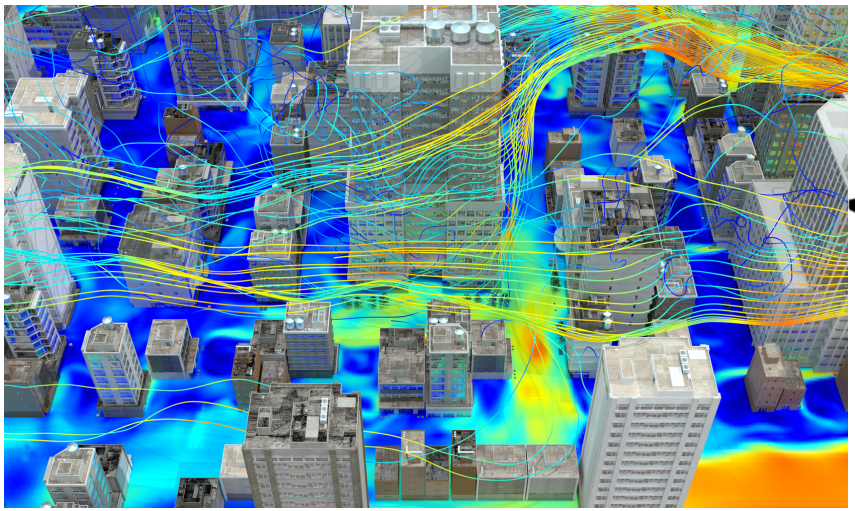
Mitigating distribution shift in machine learning-augmented hybrid simulation[1]

Jiaxi Zhao (NUS)
joint work with Prof. Qianxiao Li (NUS)

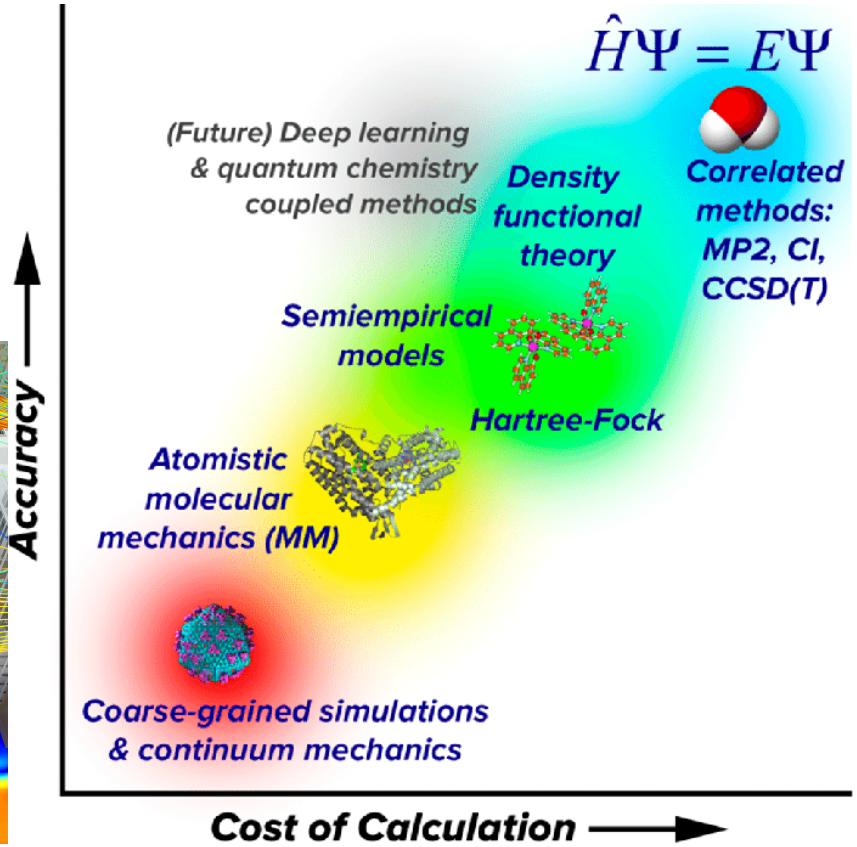
The 13th Symposium of the SIAM Student Chapter@NUS
May 28, 2024

[1] https://arxiv.org/abs/2401.09259

Practical scientific problems



(a) Computational fluid dynamics for urban environment



(b) Quantum chemistry for material science

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: Learn the surrogate models and Perform effective sensitivity analysis to do inverse designs.

What are methods we focus on?

1. 100% data-driven: AlphaFold series, FermiNet, Fourier neural operator, DeepONet.

2. 50 % Numerical + 50 % data-driven: Machine learning turbulence modeling, force fields, exchange-correlation functionals.

A Framework for the hybrid 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. Benchmark algorithm solves the ordinary least square:

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

Why surrogate models? The dynamics are unknown, nonlinear, computationally expensive etc.

An example

Let us use the incompressible NS equation as an example

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

Consider solving it using the projection method, in each step, we need to solve the following equation

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \Delta t(\nu \Delta \mathbf{u}_k - (\mathbf{u}_k \cdot \nabla)\mathbf{u}_k - \nabla p_k),$$
$$p_k = \phi(\mathbf{u}_k) = \Delta^{-1}(\nabla \cdot (\nu \Delta \mathbf{u}_k - (\mathbf{u}_k \cdot \nabla)\mathbf{u}_k)),$$

The most important features are:

1. iterative solver
2. data-driven

Dilemma of data-driven scientific computing

In data-driven scientific computing, dynamics itself can cause distribution mismatch between the training and testing data. Similarly to the extrapolation, out-of-distribution issue in NLP.

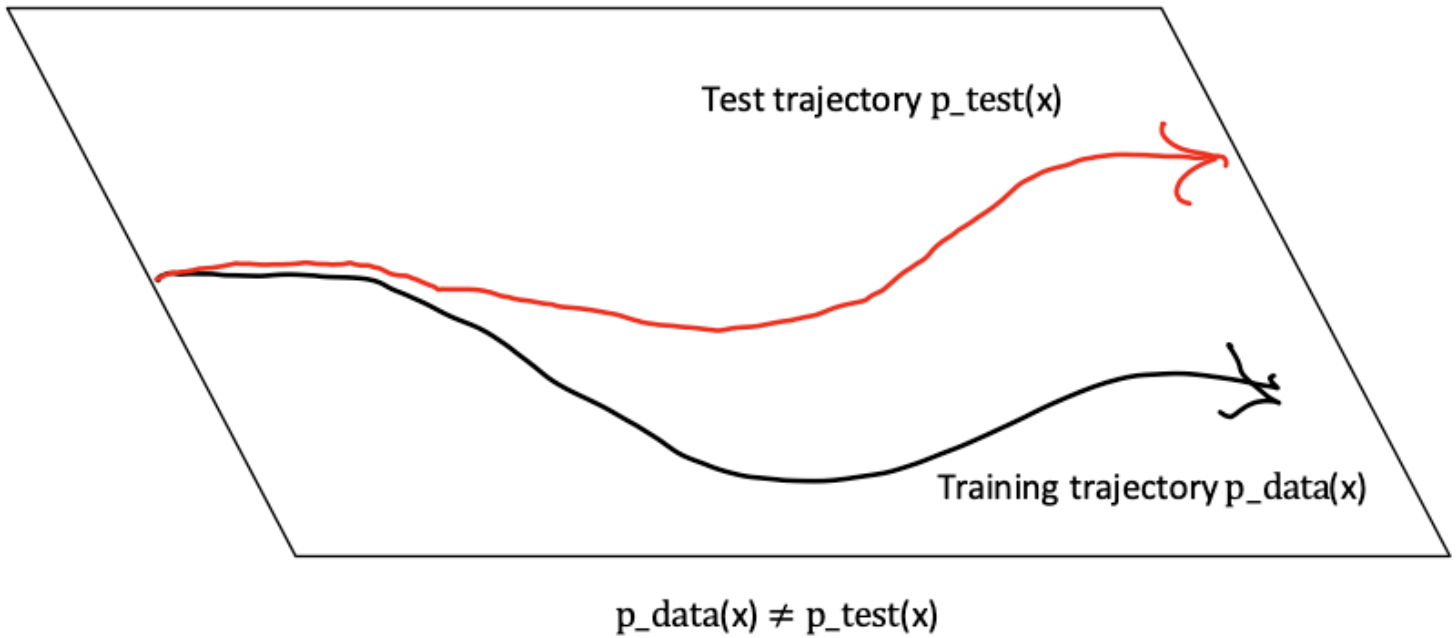


Figure: Distribution shift illustration

Different from classical numerical stability

This is different from the classical numerical stability issue. For the ablation study, we add noises of the same scale to the numerical solver (without the data-driven part) and compare the simulations.

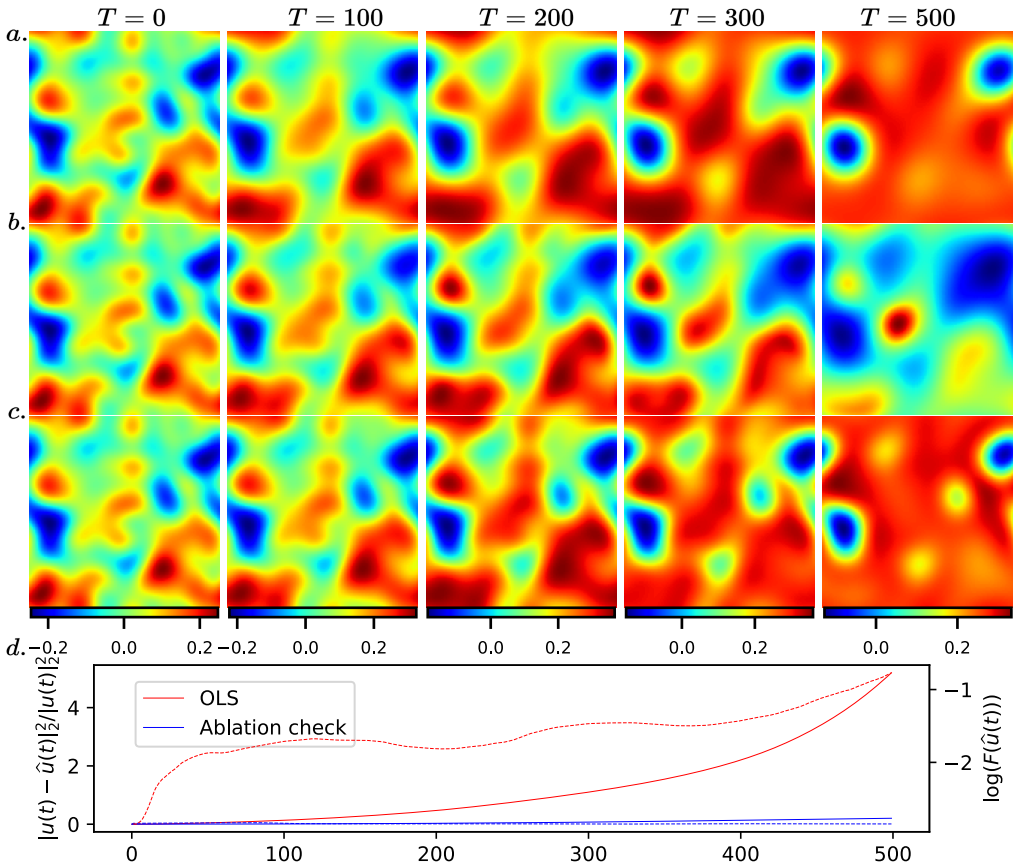
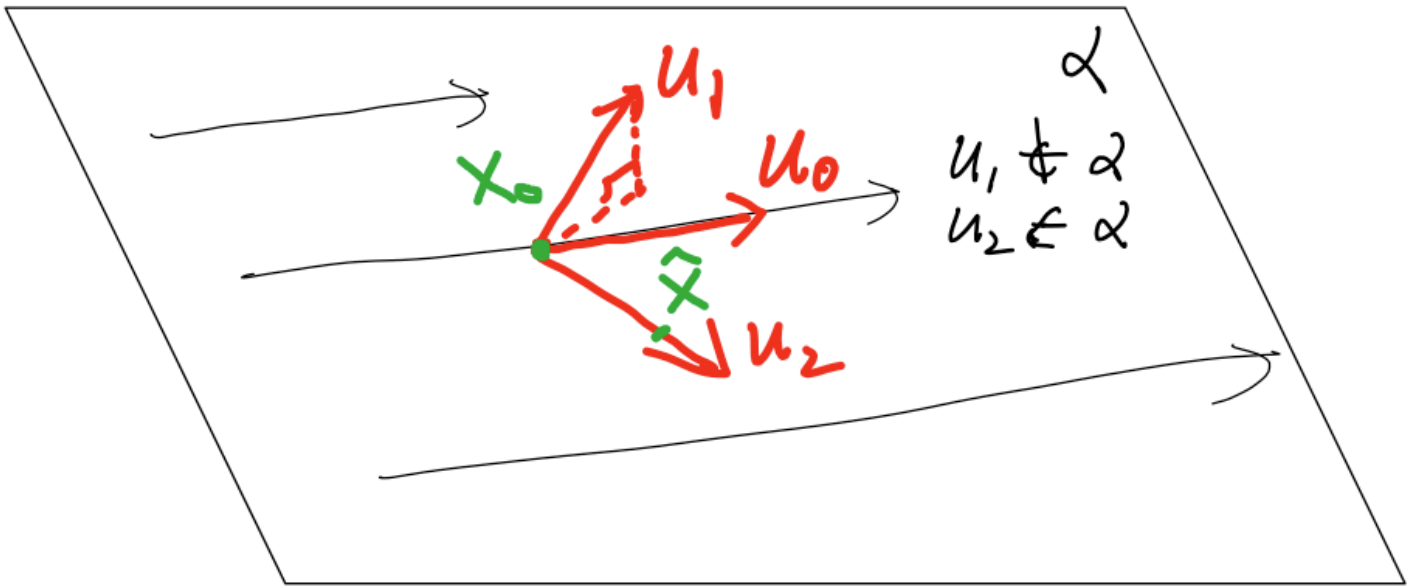


Figure: Distribution shift in reaction-diffusion equation

An heuristic solution



We design an algorithm that favors $\mathbf{u}_2$ than $\mathbf{u}_1$ by adding some regularization.

Linear dynamics

We consider the following the linear hybrid simulation problem

$$\frac{d\mathbf{u}}{dt} = A\mathbf{u} + B\mathbf{y}, \quad \mathbf{u} \in \mathbb{R}^m, \mathbf{y} \in \mathbb{R}^n, A \in \mathbb{R}^{m\times m}, B \in \mathbb{R}^{m\times n}$$
$$\mathbf{y} = C^*\mathbf{u}, \quad C^* \in \mathbb{R}^{n\times m}.$$

The least squares estimator aims to minimize the following loss

$$l_{\text{OLS}}(\widehat{C}) = \mathbb{E}\left\|(\widehat{C} - C^*)\mathbf{u}\right\|^2,$$

while the proposed estimator minimizes

$$l_{\text{TR}}(C) := \mathbb{E}_{(\mathbf{u},\mathbf{y})}\left(\left\|(\widehat{C} - C^*)\mathbf{u}\right\|_2^2 + \lambda \left\|P_{V^\perp}(A + BC)\mathbf{u}\right\|_2^2\right),$$

Linear theory

The tangent-space regularized estimator performs a weighted least squares

Proposition

Given a data matrix $\mathbf{U} \in \mathbb{R}^{m \times N}$ *with observation noise* ϵ *of the same shape*

$$\widehat{C}_{OLS} = C^* P_V + \epsilon \mathbf{U}^\dagger,$$
$$\widehat{C}_{TR} = (\mathbf{I} + \lambda B^T P_{V^\perp} B)^{-1}(C^* P_V + \epsilon \mathbf{U}^\dagger - \lambda B^T P_{V^\perp} A P_V).$$

Specifically, in the noiseless scenarios, we have

$$\widehat{C}_{\mathrm{OLS}} = C^* P_V, \quad \widehat{C}_{\mathrm{TR}} = (\mathbf{I} + \lambda P_{V^\perp})^{-1}(C^* P_V + \epsilon \mathbf{U}^\dagger) = (\mathbf{I} + \lambda P_{V^\perp})^{-1} \widehat{C}_{\mathrm{OLS}}.$$

Linear theory

The tangent-space regularized estimator has slower error scaling for large λ:

Theorem

With $Q_m(r, T)$ *defined by* $m^2 \int_0^T (2 + t^{m-1}) e^{rt} dt$ *and*

$$e_1 = eig_{\max}(A + B\widehat{C}), \quad e_2 = eig_{\max}((A + B\widehat{C})P_V),$$
$$eig_{\max}(A) = \max\{\Re(s) : \exists v \neq \mathbf{0}, Av = sv\},$$

the errors of OLS and our algorithm are bounded respectively by

$$\mathbb{E}\,\|\widehat{\mathbf{u}}_{OLS}(T) - \mathbf{u}(T)\| \leq c_1\sqrt{\delta}\,\|B\|_2\, Q_m(e_1, T),$$

$$\mathbb{E}\,\|\widehat{\mathbf{u}}_{TR}(T) - \mathbf{u}(T)\| \leq c_2\sqrt{\delta}\Big(\,\|B\|_2\, Q_m(e_2, T)$$

$$+ \frac{9m^4 c_3}{\sqrt{\lambda}}\left(1 + 3m^2 c_1\sqrt{\delta}\,\|B\|_2\right)\left\|A + B\widehat{C}\right\|_2 (1 \vee T^{3m})(1 \vee e^{e_1 T})\Big).$$

Linear experiments

Consider a linear synthetic dynamics where the data subspace $V(V^{\perp})$ is the stable (unstable) manifold.

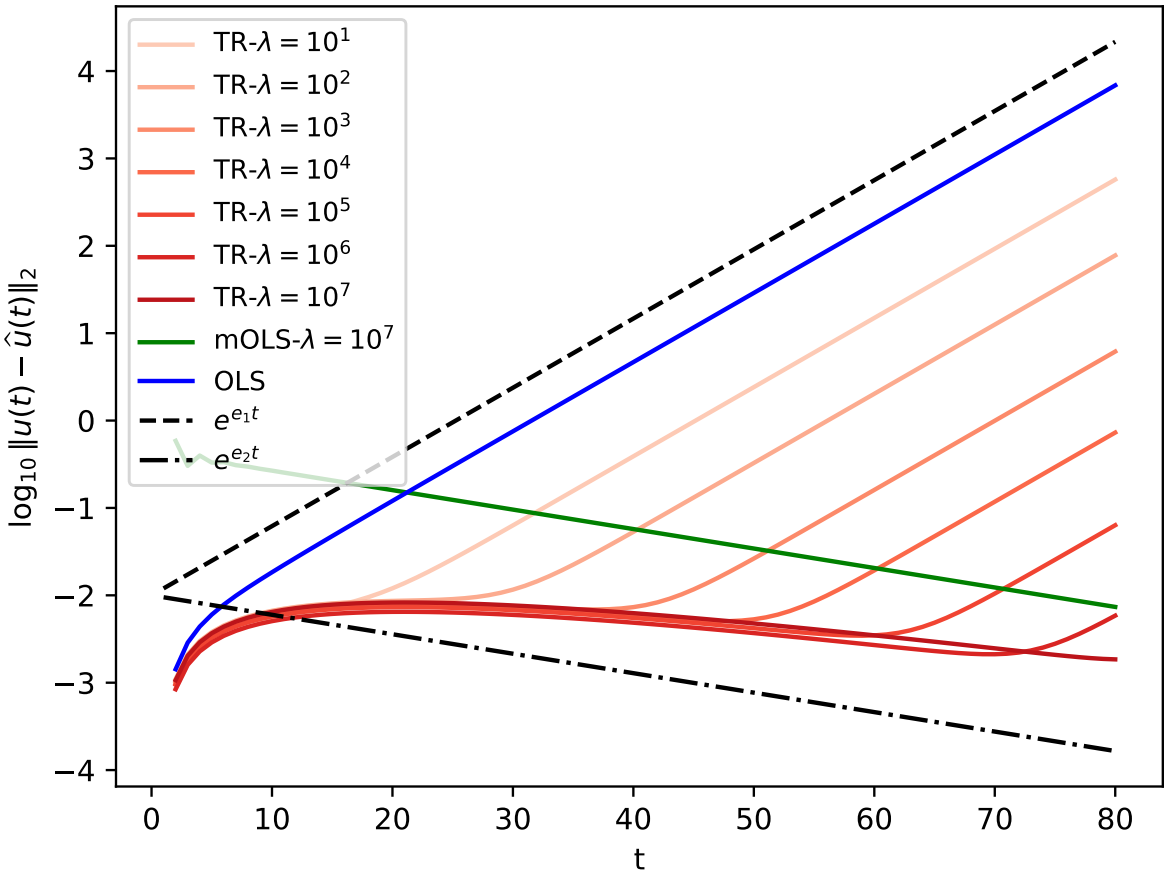


Figure: Linear synthetic dynamics

Overall algorithm

Use an autoencoder to parametrize the nonlinear data-manifold.

Algorithm 3.1 MLHS with tangent-space regularized estimator

input $\{(\mathbf{u}_1, \mathbf{y}_1, t_1), \cdots, (\mathbf{u}_N, \mathbf{y}_N, t_N)\}$, resolved model, penalty strength λ.

1: Learn a parameterized model which encodes the structure of training data, i.e. $F_\eta(\mathbf{u}) \geq 0, F_\eta(\mathbf{u}_k) = 0$.
2: Freeze the parameters of this learned model.
3: Introduce another surrogate model $\phi_\theta(\mathbf{u})$.
4: **for** $k = 1, 2, \cdots, N$ **do**
5: Predict the control variable $\widehat{\mathbf{y}}_k = \phi_\theta(\mathbf{u}_k)$.
6: Calculate the state variable after one-step iteration, i.e. $\widehat{\mathbf{u}}_{k+1} = \mathbf{u}_k + \Delta t L(\mathbf{u}_k, \widehat{\mathbf{y}}_k, t_k)$.
7: Form the loss $l(\theta) = \mathbb{E}\left[\|\mathbf{y}_k - \widehat{\mathbf{y}}_k\|_2^2 + \lambda\left(\left(\nabla F(\mathbf{u}_k)\right)^T L(\mathbf{u}_k, \widehat{\mathbf{y}}_k, t_k)\right)^2\right]$.
8: Backpropogate to update θ.
9: **end for**

output tangent-space regularized estimator: ϕ_θ.

Figure: Algorithm pseudocode

Network architecture

We choose U-net for

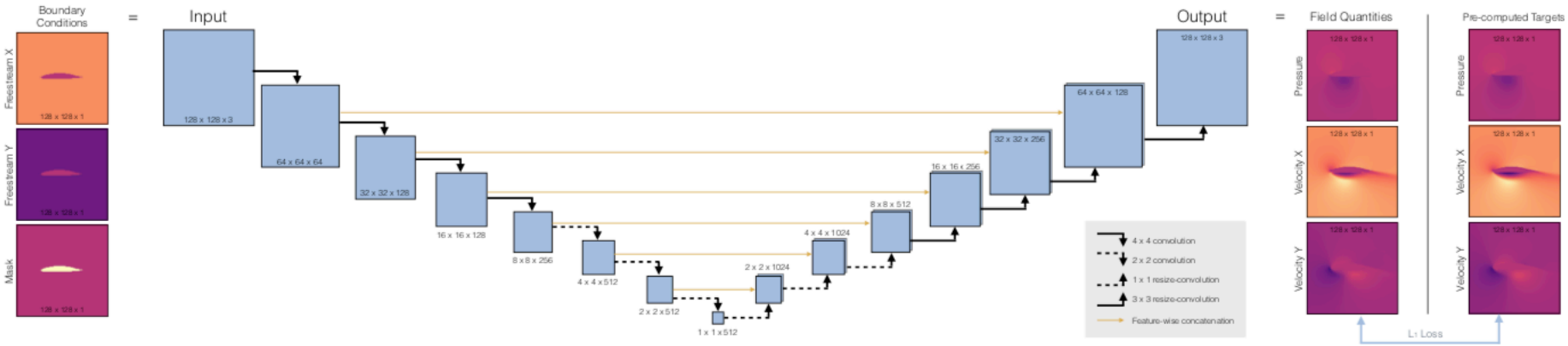


Figure: U-net structure for flow prediction[2]

[2]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

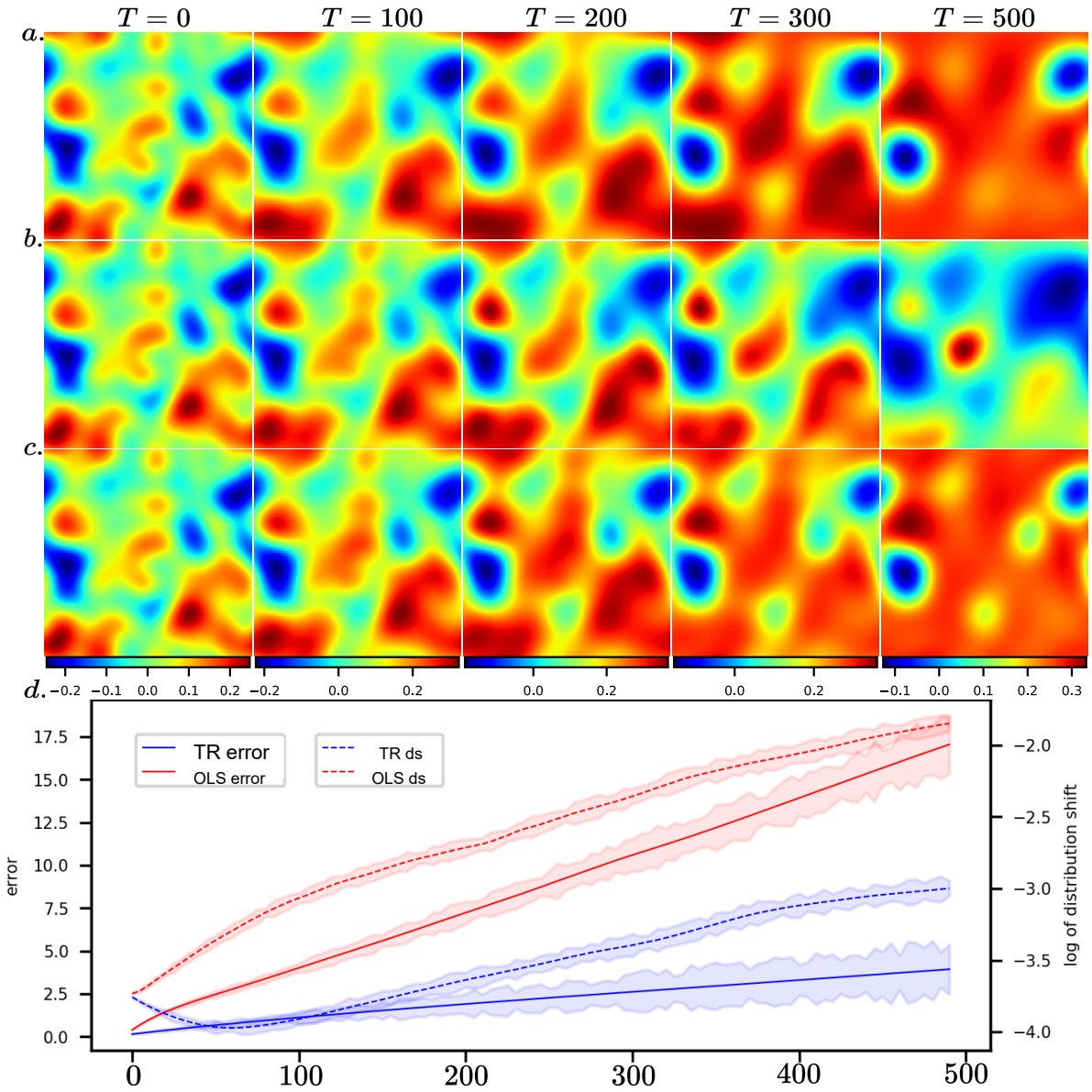


Figure: Comparison of our method and naive method

Performance comparison

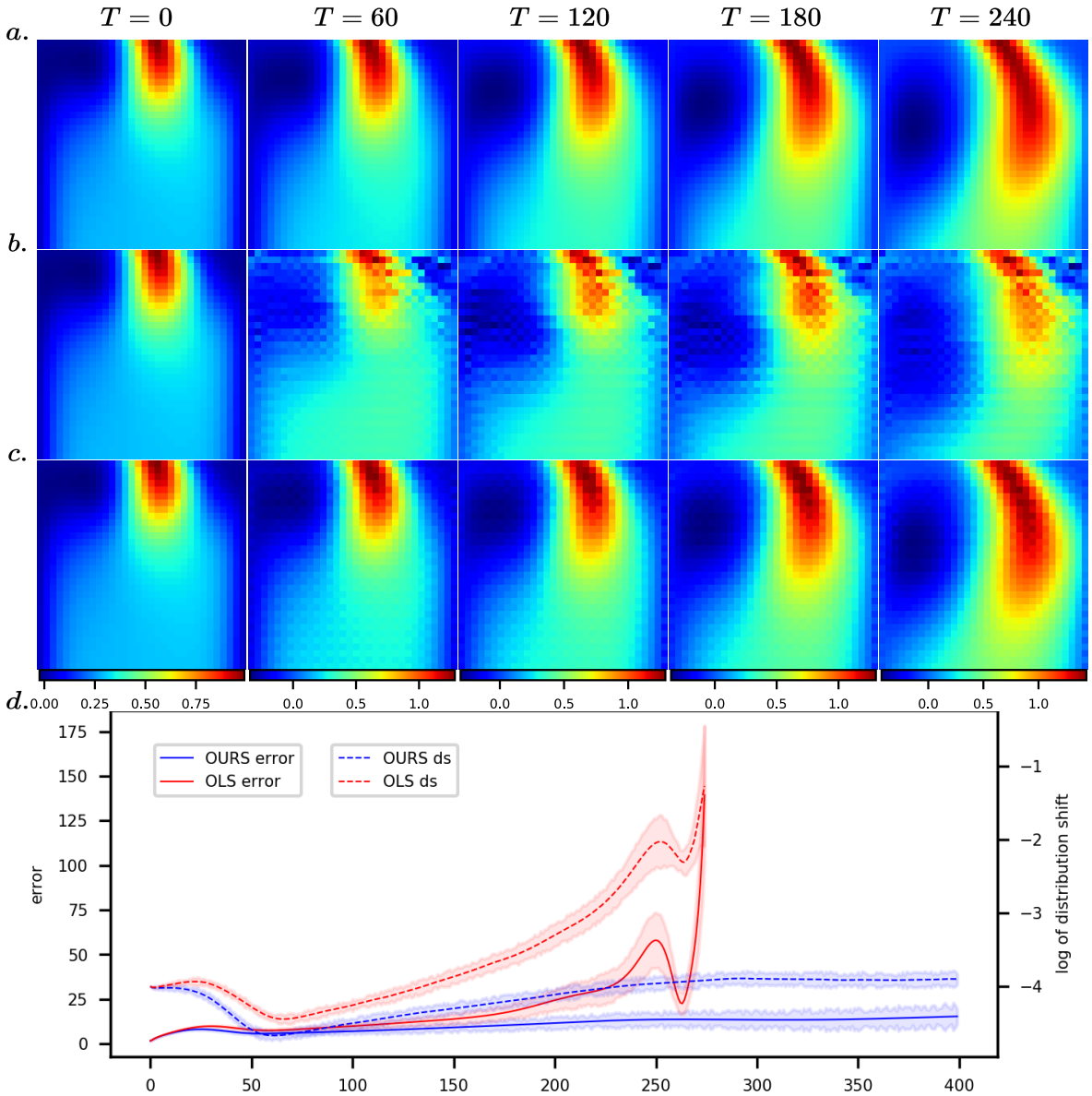


Figure: Comparison of our method and naive method

Future work

1. Deploy to practical problems: Subgrid-scale modeling in large eddy simulation of the urban environment.
2. Extend the framework to solve the inverse problem via surrogate modeling.
3. Theoretical analysis of the proposed framework, a.k.a. numerical analysis for SciML.

Reaction-diffusion equation

Consider the following FitzHugh-Nagumo reaction-diffusion equation:

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

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 problem. Computational domain is given by $[0, 6.4] \times [0, 6.4]$.