Estimating divergence-free flows via neural networks

D. I. Kabanov [1] L. Espath [1] J. Kiessling [2] R. F. Tempone [1,3]

18 March 2021, GAMM 91st Annual Meeting

[1]RWTH Aachen, Germany

[2]KTH, Sweden

[3]KAUST, Saudi Arabia

Outline

- Problem formulation
- How neural networks are used for the problem
- 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 divergence-free 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

$$\hat{\boldsymbol{u}}(\boldsymbol{x};\boldsymbol{\theta})$$

parameterized by

$$\boldsymbol{\theta} \in \mathbb{R}^n$$

that satisfies the measurements and the divergence-free condition

$$\arg\min_{\boldsymbol{\theta}} \quad \mathbb{E}\left[\|\boldsymbol{u}(\boldsymbol{x}) - \hat{\boldsymbol{u}}(\boldsymbol{x};\boldsymbol{\theta})\|^2_{L^2(\mathbb{R}^2)}\right] + \gamma\|\nabla \cdot \hat{\boldsymbol{u}}(\boldsymbol{x};\boldsymbol{\theta})\|^2_{L^2(\mathbb{R})}$$

where γ is a regularization parameter that determines how strongly the divergence-free condition is enforced.

Problem formulation

$$\arg\min_{\theta} \quad \mathbb{E}\left[\|\boldsymbol{u}(\boldsymbol{x}) - \hat{\boldsymbol{u}}(\boldsymbol{x};\boldsymbol{\theta})\|^2_{L^2(\mathbb{R}^2)}\right] + \gamma\|\nabla\cdot\hat{\boldsymbol{u}}(\boldsymbol{x};\boldsymbol{\theta})\|^2_{L^2(\mathbb{R})}$$

Finite number of measurements to evaluate the first term.

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

Then the problem is to find

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

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":

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

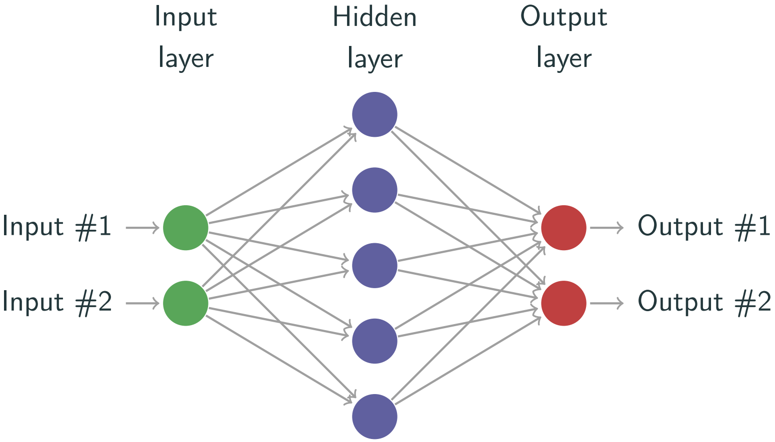
where

- 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 $\mathbb{R}^2 \to \mathbb{R}^2$



[https://texample.net/tikz/examples/neural-network/, with modifications]

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 of the parameters is used for prediction
- Code is written using libraries Keras and Tensorflow 2

Example 1

Example 1

Dataset of synthetic noiseless data

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

where

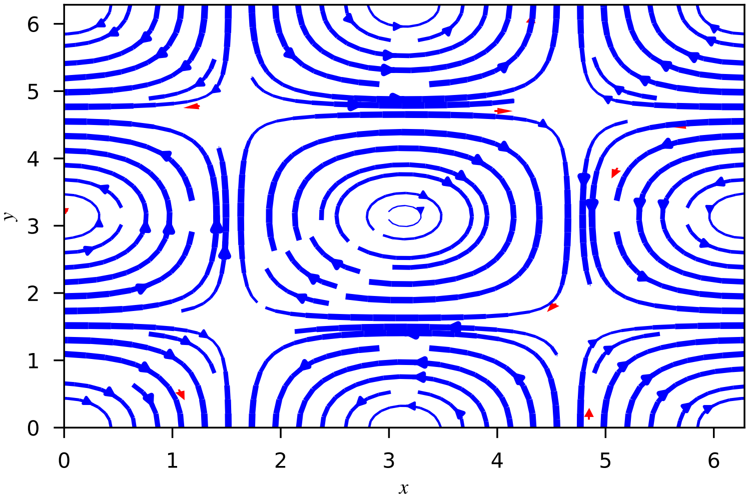
$$\boldsymbol{u}(x, y) = \begin{pmatrix} \cos x \, \sin y \\ -\sin x \, \cos y \end{pmatrix}$$

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.

Test data are 21×21 uniform grid in this domain.

Example 1, true field



Model performance score

Model performance is measured with R^2 score:

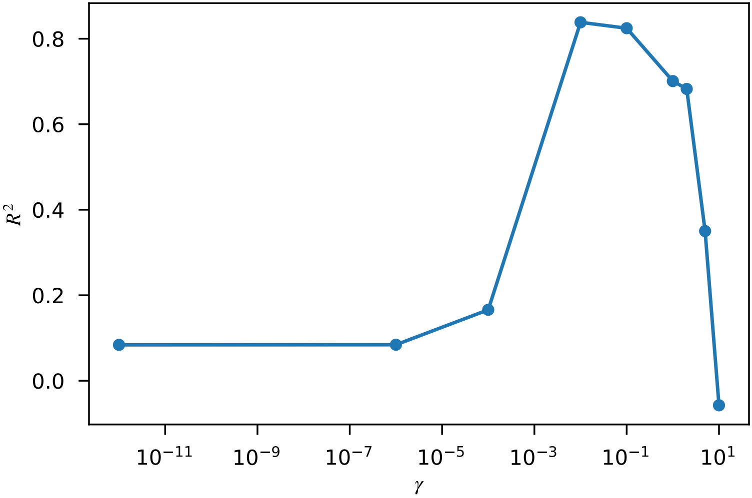
$$R^2 = 1 - \frac{\sum_{i=1}^{N} \|\boldsymbol{u}_i - \hat{\boldsymbol{u}}(\boldsymbol{x};\theta)\|_2^2}{\sum_{i=1}^{N} \|\boldsymbol{u}_i\|_2^2}$$

Interpretation:

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

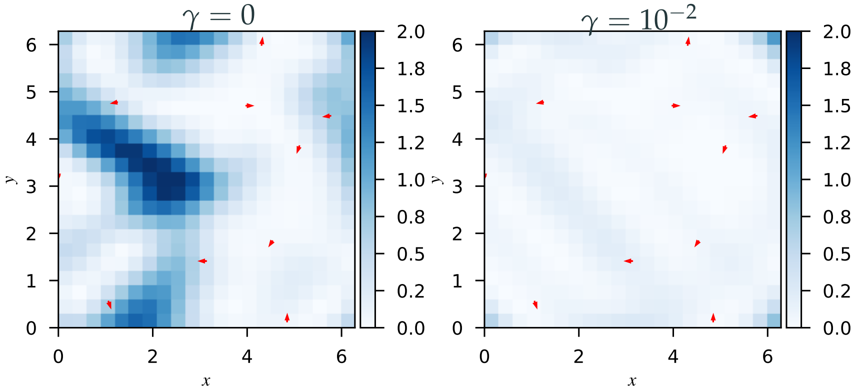
R^2 score versus γ

Training by minimizing misfit + γ div-free condition



Error fields

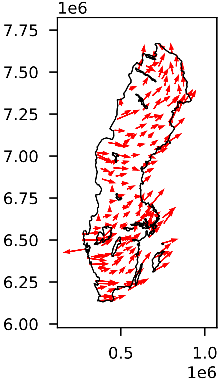
Training by minimizing misfit + γ div-free condition



Example 2

Example 2: Reconstruction of wind field over Sweden

- We apply a neural network with div-free regularization to the problem of reconstruction of the wind fields over Sweden
- Data are collected from 165 weather stations in 2018
- Measurements are hourly but a lot of missing data points form different stations



Basic model to compare with

As a basic model, we use Inverse Distance Weighting (averaging) model, which estimates velocity at point $\boldsymbol{x}$ as

$$\hat{\boldsymbol{u}}_{\mathsf{IDW}}(\boldsymbol{x}) = \frac{\sum_{i=1}^{N} W(r_{i,j})\boldsymbol{u}(\boldsymbol{x})}{\sum_{i=1}^{N} W(r_{i,j})}$$

where W is a weight: decaying function of distance $r_{i,j}$.

Performance score

To assess the quality of the model, we estimate the prediction error by selecting at random from the dataset T=500 time snapshots and computing an average R^2 score using five-fold cross-validation:

$$R^2 = 1 - \frac{\sum_{i=1}^{T} \sum_{j=1}^{5} \frac{1}{N'_{ij}} \sum_{k=1}^{N'_{ij}} \|\boldsymbol{u}_{ijk} - \hat{\boldsymbol{u}}_{ijk}\|_2^2}{\sum_{i=1}^{T} \sum_{j=1}^{5} \frac{1}{N'_{ij}} \sum_{k=1}^{N'_{ij}} \|\boldsymbol{u}_{ijk}\|_2^2}$$

where N'_{ij} denotes number of test measurements for the ith time snapshot and jth cross-validation fold.

That is, we train $500 \times 5 = 2500$ neural networks.

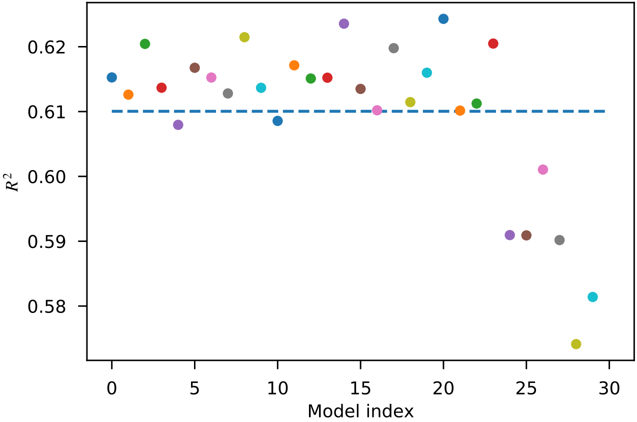
Search for neural network better than the base model

We train neural networks with configuration specified by a triplet:

Divergence-free multiplier: $\gamma \in \{0, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$

Number of neurons in the 1st hidden layer: $h_1 \in \{20, 30\}$

Number of neurons in the 2nd hidden layer: $h_2 \in \{0, 10, 20\}$



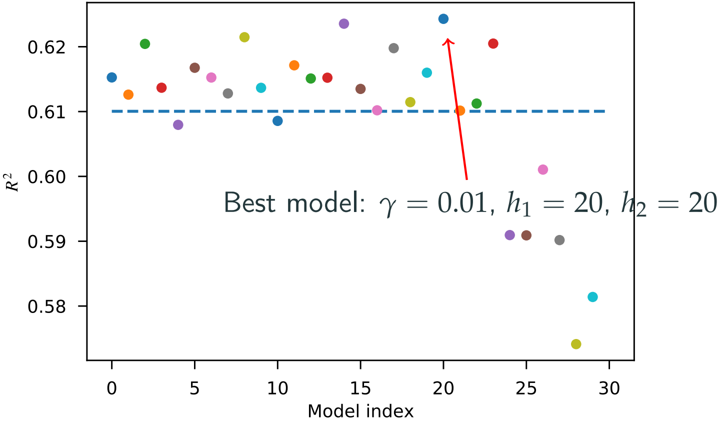
Search for neural network better than the base model

We train neural networks with configuration specified by a triplet:

Divergence-free multiplier: $\gamma \in \{0, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$

Number of neurons in the 1st hidden layer: $h_1 \in \{20, 30\}$

Number of neurons in the 2nd hidden layer: $h_2 \in \{0, 10, 20\}$



Summary

- We implemented a neural network which learns divergence-free velocity fields
- True div-free data with scarce measurements greatly benefit from such network: prediction error decreases 4x
- Application to real data of wind velocity measurements over Sweden does not show much advantage of the neural network over simple Inverse Distance Weighting model. Moreover, div-free regularizer improves prediction abilities of the neural network only marginally

Thank you!