Estimating divergence-free flows via neural networks

Dmitry I. Kabanov, Luis Espath, Raúl F. Tempone

18 March 2021

Outline

- Problem formulation
- How neural networks are used for the problem
- Intro to neural networks
- Example 1: synthetic dataset
- Example 2: real wind dataset from Sweden

Problem formulation

Consider dataset

$$\boldsymbol{D} = \{\boldsymbol{x}_i, \boldsymbol{u}_i\}, \quad i = 1, ..., N$$

where $\boldsymbol{x}_i = (x_i, y_i) \in \mathcal{D} \subset \mathbb{R}^2$ is a spatial point, $\boldsymbol{u}_i = (u_i, v_i) \in \mathbb{R}^2$ is the velocity field at point i.

Assume that the velocity field $\boldsymbol{u}(\boldsymbol{x})$ satisfies the incompressibility condition, at least, approximately:

$$\nabla \cdot \boldsymbol{u}(\boldsymbol{x}) \approx 0.$$

Goal: estimate the velocity field $\boldsymbol{u}(\boldsymbol{x})$ using the above condition.

Problem formulation

Instead of the true velocity field, we seek for estimator $\hat{\boldsymbol{u}}(\boldsymbol{x};\theta)$, parameterized by $\theta \in \mathbb{R}^n$, that satisfies the measurements and the incompressibility condition.

Introduce an additional set of P points, at which the condition is enforced.

Then the problem is find

$$\arg\min_{\theta} \quad \frac{1}{N}\sum_{i=1}^{N}\|\boldsymbol{u}_i - \hat{\boldsymbol{u}}(\boldsymbol{x}_i;\theta)\|_2^2 \; + \; \frac{\gamma}{P}\sum_{i=1}^{P}\left(\nabla \cdot \hat{\boldsymbol{u}}(\boldsymbol{x}_i;\theta)\right)^2 .$$

where γ is a regularization parameter that determines how strongly the incompressibility condition is enforced.

Estimator function

We use a neural network as an estimator function $\hat{\boldsymbol{u}}(\boldsymbol{x};\theta)$.

This neural network is of type "multi-layer perceptron":

$$\hat{\boldsymbol{u}}(\boldsymbol{x};\boldsymbol{\theta}) = A \circ F_{L-1} \circ \cdots \circ F_1(\boldsymbol{x})$$

where

- Hidden layers $F_j : \mathbb{R}^{n_{j-1}} \to \mathbb{R}^{n_j}$ are defined as

  $$F_j\left(\boldsymbol{z}_{j-1}\right) = \sigma\left(\boldsymbol{W}_j \boldsymbol{z}_{j-1} + \boldsymbol{b}_j\right)$$

  with σ being a nonlinear function applied componentwise
- $A(z_L) = W_L z_L + b_L$ the output layer
- Parameter vector $\boldsymbol{\theta}$ is a concatenation of weight matrices $\boldsymbol{W}_j$ and bias vectors $\boldsymbol{b}_j$, $j = 1, \ldots, L$.

Visualization of a neural network????

Should I have this slide?

Other details

- Points where divergence-free condition is enforced, are on the uniform grid
- To solve the optimization problem, we use ADAM optimizer
- Exponential averaging?
- Codes written with libraries Keras and Tensorflow 2

Example 1

Dataset of synthetic noiseless data ??????? Exact field?

$$\{\boldsymbol{x}_i, \boldsymbol{u}_i\}, \quad i = 1, ..., 10$$

where

$$\boldsymbol{u}(x, y) = \begin{pmatrix} \cos x \, \sin y \\ -\sin x \, \cos y \end{pmatrix}$$

and points $\boldsymbol{x}_i = (x_i, y_i)$ are sampled uniformly from domain $[0; 2\pi]^2$.

This dataset is a Taylor–Green 2D vortex fixed in time.

Model performance score

Model performance is measured with R^2 score:

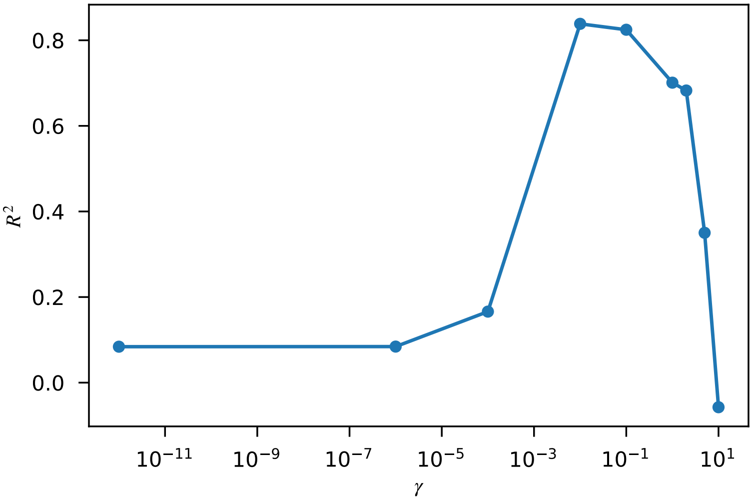
$$R^2 = 1 - \frac{\sum_{i=1}^{N} \|\boldsymbol{u}_i - \hat{\boldsymbol{u}}(\boldsymbol{x};\theta)\|_2^2}{\sum_{i=1}^{N} \|\boldsymbol{u}_i\|_2^2}$$

Interpretation:

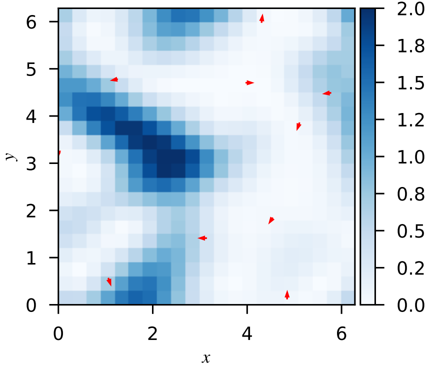
- measure model performance relative to the model that predicts zero velocity field at each point
- perfect model has $R^2 = 1$.

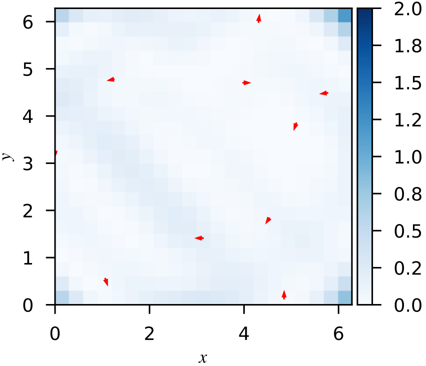
R^2 score versus γ

We train neural network by minimizing misfit + γ div-free condition



Error fields





Thank you!