Developing turbulence models using Machine Learning in Python

Lars Davidson
Division of Fluid Dynamics, Mechanics and Maritime Sciences (M2)
Chalmers University of Technology, Gothenburg, Sweden

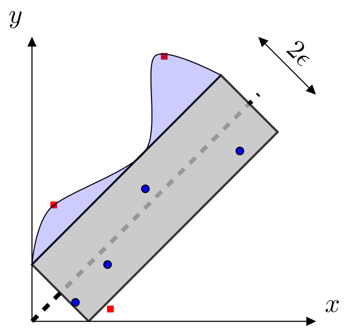

Figure 1: Sketch of support vector regression (SVR). Data point inside (●) and outside (🟥) the tube (gray area). The gray dashed line is the hyperplane (regression plane) predicted by (the non-linear) `svrLINEAR`. A large C enlarges the area of the tube (blue area)

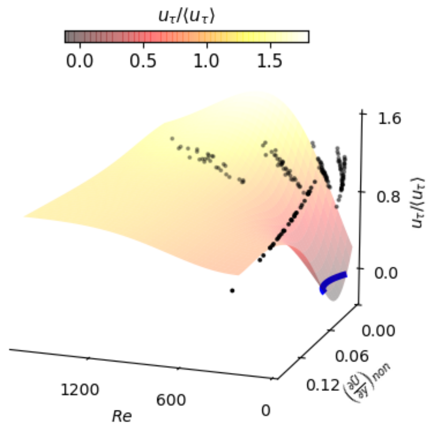

Figure 2: Colored surface is the hypersurface (instead of hyperplane since it is non-liner). In [1] I get non-physical (negative) friction velocities, u_τ, predicted by `svr-LINEAR` when I use too small "slack" (i.e. too large C). Thick blue line: $u_\tau = 0$.

Background

Machine learning is a method where known data are used for teaching the algorithm to classify a set of data. The data may be photographs where the machine learning algorithm should recognize, for example, traffic lights or traffic signs [2]. Another example may be ECG signals where the machine learning algorithm should recognize certain unhealthy conditions of the heart [3]. A third example is detecting fraud for credit card payments [4]. Machine learning methods such as Support Vector Machines (SVM) and neural networks are often used for solving this type of problems.

The examples above are classification problems using supervised learning. However, in the present project input and output are numerical values. In this case, machine learning in the form of regression methods should be used [3]; we will use support vector regression (SVR) methods available in Python.

In SVR a regression multi-dimensional "surface" is created which has as many dimensions as number of influence parameters. Let's make a simple example. In Fig. 1 there is one influence parameter, x, and one parameter to predict, y. Two main input parameters may be given to the SVR methods. The first is ϵ which determines the width of the tube around the hyperplane [1]. Points that lie inside this tube are considered as correct predictions and are not penalized by the algorithm. The

[1]A hyperplane is a plane whose number of dimension is the same the number of influence parameters. For example, a two-dimensional hyperplane has two influence parameters.

support vectors are the points that lie outside the tube. The second parameter given to SVR models is the C value. It controls the "slack" (ξ), see Fig. 1, which is the distance to points outside the tube. If C is increased the size of the tube is increased so that some or all of the data points are located inside the tube. It was found in [1] that the parameter C may have a large influence on the form of the hyperplane/hypersurface and give non-physical friction velocities, see Fig. 2.

Methodology

The time-averaged Navier-Stokes for $\bar{u}$ in two dimensions read

$$\bar{u}\frac{\partial \bar{u}}{\partial x} + \bar{v}\frac{\partial \bar{u}}{\partial y} = -\frac{1}{\rho}\frac{\partial \bar{p}}{\partial x} + \nu\left(\frac{\partial^2 \bar{u}}{\partial x^2} + \frac{\partial^2 \bar{u}}{\partial y^2}\right) - \frac{\partial \overline{u'u'}}{\partial x} - \frac{\partial \overline{u'v'}}{\partial y}$$

The two last terms include the unknown turbulent Reynolds stresses which must be modeled. For example, In the $k-\varepsilon$ model, the shear stress, for example, is modeled as

$$\overline{u'v'} = -\nu_t\left(\frac{\partial \bar{u}}{\partial y} + \frac{\partial \bar{v}}{\partial x}\right), \quad \nu_t = C_\mu \frac{k^2}{\varepsilon} \tag{1}$$

where k is the turbulent kinetic energy, ε is its dissipation and C_μ is assumed to be constant, $C_\mu = 0.09$.

- I will provide "exact" solutions (i.e. solutions of Direct Numerical Simulations [DNS]) of both simple flows (channel and boundary flows) and more complex flows including re-circulation.
- The students will then use `svr` and/or `svrLINEAR` in Machine Learning (ML) for finding a more general expression for C_μ. Influence parameters may be velocity gradients (coordinate-invariant) and/or the turbulent time-scale, k/ε, both functions of x and y). The output parameter will be $C_\mu = C_\mu(x,y)$. An example of Python code is found here. When you execute the Python code you find that the ML-based turbulence model is indeed better than the standard model.
- Next, the students will perform simple 2D CFD simulations (using my Python CFD code **pyCALC-RANS** [5]) comparing the original turbulence model (Eq. 1) with the improved ML turbulence model. Students may choose to use a commercial code such as STAR-CCM+ or ANSYS instead of **pyCALC-RANS** .
- Equation 1 is a very simple model for the Reynolds shear stress. More complex models include non-linear velocity gradient, see Chapter 14 in my eBook [6]. Then there are six coefficients to optimize ($c_1, c_2, \ldots c_6$)

Objective

The object is to

- learn how to use Machine Learning methods such as `svr` and `svrLINEAR` and to improve existing turbulence models.
- implement the new turbulence model(s) in a CFD code (e.g. **pyCALC-RANS**)

Pre-requisites

The students should have an interest in either fluid mechanics or Machine Learning. You should also have basic knowledge in Python (if you don't, please consider to take the course *DAT171 – Object-oriented programming in Python* in Study Period 3).

Target Group

Teknisk fysik, Teknisk design, maskinteknik, teknisk matematik, informatonsteknik, kemiteknik, kemiteknik med fysik, datateknik, eller motsvarande

Group Size

4-6 students

Supervisor

Lars Davidson lada@chalmers.se

Examiner

Niklas Andersson, nian@chalmers.se

References

[1] L. Davidson. Using Machine Learning for formulating new wall functions for Large Eddy Simulation: A first attempt. Technical report, Division of Fluid Dynamics, Dept. of Mechanics and Maritime Sciences, Chalmers University of Technology, Gothenburg, 2018.

[2] Sudarshana S Rao and Santosh R Desai. Machine learning based traffic light detection and ir sensor based proximity sensing for autonomous cars. In *Proceedings of the International Conference on IoT Based Control Networks & Intelligent Systems – ICICNIS*, 2021. URL `http://dx.doi.org/10.2139/ssrn.3883931`.

[3] Andreas Lindholm, Niklas Wahlström, Fredrik Lindsten, and Thomas Schön. *Machine Learning: A First Course for Engineers and Scientists*. Cambridge University Press, 03 2022. ISBN 9781108843607. doi: 10.1017/9781108919371. URL `http://smlbook.org/`.

[4] Menneni Rachana, Jegadeesan Ramalingam, Gajula Ramana, Adigoppula Tejaswi, Sagar Mamidala, and G Srikanth. Fraud detection of credit card using machine learning. *GIS-Zeitschrift für Geoinformatik*, 8:1421–1436, 10 2021.

[5] L. Davidson. pyCALC-RANS: a 2D Python code for RANS. Division of Fluid Dynamics, Dept. of Mechanics and Maritime Sciences, Chalmers University of Technology, Gothenburg, 2021.

[6] L. Davidson. Fluid mechanics, turbulent flow and turbulence modeling. eBook, Division of Fluid Dynamics, Dept. of Mechanics and Maritime Sciences, Chalmers University of Technology, Gothenburg, 2021.