Reynolds stress anisotropy tensor predictions using neural networks

Jiayi Cai, Pierre-Emmanuel Angeli, Jean-Marc Martinez, Guillaume Damblin, Didier Lucor

REYNOLDS STRESS ANISOTROPY TENSOR PREDICTIONS USING NEURAL NETWORKS

Journées Écoulements & Fluides à Saclay - JEFS 2023, June 19-20, 2023

J. Cai[1,3] (jiayi.cai@cea.fr), P.-E. Angeli[1], J.-M. Martinez[2], G. Damblin[2], D. Lucor[3]

[1] CEA Paris-Saclay, DES/ISAS/DM2S/STMF/LMSF, France
[2] CEA Paris-Saclay, DES/ISAS/DM2S/SGLS/LIAD, France
[3] CNRS-LISN, France

June 20, 2023

Contents

1 Introduction

2 Deep neural network models

3 Case study: Plane channel flow

4 Case study: Square duct flow

5 Conclusion and future work

INTRODUCTION

Introduction

The motion of fluids is governed by Navier-Stokes equations, which are highly nonlinear:

$$\frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + \nu \frac{\partial^2 u_i}{\partial x_j^2} \text{ for } i = 1, 2, 3$$

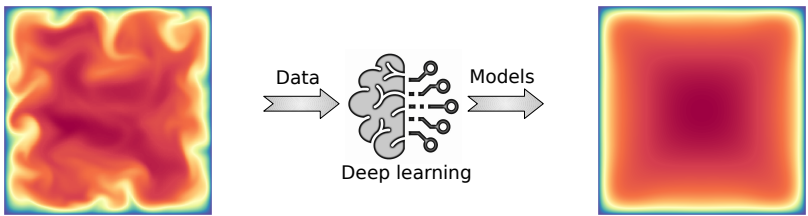
Turbulence is chaotic, characterized by a wide range of temporal and spatial variations ⇒ numerical difficulties in CFD

Direct Numerical Simulation (DNS)

- *exact* representation
- *small-scale* turbulence
- *high* computational cost

Reynolds-Averaged Navier-Stokes (RANS)

- *averaged* representation
- *large-scale* flow statistics
- *low* computational cost



Introduction

Turbulence RANS modeling

- By decomposing flow quantities (e.g. $u_i = \bar{u}_i + u_i'$) and by averaging Navier-Stokes equations, we can obtain RANS equations:

$$\frac{\partial \bar{u}_i}{\partial t} + \bar{u}_j \frac{\partial \bar{u}_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \bar{p}}{\partial x_i} + \frac{\partial}{\partial x_j}\left[\nu \frac{\partial \bar{u}_i}{\partial x_j} - \overline{u_i' u_j'}\right] \text{ for } i = 1, 2, 3$$

- Reynolds stress tensor $\overline{u_i' u_j'}$ $\Rightarrow$ closure problem, additional models should be included.

- We will consider the anisotropy tensor $b_{ij} = \overline{u_i' u_j'}/2k - 1/3\delta_{ij}$

- We define the normalized strain rate tensor $\boldsymbol{S}^\star$ and the rotation rate tensor $\boldsymbol{R}^\star$ as follow:

$$\boldsymbol{S}^\star = \frac{k}{2\varepsilon} \times (\nabla \overline{\boldsymbol{u}} + \nabla \overline{\boldsymbol{u}}^t) \text{ et } \quad \boldsymbol{R}^\star = \frac{k}{2\varepsilon} \times (\nabla \overline{\boldsymbol{u}} - \nabla \overline{\boldsymbol{u}}^t)$$

  - $k = \overline{u_k' u_k'}/2$ = turbulent kinetic energy
  - ϵ the turbulent dissipation rate

Introduction

Turbulence RANS modeling

- Boussinesq's hypothesis (LEVM = "Linear Eddy Viscosity Model") relates anisotropy tensor linearly to strain rate tensor:

$$\boldsymbol{b} \propto \boldsymbol{S}^{\star}$$

- Higher order models (NLEVM = "Non Linear Eddy Viscosity Model", QEVM = "Quadratic Eddy Viscosity Model") :

$$\boldsymbol{b} = F\left(\boldsymbol{S}^{\star}, \boldsymbol{S}^{\star\,2}, \boldsymbol{R}^{\star}, \boldsymbol{R}^{\star\,2}, \ldots\right)$$

Bottlenecks: Simplistic modeling with coefficients experimentally calibrated on simple flow configurations
$\Rightarrow$ limited validity for complex flows (secondary flows, rotational effects...)

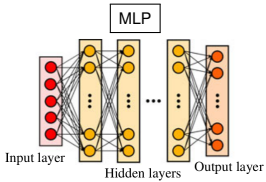
Goal: Provide a better closure model to predict anisotropy tensor via deep neural networks trained by DNS data
$\Rightarrow$ deep neural network to be determined

DEEP NEURAL NETWORK MODELS

Deep neural network models

MLP and TBNN models

- MLP : velocity gradients, $\boldsymbol{S}^\star$, $\boldsymbol{R}^\star$ $\Rightarrow$ various inputs; $\boldsymbol{b}$ $\Rightarrow$ 3 outputs (2D), 6 outputs (3D)

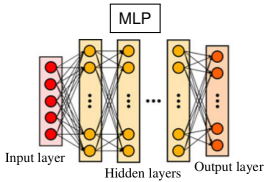


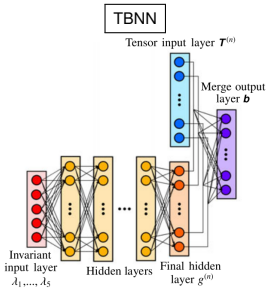
Deep neural network models

MLP and TBNN models

- MLP : velocity gradients, $\boldsymbol{S}^\star$, $\boldsymbol{R}^\star$ $\Rightarrow$ various inputs; $\boldsymbol{b} \Rightarrow$ 3 outputs (2D), 6 outputs (3D)
- TBNN : n inputs (invariants λ) for N unknowns (coefficient functions g)

Neural networks	
Multilayer Perceptron (MLP)	$\mathbf{g}_{\text{MLP}} : \mathbb{R}^n \mapsto \mathbb{R}^3$ or $\mathbb{R}^6$
Tensor Basis Neural Network (TBNN)	$\mathbf{g}_{\text{TBNN}} : \mathbb{R}^n \mapsto \mathbb{R}^N$





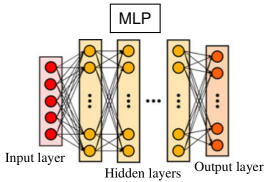
(Ling *et al.*, 2016)

Deep neural network models

MLP and TBNN models

- MLP : velocity gradients, $\boldsymbol{S}^\star$, $\boldsymbol{R}^\star$ $\Rightarrow$ various inputs; $\boldsymbol{b} \Rightarrow$ 3 outputs (2D), 6 outputs (3D)
- TBNN : n inputs (invariants λ) for N unknowns (coefficient functions g)

Neural networks	
Multilayer Perceptron (MLP)	$\mathbf{g}_{\texttt{MLP}} : \mathbb{R}^n \mapsto \mathbb{R}^3$ or $\mathbb{R}^6$
Tensor Basis Neural Network (TBNN)	$\mathbf{g}_{\texttt{TBNN}} : \mathbb{R}^n \mapsto \mathbb{R}^N$

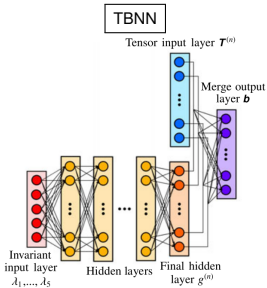


- Pope's model:

$$\boldsymbol{b} = \sum_{k=1}^{N} g^{(k)}(\lambda_1, \ldots, \lambda_n)\boldsymbol{T}^{(k)}$$

(Pope, 1975)

- $\boldsymbol{T}(\boldsymbol{S}^\star, \boldsymbol{R}^\star)$ symmetric and zero-trace tensors
- $\lambda_1, \ldots, \lambda_n$ polynomial invariants of tensors $\boldsymbol{T}$
- g coefficient functions to be determined
- This model satisfies Galilean and rotational invariances.



(Ling *et al.*, 2016)

Deep neural network models

TBNN based on Pope's model

- Expressions of tensors and invariants for 3D case:
  - Tensors :

$$\boldsymbol{T}^{(1)} = \boldsymbol{S}^\star \qquad \boldsymbol{T}^{(6)} = \boldsymbol{R}^{\star 2}\boldsymbol{S}^\star + \boldsymbol{S}^\star\boldsymbol{R}^{\star 2} - \frac{2}{3}\mathrm{tr}(\boldsymbol{S}^\star\boldsymbol{R}^{\star 2})\boldsymbol{I}_3$$

$$\boldsymbol{T}^{(2)} = \boldsymbol{S}^\star\boldsymbol{R}^\star - \boldsymbol{R}^\star\boldsymbol{S}^\star \qquad \boldsymbol{T}^{(7)} = \boldsymbol{R}^\star\boldsymbol{S}^\star\boldsymbol{R}^{\star 2} - \boldsymbol{R}^{\star 2}\boldsymbol{S}^\star\boldsymbol{R}^\star$$

$$\boldsymbol{T}^{(3)} = \boldsymbol{S}^{\star 2} - \frac{1}{3}\mathrm{tr}(\boldsymbol{S}^{\star 2})\boldsymbol{I}_3 \qquad \boldsymbol{T}^{(8)} = \boldsymbol{S}^\star\boldsymbol{R}^\star\boldsymbol{S}^{\star 2} - \boldsymbol{S}^{\star 2}\boldsymbol{R}^\star\boldsymbol{S}^\star$$

$$\boldsymbol{T}^{(4)} = \boldsymbol{R}^{\star 2} - \frac{1}{3}\mathrm{tr}(\boldsymbol{R}^{\star 2})\boldsymbol{I}_3 \qquad \boldsymbol{T}^{(9)} = \boldsymbol{R}^{\star 2}\boldsymbol{S}^{\star 2} + \boldsymbol{S}^{\star 2}\boldsymbol{R}^{\star 2} - \frac{2}{3}\mathrm{tr}(\boldsymbol{S}^{\star 2}\boldsymbol{R}^{\star 2})\boldsymbol{I}_3$$

$$\boldsymbol{T}^{(5)} = \boldsymbol{R}^\star\boldsymbol{S}^{\star 2} - \boldsymbol{S}^{\star 2}\boldsymbol{R} \qquad \boldsymbol{T}^{(10)} = \boldsymbol{R}^\star\boldsymbol{S}^{\star 2}\boldsymbol{R}^{\star 2} - \boldsymbol{R}^{\star 2}\boldsymbol{S}^{\star 2}\boldsymbol{R}^\star$$

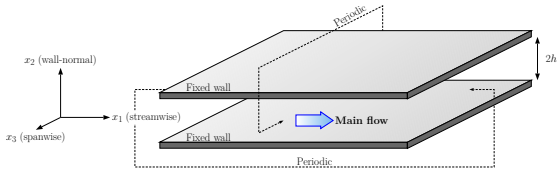
  - Invariants : $\lambda_1 = \mathrm{tr}(\boldsymbol{S}^{\star 2})$, $\lambda_2 = \mathrm{tr}(\boldsymbol{R}^{\star 2})$, $\lambda_3 = \mathrm{tr}(\boldsymbol{S}^{\star 3})$, $\lambda_4 = \mathrm{tr}(\boldsymbol{R}^{\star 2}\boldsymbol{S}^\star)$, $\lambda_5 = \mathrm{tr}(\boldsymbol{R}^{\star 2}\boldsymbol{S}^{\star 2})$

- Expressions of tensors and invariants for statistically 2D case (there exists one direction i, $\bar{u}_i = 0$ and $\dfrac{\partial}{\partial x_i} = 0$):
  - Tensors : $\boldsymbol{T}^{(0)} = \dfrac{1}{2}\boldsymbol{I}_2 - \dfrac{1}{3}\boldsymbol{I}_3$, $\boldsymbol{T}^{(1)} = \boldsymbol{S}^\star$, $\boldsymbol{T}^{(2)} = \boldsymbol{S}^\star\boldsymbol{R}^\star - \boldsymbol{R}^\star\boldsymbol{S}^\star$, e.g. if $i = 3$, $\boldsymbol{T}^{(0)} = \mathrm{diag}\left(\dfrac{1}{6}; \dfrac{1}{6}; -\dfrac{1}{3}\right)$
  - Invariants : $\lambda_1 = \mathrm{tr}(\boldsymbol{S}^{\star 2})$, $\lambda_2 = \mathrm{tr}(\boldsymbol{R}^{\star 2})$

CASE STUDY: PLANE CHANNEL FLOW

Case study: Plane channel flow

- Turbulent channel flow in the x_1 direction:

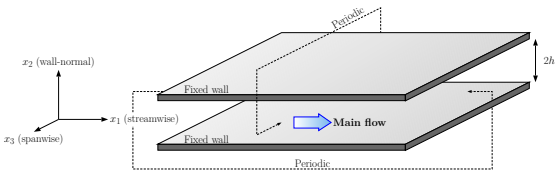


- We have $\bar{u}_3 = 0$ and $\dfrac{\partial}{\partial x_3} = 0 \Rightarrow$ Pope's statistically 2D case: 3 tensors and 2 invariants

Persisting ambiguities exist in the literature concerning the application of Pope's model to the Tensor Basis Neural Network (TBNN) even on this simple flow case, especially for the input feature and basis tensor selection.

Case study: Plane channel flow

- Turbulent channel flow in the x_1 direction:



Case	Model	Features	$\boldsymbol{T}^{*(0)}$
1	MLP	$\{\alpha\}$	/
2	MLP	$\{\alpha, y^+\}$	/
3	MLP	$\{\alpha, \mathrm{Re}_\tau\}$	/
4	MLP	$\{\alpha, y^+, \mathrm{Re}_\tau\}$	/
5	Augmented TBNN	$\{\alpha, y^+, \mathrm{Re}_\tau\}$	$\boldsymbol{T}^{(01)}$
6	Augmented TBNN	$\{\alpha, y^+, \mathrm{Re}_\tau\}$	$\boldsymbol{T}^{(02)}$
7	Augmented TBNN	$\{\alpha, y^+, \mathrm{Re}_\tau\}$	$\boldsymbol{T}^{(03)}$
8	Augmented TBNN	$\{\alpha, y^+, \mathrm{Re}_\tau\}$	$\boldsymbol{T}^{(0)}_{\mathrm{gen}}$

with $\alpha = \dfrac{k}{\epsilon}\dfrac{d\bar{u}_1}{dx_2}$,

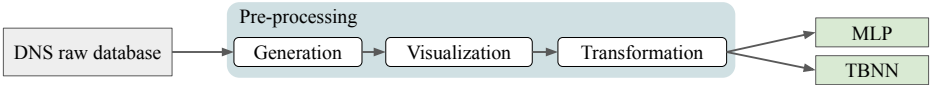
y^+: normalized wall distance and Re_τ: friction Reynolds number

- We have $\bar{u}_3 = 0$ and $\dfrac{\partial}{\partial x_3} = 0 \Rightarrow$ Pope's statistically 2D case: 3 tensors and 2 invariants

Persisting ambiguities exist in the literature concerning the application of Pope's model to the Tensor Basis Neural Network (TBNN) even on this simple flow case, especially for the input feature and basis tensor selection.

Case study: Plane channel flow

DNS database and pre-processing



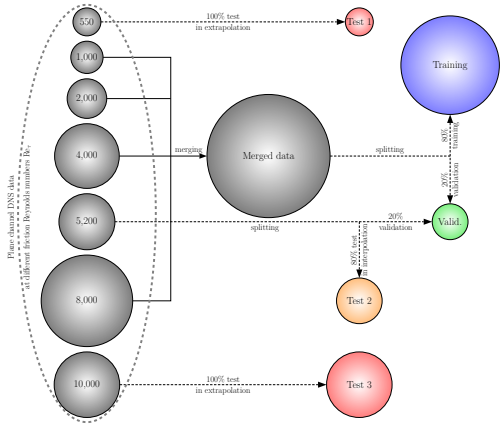
- Training with turbulent channel flow DNS dataset at 7 friction Reynolds numbers, test on interpolation and extrapolation:

  R. D. Moser *et al.*, 1999:
  Re_τ=[550; 1,000; 2,000; 5,200]

  Y. Kaneda et Y. Yamamoto, 2021:
  Re_τ=[4,000; 8,000]

  S. Hoya *et al.*, 2022:
  Re_τ=[10,000]

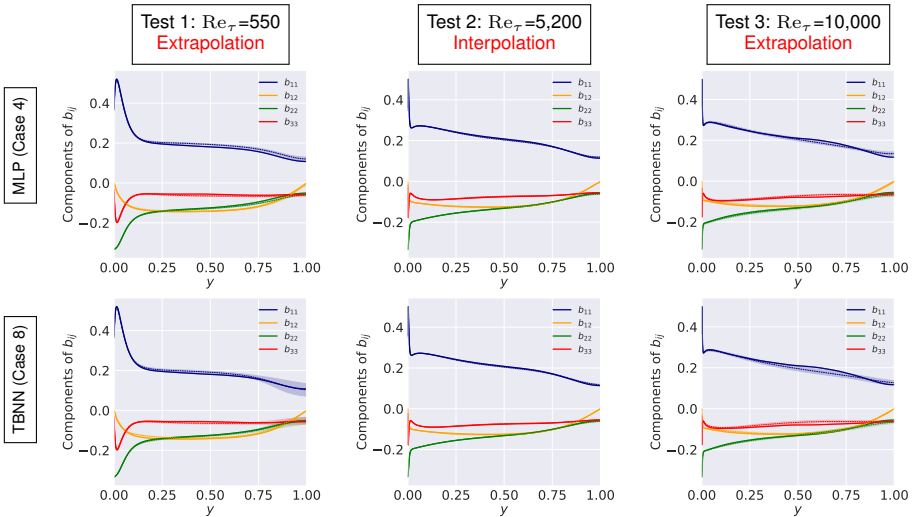
- Data pre-processing (large impact on the performance of deep learning):
  - Features: max normalization, log-transformation $\Rightarrow$ range rescaled to $[-1, 1]$
  - Targets: global reduction by the Frobenius norm



Case study: Plane channel flow

Results analaysis: accuracy and robustness

Averaged predictions by the best MLP and the TBNN models after 10 runs

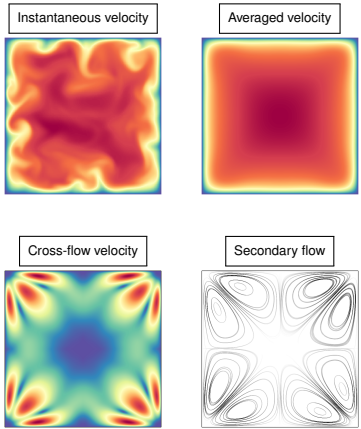


CASE STUDY: SQUARE DUCT FLOW

Case study: Square duct flow

- Augmented TBNN trained with square duct flow DNS dataset at 4 friction Reynolds numbers:

  Pirozzoli *et al.*, 2018:
  Re_τ=[150; 250; 500; 1,000]

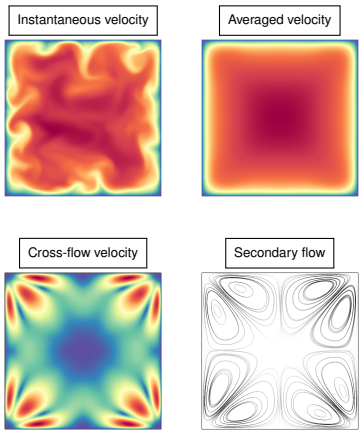


Case study: Square duct flow

- Augmented TBNN trained with square duct flow DNS dataset at 4 friction Reynolds numbers:

  Pirozzoli *et al.*, 2018:
  Re_τ=[150; 250; 500; 1,000]

  - the anisotropy of Reynolds stress tensor ($\overline{u_1' u_1'} \neq \overline{u_2' u_2'}$) induce secondary flows $\Rightarrow$ interest in a more realistic representation of the Reynolds tensor than the Boussinesq hypothesis

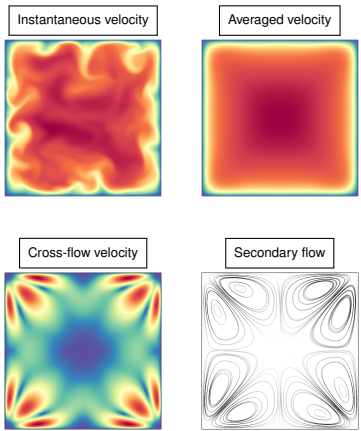


Case study: Square duct flow

- Augmented TBNN trained with square duct flow DNS dataset at 4 friction Reynolds numbers:

  Pirozzoli *et al.*, 2018:
  Re_τ=[150; 250; 500; 1,000]

  - the anisotropy of Reynolds stress tensor ($\overline{u_1' u_1'} \neq \overline{u_2' u_2'}$) induce secondary flows $\Rightarrow$ interest in a more realistic representation of the Reynolds tensor than the Boussinesq hypothesis
  - Different distributions can be observed on data at different Re_τ according to data visualization $\Rightarrow$ out of distribution problem?



Case study: Square duct flow

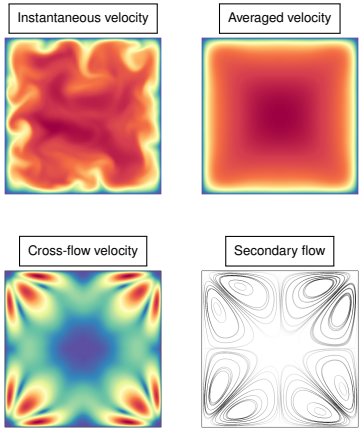
- Augmented TBNN trained with square duct flow DNS dataset at 4 friction Reynolds numbers:

  Pirozzoli *et al.*, 2018:
  Re_τ=[150; 250; 500; 1,000]

  - the anisotropy of Reynolds stress tensor ($\overline{u_1' u_1'} \neq \overline{u_2' u_2'}$) induce secondary flows $\Rightarrow$ interest in a more realistic representation of the Reynolds tensor than the Boussinesq hypothesis
  - Different distributions can be observed on data at different Re_τ according to data visualization $\Rightarrow$ out of distribution problem?

- Random mix study:

  - Training: 80% of all data at Re_τ=[150; 250; 500; 1,000]
  - Validation: 10% of all data at Re_τ=[150; 250; 500; 1,000]
  - Test: 10% of all data at Re_τ=[150; 250; 500; 1,000]



Case study: Square duct flow

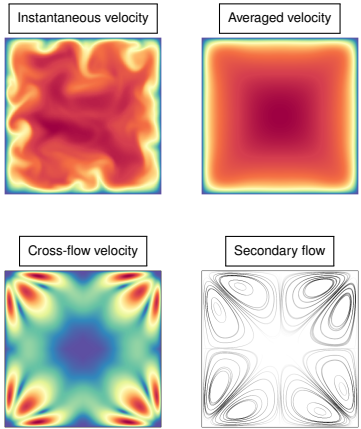
- Augmented TBNN trained with square duct flow DNS dataset at 4 friction Reynolds numbers:

  Pirozzoli *et al.*, 2018:
  Re_τ=[150; 250; 500; 1,000]

  - the anisotropy of Reynolds stress tensor ($\overline{u_1' u_1'} \neq \overline{u_2' u_2'}$) induce secondary flows $\Rightarrow$ interest in a more realistic representation of the Reynolds tensor than the Boussinesq hypothesis
  - Different distributions can be observed on data at different Re_τ according to data visualization $\Rightarrow$ out of distribution problem?

- Random mix study:
  - Training: 80% of all data at Re_τ=[150; 250; 500; 1,000]
  - Validation: 10% of all data at Re_τ=[150; 250; 500; 1,000]
  - Test: 10% of all data at Re_τ=[150; 250; 500; 1,000]

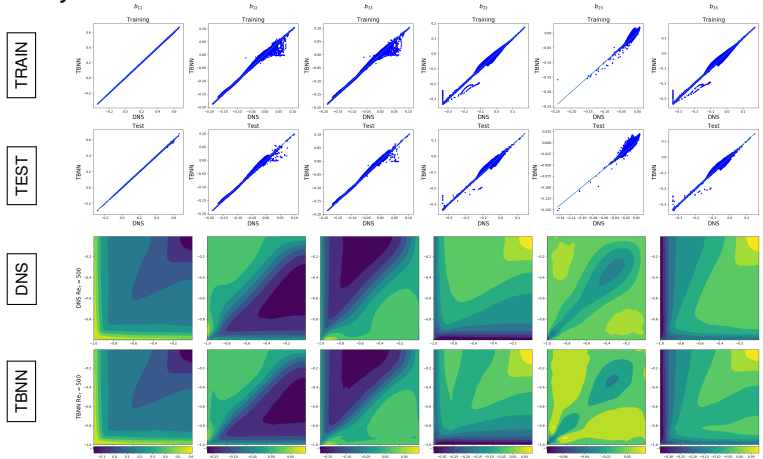
- Interpolation study:
  - Training: 80% of all data at Re_τ=[150; 250; 1,000]
  - Validation: 20% of all data at Re_τ=[150; 250; 1,000]
  - Test: data at Re_τ=500



Case study: Square duct flow

Result analysis: random mix study

Augmented TBNN with $N = 11$ and $n = 5$ (direct application of former study)

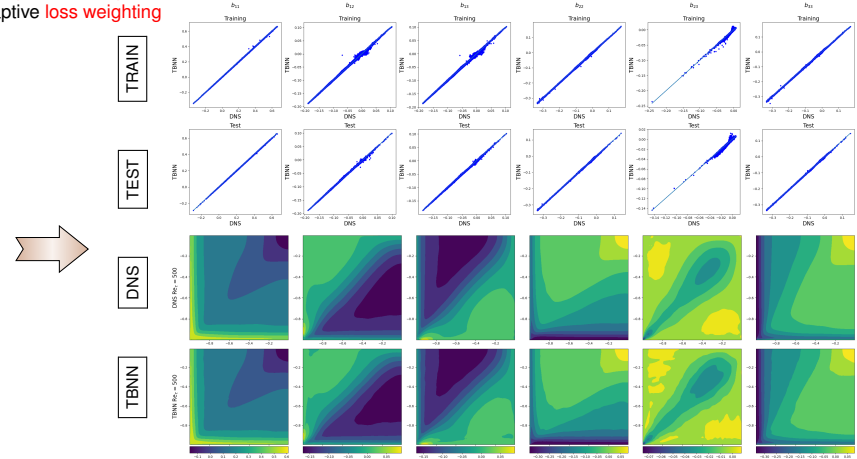


- Learning difficulties within certain regions
- Unbalanced predictions on different components

Case study: Square duct flow

Result analysis: random mix study

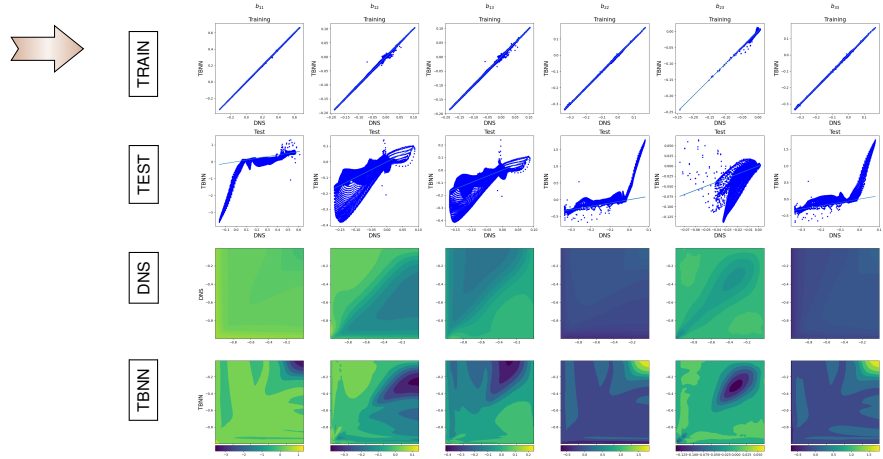
- Improvements:
  - hyperparameter optimization
  - adaptive loss weighting



Case study: Square duct flow

Result analysis: interpolation study

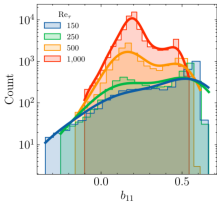
Training on Re_τ=[150; 250; 1,000];
test on Re_τ=500

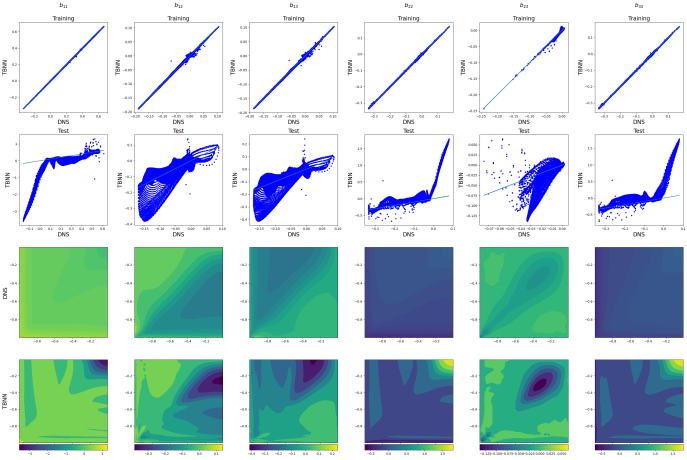


Case study: Square duct flow

Result analysis: interpolation study

Training on Re_τ=[150; 250; 1,000]; test on Re_τ=500





- Out of distribution: good on training, very bad on test
- Lack of data, difficult to interpolate

CONCLUSION AND FUTURE WORK

Conclusion and future work

- Interdomain study coupled with turbulence modeling, deep learning and data science:
  - necessity for robust turbulence RANS models
  - advances in computer science

Conclusion and future work

- Interdomain study coupled with turbulence modeling, deep learning and data science:
  - necessity for robust turbulence RANS models
  - advances in computer science

- Case studies on plane channel flow and square duct flow configuration:
  - deep neural network implementation (MLP and TBNN), input feature selection
  - efficient data pipeline (raw DNS data $\Rightarrow$ features and targets generation $\Rightarrow$ pre-processing)
  - evaluation and enhancement of TBNN model
  - robust predictions of anisotropy tensor, in both interpolation and extrapolation scenarios on plane channel flow

    Cai et al., *Reynolds Stress Anisotropy Tensor Predictions for Turbulent Channel Flow using Neural Networks*, arXiv: 2208.14301 (2022)

Conclusion and future work

- Interdomain study coupled with turbulence modeling, deep learning and data science:
  - necessity for robust turbulence RANS models
  - advances in computer science

- Case studies on plane channel flow and square duct flow configuration:
  - deep neural network implementation (MLP and TBNN), input feature selection
  - efficient data pipeline (raw DNS data $\Rightarrow$ features and targets generation $\Rightarrow$ pre-processing)
  - evaluation and enhancement of TBNN model
  - robust predictions of anisotropy tensor, in both interpolation and extrapolation scenarios on plane channel flow

    Cai et al., *Reynolds Stress Anisotropy Tensor Predictions for Turbulent Channel Flow using Neural Networks*, arXiv: 2208.14301 (2022)

- Ongoing work:
  - *a-posteriori* validation via in-house CFD code TrioCFD
  - extension to square/rectangular duct flow configuration, try interpolation/extrapolation scenarios on different geometries

isas

THANKS! ANY QUESTIONS?

Case study: Plane channel flow

Application of Pope's model

- Clarification 1: We have $\bar{u}_3 = 0$ and $\dfrac{\partial}{\partial x_3} = 0 \Rightarrow$ Pope's statistically 2D case: 3 tensors and 2 invariants

$$\boldsymbol{b} = g^{(0)}(\lambda_1, \lambda_2)\boldsymbol{T}^{(0)} + g^{(1)}(\lambda_1, \lambda_2)\boldsymbol{T}^{(1)} + g^{(2)}(\lambda_1, \lambda_2)\boldsymbol{T}^{(2)}$$

Case study: Plane channel flow

Application of Pope's model

- Clarification 1: We have $\bar{u}_3 = 0$ and $\dfrac{\partial}{\partial x_3} = 0 \Rightarrow$ Pope's statistically 2D case: 3 tensors and 2 invariants

$$\boldsymbol{b} = g^{(0)}(\lambda_1, \lambda_2)\boldsymbol{T}^{(0)} + g^{(1)}(\lambda_1, \lambda_2)\boldsymbol{T}^{(1)} + g^{(2)}(\lambda_1, \lambda_2)\boldsymbol{T}^{(2)}$$

- Clarification 2: We demonstrated that, specifically for turbulent channel flow:
  - Tensors: three constant alternatives $\boldsymbol{T}^{(0)}$ can all form a tensor basis with $\boldsymbol{T}^{(1)}$ and $\boldsymbol{T}^{(2)}$

$$\boldsymbol{T}^{(0)} = \frac{1}{2}\boldsymbol{I}_2 - \frac{1}{3}\boldsymbol{I}_3 = \mathrm{diag}\left(-\frac{1}{3}; \frac{1}{6}; \frac{1}{6}\right) \text{ or } \mathrm{diag}\left(\frac{1}{6}; -\frac{1}{3}; \frac{1}{6}\right) \text{ or } \mathrm{diag}\left(\frac{1}{6}; \frac{1}{6}; -\frac{1}{3}\right)$$

$$\boldsymbol{T}^{(1)} = \boldsymbol{S}^\star = \begin{pmatrix} 0 & \alpha/2 & 0 \\ \alpha/2 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

$$\boldsymbol{T}^{(2)} = \boldsymbol{S}^\star\boldsymbol{R}^\star - \boldsymbol{R}^\star\boldsymbol{S}^\star = \begin{pmatrix} -\alpha^2/2 & 0 & 0 \\ 0 & \alpha^2/2 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

  - Invariant: only one invariant to be considered

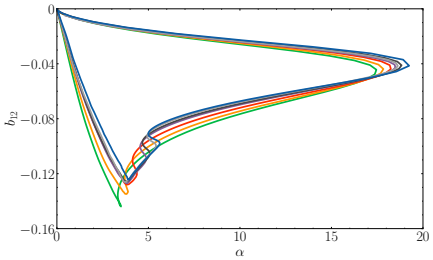
$$\lambda_1 = \mathrm{tr}(\boldsymbol{S}^{\star 2}) = \frac{\alpha^2}{2} = -\lambda_2$$

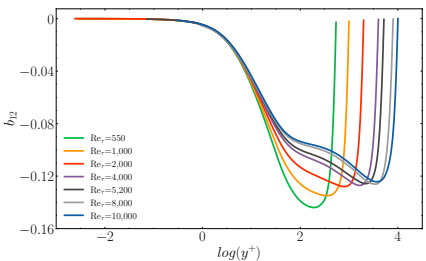
$\Rightarrow \boldsymbol{b} = g^{(0)}(\alpha)\boldsymbol{T}^{(0)} + g^{(1)}(\alpha)\boldsymbol{T}^{(1)} + g^{(2)}(\alpha)\boldsymbol{T}^{(2)}$, only α is needed to predict $\boldsymbol{b}$ according to Pope's model.

Case study: Plane channel flow

Application of Pope's model

- Pope's model: $\boldsymbol{b} = g^{(0)}(\alpha)\boldsymbol{T}^{(0)} + g^{(1)}(\alpha)\boldsymbol{T}^{(1)} + g^{(2)}(\alpha)\boldsymbol{T}^{(2)}$
- Clarification 3: α alone is not sufficient to predict $\boldsymbol{b}$.





- Other features should be included in the neural networks:
  - Friction Reynolds number: Re_τ
  - Dimensionless wall distance: y^+
- Comparison among neural networks using different features
  $\Rightarrow$ to determine an optimal input set

Case study: Plane channel flow

Application of Pope's model

- Channel flow: $\boldsymbol{T}^{(0)}$ can be $\boldsymbol{T}^{(01)} = \mathrm{diag}\left(-\frac{1}{3}; \frac{1}{6}; \frac{1}{6}\right)$ or $\boldsymbol{T}^{(02)} = \mathrm{diag}\left(\frac{1}{6}; -\frac{1}{3}; \frac{1}{6}\right)$ or $\boldsymbol{T}^{(03)} = \mathrm{diag}\left(\frac{1}{6}; \frac{1}{6}; -\frac{1}{3}\right)$
- Clarification 4: Only one of the three alternatives $\boldsymbol{T}^{(0)}$ fits the value of $\boldsymbol{b}$ at the channel center:

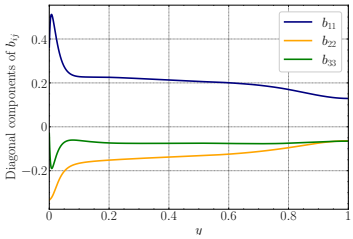
$$\boldsymbol{b}^{\mathrm{DNS}}_{\mathrm{center}} \propto \mathrm{diag}\left(-\frac{1}{3}; \frac{1}{6}; \frac{1}{6}\right) = T^{(01)}$$

- We newly proposed a generalized $\boldsymbol{T}^{(0)}$:

$$\boldsymbol{T}^{(0)}_{\mathrm{gen}} = g_{01}\boldsymbol{T}^{(01)} + g_{02}\boldsymbol{T}^{(02)} + g_{03}\boldsymbol{T}^{(03)} = \mathrm{diag}\left(f_{01}; f_{02}; f_{03}\right)$$

with $f_{01} + f_{02} + f_{03} = 0$ to preserve the zero-trace of $\boldsymbol{b}$.

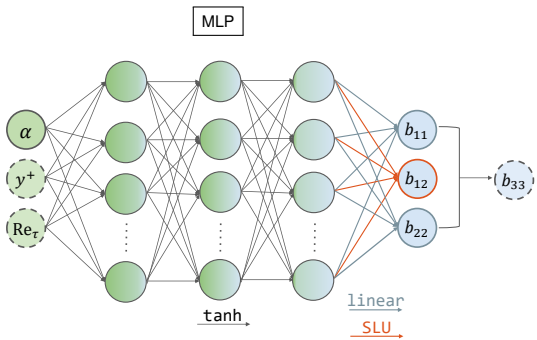
- Comparison among neural networks using different $\boldsymbol{T}^{(0)} \Rightarrow$ to determine an optimal $\boldsymbol{T}^{(0)}$
  - $\boldsymbol{b} = g^{(0)}(\alpha, \mathrm{Re}_\tau, y^+)\boldsymbol{T}^{(0)} + g^{(1)}(\alpha, \mathrm{Re}_\tau, y^+)\boldsymbol{T}^{(1)} + g^{(2)}(\alpha, \mathrm{Re}_\tau, y^+)\boldsymbol{T}^{(2)}$
  - $\boldsymbol{b} = \boldsymbol{T}^{(0)}_{\mathrm{gen}}(\alpha, \mathrm{Re}_\tau, y^+) + g^{(1)}(\alpha, \mathrm{Re}_\tau, y^+)\boldsymbol{T}^{(1)}$
    $= \left[\mathrm{diag}\left(f_{01}; f_{02}; f_{03}\right)\right](\alpha, \mathrm{Re}_\tau, y^+) + g^{(1)}(\alpha, \mathrm{Re}_\tau, y^+)\boldsymbol{T}^{(1)}$

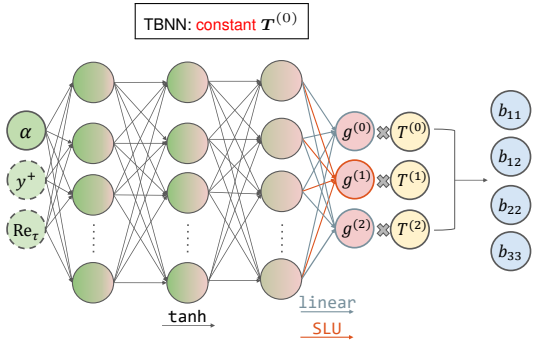


Case study: Plane channel flow

Implementation of deep neural networks

- Neural networks implemented with TensorFlow:
  - MLP and TBNN architectures, quite shallow (3 hidden layers and 10 nodes per layer)
  - activation functions: hyperbolic tangent (tanh) for hidden layers, linear or softplus linear (SLU) for the output layer
  - early stopping to avoid over-fitting, batch learning, Adams

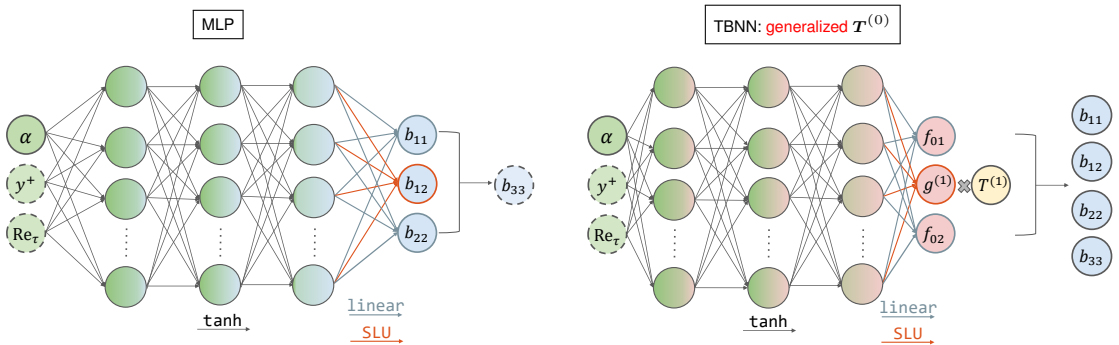




Case study: Plane channel flow

Implementation of deep neural networks

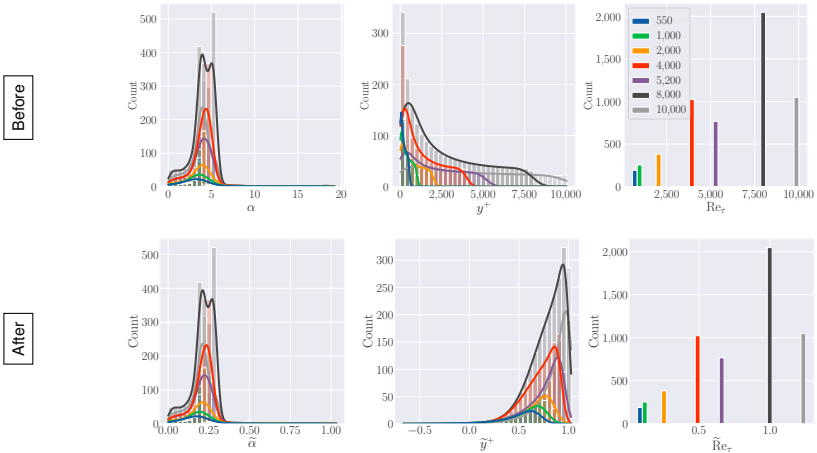
- Neural networks implemented with TensorFlow:
  - MLP and TBNN architectures, quite shallow (3 hidden layers and 10 nodes per layer)
  - activation functions: hyperbolic tangent (tanh) for hidden layers, linear or softplus linear (SLU) for the output layer
  - early stopping to avoid over-fitting, batch learning, Adams



Case study: Plane channel flow

Data pre-processing

- Data pre-processing (large impact on the performance of deep learning):
  - Features: max normalization, log transformation $\Rightarrow$ rescale feature range



$$\widetilde{\alpha} = \frac{\alpha}{\max(\alpha)}$$

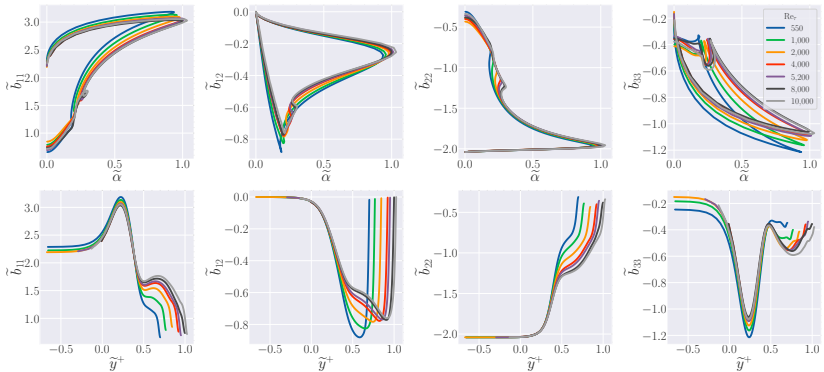
$$\widetilde{y}^+ = \frac{\log(y^+)}{\max(\log(y^+))}$$

$$\widetilde{\mathrm{Re}}_\tau = \frac{\mathrm{Re}_\tau}{\max(\mathrm{Re}_\tau)}$$

Case study: Plane channel flow

Data pre-processing

- Data pre-processing (large impact on the performance of deep learning):
  - Targets: global reduction based on the Frobenius norm



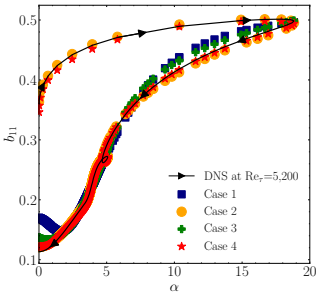
$$\sigma_{\mathbf{b}} = \sqrt{\frac{1}{m}\left[\sum_{k=1}^{3} b_{kk}^2 + b_{12}^2\right]}$$

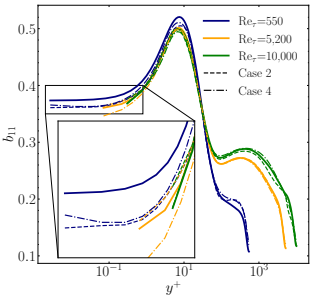
$$\widetilde{\mathbf{b}} = \frac{\mathbf{b}}{\sigma_{\mathbf{b}}}$$

Case study: Plane channel flow

Results analysis on feature selection

- Input features:
  - $\alpha = \dfrac{k}{\epsilon}\dfrac{d\bar{u}_1}{dx_2}$
  - y^+: normalized wall distance
  - Re_τ: friction Reynolds number
- Case studies with different feature combinations upon the MLP model:
  - Case 1: $\{\alpha\}$
  - Case 2: $\{\alpha, y^+\}$
  - Case 3: $\{\alpha, \mathrm{Re}_\tau\}$
  - Case 4: $\{\alpha, y^+, \mathrm{Re}_\tau\}$





- Prediction difficulties mostly lie in the near-wall region.
- By including y^+, models in Case 2 and Case 4 overcome the multi-value issue.
- The adding of Re_τ in Case 4 allows to better distinguish flows at different turbulence levels in the near-wall region.

Case study: Plane channel flow

Results analysis on $T^{(0)}$ selection

- Case studies with different $T^{(0)}$ upon the TBNN model:
  - Case 5: $\boldsymbol{T}^{(0)} = \boldsymbol{T}^{(01)} = \text{diag}\left(-\frac{1}{3}; \frac{1}{6}; \frac{1}{6}\right)$
  - Case 6: $\boldsymbol{T}^{(0)} = \boldsymbol{T}^{(02)} = \text{diag}\left(\frac{1}{6}; -\frac{1}{3}; \frac{1}{6}\right)$
  - Case 7: $\boldsymbol{T}^{(0)} = \boldsymbol{T}^{(03)} = \text{diag}\left(\frac{1}{6}; \frac{1}{6}; -\frac{1}{3}\right)$
  - Case 8: $\boldsymbol{T}^{(0)} = \boldsymbol{T}^{(0)}_{\text{gen}} = \text{diag}\left(f^{(01)}; f^{(02)}; f^{(03)}\right)$

- Different $\boldsymbol{T}^{(0)} \Rightarrow$ different TBNN behavior
- Constant $\boldsymbol{T}^{(0)} \Rightarrow$ incapability at the center or near the wall
  - $\boldsymbol{T}^{(01)} \propto \boldsymbol{b}^{\text{DNS}}_{\text{center}}$: correct predictions at the center, but fails near the wall
  - $\boldsymbol{T}^{(02)}$ and $\boldsymbol{T}^{(03)}$: fail in both scenarios
- Proposed $\boldsymbol{T}^{(0)}_{\text{gen}} \Rightarrow$ good agreement with DNS data