Recognizing Generalized Gradient Dynamics by Means of Machine Learning

Vojtěch Votruba, Michal Pavelka

Mathematical Institute, Charles University

June 9, 2025

Introduction

- Motivation: distilling consistent models from time series
- Combining generalized gradient dynamics with deep learning
- Related work:
  - *Variational Onsager Neural Networks (VONNs): A thermodynamics-based variational learning strategy for non-equilibrium PDEs* — Reina et al, 2022
  - *Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations* — M. Raissi et al., 2019

Outline

- Gradient Dynamics
- Deep Learning
- Employed design
- Results:
  - Overdamped particle
  - Chemical reactions
  - Fickian diffusion
- Future outlook

Generalized Gradient Dynamics

- Grmela, Öttinger (in the context of GENERIC)
- Derivation using the Theory of Large Deviations (Mielke, Peletier, Renger, 2015)
- Let $\mathbf{x} = (x^1, \dots, x^n)$ be n state variables of a purely dissipative system
- Introduce entropy $S = S(\mathbf{x})$ and the entropic conjugates $\mathbf{x}^* = \frac{\delta S}{\delta \mathbf{x}}$
- Introduce dissipation potential $\Xi(\mathbf{x}, \mathbf{x}^*)$ such that
  1. $\Xi(\mathbf{x}, \mathbf{x}^*) \geq 0$ and $\Xi(\mathbf{x}, \mathbf{0}) = 0$
  2. $\frac{\delta \Xi}{\delta \mathbf{x}^*} \cdot \mathbf{x}^* \geq 0, \quad \forall \mathbf{x}^*$
  3. near $\mathbf{x}^* = \mathbf{0}$ the potential Ξ must be convex

Generalized Gradient Dynamics

- Evolution equations

$$\boxed{\dot{\mathbf{x}} = \left.\frac{\delta \Xi}{\delta \mathbf{x}^*}\right|_{\mathbf{x}^* = \frac{\delta S}{\delta \mathbf{x}}}}$$

- The consequences of imposed restrictions:
  - 2nd law: $\dot{S} = \frac{\delta S}{\delta \mathbf{x}} \cdot \dot{x} = \mathbf{x}^* \cdot \frac{\delta \Xi}{\delta \mathbf{x}^*} \geq 0$
  - End of evolution at $\mathbf{x}^* = \frac{\delta S}{\delta \mathbf{x}} = \mathbf{0}$
- $\iff$ Onsager's principle $\frac{\delta S}{\delta \mathbf{x}} = \frac{\delta \Xi^*}{\delta \dot{\mathbf{x}}}$, through Legendre transform

Deep Learning

- $\in$ Machine learning
- Training and designing deep neural networks
- Feedforward network: $\mathbf{\Lambda} : \mathbb{R}^N \to \mathbb{R}^M$

$$\mathbf{\Lambda}(\mathbf{x}; \theta) := \varphi_n(W_n \cdot (\varphi_{n-1}(...\varphi_1(W_1 \cdot \mathbf{x} + \mathbf{b}_1) + \mathbf{b}_2)) + \mathbf{b}_n)$$

- $\theta = \textit{weights } \{W_i\}_{i=1}^n + \textit{biases } \{b_i\}_{i=1}^n$. φ are *activation functions*
- Why use neural nets? They are **universal function approximators**

Deep Learning

- Training := optimizing a loss function $L : \mathbb{R}^M \to [0, +\infty)$

$$L(\theta) \propto ||\mathbf{y}^{(\text{true})}(\mathbf{x}) - \mathbf{\Lambda}(\mathbf{x}, \theta)||^2 \implies \frac{\partial L}{\partial \theta} = \mathbf{0}.$$

- Automatic differentiation engine
- First-order and Second-order methods
  - SGD, **Adam**, RMSProp
  - Newton's method, conjugate gradients, BFGS, **L-BFGS**

Employed Design

- Variational Onsager Neural Networks
  - Dissipation neural net – Partially convex
  - Entropy neural net – Concave (in our case)
  - Foundation in Amos, Xu, Kolter, 2017
  - Using reparameterization to fulfill additional restrictions
- Data generation via numerical integration
- Calculating $\frac{\delta S}{\delta \mathbf{x}}, \frac{\delta \Xi}{\delta \mathbf{x}^*}$ through automatic differentiation

$$L \propto ||\dot{\mathbf{x}}^{(\mathrm{nn.})} - \dot{\mathbf{x}}^{(\mathrm{true})}||^2 = \left|\left| \left.\frac{\partial \Xi^{(\mathrm{nn.})}}{\partial \mathbf{x}^*}\right|_{\mathbf{x}^* = \frac{\partial S^{(\mathrm{nn.})}}{\partial \mathbf{x}}} - \dot{\mathbf{x}}^{(\mathrm{true})} \right|\right|^2$$

Overdamped Particle

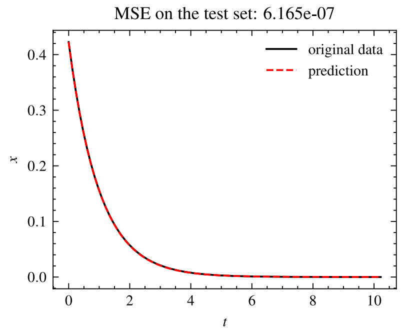
- Evolution equation $\dot{\mathbf{x}} = -\gamma \mathbf{x}$, where $\gamma > 0$
- This implies quadratic dissipation potential and entropy

$$S(\mathbf{x}) = -\frac{1}{2}||\mathbf{x}||^2, \quad \Xi(\mathbf{x}, \mathbf{x}^*) = \frac{1}{2}\gamma||\mathbf{x}||^2$$

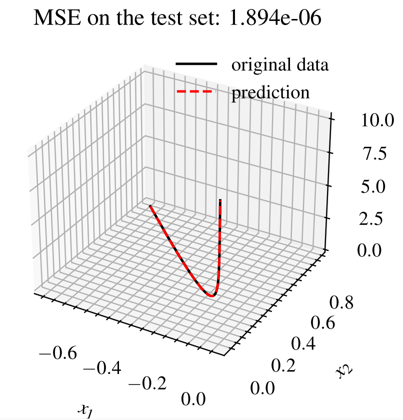
- Evidently, there is non-uniqueness
  - Addition $S := S + C, \quad C \in \mathbb{R}$
  - Multiplication $S := S/C$ and $\Xi := C\Xi$, where $C > 0$

Overdamped Particle

- Prediction results



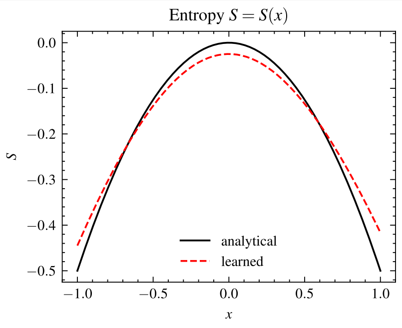
(a) 1D



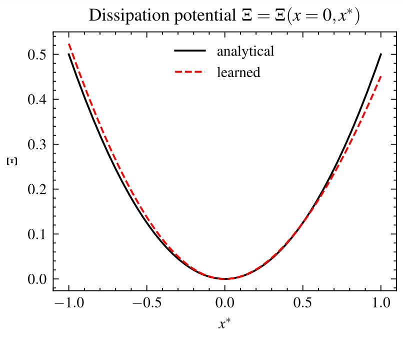
(b) 2D

Overdamped Particle

- Learned functions 1D



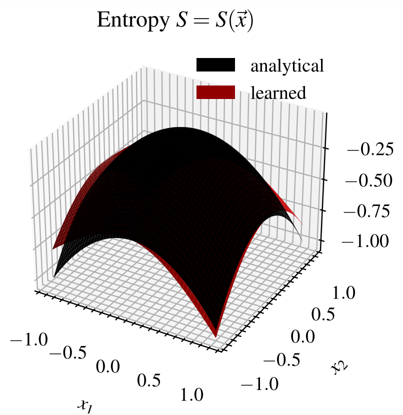
(c) Entropy



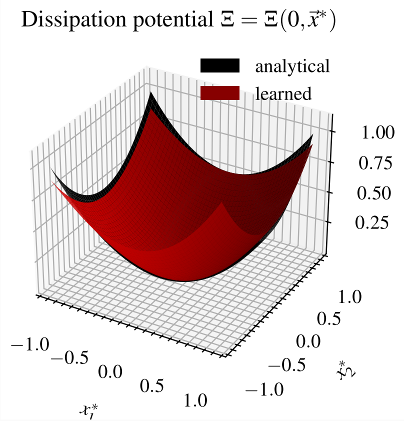
(d) Dissipation Potential

Overdamped Particle

- Learned functions 2D



(e) Entropy



(f) Dissipation Potential

Chemical Reactions

- System of N chemical reactions with the concentrations $\mathbf{c} = (c^1, \dots, c^M)$
- Each reaction looks like

$$\alpha_1^{(n)} \mathbb{A}_1^{(n)} + \ldots + \alpha_M^{(n)} \mathbb{A}_M^{(n)} \rightleftharpoons \beta_1^{(n)} \mathbb{A}_1^{(n)} + \ldots + \beta_M^{(n)} \mathbb{A}_M^{(n)}$$

- Stoichiometric matrix $\nu_{mn} := \beta_m^{(n)} - \alpha_m^{(n)}$
- Kinetics vector $\Gamma_n = k_f^{(n)} \prod_{m=1}^M c_m^{\alpha_m^{(n)}} - k_b^{(n)} \prod_{m=1}^M c_m^{\beta_m^{(n)}}$
- Law of mass action: $\dot{\mathbf{c}} = \nu \cdot \mathbf{\Gamma}$

Chemical Reactions

- Through generalized gradient dynamics:
  - Entropy: $S(\mathbf{c}) = -\sum_{m=1}^{M} c_m(\ln c_m + Q_m)$
  - Dissipation: $\Xi(\mathbf{c}, \mathbf{c}^*) = \sum_{n=1}^{N} W_n(\mathbf{c})(\cosh(X_n/2) - 1)$
  - Where $X_n = -\sum_{m=1}^{M} c_m^* \nu_{mn}$, $W_n = W_0 \sqrt{\prod_{m=1}^{M} c_m^{|\nu_{mn}|}}$
  - Then if we set $k_f^{(n)} := \frac{W_0}{4} \exp\left(-\sum_m \frac{(Q_m+1)\nu_{mn}}{2}\right)$, $k_b^{(n)} := \frac{W_0}{4} \exp\left(\sum_m \frac{(Q_m+1)\nu_{mn}}{2}\right)$
- $\implies$ We obtain the right equations

Chemical Reactions

- Again, non-uniqueness.
- Example reaction $A \rightleftharpoons B$
  - According to the law of mass action: $\dot{c}_A = c_B - c_A, \quad \dot{c}_B = c_A - c_B.$
  - This can be clearly satisfied by

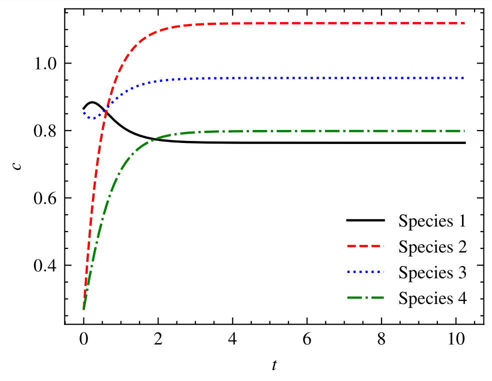
$$S(\mathbf{c}) = -\frac{1}{2}||\mathbf{c}||^2, \quad \Xi(\mathbf{c}, \mathbf{c}^*) = \frac{1}{2}\mathbf{c}^* \cdot \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \cdot \mathbf{c}^*$$

- Therefore, fix one quantity to obtain the other

Chemical Reactions

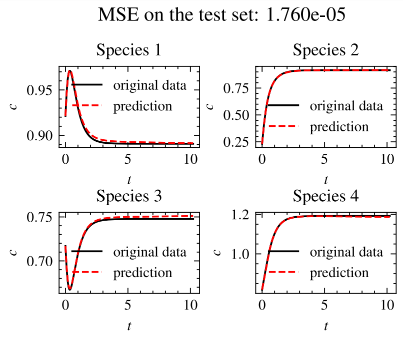
- Example system:

$$A + 2B \rightleftharpoons C$$
$$C + D \rightleftharpoons A$$

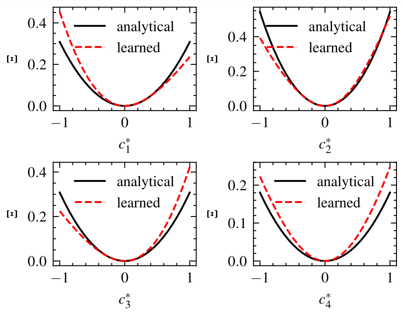


Chemical Reactions

- Prescribed entropy



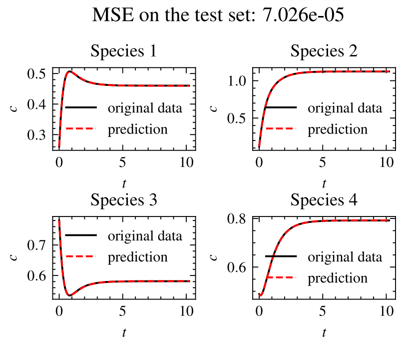
(g) Evolution for each species



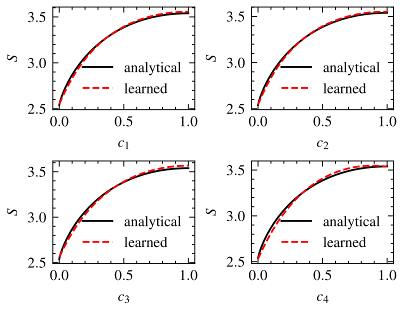
(h) Dissipation potential

Chemical Reactions

- Prescribed dissipation



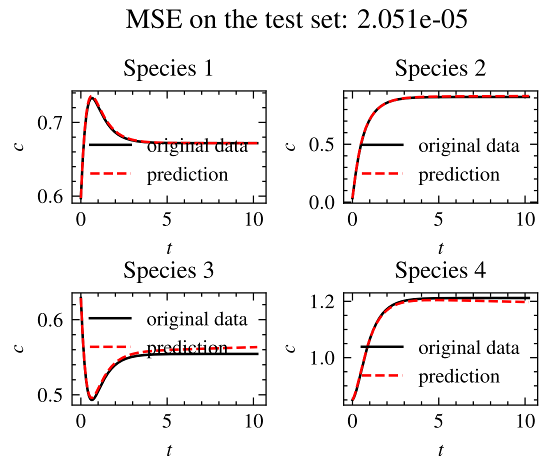
(i) Evolution for each species



(j) Entropy

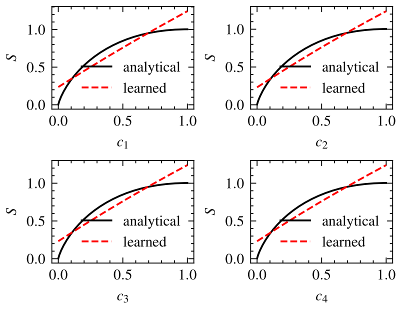
Chemical Reactions

- Without prescription

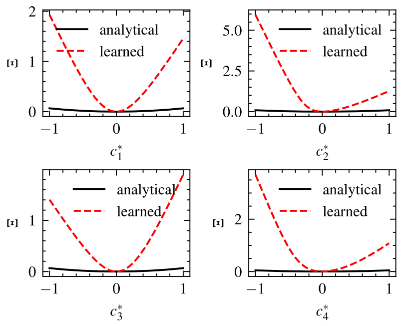


Chemical Reactions

- Without prescription



(k) Entropy



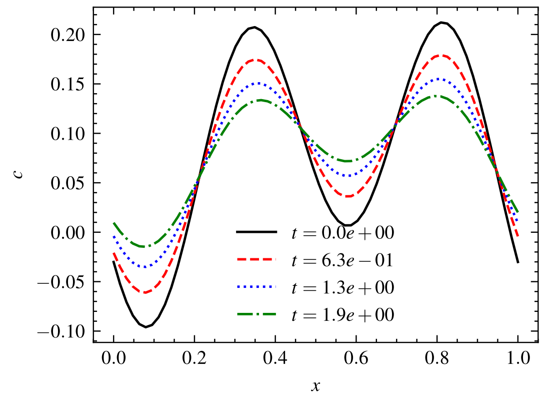
(l) Dissipation potential

Fickian Diffusion

- State vector $\rightarrow$ State function: $c(\mathbf{r})$
- Linear diffusion equation: $\dot{c}(\mathbf{r}) = \nabla \cdot (D\nabla c(\mathbf{r})) = D\Delta c(\mathbf{r})$
- Through generalized gradient dynamics:
  - Entropy $S[c(\mathbf{r})] = -\int c \log(c)\, \mathrm{d}^3 r$
  - Dissipation $\Xi[c(\mathbf{r}), \nabla c^*(\mathbf{r})] = \frac{1}{2}\int Dc||\nabla c^*||^2\, \mathrm{d}^3 r$
- 1D discretization $\Delta c \approx \frac{c_{i+1} - 2c_i + c_{i-1}}{h^2}$, where h is the grid resolution
- Training on Fourier modes with PBC, evaluation on 3 different initial conditions

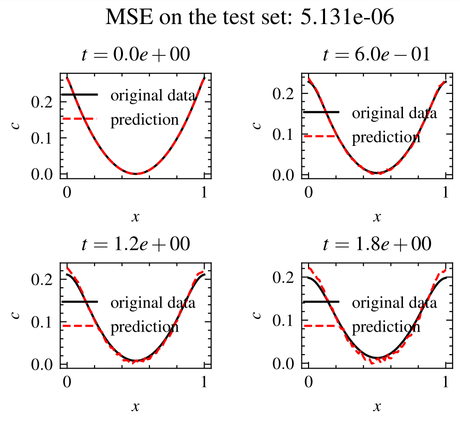
Fickian Diffusion

- Example training trajectory



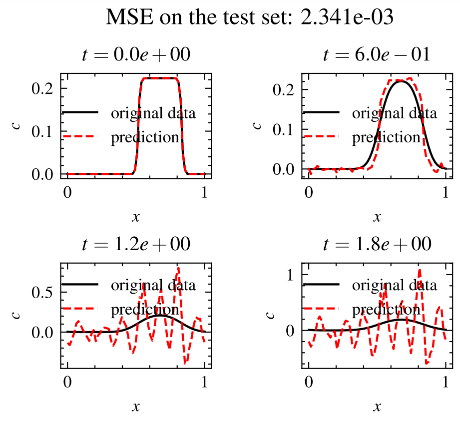
Fickian Diffusion

- Polynomial initial conditions



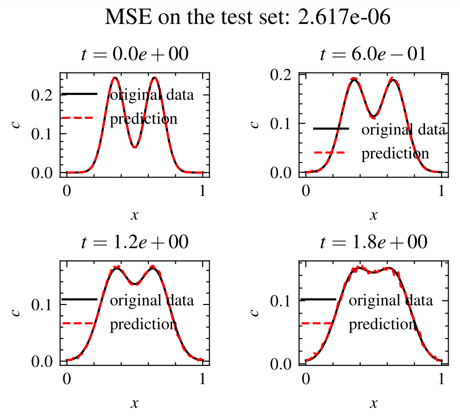
Fickian Diffusion

- Discontinuous initial conditions



Fickian Diffusion

- Gaussian spikes initial conditions



Future Outlook

- Integrating with reversible networks, i.e., Deep Poisson Neural Networks
  - Building an entire GENERIC coupling $\dot{\mathbf{x}} = \{\mathbf{x}, H\} + \frac{\delta \Xi}{\delta \mathbf{x}^*}\Big|_{\mathbf{x}^* = \frac{\delta S}{\delta \mathbf{x}}}$
- Extending to functional spaces via Neural Operators (Kovachki, Li et al, 2023)
- Very open to collaboration!
  - Vojtěch Votruba, votruba@karlin.mff.cuni.cz
  - Michal Pavelka, pavelka@karlin.mff.cuni.cz

Thank you for your attention.