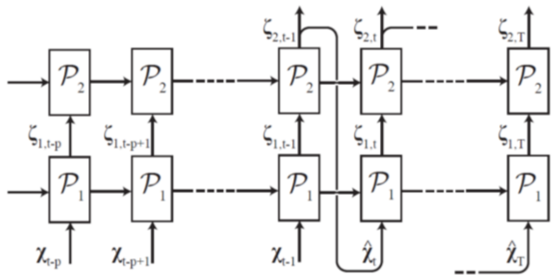
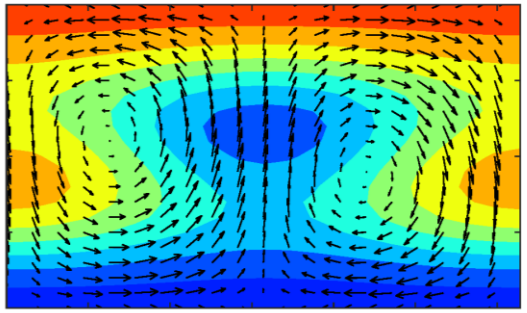
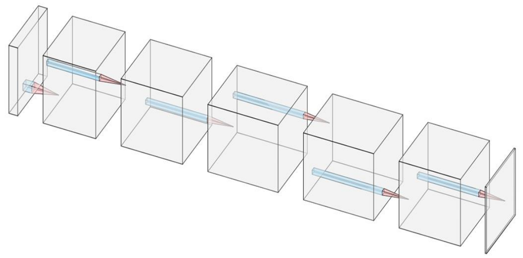
Modeling and controlling turbulent flows through deep learning

Ricardo Vinuesa[1,2]





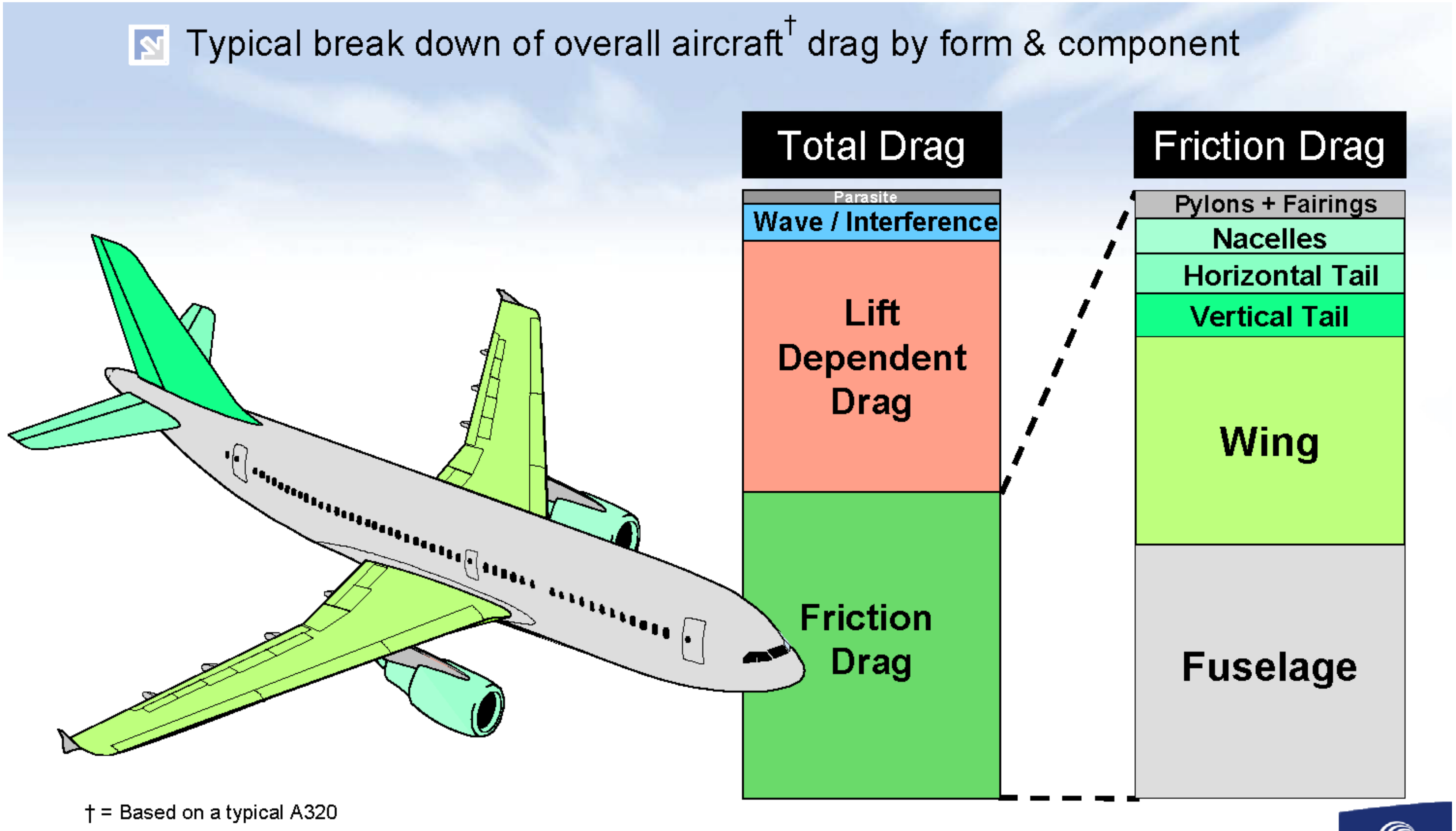


[1]FLOW, KTH Engineering Mechanics, Stockholm, Sweden.

[2]Swedish e-Science Research Centre (SeRC), Stockholm, Sweden.

16 August 2022, FEM Seminar, Lawrence Livermore National Laboratory (US)

Motivation



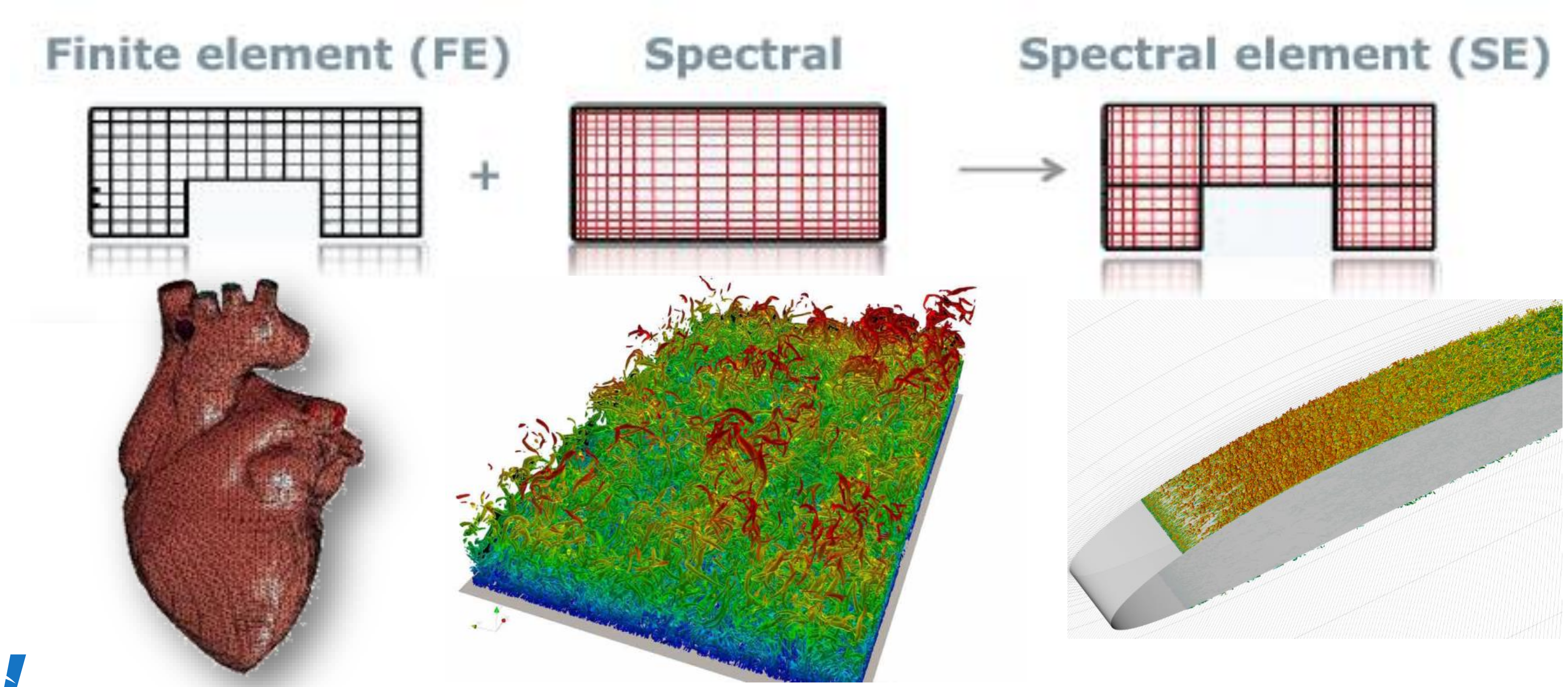
Hills. The Airbus Challenge. EADS Engineering Europe, 2008

Page 27

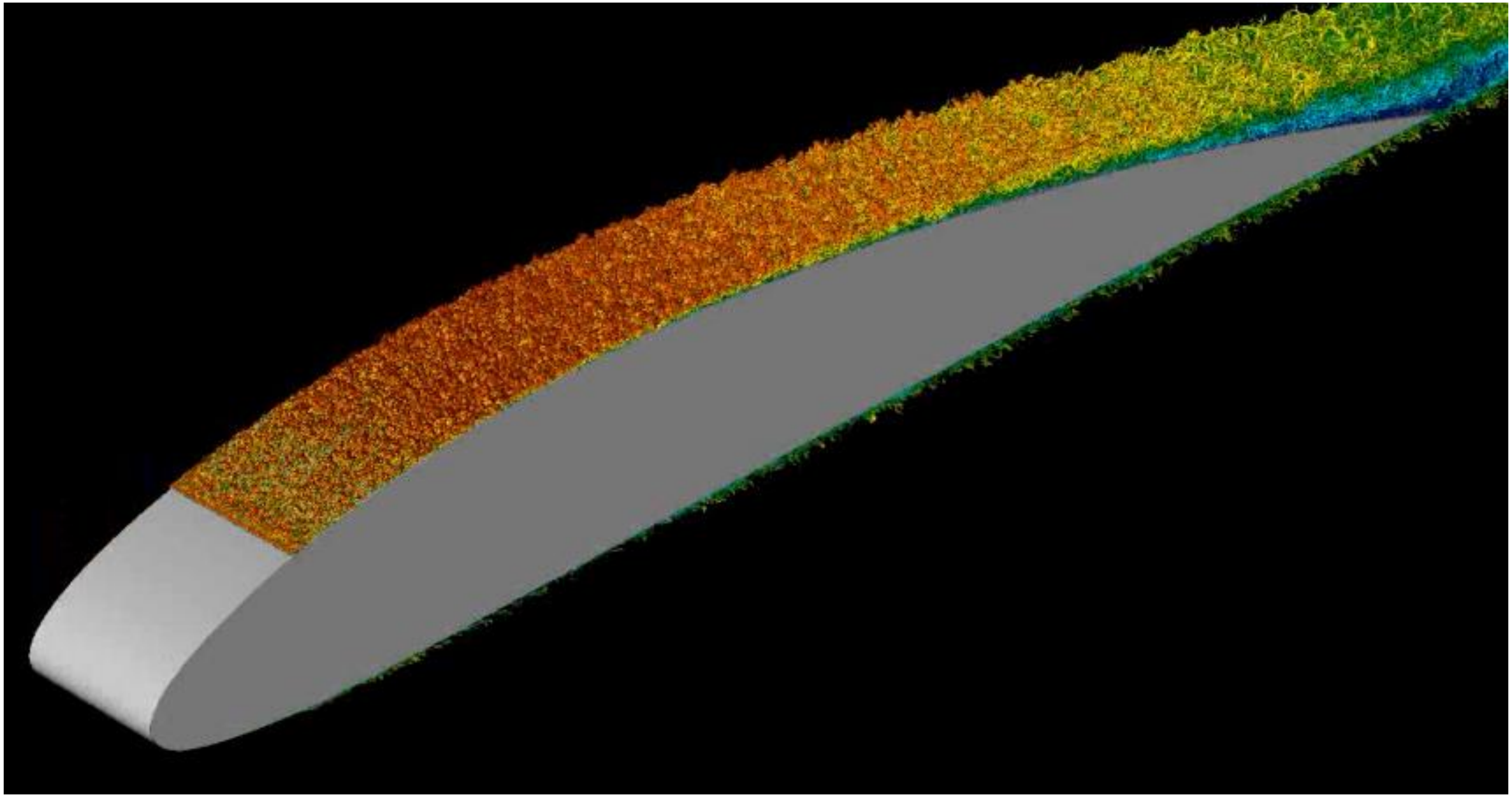
Numerical code: Nek5000 (Fischer *et al.*, 2008)

- **High-order numerical methods** are required for accurate simulations of turbulent flows due to the significant scale disparity of the flow structures, both in time and space.

- Spectral elements allow to solve **flows in complex geometries**.

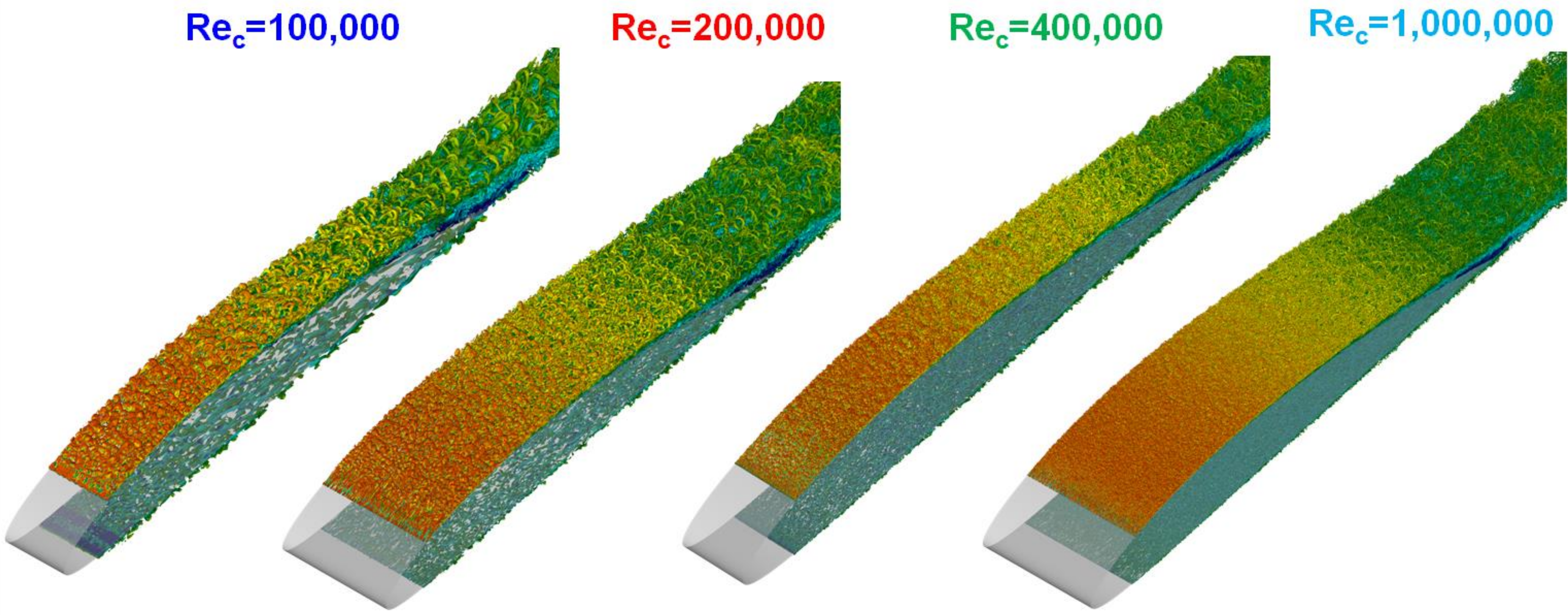


DNS of turbulent flow around a NACA4412 wing section at Re_c=400,000 and AoA=5 degrees



https://www.youtube.com/watch?v=aR-hehP1pTk

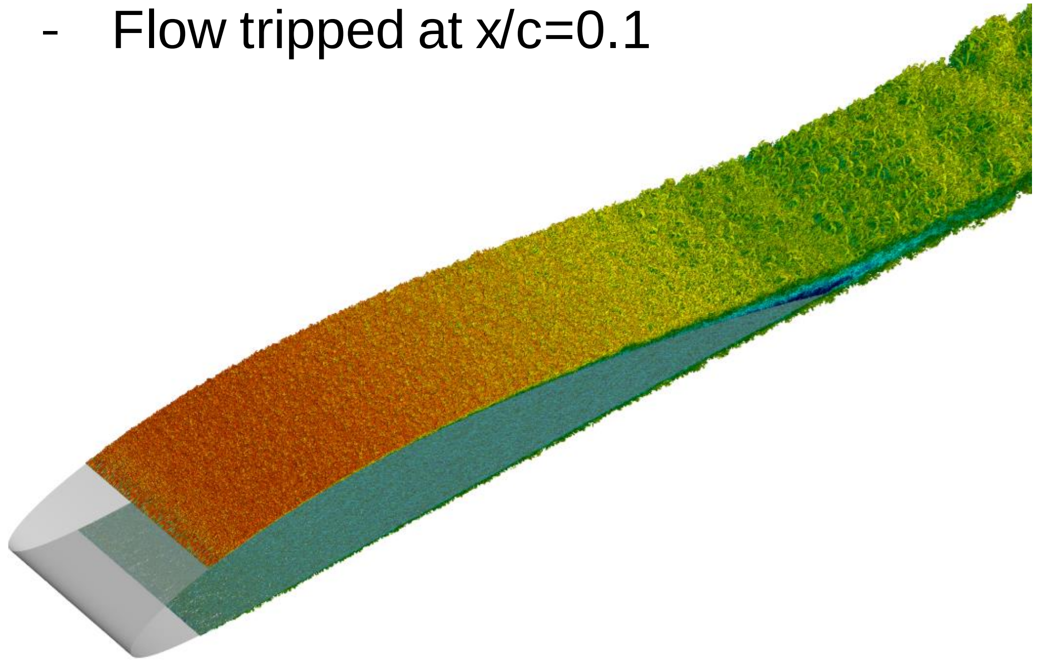
Effect of Reynolds number for a given pressure-gradient history: well-resolved LES

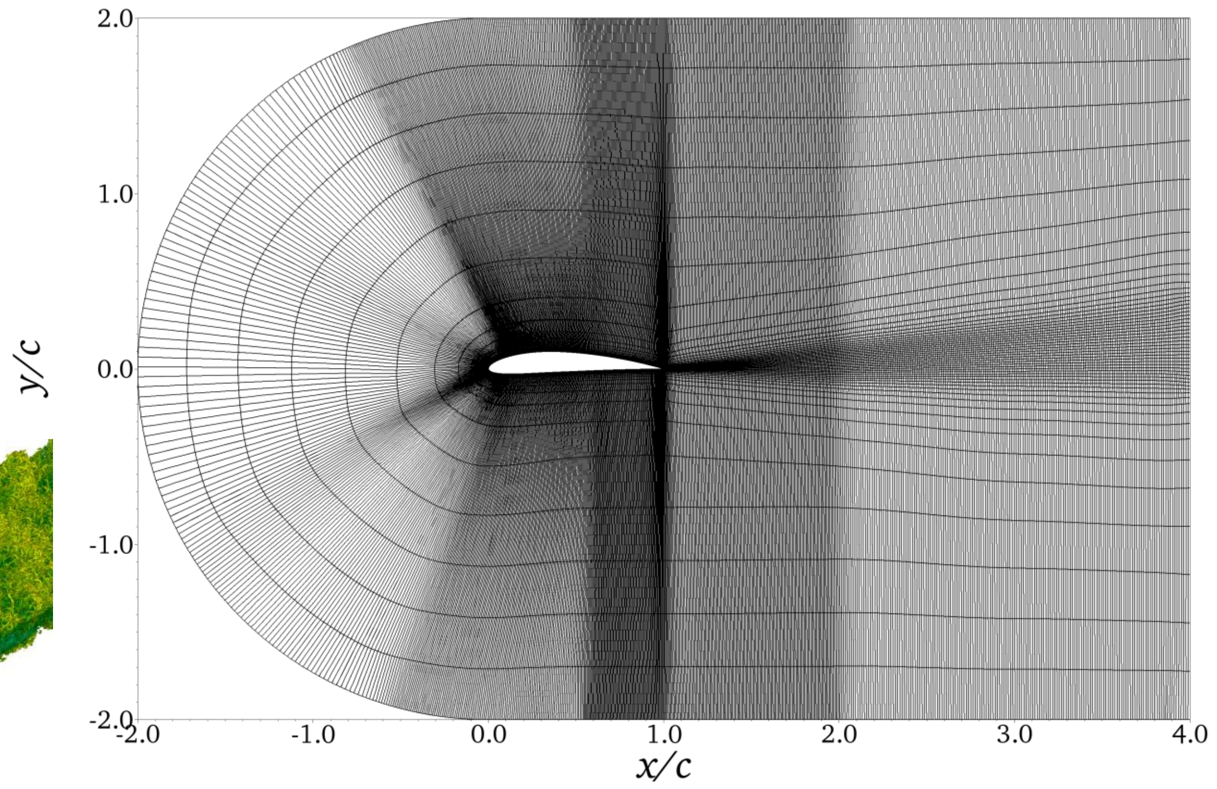


Vinuesa et al., IJHFF, 2018

Well-resolved LES of NACA4412 wing section. Re_c=1,000,000 and AoA=5°

- Relaxation-term LES with Nek5000 (as in Schlatter *et al.*, 2004).
- *(Lx, Ly, Lz) = (6c, 4c, 0.2c)*
- Use of RANS BCs.
- $Re_\theta \approx 6{,}000$ and $Re_\tau \approx 707$
- Flow tripped at x/c=0.1





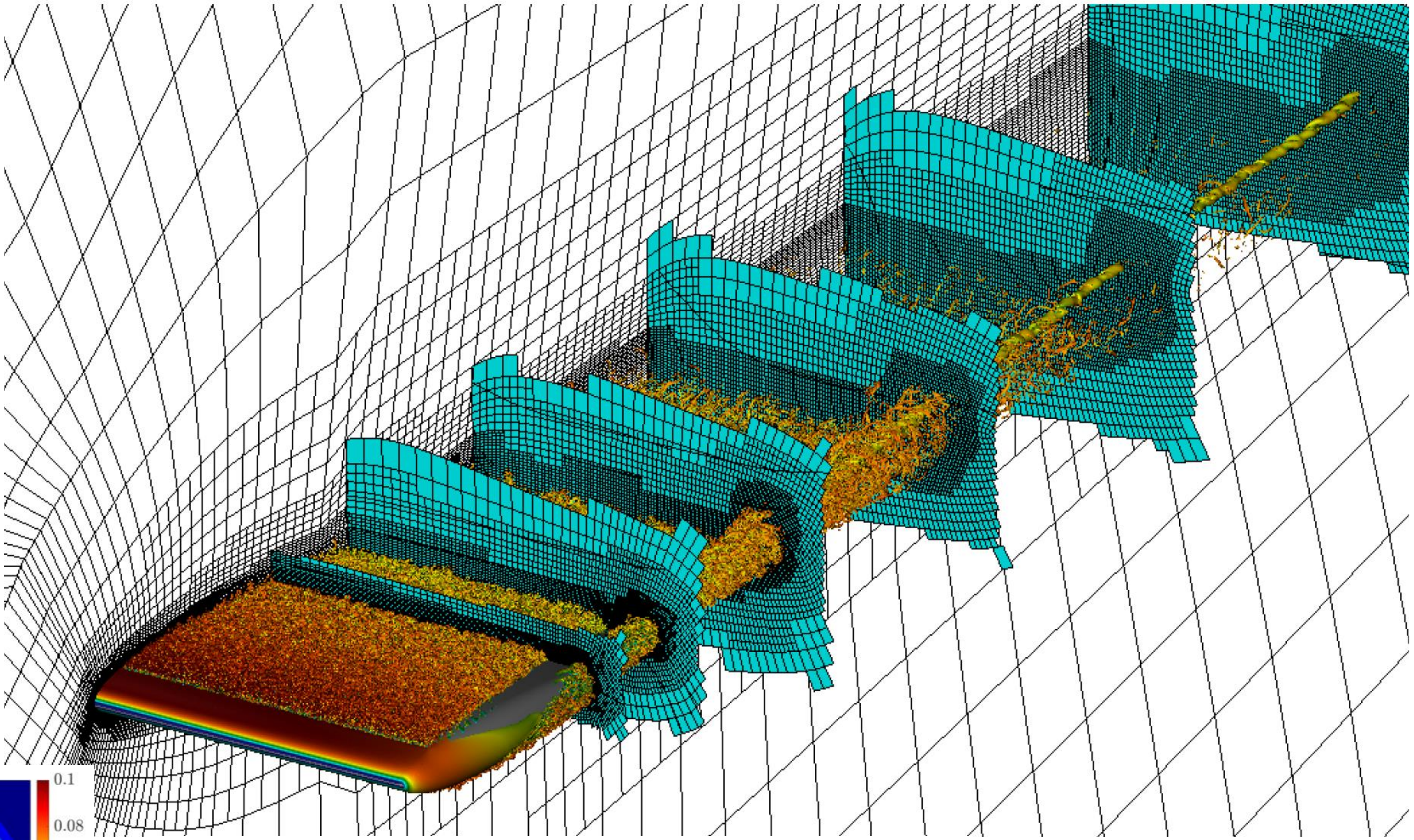
- 4.5 million elements, N=7
- **2.3 billion grid points.**
- On-going accumulation of statistics.
- $\Delta x^+ < 27$, $\Delta y^+ < 0.96$, $\Delta z^+ < 13$, $\Delta x < 13\eta$

Vinuesa et al., IJHFF, 2018

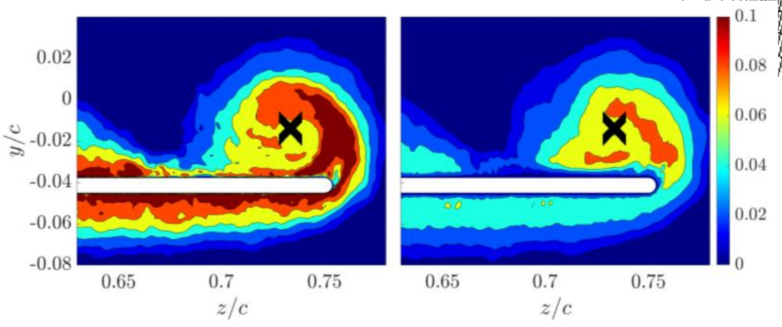
Adaptive simulations of NACA0012 profile with rounded wing tip

Toosi et al., DLES conference (2022)

Statistics, time series, fields and reduced-order models of the wing-tip vortex and the wake.



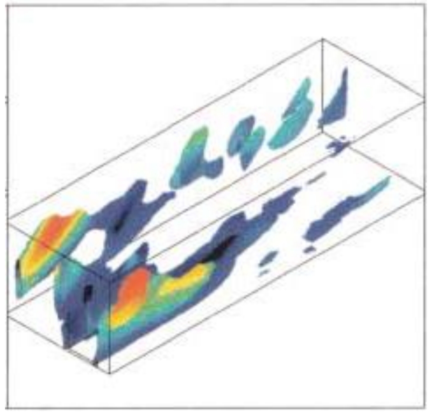
u_{rms} and v_{rms}

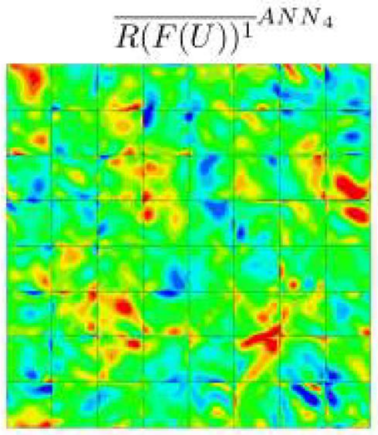


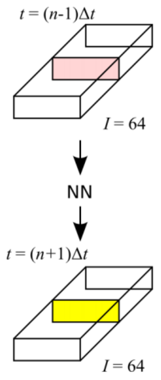
Grid	N_{GLL}	N_{grid}	$(\Delta x^+_{\mathrm{mean}}, y^+_1, \Delta z^+_{\mathrm{mean}})$
P-0	376×10^6	249×10^6	(10.3,0.72,8.7)
P-5	381×10^6	252×10^6	(10.5,0.73,8.5)
P-10	435×10^6	288×10^6	(12,0.8,9)
RWT-0	950×10^6	630×10^6	(10.3,0.7,5.5)
RWT-5	1.58×10^9	1.05×10^9	(10.5,0.75,5.7)
RWT-10	2.16×10^9	1.43×10^9	(12,0.8,6)

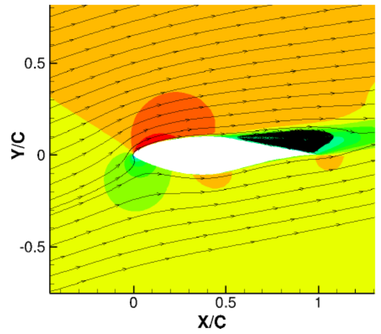
Applications of machine learning to fluid mechanics

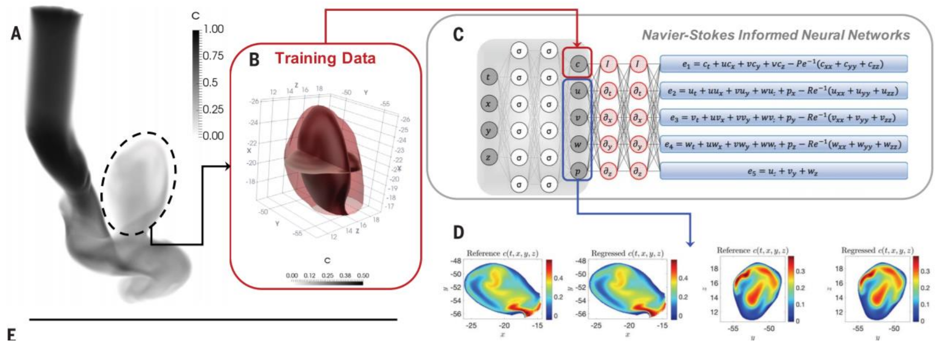
- **Milano and Koumoutsakos (2002)** used neural networks to model the **near-wall region of turbulent channel flows**.
- **Beck et al. (2018)** employed neural networks to develop **subgrid-scale (SGS) models for LES**. They used DNS data to train and predict the SGS term.
- **Fukami et al. (2019)** used neural networks to obtain **inflow conditions** and for super-resolution.
- **Duraisamy et al. (2019)** showed the potential of machine learning for developmet of **RANS models**
- **Raissi et al. (2020)** identified the potential of physics-informed neural networks (PINNs) for reconstrcuting flow fields from tracer input.





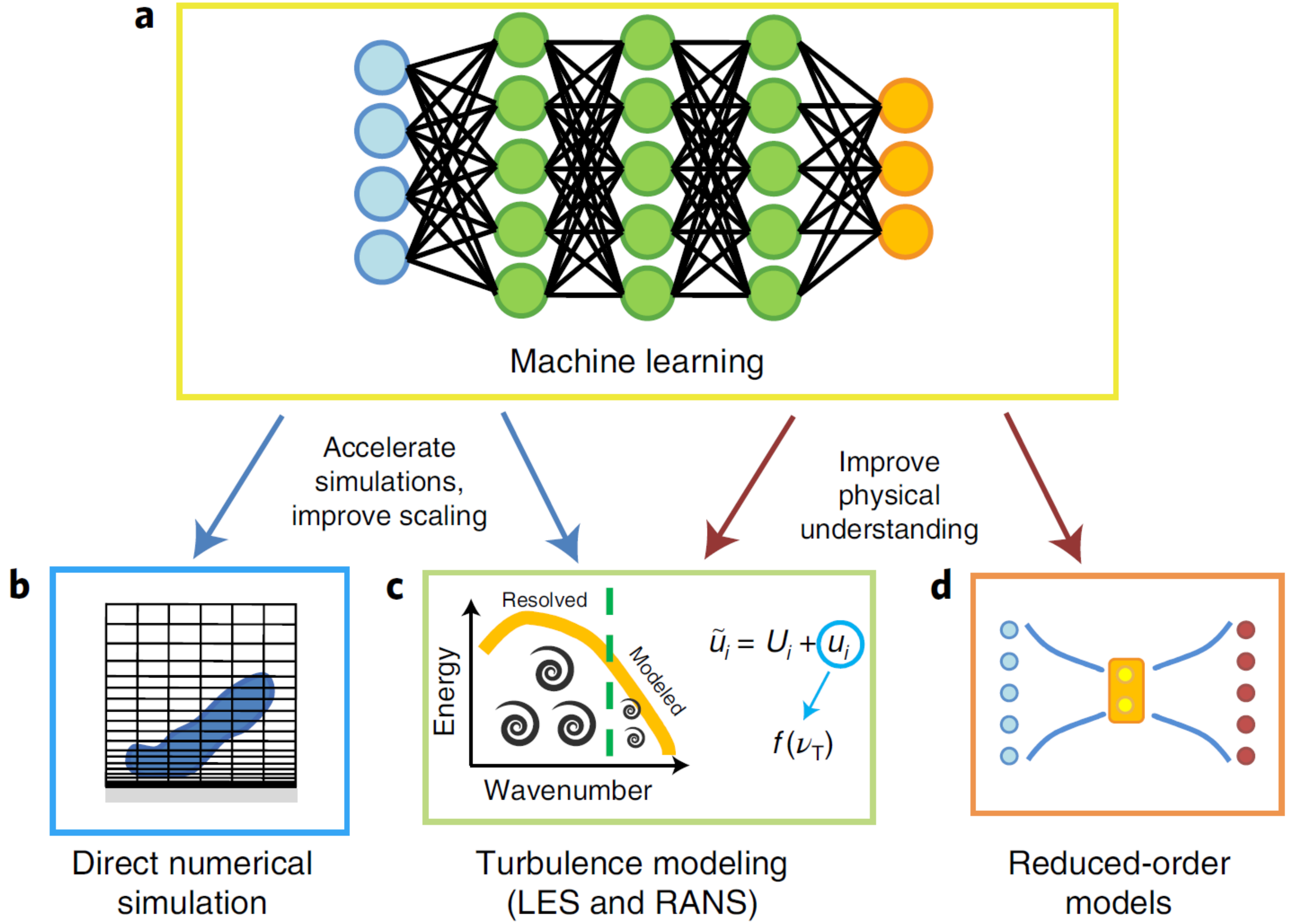






- **Brunton et al. (2019)** showed, among others, applications to **flow control**.

Enhanced CFD with machine learning

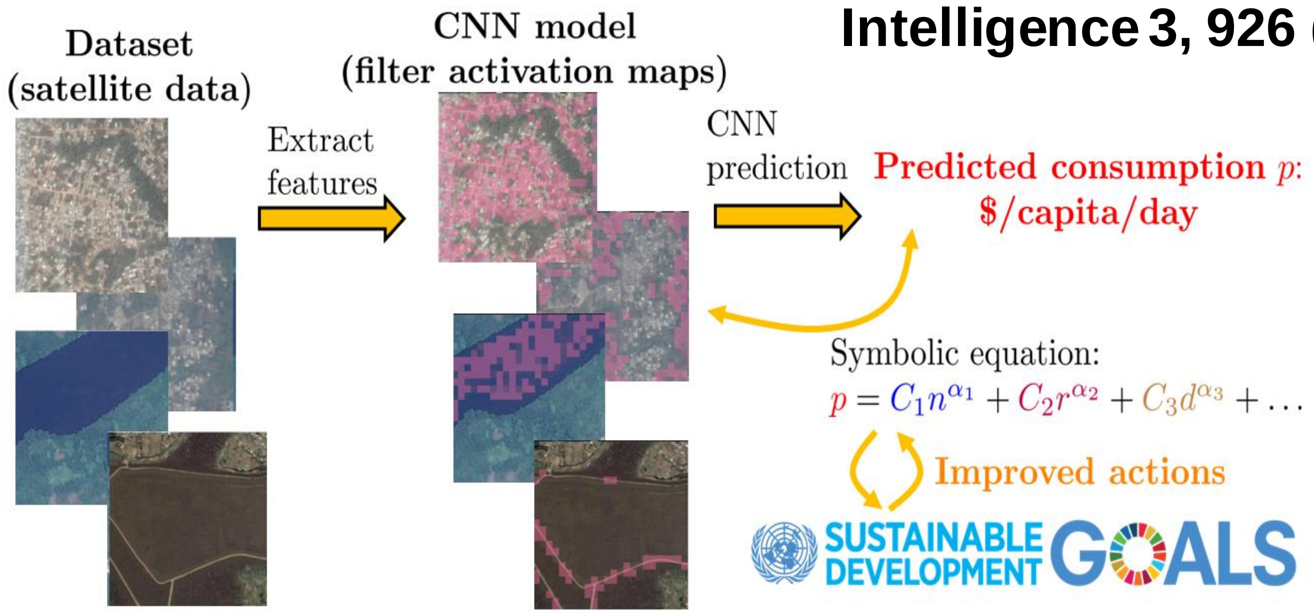

a
Machine learning
Accelerate simulations, improve scaling
Improve physical understanding
b
c
Resolved
Modeled
Energy
Wavenumber
$\tilde{u}_i = U_i + u_i$
$f(\nu_T)$
d
Direct numerical simulation
Turbulence modeling (LES and RANS)
Reduced-order models

Vinuesa & Brunton, Nature Comp. Science 2, 358–366 (2022)

Importance of developing interpretable deep-learning models

- Deep-learning models act as **black boxes**, and **lack of interpretability** is a challenge (Rudin, 2019).
- A-posterior **explainability** (e.g. saliency methods, which parts of the input contribute the most to the output) may be insufficient.
- Method **for a-posteriori interpretability**? Combination of **inductive biases and genetic programming** to produce **symbolic expressions**, increasing the output interpretability (Cranmer et al., 2020)

Vinuesa and Sirmacek, Nature Machine Intelligence 3, 926 (2021)


Dataset (satellite data)
Extract features
CNN model (filter activation maps)
CNN prediction
Predicted consumption p: \$/capita/day
Symbolic equation:
$p = C_1 n^{\alpha_1} + C_2 r^{\alpha_2} + C_3 d^{\alpha_3} + \dots$
Improved actions
SUSTAINABLE DEVELOPMENT GOALS

Outline of machine-learning applications to fluid mechanics

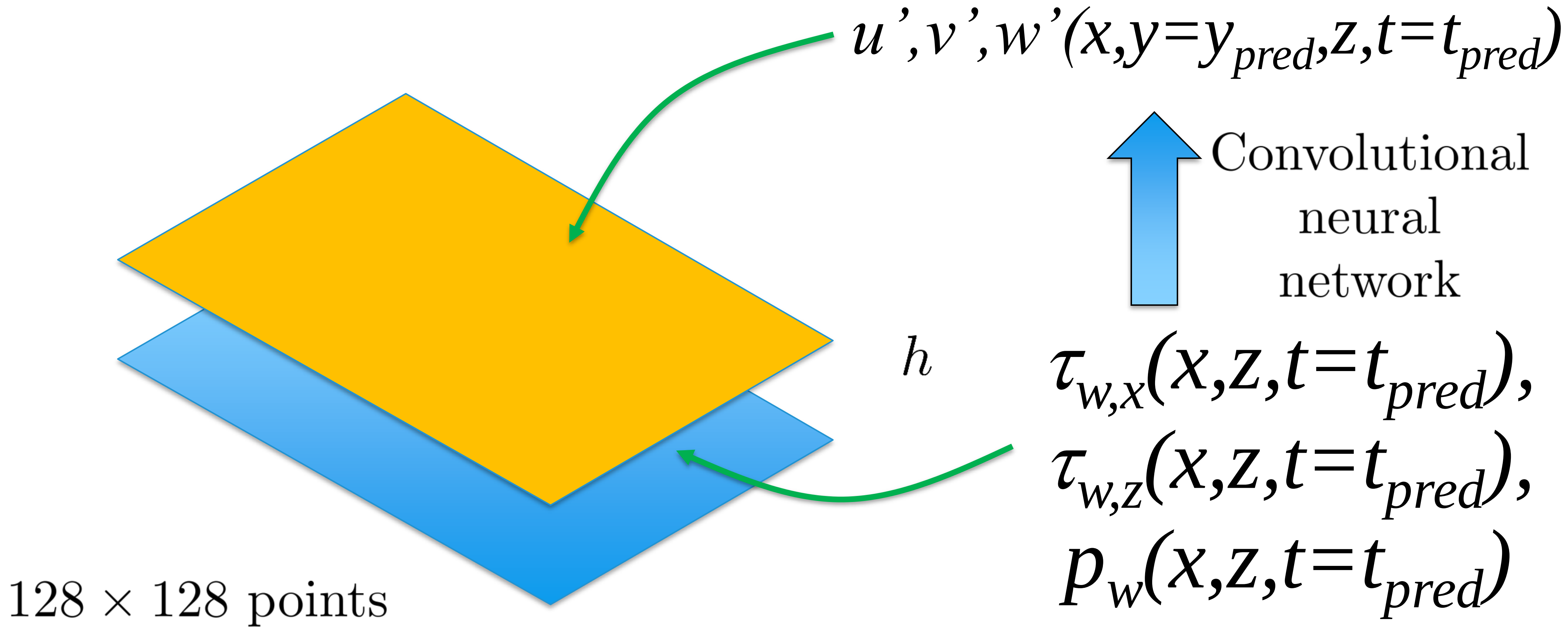
- Non-intrusive sensing in a turbulent channel via CNNs.
- Non-intrusive sensing with coarse measurements via GANs.
- Predictions for wall models via CNNs.
- Reduced-order models in turbulence via AEs.
- Flow control via DRL.

Outline of machine-learning applications to fluid mechanics

- Non-intrusive sensing in a turbulent channel via CNNs.
- Non-intrusive sensing with coarse measurements via GANs.
- Predictions for wall models via CNNs.
- Reduced-order models in turbulence via AEs.
- Flow control via DRL.

Flow reconstruction with a convolutional neural network (CNN)

- Prediction of **off-wall planes** based on information **at the wall**.
- **Additional motivation: closed-loop reactive control.**



$u',v',w'(x,y=y_{pred},z,t=t_{pred})$
Convolutional neural network
h
$\tau_{w,x}(x,z,t=t_{pred}),$
$\tau_{w,z}(x,z,t=t_{pred}),$
$p_w(x,z,t=t_{pred})$
128×128 points

Guastoni et al., J. Fluid Mech. 928, A27 (2021)

DNS of turbulent open channel flow

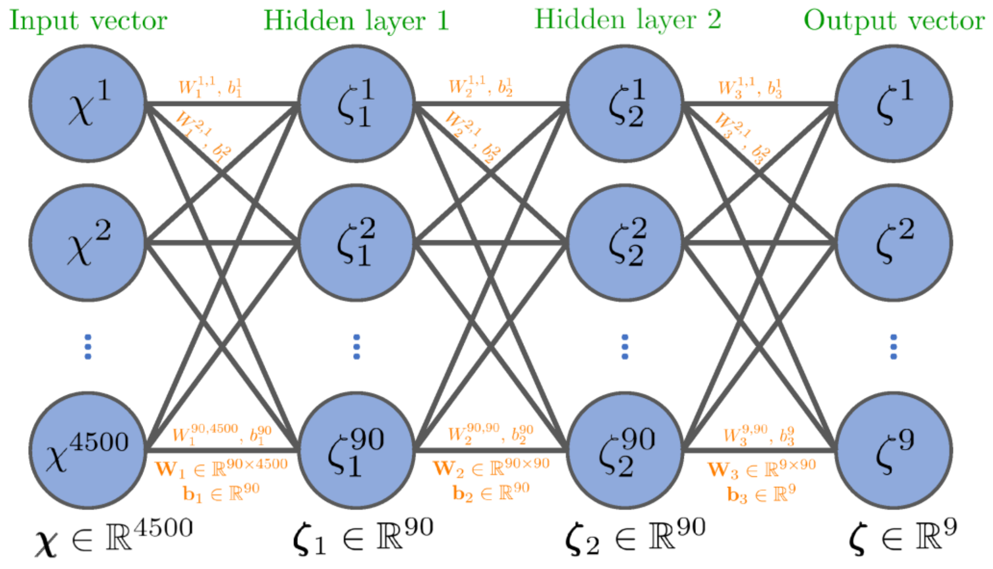
- DNS based on the Fourier-Chebyshev code **SIMSON** from KTH (Chevalier et al., 2007).
- Open channel flow (**simplicity**): no large-scale interactions between two walls.

- Re_τ=180 & 550
- $(L_x,L_y,L_z)=(4\pi h\times h\times 2\pi h)$
- No slip at y/h=0
- Symmetry at y/h=1

h

Guastoni et al., J. Fluid Mech. 928, A27 (2021)

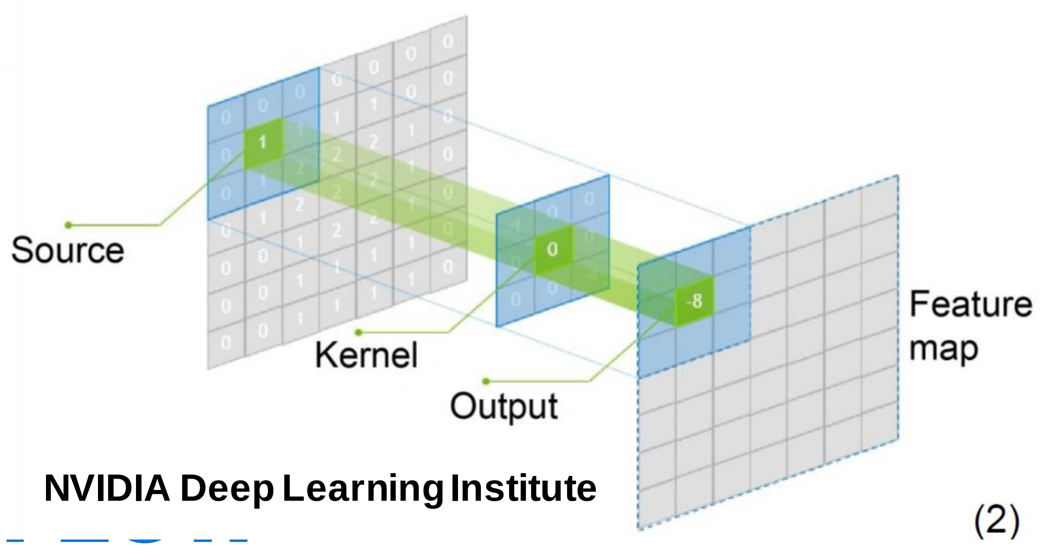
Neural-network architecture



Input vector
Hidden layer 1
Hidden layer 2
Output vector
χ^1
χ^2
χ^{4500}
ζ_1^1
ζ_1^2
ζ_1^{90}
ζ_2^1
ζ_2^2
ζ_2^{90}
ζ^1
ζ^2
ζ^9
$W_1^{1,1}, b_1^1$
$W_1^{2,1}, b_1^2$
$W_1^{90,4500}, b_1^{90}$
$\mathbf{W}_1 \in \mathbb{R}^{90\times4500}$
$\mathbf{b}_1 \in \mathbb{R}^{90}$
$W_2^{1,1}, b_2^1$
$W_2^{2,1}, b_2^2$
$W_2^{90,90}, b_2^{90}$
$\mathbf{W}_2 \in \mathbb{R}^{90\times90}$
$\mathbf{b}_2 \in \mathbb{R}^{90}$
$W_3^{1,1}, b_3^1$
$W_3^{2,1}, b_3^2$
$W_3^{9,90}, b_3^9$
$\mathbf{W}_3 \in \mathbb{R}^{9\times90}$
$\mathbf{b}_3 \in \mathbb{R}^{9}$
$\chi \in \mathbb{R}^{4500}$
$\zeta_1 \in \mathbb{R}^{90}$
$\zeta_2 \in \mathbb{R}^{90}$
$\zeta \in \mathbb{R}^{9}$

Srinivasan et al., Phys. Rev. Fluids (2019)

- **Fully connected layers:** weighted sum of the inputs and non-linear activation function.
- Very **large numbers of parameters**: they are prone to over-fitting the data.



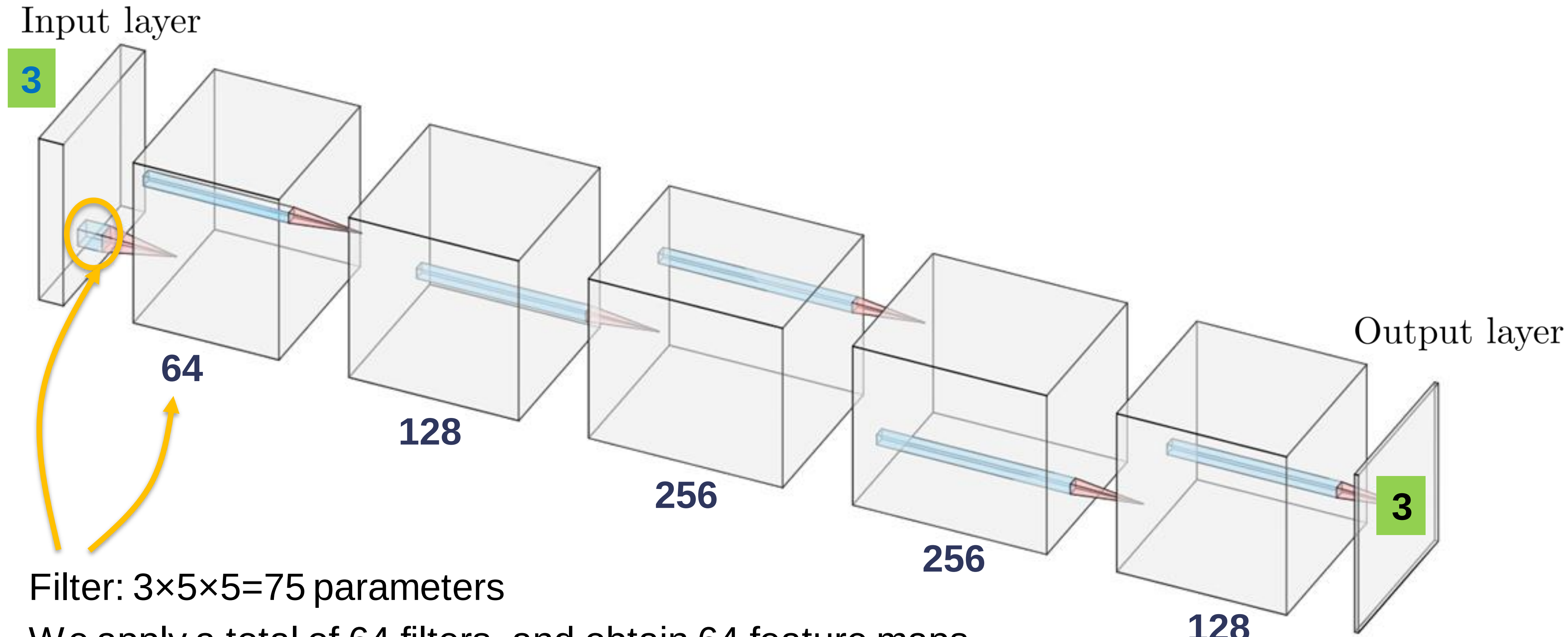
NVIDIA Deep Learning Institute

- **Convolutional layers: small set of parameters** (kernel of 3x3 or 5x5 **sweeps** the domain).
- Extensively used in **computer vision** (easy to train and suitable for identification of local features).

(2)

CNN architecture

- Prediction of **off-wall planes** based on information **at the wall** with CNNs (Li et al., 1984).



Filter: 3×5×5=75 parameters

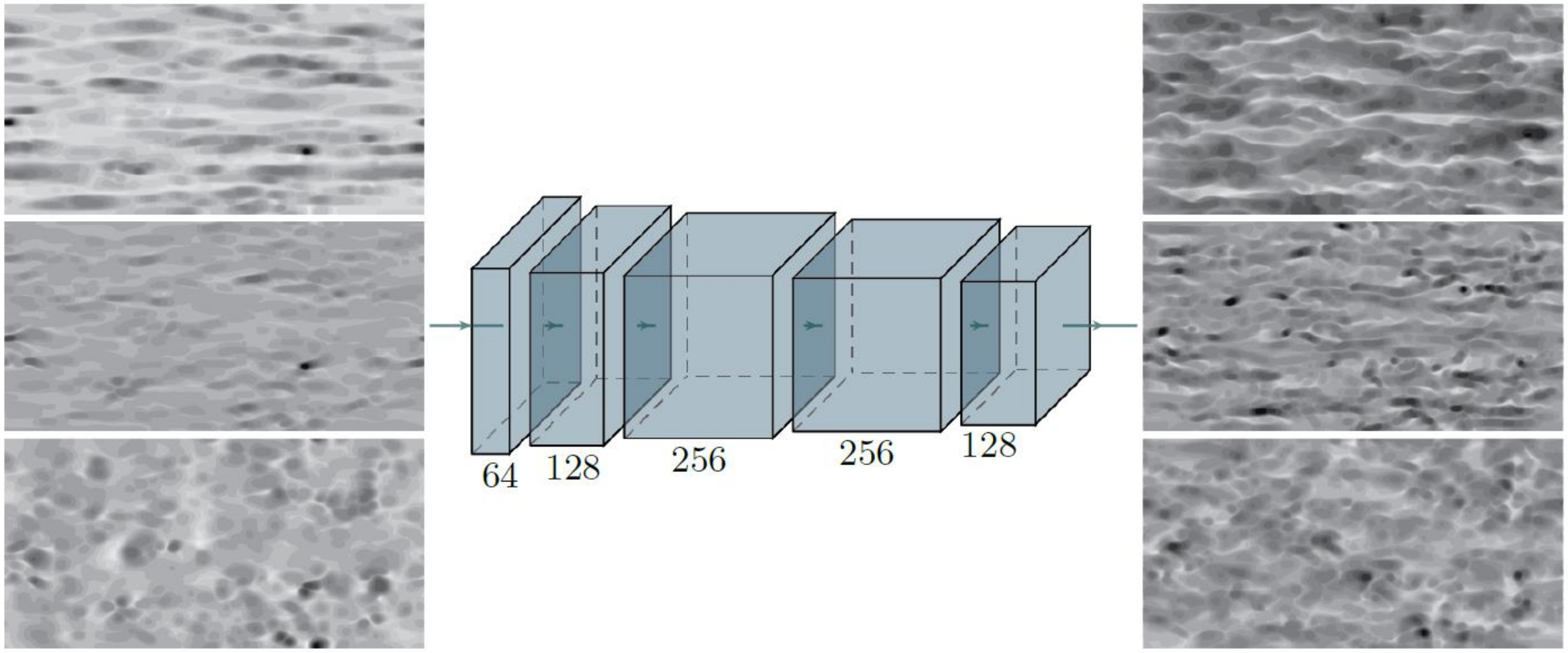
We apply a total of 64 filters, and obtain 64 feature maps

Each filter identifies one (or several) feature. More complex towards the output.

Schematic representation obtained from: http://alexlenail.me/NN-SVG/index.html

An extension to Re_τ=550

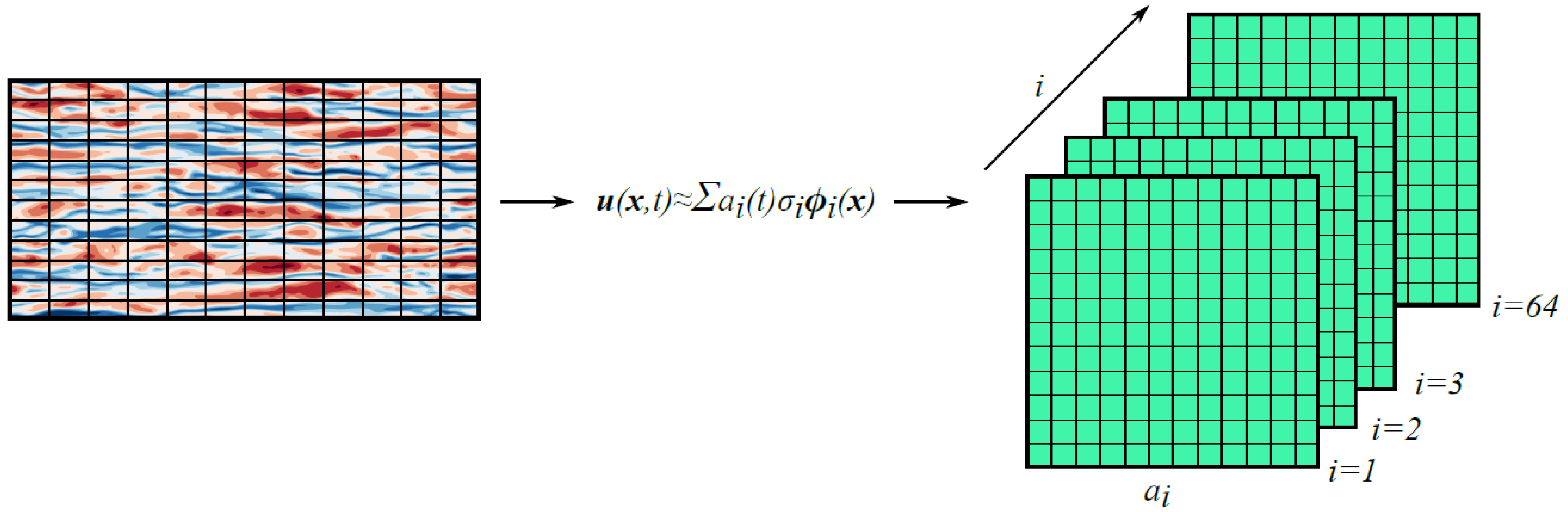
- We consider **3 input fields** (adding the pressure).
- We predict the **3 velocity fluctuations** (scaled to be in the same range).
- Note that we consider **periodic padding** (CNN output is deterministic, local influence).



Guastoni et al., J. Fluid Mech. 928, A27 (2021)

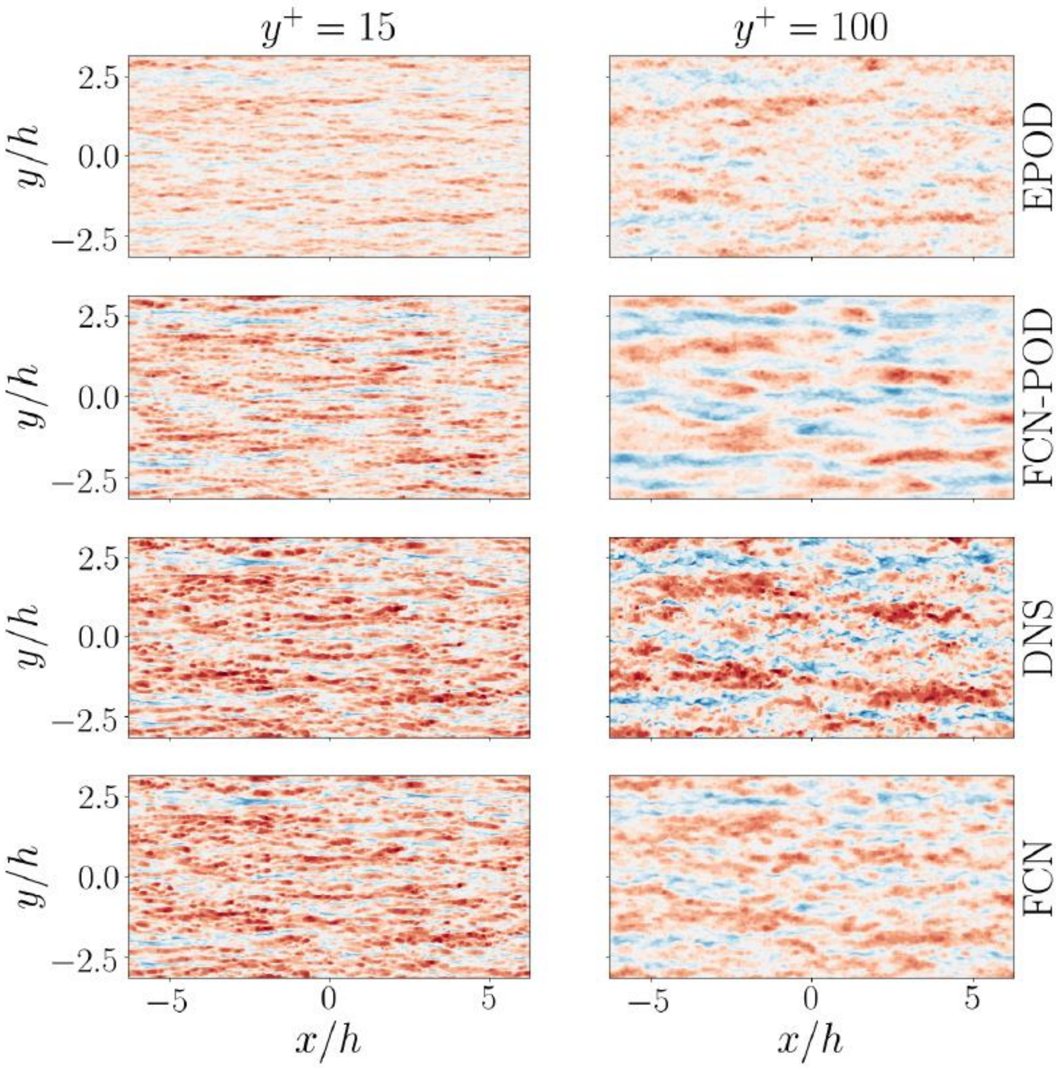
A new method: FCN-POD

- In addition to the direct field prediction, we consider predicting the **POD coefficients** (embedding known physical information).
- We split the target plane into **subdomains** (reconstruction with fewer POD modes).


$\boldsymbol{u}(\mathbf{x},t) \approx \Sigma a_i(t)\sigma_i\phi_i(\mathbf{x})$
i
$i=64$
$i=3$
$i=2$
$i=1$
a_i

Güemes et al., Phys. Fluids (2019)
Guastoni et al., J. Fluid Mech. 928, A27 (2021)

Instantaneous predictions at Re_τ=550

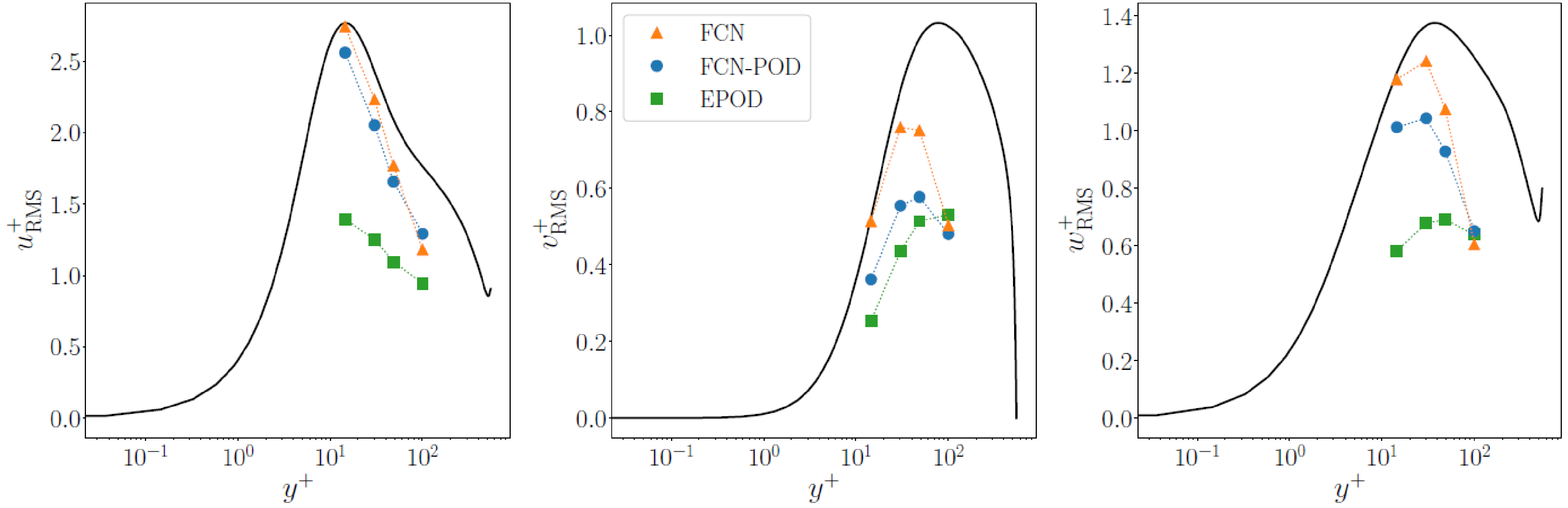


- **EPOD** is formally equivalent to LSE (linear method).
- **Close to the wall FCN is best**, **farther from it FCN-POD is better** due to the encoded information in the POD modes.

Guastoni et al., J. Fluid Mech. 928, A27 (2021)

Turbulence statistics at Re_τ=550

- **Close to the wall FCN is best**, **farther from it FCN-POD is better** due to the encoded information in the POD modes.

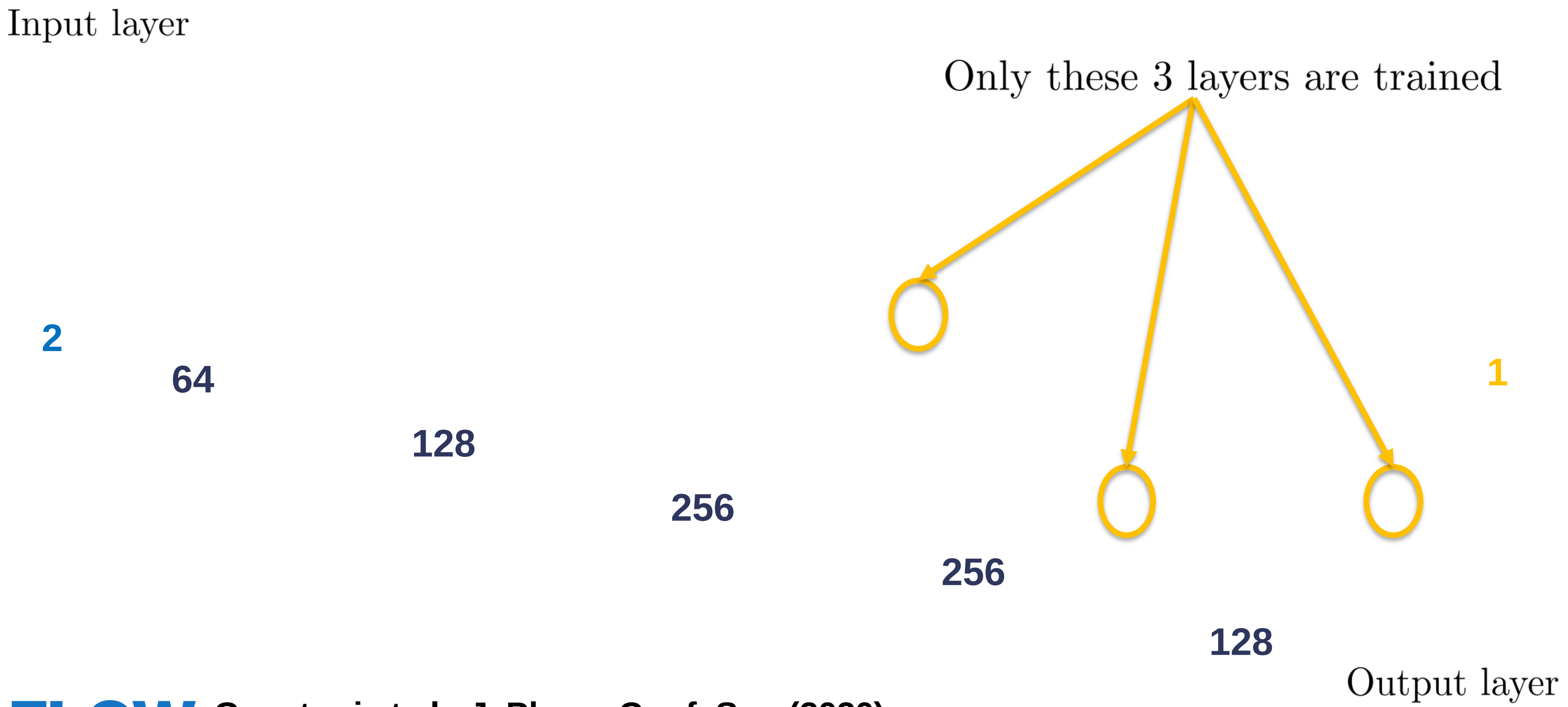


$E^+_{\mathrm{RMS}}(\cdot)$ [%]		$y^+ = 15$	$y^+ = 30$	$y^+ = 50$	$y^+ = 100$
	EPOD	49.69	48.53	47.62	46.53
u	FCN	0.98 (±0.66)	8.10 (±0.62)	15.33 (±0.22)	33.06 (±0.30)
	FCN-POD	7.53 (±0.27)	15.53 (±0.39)	20.75 (±0.85)	26.79 (±0.36)

Guastoni et al., J. Fluid Mech. 928, A27 (2021)

Improving training performance: Transfer learning at Re_τ=180

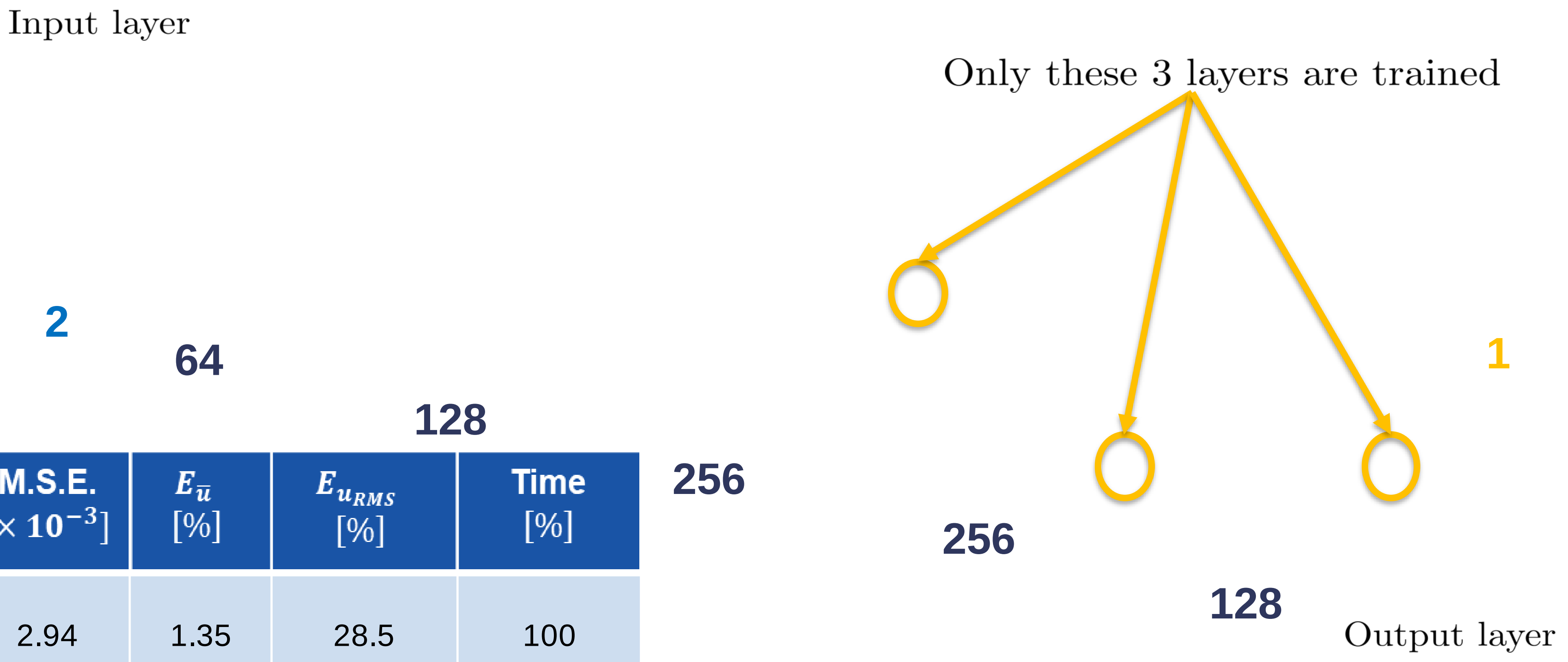
- The deeper layers identify progressively **larger features**.
- Fix the first 3 layers (trained at y^+=15) and **train the last 3** (to predict at y^+=50).



Guastoni et al., J. Phys.: Conf. Ser. (2020)

Improving training performance: Transfer learning at Re_τ=180

- The deeper layers identify progressively **larger features**.
- Fix the first 3 layers (trained at y^+=15) and **train the last 3** (to predict at y^+=50).



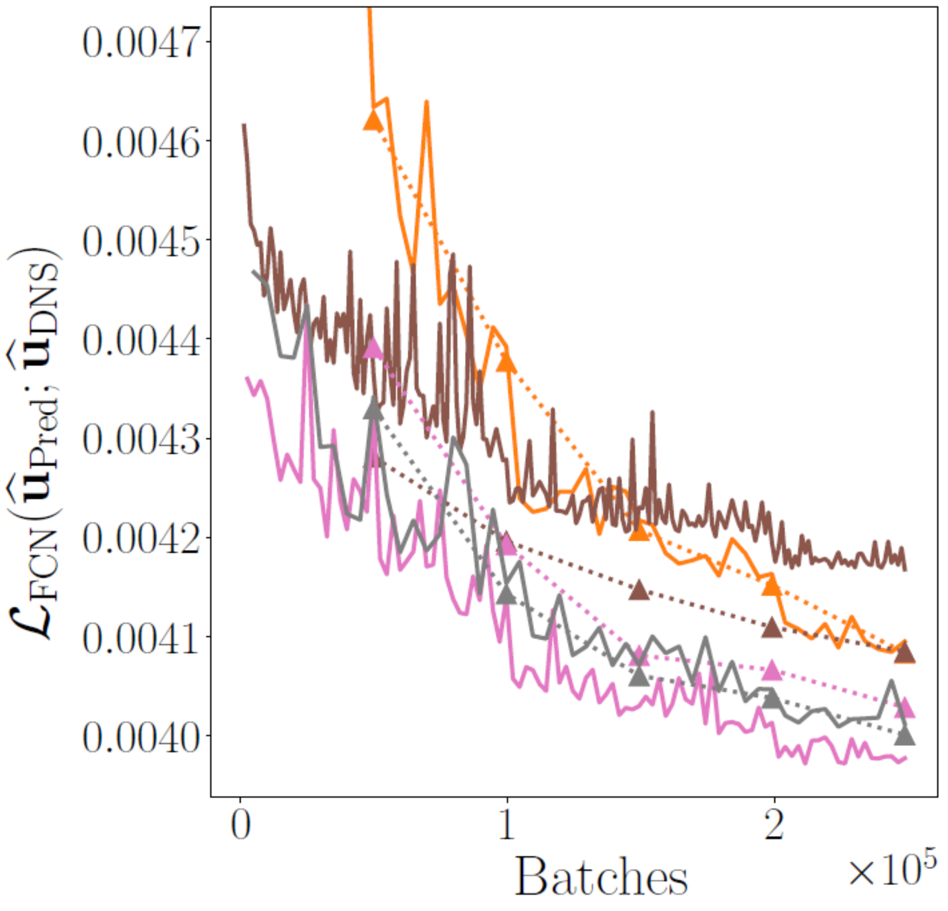
	M.S.E. $[\times 10^{-3}]$	$E_{\overline{u}}$ [%]	$E_{u_{RMS}}$ [%]	Time [%]
Fully trainable	2.94	1.35	28.5	100
Transfer learning	**3.17**	**0.5**	**30.2**	**23**

Guastoni et al., J. Phys.: Conf. Ser. (2020)

Transfer learning from Re_τ=180 to 550

Guastoni et al., J. Fluid Mech. 928, A27 (2021)

- We **initialize the training** for Re_τ=550 with the weights from the Re_τ=180 case.
- The **learning rate** needs to be reduced.



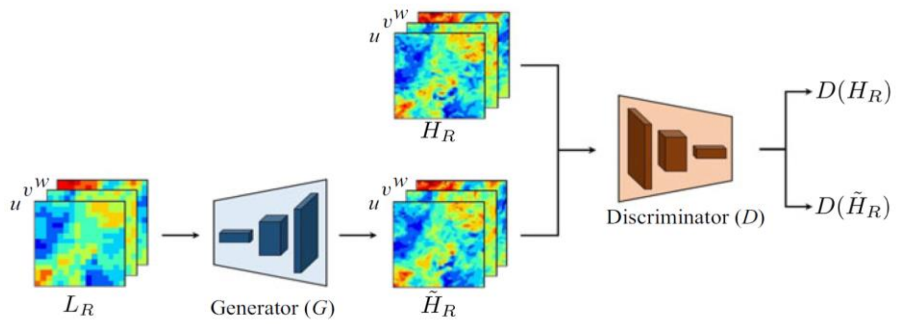
$E^+_{\mathrm{RMS}}(\cdot)$ [%]		$y^+ = 15$	$y^+ = 30$	$y^+ = 50$	$y^+ = 100$
u	Ref.	0.98 (±0.66)	8.10 (±0.62)	15.33 (±0.22)	33.06 (±0.30)
	100%	1.22	7.15	16.09	32.75
	50%	2.94	7.11	16.33	34.11
	25%	1.15	7.74	14.78	33.78
v	Ref.	1.74 (±0.11)	11.21 (±1.41)	24.20 (±1.38)	50.82 (±0.26)
	100%	1.86	9.40	24.40	51.59
	50%	2.40	9.46	25.96	50.90
	25%	1.71	11.33	23.15	50.43
w	Ref.	1.86 (±0.60)	9.03 (±0.31)	21.21 (±1.27)	51.83 (±0.38)
	100%	1.75	8.65	21.05	53.04
	50%	2.61	7.70	21.34	52.60
	25%	1.22	9.67	20.35	49.75

- **Similar error levels with 25% of the training data!**

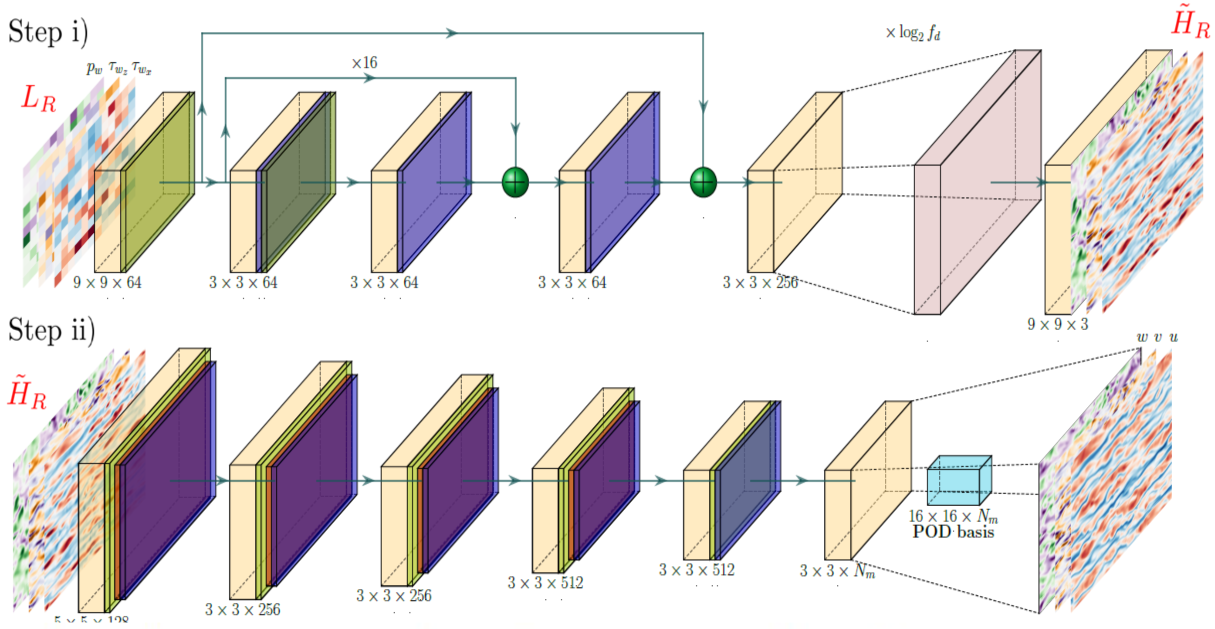
Outline of machine-learning applications to fluid mechanics

- Non-intrusive sensing in a turbulent channel via CNNs.
- Non-intrusive sensing with coarse measurements via GANs.
- Predictions for wall models via CNNs.
- Reduced-order models in turbulence via AEs.
- Flow control via DRL.

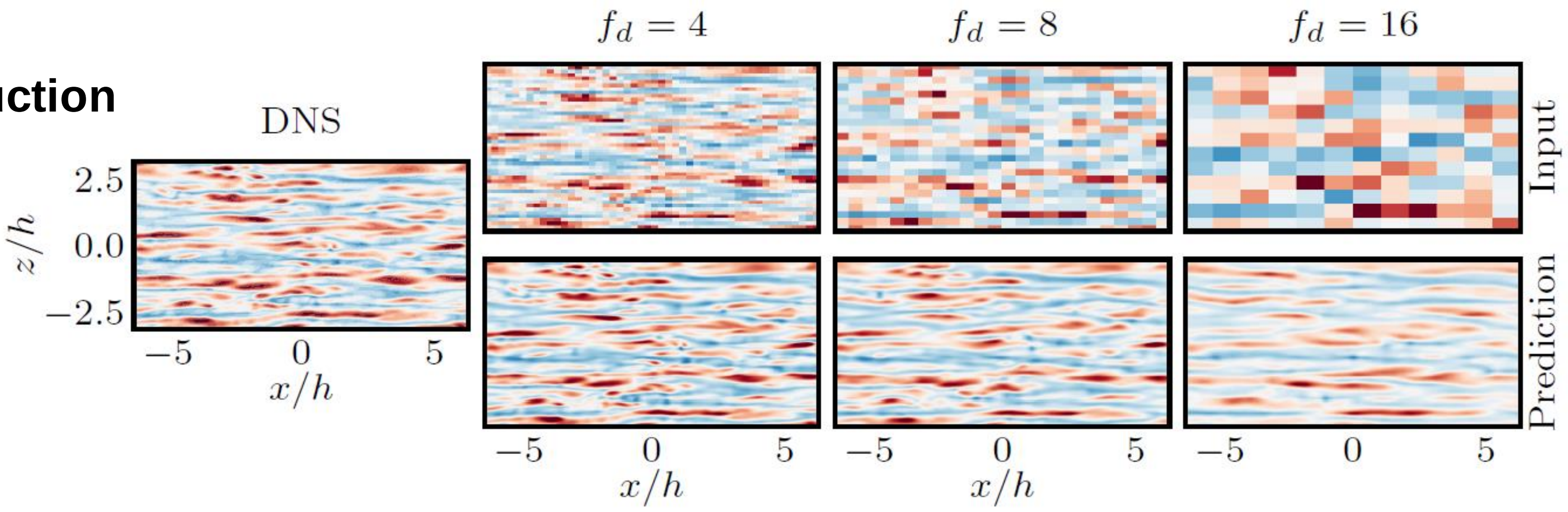
From sparse measurements to high-resolution predictions using GANs



Kim et al., (2021)



Wall reconstruction



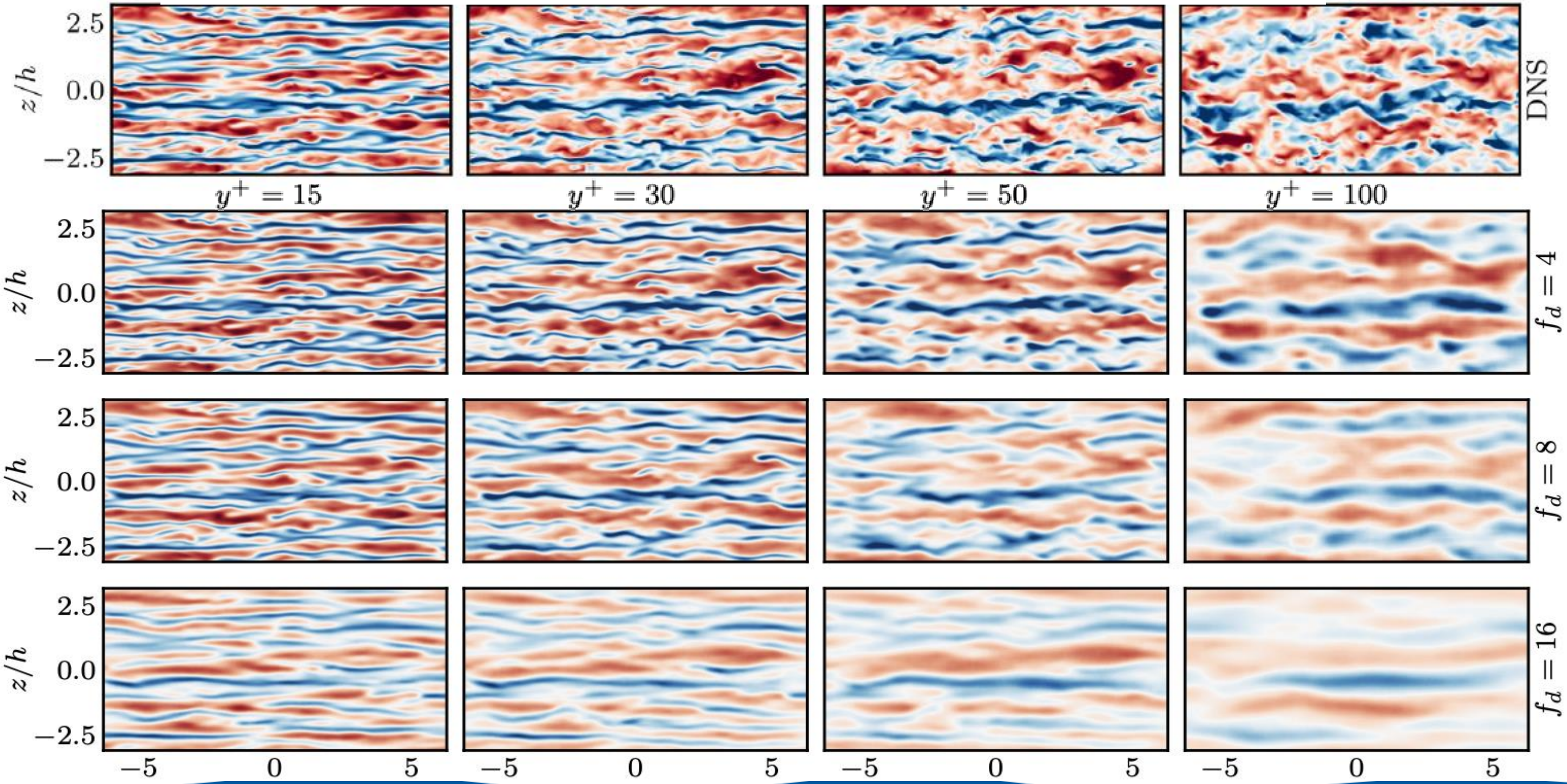
Güemes et al., Phys. Fluids 33, 075121 (2021)

Flow reconstruction from coarse measurements

1) Different range of scales in each study.
2) Parameter f_d is not an adequate parameter to compare turbulence databases.

$$\tilde{f}_d = f_d \sqrt{\Delta x^{+2} + \Delta z^{+2}}$$

Güemes et al., Phys. Fluids 33, 075121 (2021)



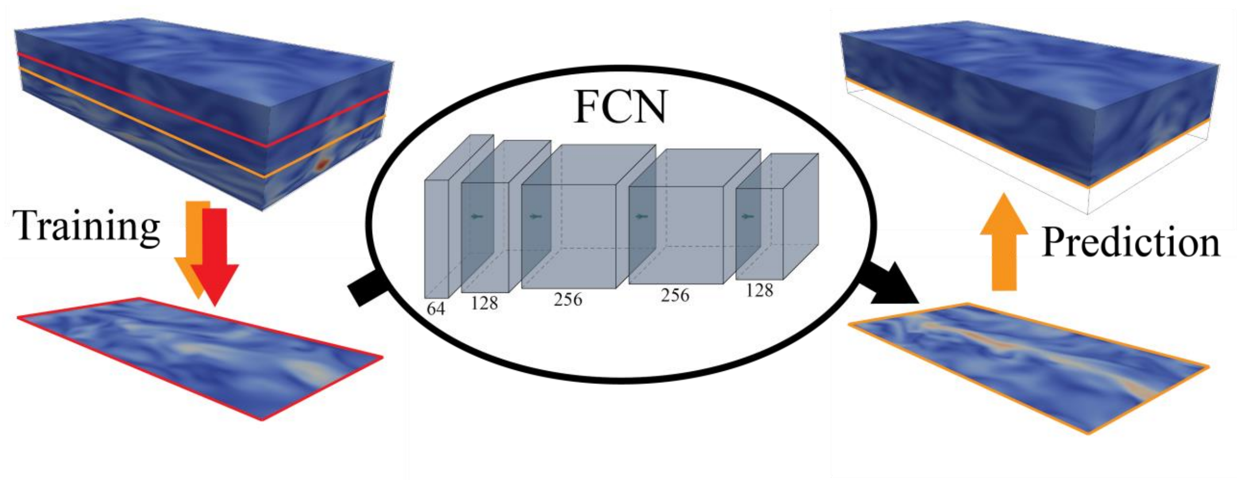
Outline of machine-learning applications to fluid mechanics

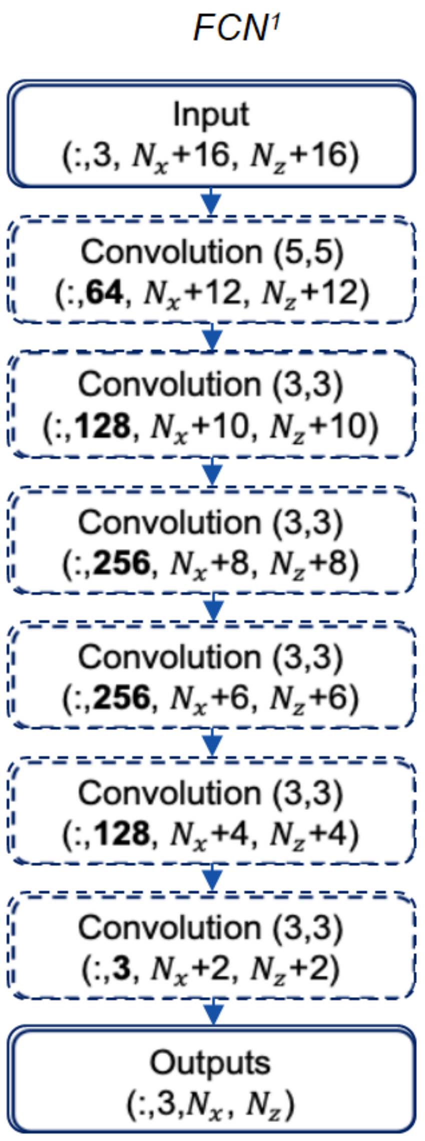
- Non-intrusive sensing in a turbulent channel via CNNs.
- Non-intrusive sensing with coarse measurements via GANs.
- Predictions for wall models via CNNs.
- Reduced-order models in turbulence via AEs.
- Flow control via DRL.

FCN model for predictions closer to the wall

Balasubramanian et al. (2021), arXiv:2107.07340

- Is it possible to predict the inner region based on the outer region?
- **Application to off-wall boundary conditions?**



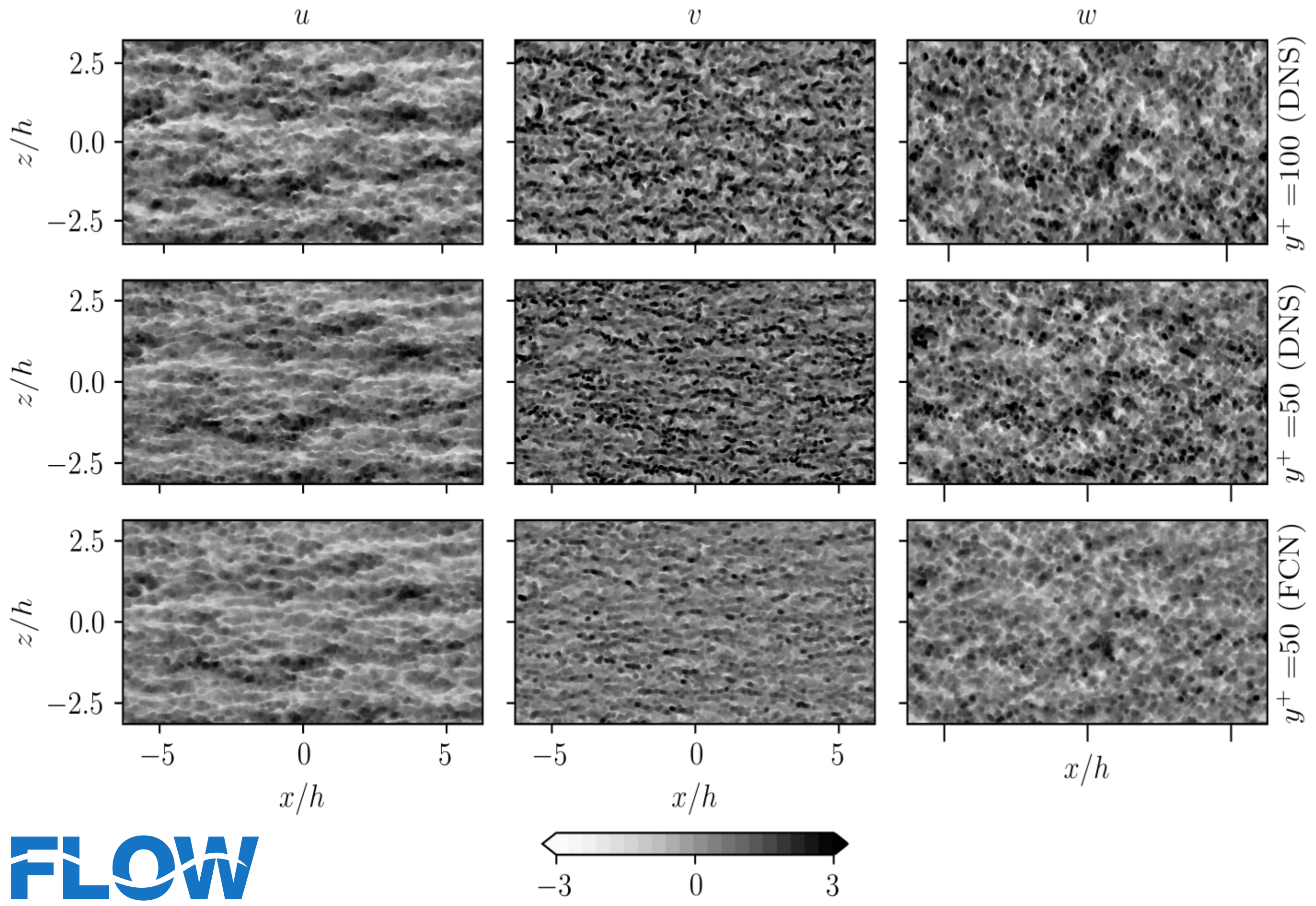


FCN[1]
Input
(:,3, N_x+16, N_z+16)
Convolution (5,5)
(:,**64**, N_x+12, N_z+12)
Convolution (3,3)
(:,**128**, N_x+10, N_z+10)
Convolution (3,3)
(:,**256**, N_x+8, N_z+8)
Convolution (3,3)
(:,**256**, N_x+6, N_z+6)
Convolution (3,3)
(:,**128**, N_x+4, N_z+4)
Convolution (3,3)
(:,**3**, N_x+2, N_z+2)
Outputs
(:,3,N_x, N_z)

Activation function - ReLU

Self similarity in the overlap region: Off-wall boundary conditions

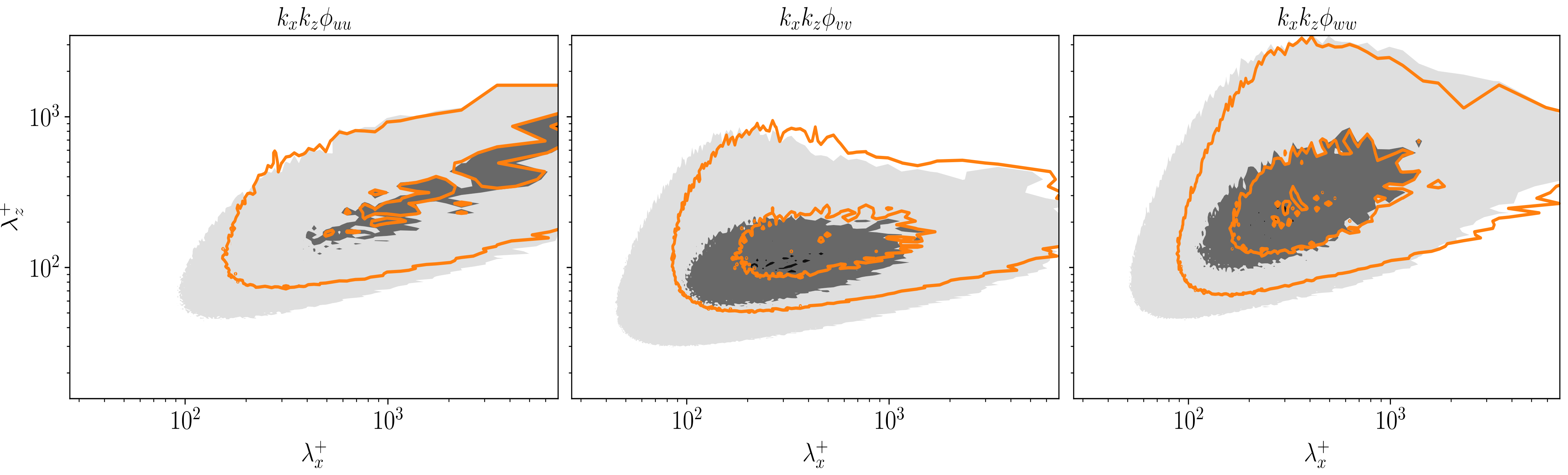
- Exploit the self similarity of scales with y.
- **Mizuno and Jiménez (2013)**: Channel at Re_τ=1,000; from y/h=0.2 to 0.1, rescaling.
- Predictions of y^+=50 based on data at 100 with CNNs, Re_τ=550.



- **Small receptive field:** information from large structures cannot be captured.
- **Missing scales:** this information is not present in the large scales.

Balasubramanian et al. (2021), arXiv:2107.07340

Spectra and statistics at Re_τ=550, y^+=50



	u	v	w
Parameters	i		
$\mathcal{L}(i_{\mathrm{FCN}}; i_{\mathrm{DNS}})/i_{\mathrm{RMS}}^2$	0.365 ± 0.005	0.548 ± 0.005	0.468 ± 0.006
$E_{\mathrm{RMS}}(i)$ [%]	19.04 ± 1.17	31.8 ± 1.25	25.7 ± 1.18
$R_{\mathrm{FCN;DNS}}$	0.801 ± 0.003	0.681 ± 0.004	0.733 ± 0.004

- **Small receptive field:** max pooling to better identify features.
- **Missing scales:** Perhaps GANs-based recontruction **(Güemes et al., Phys. Fluids, 2021).**

Balasubramanian et al. (2021), arXiv:2107.07340

Outline of machine-learning applications to fluid mechanics

- Non-intrusive sensing in a turbulent channel via CNNs.
- Non-intrusive sensing with coarse measurements via GANs.
- Predictions for wall models via CNNs.
- Reduced-order models in turbulence via AEs.
- Flow control via DRL.

Non-linear orthogonal modal decomposition in turbulent flows via autoencoders

Objectives:

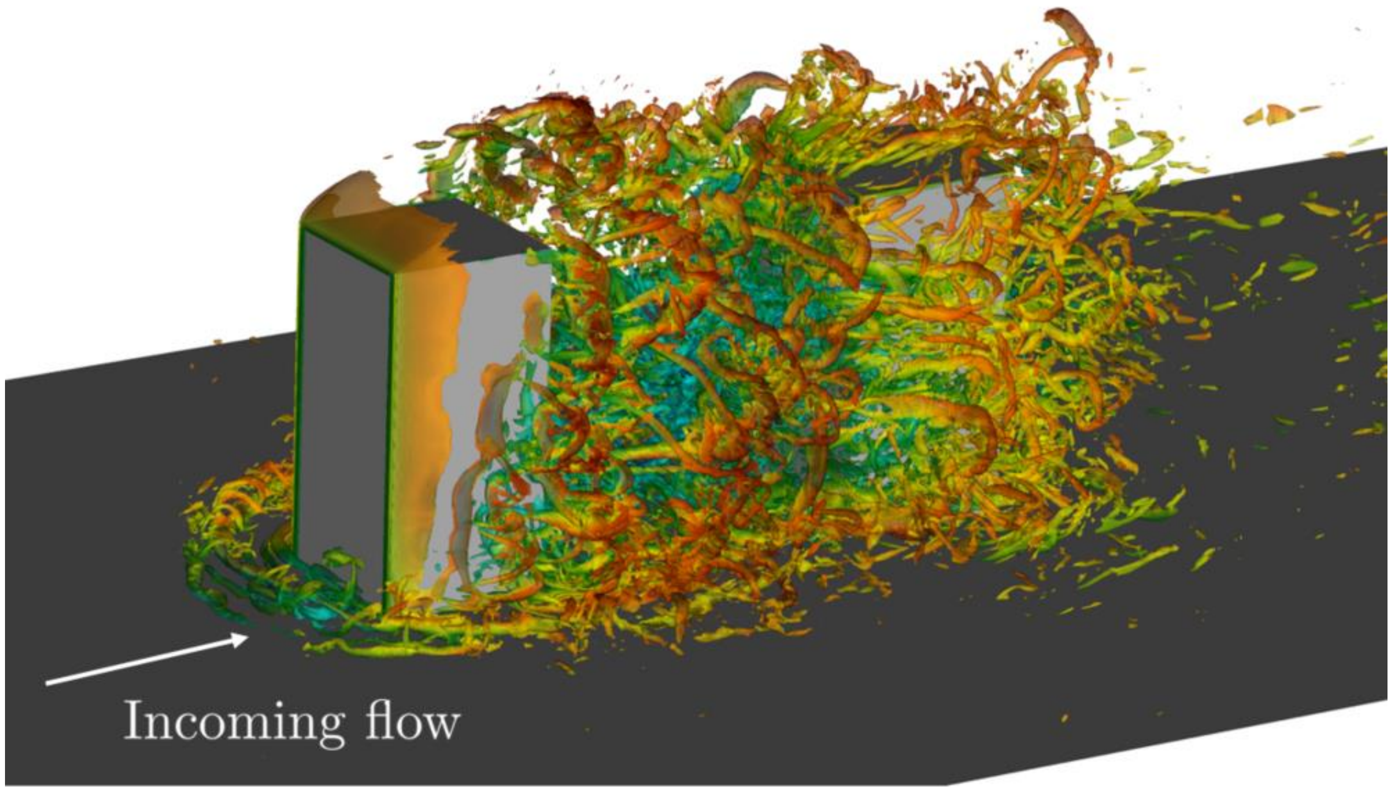
- Learning a compact latent representation of high-dimensional data by accounting for the **non-linearity** in the low-dimensional mapping.
- Ranking modes based on their contribution to the reconstruction (**Optimality**).
- Learning a compact, **near-orthogonal** and parsimonious latent representation of high-dimensional data by minimizing the correlation between the latent variables, as well as penalizing the size of the latent vector.
- Extraction of **interpretable non-linear modes**.

Eivazi et al., Expert Syst. Appl. 202, 117038 (2022)

Turbulent flow in a simplified urban environment

We employ a database (Torres et al., 2021; Lazpita et al., 2022) of the flow through a simplified urban environment:

- Well-resolved large-eddy simulation (LES) using Nek5000.
- The x–z cross-sectional velocity fields at y = 0.5h are extracted and used as the input data.
- **1,200 instantaneous fields.**
- Number of grid points in x and z: $(N_x, N_z) = (96, 192)$.



Eivazi et al., Expert Syst. Appl. 202, 117038 (2022)

Convolutional-neural-network-based autoencoders (CNN-AEs)

An autoencoder comprises two parts: an encoder that maps the input data to a low-dimensional latent space $\boldsymbol{x} \mapsto \boldsymbol{r}$, and a decoder that projects the latent vector $\boldsymbol{r}$ back to the reference space $\boldsymbol{r} \mapsto \tilde{\boldsymbol{x}}$.

$$\mathcal{F} = \mathcal{D} \circ \mathcal{E}, \quad \tilde{\mathbf{x}} = \mathcal{F}(\mathbf{x}; \mathbf{w}), \quad \mathcal{L}_{\mathrm{rec}} = \epsilon(\mathbf{x}, \tilde{\mathbf{x}}), \tag{7}$$

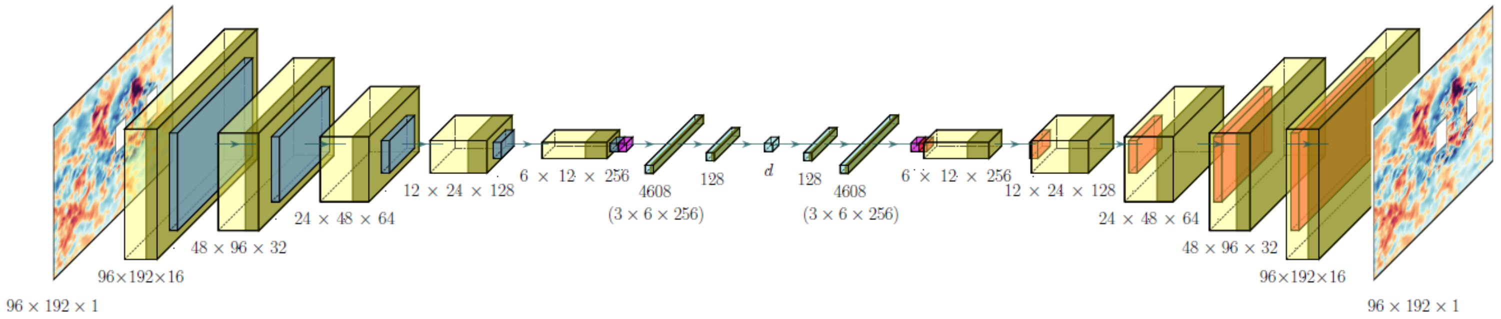


Figure: Schematic view of the CNN-AE architecture. The color coding for each layer is: 2D-convolution (🟨), tanh activation (🟫), max pooling (🟦), reshape (🟪), fully-connected layer (⬜), upsampling (🟧).

- **Latent variables** contain temporal information.
- **Obtain modes** by using decoder, setting rest of latent variables to zero.

Eivazi et al., Expert Syst. Appl. 202, 117038 (2022)

CNN-based hierarchical autoencoders (CNN-HAEs)

Fukami et al. (2020) proposed a CNN-based hierarchical autoencoder for modal decomposition of fluid flows in order to extract the modes ranked in terms of their contribution to the reconstruction:

d=1, $[\mathbf{r}_1]$
$\Rightarrow \mathbf{r}_1$

d=2, $[\mathbf{r}_1, \mathbf{r}_2]$
$\Rightarrow \mathbf{r}_2$

d=3, $[\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3]$
$\Rightarrow \mathbf{r}_3$

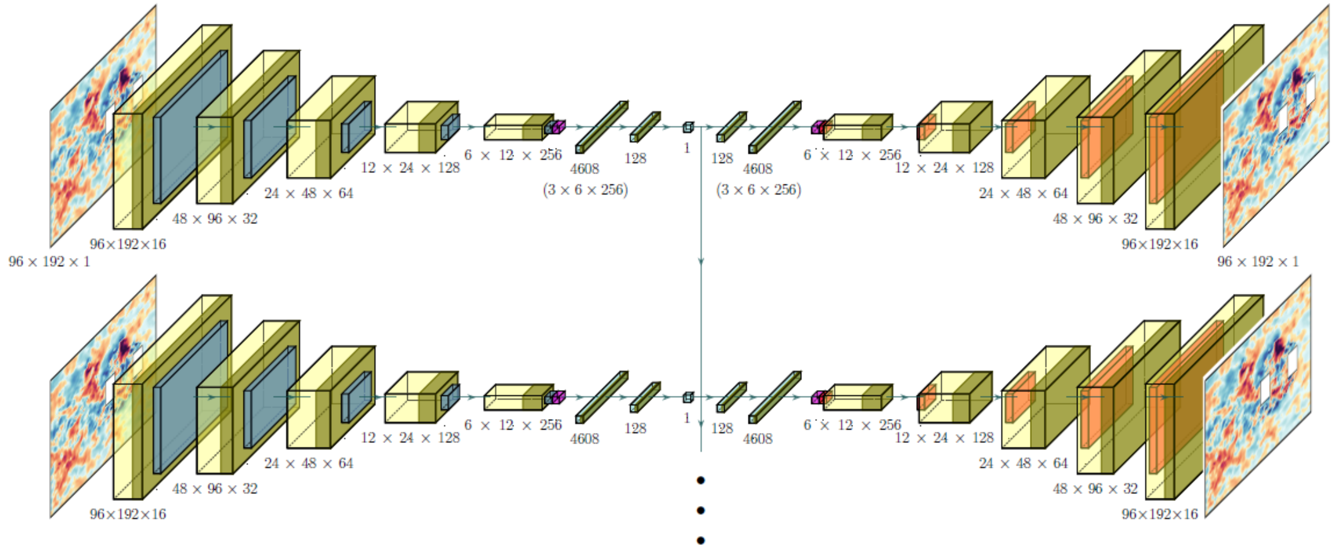


Figure: Schematic view of the CNN-HAE architectures with the same color coding as CNN-AE.

Eivazi et al., Expert Syst. Appl. 202, 117038 (2022)

CNN-based β-variational autoencoders (CNN-βVAE) Introducing stochasticity

Let us consider:

- p(**x**) is the distribution of the **original data.**
- p(**r**) is the distribution of the data in the **latent space.**
- We aim to maximize:

$$p_{\boldsymbol{\theta}}(\mathbf{x}) = \int p_{\boldsymbol{\theta}}(\mathbf{x}|\boldsymbol{r})p(\boldsymbol{r})\mathrm{d}\boldsymbol{r},$$

- p_θ(**x**) is the **marginal likelihood:** approximation of p(**x**) with parameters θ.

- In VAEs, p_θ(**x**|**r**) is a **probabilistic decoder** (**generative model**).
- In VAEs, q_θ(**r**|**x**) is a **probabilistic encoder** (**recognition model**).

Eivazi et al., Expert Syst. Appl. 202, 117038 (2022)

CNN-based β-variational autoencoders (CNN-βVAE) Introducing stochasticity

We assume that the **encoder** $q_\theta(\mathbf{r}|\mathbf{x})$ is a Gaussian distribution:

$$\log q_{\phi(\boldsymbol{r}|\mathbf{x})} = \log \mathcal{N}(\boldsymbol{r}; \boldsymbol{\mu}, \boldsymbol{\sigma}^2 \mathbf{I}),$$

We sample from $q_\theta(\mathbf{r}|\mathbf{x})$ using an auxiliary normally-distributed random number:

$$\boldsymbol{r} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \varepsilon \text{ where } \varepsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

The βVAE loss function is:

$$\mathcal{L}(\boldsymbol{\theta}, \phi; \mathbf{x}) = \mathcal{L}_{\mathrm{rec}} - \frac{\beta}{2} \sum_{i=1}^{d} (1 + \log(\sigma_i^2) - \mu_i^2 - \sigma_i^2)$$

β is **a penalization factor** which promotes learning **statistically-independent latent variables** (minimizing the distance between p(**r**) and the product of its marginals), and also penalizing the **size of the latent vector d.** Disentangled and parsimonious latent space.

Eivazi et al., Expert Syst. Appl. 202, 117038 (2022)

CNN-based β-variational autoencoders (CNN-βVAE) Introducing stochasticity

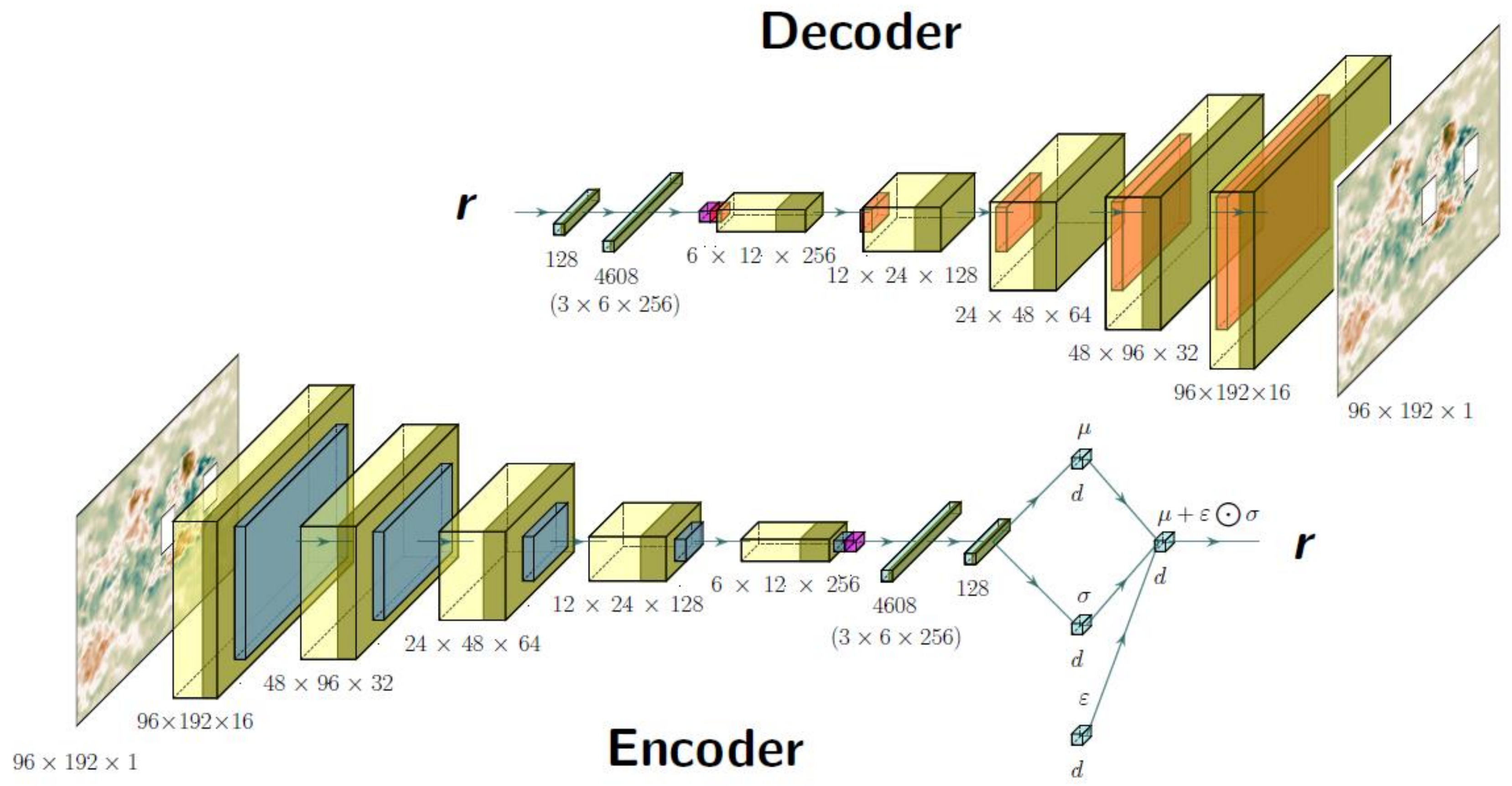


Figure: Schematic view of the CNN-βVAE with the same color coding as CNN-AE.

Eivazi et al., Expert Syst. Appl. 202, 117038 (2022)

Flow-field reconstruction

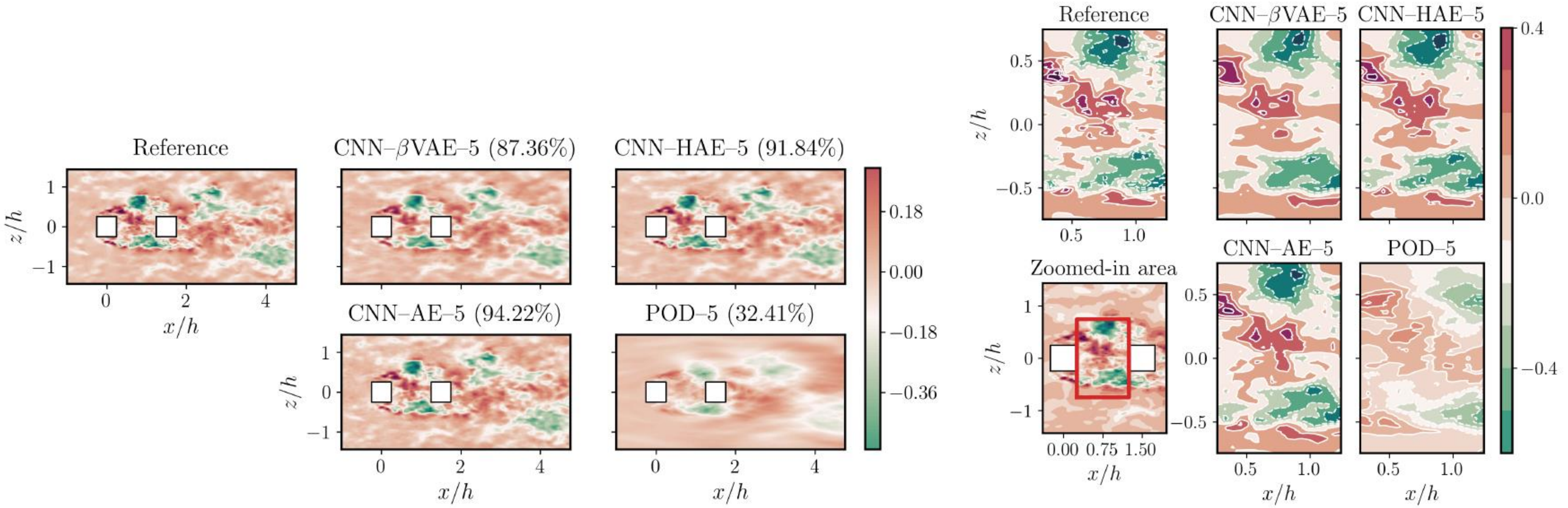


Figure: Reconstruction of the fluctuating component of the streamwise velocity obtained from different methods, as indicated in each panel, in comparison with the reference data. The value in brackets on each panel indicates the obtained E_u.

Eivazi et al., Expert Syst. Appl. 202, 117038 (2022)

Orthogonality: determinant of the cross-correlation matrix

Our results show that it is possible to motivate the independence of the latent variables using CNN-βVAEs and obtain near-orthogonal modes, where $\det_{\mathbf{R}}$ is equal to 99.20 for the CNN-βVAE method and 87.59 for the CNN-AE technique.

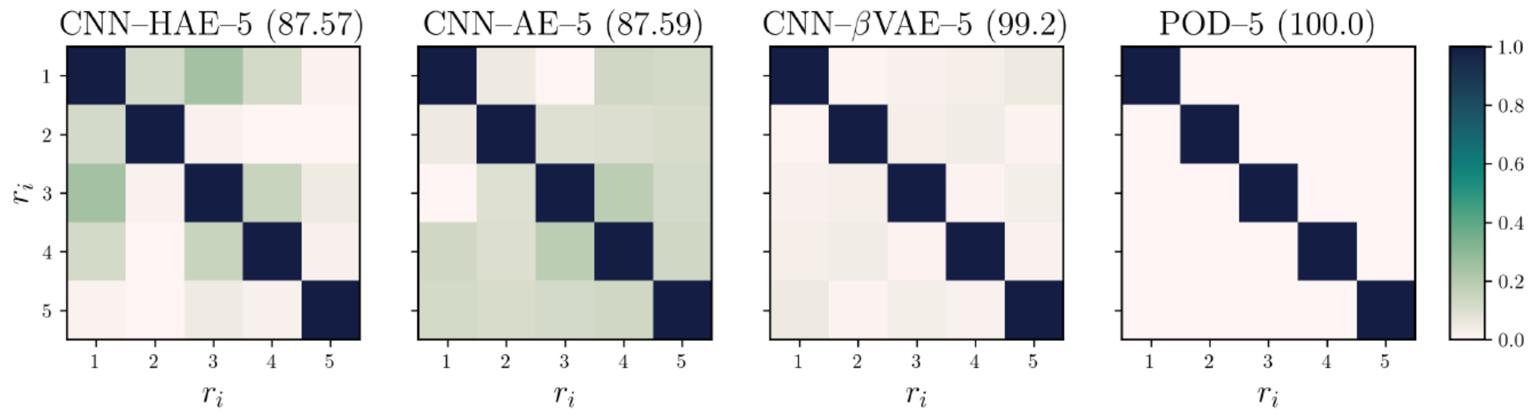


Figure: Correlation matrix $\mathbf{R}$ for the latent variables obtained from different models as indicated on the panels. The value in brackets indicates the corresponding $\det_{\mathbf{R}}$ for each case.

Eivazi et al., Expert Syst. Appl. 202, 117038 (2022)

Effect of the penalization factor β

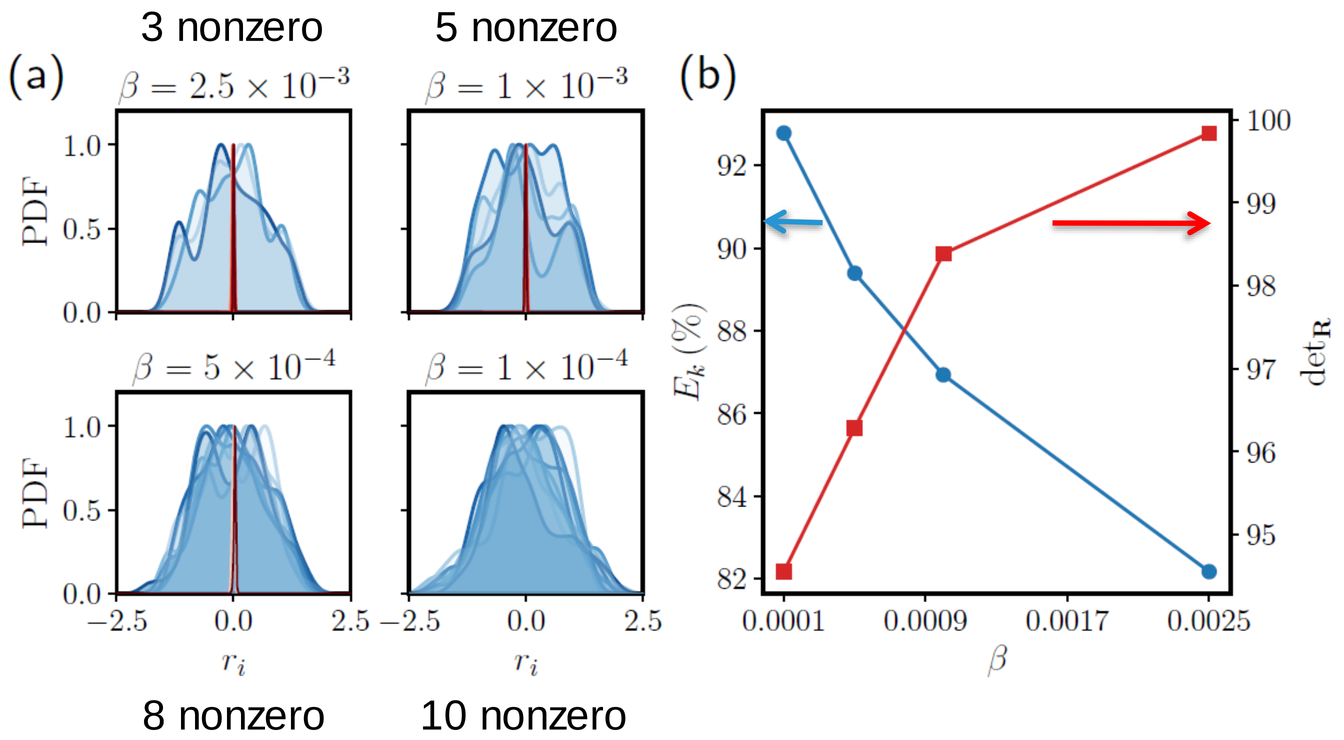


Figure: Effects of the penalization factor β on the performance of the CNN-βVAE models. (a) Normalized probability density function (PDF) of the latent variables obtained from models with different values of β as indicated in each panel. The variables with considerably large values are colored by blue and the variables with very small deviations from zero are depicted by red. (b) Effect of β on E_k (blue) and $\det_{\mathbf{R}}$ (red).

Eivazi et al., Expert Syst. Appl. 202, 117038 (2022)

Optimality: ranking CNN-βVAE modes and interpretability

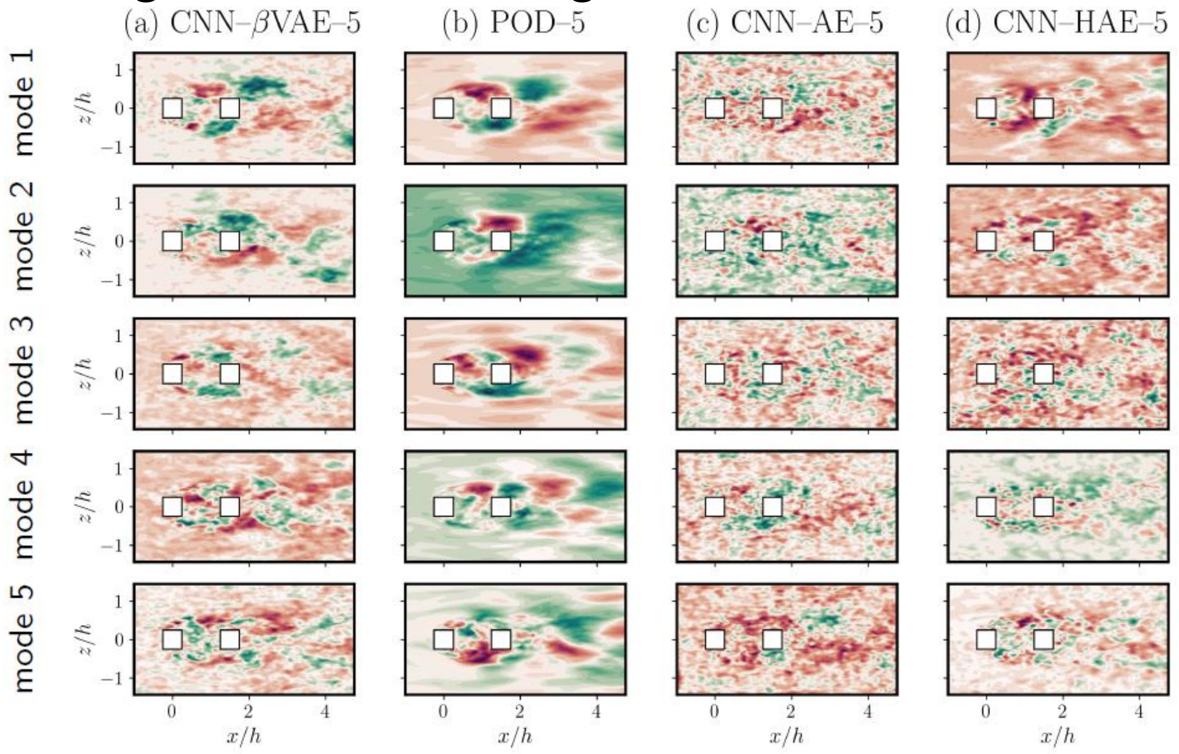
Ranking process:

- Make all latent variables zero except 1, and take the one with highest reconstructed energy.
- The second mode is the one that, together with the first one, has best reconstruction.
- And so on for all the latent variables.

Interpretability: Orthogonal modes related to **large-scale shedding** also found in POD.

Algorithm 2: Ranking the CNN-βVAE modes.

```
Model trained encoder $\mathcal{E}$ and decoder $D$ of the CNN-$\beta$VAE.
Data $\boldsymbol{x}$
Initialize $\boldsymbol{J}$ vector containing indices of the ranked modes.
$\boldsymbol{d}$ ← vector containing indices of the modes.
$\boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\epsilon} \leftarrow \mathcal{E}(\boldsymbol{x})$
$\boldsymbol{r} \leftarrow \boldsymbol{\mu}$
for $j$ in $\boldsymbol{d}$ :
    Initialize $\boldsymbol{E}$
    for $i$ in $\boldsymbol{d}[\sim\boldsymbol{J}]$ :
        $\boldsymbol{I} \leftarrow [i, \boldsymbol{J}]$   # concatenation
        $\hat{\boldsymbol{r}}_i$ ← zero out all the latent variables except members of $\boldsymbol{I}$
        $\tilde{\boldsymbol{x}}_i \leftarrow D(\hat{\boldsymbol{r}}_i)$
        $\boldsymbol{E} \leftarrow E_k(\boldsymbol{x}, \tilde{\boldsymbol{x}}_i)$   # append
    $\boldsymbol{J} \leftarrow \boldsymbol{d}[\sim\boldsymbol{J}][\mathrm{argmax}(\boldsymbol{E})]$   # append
Output: $\boldsymbol{J}$
```



Eivazi et al., Expert Syst. Appl. 202, 117038 (2022)

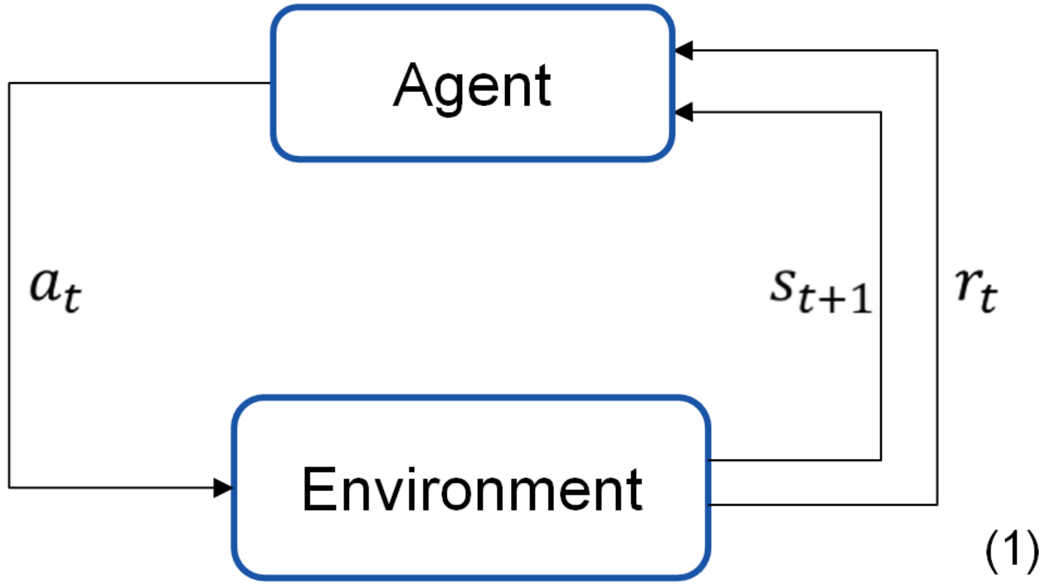
Outline of machine-learning applications to fluid mechanics

- Non-intrusive sensing in a turbulent channel via CNNs.
- Non-intrusive sensing with coarse measurements via GANs.
- Predictions for wall models via CNNs.
- Reduced-order models in turbulence via AEs.
- Flow control via DRL.

Deep reinforcement learning for flow control
Introduction

- Constituting elements:
  - Agent
  - Environment
  - State space $\mathrm{S} = \{s_1, s_2, \cdots, s_N\}$
  - Action space $\mathrm{A} = \{a_1, a_2, \cdots, a_M\}$
  - Transition function $\mathbb{P}(s_{t+1} | s_t, a_t)$
  - Reward function $r_t = R(s_t, a_t, s_{t+1})$
- Goal:

  "Define a policy $\pi(a_t | s_t)$ that maximizes the reward"



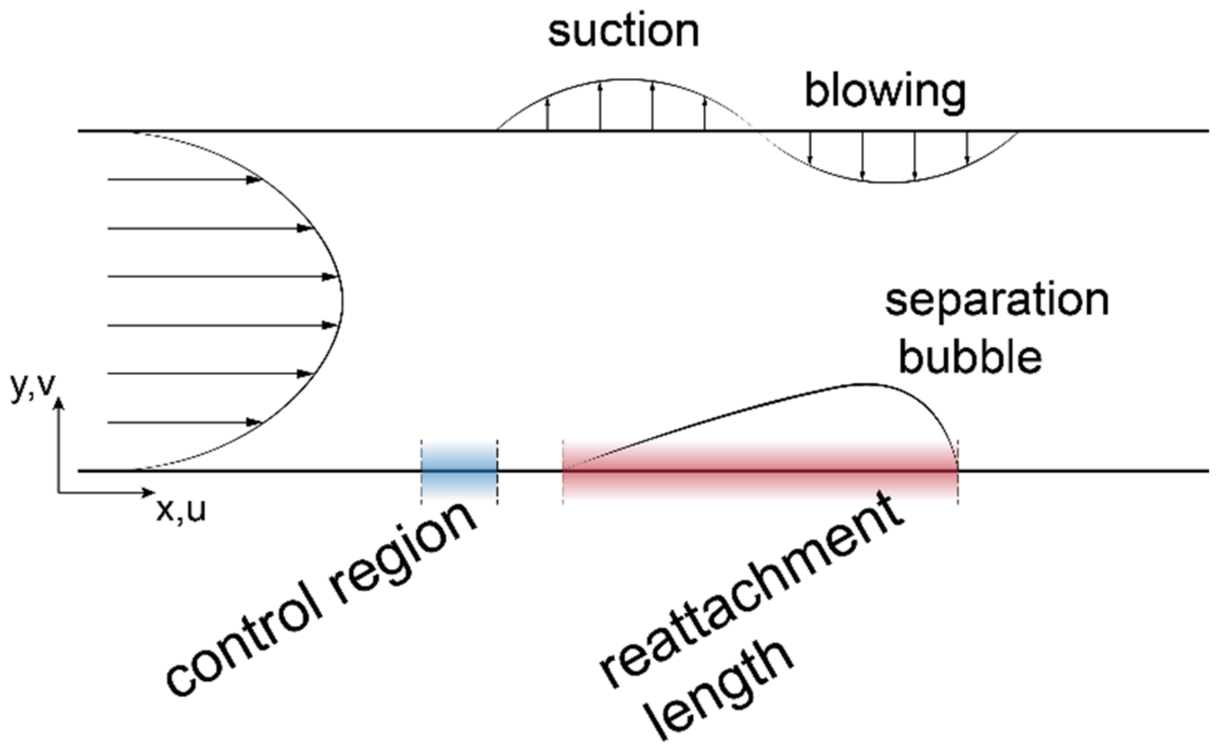
1. *Sutton, R.S. Learning to predict by the methods of temporal differences. Mach Learn* **3,** *9–44 (1988)*

Guastoni et al., APS (2021)

Control of a 2D separation bubble
Case definition

2D channel flow

- Simple environment
- Harmonic forcing can be used as a baseline control
- Computationally cheap



Guastoni et al., APS (2021)

Control of a 2D separation bubble Implementation in SIMSON

Observed state

- 24 velocity probes (streamwise velocity)

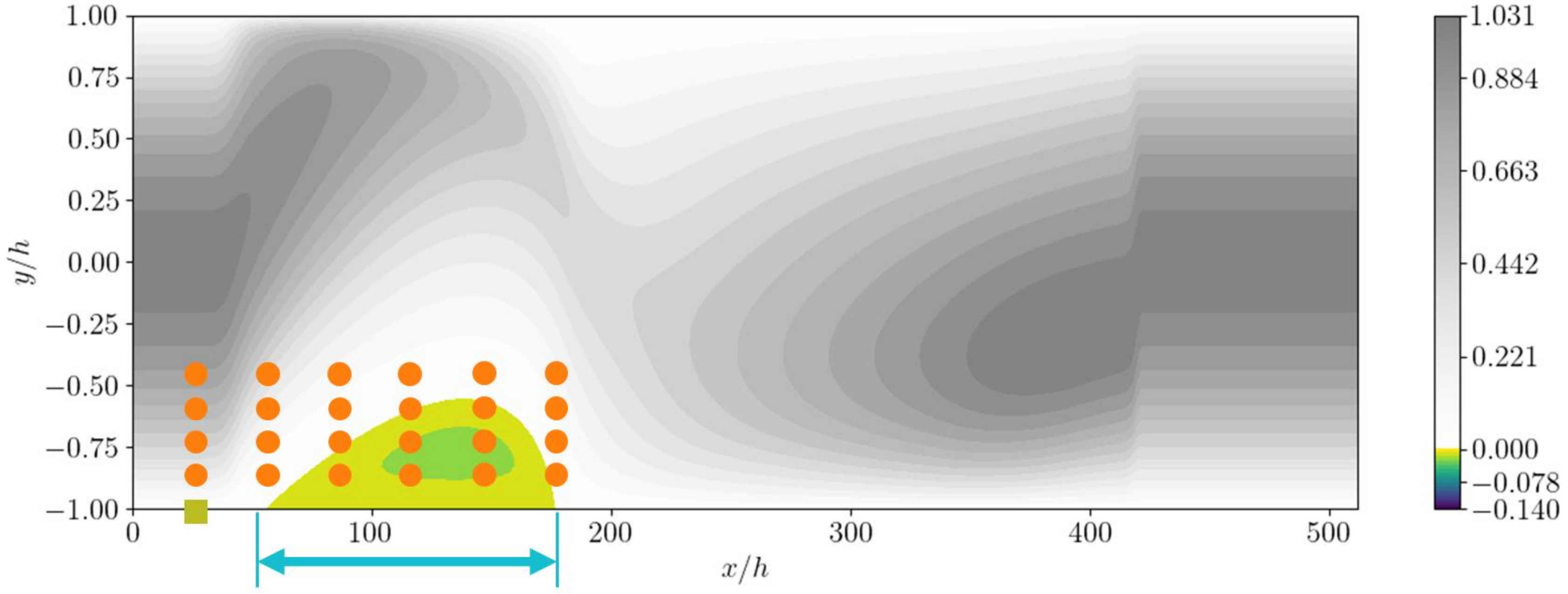
Reward

- Bubble length (lower is better)
- Moving average over time

Control

- Agent learns the parameters of a prescribed forcing function

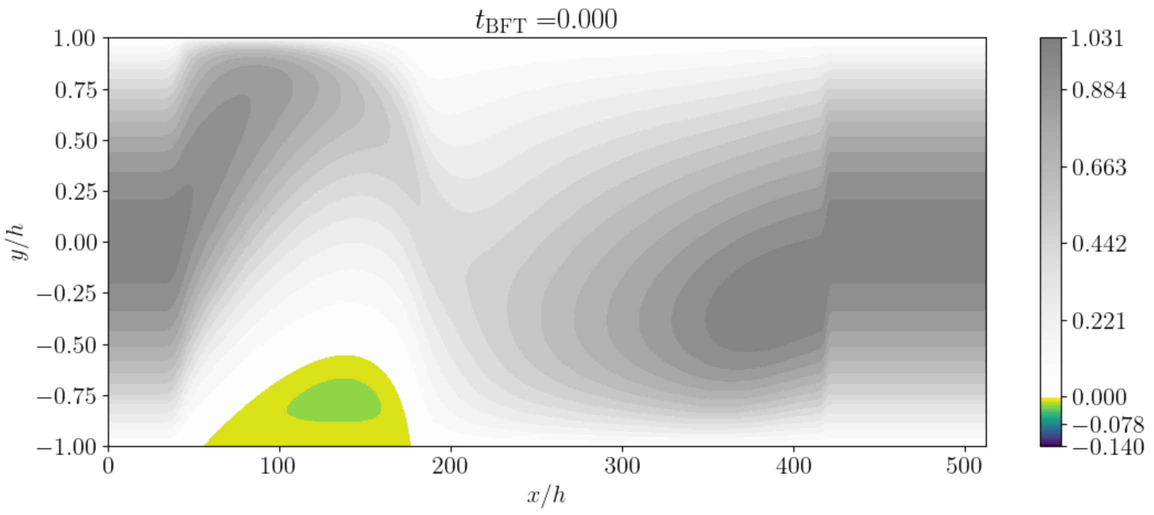
$$\boldsymbol{f}(\boldsymbol{x},t) = [0, a_y(t)]e^{-(y/y_s)^2}e^{-(x/x_s)^2}$$

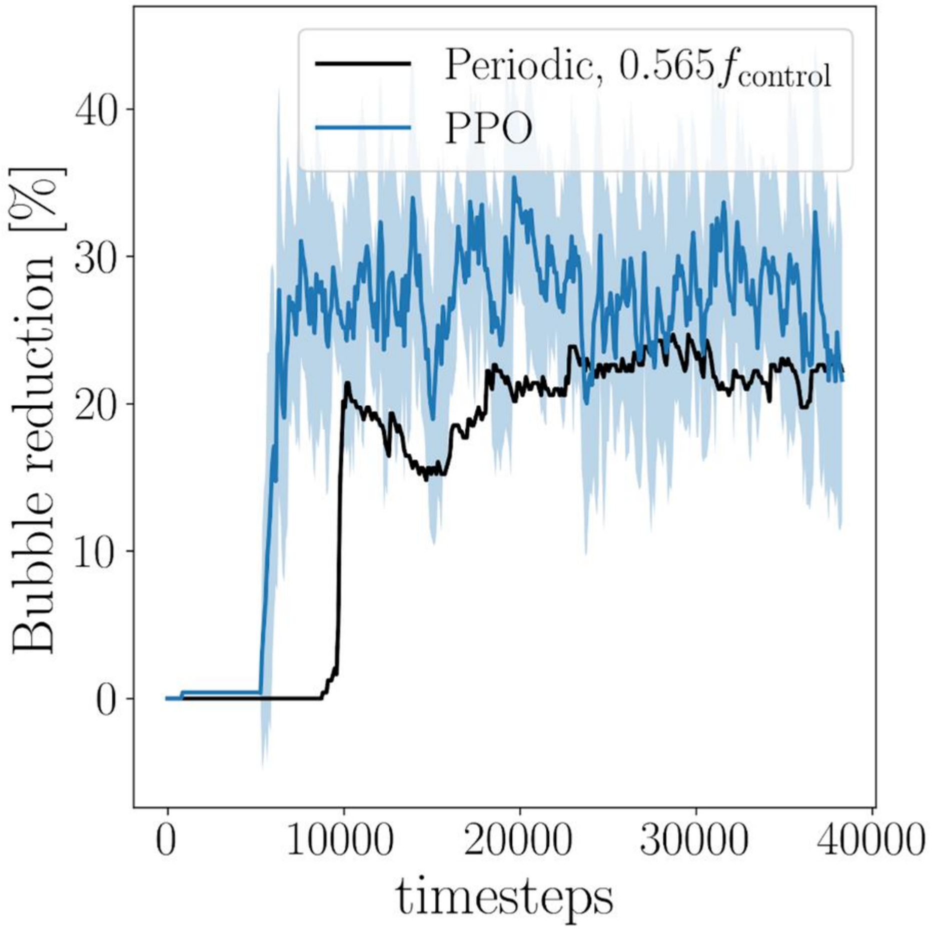


Guastoni et al., APS (2021)

Control of a 2D separation bubble Performance

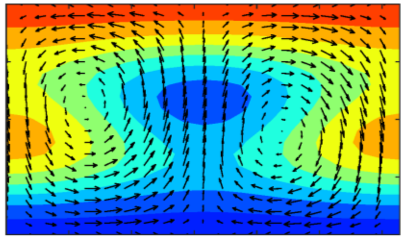
- Larger bubble reduction with **DRL (~30%)** than with **periodic forcing (~20%).**
- Currently extending to **opposition control** in turbulent channel flow.





Guastoni et al., APS (2021)

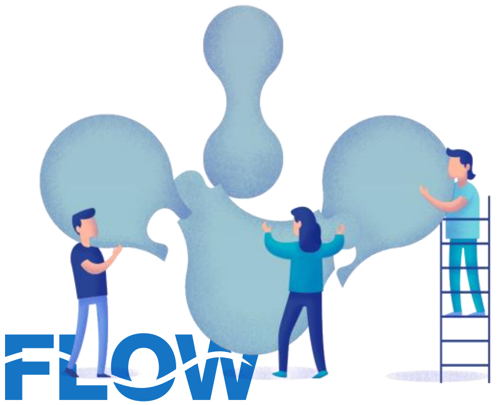
Summary and Conclusions



- CNNs are valuable tools for spatial reconstruction of turbulent fields.
- Excellent predictions at y+=15 (<1% error), outperforming linear reconstruction methods.
- Off-wall boundary conditions and GANs for super-resolution.
- Non-linear modal decompositions using autoencoders.
- Deep-reinforcement-learning-based flow control.

@ricardovinuesa

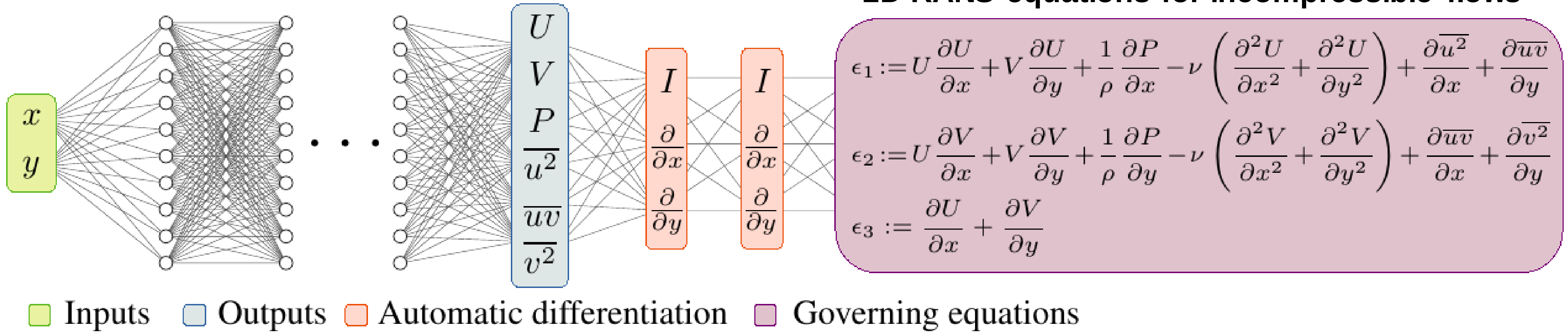
Thank you for your attention!



PINNs application to RANS simulations

Physics-informed neural networks for solving Reynolds-averaged Navier–Stokes equations. Eivazi et al., Phys. Fluids 34, 075117 (2022)

- **2D RANS equations for incompressible flows**



x
y
U
V
P
$\overline{u^2}$
$\overline{uv}$
$\overline{v^2}$
I
$\frac{\partial}{\partial x}$
$\frac{\partial}{\partial y}$
I
$\frac{\partial}{\partial x}$
$\frac{\partial}{\partial y}$
🟩 Inputs
⬜ Outputs
🟧 Automatic differentiation
🟪 Governing equations

$$\epsilon_1 := U\frac{\partial U}{\partial x} + V\frac{\partial U}{\partial y} + \frac{1}{\rho}\frac{\partial P}{\partial x} - \nu\left(\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2}\right) + \frac{\partial \overline{u^2}}{\partial x} + \frac{\partial \overline{uv}}{\partial y}$$

$$\epsilon_2 := U\frac{\partial V}{\partial x} + V\frac{\partial V}{\partial y} + \frac{1}{\rho}\frac{\partial P}{\partial y} - \nu\left(\frac{\partial^2 V}{\partial x^2} + \frac{\partial^2 V}{\partial y^2}\right) + \frac{\partial \overline{uv}}{\partial x} + \frac{\partial \overline{v^2}}{\partial y}$$

$$\epsilon_3 := \frac{\partial U}{\partial x} + \frac{\partial V}{\partial y}$$

DATA + Underdetermined sytem of Eqs. ➡ Solve RANS equations

PINNs application to RANS simulations

Physics-informed neural networks for solving Reynolds-averaged Navier–Stokes equations. Eivazi et al., Phys. Fluids 34, 075117 (2022)

$$L = L_e + L_b,$$

Total loss

$$L_e = \frac{1}{N_e}\sum_{i=1}^{3}\sum_{n=1}^{N_e}|\epsilon_i^n|^2,$$

Residual of the RANS equations (Unsupervised loss)

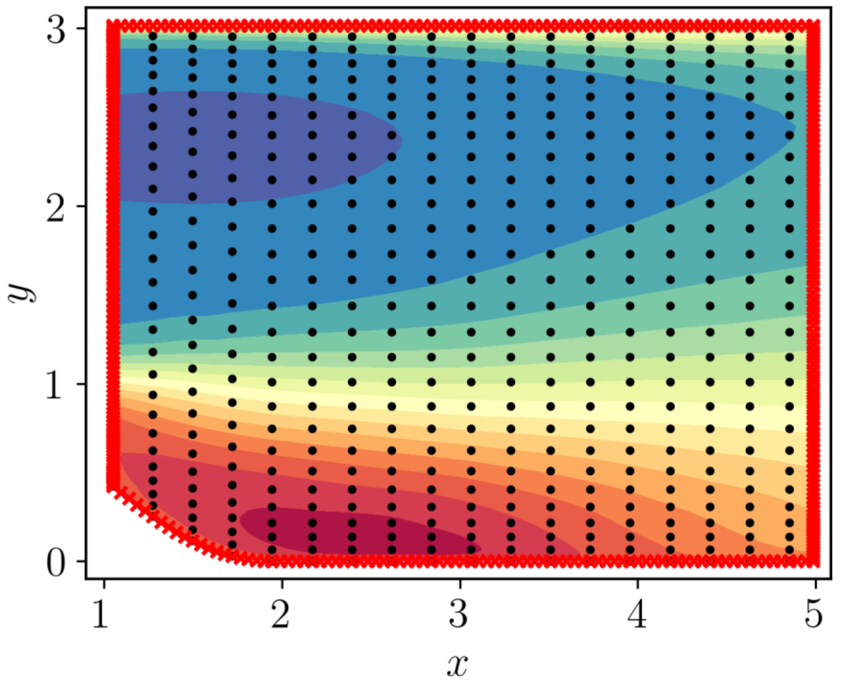
$$L_b = \frac{1}{N_b}\sum_{n=1}^{N_b}|\mathbf{U}_b^n - \tilde{\mathbf{U}}_b^n|^2,$$

Loos for the BCs (Supervised loss)

$$\mathbf{U}_b^n = [U_b^n, V_b^n, P_b^n, \overline{u^2}_b^n, \overline{uv}_b^n, \overline{v^2}_b^n]^\mathsf{T}$$

N_e : Number of collocation points

N_b : Number of points on the domain boundaries



● **Collocation points**

X Boundary points

PINNs application to RANS simulations

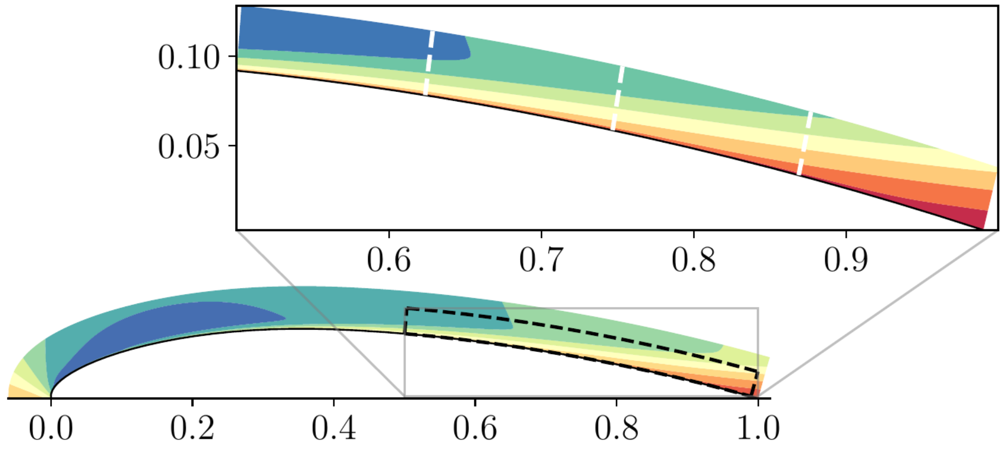
Physics-informed neural networks for solving Reynolds-averaged Navier–Stokes equations. Eivazi et al., Phys. Fluids 34, 075117 (2022)

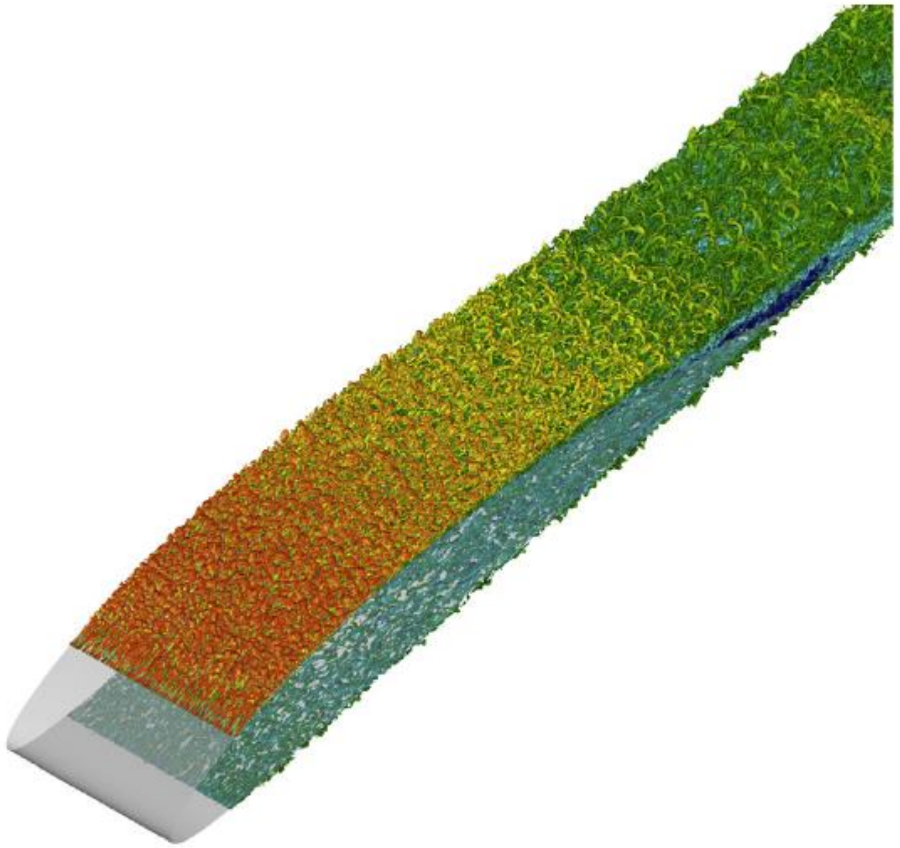
Test case 4: NACA4412 airfoil

DATA from:
Vinuesa et al., Int. J. Heat Fluid Flow 72, 86 (2018).

$$Re_c = U_\infty c/\nu = 200,000$$

$$0.5 < x/c < 1$$

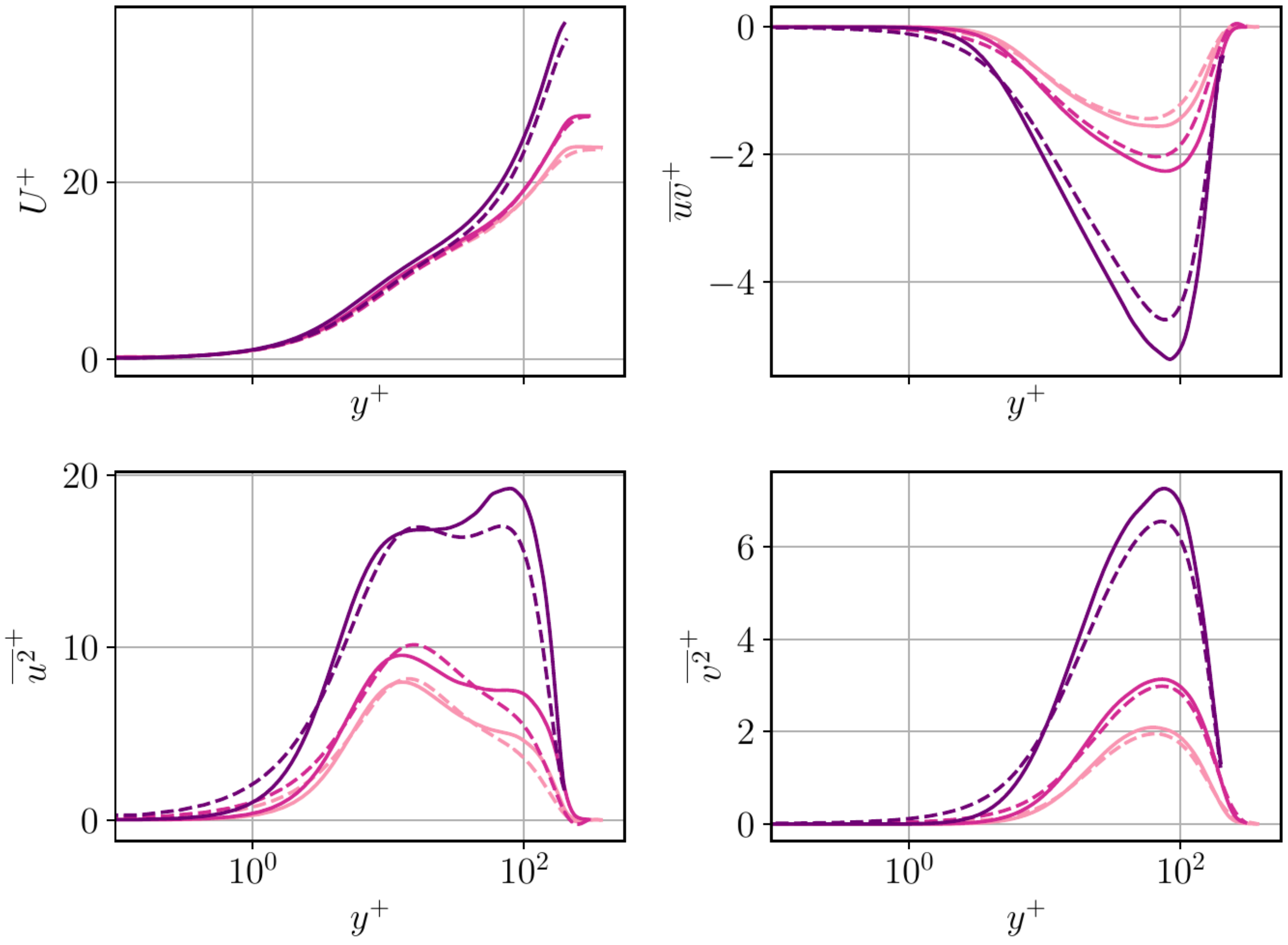




PINNs application to RANS simulations

Physics-informed neural networks for solving Reynolds-averaged Navier–Stokes equations. Eivazi et al., Phys. Fluids 34, 075117 (2022)

Test case 4: NACA4412 airfoil



$$E_U = 1.56\%$$
$$E_V = 2.17\%$$
$$E_P = 7.30\%$$
$$E_{\overline{u^2}} = 9.43\%$$
$$E_{\overline{uv}} = 11.36\%$$
$$E_{\overline{v^2}} = 4.69\%$$

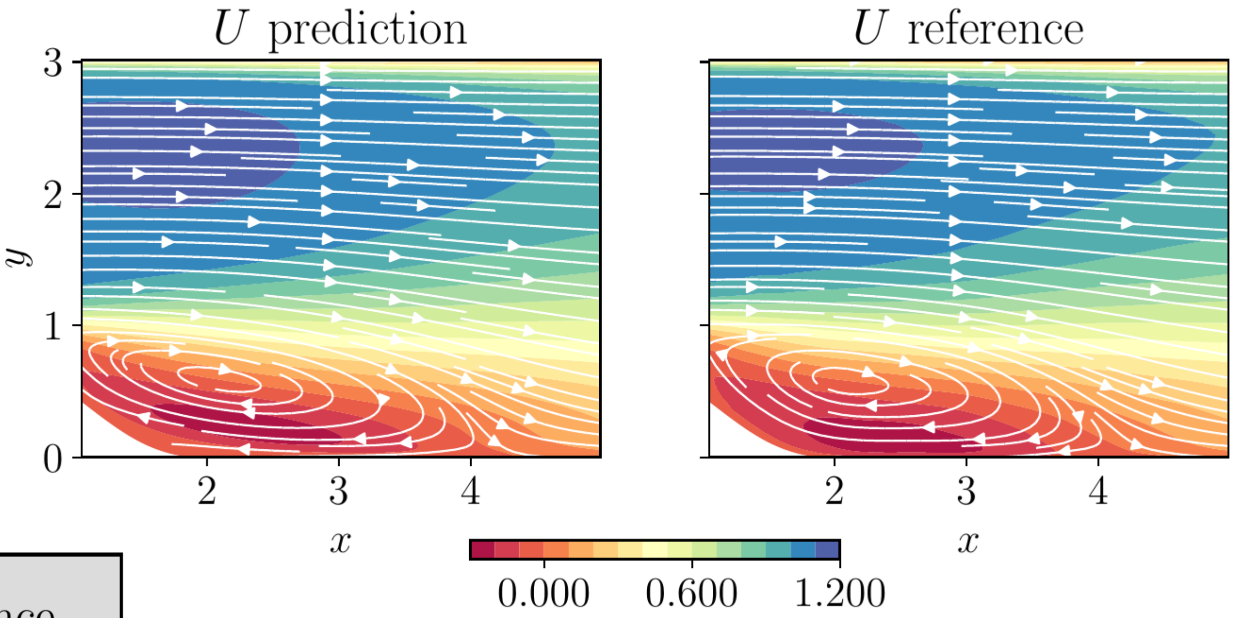
PINNs application to RANS simulations

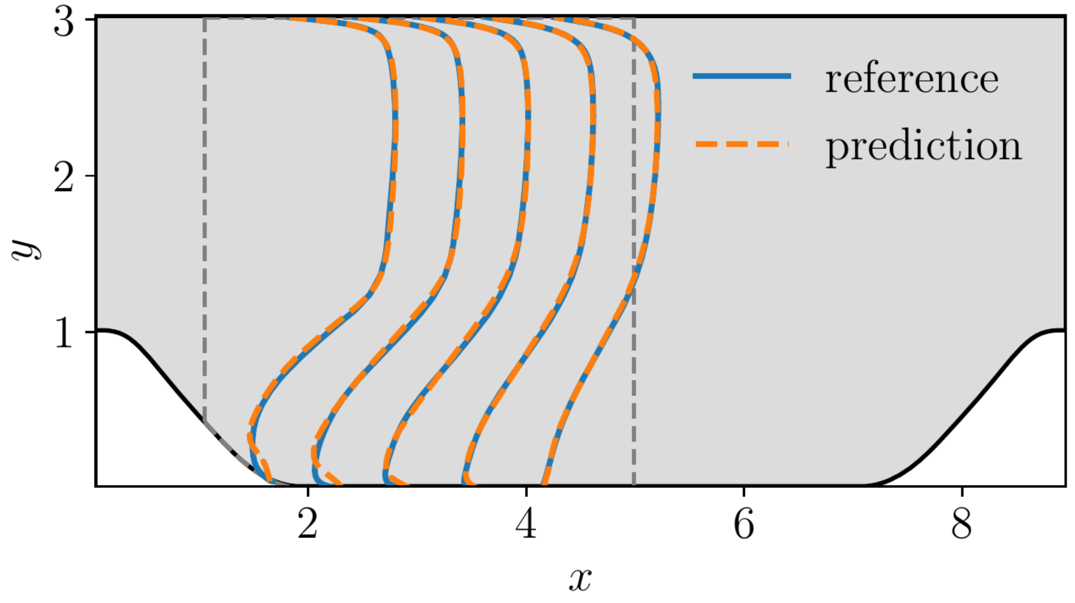
Physics-informed neural networks for solving Reynolds-averaged Navier–Stokes equations. Eivazi et al., Phys. Fluids 34, 075117 (2022)

Test case 5: Periodic hill

$$Re_b = U_b H/\nu = 2,800$$

$$1 < x/H < 5$$





$$E_U = 2.77\%, \ E_V = 19.70\%$$

$$E_P = 8.61\%, \ E_{\overline{u^2}} = 28.18\%$$

$$E_{\overline{uv}} = 16.70\%, \ E_{\overline{v^2}} = 20.24\%$$

CNN-based β-variational autoencoders (CNN-βVAE)

Let us consider:

- $\boldsymbol{x}$ is a data sample in some high-dimensional space $\mathcal{X}$ with the distribution $p(\boldsymbol{x})$.
- $\boldsymbol{r}$ is a vector of latent variables in a low-dimensional space $\mathcal{R}$ with the probability density function (PDF) $p(\boldsymbol{r})$.
- $f(\boldsymbol{r};\boldsymbol{\theta})$ is a family of deterministic functions, parameterized by a vector $\boldsymbol{\theta}$ in some space Θ in such a way that $f : \mathcal{R} \times \Theta \mapsto \mathcal{X}$.

We aim to maximize:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \int p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{r})p(\boldsymbol{r})\mathrm{d}\boldsymbol{r}, \tag{8}$$

where $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ is the so-called marginal likelihood, which is the approximation of $p(\boldsymbol{x})$ with parameters $\boldsymbol{\theta}$.

CNN-based β-variational autoencoders (CNN-βVAE)

VAEs define another probability distribution $q_\phi(\boldsymbol{r}|\boldsymbol{x})$, a so-called probabilistic encoder (or recognition model). In a similar vein $p_\theta(\boldsymbol{x}|\boldsymbol{r})$ is referred to as a probabilistic decoder (or generative model). The marginal likelihood for each $\boldsymbol{x}$ can be defined as:

$$\log p_\theta(\boldsymbol{x}) = D_{\mathrm{KL}}(q_\phi(\boldsymbol{r}|\boldsymbol{x})||p_\theta(\boldsymbol{r}|\boldsymbol{x})) + \mathcal{C}(\boldsymbol{\theta}, \phi; \boldsymbol{x}), \tag{9}$$

The KL-divergence is non-negative, which indicates that the second RHS term $\mathcal{C}(\boldsymbol{\theta}, \phi; \boldsymbol{x})$ is a lower bound on the marginal likelihood, and can be written as:

$$\log p_\theta(\boldsymbol{x}) \geq \mathcal{C}(\boldsymbol{\theta}, \phi; \boldsymbol{x}) = -D_{\mathrm{KL}}(q_\phi(\boldsymbol{r}|\boldsymbol{x})||p_\theta(\boldsymbol{r})) + \mathbb{E}_{q_\phi(\boldsymbol{r}|\boldsymbol{x})}\left[\log p_\theta(\boldsymbol{x}|\boldsymbol{r})\right]. \tag{10}$$

This term is usually called evidence lower bound (ELBO).

CNN-based β-variational autoencoders (CNN-βVAE)

We assume $q_\phi(\boldsymbol{r}|\boldsymbol{x})$ to be a Gaussian distribution,

$$\log q_{\phi(\boldsymbol{r}|\boldsymbol{x})} = \log \mathcal{N}(\boldsymbol{r}; \boldsymbol{\mu}, \boldsymbol{\sigma}^2 \mathbf{I}), \tag{11}$$

where the mean $\boldsymbol{\mu}$ and the standard deviation $\boldsymbol{\sigma}$ are outputs of the encoder, and $\mathbf{I}$ is the identity matrix. We sample from $q_\phi(\boldsymbol{r}|\boldsymbol{x})$ using $\boldsymbol{r} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \varepsilon$ where $\varepsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is an auxiliary normally-distributed random number.

β-VAE loss function:

$$\mathcal{L}(\boldsymbol{\theta}, \phi; \boldsymbol{x}) = \boldsymbol{\mathcal{L}}_{\mathrm{rec}} - \frac{\beta}{2} \sum_{i=1}^{d} (1 + \log(\sigma_i^2) - \mu_i^2 - \sigma_i^2) \tag{12}$$

Eivazi et al., Expert Syst. Appl. 202, 117038 (2022)