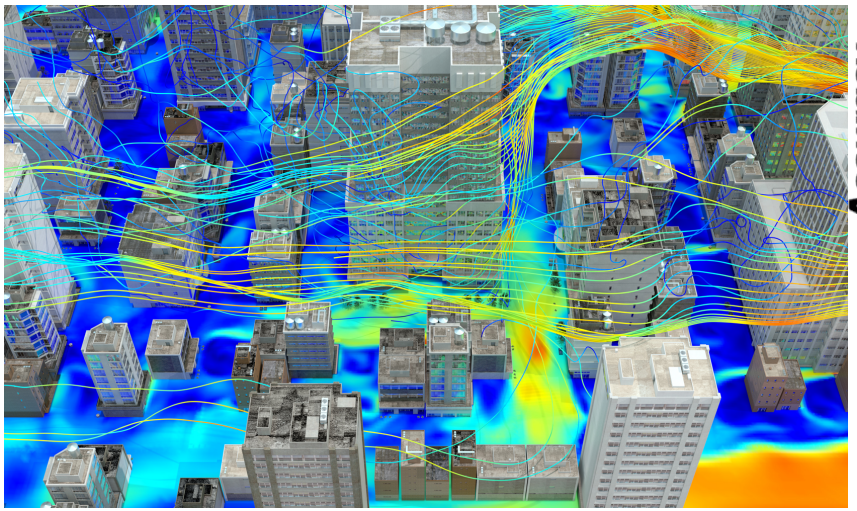
Mitigating distribution shift in machine learning-augmented hybrid fluid simulation

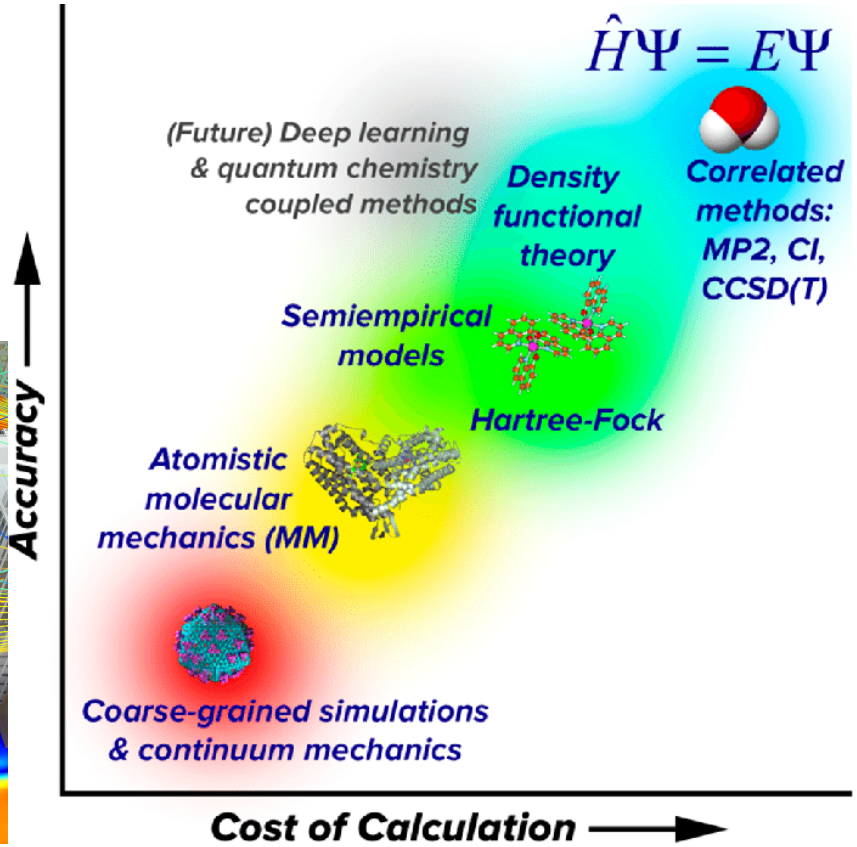
Jiaxi Zhao (NUS)
joint work with Sohei Arisaka and Prof. Qianxiao Li (NUS)

AMSS, CAS
August 14, 2024

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:

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

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

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

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

An example: projection scheme

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

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

An example: coarse-fine grid

Suppose we have a fine grid with size $2n \times 2n$ and a coarse grid with size $n \times n$. We want to learn a correction term between the numerical simulation in these two grid sizes. Fix an interpolation and a restriction operator:

$$\mathsf{I}_n^{2n} : \mathbb{R}^{n\times n} \to \mathbb{R}^{2n\times 2n}, \quad \mathsf{R}_{2n}^{n} : \mathbb{R}^{2n\times 2n} \to \mathbb{R}^{n\times n}.$$

Now, we can state the second hybrid simulation problem as follows:

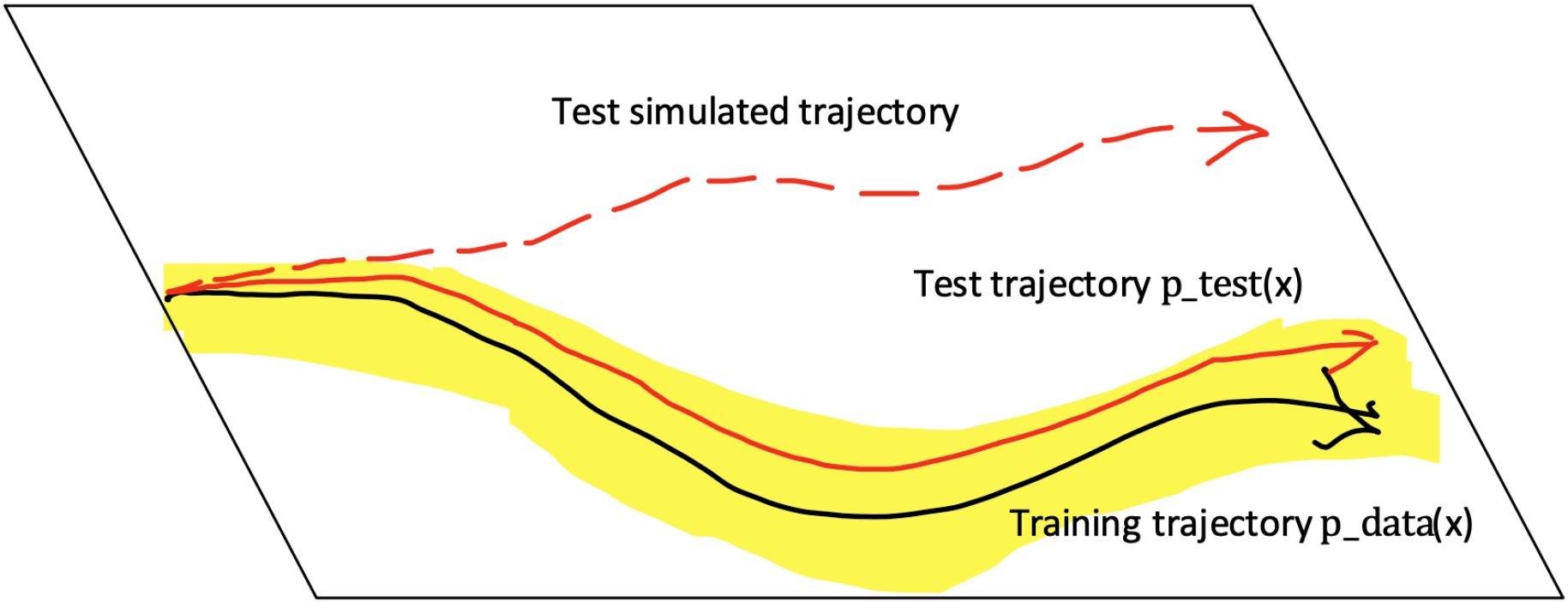
$$\begin{cases} \mathbf{u}_{k+1}^{2n} = \mathsf{I}_n^{2n} \circ f_n(\mathsf{R}_{2n}^{n}(\mathbf{u}_k^{2n})) + \mathbf{y}_k^{2n}, \\ \mathbf{y}_k^{2n} = \phi(\mathbf{u}_k^{2n}). \end{cases}$$

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.



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

Comparison with 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 ground truth (without the data-driven part) and compare the simulations.

$$\begin{aligned} \text{Data-driven model:} \quad & \phi_\theta, \\ \text{Perturbed ground truth:} \quad & \phi_* + \mathbb{E}\left\|\mathbf{y} - \phi_\theta(\mathbf{u}, t)\right\| \epsilon, \ \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \end{aligned}$$

where ϕ_* can be replaced by $\phi^{\Delta x}$, a fine-grid numerical solver.

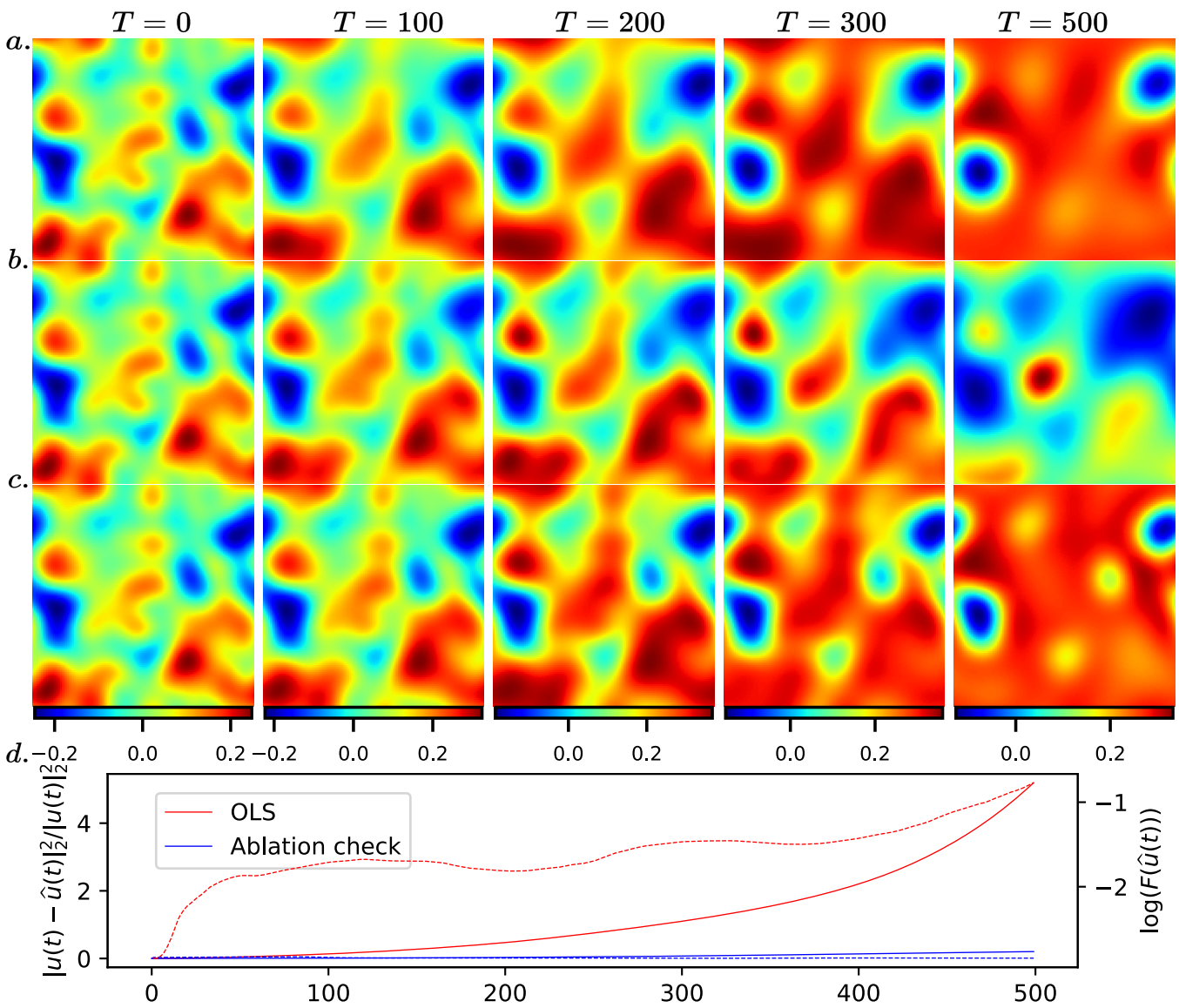
Reaction-diffusion equation

Consider the following FitzHugh-Nagumo reaction-diffusion equation:

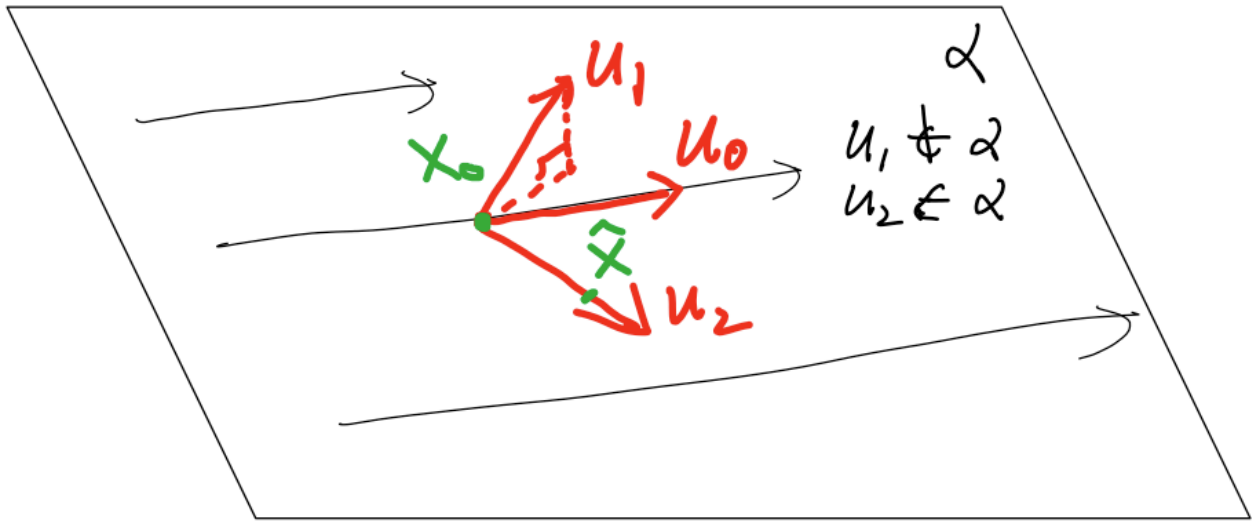
$$\begin{aligned} \frac{\partial \mathbf{u}}{\partial t} &= \gamma \Delta \mathbf{u} + \mathbf{R}(\mathbf{u}), \quad T \in [0, 1], \\ \mathbf{R}(\mathbf{u}) = \mathbf{R}(u, v) &= \begin{pmatrix} u - u^3 - v - \alpha \\ \beta(u - v) \end{pmatrix}, \end{aligned} \tag{1}$$

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 $[0, 6.4] \times [0, 6.4]$.

Comparison with classical numerical stability



An heuristic solution

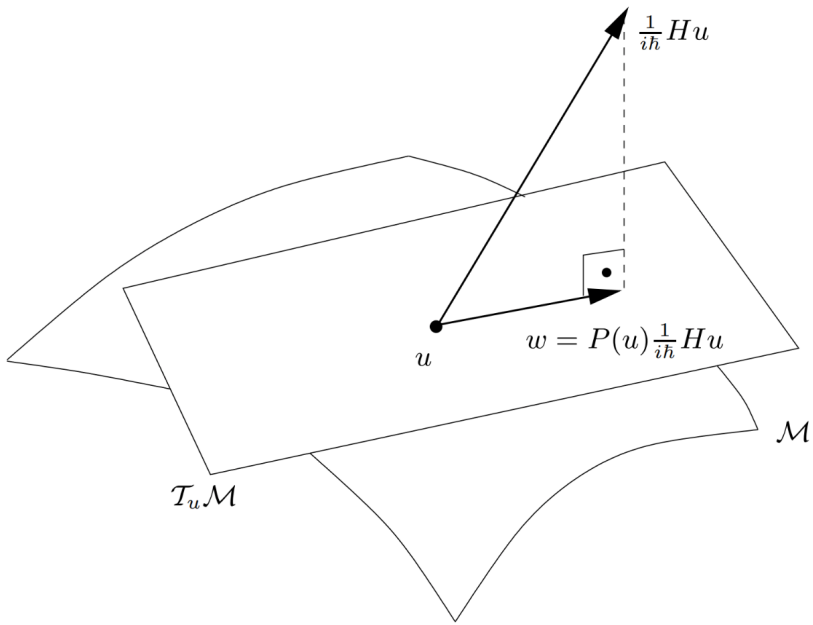


We design an algorithm that favors $\mathbf{u}_2$ than $\mathbf{u}_1$ by adding some regularization. This is very similar to the Dirac-Frenkel variational principle in dynamical low-rank approximation.

Dirac-Frenkel variational principle

The Dirac-Frenkel variational principle is originally developed for the Schrodinger equation $\frac{d\psi}{dt} = \frac{1}{i\hbar} H\psi$,

$$P\frac{d\psi}{dt} \in T_\psi \mathcal{M}, s.t. \left\langle v \middle| P\frac{d\psi}{dt} - \frac{1}{i\hbar} H\psi \right\rangle = 0, \forall v \in T_\psi \mathcal{M}. \quad (2)$$



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

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

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*

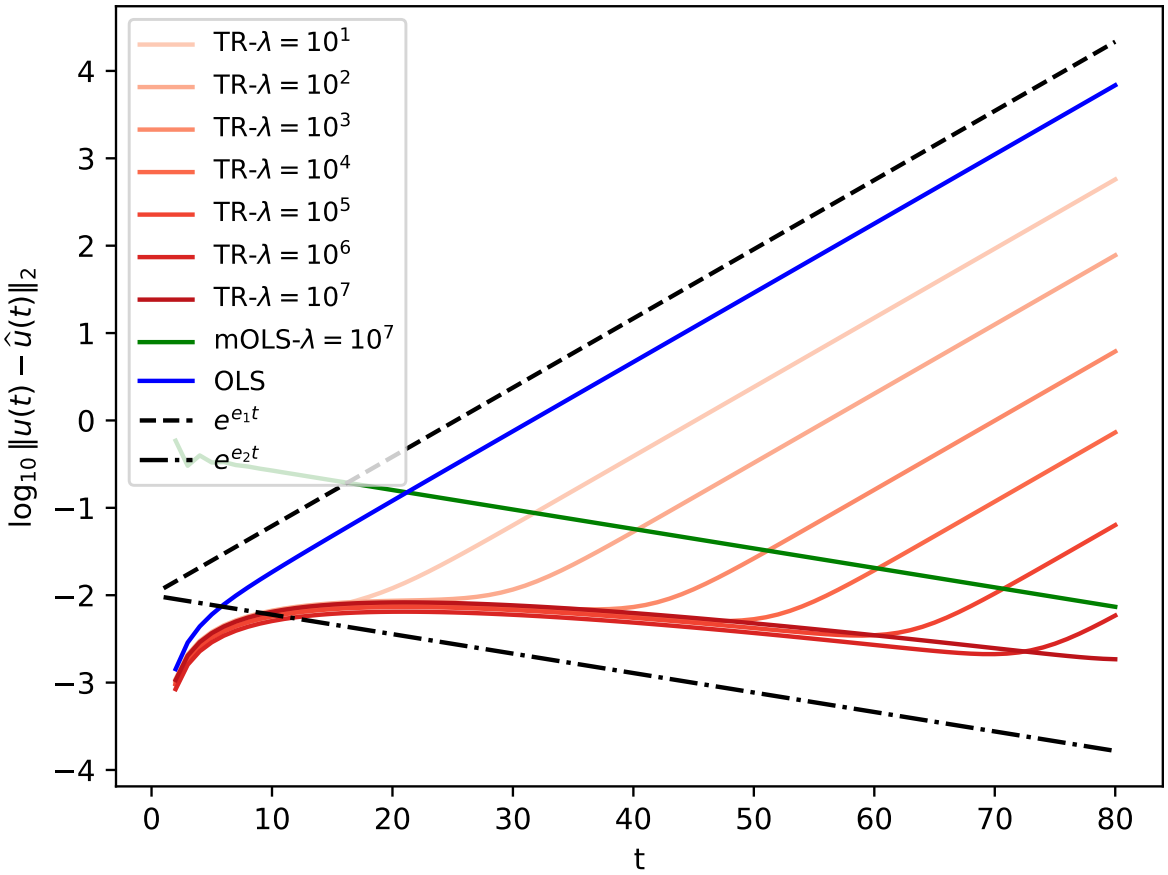
$$e_1 = eig_{\max}(A+B\widehat{C}), \quad e_2 = eig_{\max}((A+B\widehat{C})P_V),$$
$$eig_{\max}(A) = \max\{\Re(s) : \exists v \neq \mathbf{0}, Av = sv\},$$

the errors of OLS and our algorithm are bounded respectively by

$$\mathbb{E}\,\|\widehat{\mathbf{u}}_{OLS}(T) - \mathbf{u}(T)\| \leq c_1\sqrt{\delta}\,\|B\|_2\,Q_m(e_1,T),$$
$$\mathbb{E}\,\|\widehat{\mathbf{u}}_{TR}(T) - \mathbf{u}(T)\| \leq c_2\sqrt{\delta}\Big(\,\|B\|_2\,Q_m(e_2,T)$$
$$+\frac{9m^4c_3}{\sqrt{\lambda}}\left(1+3m^2c_1\sqrt{\delta}\,\|B\|_2\right)\left\|A+B\widehat{C}\right\|_2(1\vee T^{3m})(1\vee e^{e_1T})\Big).$$

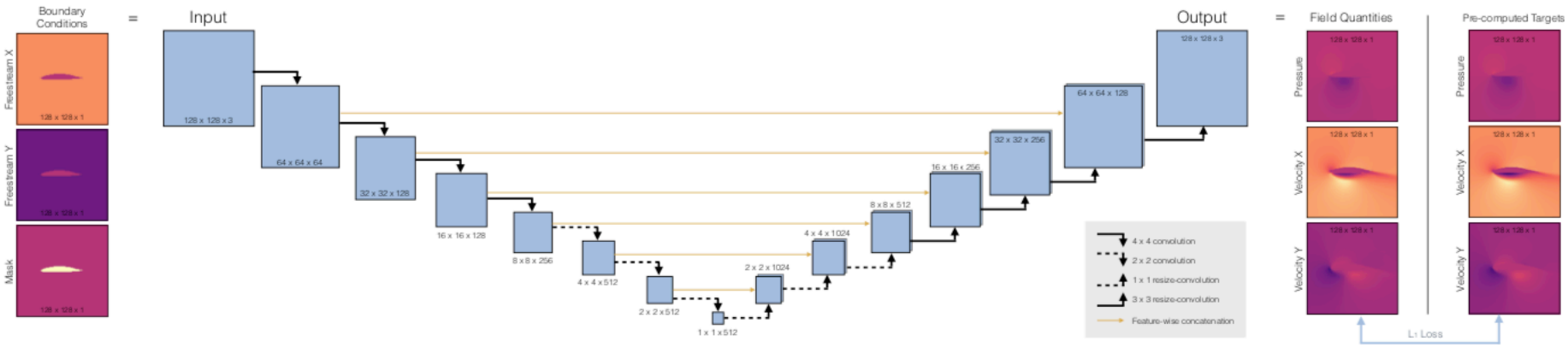
Linear experiments

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



Nonlinear dynamics

Use an autoencoder to parametrize the nonlinear data manifold. We choose U-net[1] as our autoencoder architecture trained via $\min_{D,E} \mathbb{E} \|\mathbf{u} - D(E(\mathbf{u}))\|^2$. Moreover, one can use $F(\mathbf{u}) = \|\mathbf{u} - D(E(\mathbf{u}))\|^2$ to quantify the distribution shift and $\nabla F(\mathbf{u})$ as a normal vector! But there is a small caveat.



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

Overall algorithm

Algorithm 3.1 MLHS with tangent-space regularized estimator

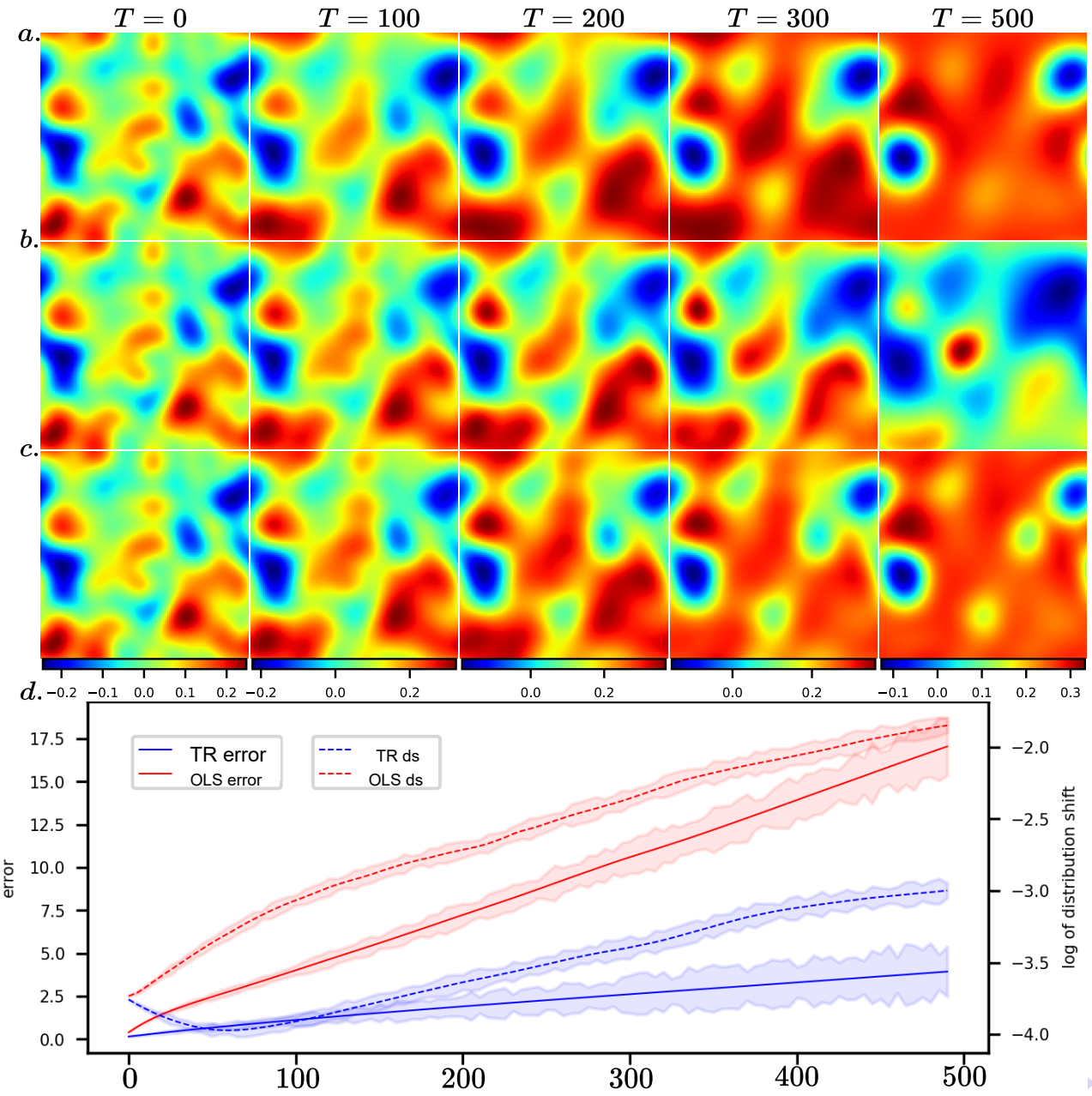
input $\{(\mathbf{u}_1, \mathbf{y}_1, t_1), \cdots, (\mathbf{u}_N, \mathbf{y}_N, t_N)\}$, resolved model, penalty strength λ.

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 $l(\theta) = \mathbb{E}\left[\|\mathbf{y}_k - \widehat{\mathbf{y}}_k\|_2^2 + \lambda \left((\nabla F(\mathbf{u}_k))^T L(\mathbf{u}_k, \widehat{\mathbf{y}}_k, t_k)\right)^2\right]$.
8: Backpropogate to update θ.
9: **end for**

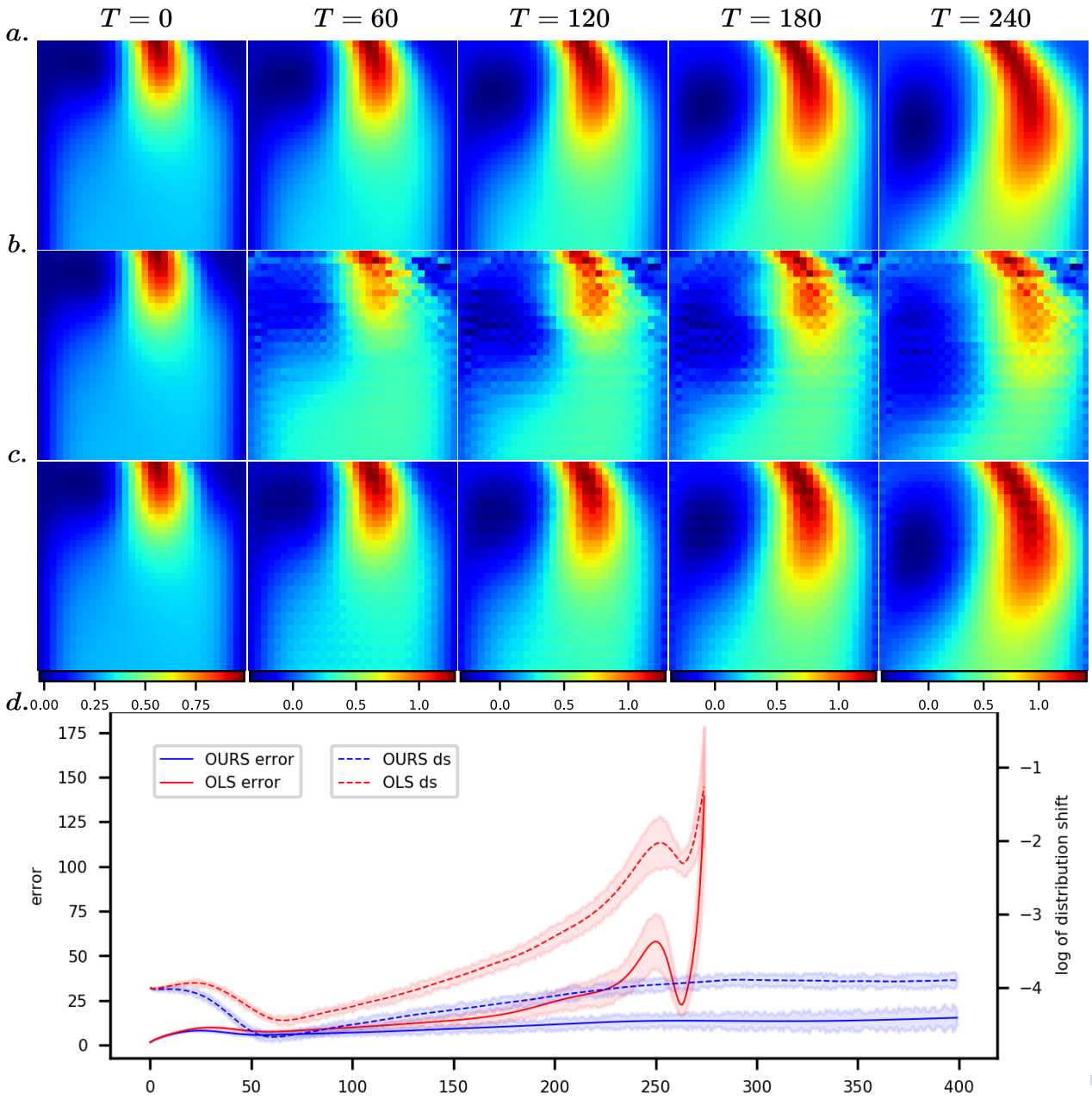
output tangent-space regularized estimator: ϕ_θ.

Our method is non-intrusive.

Performance comparison



Performance comparison



Subgrid-scale stress modeling based on NN

$$\frac{\partial \overline{u}_i}{\partial t} + \frac{\partial}{\partial x_j}(\overline{u}_i \overline{u}_j) = -\frac{\partial \overline{p}}{\partial x_i} + \nu \Delta \overline{u}_i - \frac{\partial \tau_{ij}}{\partial x_j},$$

$$\frac{\partial \overline{u}_i}{\partial x_i} = 0.$$

1. Performing DNS is unaffordable, even a LES with 50M grids of length 1000s takes several days.
2. Cheap simulations such as RANS and coarse-grid LES can not obtain accurate quantities such as peak pressure.

Can we design or learn better SGS models based on the LES data so that it can achieve accurate results even on coarse grid LES?

SGS stress modeling

There are three main issues for stress modeling:

1. The mapping from the input features. e.g. filtered velocity to the stress tensor is non-deterministic while most classical turbulence models and data-driven models are deterministic.

$$\overline{\mathbf{U}}, \nabla\overline{\mathbf{U}}, \overline{p} \overset{\text{NOT DETERMINISTIC}}{\Longrightarrow} \tau, \quad \min_{\phi} \left\| \tau - \phi(\overline{\mathbf{U}}) \right\|^2 .$$

We could try to mention that we can include a larger region of the filtered information so that the mapping may become deterministic.

2. Discrepancy between a-priori error and a-posteriori error.

$$\left\| \widehat{\tau} - \tau \right\|^2 , \quad \left\| \overline{\widehat{\mathbf{U}}}(T) - \overline{\mathbf{U}}(T) \right\|^2$$

3. Difficult to combine the OpenFOAM solver with gradient-based optimization algorithms.

Learning the SGS stress model

We test the following three approaches:

1. Directly predict the stress tensor from the input features.

$$\tau = \mathrm{NN}(\overline{\mathbf{U}}, \nabla\overline{\mathbf{U}}, \overline{p}). \tag{3}$$

2. Learn a correction of the constants to the Smagorinsky model.

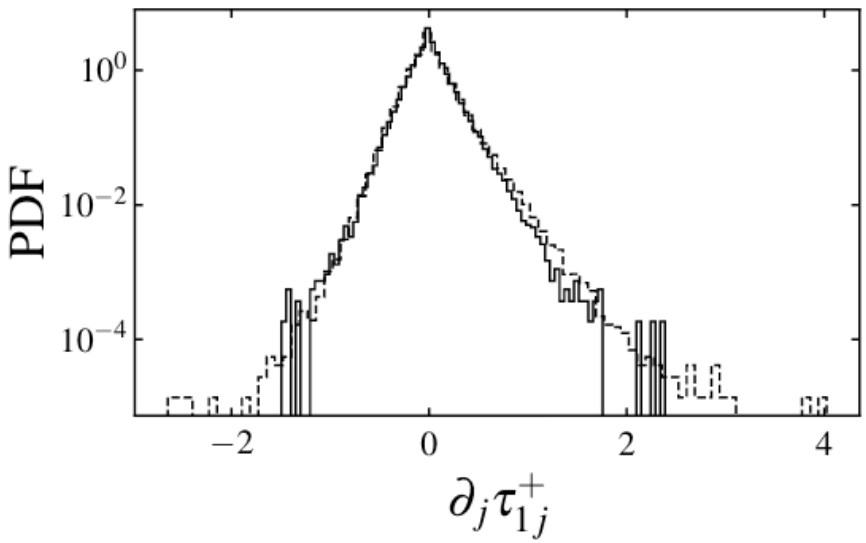
$$\widetilde{C} = \mathrm{NN}(\overline{\mathbf{U}}, \nabla\overline{\mathbf{U}}, \overline{p}) + C, \quad \nu_t = \widetilde{C}\Delta^2 \left|\bar{S}\right|. \tag{4}$$

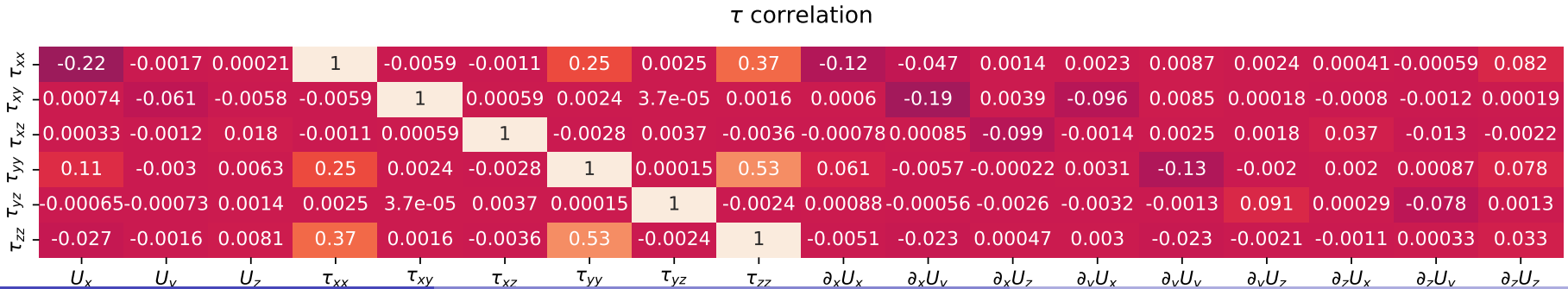
3. Learn a conditional generative model from the input features.

The first approach usually provides a much better a-priori error estimate than the second approach.

Probabilistic SGS stress modeling

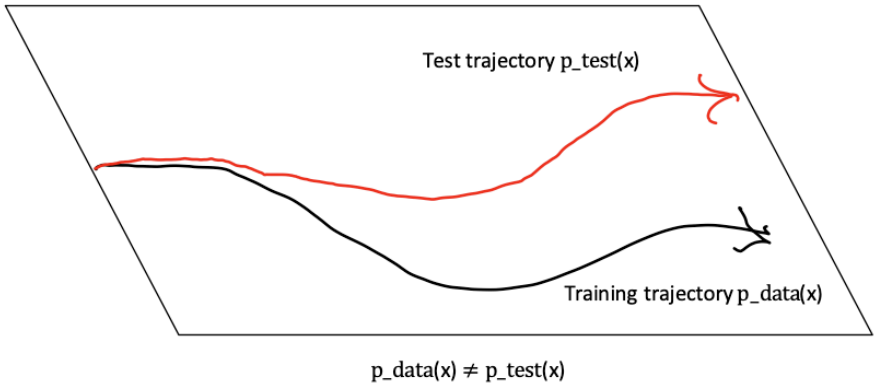
While the first two deterministic stress model can not capture the statistical behavior of the SGS stress, our probabilistic stress model manages this:





A-priori and a-posteriori discrepancy

The inconsistency between the a priori error and a posteriori error arises because the training algorithm does not take the solver dynamics into account.



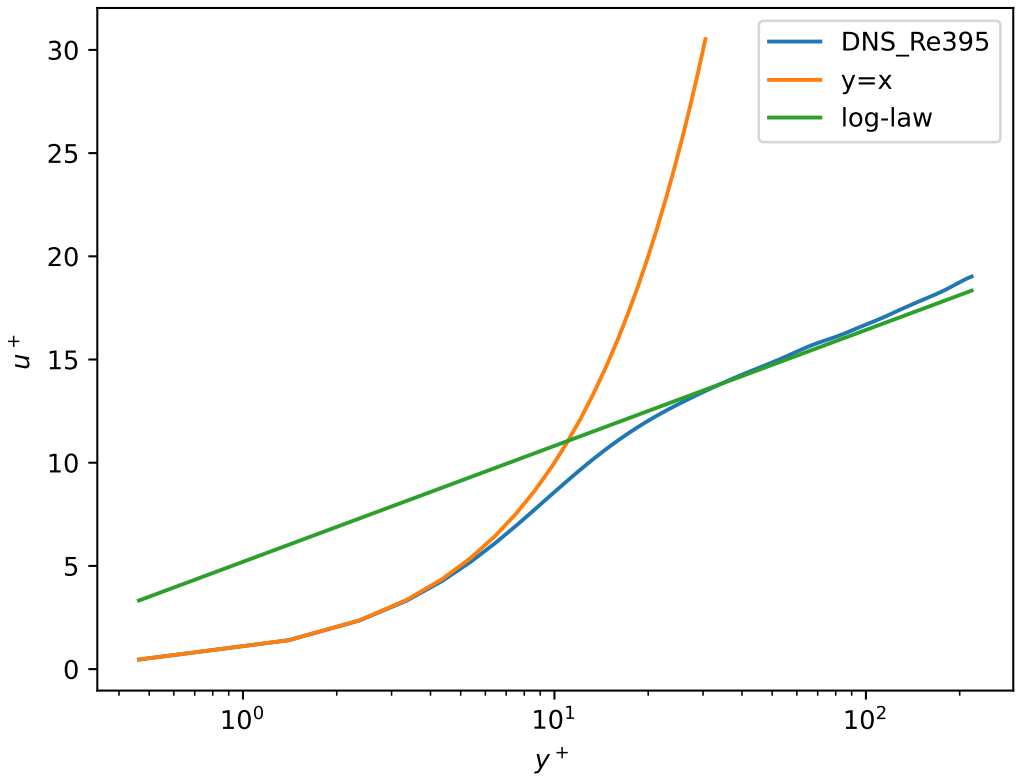
The a-priori and a-posteriori performance are not consistent.

TABLE 3. Network and performance details

	Network inputs	Network outputs	*A priori* correlations	*A posteriori* simulations
NN-1	Local $\bar{S}_{ij}$	$\partial_j \tau_{ij}$	0.6	Stable; varying accuracy
NN-2	19-point stencil $\bar{S}_{ij}$	$\partial_j \tau_{ij}$	0.9	Unstable
NN-3	Local $\bar{S}_{ij}$	$L_i - D_i$ (Eq. 2.4)	0.7	Unstable

Dateset preparation and preprocessing

Judging the quality of the dataset is crucial for our application, which is also very different from the numerical PDE cases as grid convergence tests are inapplicable.



Future work

1. Generate a systematical dataset of the flow around bluffs; train SGS models within the dataset and design algorithms which improves the a-posteriori performance.

2. Deploy to practical problems: Subgrid-scale modeling in large eddy simulation of the urban environment.

3. Investigate the statistical and numerical properties of the data-driven turbulence modeling, with special focus on the dataset characteristic and the relation between data with different resolutions.

References

M. Benjamin, S. Domino, and G. Iaccarino
Neural Networks for Large Eddy Simulations of Wall-bounded Turbulence: Numerical Experiments and Challenges
The European Physical Journal E

J. Zhao and Q. Li (2024)
Mitigating Distribution Shift in Machine Learning-augmented Hybrid Simulation
Arxiv preprint https://arxiv.org/pdf/2401.09259

S. Arisaka and Q. Li (2024)
Accelerating Legacy Numerical Solvers by Non-intrusive Gradient-based Meta-solving
International Conference on Machine Learning 2024