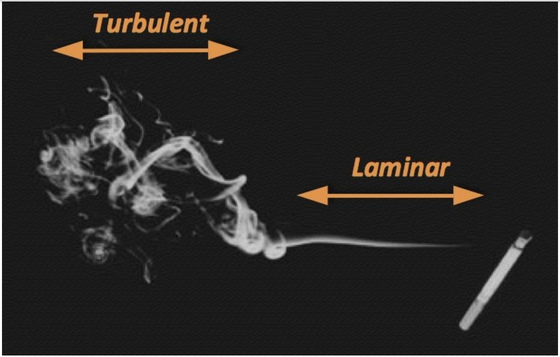
# Applying Machine Learning Approaches for Understanding Turbulent Flow

**Reza Momenifar & Long Zhang**

Health and Environmental Data Science (CEE.690.05)

Spring 2019

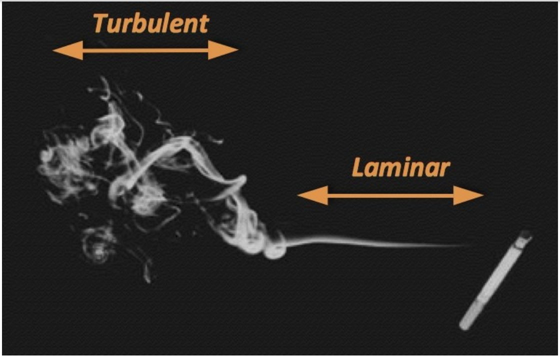
# Introduction




*(https://aerospaceengineeringblog.com/boundary-layers/ )*

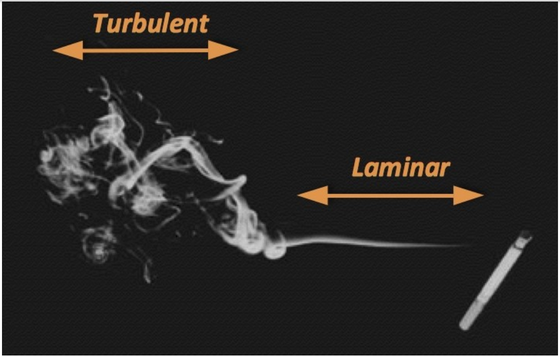
- In the context of fluid dynamics, a flow regime can be categorized into a laminar and turbulent flow.

# Introduction




*(https://aerospaceengineeringblog.com/boundary-layers/ )*

- In the context of fluid dynamics, a flow regime can be categorized into a laminar and turbulent flow.
- Fluid layers move across each other smoothly in a laminar flow, while turbulence is characterized by random fluctuations between those layers.




*(https://aerospaceengineeringblog.com/boundary-layers/ )*

- In the context of fluid dynamics, a flow regime can be categorized into a laminar and turbulent flow.
- Fluid layers move across each other smoothly in a laminar flow, while turbulence is characterized by random fluctuations between those layers.
- Turbulent flow is a chaotic, rotating, multi-scale flow which is very difficult to analyze.

# Motivation

- A long standing goal of turbulence modeling has been to understand the small-scales in turbulent flow via modeling the velocity gradient tensor.

# Motivation

- A long standing goal of turbulence modeling has been to understand the small-scales in turbulent flow via modeling the velocity gradient tensor.
- The velocity gradient tensor, $\boldsymbol{A} \in \mathbb{R}^{3\times 3}$, has components defined by $A_{ij} = \partial u_i / \partial x_j$, in which $u_i$ is a component of the fluid velocity, and $x_i$ is a spatial coordinate.

# Motivation

- A long standing goal of turbulence modeling has been to understand the small-scales in turbulent flow via modeling the velocity gradient tensor.
- The velocity gradient tensor, $\boldsymbol{A} \in \mathbb{R}^{3\times 3}$, has components defined by $A_{ij} = \partial u_i / \partial x_j$, in which $u_i$ is a component of the fluid velocity, and $x_i$ is a spatial coordinate.
- The velocity field can be computed by solving the following incompressible Navier-Stokes equations for a specific domain and boundary conditions:

$$\frac{\partial}{\partial t}\boldsymbol{u} + \boldsymbol{u}\cdot\boldsymbol{\nabla}\boldsymbol{u} = -\frac{1}{\rho}\boldsymbol{\nabla} p + \nu\boldsymbol{\Delta}\boldsymbol{u} \tag{1}$$

Taking the spatial gradient of the NS equations yields:

$$\frac{\partial}{\partial t}\boldsymbol{A} + \boldsymbol{u}\cdot\boldsymbol{\nabla}\boldsymbol{A} = -\boldsymbol{A}^2 - \boldsymbol{H} + \nu\boldsymbol{\Delta}\boldsymbol{A} \tag{2}$$

in which $\boldsymbol{H}$ is the Hessian of the pressure field (a symmetric tensor), with components $H_{ij} = \frac{\partial^2 p}{\partial x_i \partial x_j}$. This equation is unclosed because it requires information of the non-local part of pressure Hessian and viscous term.

# Problem Description

- Formally, the pressure Hessian depends on a spatial integral of the invariants of the tensor $\boldsymbol{A}$ (such as $Q = -\frac{1}{2}tr(\boldsymbol{A}^2)$ and $R = -\frac{1}{2}tr(\boldsymbol{A}^3)$) and the challenge is to develop a simplified model that relates $\boldsymbol{H}$ to the $\boldsymbol{A}$.

# Problem Description

- Formally, the pressure Hessian depends on a spatial integral of the invariants of the tensor $\boldsymbol{A}$ (such as $Q = -\frac{1}{2}tr(\boldsymbol{A}^2)$ and $R = -\frac{1}{2}tr(\boldsymbol{A}^3)$) and the challenge is to develop a simplified model that relates $\boldsymbol{H}$ to the $\boldsymbol{A}$.
- The hypothesis of this project is that given data for the tensor fields $\boldsymbol{H}$, $\boldsymbol{A}$ and its invariants, machine learning algorithms can be used to extract a (data-driven algorithm) relationship between them.

# Problem Description

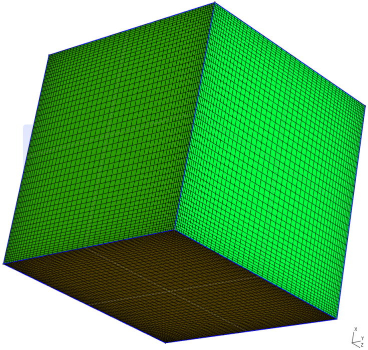
- Formally, the pressure Hessian depends on a spatial integral of the invariants of the tensor $\boldsymbol{A}$ (such as $Q = -\frac{1}{2}tr(\boldsymbol{A}^2)$ and $R = -\frac{1}{2}tr(\boldsymbol{A}^3)$) and the challenge is to develop a simplified model that relates $\boldsymbol{H}$ to the $\boldsymbol{A}$.

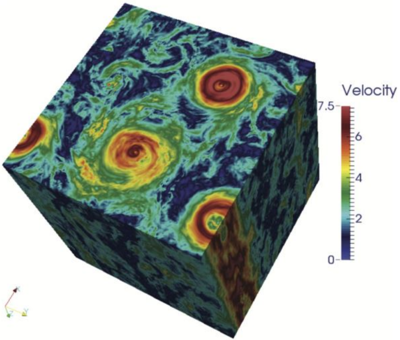
- The hypothesis of this project is that given data for the tensor fields $\boldsymbol{H}$, $\boldsymbol{A}$ and its invariants, machine learning algorithms can be used to extract a (data-driven algorithm) relationship between them.

- Here, the problem is treated as a supervised learning, in which a multi-target regression model is needed that takes velocity gradients information as input features and predicts the 6 components of $\boldsymbol{H}$.

# Data Gathering

- We solve the three-dimensional Direct Numerical Simulation (DNS) of the NS equation on a triperiodic cube which has a uniform mesh with $N^3$ grid points, where $N = 128$, to generate velocity field at all the grid points. By post-processing this data, we can obtain $\boldsymbol{A}$, $tr(\boldsymbol{A}^2)$, $tr(\boldsymbol{A}^3)$ and $\boldsymbol{H}$.





# Preprocessing

- There are $128^3 = 2,097,152$ grid points and so in total $17 * 128^3 = 35,651,584$ datapoints are available ($9 * 128^3$ corresponding to the 9 components of $\boldsymbol{A}$, $2 * 128^3$ corresponding to $R$ and $Q$ as input features and $6 * 128^3$ outputs).

# Preprocessing

- There are $128^3 = 2,097,152$ grid points and so in total $17 * 128^3 = 35,651,584$ datapoints are available ($9 * 128^3$ corresponding to the 9 components of $\boldsymbol{A}$, $2 * 128^3$ corresponding to $R$ and $Q$ as input features and $6 * 128^3$ outputs).
- The input feature are standardized using the "StandardScaler" approach. Furthermore, 10 percent of dataponits (with random selection) are reserved as a test set and the rest are considered as training and validation sets.

# Preprocessing

- There are $128^3 = 2,097,152$ grid points and so in total $17 * 128^3 = 35,651,584$ datapoints are available ($9 * 128^3$ corresponding to the 9 components of $\boldsymbol{A}$, $2 * 128^3$ corresponding to $R$ and $Q$ as input features and $6 * 128^3$ outputs).

- The input feature are standardized using the "StandardScaler" approach. Furthermore, 10 percent of dataponits (with random selection) are reserved as a test set and the rest are considered as training and validation sets.

- In order to build the input features, three different scenarios are compared.
  - 2 input features : $R$ and $Q$ are considered as the inputs.
  - 9 input features : components of $\boldsymbol{A}$ ($A_{ij}$, where $i, j = 1, 2, 3$)
  - 11 input features: combination of $\boldsymbol{A}$, $R$ and $Q$.

# Preprocessing

- There are $128^3 = 2,097,152$ grid points and so in total $17 * 128^3 = 35,651,584$ datapoints are available ($9 * 128^3$ corresponding to the 9 components of $\boldsymbol{A}$, $2 * 128^3$ corresponding to $R$ and $Q$ as input features and $6 * 128^3$ outputs).

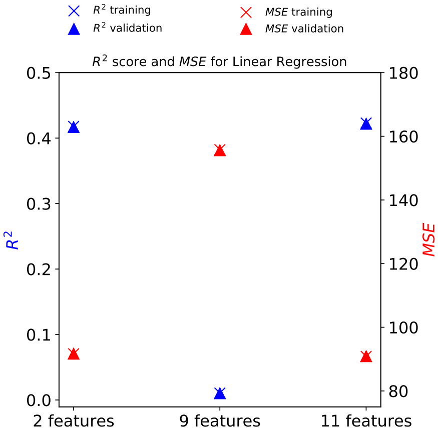
- The input feature are standardized using the "StandardScaler" approach. Furthermore, 10 percent of dataponits (with random selection) are reserved as a test set and the rest are considered as training and validation sets.

- In order to build the input features, three different scenarios are compared.
  - 2 input features : $R$ and $Q$ are considered as the inputs.
  - 9 input features : components of $\boldsymbol{A}$ ($A_{ij}$, where $i, j = 1, 2, 3$)
  - 11 input features: combination of $\boldsymbol{A}$, $R$ and $Q$.

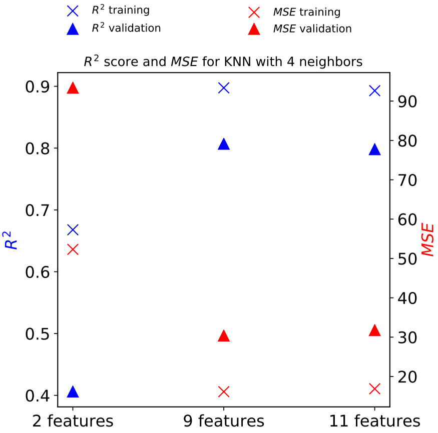
- The Linear Regression (LR), the K Nearest Neighbors (KNN) and the multilayer perceptron (MLP) methods have been tested, and the $R^2$ score and the mean squared error ($MSE$) metrics are employed to evaluate their performance.

# Results

- Evaluating the performance of the LR and the KNN models on both the training and validation sets:



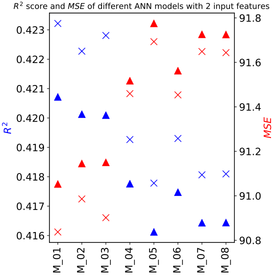

*(Linear Regression)*



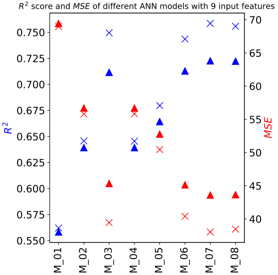

*(K Nearest Neighbors)*

# Results

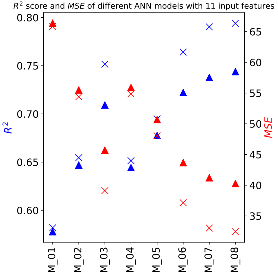
- For each input case, 8 different architecture of the MLP have been tested using the $MLPRegressor$ package with 'Adam' as the optimization algorithm, 'tanh' as the activation function. The architecture of hidden layers of different MLP models are as follows: $Model\,1 = [100]$, $Model\,2 = [1000]$, $Model\,3 = [1000, 500]$, $Model\,4 = [3000]$, $Model\,5 = [1000, 1000, 500]$, $Model\,6 = [1000, 1000]$, $Model\,7 = [1000, 1500]$ , $Model\,8 = [1000, 2500]$.




*(2 input features)*



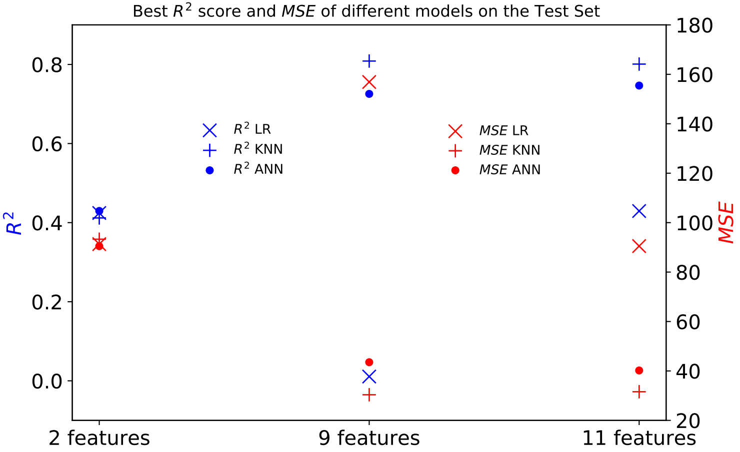

*(9 input features)*




*(11 input features)*

# Results

- The results of the test set are almost the same as validation set and this characteristic implies that the models generalize well to the unseen data.



# Summary

- multiple models using the LR, the KNN and the MLP algorithms were tested to build an efficient model that relates the velocity gradients of a turbulent flow field to its pressure Hessian.

# Summary

- multiple models using the LR, the KNN and the MLP algorithms were tested to build an efficient model that relates the velocity gradients of a turbulent flow field to its pressure Hessian.

- Although the performance of the MLP ($R^2 = 75\%$) was lower than the KNN ($R^2 = 80\%$), it showed that adding more input features can improve the performance. Furthermore, only the MLP could capture the correlations between the outputs in a multi-target task.

# Summary

- multiple models using the LR, the KNN and the MLP algorithms were tested to build an efficient model that relates the velocity gradients of a turbulent flow field to its pressure Hessian.
- Although the performance of the MLP ($R^2 = 75\%$) was lower than the KNN ($R^2 = 80\%$), it showed that adding more input features can improve the performance. Furthermore, only the MLP could capture the correlations between the outputs in a multi-target task.
- Future works:
  - Apply the MLP using TensorFlow library.

# Summary

- multiple models using the LR, the KNN and the MLP algorithms were tested to build an efficient model that relates the velocity gradients of a turbulent flow field to its pressure Hessian.

- Although the performance of the MLP ($R^2 = 75\%$) was lower than the KNN ($R^2 = 80\%$), it showed that adding more input features can improve the performance. Furthermore, only the MLP could capture the correlations between the outputs in a multi-target task.

- Future works:
  - Apply the MLP using TensorFlow library.
  - Enriching the model by feeding more relevant features (such as components of $\boldsymbol{A^2}$ or other invariants of $\boldsymbol{A}$).

# Summary

- multiple models using the LR, the KNN and the MLP algorithms were tested to build an efficient model that relates the velocity gradients of a turbulent flow field to its pressure Hessian.

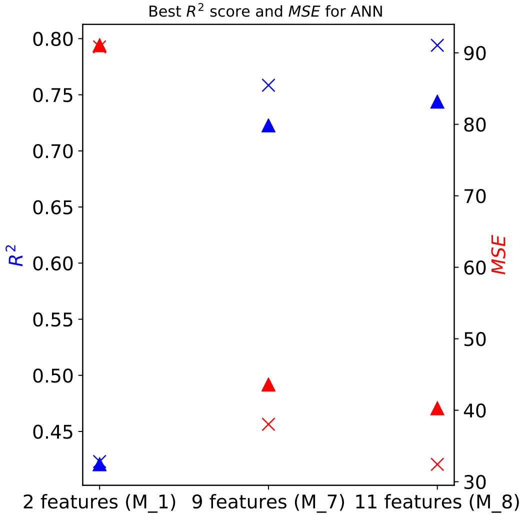
- Although the performance of the MLP ($R^2 = 75\%$) was lower than the KNN ($R^2 = 80\%$), it showed that adding more input features can improve the performance. Furthermore, only the MLP could capture the correlations between the outputs in a multi-target task.

- Future works:
  - Apply the MLP using TensorFlow library.
  - Enriching the model by feeding more relevant features (such as components of $\boldsymbol{A^2}$ or other invariants of $\boldsymbol{A}$).
  - Train time series data.


Thank You

# Results

- The MLP with 11 features outperforms the 9 features' case, which was not apparent in the KNN model.



# Data Gathering

- Firstly we need to compute $A_{ij} = \partial u_i / \partial x_j$. we know that $A = \mathscr{F}^{-1}[ik_j \widehat{u_i}]$ To this aim, using velocity field $\widehat{u_i}$ ($\widehat{\ }$ means Fourier space) at each grid point, the derivative of in Fourier space and converting it to physical space. Once we obtain $A$, we can compute $A^2$ and $A^3$ and eventually their trace.

- Secondly, we need to compute the Hessian of the pressure field, $H_{ij} = \frac{\partial^2 p}{\partial x_i \partial x_j} = \mathscr{F}^{-1}[-ik_m k_n \frac{ik_j G_j}{k^2}]$. To this goal, having velocity field $\widehat{u_i}$, we need to compute $u_i$ in order to construct $u_i u_j$. Then this tensor is converted back to Fourier space to compute $\widehat{u_i u_j}$. Afterwards, $G_j = ik_1(\widehat{u_j u_1}) + ik_2(\widehat{u_j u_2}) + ik_3(\widehat{u_j u_3})$ will be computed and by converting $[-ik_m k_n \frac{ik_j G_j}{k^2}]$ to physical space, Hessian of the pressure will be obtained.