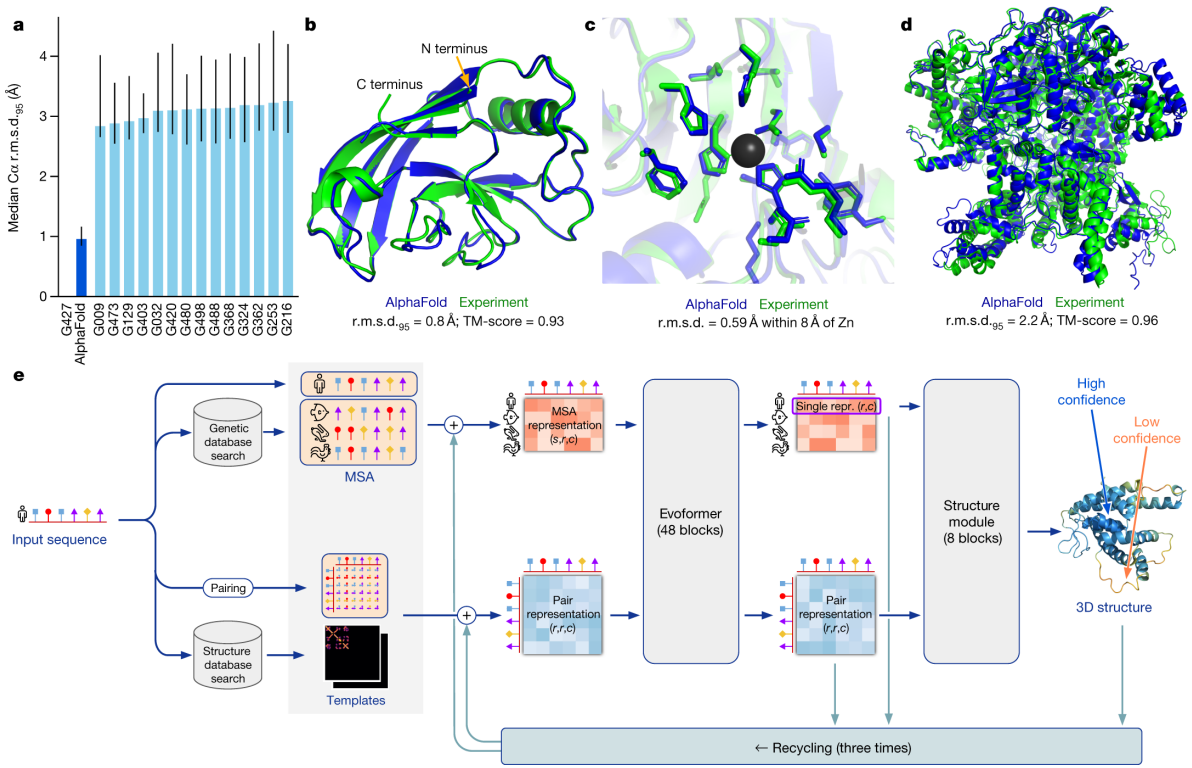
On distribution mismatch in Data-driven Scientific Computing

Jiaxi Zhao

16th August, 2022

Data-driven scientific computing

Data-driven method is becoming a prevalent surrogate model in mathematical modeling and scientific computing, due to its power on approximation the data in complicated and high dimensional setting.



Data-driven scientific computing: Three Categories

1. **100% data-driven**: Physics-informed neural networks (PINN), Fourier neural operator, DeepONet.
2. **50 % Numerical + 50 % data-driven**: Machine learning turbulence modeling, DeepPotential, Quasipotential.
3. **Discovering physics law from data**: Symbolic regression for conservation law, Principle component analysis for phase transition, cluster algorithm for space-time classification.

PINN's formulation

Numerical PDE $\Longrightarrow$ **Regression problem**[1] with regularization:

$$\Delta u(x) = f(x) \quad x \in \Omega, \quad u(x) = g(x) \quad x \in \partial\Omega.$$

$$\arg\min_{u} \sum_{i=1}^{N_i} |\Delta u(X_i) - f(X_i)|^2 + \sum_{i=1}^{N_b} |u(Y_i) - g(Y_i)|^2.$$

Satisfying performance for inverse problem!

[1] Raissi, Maziar, Paris Perdikaris, and George E. Karniadakis. "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations." Journal of Computational physics 378 (2019): 686-707.

PINN for Boussinesq equation

$$\partial_t u + u \cdot \nabla u + \nabla p = (0, -\theta), \quad \partial_t \theta + u \cdot \nabla \theta = 0.$$

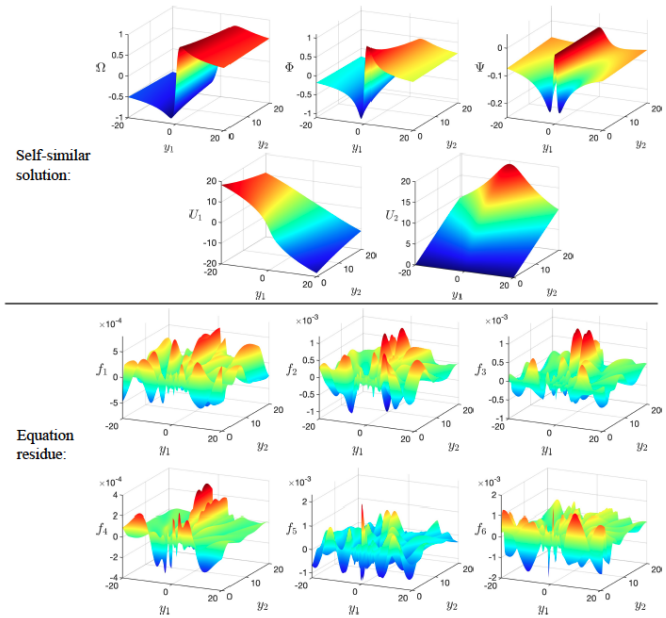


Figure 1: Smooth solution for the 2D Bousinessq equations (2.1) derived by the physics-informed neural network. f_1 to f_6 indicate the residue of the 6 equations defined in (3.3), which are around four orders of magnitude smaller than the output variables of the solution. The inferred value of λ for the smooth solution is $\lambda = 1.95$.

[1] Drivas, Theodore D., and Tarek M. Elgindi. "Singularity formation in the incompressible Euler equation in finite and infinite time." arXiv preprint arXiv:2203.17221 (2022).

Data-driven for Reynolds stresses modeling

$\langle \mathbf{U} \rangle$ = time average of $\mathbf{U}$, $\mathbf{u} = \mathbf{U} - \langle \mathbf{U} \rangle$, Reynolds-averaged Navier-Stokes (RANS) equation reads

$$\begin{aligned} \partial_t \langle \mathbf{U} \rangle + (\langle \mathbf{U} \rangle \cdot \nabla) \langle \mathbf{U} \rangle + \frac{\partial \langle \mathbf{u} u_j \rangle}{\partial x_j} &= -\frac{1}{\rho} \nabla p + \nu \Delta \langle \mathbf{U} \rangle , \\ \nabla \cdot \langle \mathbf{U} \rangle &= 0. \end{aligned} \tag{1}$$

Extra term: $\langle \mathbf{u} u_j \rangle$ (Reynolds stress). **The equation is not closed**!!

Ling et al.[2]proposed a data-driven surrogate model to estimate the Reynolds stresses.

[2]Ling, Julia, Andrew Kurzawski, and Jeremy Templeton. "Reynolds averaged turbulence modelling using deep neural networks with embedded invariance." Journal of Fluid Mechanics 807 (2016): 155-166.

Improve the prediction of flow separation

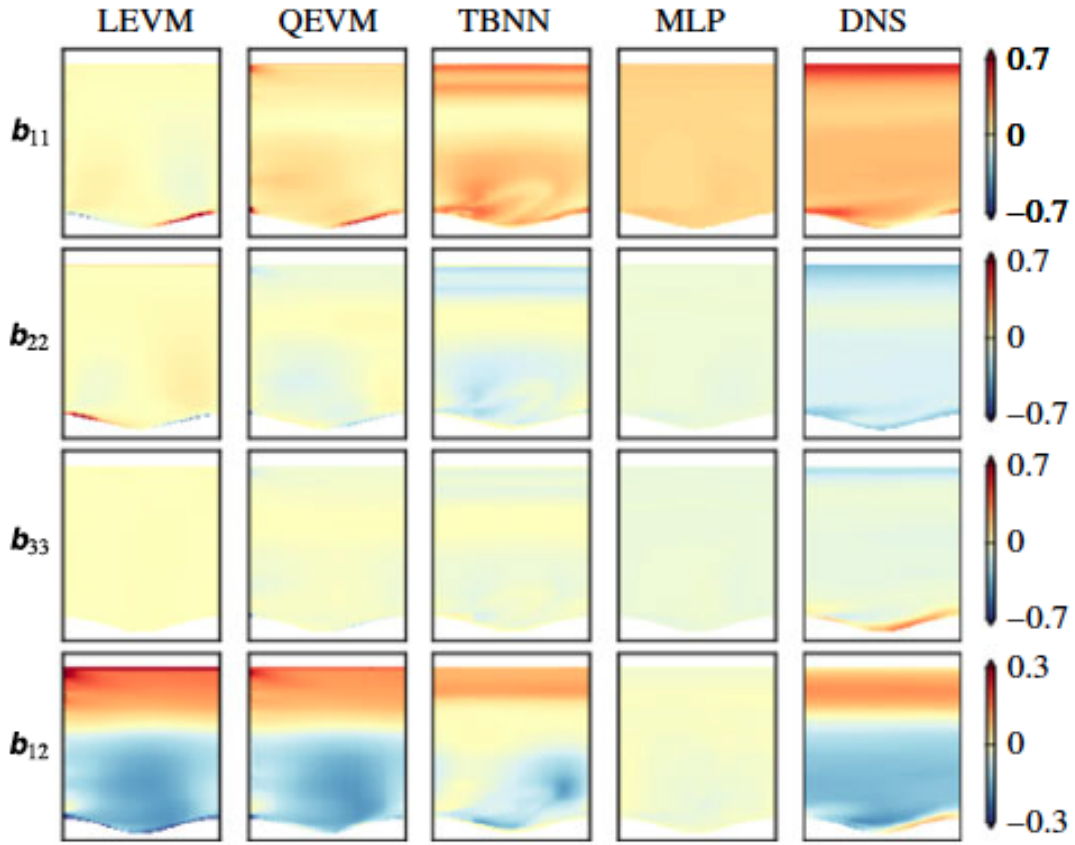


FIGURE 3. Predictions of Reynolds stress anisotropy $\boldsymbol{b}$ tensor on the wavy wall test case. The columns show the predictions of the LEVM, QEVM, TBNN and MLP models. The true DNS anisotropy values are shown in the right-most column for comparison.

Data-driven scientific computing: Three Categories

1. **100% data-driven**: Physics-informed neural networks (PINN), Fourier neural operator, DeepONet.
2. **50 % Numerical + 50 % data-driven**: Machine learning turbulence modeling, DeepPotential, Quasipotential.
3. **Discovering physics law from data**: Symbolic regression for conservation law, Principle component analysis for phase transition.

Framework for the second approach

Simulating the dynamics:

$$\partial_t \mathbf{x} = \mathcal{L}(\mathbf{x}, \mathbf{u}, t), \quad \mathbf{x} \in \mathcal{X}, \mathbf{u} \in \mathcal{U}, \mathcal{L} : \mathcal{X} \times \mathcal{U} \times \mathbb{R}_+ \to T\mathcal{X},$$
$$\mathbf{u} = \phi(\mathbf{x}, t), \quad \phi : \mathcal{X} \times \mathbb{R}_+ \to \mathbf{u}.$$

1. $\mathcal{L}$ is known, possibly non-linear.
2. ϕ is un-known.
3. A set of data pairs
$\{(\mathbf{x}_1, \mathbf{u}_1, t_1), (\mathbf{x}_2, \mathbf{u}_2, t_2), \cdots, (\mathbf{x}_N, \mathbf{u}_N, t_N)\}$.
4. Bench mark algorithm solves the ordinary least square:

$$\arg\min_{\theta} \mathbb{E} \left\| \mathbf{u} - \phi_\theta(\mathbf{x}, t) \right\|^2.$$

Inner-outer loop: Linear outer loop

In many scientific computing examples, further simplification:

$$\begin{aligned} \mathbf{X}_{k+1} &= A\mathbf{X}_k + B\mathbf{Z}_k, \quad \mathbf{X}_k \in \mathbb{R}^n, \mathbf{Z}_k \in \mathbb{R}^m, \\ \mathbf{Z}_k &= f(\mathbf{X}_k). \end{aligned}$$

The outer loop is known to us and usually has some good numerical properties, i.e. linear, stable, etc, while the inner loop can be complicated.

Inner-outer loop: examples

Example (RANS)

$$\begin{cases} \mathbf{u}\cdot\nabla\mathbf{u} - \nu\nabla^2\mathbf{u} + \nabla p - \nabla\cdot\tau = \mathbf{0}, \\ \nabla\cdot\mathbf{u} = 0. \end{cases}$$

$$\tau = R(\mathbf{u}).$$

Example (Quasi-potential)

$$\begin{aligned} \mathbf{X}_{k+1} &= \mathbf{X}_k - \Delta t \mathbf{Z}_k, \\ \mathbf{Z}_k &= \nabla V_\theta(\mathbf{X}_k) + g_\theta(\mathbf{X}_k). \end{aligned} \tag{2}$$

Imitation learning

This structure also appears in **reinforcement learning** problems, or more specifically: **imitation learning**.

Definition

For a system with transition model $p(x_t|x_{t-1}, u_{t-1})$ *with states* $x \in \mathcal{X}$ *and controls* $u \in \mathcal{U}$*, the imitation learning problem is to leverage a set of demonstrations* $\Xi = \{(x_0, u_0, t_0), (x_1, u_1, t_1), \cdots\}$ *from an expert policy* π^* *to find a policy* $\widehat{\pi}$ *that imitates the expert policy.*

Algorithms in the imitation learning

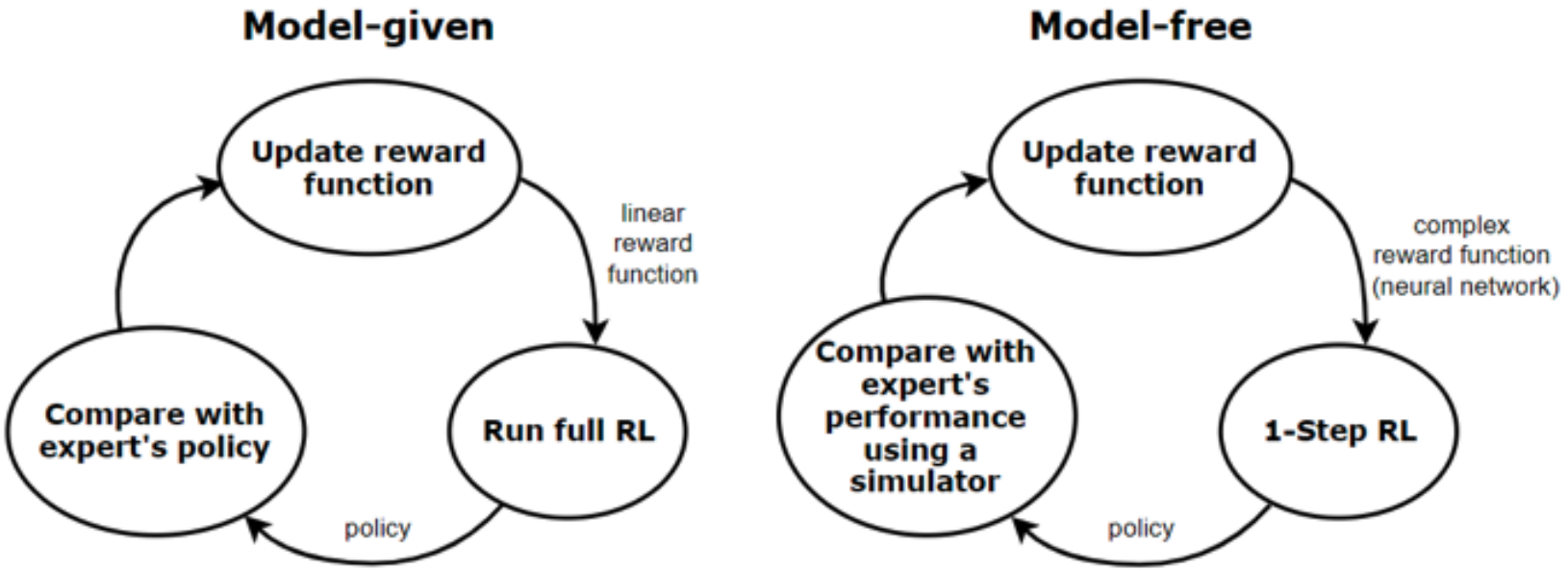
behavior cloning:

$$\arg\min_{\theta} \mathbb{E}_x l(u, \phi(x, \theta)),$$

$$\phi : \mathcal{X} \times \Theta \to \mathcal{U}.$$

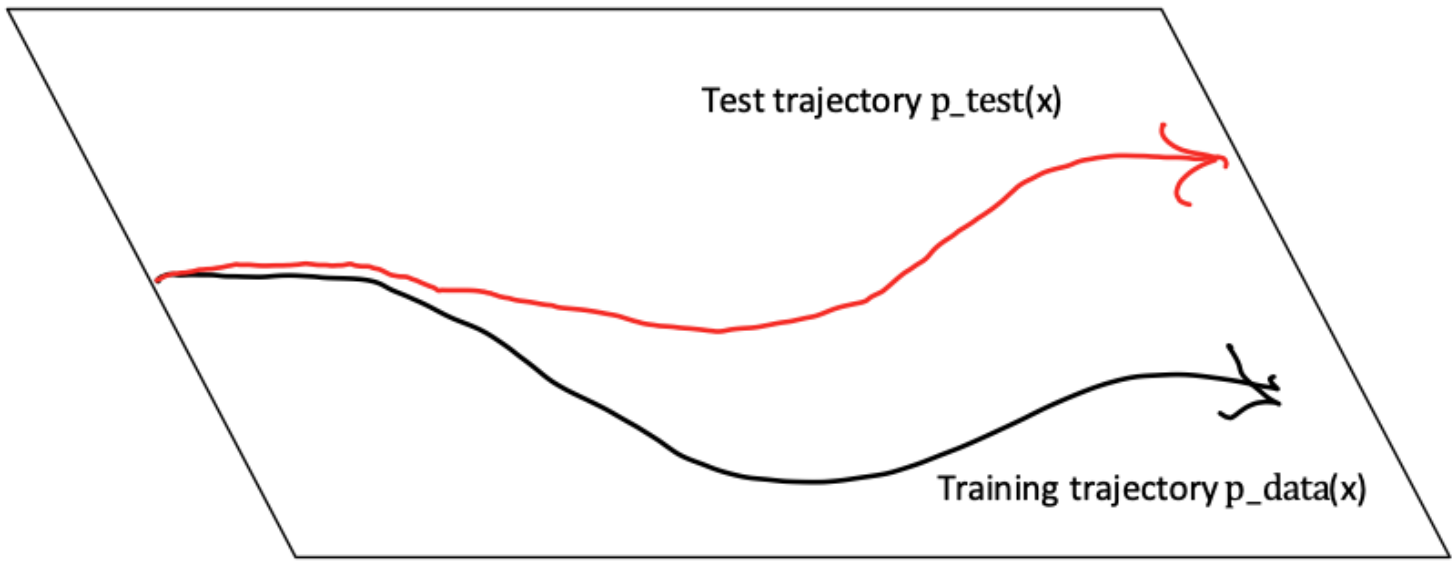
If you take l as 2-norm, this is nothing but **ordinary least square**!

Inverse reinforcement learning: Learn the reward function based on the policy.



Dilemma of data-driven scientific computing

In the data-driven scientific computing, **dynamics structure** can cause **distribution mismatch** between the training and testing data.



Algorithm to mitigate the distribution mismatch

Modified the training dataset to mitigate the distribution shift.

Algorithm 1: DAgger: Dataset Aggregation

Data: π^*

Result: $\hat{\pi}^*$

$\mathcal{D} \leftarrow 0$

Initialize $\hat{\pi}$

for $i = 1$ **to** N **do**

  $\pi_i = \beta_i \pi^* + (1 - \beta_i)\hat{\pi}$

  Rollout policy π_i to sample trajectory $\tau = \{x_0, x_1, \dots\}$

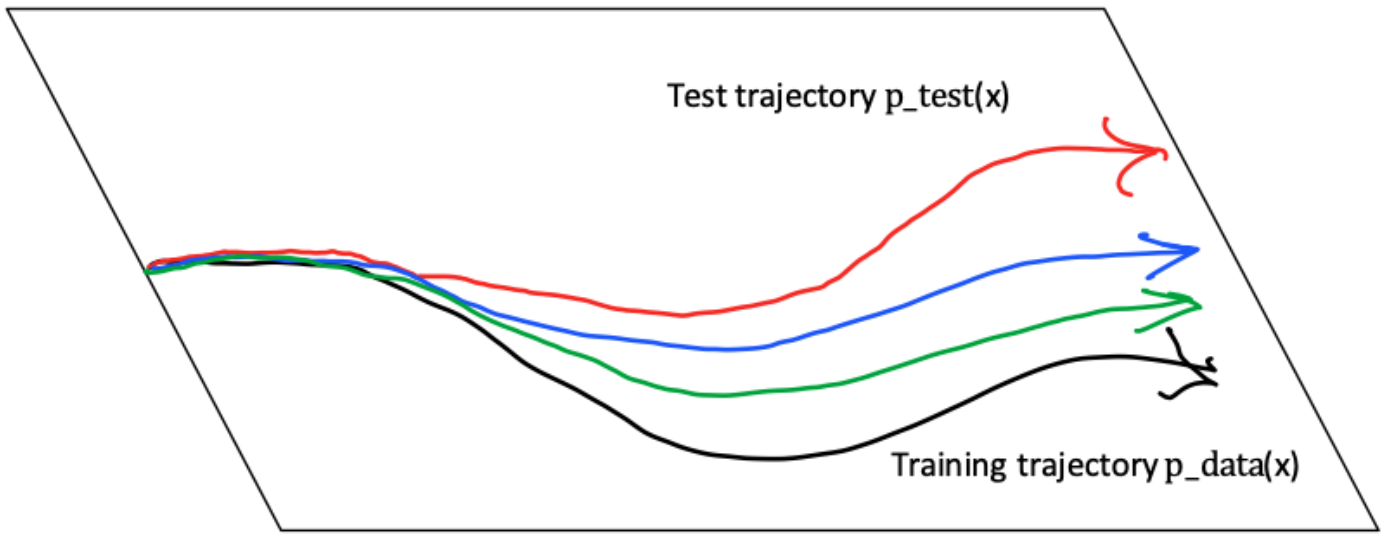
  Query expert to generate dataset $\mathcal{D}_i = \{(x_0, \pi^*(x_0)), (x_1, \pi^*(x_1)), \dots\}$

  Aggregate datsets, $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_i$

  Retrain policy $\hat{\pi}$ using aggregated dataset $\mathcal{D}$

return $\hat{\pi}$

Algorithm to mitigate the distribution mismatch



$p\_data(x) \neq p\_test(x)$

Algorithm to mitigate the distribution mismatch

Add regularization to the network model to enhance its stability.

$$\dot{h}_i = \sum_{k=1}^{m} \Big(L(h)L(h)^T + \alpha I + \tilde{W}(h) \Big)_{i,k} \big(-\partial_{h_k} V(h) \big) + f_i(h), \quad i = 1, \ldots, m,$$

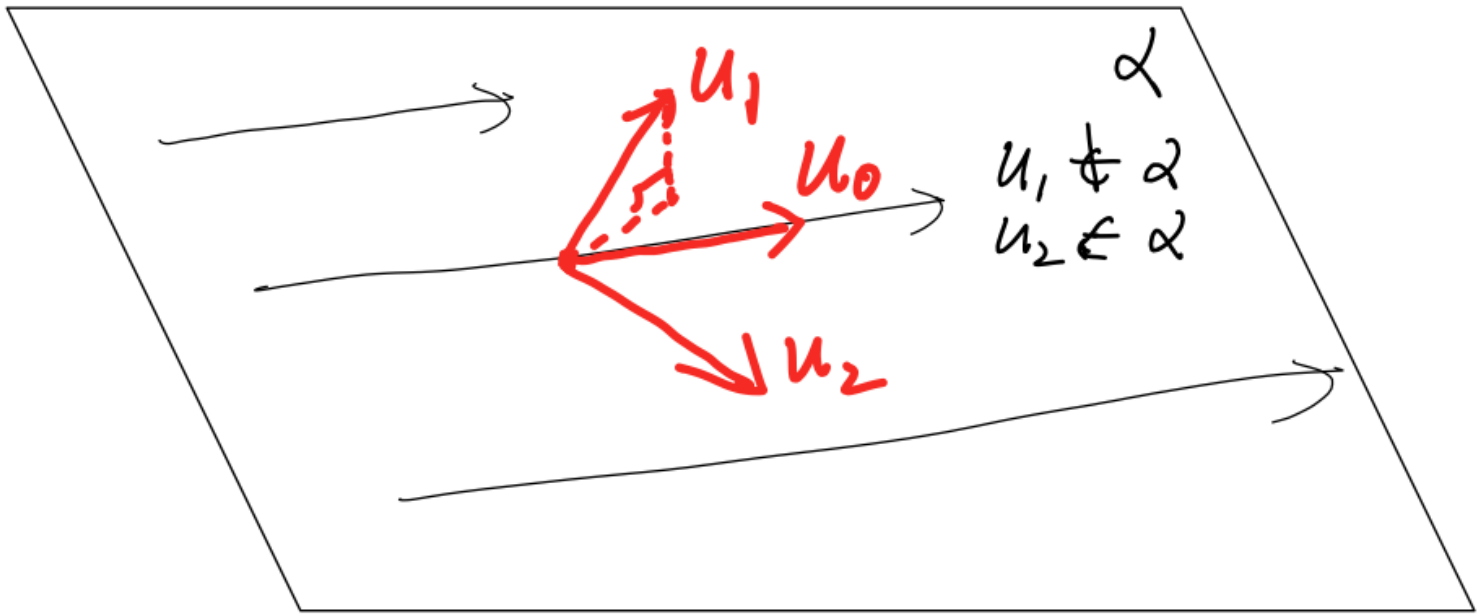
$$\mathcal{L}_{\text{ODE}} = \frac{1}{|S|} \sum_{(h(t),h(t+\tau))\in S} \frac{1}{\tau^2} \big\| h(t+\tau) - \text{RK}\,2(\text{OnsagerNet}; h(t), \tau/n_s, n_s) \big\|^2.$$

[2] Yu, Haijun, et al. "OnsagerNet: Learning stable and interpretable dynamics using a generalized Onsager principle." Physical Review Fluids 6.11 (2021): 114402.

An intuition

$$\partial_t \mathbf{x} = \mathcal{L}(\mathbf{x}, \mathbf{u}, t), \quad \mathbf{x} \in \mathcal{X}, \mathbf{u} \in \mathcal{U}, \mathcal{L} : \mathcal{X} \times \mathcal{U} \times \mathbb{R}_+ \to T\mathcal{X},$$
$$\mathbf{u} = \phi(\mathbf{x}), \quad \phi : \mathcal{X} \to \mathbf{u}.$$

Ordinary least square $\|\mathbf{u} - \phi(\mathbf{x}, t)\|^2$ cannot distinguish the following case. Add regularization for this as $R(\widehat{\mathbf{x}})$.



Algorithms and numerical experiments

Linear control:

$$\begin{aligned} \dot{x}_t &= v_t, \\ \dot{v}_t &= -\alpha(t)v_t + u_t, \\ x_0 &= 1, v_0 = 0, \\ x_1 &= 0, \\ \alpha(t) &= \sin 10t. \end{aligned} \tag{3}$$

And we use a **neural network** with 2 hidden layer to parameterize the mapping from (x_k, v_k, t_k) to u_k. Hence the neurons in each layer is given by $3, n, n, 1$ where n can be varied in the experiments.

Algorithms and numerical experiments

Inner-Outer loop structure of this linear control:

$$\begin{aligned} \begin{pmatrix} x_{k+1} \\ v_{k+1} \end{pmatrix} &= \begin{pmatrix} 1 & dt \\ 0 & 1 - \alpha_k dt \end{pmatrix} \begin{pmatrix} x_k \\ v_k \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u_k, \\ u_k &= \phi_{NN}(x_k, v_k, t_k). \end{aligned} \tag{4}$$

And ordinary least square (behavior cloning) simply solve the model by minimizing the following function:

$$l(\phi_{NN}(x_k, v_k, t_k), u_k) = \|\phi_{NN}(x_k, v_k, t_k) - u_k\|_2^2 .$$

Bench mark: Ordinary least square

Averaged error is given by 0.192341.

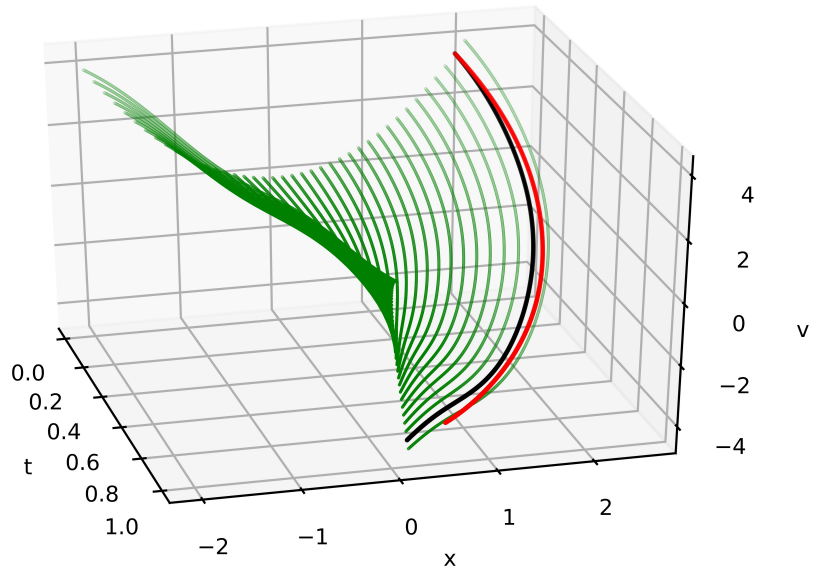


Figure: Sampled training trajectory of an estimator with underlying net of 10 hidden neurons.

Manifold regularization

Several choices of empirical manifold:

Formed by the sample trajectories in the training data, i.e.

$$\mathcal{M}_t := \{(x(t), v(t), t) \quad t \in [0, 1]\},$$

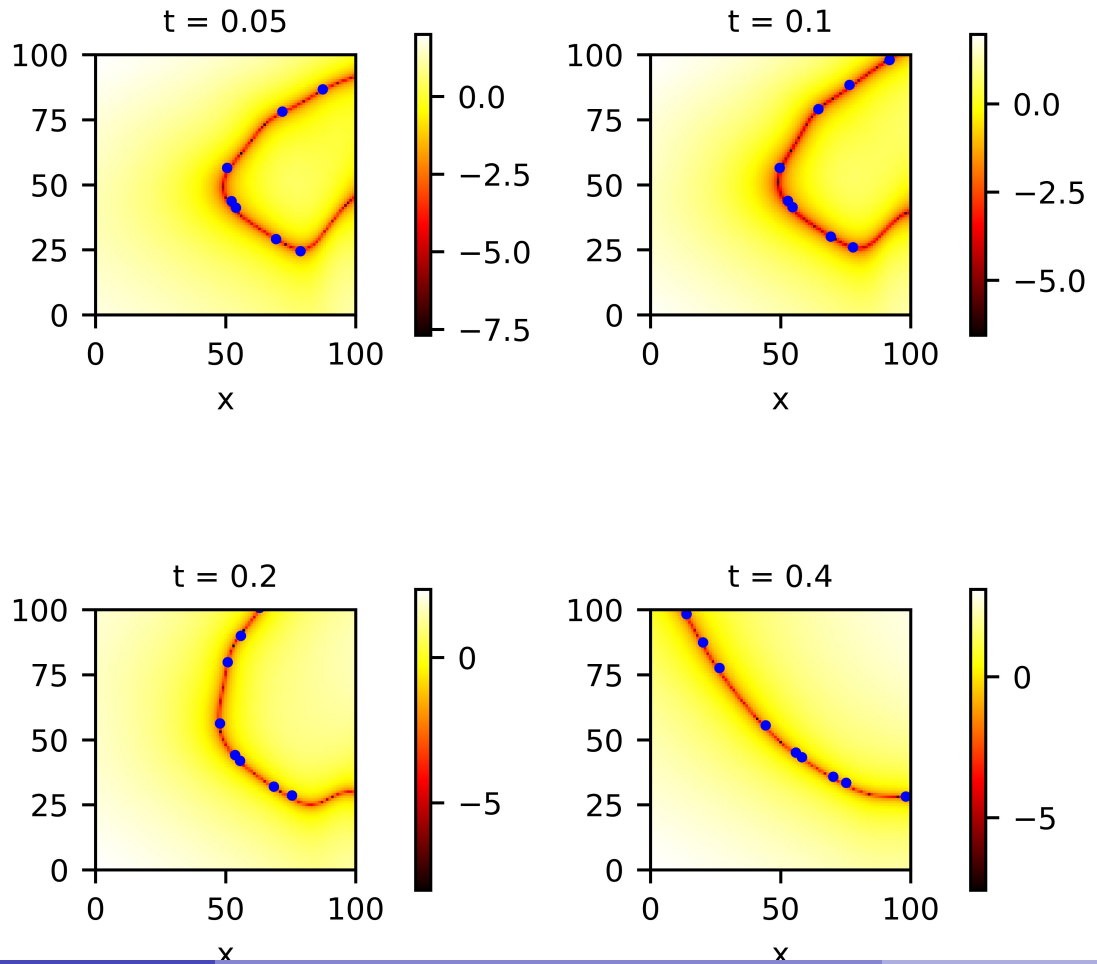
The other empirical manifold related to the data-driven surrogate model and can be defined as following set:

$$\widehat{\mathcal{M}}_t := \left\{(x, v, t) \middle| \, \|\phi_{NN}(x_k, v_k, t_k) - u_k\|_2 < \epsilon \right\}.$$

$\mathcal{M}_t$

$$\min_{\theta} \mathbb{E} \left\| \mathbf{u} - \phi_\theta(\mathbf{x}, t) \right\|^2 + \lambda \left\| \widehat{\mathbf{u}} - \phi_\theta(\widehat{\mathbf{x}}, t) \right\|^2$$

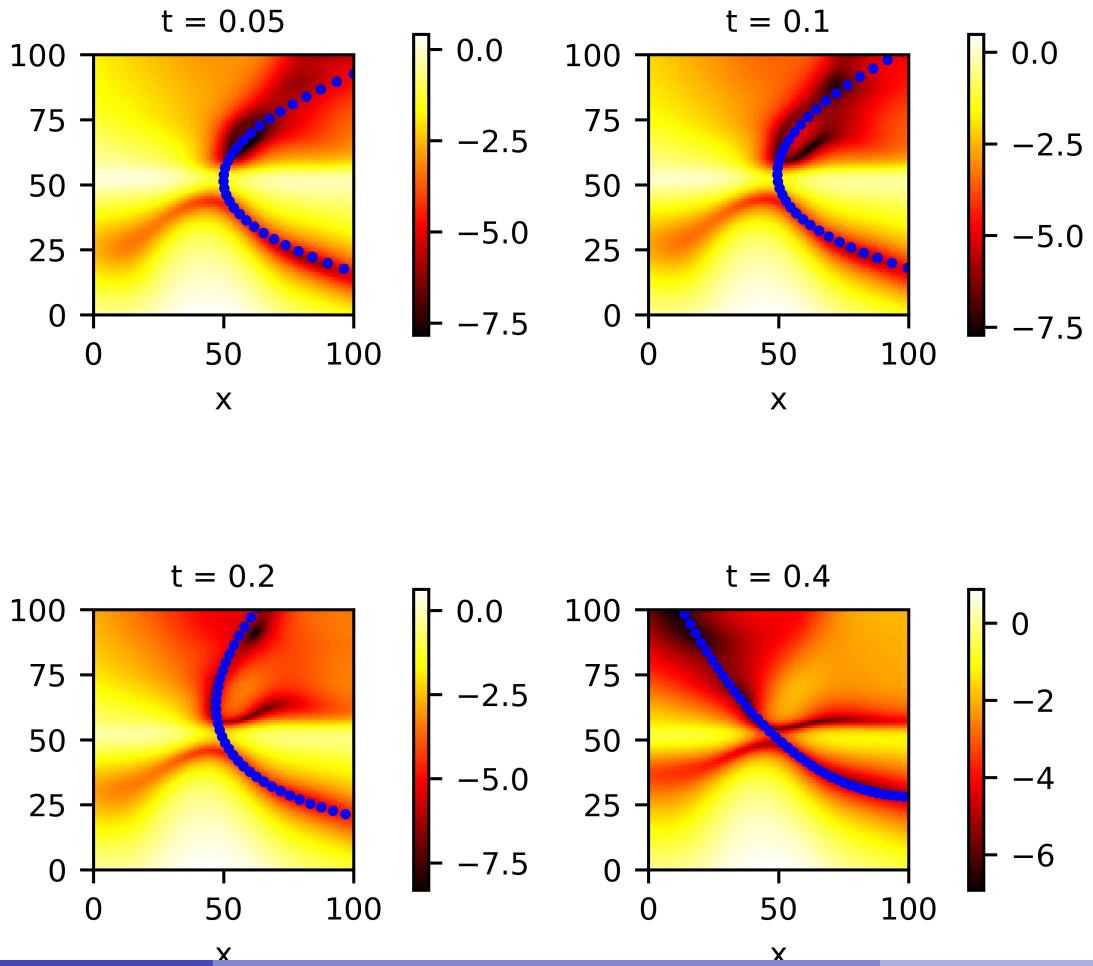
Averaged error is given by 0.160128.



$\widehat{\mathcal{M}}_t$

$$\min_{\theta} \mathbb{E} \left\| \mathbf{u} - \phi_\theta(\mathbf{x}, t) \right\|^2 + \lambda \left\| D(E(\widehat{\mathbf{x}})) - \widehat{\mathbf{x}} \right\|^2$$

Averaged error is given by 0.173001.



Future work: Algorithm

In the algorithmic perspective

1. Complete the work on encoder-decoder regularized method to mitigate the distribution mismatch in scientific computing.
2. Scientific computing + Manifold learning/self-supervised learning.

Future work: Application

In terms of application, we will further investigate the following thing:

1. Try to implant the best performed algorithm to the problems in fluid mechanics such as turbulence transition and flow separation.
2. Consider interesting problems in physics such as black holes dynamics, space-time classification in general relativity.

Future work: Theory

Theoretically, we will consider

1. Prove the effectiveness of the algorithm like Dagger and manifold learning method.

2. Consider algorithms belong to inverse reinforcement learning in our model-based setting.