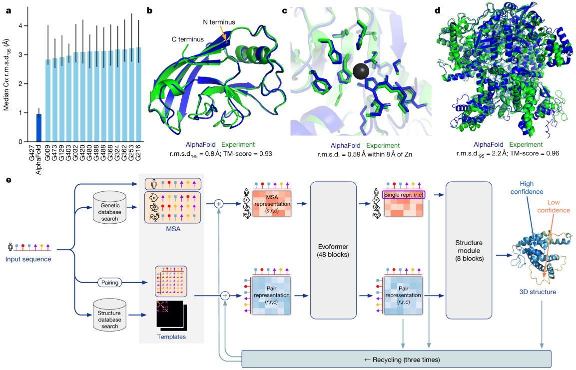
On distribution mismatch in Data-driven Scientific Computing

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

Among most application of data-driven scientific computing, there are mainly three categories of approach

1. Replace the whole numerical approach by a data-driven method such as supervised learning: Physics-informed neural networks (PINN) and related work.
2. Use data-driven method as a surrogate model to substitute part of the empirical model in a numerical method: Machine learning turbulence modeling, DeepPotential for molecular dynamics.
3. Discovering physics law such as conservation law, phase transition using symbolic regression, principle component analysis and other methods.

PINN

The Laplacian equation is formulated as a regression problem[1]with regularization:

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

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

[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

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

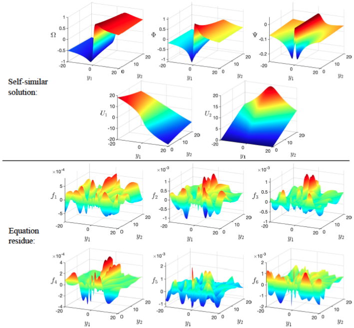


Figure 1: Smooth solution for the 2D Bousinessq equations (2.1) derived by the physics-informed neural network. f_1 to f_6 indicate the residue of the 6 equations defined in (3.3), which are around four orders of magnitude smaller than the output variables of the solution. The inferred value of λ for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

We briefly review the Reynolds average Navier-Stokes (RANS) equation. Denote by $\langle \mathbf{U} \rangle$ the time average of $\mathbf{U}$ and $\mathbf{u} = \mathbf{U} - \langle \mathbf{U} \rangle$:

$$\partial_t \langle \mathbf{U} \rangle + (\langle \mathbf{U} \rangle \cdot \nabla) \langle \mathbf{U} \rangle + \frac{\partial \langle \mathbf{u} u_j \rangle}{\partial x_j} = -\frac{1}{\rho} \nabla p + \nu \Delta \langle \mathbf{U} \rangle , \quad (1)$$

An extra term comes in, i.e. $\langle \mathbf{u} u_j \rangle$ Reynolds stress. The equation is no longer close!!

Ling et al.[2] proposed a data-driven surrogate model to estimate the Reynolds stresses based on the averaged velocity field via neural network.

[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.

Data-driven method for turbulence modeling

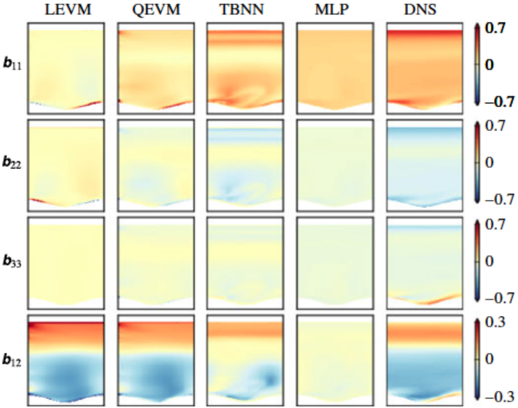


FIGURE 3. Predictions of Reynolds stress anisotropy $\boldsymbol{b}$ tensor on the wavy wall test case. The columns show the predictions of the LEVM, QEVM, TBNN and MLP models. The true DNS anisotropy values are shown in the right-most column for comparison.

Imitation learning

In contrast to the task in the computer science society, scientific computing is more close to reinforcement learning. In most setting that is important to us, the task belongs to the regime of imitation learning.

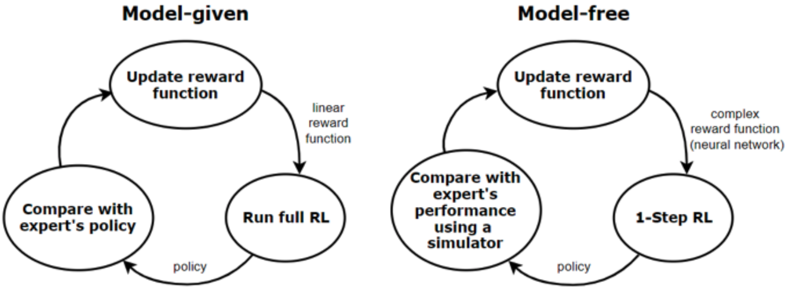
Definition

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* $\Xi = \{(x_0, u_0), (x_1, u_1), \cdots\}$ *from an expert policy* π^* *to find a policy* $\widehat{\pi}$ *that imitates the expert policy.*

What has been done in the imitation learning community?

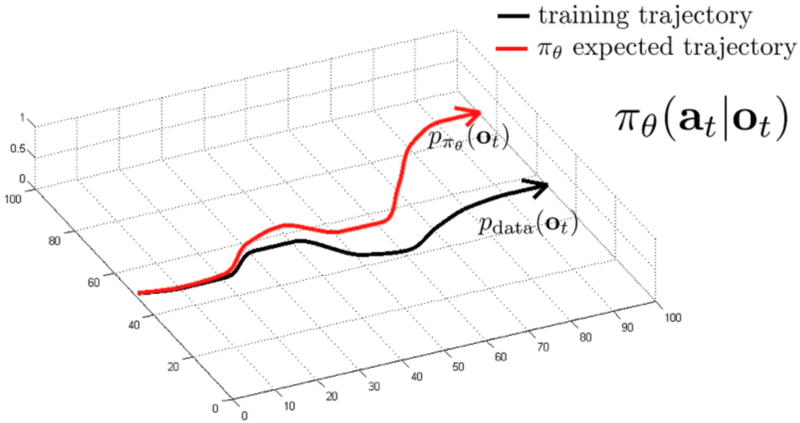
Basic algorithm called **behavior cloning** defines a loss function $l(\cdot, \cdot)$ and a parametrised model $\phi(\cdot, \theta)$ to learn the mapping between x and u, i.e. $\arg\min \mathbb{E}_x l(u, \phi(x, \theta))$.

Inverse reinforcement learning attempt to learn the reward function based on the policy.



Dilemma of data-driven scientific computing

One of most important dilemma in the data-driven scientific computing is the distribution mismatch between the training and testing data, due to the dynamics structure of the problem.



the problem: $p_{\text{data}}(\mathbf{o}_t) \neq p_{\pi_\theta}(\mathbf{o}_t)$

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

Add regularization to the network model to enhance its stability:

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

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

[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

Since the dilemma comes from the mismatch between training and test distribution, one can add regularization to the optimization step to force them become similar. In other word, one can force the test trajectories stays near the training set. This can be formed as a standard problem in manifold learning.

Inner-outer loop

A common structure in the data-driven scientific computation method: inner-outer loop:

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

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)

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

Algorithms and numerical experiments

We test different methods in a toy model of linear control.

$$\begin{aligned} \dot{x}_t &= v_t, \\ \dot{v}_t &= -\alpha(t)v_t + u_t, \end{aligned} \tag{3}$$

The initial value is given by $x_0 = 1, v_0 = 0$ and the goal is that $x_1 = 0$. u_t is the control variable to be determined. There are two different kinds of the time-dependent condition $\alpha(t) = t^2$ and $\alpha(t) = \sin 10t$.

In fact, u obey the Ricatti equation. 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 behavior cloning method simply solve the model by minimizing the following function:

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

Ordinary least square

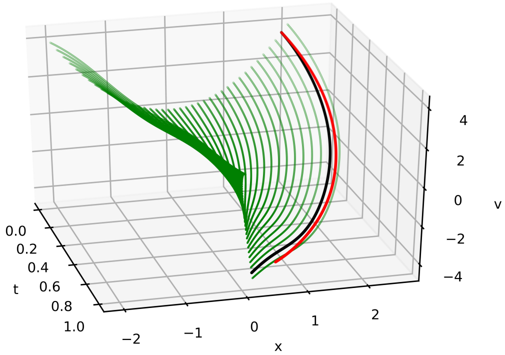


Figure: Sampled training trajectory of an estimator with underlying net of 10 hidden neurons.

Ordinary least square

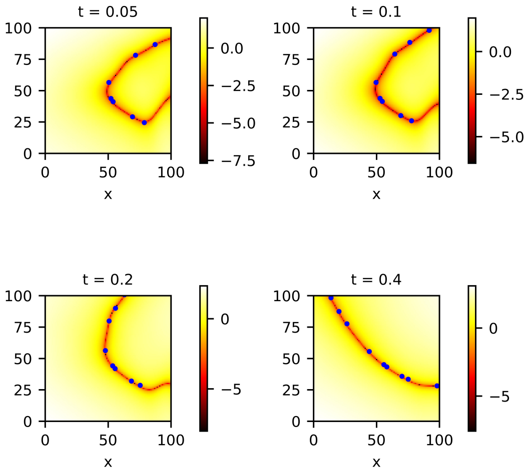


Figure: Sampled training trajectory of an estimator with underlying net of 50 hidden neurons.

Manifold regularization

Following our intuition, we introduce manifold regularization to the loss function. Several choices of empirical manifold is listed:

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:

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

Manifold regularization

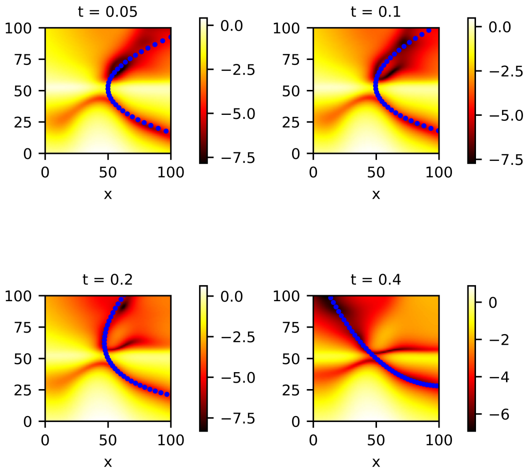


Figure: Sampled training trajectory of an estimator with underlying net of 50 hidden neurons.

Manifold regularization

The manifold related to the data-driven surrogate model characterizes the underlying manifold structure better than the manifold encoded by the training dataset, and it also outperform this latter one in sampling new trajectories.

Future work: Algorithm

In the algorithmic perspective

1. Develop effective method to mitigate the distribution mismatch in scientific computing problems.

2. Combine the method of manifold learning and self-supervised learning in ML community to further explore our algorithm.

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 other interesting problems in physics which may lack the current focus.

Future work: Theory

Theoretically, we will consider

1. In the toy control model we are considering now, what is the specific form and effectiveness of the algorithm like Dagger and manifold learning method?