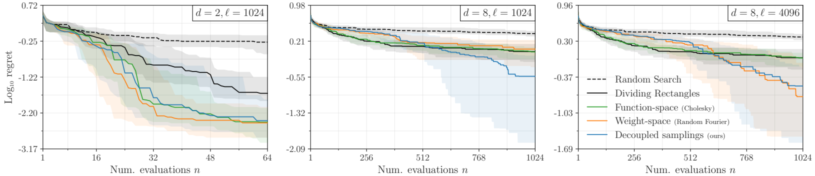
$$\underbrace{W_{2,L^2(\mathcal{X})}(f^{(d)}, f \mid \boldsymbol{y})}_{\text{total approximation error}} \leq \underbrace{W_{2,L^2(\mathcal{X})}(f^{(s)}, f \mid \boldsymbol{y})}_{\text{error in sparse posterior}} + \underbrace{C\, W_{2,L^2(\mathcal{X})}(f^{(w)}, f)}_{\text{error in approximate prior}}$$



Empirical Wasserstein error smaller than in RFF

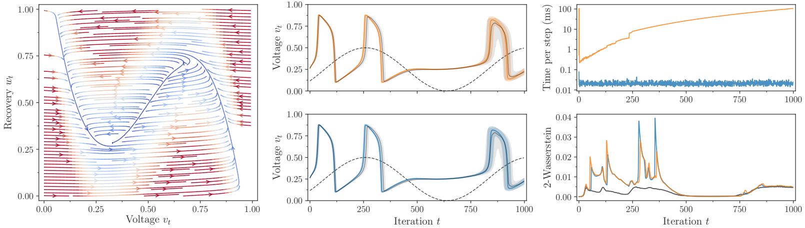
$$C = \sqrt{2\,\mathrm{diam}(\mathcal{X})^d \left(1 + \|k\|_\infty^2 \|\mathbf{K}_{\boldsymbol{zz}}^{-1}\|_{L(\ell^\infty;\ell^2)}^2\right)}$$

Thompson sampling



Improved performance owing to smaller error

FitzHugh-Nagumo model neuron dynamical system



Significantly more efficient time-stepping

Towards geometric PILCO: concluding remarks

Controlled Hamilton's equations

$$\dot{q}(t) = \frac{\partial H}{\partial p} \qquad \dot{p}(t) = -\frac{\partial H}{\partial q} + F_u$$

✓ time-step using VINs (or other symplectic residual RNNs)

✓ use path-wise formula for discretizing GP

- how to properly handle external and contact forces?
- how to define GPs on Riemannian manifolds? (current idea: SPDEs)
- does geometry and mechanics help on the policy side?

Concluding remarks

Thank you for your attention!

HTTPS://AVT.IM/

@AVT_IM

Imperial College London

UCL

S. Sæmundsson, A. Terenin, K. Hofmann, M. P. Deisenroth. Variational integrator networks for physically structured embeddings. Artificial Intelligence and Statistics, 2020. Available at: HTTPS://ARXIV.ORG/ABS/1910.09349

J. T. Wilson,* V. Borovitskiy,* A. Terenin,* P. Mostowsky,* M. P. Deisenroth. Efficiently sampling functions from Gaussian process posteriors, 2020. *Equal contribution. Available at: HTTPS://ARXIV.ORG/ABS/2002.09309.