Using Machine Learning for formulating new wall functions for Large Eddy Simulation: A Second Attempt

Lars Davidson

Division of Fluid Dynamics, Dept. Mechanics and Maritime Sciences
Chalmers University of Technology, Gothenburg, Sweden.

presented at

Workshop on hybrid RANS-LES, FSI and aeroacoustics

Saab Aeronautics, Linköping, 6 December, 2022.

Machine Learning

- Machine learning (ML) is a method where known data are used for teaching the algorithm to classify another set of data.

Machine Learning

- Machine learning (ML) is a method where known data are used for teaching the algorithm to classify another set of data.
  - Photographs where the machine learning algorithm should recognize, e.g., traffic lights [4].

Machine Learning

- Machine learning (ML) is a method where known data are used for teaching the algorithm to classify another set of data.
  - Photographs where the machine learning algorithm should recognize, e.g., traffic lights [4].
  - ECG signals where the machine learning algorithm should recognize certain unhealthy conditions of the heart [2].

Machine Learning

- Machine learning (ML) is a method where known data are used for teaching the algorithm to classify another set of data.
  - Photographs where the machine learning algorithm should recognize, e.g., traffic lights [4].
  - ECG signals where the machine learning algorithm should recognize certain unhealthy conditions of the heart [2].
  - Detecting fraud for credit card payments [3].

Machine Learning

- Machine learning (ML) is a method where known data are used for teaching the algorithm to classify another set of data.
  - Photographs where the machine learning algorithm should recognize, e.g., traffic lights [4].
  - ECG signals where the machine learning algorithm should recognize certain unhealthy conditions of the heart [2].
  - Detecting fraud for credit card payments [3].
  - Machine learning methods such as Support Vector Machines (SVM) and neural networks are used for solving this type of problems.

Machine Learning

- Machine learning (ML) is a method where known data are used for teaching the algorithm to classify another set of data.
  - Photographs where the machine learning algorithm should recognize, e.g., traffic lights [4].
  - ECG signals where the machine learning algorithm should recognize certain unhealthy conditions of the heart [2].
  - Detecting fraud for credit card payments [3].
  - Machine learning methods such as Support Vector Machines (SVM) and neural networks are used for solving this type of problems.
  - Through as much data as possible at ML?

Machine Learning

- Machine learning (ML) is a method where known data are used for teaching the algorithm to classify another set of data.
  - Photographs where the machine learning algorithm should recognize, e.g., traffic lights [4].
  - ECG signals where the machine learning algorithm should recognize certain unhealthy conditions of the heart [2].
  - Detecting fraud for credit card payments [3].
  - Machine learning methods such as Support Vector Machines (SVM) and neural networks are used for solving this type of problems.
  - Through as much data as possible at ML?
- In my case, input and output are numerical values. Regression methods should then be used [2]; I use support vector regression (SVR) methods available in Python.

Training: I need a target database

$$\frac{\partial \bar{v}_i}{\partial x_i} = 0$$

$$\frac{\partial \bar{v}_i}{\partial t} + \frac{\partial}{\partial x_j}(\bar{v}_i \bar{v}_j) = -\frac{\partial \bar{p}}{\partial x_i} + \frac{\partial}{\partial x_j}\left[(\nu + \nu_{sgs})\frac{\partial \bar{v}_i}{\partial x_j}\right]$$

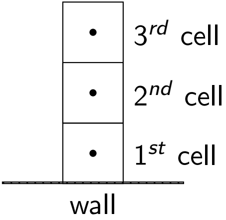
- Fully-developed Channel flow
- IDDES. $96 \times 96 \times 96$, Reynolds number is 5 200
- Database: average in x and z

$$\bar{U}_{1^{st}}(x,z) = \frac{1}{\Delta X \Delta Z}\int_{x,z}^{x+\Delta X, z+\Delta Z} \bar{u}\, dxdz$$

$$\bar{u}_\tau(x,z) = \frac{1}{\Delta X \Delta Z}\int_{x,z}^{x+\Delta X, z+\Delta Z} u_\tau\, dxdz$$

Machine Learning

- LES with wall functions: the object is to develop a model for the wall shear stress, $\tau_w = \rho u_\tau^2$

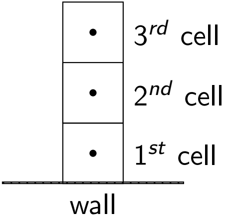


1^{st} cell	$\langle \Delta y^+ \rangle$
Location 1	12
Location 2	31
Location 3	49
Location 4	66
Location 5	76
Location 6	88
Location 7	135
Location 8	155
Location 9	207

300 independent instantaneous samples of $\bar{U}$ stored at all 3×9 cells

Machine Learning

- LES with wall functions: the object is to develop a model for the wall shear stress, $\tau_w = \rho u_\tau^2$
- Input data: U_P, y_P, $\partial \bar{U}/\partial y$, $\partial^2 \bar{U}/\partial^2 y$

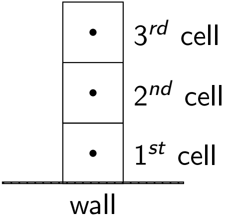


1^{st} cell	$\langle \Delta y^+ \rangle$
Location 1	12
Location 2	31
Location 3	49
Location 4	66
Location 5	76
Location 6	88
Location 7	135
Location 8	155
Location 9	207

300 independent instantaneous samples of $\bar{U}$ stored at all 3×9 cells

Machine Learning

- LES with wall functions: the object is to develop a model for the wall shear stress, $\tau_w = \rho u_\tau^2$
- Input data: U_P, y_P, $\partial \bar{U}/\partial y$, $\partial^2 \bar{U}/\partial^2 y$
- output data: u_τ

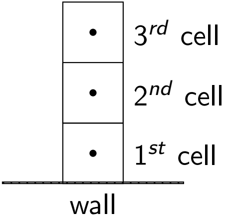


1^{st} cell	$\langle \Delta y^+ \rangle$
Location 1	12
Location 2	31
Location 3	49
Location 4	66
Location 5	76
Location 6	88
Location 7	135
Location 8	155
Location 9	207

300 independent instantaneous samples of $\bar{U}$ stored at all 3×9 cells

Machine Learning

- LES with wall functions: the object is to develop a model for the wall shear stress, $\tau_w = \rho u_\tau^2$
- Input data: U_P, y_P, $\partial \bar{U}/\partial y$, $\partial^2 \bar{U}/\partial^2 y$
- output data: u_τ
- Non-dimensional: $\frac{u_\tau}{\langle u_\tau \rangle} = f\left(Re, y^+, T\partial \bar{U}/\partial y, \partial^2 \bar{U}/\partial y^2/(\bar{U}T^2)\right)$

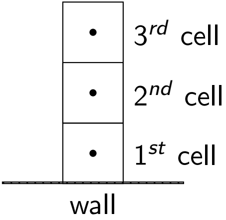


1^{st} cell	$\langle \Delta y^+ \rangle$
Location 1	12
Location 2	31
Location 3	49
Location 4	66
Location 5	76
Location 6	88
Location 7	135
Location 8	155
Location 9	207

300 independent instantaneous samples of $\bar{U}$ stored at all 3×9 cells

Machine Learning

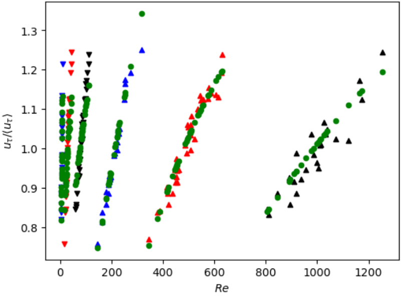
- LES with wall functions: the object is to develop a model for the wall shear stress, $\tau_w = \rho u_\tau^2$
- Input data: U_P, y_P, $\partial \bar{U}/\partial y$, $\partial^2 \bar{U}/\partial^2 y$
- output data: u_τ
- Non-dimensional: $\frac{u_\tau}{\langle u_\tau \rangle} = f\left(Re, y^+, T\partial \bar{U}/\partial y, \partial^2 \bar{U}/\partial y^2/(\bar{U}T^2)\right)$
- $T = \nu/\bar{U}^2$



1^{st} cell	$\langle \Delta y^+ \rangle$
Location 1	12
Location 2	31
Location 3	49
Location 4	66
Location 5	76
Location 6	88
Location 7	135
Location 8	155
Location 9	207

300 independent instantaneous samples of $\bar{U}$ stored at all 3×9 cells

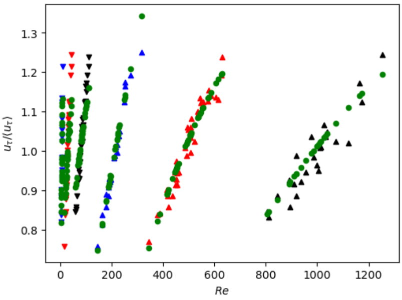
Predicted output using ML: 1st attempt



(A) ▲: IDDES, Location 1; ▲: IDDES, Location 2; ▲: IDDES, Location 3; ▼: IDDES, Location 4; ▼: IDDES, Location 5; ▼: IDDES, Location 6. ○: svr.

- Output on y axis

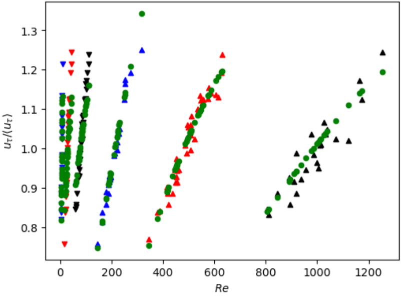
Predicted output using ML: 1st attempt



(A) ▲: IDDES, Location 1; ▲: IDDES, Location 2; ▲: IDDES, Location 3; ▼: IDDES, Location 4; ▼: IDDES, Location 5; ▼: IDDES, Location 6. ○: svr.

- Output on y axis
- Input on x axis

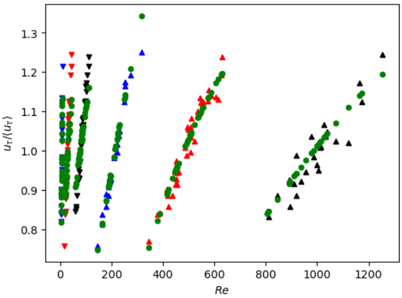
Predicted output using ML: 1st attempt



(A) ▲: IDDES, Location 1; ▲: IDDES, Location 2; ▲: IDDES, Location 3; ▼: IDDES, Location 4; ▼: IDDES, Location 5; ▼: IDDES, Location 6. ○: svr.

- Output on y axis
- Input on x axis
- Location 1 – 6 of data

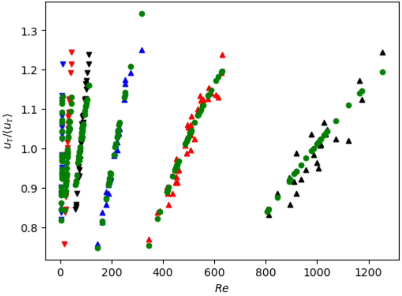
Predicted output using ML: 1st attempt



(A) ▲: IDDES, Location 1; ▲: IDDES, Location 2; ▲: IDDES, Location 3; ▼: IDDES, Location 4; ▼: IDDES, Location 5; ▼: IDDES, Location 6. ○: svr.

- Output on y axis
- Input on x axis
- Location 1 – 6 of data
- Difficult to interpolate

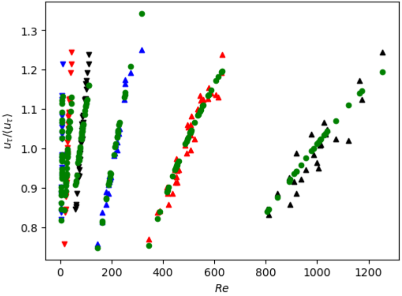
Predicted output using ML: 1st attempt



(A) ▲: IDDES, Location 1; ▲: IDDES, Location 2; ▲: IDDES, Location 3; ▼: IDDES, Location 4; ▼: IDDES, Location 5; ▼: IDDES, Location 6. ○: svr.

- Output on y axis
- Input on x axis
- Location 1 – 6 of data
- Difficult to interpolate
- Remedy: I included $\langle y^+ \rangle$ as input parameter = Location

Predicted output using ML: 1st attempt



(A) ▲: IDDES, Location 1; ▲: IDDES, Location 2; ▲: IDDES, Location 3; ▼: IDDES, Location 4; ▼: IDDES, Location 5; ▼: IDDES, Location 6. ○: svr.

- Output on y axis
- Input on x axis
- Location 1 – 6 of data
- Difficult to interpolate
- Remedy: I included $\langle y^+\rangle$ as input parameter = Location
- $\frac{u_\tau}{\langle u_\tau\rangle} = f\left(Re, \langle y^+\rangle\right)$

Bad choice of input/output: 2nd attempt

- Traditional wall laws:
  $\frac{U}{u_\tau} = f\left(\frac{u_\tau y}{\nu}\right)$

Bad choice of input/output: 2nd attempt

- Traditional wall laws:
  $\frac{U}{u_\tau} = f\left(\frac{u_\tau y}{\nu}\right)$
- Do the same in ML

Bad choice of input/output: 2nd attempt

- Traditional wall laws:
  $\frac{U}{u_\tau} = f\left(\frac{u_\tau y}{\nu}\right)$
- Do the same in ML

$$\begin{aligned} y^+ &: \quad \text{influence par.} \\ U^+ &: \quad \text{output par.} \\ u_\tau &: \quad \bar{u}/U^+ \end{aligned}$$

Bad choice of input/output: 2nd attempt

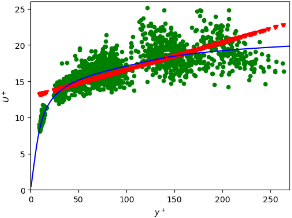
- Traditional wall laws:
  $\frac{U}{u_\tau} = f\left(\frac{u_\tau y}{\nu}\right)$
- Do the same in ML

$$
\begin{aligned}
y^+ &: \quad \text{influence par.}\\
U^+ &: \quad \text{output par.}\\
u_\tau &: \quad \bar{u}/U^+
\end{aligned}
$$

$$
\begin{aligned}
\rho u_\tau^2 &: \bar{u}\,\text{eq.}\\
C_\mu^{-1/2} u_\tau^2 &: k\,\text{eq.}\\
\frac{u_\tau^3}{\kappa y} &: \varepsilon\,\text{eq.}
\end{aligned}
$$

Bad choice of input/output: 2nd attempt

- Traditional wall laws:
  $\frac{U}{u_\tau} = f\left(\frac{u_\tau y}{\nu}\right)$
- Do the same in ML



—: $\langle\bar{u}\rangle$, IDDES; ▼: `svrLINEAR`: ●: IDDES, test data. 9% normalized error.

$$y^+ \quad : \quad \text{influence par.}$$
$$U^+ \quad : \quad \text{output par.}$$
$$u_\tau \quad : \quad \bar{u}/U^+$$

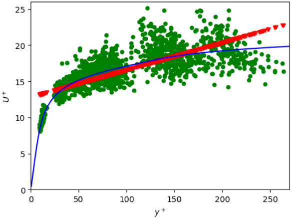
$$\rho u_\tau^2 \quad : \quad \bar{u}\ \text{eq.}$$
$$C_\mu^{-1/2} u_\tau^2 \quad : \quad k\ \text{eq.}$$
$$\frac{u_\tau^3}{\kappa y} \quad : \quad \varepsilon\ \text{eq.}$$

Bad choice of input/output: 2nd attempt

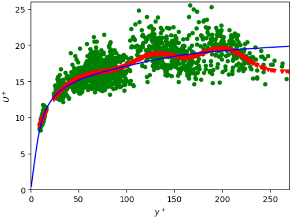
- Traditional wall laws: $\frac{U}{u_\tau} = f\left(\frac{u_\tau y}{\nu}\right)$
- Do the same in ML

$$
\begin{aligned}
y^+ &: \quad \text{influence par.}\\
U^+ &: \quad \text{output par.}\\
u_\tau &: \quad \bar{u}/U^+
\end{aligned}
$$

$$
\begin{aligned}
\rho u_\tau^2 &: \quad \bar{u}\ \text{eq.}\\
C_\mu^{-1/2} u_\tau^2 &: \quad k\ \text{eq.}\\
\frac{u_\tau^3}{\kappa y} &: \quad \varepsilon\ \text{eq.}
\end{aligned}
$$



: $\langle\bar{u}\rangle$, IDDES; ▼: `svrLINEAR`: ●: IDDES, test data. 9% normalized error.



: $\langle\bar{u}\rangle$, IDDES; ▼: `svr`; ●: IDDES, test data. 9% normalized error

Standard wall functions

- The machine-learning wall functions will be compared to wall functions based on Reichardt's law

$$\frac{\bar{u}_P}{u_\tau} \equiv U^+ = \frac{1}{\kappa}\ln(1-0.4y^+) + 7.8\left[1-\exp\left(-y^+/11\right) - (y^+/11)\exp\left(-y^+/\right.\right.$$

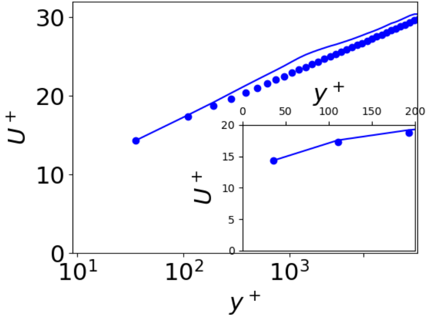
- The friction velocity is then obtained by solving the implicit equation

$$u_\tau - \bar{u}_P\left(\ln(1-0.4y^+)/\kappa + 7.8\left[1-\exp\left(-y^+/11\right) - (y^+/11)\exp\left(-y^+/3\right)\right]\right)^{-1} = 0$$

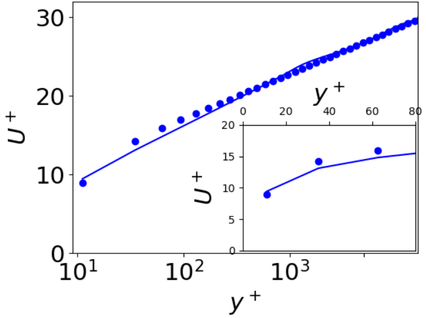
using the Newton-Raphson method `scipy.optimize.newton` in Python.

- $\bar{u}_P$ denotes the wall-parallel velocity in the first, second or third wall-adjacent cell.

Results, channel flow, ML, $Re_\tau = 16\,000$



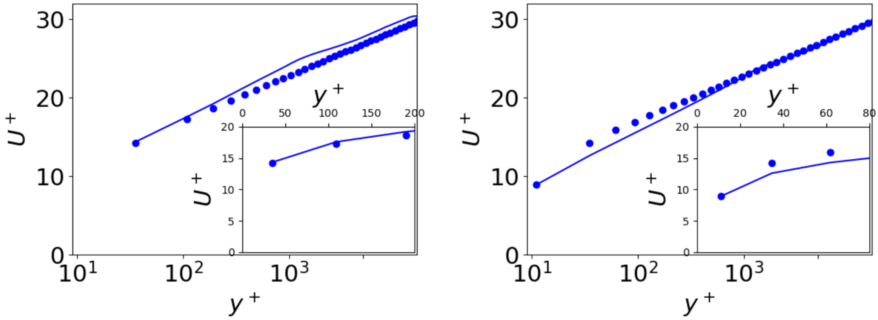
(A) $N_y = 66$, stretching 11%.



(B) $N_y = 68$, stretching 14.7%.

Figure: Channel flow. svr. $Re_\tau = 16\,000$. Velocity. •: Reichardt's law.

Reichardt's wall function, $Re_\tau = 16\,000$

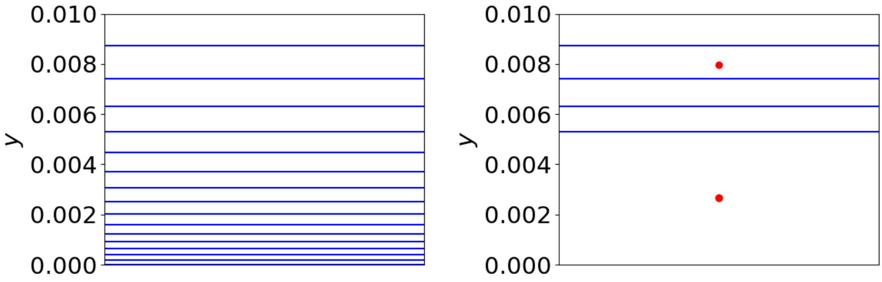


(A) $N_y = 66$, stretching 11%.

(B) $N_y = 68$, stretching 14.7%.

Figure: Channel flow. Reichardt's wall function. $Re_\tau = 16\,000$. Velocity.
•: Reichardt's law.

New Grid Strategy

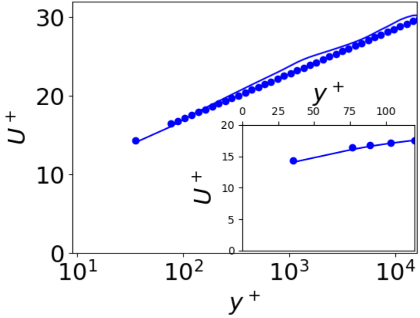


(A) Low-Re number IDDES grid. (B) Wall function grid. New grid strategy.

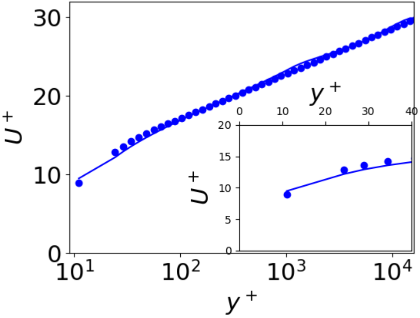
Figure: Different grids. ——— : grid lines.

- This strategy was used in [1] for channel flow and impinging jets (RANS)

Channel flow, ML, $Re_\tau = 16\,000$, new grid



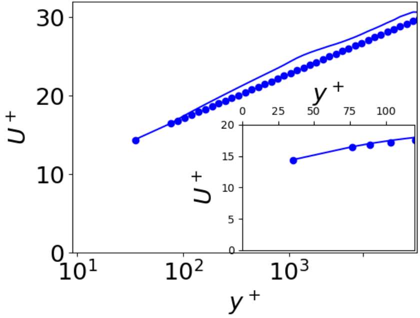
(A) $N_y = 78$.



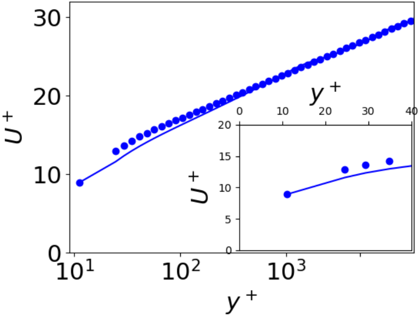
(B) $N_y = 92$.

FIGURE: Channel flow. $Re_\tau = 16\,000$. Velocity. svr. •: Reichardt's law.

Reichardt's wall function, $Re_\tau = 16\,000$



(A) $N_y = 78$.



(B) $N_y = 92$.

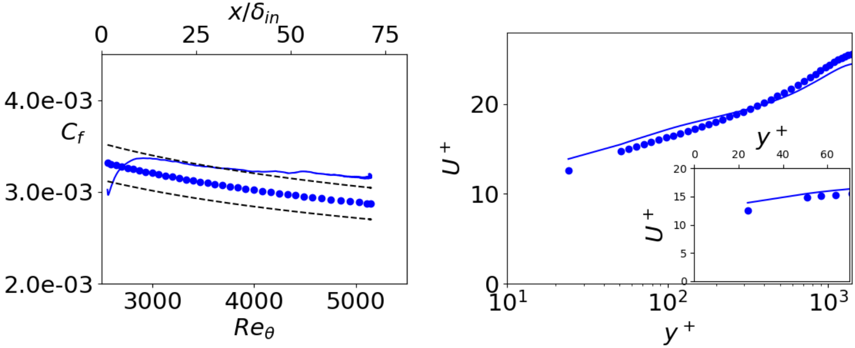
Figure: Channel flow. $Re_\tau = 16\,000$. Velocity. Reichardt's wall function.
●: Reichardt's law.

Developing Boundary Layer Flow

- $Re_\theta = U_{free}\theta/\nu = 2\,550$ at the inlet.
- Domain $(96 \times 7 \times 5)\delta_{in}$.
- Grid $(550 \times 82 \times 64)$.

Developing Boundary Layer Flow

- u_τ computed using 3^{rd} cell

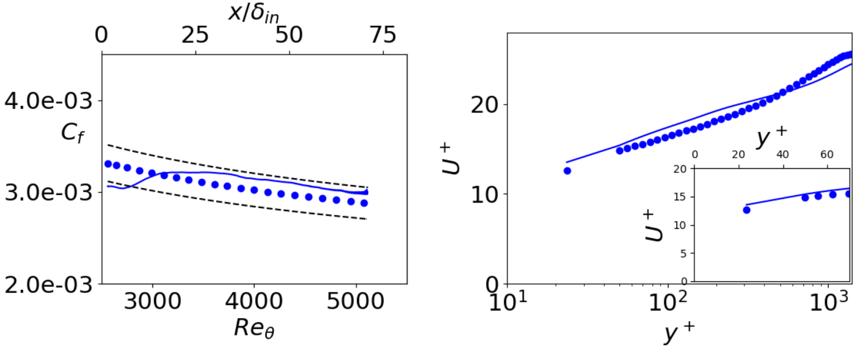


(A) Skin friction.

(B) Velocity at $Re_\theta = 4\,000$. Markers: DNS [5]

Figure: Boundary layer flow. $Re_\theta = 2\,500$ at inlet. `svr`. $N_y = 82$

Developing Boundary Layer Flow, $2\Delta x$, $2\Delta z$



(A) Skin friction.

(B) Velocity at $Re_\theta = 4\,000$. Markers: DNS [5]

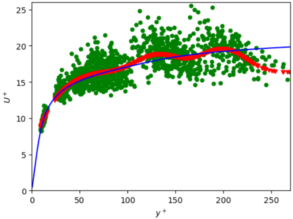
Figure: Boundary layer flow. svr. $N_y = 82$, $N_k = 32$, $\Delta x_{in} = 2\Delta x_{in,base}$

Conclusions

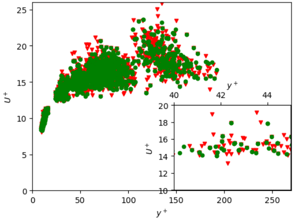
- Machine Learning (`svr`) wall functions have been developed
- Good results for channel flow placing the wall-adjacent cell at different locations
- Good results for developing boundary layer flow

Attempt 3? I

- Instantaneous data are used for training `svr`
- `svr` finds the time-averaged regression line (shown by ▼ in Fig. A)
- If I want *instantaneous* u_τ, I could find nearest neighbour (shown by ● in Fig. B)



(A) ——— : $\langle \bar{u} \rangle$, IDDES; ▼: `svr`: ●: IDDES, target data. 9% normalized error.



(B) Nearest neighbor using Python's `scipy.spatial.KDTree` ▼: KDTree; ●: IDDES, target data; 0.7% normalized error.

Attempt 3? II

- I have trained `svr` using instantaneous data but the model gives time-averaged regression line
- Maybe better to train `svr` using time-averaged data.
  - In this case many time-averaged $\langle \bar{u} \rangle$ (and $\langle \bar{p} \rangle$) profiles can be used for training
  - Maybe `svm` (Support Vector Machines) and/or Neural Network could be used to find the most appropriate profile

References I

[1] J.-A. Bäckar and L. Davidson. Evaluation of numerical wall functions on the axisymmetric impinging jet using OpenFOAM. *International Journal of Heat and Fluid Flow*, 67:27–42, 2017.

[2] Andreas Lindholm, Niklas Wahlström, Fredrik Lindsten, and Thomas Schön. *Machine Learning: A First Course for Engineers and Scientists*. Cambridge University Press, 2022.

[3] 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.

[4] 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.

References II

[5] J.A. Sillero, J. Jimenez, and R.D. Moser. One-point statistics for turbulent wall-bounded flows at Reynolds numbers up to $\delta^+ \simeq 2000$. *Physics of Fluids*, 25(105102), 2014.