Data-driven numerical simulation with application in computational fluid dynamics

Jiaxi Zhao

NUS-Kajima workshop

25th Oct, 2023

Data-driven scientific computing

What are the problems we are interested in?

1. Forward problem: Increase the stability and accuracy of machine learning-augmented simulation

2. Inverse problem: Perform effective sensitivity analysis

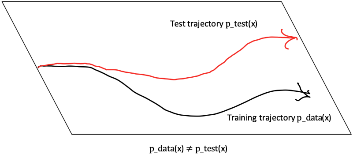
What are the method we focus on?

1. 100% data-driven: Physics-informed neural networks (PINN), Fourier neural operator, DeepONet.

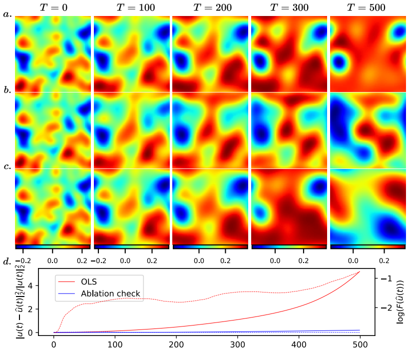
2. 50 % Numerical + 50 % data-driven: Machine learning turbulence modeling, DeepPotential, Quasipotential.

Dilemma of data-driven scientific computing

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



(a) Distribution shift illustration



(b) Distribution shift in reaction-diffusion equation

Figure: Distribution shift in data-driven scientific computing

Network architecture

We choose U-net for

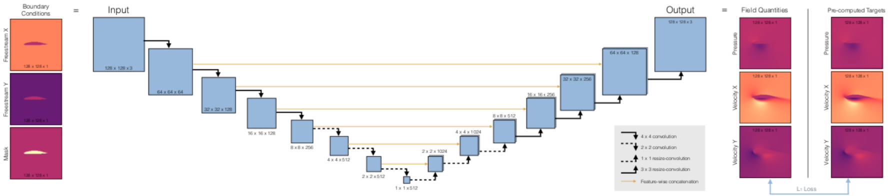


Figure: U-net structure for flow prediction[1]

[1] Thuerey, Nils, et al. "Deep learning methods for Reynolds-averaged Navier–Stokes simulations of airfoil flows." AIAA Journal 58.1 (2020): 25-36.

Performance comparison

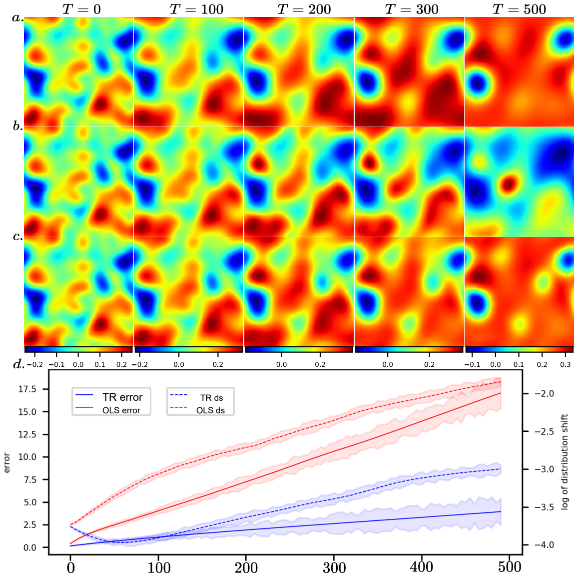


Figure: Comparison of our method and naive method

Performance comparison

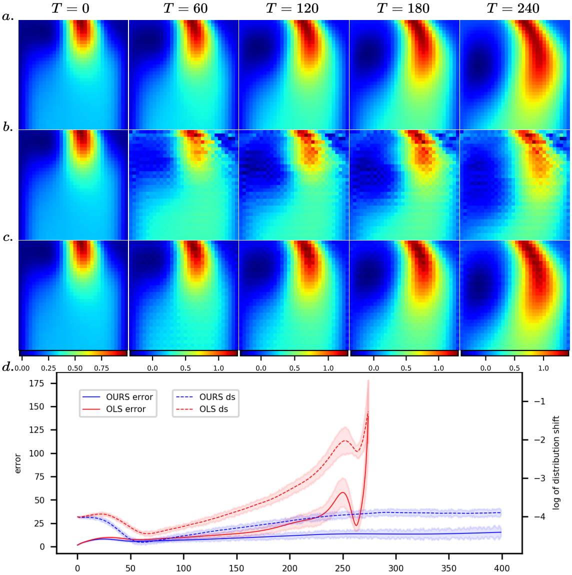


Figure: Comparison of our method and naive method

Further Application

1. Various turbulence modeling: Subgrid modeling, Wall modeling, Transition modeling, etc.

2. Coupled CFD: Fluid-structure interaction (multiphase flow), flow with heat transfer, etc.

The idea of shadowing

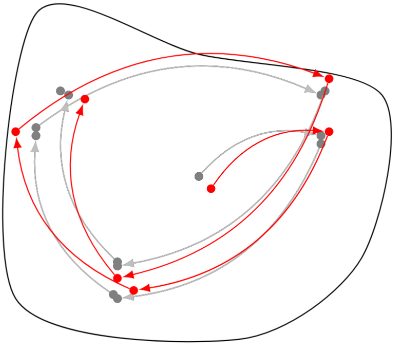


Figure: Shadowing trajectory

Solve LSS: Least square

Writing the linearized equation as a linear constraint

$$\min \sum_{t=1}^{T} v_t^T v_t$$

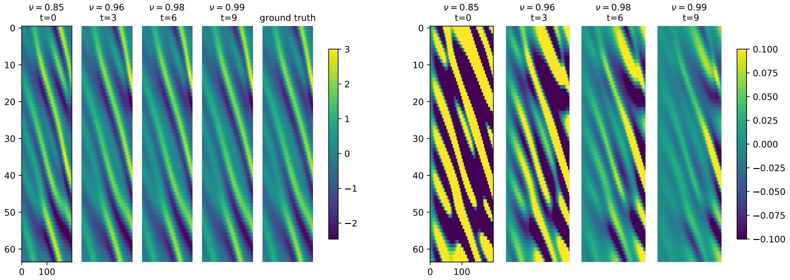
$$\begin{pmatrix} \mathbf{I} & -\nabla_u f(u_{T-1}) & \cdots & 0 & 0 \\ 0 & \mathbf{I} & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & \mathbf{I} & -\nabla_u f(u_1) \\ 0 & 0 & \cdots & 0 & 0 \end{pmatrix} \begin{pmatrix} v_T \\ v_{T-1} \\ v_{T-2} \\ \vdots \\ v_2 \\ v_1 \end{pmatrix} = \begin{pmatrix} \partial_s f(u_{T-1}) \\ \partial_s f(u_{T-2}) \\ \partial_s f(u_{T-3}) \\ \vdots \\ \partial_s f(u_1) \\ 0 \end{pmatrix}, \tag{1}$$

This is just a least square problem of size $T \times N$.

Sensitivity analysis of KS equation

Consider the 1D Kuramoto–Sivashinsky equation is written as

$$\partial_t u + u u_x + u_{xx} + \nu u_{xxxx} = 0, \quad x \in [0, L], \tag{2}$$



(a) Cloudmap along optimization

(b) Error along optimization

Figure: Performance of LSS

Ongoing work: Sensitivity analysis of LES

What is the problem to apply least square shadowing to LES?

We need to solve a linear system of size $N \times T$, with N the number of cells or grid points and T the number of time steps. This is computationally prohibited for even moderate LES.

Then, why not have a try with machine learning?

Application

1. Flow control
2. Inverse design & shape optimization
3. Uncertainty quantification of fluid system

Appendix: Reaction-diffusion equation

Consider following FitzHugh-Nagumo reaction diffusion equation:

$$\frac{\partial \mathbf{u}}{\partial t} = \gamma \Delta \mathbf{u} + \mathbf{R}(\mathbf{u}), \quad T \in [0, 1],$$
$$\mathbf{R}(\mathbf{u}) = \mathbf{R}(u, v) = \begin{pmatrix} u - u^3 - v - \alpha \\ \beta(u - v) \end{pmatrix}, \tag{3}$$

The initial data is given by $\mathbf{u}_0$ is a random field and generated by i.i.d. sampling from a normal distribution and $\alpha = 0.001, \beta = 1.0, \gamma = \begin{pmatrix} 0.05 & 0 \\ 0 & 0.1 \end{pmatrix}$. We use mesh size 128×128 for the whole problems. Computational domain is given by $[0, 6.4] \times [0, 6.4]$.

Appendix: Incompressible Navier-Stockes equation

Consider incompressible NS equation:

$$\begin{aligned} \frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} - \nu \Delta \mathbf{u} = \nabla p, \quad T \in [0, 1], \\ \nabla \cdot \mathbf{u} = 0, \end{aligned} \tag{4}$$

The computational domain is a rectangular $[0, 4] \times [0, 1]$. The boundary condition on upper and lower boundary is no-slip for velocity. The boundary condition for outlet is zero-gradient on pressure while we specify the inlet velocity.

Appendix: Shadowing lemma

Theorem (Shadowing lemma)

Let Γ be a hyperbolic invariant set of a diffeomorphism f. There exists a neighborhood U of Γ with the following property: for any $\delta > 0$ there exists $\epsilon > 0$, such that any (finite or infinite) ϵ-pseudo-orbit that stays in U also stays in a δ-neighborhood of some true orbit[2].

In a hyperbolic invariant set, the dynamics exhibit a combination of stable and unstable behavior.

Shadowing trajectory demo:

[2] Pilyugin SY. Shadowing in dynamical systems. Lecture notes in mathematics, vol. 1706. Springer; 1999.