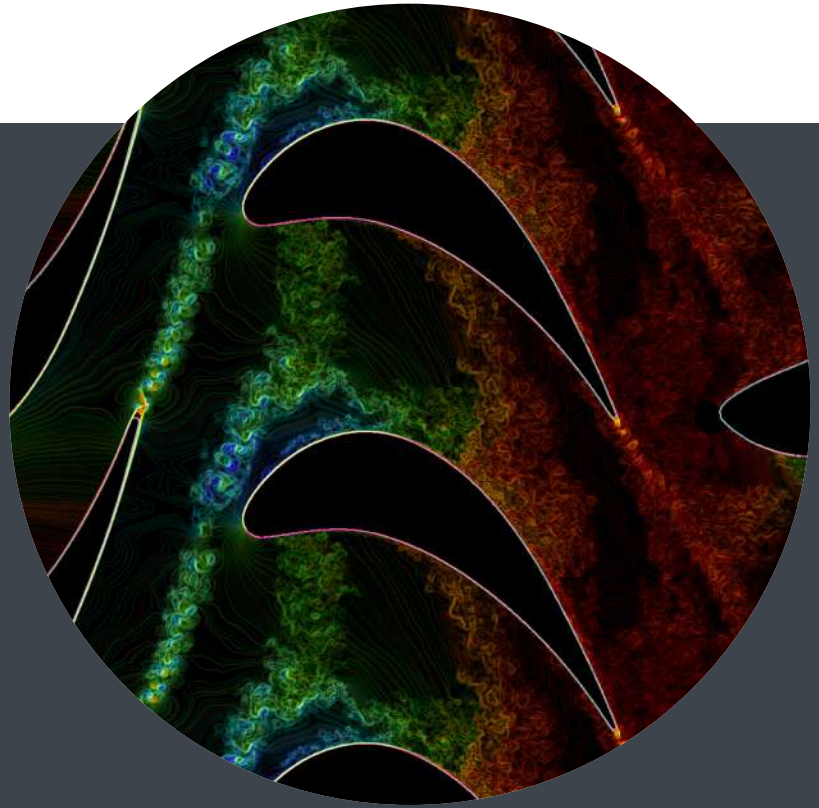
University of Stuttgart
Institute of Aerodynamics and Gas Dynamics

NRG

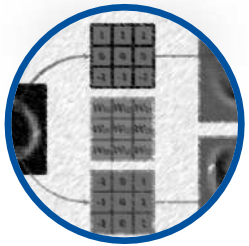


Data-Driven, Discretization Consistent LES models

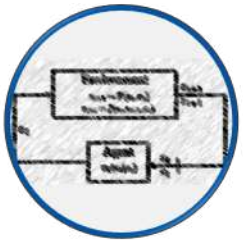
Andrea Beck
Institute of Aerodynamics and Gas Dynamics (IAG) &
Stuttgart Center for Simulation Science (SC SimTech)

Overview

Part 1: Multi-X problems and PDE solvers



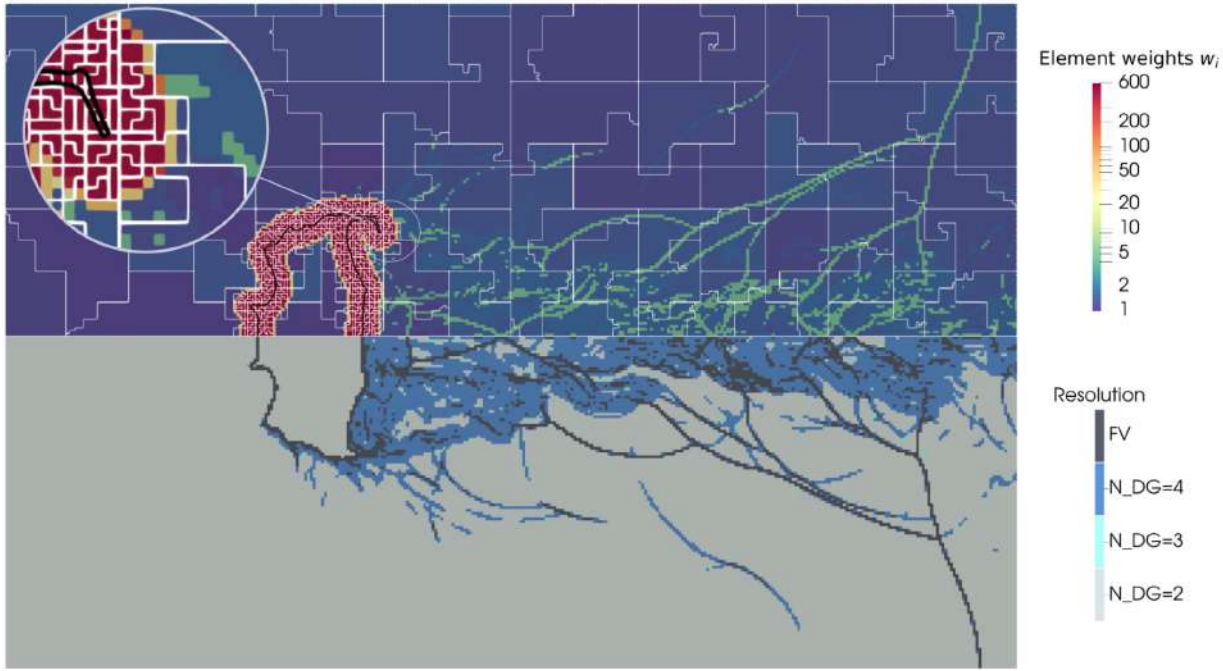
Part 2: Supervised learning for PDE submodels



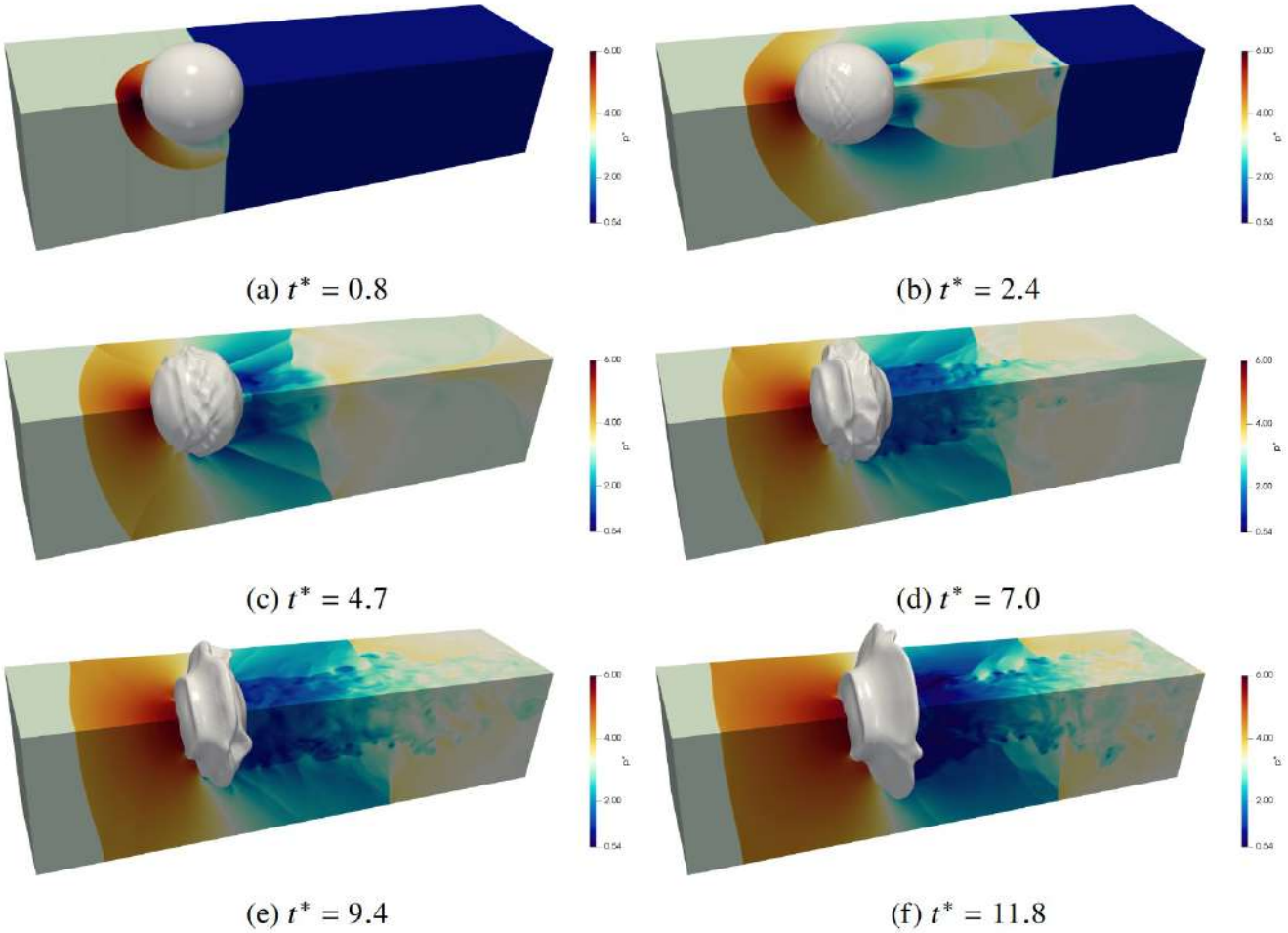
Part 3: From data-driven to integrated CFD/ML

Multiphase, Multiscale, Multimethod: Shock Droplet Interaction

- Sharp-Interface simulation of 3D water droplet – shock interaction at Ma = 2.4 and We = 100
- About 140 DOF / droplet diameter
- hp-adaptive DGSEM / FV scheme with DLB
- 7200 CPUh on HAWK



Mossier et al., An Efficient hp-Adaptive Strategy for a Level-Set Ghost Fluid Method, arxiv, 2023
Jöns et al., Riemann solvers for phase transition in a compressible sharp-interface method, JCP, 2023
Mossier et al., A p-Adaptive Discontinuous Galerkin Method with hp-Shock Capturing. Journal of Scientific Computing, 2022



(a) $t^* = 0.8$

(b) $t^* = 2.4$

(c) $t^* = 4.7$

(d) $t^* = 7.0$

(e) $t^* = 9.4$

(f) $t^* = 11.8$

Multiphase, Multiscale, Multimethod: Icing on Airfoils

EIROS project via http://www.eirosproject.com/project/case-study-applications/

Stefano Fornasier via http://tesi.cab.unipd.it/50432/1/STEFANO_FORNASIER_1147354_assignsubmission_file_Fornasier_Stefano_1056548.pdf

Chicago Trib

Tuesday, November 1, 1994

Chicago-bound plane crashes; 68 are killed

By Flynn McRoberts and George Papajohn
Tribune Staff Writers

The blustery weather that roiled Chicago and Northwest Indiana on Monday will be examined closely as federal safety investigators try Tuesday to determine what caused an American Eagle turboprop to plummet from the sky, killing all 68 people aboard.

Flight 4184, a twin-engine Super ATR-72, was bound from Indianapolis to O'Hare International Airport at about 4 p.m. Monday when it plunged into a farm field about 15 miles southeast of Gary near Interstate Highway 65 and Indiana Highway 110 near Rosebush, Ind.

Twenty-one of the passengers were bound for Chicago and the other 43 passengers were connecting to other flights at O'Hare, airline officials said.

The tragedy followed by less than two months the 110 deaths in the crash of a USAir jet bound from Chicago to Pittsburgh, an accident that remains under investigation.

"It looked like there was no wing on the left-hand side" of the American Eagle plane, said Larry Midkiff, a 39-year-old construction worker who was driving home to Rensselaer when he saw the plane in a steep descent. "I'm not sure because it might have been the angle of the aircraft."

Midkiff said he saw smoke streaming from the plane.

"It was coming down real steep, real hard, nose down," he said. "It wasn't coming down like it was trying to make an emergency landing. ... It kind of dipped down behind some trees. Then it made a big puff of black smoke."

Farmer Norman Prokosky was tinkering with a grain

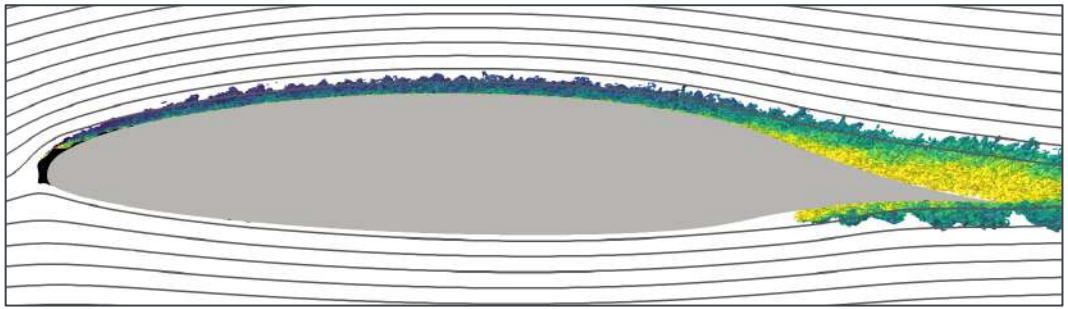
Sun Classic, Page 31

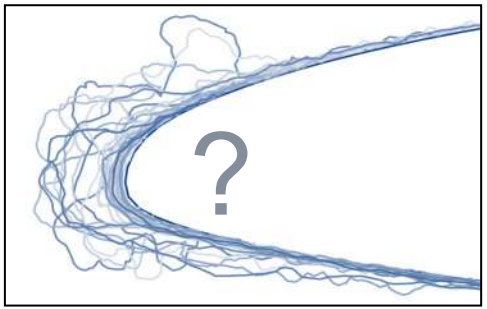
Disaster in stormy Indiana skies

■ American Eagle airline has a good safety record. Page 21.

■ List of major U.S. fatal airplane crashes since 1963. Page 21.

Chicago Tribune via https://www.newspapers.com/clip/14788830/american-eagle-flight-4184-crashes-in-in/

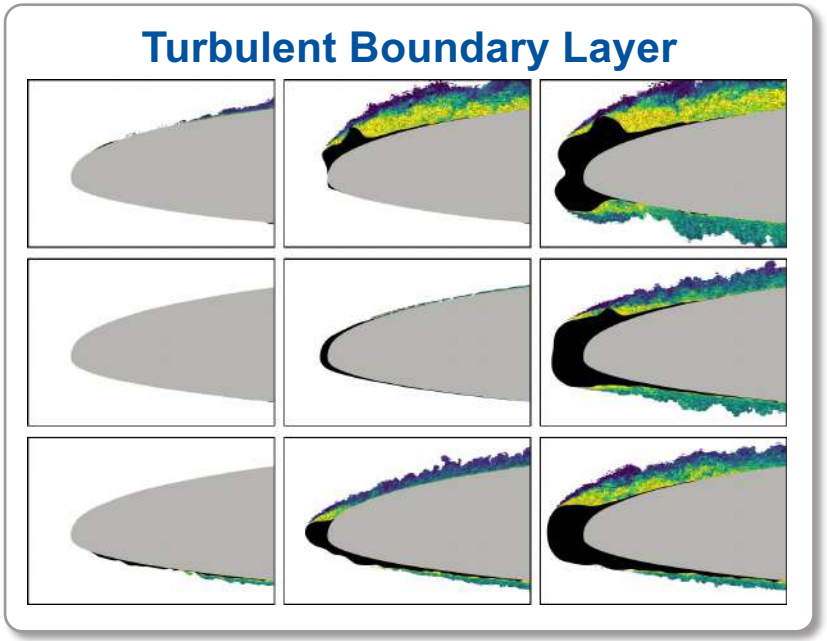


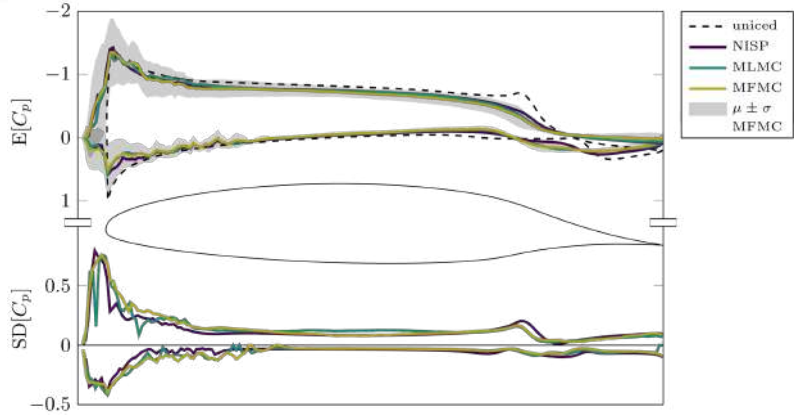


NRG

Multiphase, Multiscale, Multimethod

- Uncertainty Quantification Analysis of Icing on Airfoil Performance
- Parameterization of iced shapes via POD / PCA
- Automated grid generation for wall-resolved LES on hexa-only grids
- Non-intrusive Spectral Projection, MLMC and MFMC of 3D LES computations
- PoUnce-Framework for HPC scheduling and management of UQ workflow

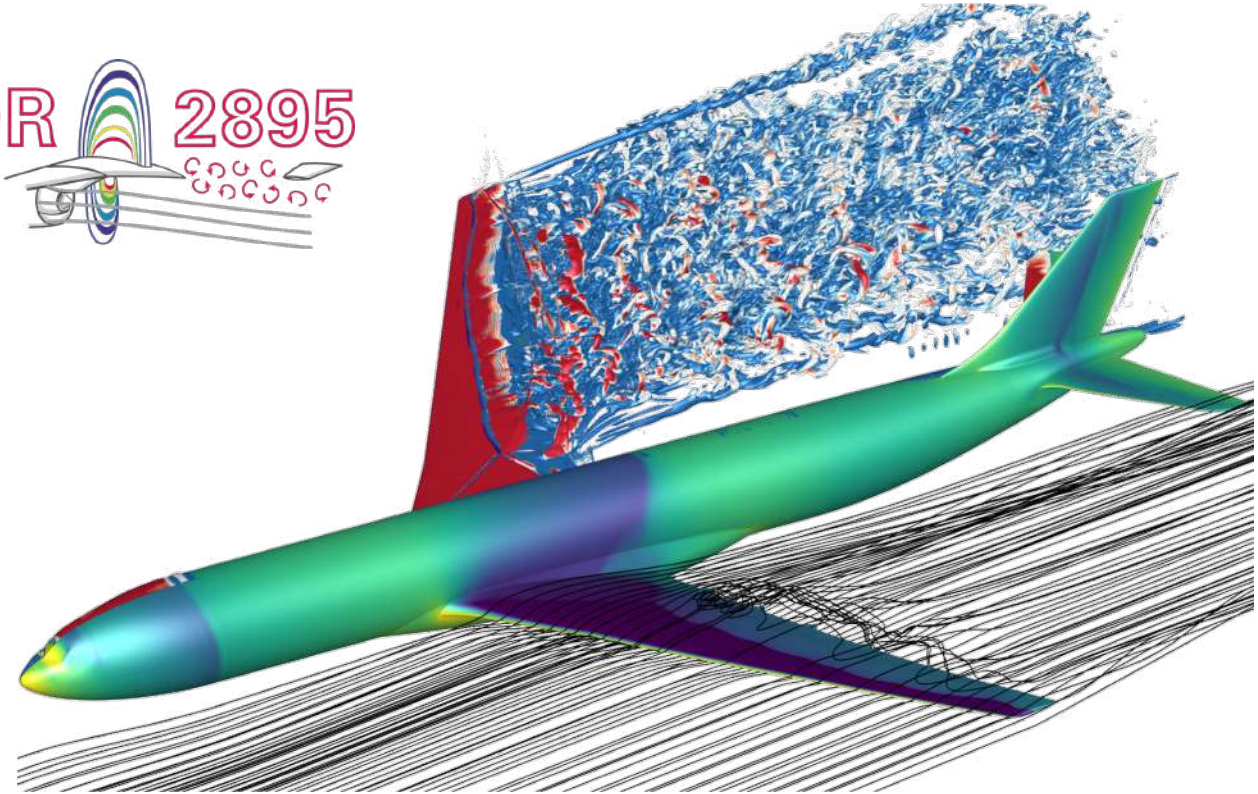


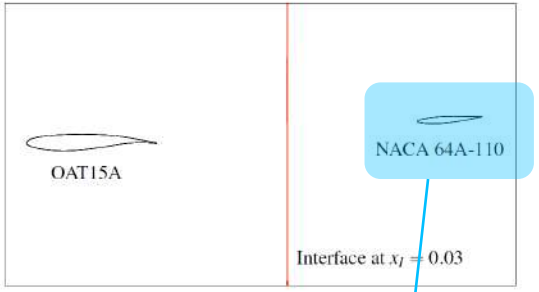


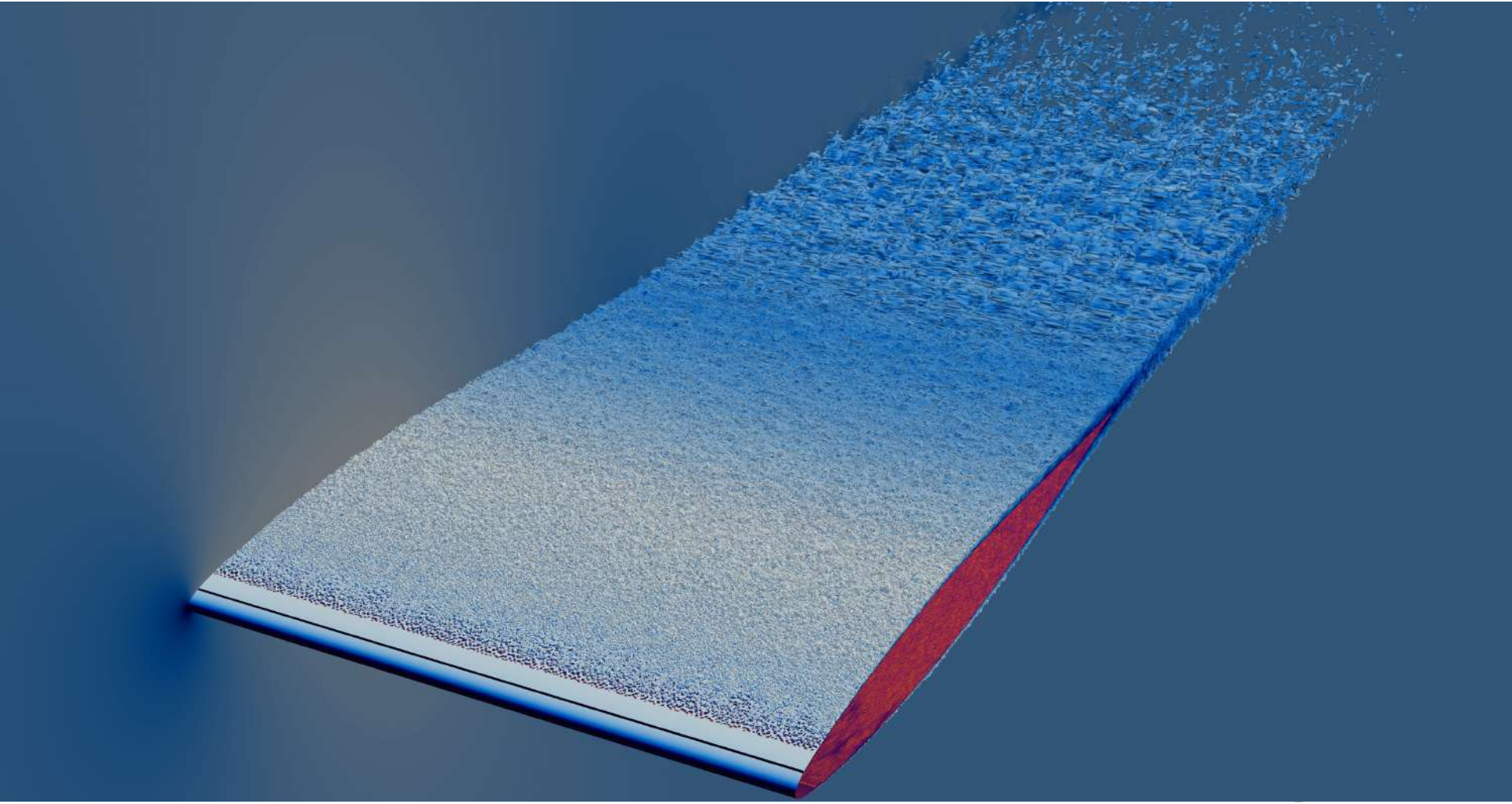
Baden-Württemberg Stiftung

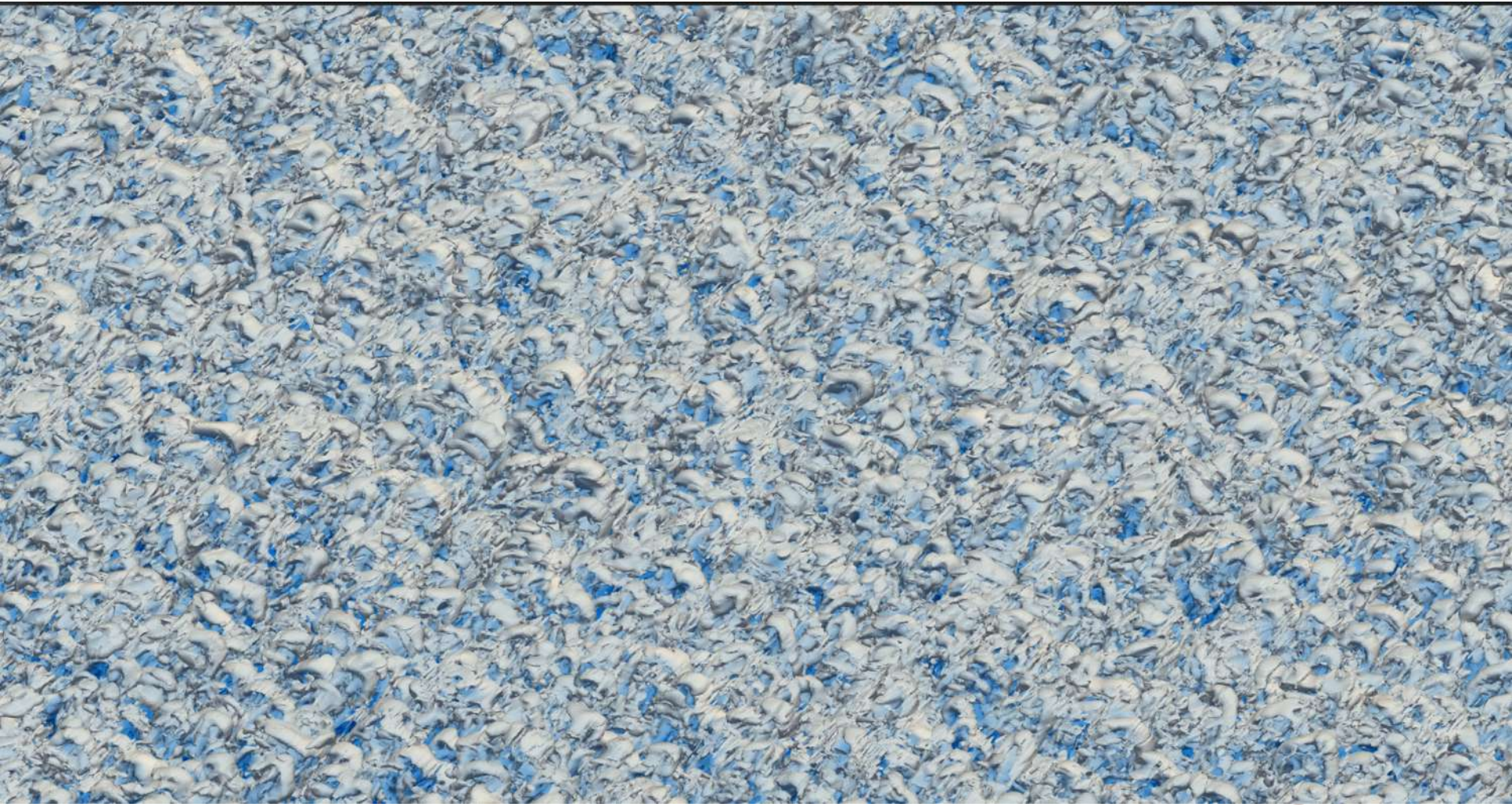
Friedrich und Elisabeth BOYSEN Stiftung für Forschung und Innovation

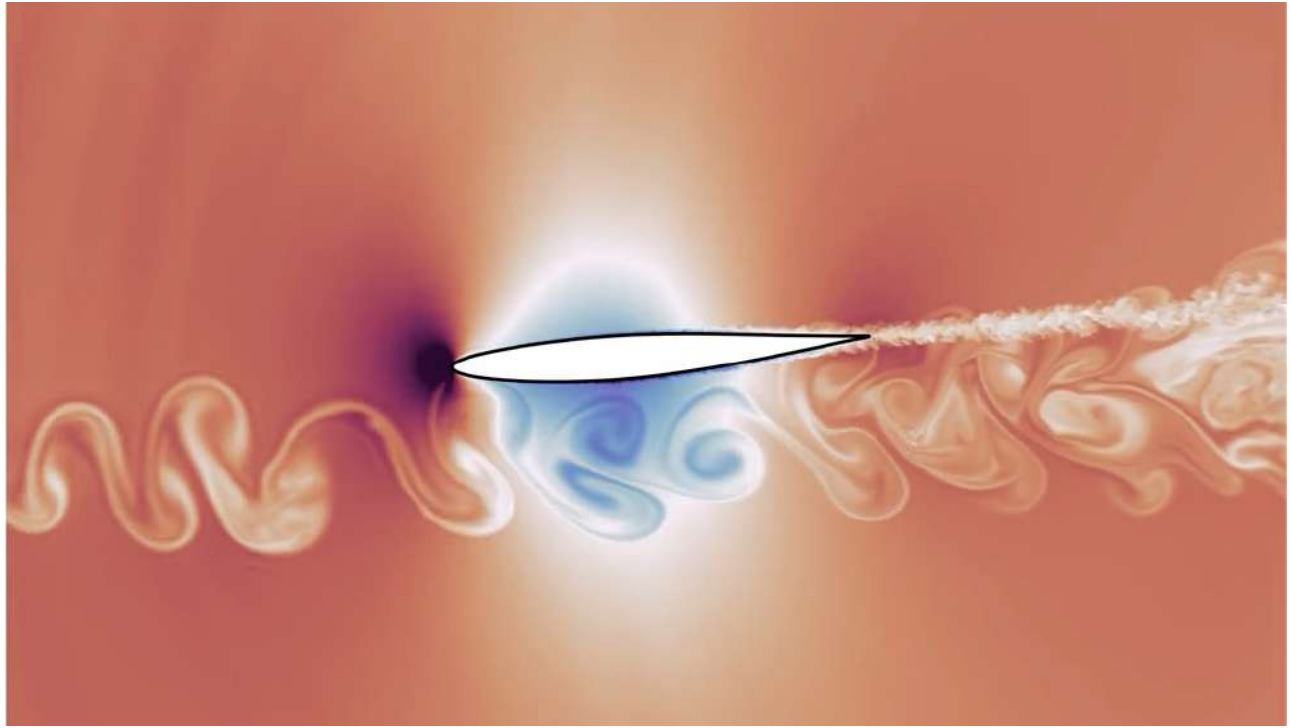
Multiphase, Multiscale, Multimethod: Wake / HTP interaction under buffet



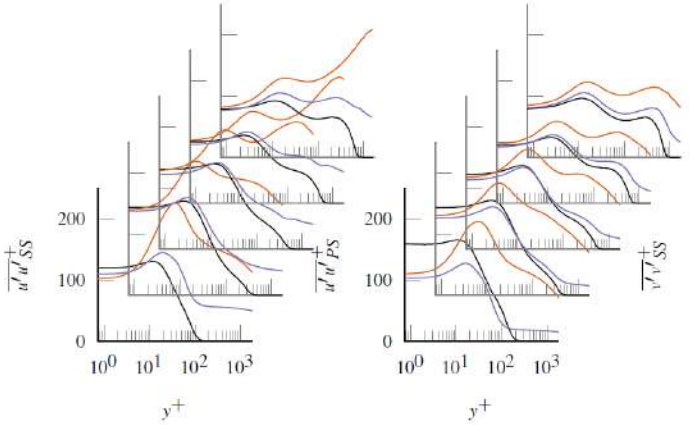

OAT15A
NACA 64A-110
Interface at $x_I = 0.03$



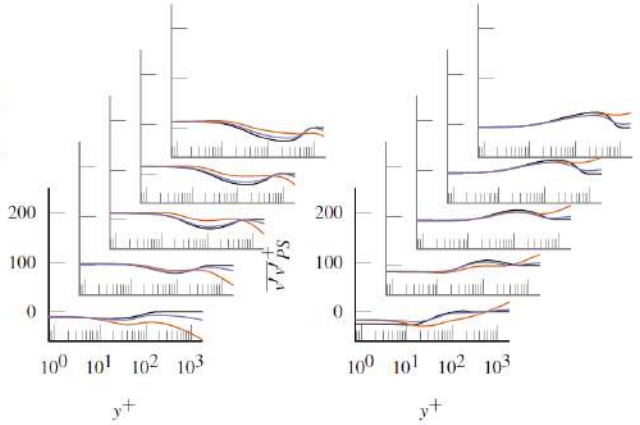




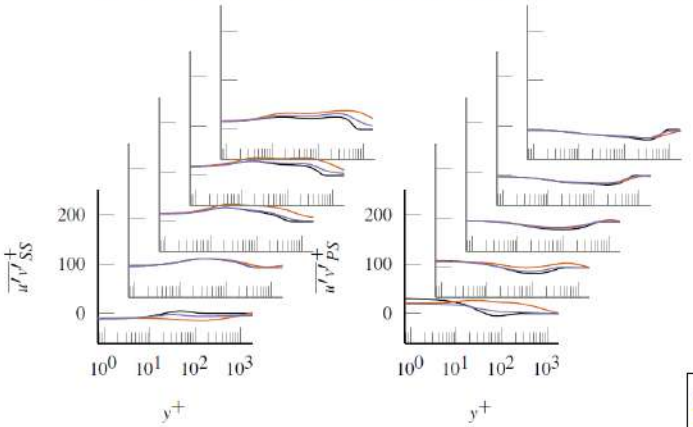
- Approx. 1 Billion DOF, N=7, 70 CTU



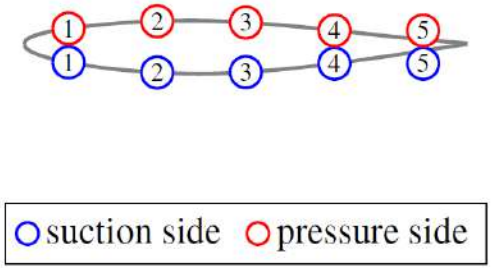
(c) Streamwise mean velocity fluctuations



(d) Wall-normal mean velocity fluctuations.



(e) $\overline{u'v'}^+$ velocity fluctuation.



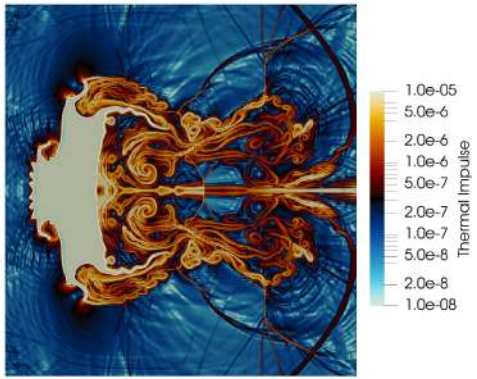
steady — unsteady: Phase 1 — unsteady: Phase 2

What we need to solve these problems – without proof

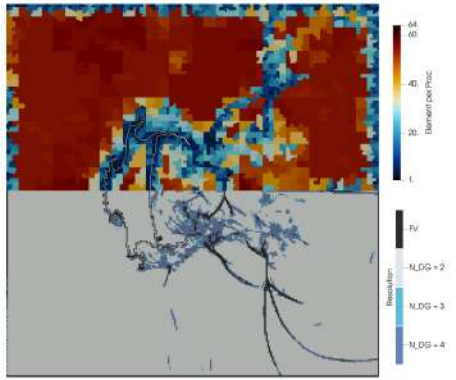
- High order accurate schemes for sufficiently smooth regions: 2nd order FV will not do
- Monotone schemes for shocks, interfaces, gradients
- h/p/r/x-adaptivity to the underlying physics: A flexible, intelligent, physics-compatible numerical scheme

FLEXI: A high order discontinuous Galerkin framework for hyperbolic–parabolic conservation laws

https://github.com/flexi-framework/flexi



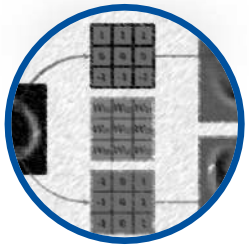
(a) Thermal impulse $|\rho \boldsymbol{j}|$



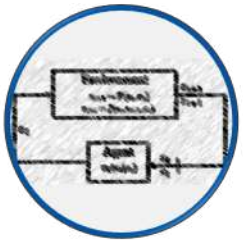
(b) Adaptive discretization

Overview

Part 1: Multi-X problems and PDE solvers



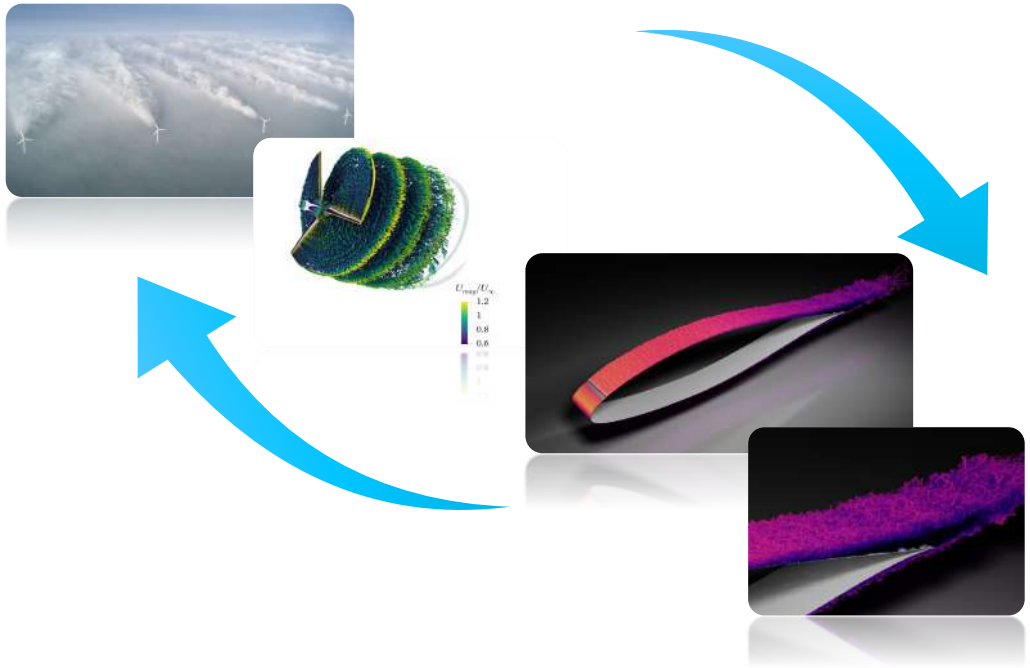
Part 2: Supervised learning for PDE submodels



Part 3: From data-driven to integrated CFD/ML

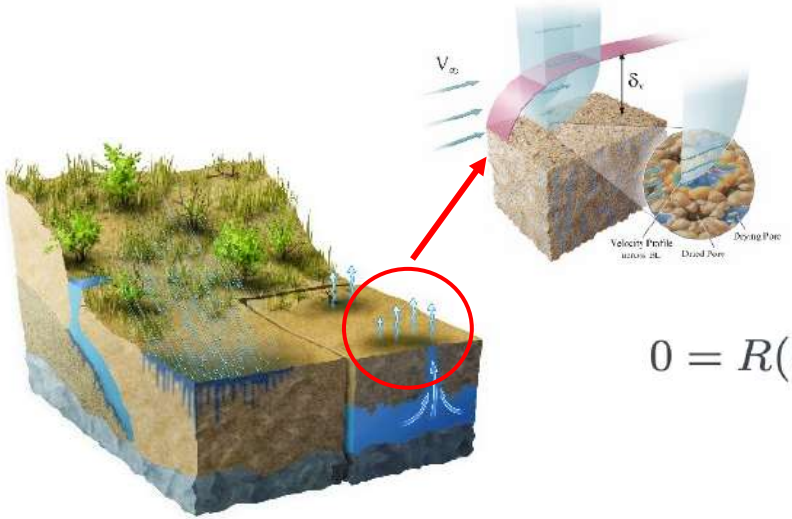
The Scale Gap: Where and why do we need ML?

- Multiscale-Multiphysics systems can cover a wide range of scales



The Scale Gap

- Multiscale problems: Formulations on micro, meso, macro, … scales
- Definition of a coarsening operator (1) → (2) possibly ambiguous
- Solving coarse-scale equations plus Models for the Closure terms $M(U_h)$



$$\underbrace{R(U_l, x_l) = 0}_{\text{Governing Equations at level l}} \quad (1)$$

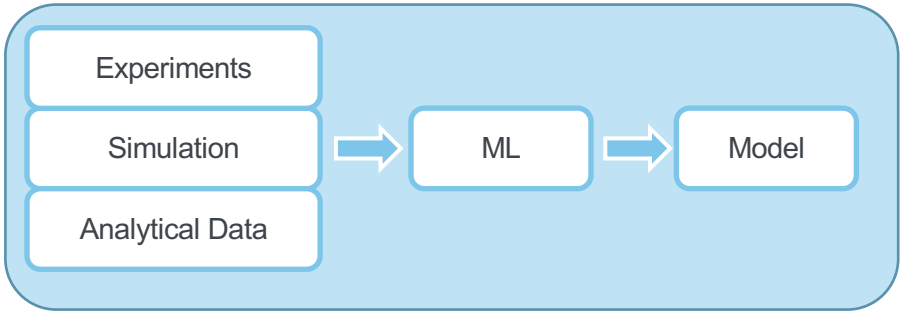
$$0 = R(U_h, x_h) + C(U_l, U_h) \approx \underbrace{R(U_h, x_h,) + \mathbf{M(U_h)}}_{\text{"Closed" Equations at level h}} \quad (2)$$

Machine Learning for data-driven submodels

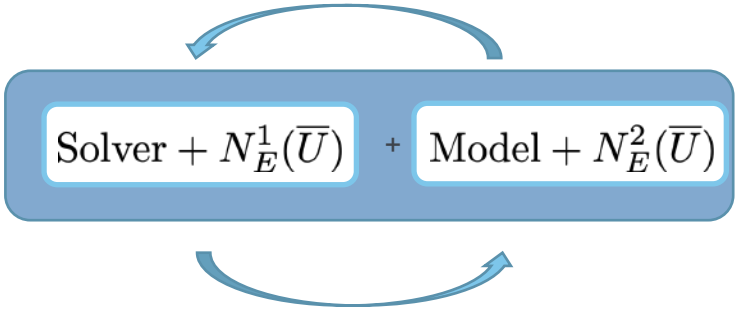
$$R\left(U_L, x_L, t_L\right) = -M\left(U_l, U_L\right)$$

$$R\left(U_L, x_L, t_L\right) \approx -\tilde{M}\left(U_L\right)$$

- Riemann solver
- Forchheimer diffusion coefficient
- RANS closure coefficients
- …

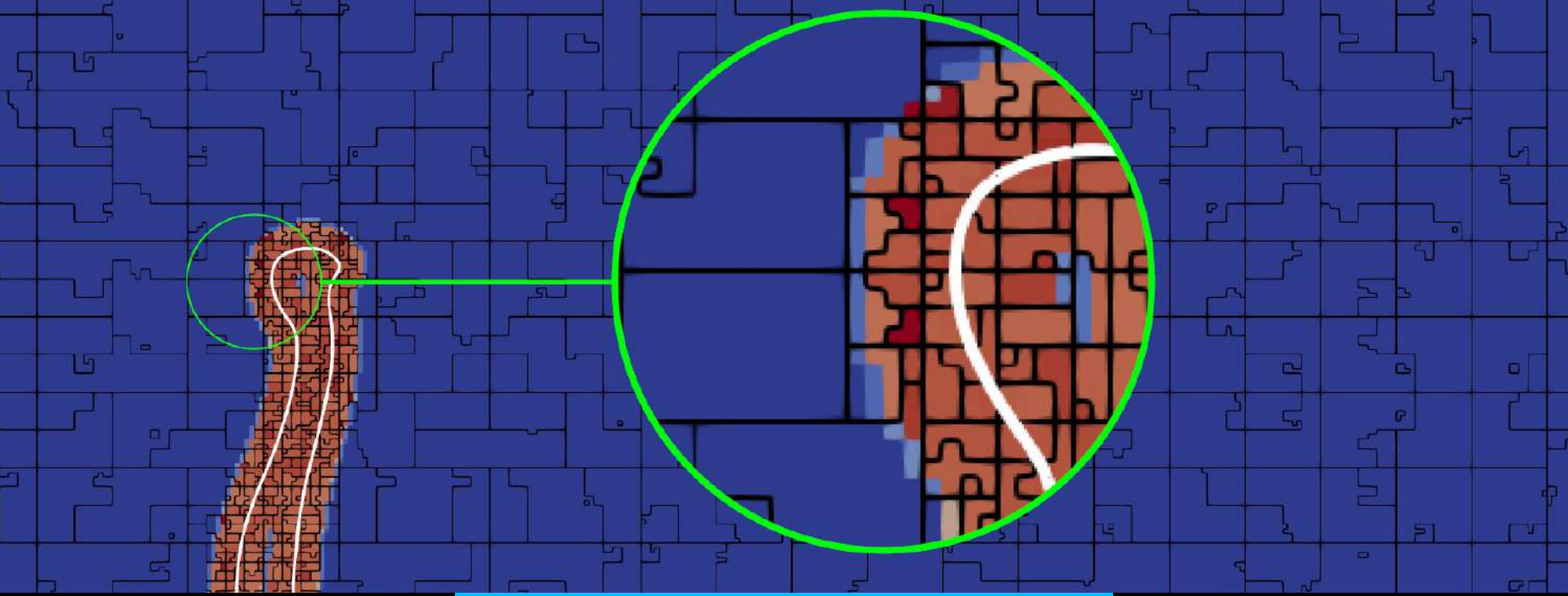


Stage 1: Offline training on well-defined data



Stage 2: Online inference on “real” data

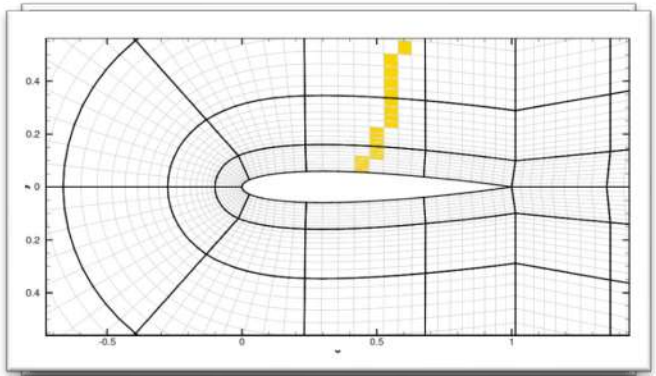
Small detour ahead: Our SL success stories



Data-Driven Local Shock Capturing for DG

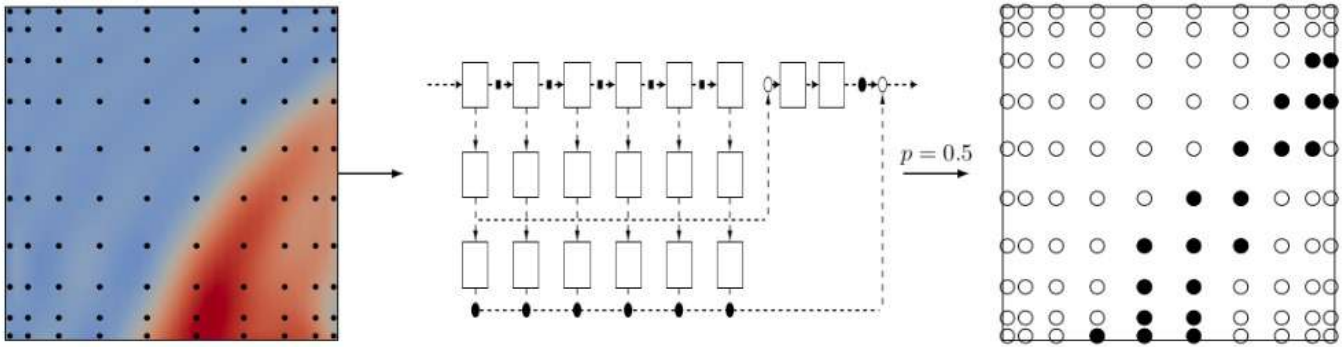
Data-informed High Order Methods

- Discontinuities conflict with high order polynomials
- Stable numerical approximation through Shock Capturing: improves stability, decreases accuracy: use sparingly!
- First step: Detecting the occurrence of shocks: non-trivial, empiricism, many parameters
- For HO methods: Just detecting a “troubled cell” is not good enough: We need localization on the element subscale $h/(N+1)$



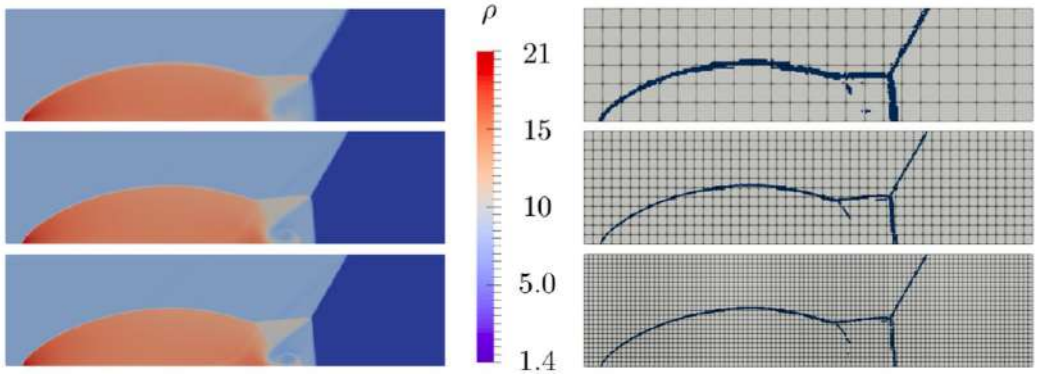
Shock Locator

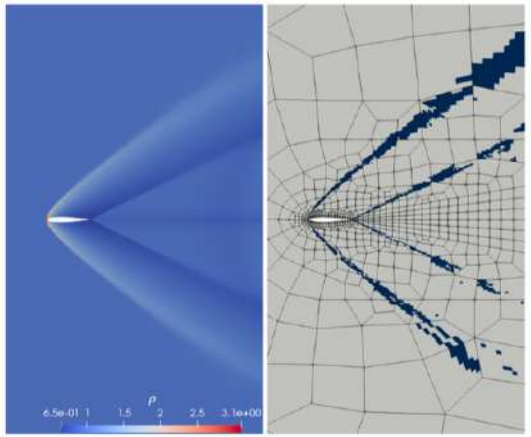
- Locate shock front within high order element on the nodal level
- Supervised learning strategy on analytically labeled data
- Multiscale CNN networks, adapted from edge detection



Shock Locator

- Invariance w.r.t to grids, cases, etc: no retraining, no parameters
- Consistent, continuous shock front on varying resolutions: unstructured grids
- Localized, polynomial artificical viscosity with inner element shock capturing





Localized AV for shock capturing

- Shu-Osher Shock Interaction

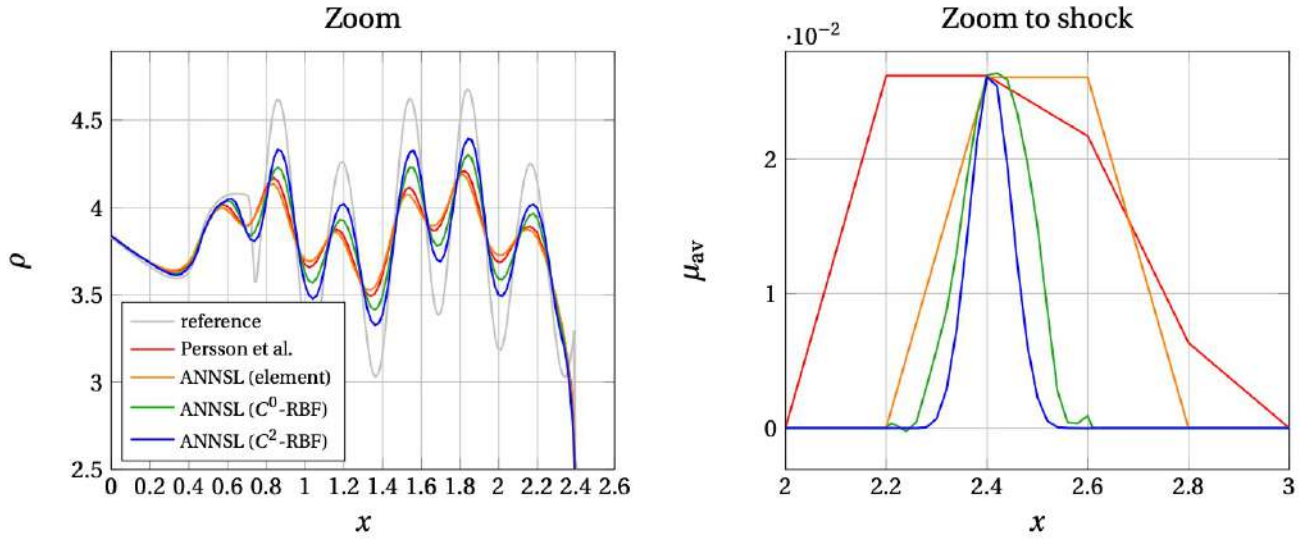


Figure 6: Density solution (left) and artificial viscosity (right) for Shu-Osher problem with $N = 9$ and $n_{\text{elems}} = 50$ at $t = 1.8$. The background grid in x-direction indicates the mesh spacing ($\Delta x = 0.2$).

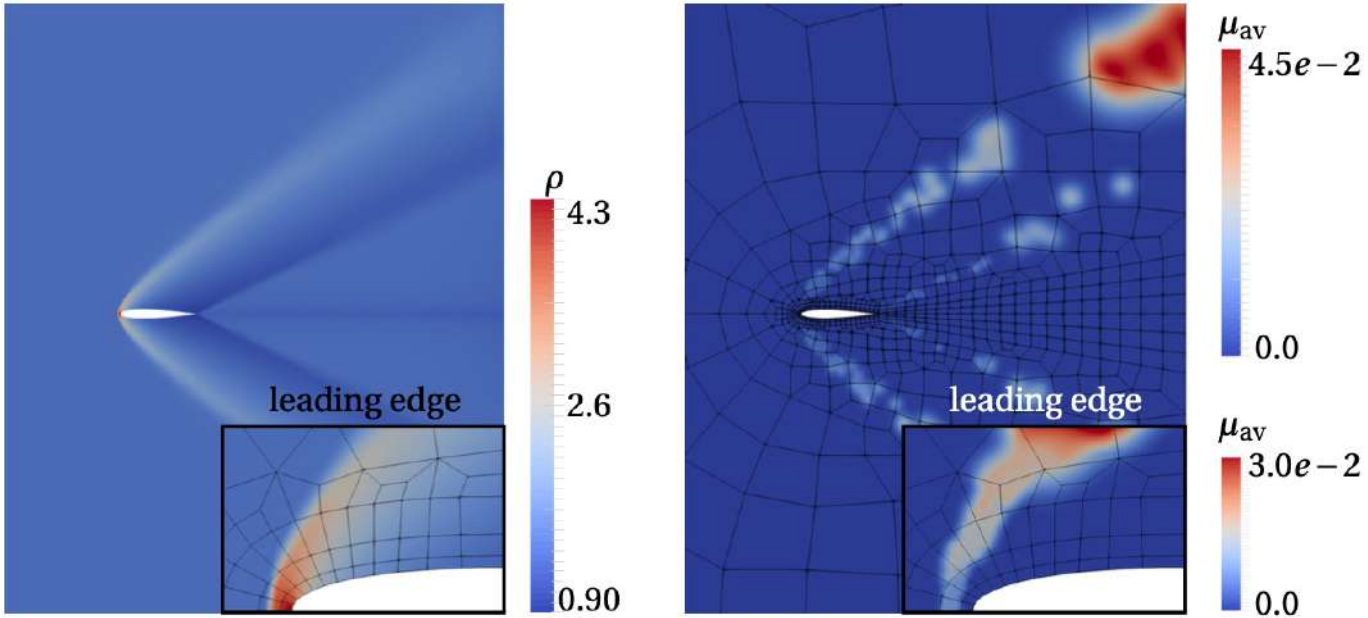
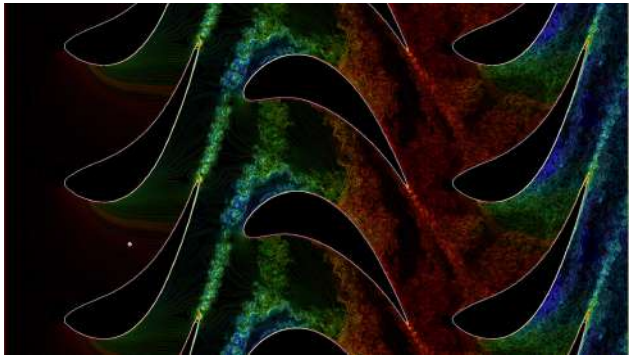


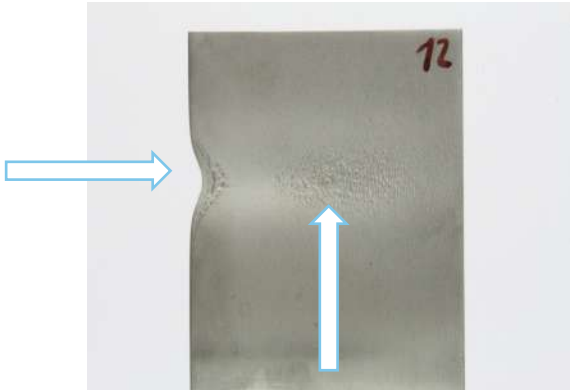
Figure 14: Density (left) and artificial viscosity with underlying computational mesh (right) of NACA0012 profile each with detailed view of leading edge at $Ma = 2$ and $t = 10$, simulated with $N = 5$, CFL $= 0.5$, $\sigma = 0.25$ and $r_s = 0.75$.

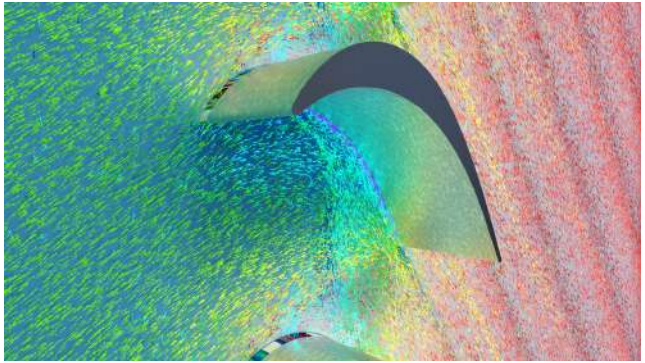
Wall Models

Erosion in Turbomachines

- Erosion due to ingested particles limits lifespan and safety (Eyjafjallajökull, 2010)
- Turbulent flow – particle and particle / wall interaction
- Model for individual P/W needed: velocities, angles, breaking, rotation, etc.

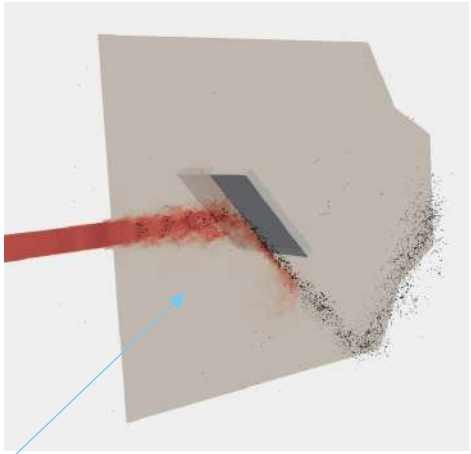


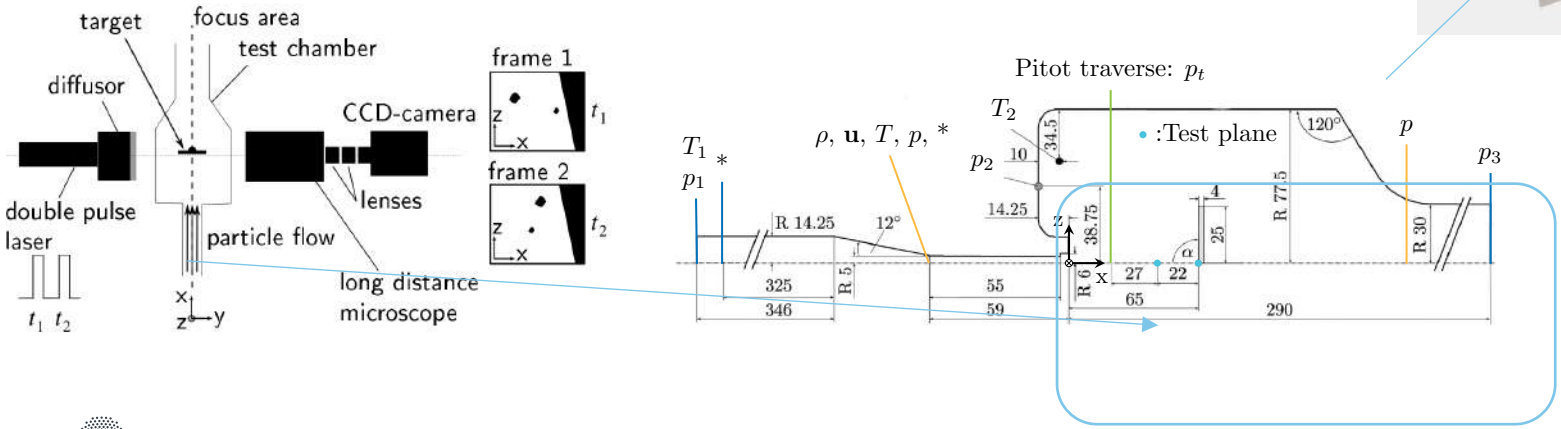




Measurement Campaign

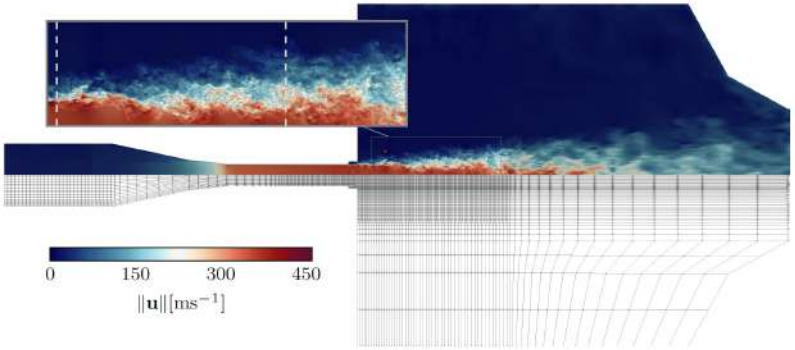
- Cooperation with Institute of Aircraft Propulsion Systems
- Problem: Particle paths are determined by consecutive particle/wall interactions
- Experimental and computational campaign
- Particles in turbomachines are fast

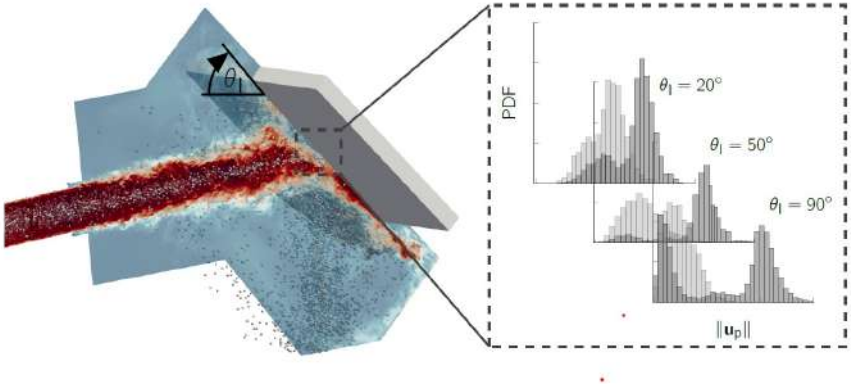


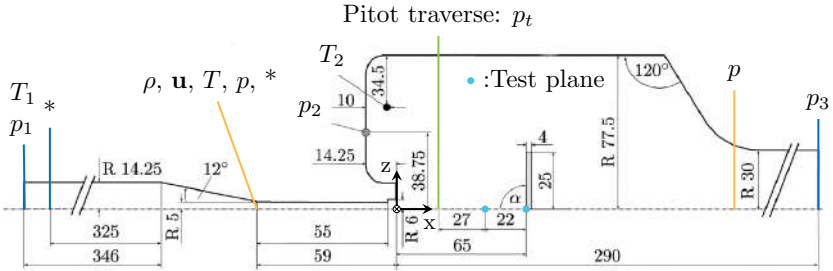


Particle / Wall models from data

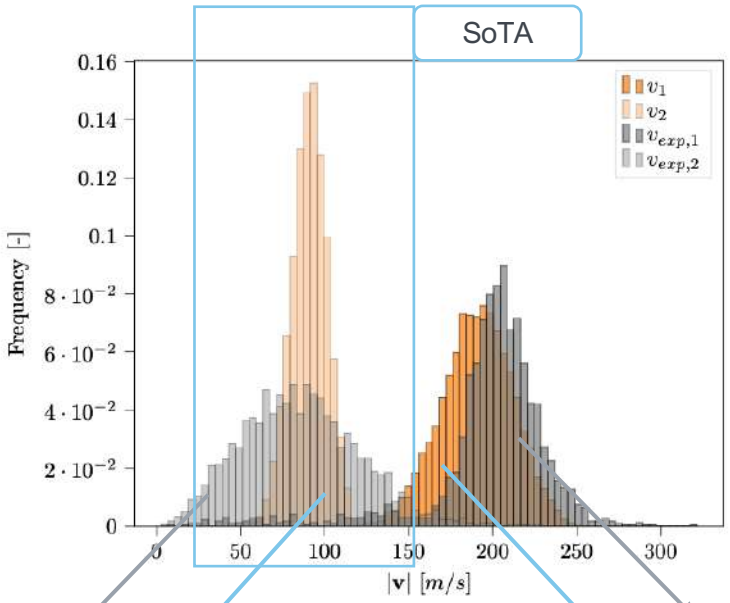
- Measurement in particle wind tunnel
- Supervised ML with a priori conditioning on physical constraints

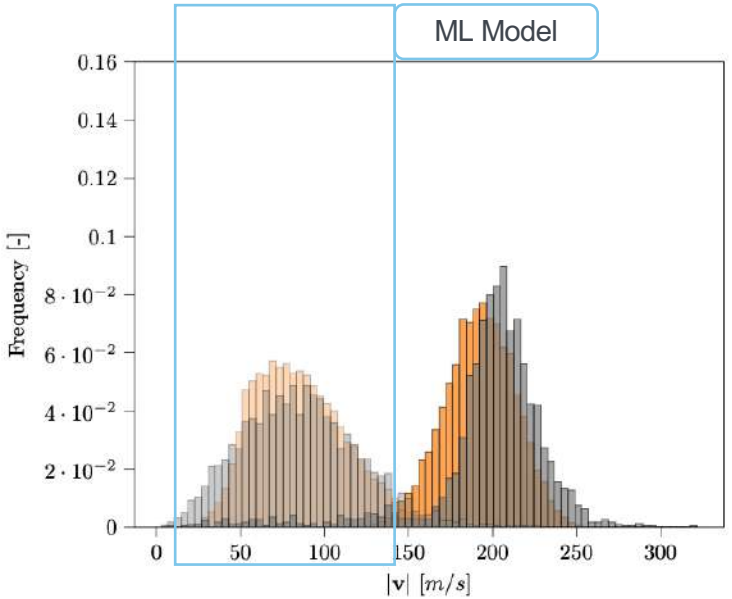






A posteriori model evaluation





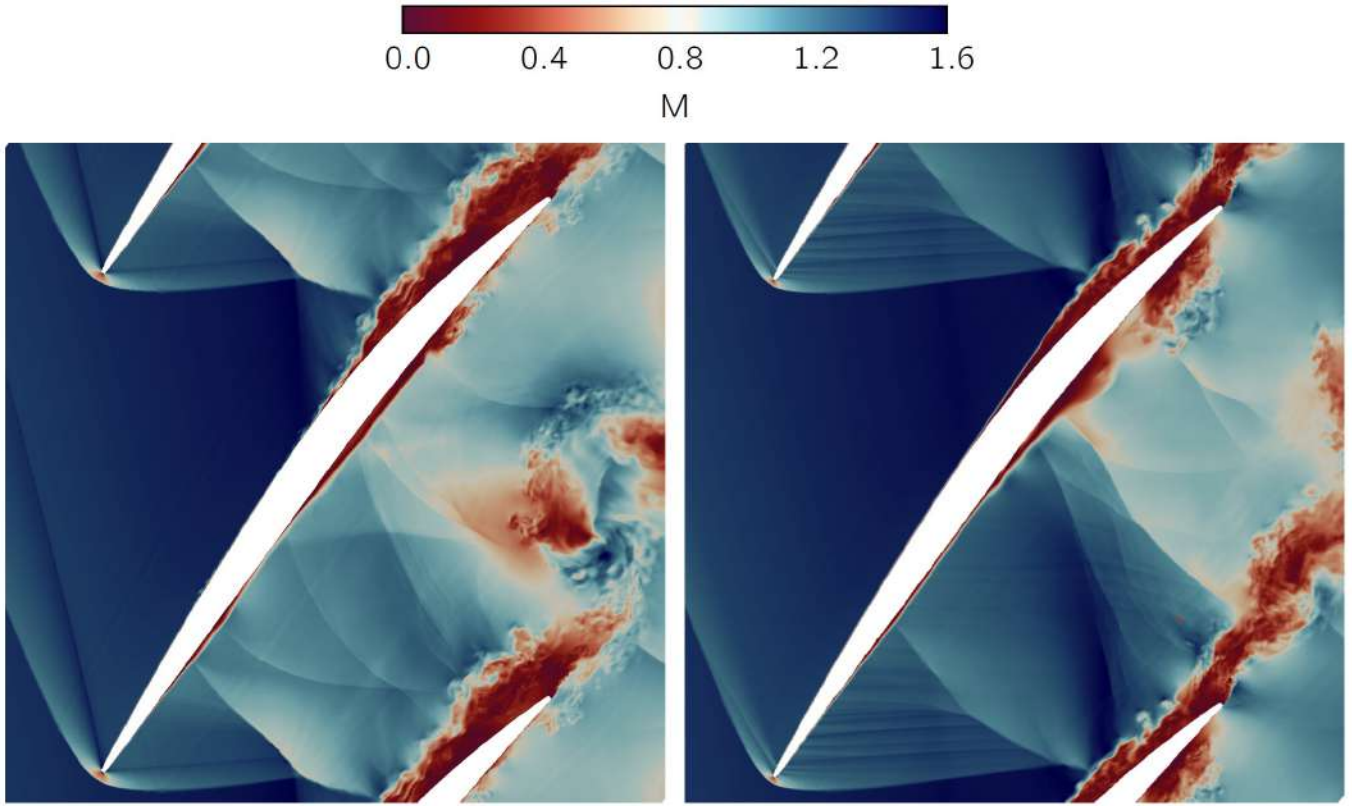
Rebound Exp.

Rebound Model

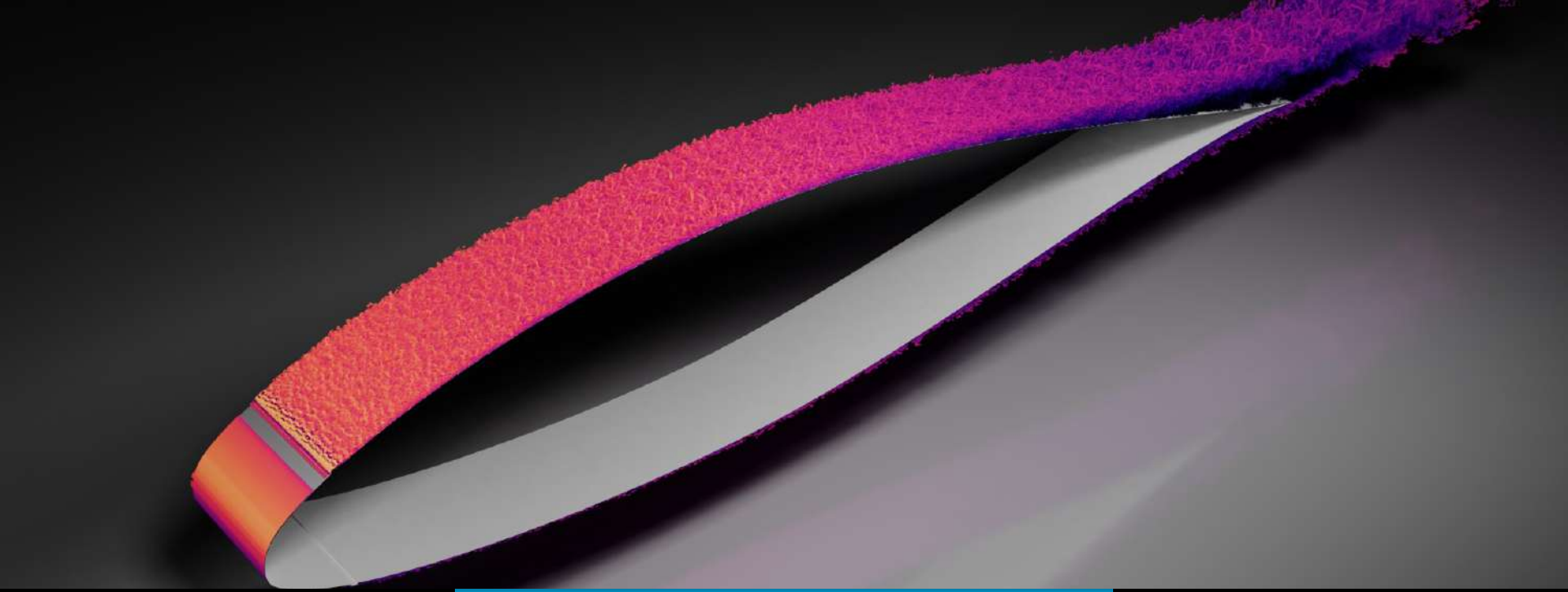
Incoming LES

Incoming Exp.

ML-driven, time-resolved erosion simulations



Back to LES

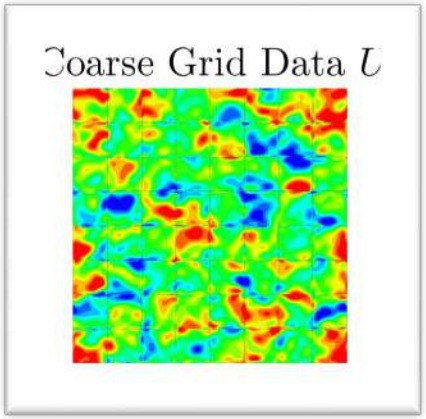


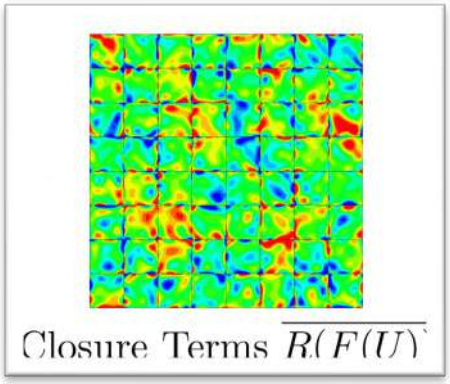
Turbulence closures from data

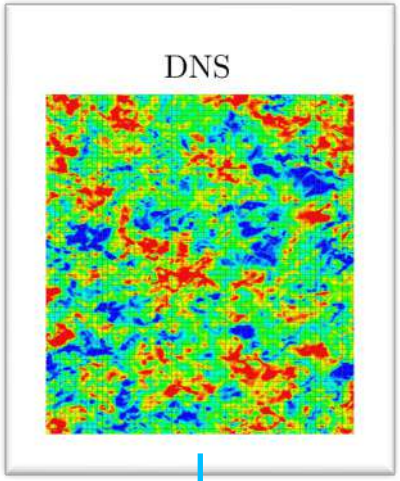
The Scale Gap in Turbulence

$$\underbrace{R\left(U_l, x_l, t_l\right) = 0}_{\text{Governing Equations at level l}}$$

$$\underbrace{R\left(U_L, x_L, t_L\right)}_{\text{Governing Equations at level L}} + \overbrace{M\left(U_l, U_L\right)}^{\text{Influence of level l on L}} = 0$$



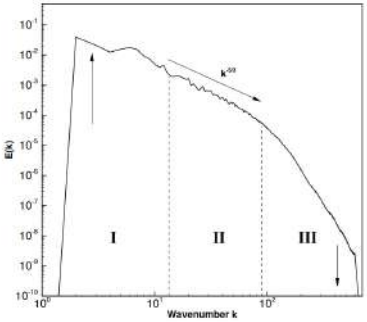




The Reynolds stresses: The footprint of the fine scale turbulence

Data-driven models

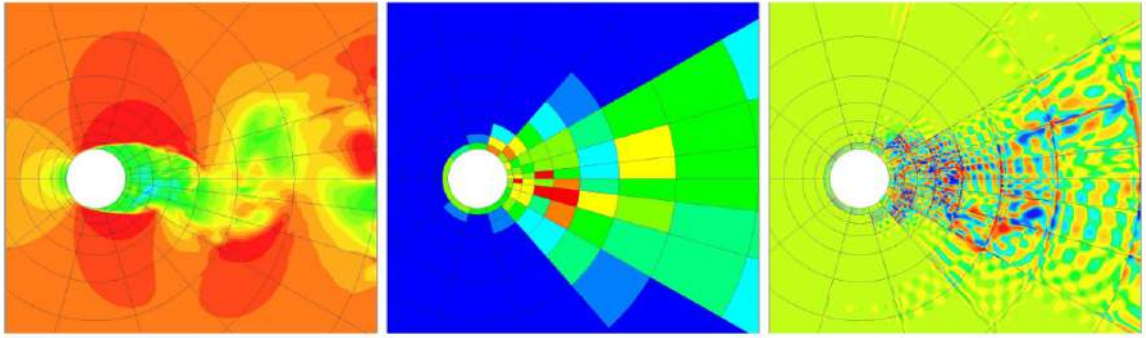
LES - Basics

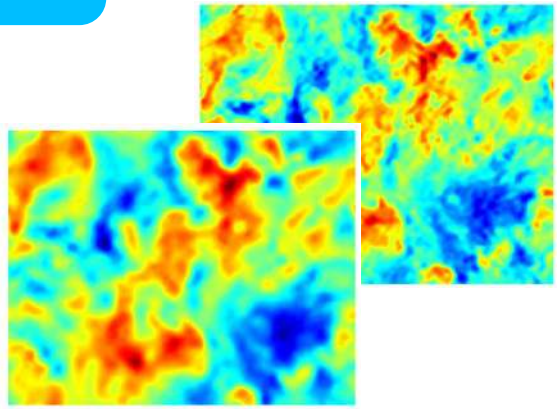


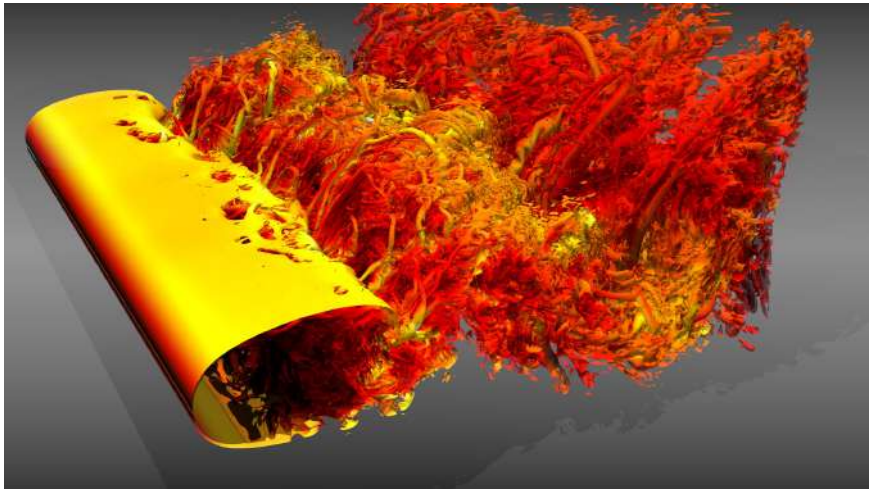
$$\overline{u(x)} = \int_{-\Delta x}^{\Delta x} u(r)G(x-r)dr$$

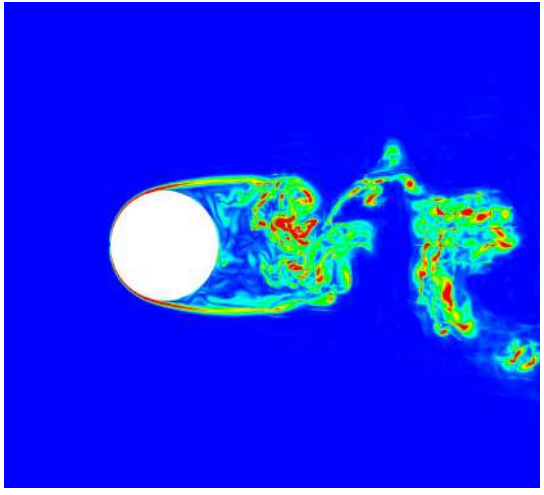
$$\frac{\partial \bar{u}_i}{\partial t} + \frac{\partial}{\partial x_j}\left(\bar{u}_i \bar{u}_j\right) = -\frac{1}{\rho}\frac{\partial \bar{p}}{\partial x_i} + 2\nu \frac{\partial}{\partial x_j}\bar{S}_{ij} - \frac{\partial \tau_{ij}}{\partial x_j}$$

$$\overline{u_i u_j} = \tau_{ij} + \bar{u}_i \bar{u}_j$$



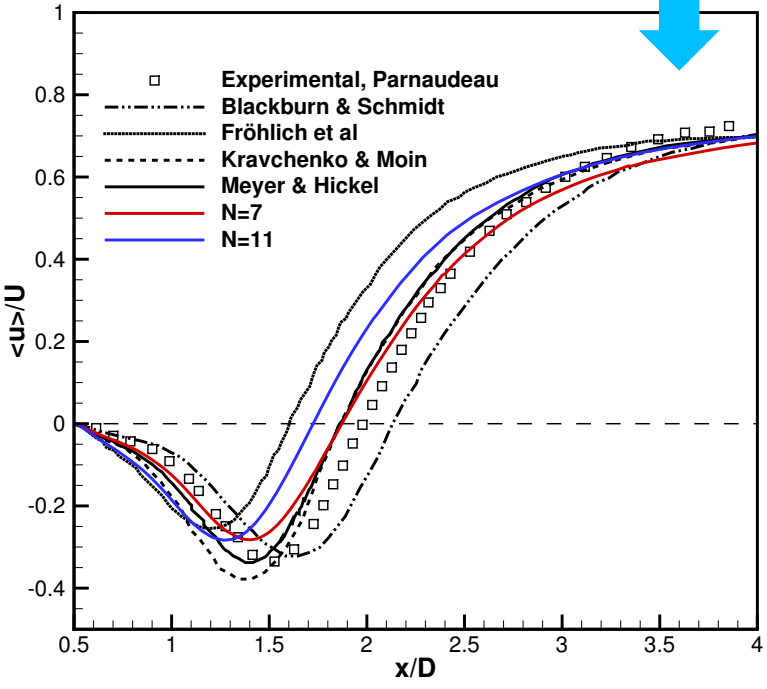




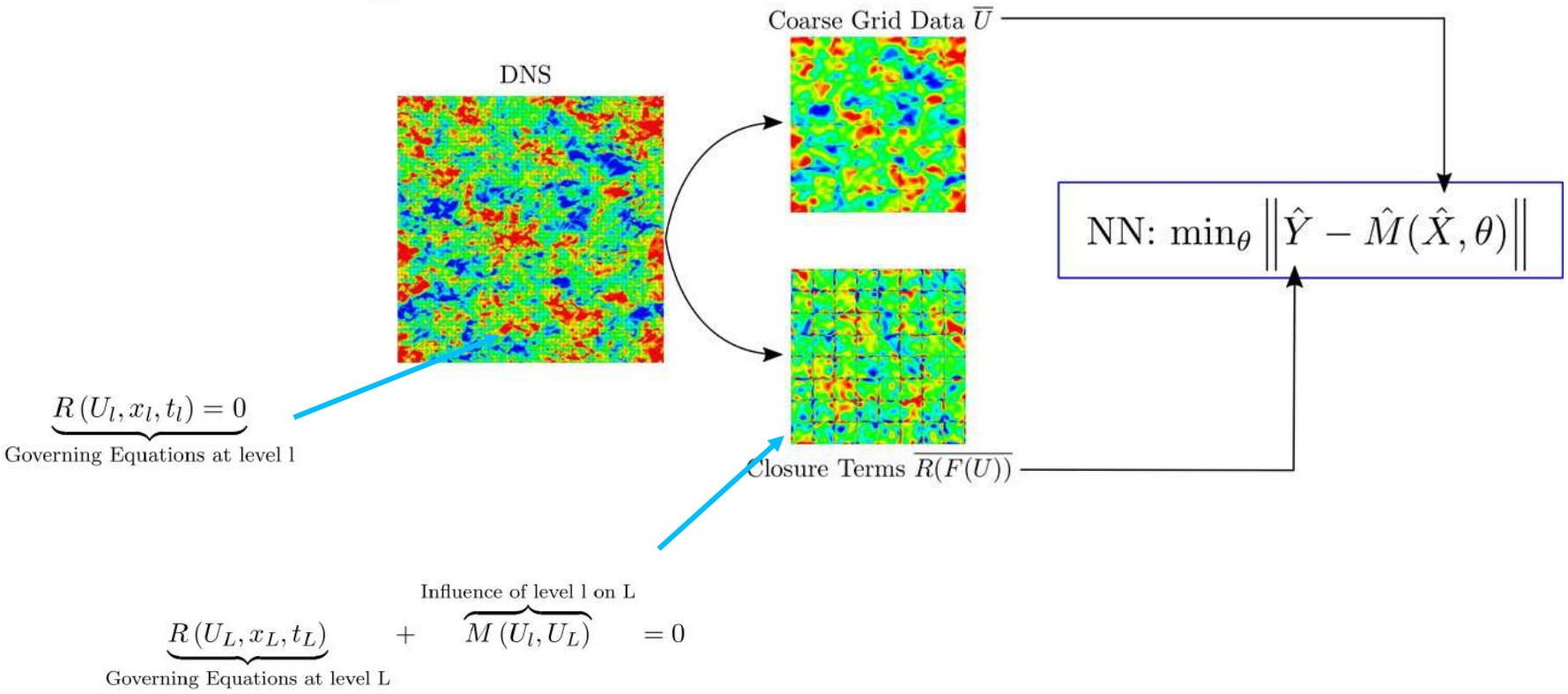


"It is the model's fault!"

"Same equations, domains, models…"



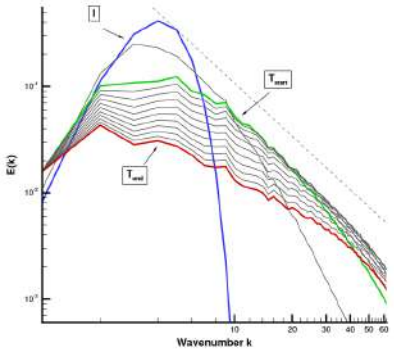
LES Closure Terms

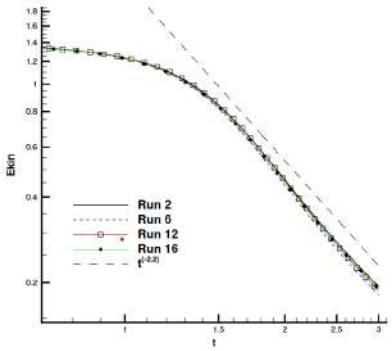


Reynolds terms only!

Data Acquisition: Decaying Homogeneous Isotropic Turbulence

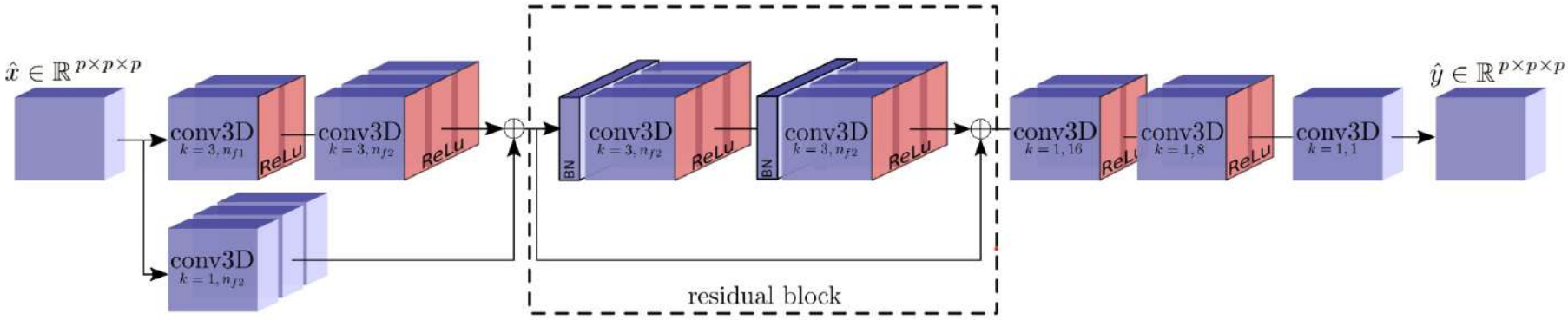
- Canonical test case of "turbulence in a box"
- Ensemble of DNS runs of decaying homogeneous isotropic turbulence with initial spectrum defined by Chasnov (1995) initialized by Rogallo (1981) procedure and Reλ = 180 at start
- Data collection in the range of exponential energy decay: 25 DHIT realizations with
- 134 Mio DOF each computed on CRAY XC40 (approx. 400,000 CPUh, 8200 cores)



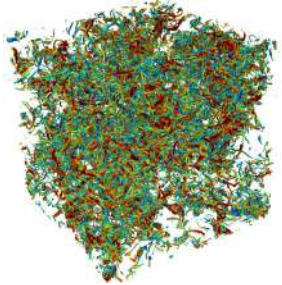


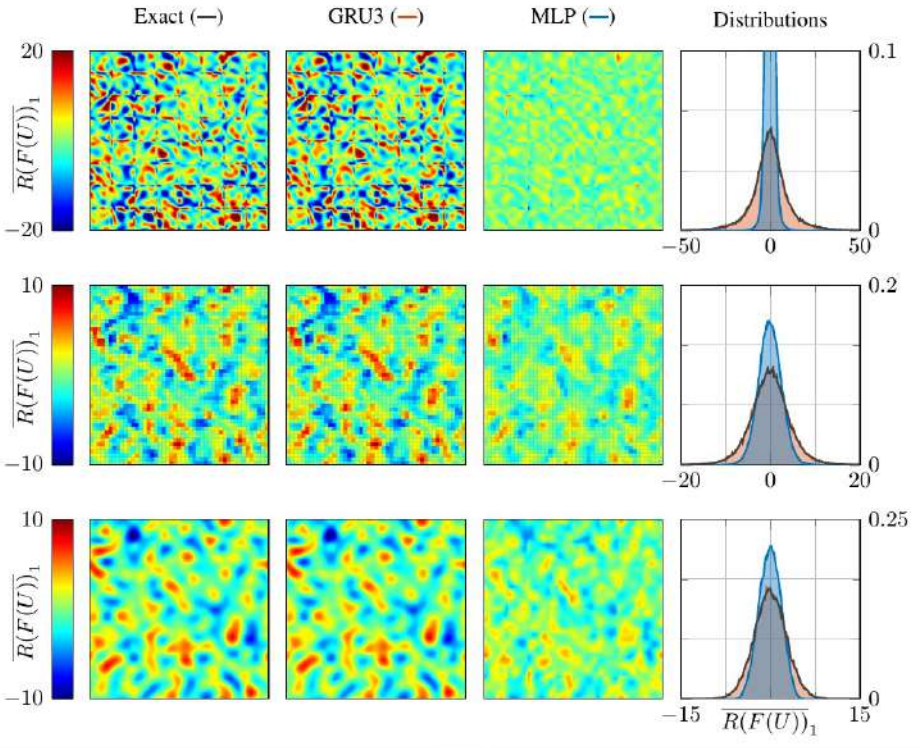
Networks and Training

- CNNs with skip connections (RNN), batch normalization, ADAM optimizer, data augmentation
- Different network depths (no. of residual blocks)
- For comparison: MLP with 100 neurons in 1 hidden layer
- Implementation in Python / Tensorflow, Training on K40c and P100 at HLRS
- Split in training, semi-blind validation and blind test DHIT runs



LES Closure Terms

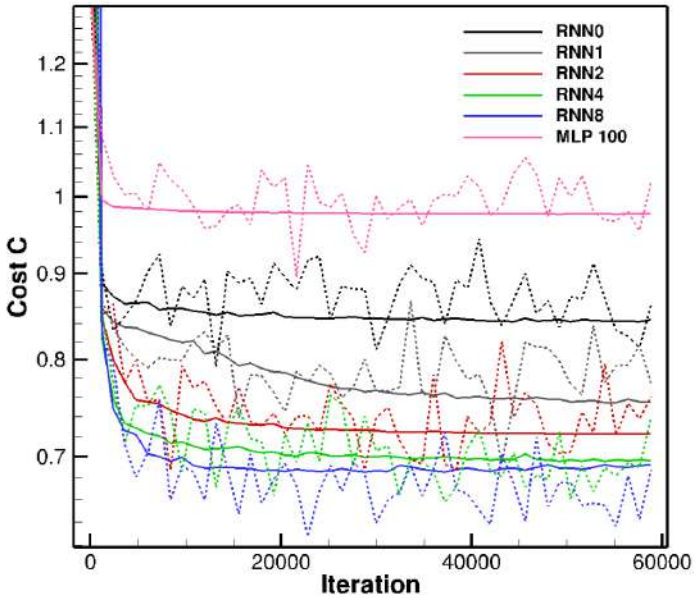




- 99.9999% Cross correlation with GRU models
- Near perfect accuracy in predicting Reynolds terms. Is this sufficient?

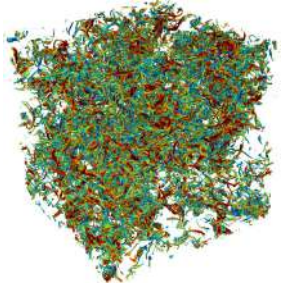
Training Results I: Cost Function

- Cost function for different network depths
- RNNs outperform MLP, deeper networks learn better
- The approach is data-limited. NNs are very data-hungry.



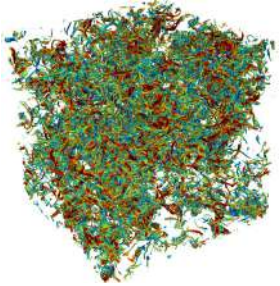
Training Results II: Cross Correlation

Smagorinsky: about 0%, scale similarity models: about 60%

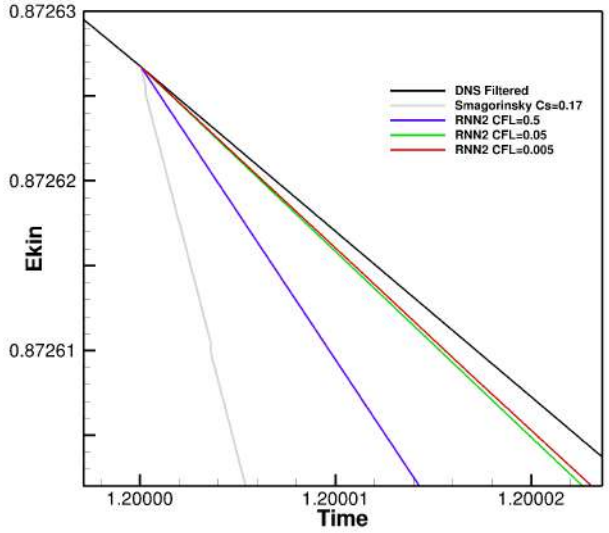


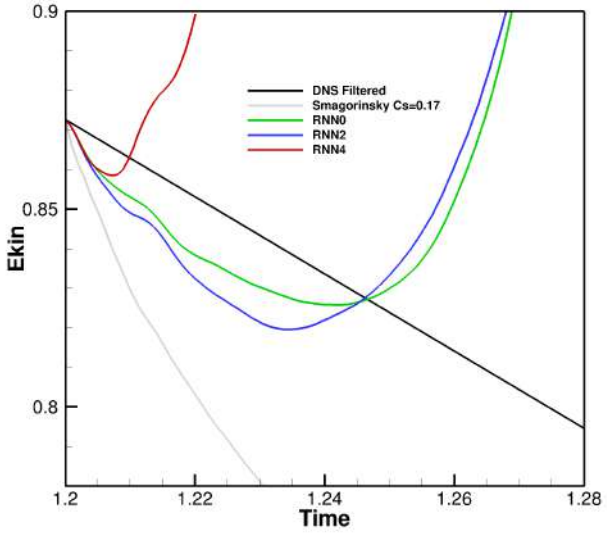
Network	a, b	$\mathcal{CC}(a,b)$	$\mathcal{CC}^{inner}(a,b)$	$\mathcal{CC}^{surf}(a,b)$
RNN0	$\overline{R(F(U))^1}, \overline{R(F(U))^1}^{ANN}$	0.347676	0.712184	0.149090
	$\overline{R(F(U))^2}, \overline{R(F(U))^2}^{ANN}$	0.319793	0.663664	0.134267
	$\overline{R(F(U))^3}, \overline{R(F(U))^3}^{ANN}$	0.326906	0.669931	0.101801
RNN4	$\overline{R(F(U))^1}, \overline{R(F(U))^1}^{ANN}$	0.470610	0.766688	0.253925
	$\overline{R(F(U))^2}, \overline{R(F(U))^2}^{ANN}$	0.450476	0.729371	0.337032
	$\overline{R(F(U))^3}, \overline{R(F(U))^3}^{ANN}$	0.449879	0.730491	0.269407

LES with trained ANN model

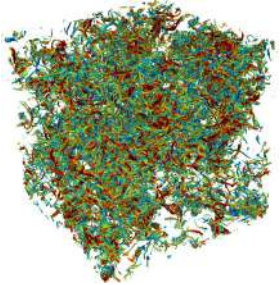


- Direct closure, no constraints incorporated
- No long term stability, but short term stability and dissipation
- Improvement: Loss function penalization of anti-dissipative stress tensor





LES Closure Terms: GRU



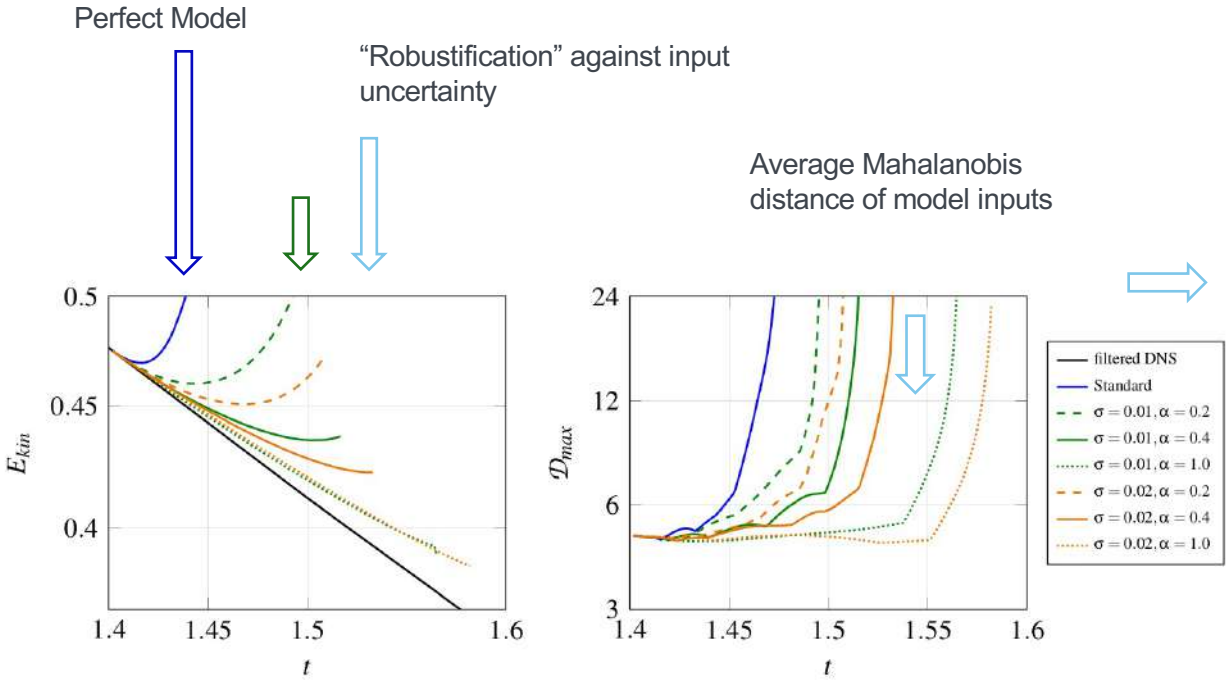
Network	# Parameter	Time (GPU)	Time (CPU)	L_2-Error	CC
MLP	$6,720$	6 ms	28 ms	$3.0 \cdot 10^{+1}$	66.0%
CNN	$187,088$	72 ms	198 ms	$2.1 \cdot 10^{+1}$	78.7%
LSTM $(3\Delta t)$	$39,744$	62 ms	340 ms	$1.3 \cdot 10^{-1}$	99.9%
GRU $(3\Delta t)$	$31,578$	59 ms	319 ms	$1.1 \cdot 10^{-1}$	99.9%

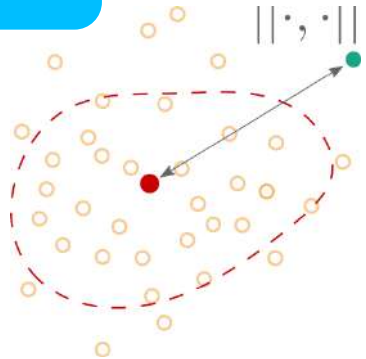
$$\frac{\partial \overline{U}}{\partial t} + \widetilde{R}((\overline{U})) = \widetilde{R}((\overline{U})) - \overline{R((U))}$$

Prediction on test set with arbitrary accuracy > 99.9999% CC

LES Closure Terms: Direct Approach

Chaotic, dynamical systems and MDI are not a good match!





- Arbitrarily accurate closure terms, but stability is lacking: Stability training and incorporating of physical constraints helps only a bit
- Can we train robust and accurate models this way for practical applications?

LES Closure Terms: Useful Models

Predict Reynolds stresses with ML, then find best fit to stable model

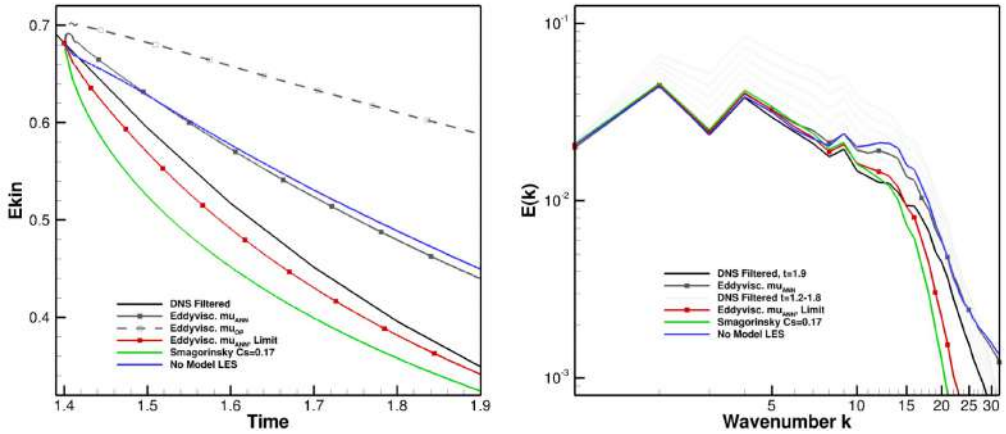
$$\frac{\partial \overline{U}}{\partial t} + \widetilde{R}(F(\overline{U})) = \underbrace{\widetilde{R}(F(\overline{U})) - \overline{R(F(U))}}_{\text{data-based eddy viscosity model}} .$$

Accurate Prediction, stability issues

- Simplest model: Eddy viscosity approach with μ_{ANN} from

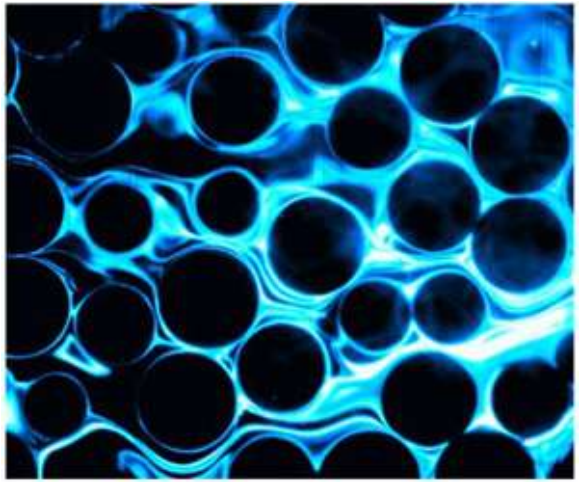
$$\widetilde{R}(F(\overline{U^i})) - \overline{R(F(U^i))} \approx \mu_{ANN}\, \widetilde{R}(F^{visc}(\overline{U^i}, \nabla \overline{U^i}))$$

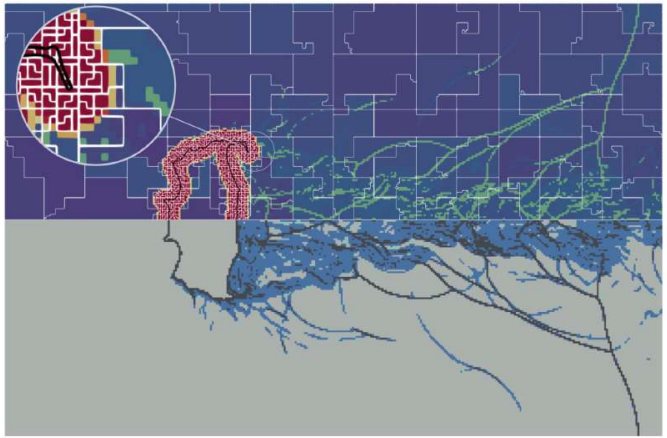
Project prediction onto guaranteed stable model

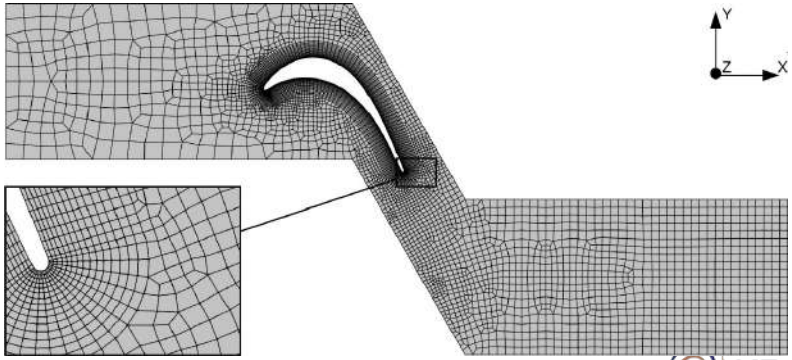


We want LES models that work for …

- Adaptive discretizations
- Unstructured grids
- All kinds of flows across Re, Ma, Sc, …





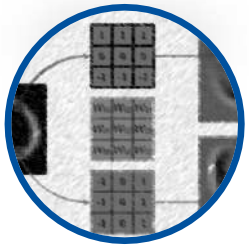


Maybe SL is not the way forward?

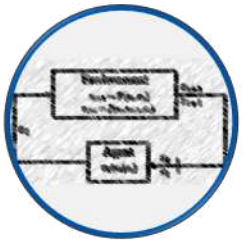
1. Do “smarter” SL
2. Do “smarter” ML

Overview

Part 1: Multi-X problems and PDE solvers

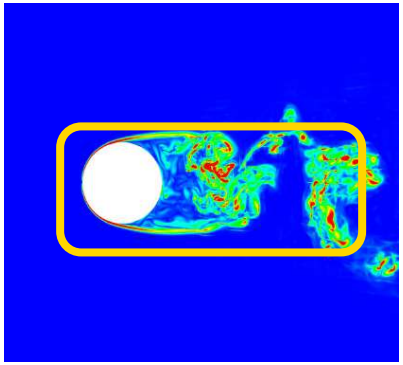


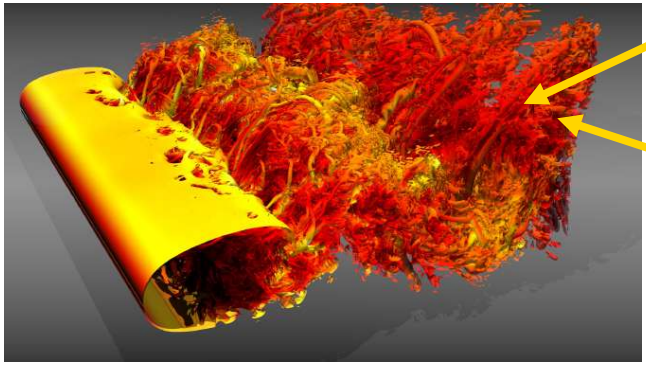
Part 2: Supervised learning for PDE submodels

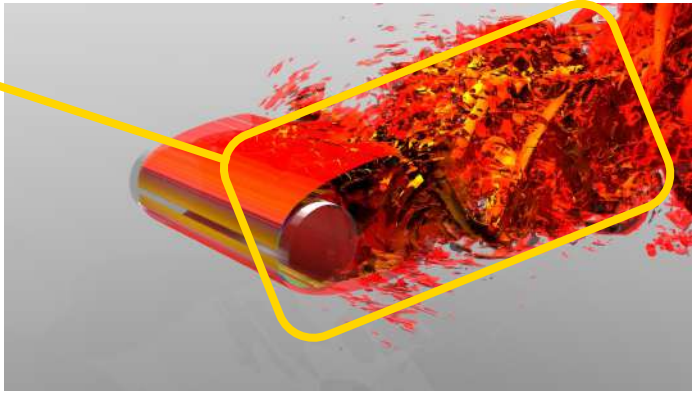


Part 3: From data-driven to integrated CFD/ML

Revisiting the LES problem







- Computationally too expensive
- **Data rich and efficiency poor?**
- SL Database and Curation?

- Closure models must account for **physical and computational** subfilter stresses

What is a computational subfilter stress?

Large Eddy Simulation - Definition

$$u = \overline{u}(x,t) + u(x,t)' \Rightarrow \overline{u}(x,t) = \int_{\Delta r} u(r,t)G(x-r)dr$$

- Separate Δr and h
- Explicit filtering: $h \rightarrow 0$
- Discretization scheme not relevant
- Caveat: homogeneity and isotropy, boundary conditions, realizability, commutation…

0.5% on google scholar

- Joined Δr and h
- Implicit filtering: $h \not\rightarrow 0$
- Discretization scheme defines filter kernel
- $\Leftarrow$ plus discretization parameters and errors

99.5% on google scholar, 100% for “industrial” LES

We derive the LES equations and the closure terms for a convolution with a known, homogenous, linear filter kernel with commutation

$$\overline{\psi}(x) := F(\psi(x)) = \int_{\infty}^{\infty} k(\xi - x, \Delta)\psi(\xi)d\xi,$$

Use Burgers equations as the simplest model for non-linearity

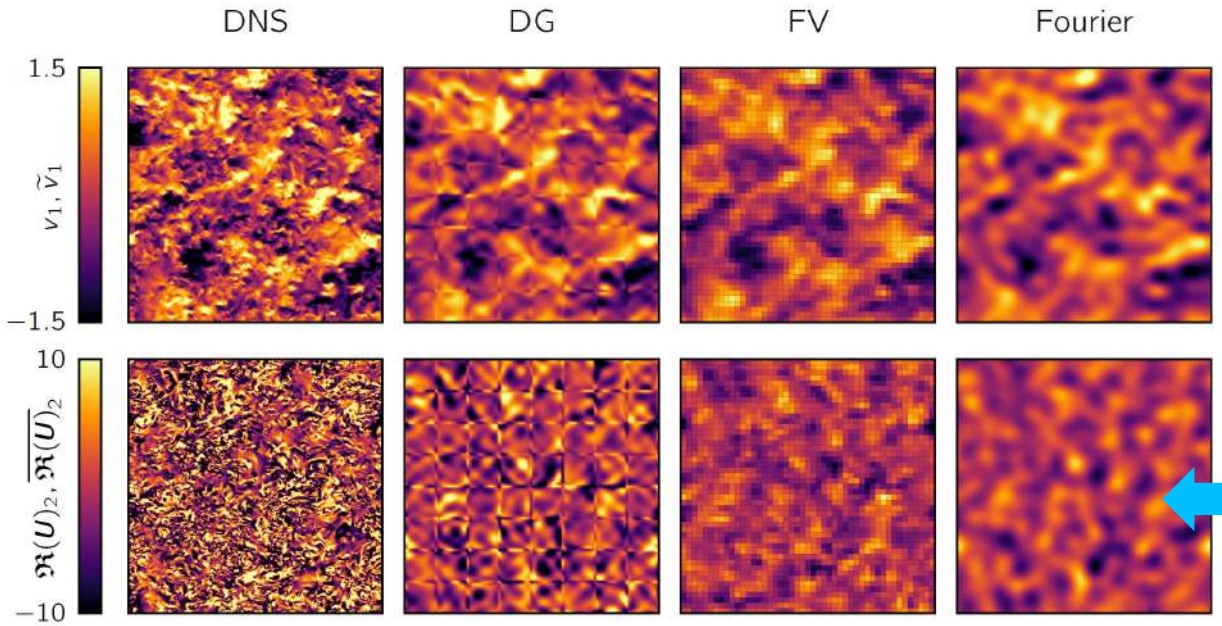
$$R(u) = \frac{\partial u}{\partial t} + \frac{1}{2}\frac{\partial u^2}{\partial x} = 0,$$

$$\overline{R(u)} = \frac{\partial \overline{u}}{\partial t} + \frac{1}{2}\frac{\partial \overline{u^2}}{\partial x} = \frac{1}{2}\underbrace{\left(\frac{\partial \overline{u^2}}{\partial x} - \frac{\overline{\partial u^2}}{\partial x}\right)}_{C_1[F;\frac{\partial}{\partial x}](u^2)}$$

$$= \frac{\partial \overline{u}}{\partial t} + \frac{1}{2}\frac{\partial \overline{u}\,\overline{u}}{\partial x} = C_1 + \frac{1}{2}\underbrace{\left(\frac{\partial \overline{u}\,\overline{u}}{\partial x} - \frac{\partial \overline{u^2}}{\partial x}\right)}_{C_2[F;()^2](u)},$$

- You find this as the Reynolds stresses in any textbook

Almost all practical LES: filtering is implicit, through grid / operator / discretization:
1.) Discretization characteristics induce filter shape



The induced filter results in different Reynolds stresses

2.) Resolved and represented scales

$$h \approx \Delta$$

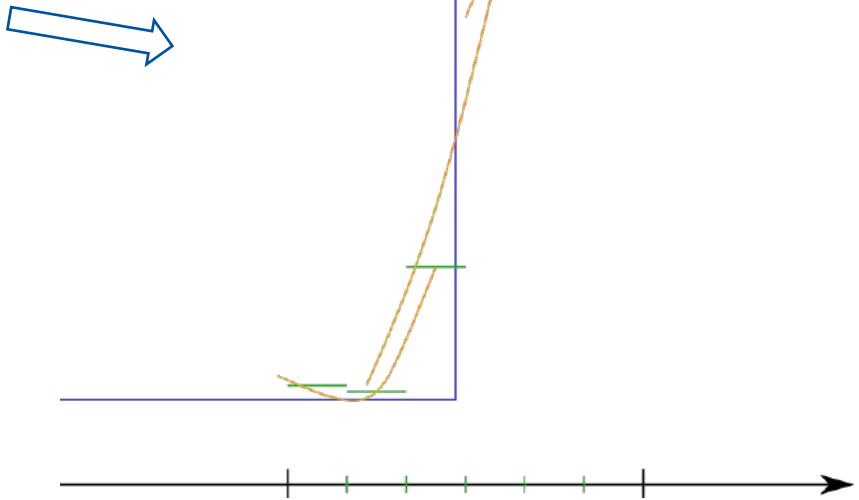
Discrete flux operator induces computational subgrid stresses

$$\overline{R(u)} = \frac{\partial \overline{u}}{\partial t} + \frac{1}{2}\frac{\delta \overline{u}\,\overline{u}}{\delta x} = C_1 + C_2 + \frac{1}{2}\underbrace{\left(\frac{\delta \overline{u}\,\overline{u}}{\delta x} - \frac{\partial \overline{u}\,\overline{u}}{\partial x}\right)}_{C_3[\partial_x;\delta_x]},$$

$$\overline{R(u)} = \frac{\partial \overline{u}}{\partial t} + \frac{1}{2}\frac{\delta \overline{u}\,\overline{u}}{\delta x} = C_1 + C_2 + \frac{1}{2}\frac{\partial}{\partial x}\underbrace{\left(\widehat{\overline{u}\,\overline{u}} - \overline{u}\,\overline{u}\right)}_{C_3^*[\widehat{\ };\cdot]}; \; \delta_x f = \partial_x \hat{f}$$

$$\overline{R(u)} = \frac{\partial \overline{u}}{\partial t} + \frac{1}{2}\frac{\delta \overline{u}\,\overline{u}}{\delta x} = C_1 + C_2 + \frac{1}{2}\left(\frac{\delta \overline{u}\,\overline{u}}{\delta x} - \frac{\partial \overline{u}\,\overline{u}}{\partial x}\right),$$

$$\overline{R(u)} = \frac{\partial \overline{u}}{\partial t} + \frac{1}{2}\frac{\delta \overline{u}\,\overline{u}}{\delta x} = C_1 + C_2 + \frac{1}{2}\frac{\partial}{\partial x}\underbrace{\left(\widehat{\overline{u}\,\overline{u}} - \overline{u}\,\overline{u}\right)}_{C_3^*[\widehat{\cdot};\cdot]}; \quad \delta_x f = \partial_x \hat{f}$$



Linear numerical schemes for solving partial differential equations, having the property of not generating new extrema (monotone schemes), can be at most first-order accurate.

Think: Upwinding schemes, characteristic based schemes, WENO, SUPG, …

Challenges for data-driven modelling

$$\overline{R(u)} = \frac{\partial \overline{u}}{\partial t} + \frac{1}{2}\frac{\delta \overline{u}\,\overline{u}}{\delta x} = C_1 + C_2 + \frac{1}{2}\underbrace{\left(\frac{\delta \overline{u}\,\overline{u}}{\delta x} - \frac{\partial \overline{u}\,\overline{u}}{\partial x}\right)}_{C_3[\partial_x;\delta_x]},$$

Make SL models scale-aware

Train SL models on Reynolds stresses from DNS or Exp.

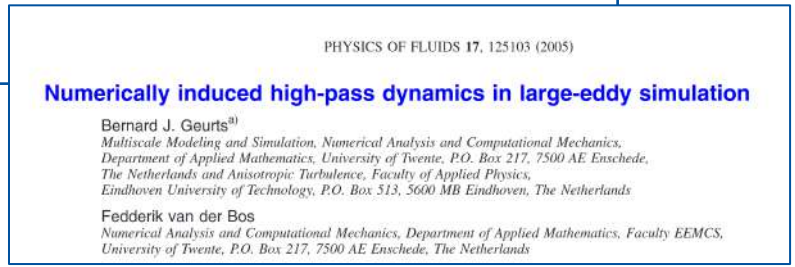
$\delta^n = \delta(\overline{u}^n), \quad \overline{u}^{n+1} = \overline{u}(\delta^n)$

Knowable a priori

Not knowable a priori

"… the computational large-eddy closure typically contains an important contribution that is sensitive to the adopted spatial discretization"

"… considerable improvements of practical large-eddy simulation may be achieved in case the closure problem in the modified equation is made to explicitly account for the coarseness as well as the type of discretization."



PHYSICS OF FLUIDS 17, 125103 (2005)

Numerically induced high-pass dynamics in large-eddy simulation

Bernard J. Geurts[a)]
Multiscale Modeling and Simulation, Numerical Analysis and Computational Mechanics, Department of Applied Mathematics, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands and Anisotropic Turbulence, Faculty of Applied Physics, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, The Netherlands

Fedderik van der Bos
Numerical Analysis and Computational Mechanics, Department of Applied Mathematics, Faculty EEMCS, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands

We derive the LES equations and the closure terms for a known, homogenous, linear filter kernel with commutation…

… Scale Separation filtering in practical LES is done implicitly with a partially unknown, inhomogeneous, possibly anisotropic and often non-linear filter kernel

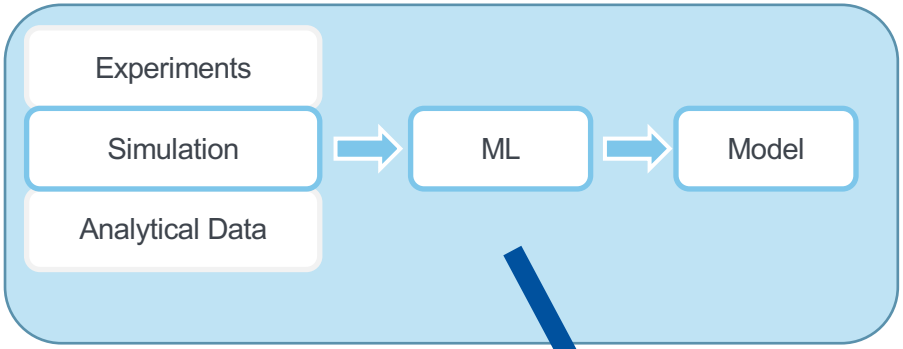
What makes LES so tricky? more nonlinearities than we thought!

Challenge: A simple example

$$\frac{\partial u}{\partial t} + \frac{1}{2}\frac{\partial u^2}{\partial x} = \nu \frac{\partial^2 u}{\partial x^2}$$

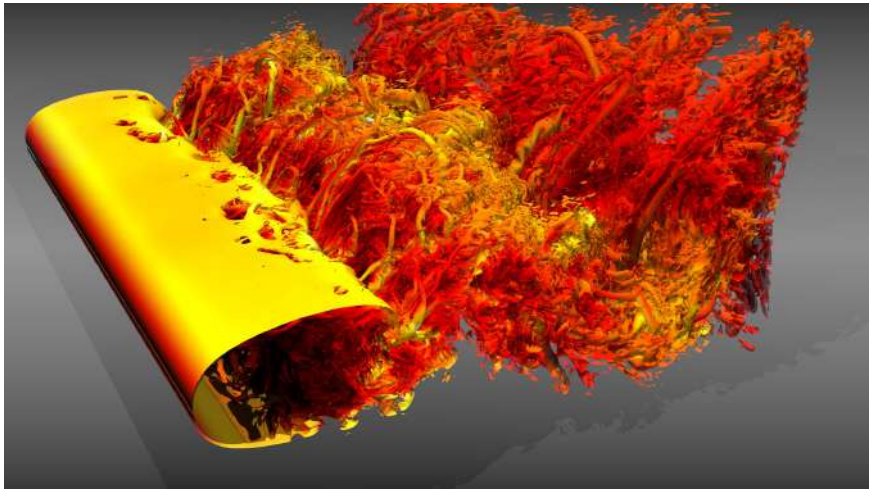
$$\frac{\partial u}{\partial t} + R(U) = L(U)$$

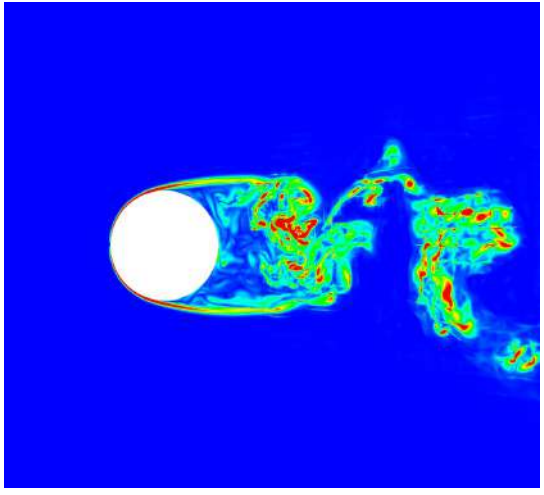
$$\frac{\partial \bar{u}}{\partial t} + R(\bar{U}) = L(\bar{u}) + \underbrace{R(\bar{u}) - \overline{R(u)}}_{S}$$



$$S \approx \frac{\partial}{\partial x}\left(\nu_{eddy}\frac{\partial \bar{u}}{\partial x}\right)$$

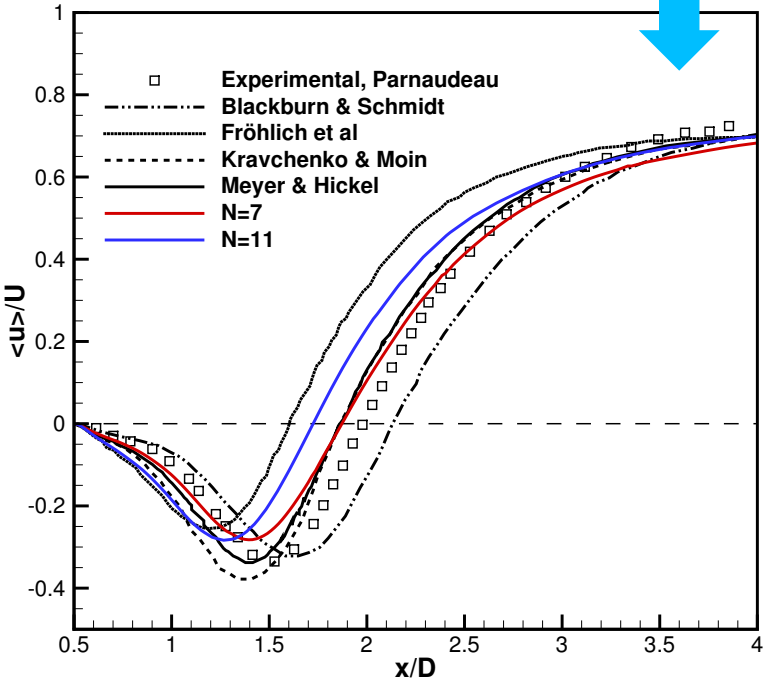
- Task: Find an optimal eddy viscosity for the Burgers equation from DNS data through SL and then use is in your favorite numerical scheme
- Beware: The gradient operator for an FD scheme, FV scheme, FE scheme looks different on the coarse grid!



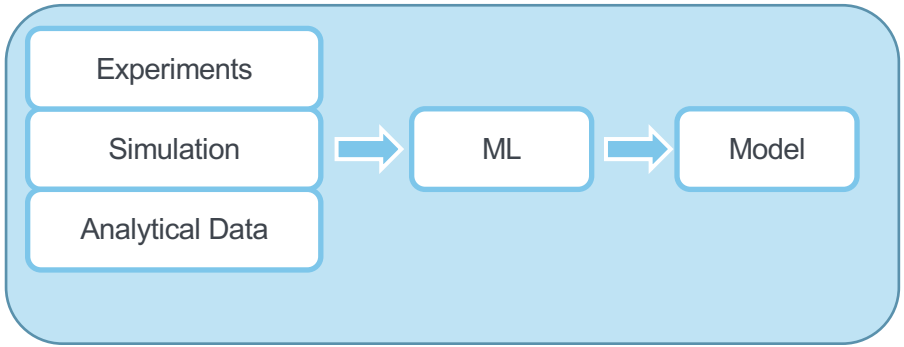


"Is it the model's fault? Did we account for C1 and C3?

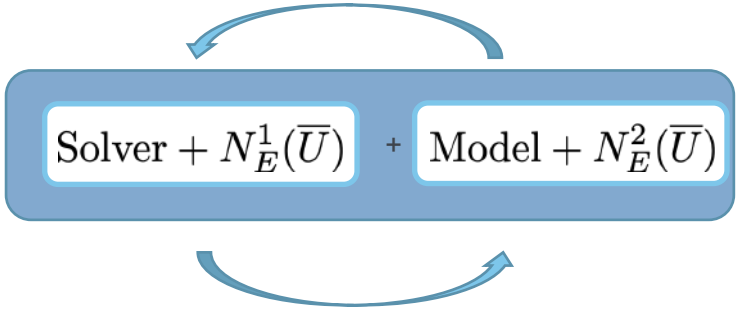
"Same equations, domains, models…"



Learning for dynamical systems: SL



Stage 1: Offline training on well-defined data



Stage 2: Online inference on “real” data

- Careful data generation and pre-processing
- Data analysis as a first step
- Dimensionality reduction
- Outlier detection

- ML model formulations with guarantees (symmetries, statistics, equivariances, scaling, …)
- KEP networks
- Cost function penalization
- Robustness against uncertain inputs

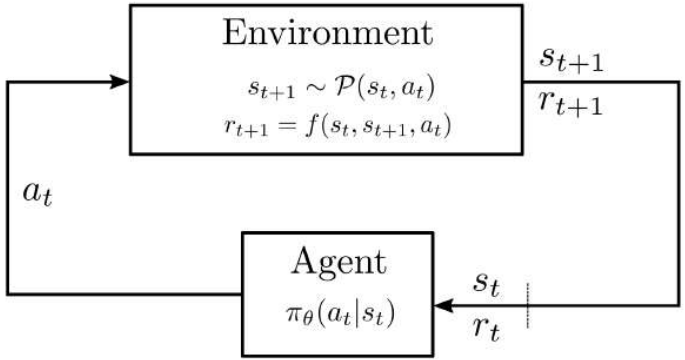
- Self-aware models with safe fallbacks
- Transfer learning
- Intermittent re-training

Learning for dynamical systems: RL

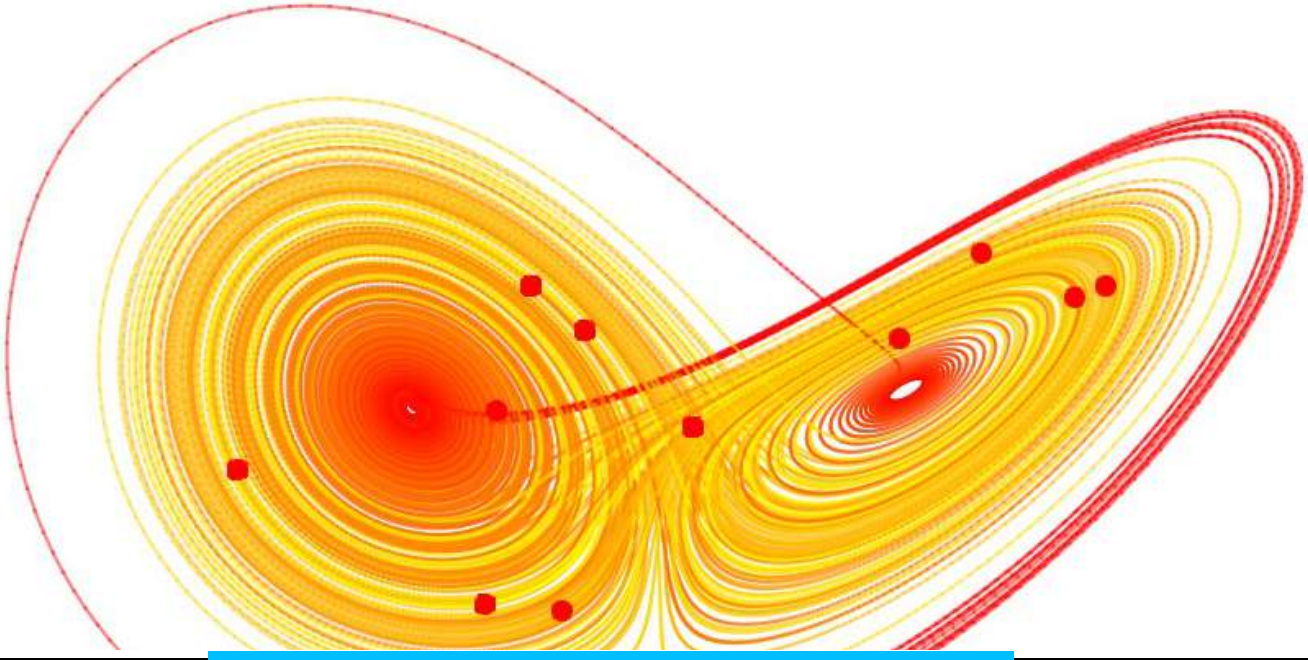
- Simulations deal with (discrete) dynamical systems
- Submodels need to be consistently integrated with the discretization: a simple gradient looks very different for a FD, FV, FE scheme
- Supervised learning has two problems: How do we define a lot of true data and make models robust?
- RL is a different paradigm from ML: Learning to take optimal actions in a dynamical system from experience

$$\widetilde{R}\left(U_L, x_L, t_L\right) = -\widetilde{M}\left(U_L\right)$$

ML with Solver-in-the-loop



Smarter ML: Learning optimal strategies for dynamical systems from observations



The briefest of brief intros to RL

Markov Decision Process (MDP)

- Extends Markov chains with actions & rewards
- Discrete-time stochastic control process defined by the tuple of actions, states, rewards and discounts

$$\langle S, A, P(\cdot), R(\cdot), \gamma \rangle, \text{with}$$

$$a \in A : \text{actions}$$

$$s \in S : \text{state of the system}$$

$$\gamma \in \mathcal{R} : \text{discount factor}$$

$$P(s_{k+1} = s' | \langle s_k = s, a_k = a \rangle) : \text{transition probability}$$

Markov Decision Process (MDP)

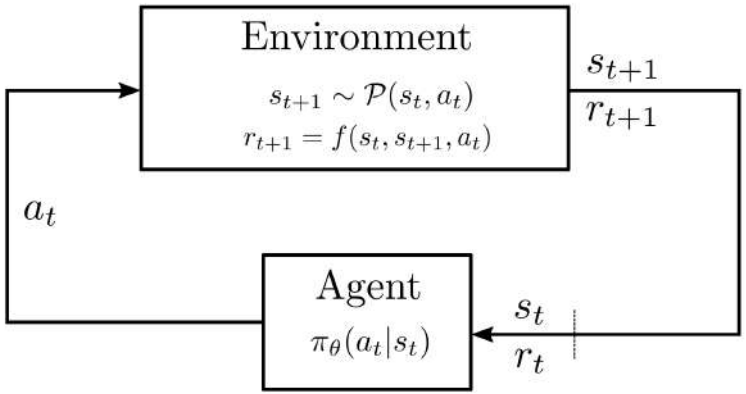
- Markovian IFF memoryless:

$$P(s_{k+1}|\langle s_k, a_k\rangle) = P(s_{k+1}|\langle s_0, a_0\rangle, ..., \langle s_k, a_k\rangle),\ \forall(s_\tau \in S, a_\tau \in A),\ \tau\{1, ..., k\}$$

- Formally solved by a policy: mapping from state-action space to the probability of taking action a when in state s:

$$\pi = S \times A \rightarrow \mathcal{R}, \langle s, a\rangle \rightarrow [0, 1]$$

Markov Decision Process (MDP)



$$\pi = S \times A \to \mathcal{R}, \langle s, a \rangle \to [0, 1]$$

Solved for the optimal policy, e.g. by reinforcement learning

Policy Optimization

- Find policy through optimization under uncertainty: Policy gradient vs. Q learning
- Some advantages: robustness, stochasticity, continuous action & state spaces
- Maximize the expected return of an episode

$$J(\theta) = E_\theta\left(\sum_{k=1}^{T} \gamma^k r_k\right)$$

- Optimization problem:

$$\max_\theta E_\theta\left(\sum_{k=1}^{T} \gamma^k r_k\right), \text{ subject to } \langle S, A, P(\cdot), R(\cdot)\rangle$$

- **Task**: find θ^* such that $\pi_{\theta*}(a_t|s_t)$ **maximizes the collected reward** r_t

- Use good old **Gradient Ascent**!

$$\theta^{new} = \theta^{old} + \alpha \nabla_\theta J(\theta)$$

- We have to estimate the **steepest gradient with respect to the weights** θ:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[\underbrace{\left(\sum_{k=1}^{N} \gamma^k r_k \right)}_{\text{Cum. reward over } \tau} \underbrace{\nabla_\theta \log \pi_\theta(\tau)}_{\text{Grad. of the policy}} \right]$$

- Approximate $\mathbb{E}_{\tau \sim \pi_\theta}$ by sampling some **discrete trajectories** $\tau^{(i)}$:

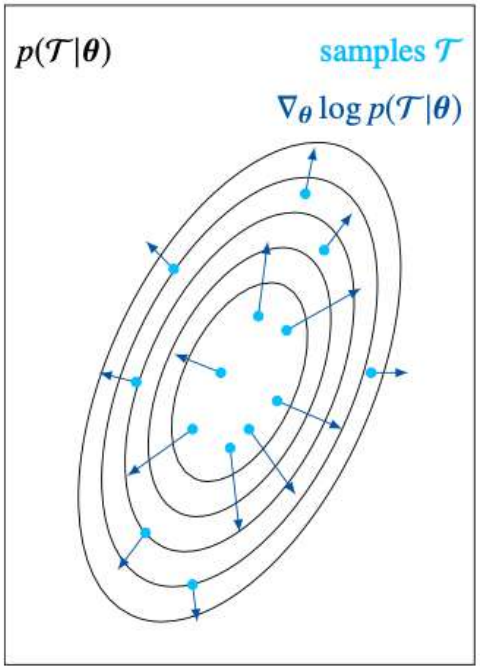
$$\tau^{(1)} = \{(s_0, a_0), (s_1, a_1, r_1), \ldots, (s_N, a_N, r_N)\}$$

Policy Optimization

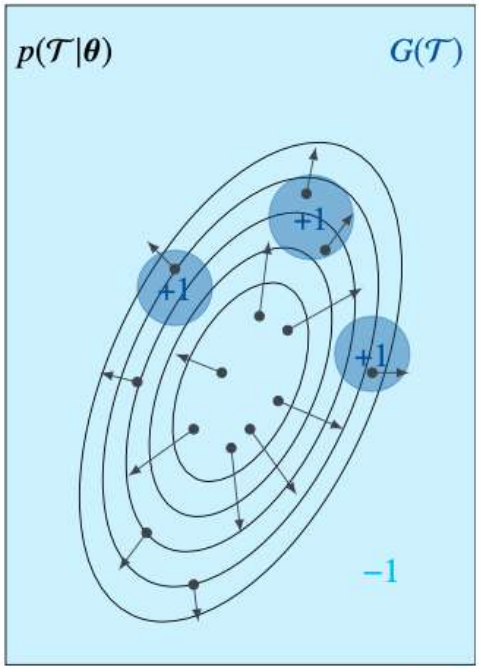
- This is the vanilla policy gradient : VPG
- Improvements: Trust Region PO: the policy update is limited so the KL divergence remains in a certain bound

$$\theta' \leftarrow \text{argmax}_{\theta'} \nabla_\theta J(\theta)^T(\theta - \theta'),$$
$$\text{such that } D_{KL}(\pi_{\theta'}(\cdot|s)||(\pi_{\theta'}(\cdot|s))) \leq \epsilon \, \forall \, s \in S$$

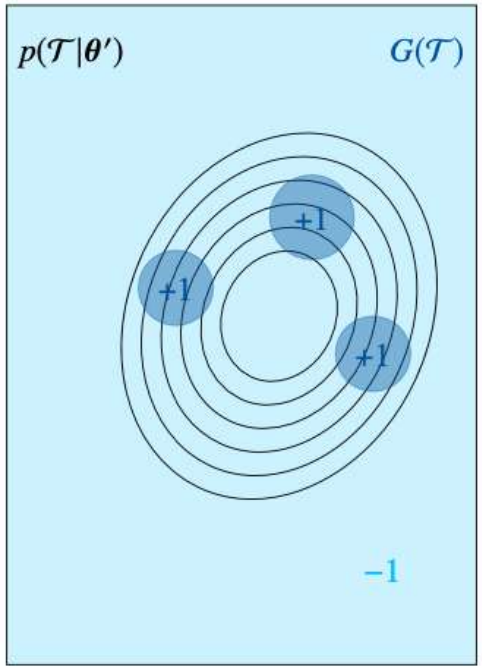
- Adding a baseline to the sampled return: Advantage function
- Clipping the objective function: Proximal Policy Optimization



(a) Samples and score functions

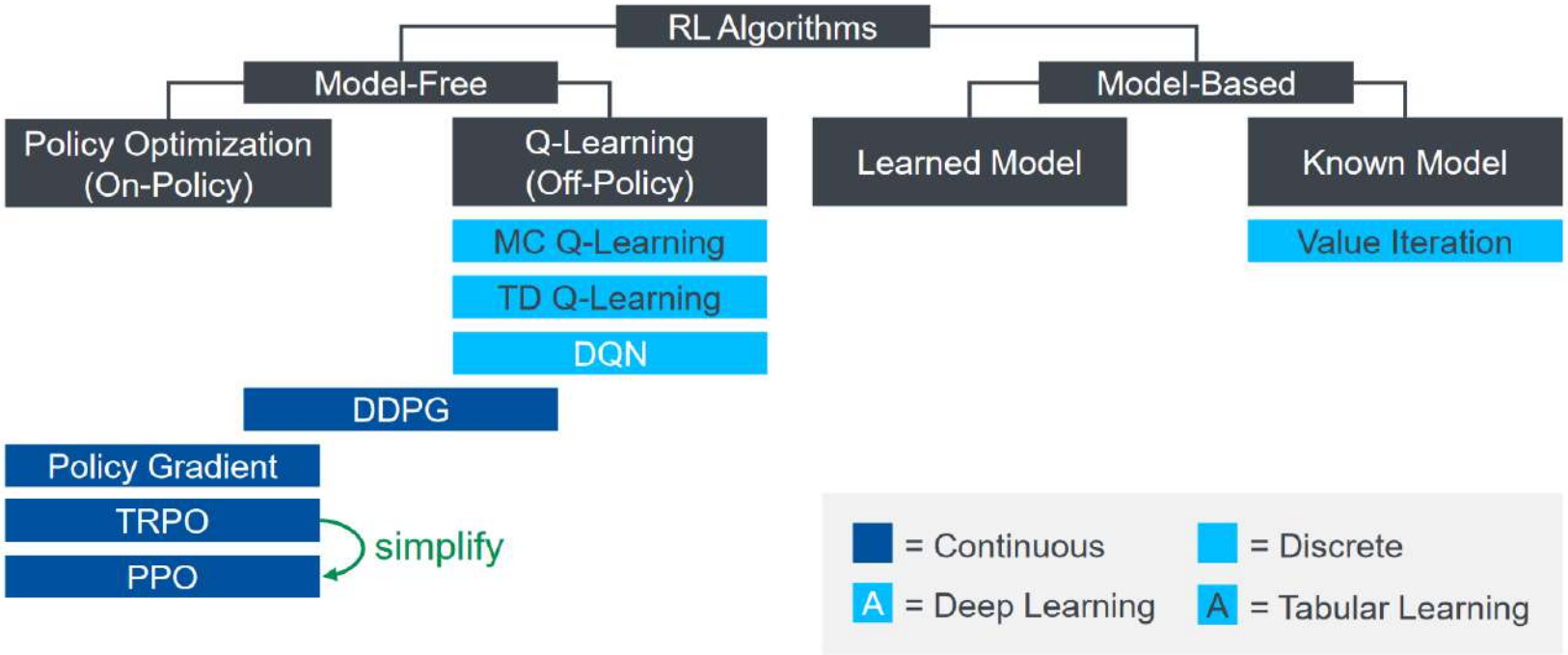


(b) Return superimposed to Fig. 5a



(c) Updated distribution

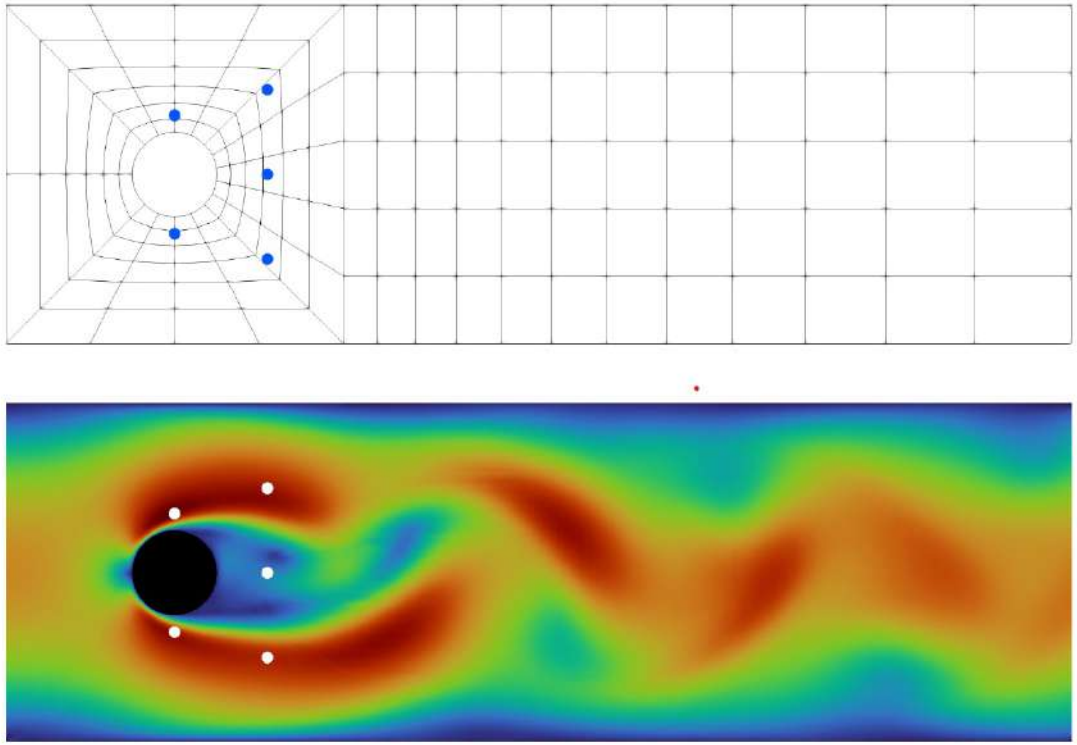
With permission from: Hierarchical Reinforcement Learning Approach Towards Autonomous Cross-Country Soaring, AIAA, Stefan Notter, IFR Uni Stuttgart


RL Algorithms
Model-Free
Model-Based
Policy Optimization (On-Policy)
Q-Learning (Off-Policy)
Learned Model
Known Model
MC Q-Learning
Value Iteration
TD Q-Learning
DQN
DDPG
Policy Gradient
TRPO
simplify
PPO
= Continuous
= Discrete
A = Deep Learning
A = Tabular Learning

A simple example first

Example 1: Flow control

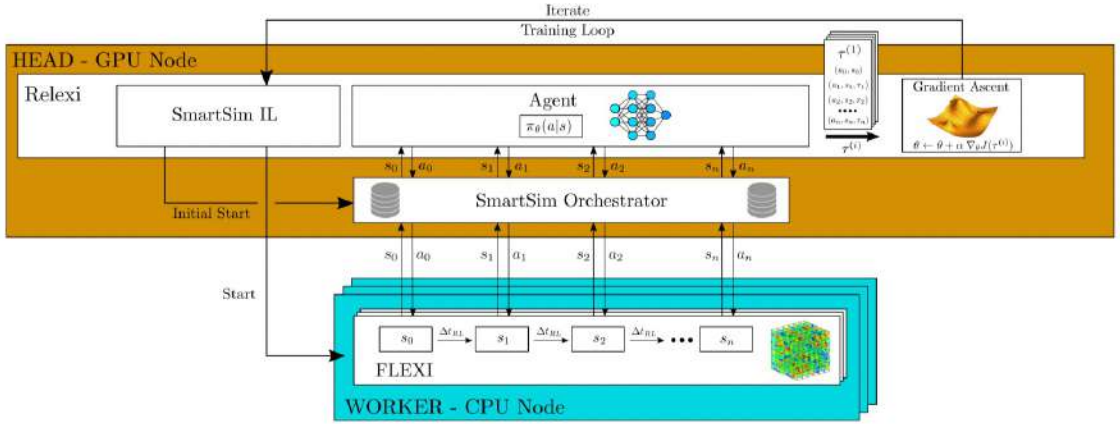
- Tight coupling of “learning” and “simulating”:



- Cylinder at Re = 100, Ma=0.2
- 5 Pressure probes as sensors
- Actions: Jets at the poles
- Reward: Drag reduction
- Training takes about 2h

Reinforcement learning framework – ReLeXI[1]: IAG, HLRS & HPE

- Distribution on hybrid HPC systems via the SmartSim Library[2]
- Dedicated GPU node („Head“) for training and model evaluation
- FLEXI instances interactively distributed across multiple CPU nodes („Workers“)
- Communication via in-memory database with the Redis library



[1]Kurz et al., Relexi—A scalable open source reinforcement learning framework for high-performance computing. Software Impacts, 2022
[2]https://github.com/CrayLabs/SmartSim

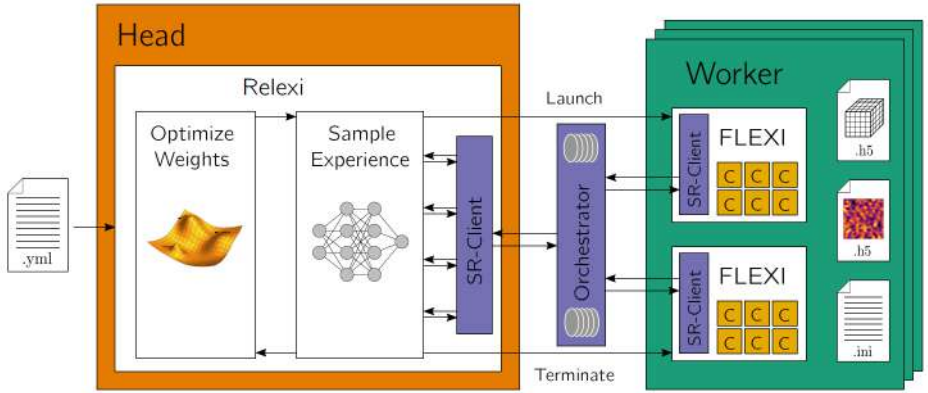
HPE Developer | HPE GreenLake | Products | OpenSource | Events | Skill Up | Community | Blog | SIGN IN | Search

Philipp Offenhäuser & Christopher Williams

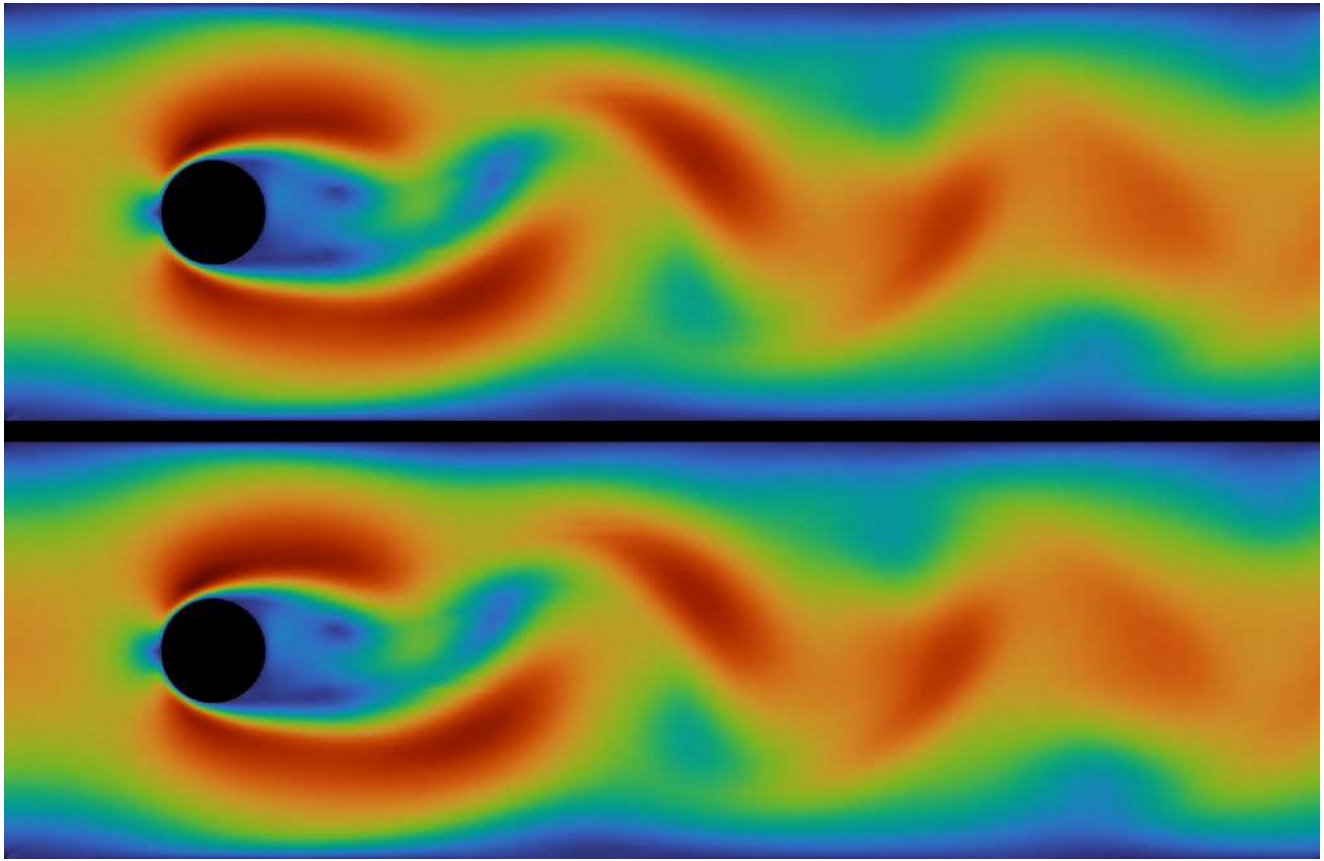
Closing the gap between High-Performance Computing (HPC) and artificial intelligence (AI)

September 15, 2023

Scientists and engineers face a number of technical hurdles to utilizing artificial intelligence (AI) techniques alongside high-performance scientific computing applications. Recently, a multidisciplinary team of researchers at Hewlett Packard Enterprise (HPE), the Institute of Aerodynamics and Gas Dynamics of the University of Stuttgart, and the High-Performance Computing Center Stuttgart (HLRS), created Relexi to overcome these technical challenges and provide the community with a large-scale simulation framework that can be tailored to their use cases. This solution was built on top of another existing open-source library developed at HPE called SmartSim.

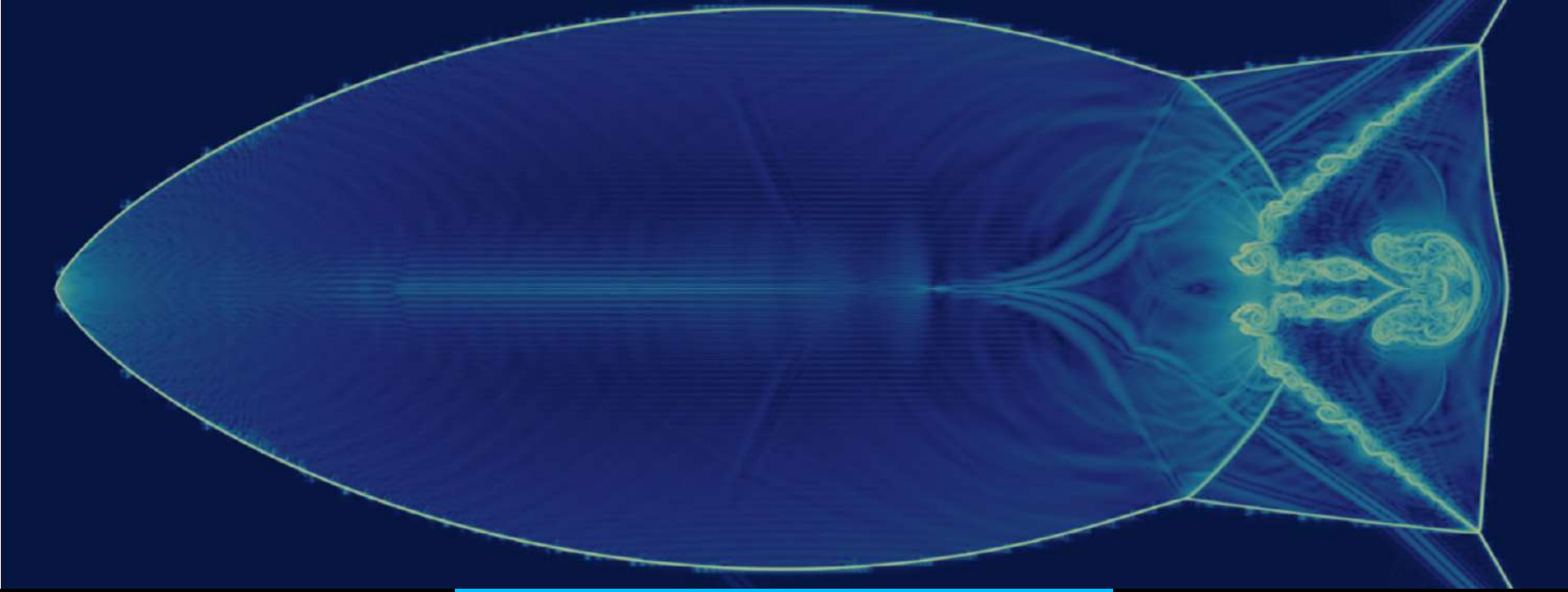


Kurz, Marius, et al. "Deep reinforcement learning for computational fluid dynamics on HPC systems." *Journal of Computational Science* 65 (2022): 101884.



- Jets suppress vortex shedding
- Drag reduction of 10%
- Stability issues with PPO
- Toy problem!

SL: an offline HPC problem.
CFD + RL = an online HPC problem.

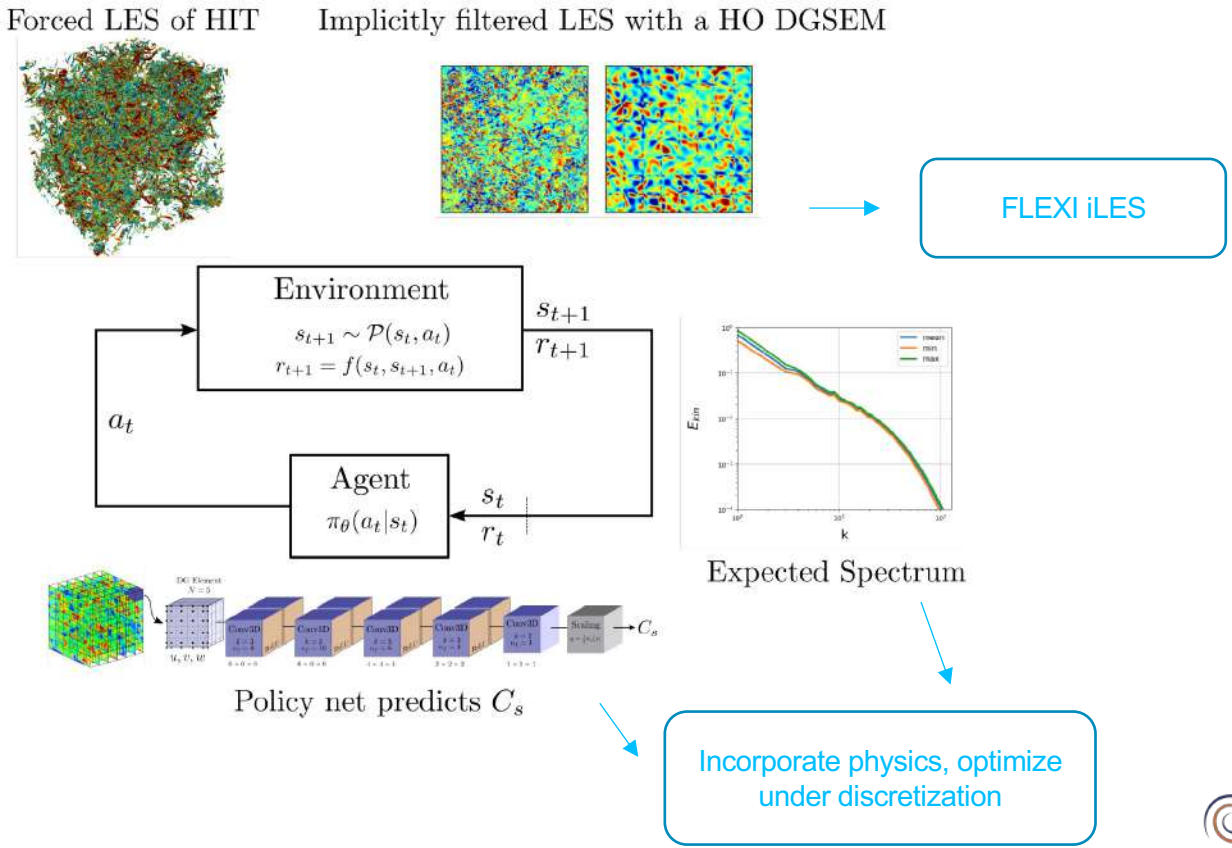


RL for discretization consistent LES closures

The problem

- The LES formulation describes a dynamical, chaotic system
- The computational SGS fluxes make it even harder
- The discrete operators can be non-linear for underresolved scales (by design)
- In this context, find discretization-consistent closure strategies through optimization
- Back to the Forced HIT
  - ALDM (genetic alg.), Filter-DG (Nelder-Mead), …

LES Closures as an MDP



- Reynolds stresses: What classical SGS models approximate

$$\overline{R(u)} = \frac{\partial \overline{u}}{\partial t} + \frac{1}{2}\frac{\delta \overline{u}\,\overline{u}}{\delta x} = C_1 + C_2 + \frac{1}{2}\frac{\partial}{\partial x}\underbrace{(\widehat{\overline{u}\,\overline{u}} - \overline{u}\,\overline{u})}_{C_3^*[\widehat{\cdot};\cdot]}; \quad \delta_x f = \partial_x \hat{f}$$

- **Complete,** discretization-consistent closure model for full SGS / SFS terms

Optimized eddy viscosity

Optimized dissipative properties

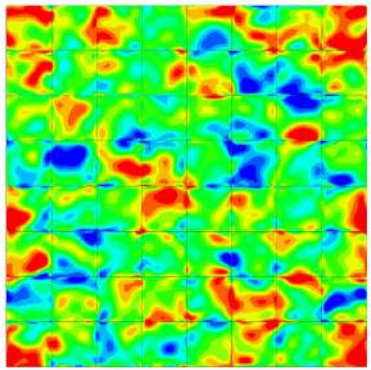
Formulate modelling strategy as an MDP, solve with RL

Things to note

- We find models for the complete closure terms C1...C3 (and others)
- We do not need to know C1…C3 a priori: We do not need DNS data, only a statistic
- Instead, “training” means running a number of LES simulations
- We optimize the solution of the overall PDE (i.e. the LES), not individual terms
- We discover new modelling schemes through RL: RL itself is not a model

Two modelling scenarios

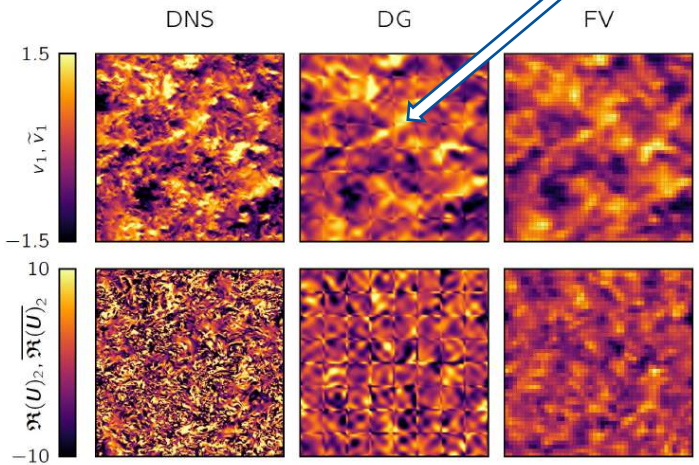
Implicit and explicit SGS modeling: Optimal Smagorinsky constant



SSM: For each grid element q_{rst} with $r,s,t \in [1, nElems] : C_s^q = C_s = const.$

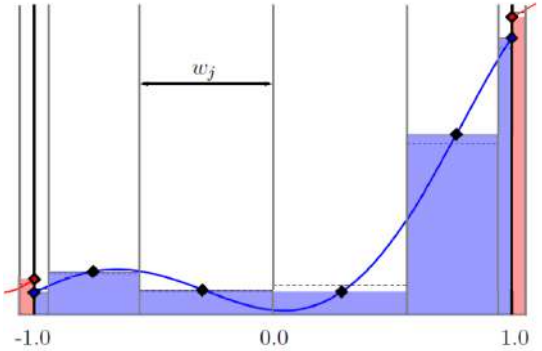
Cs0: For each grid element q_{rst} with $r,s,t \in [1, nElems] : C_s^q = C_s(t)$

Cs2: For each grid element q_{rst} with $r,s,t \in [1, nElems] : C_s^q = \sum_{i,j,k=0}^{2} c_i(t) L_i(x) L_j(y) L_k(z)$



Two modelling scenarios

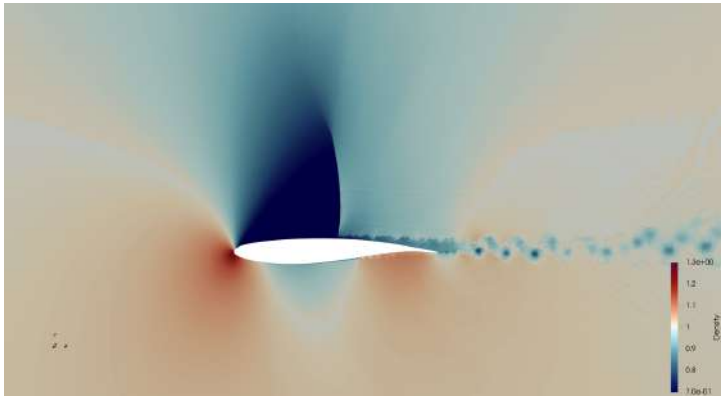
Implicit and explicit SGS modeling: Optimal Operator Blending

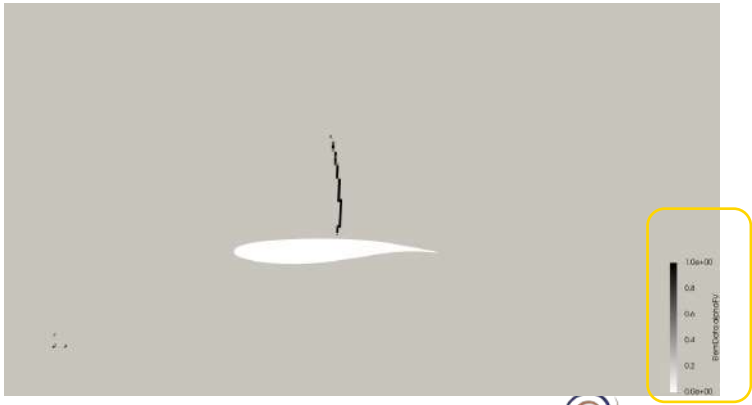


$$\dot{\underline{u}}^{BL} = \dot{\underline{u}}^{DG}\,(1-\alpha) + \dot{\underline{u}}^{FV}\,\alpha$$

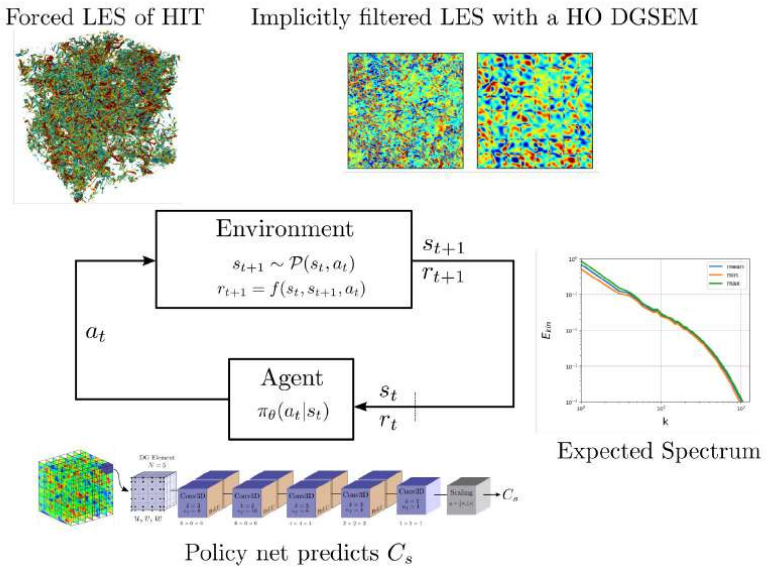
A0: For each grid element q_{rst} with $r,s,t \in [1, nElems] : \alpha_s^q = \alpha(t)$

A2: For each grid element q_{rst} with $r,s,t \in [1, nElems] : \alpha_s^q = \sum_{i,j,k=0}^{2} a_i(t) L_i(x) L_j(y) L_k(z)$



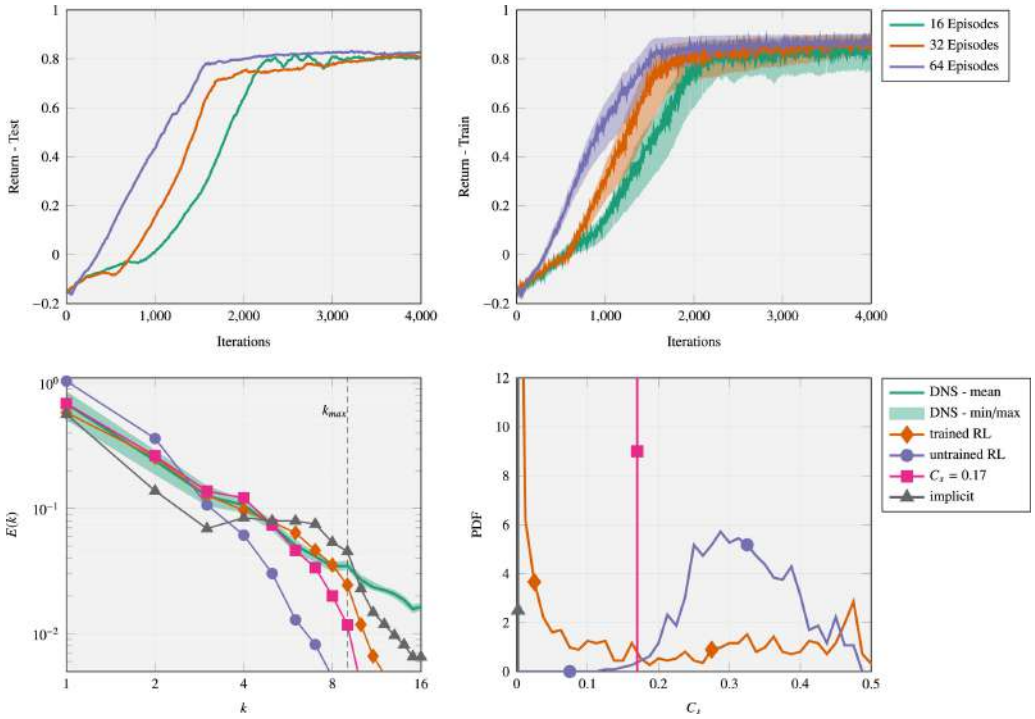


Optimization Details



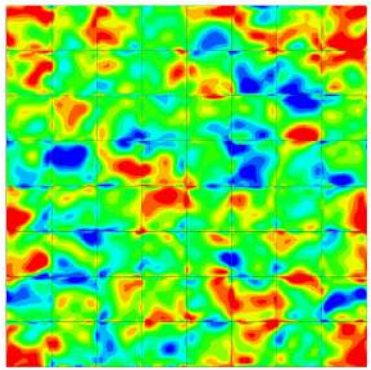
- Environment: Implicitly filtered HO DGSEM
- Reward / Goal: Spectrum up to cut-off
- Agent: Convolutional / Residual NN
- Policy: Choose Cs or Alpha
- No DNS data; optimization requires LES runs only
- Implement as hybrid CPU/GPU parallel code in SmartSim on HAWK-AI
- Training wall clock time: 24h

Example: Successful optimization for Cs0 case



Two modelling scenarios

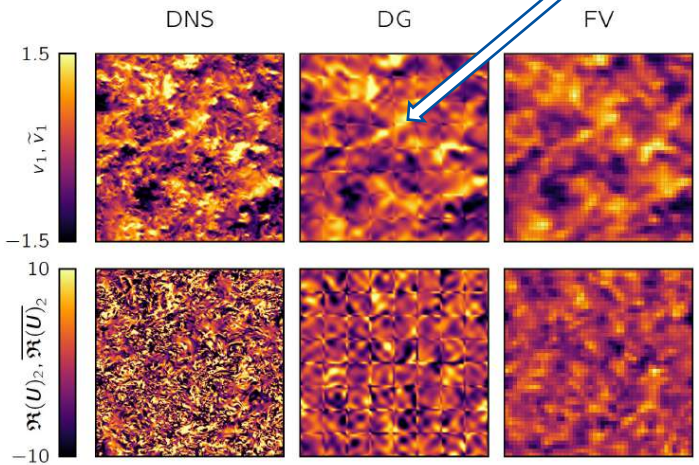
Implicit and explicit SGS modeling: Optimal Smagorinsky constant



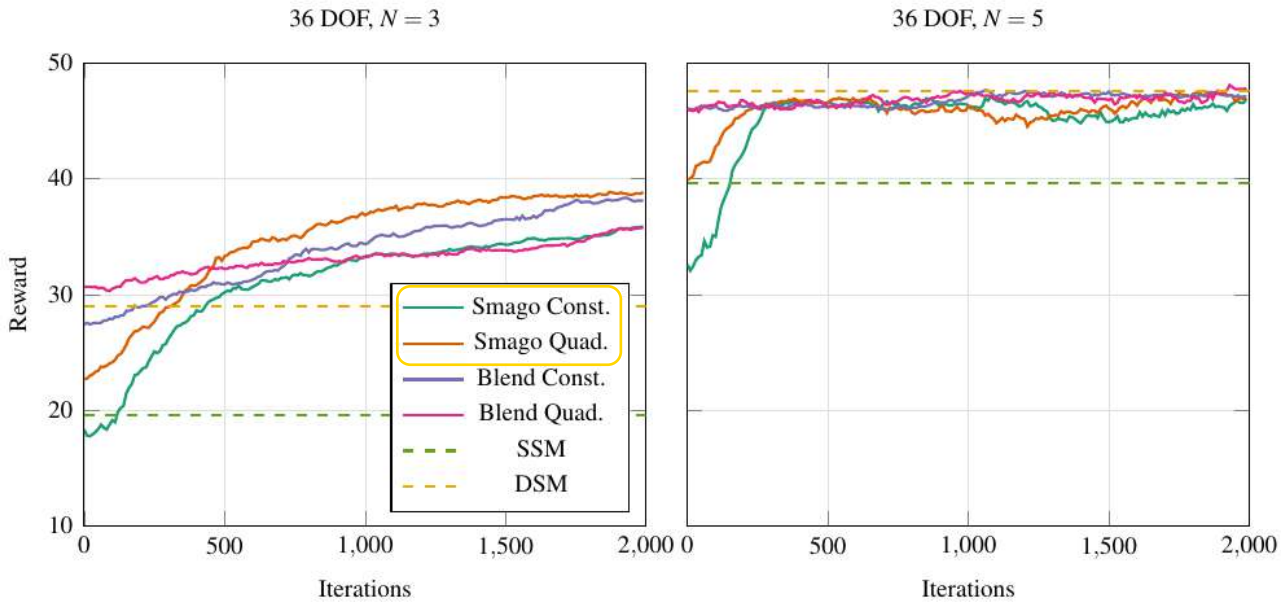
SSM: For each grid element q_{rst} with $r,s,t \in [1, nElems] : C_s^q = C_s = const.$

Cs0: For each grid element q_{rst} with $r,s,t \in [1, nElems] : C_s^q = C_s(t)$

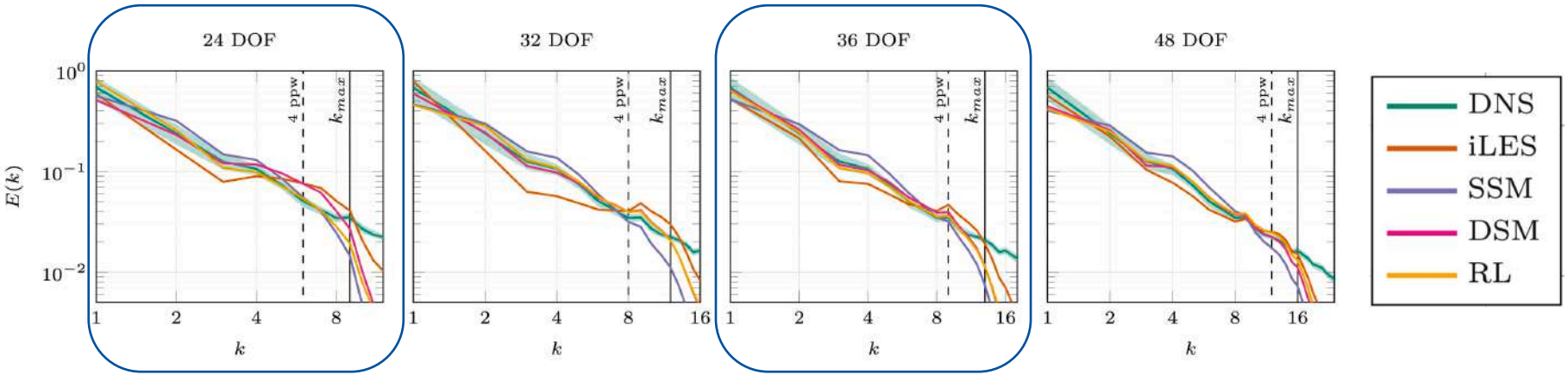
Cs2: For each grid element q_{rst} with $r,s,t \in [1, nElems] : C_s^q = \sum_{i,j,k=0}^{2} c_i(t) L_i(x) L_j(y) L_k(z)$



Results: Explicit Closure with optimized Cs

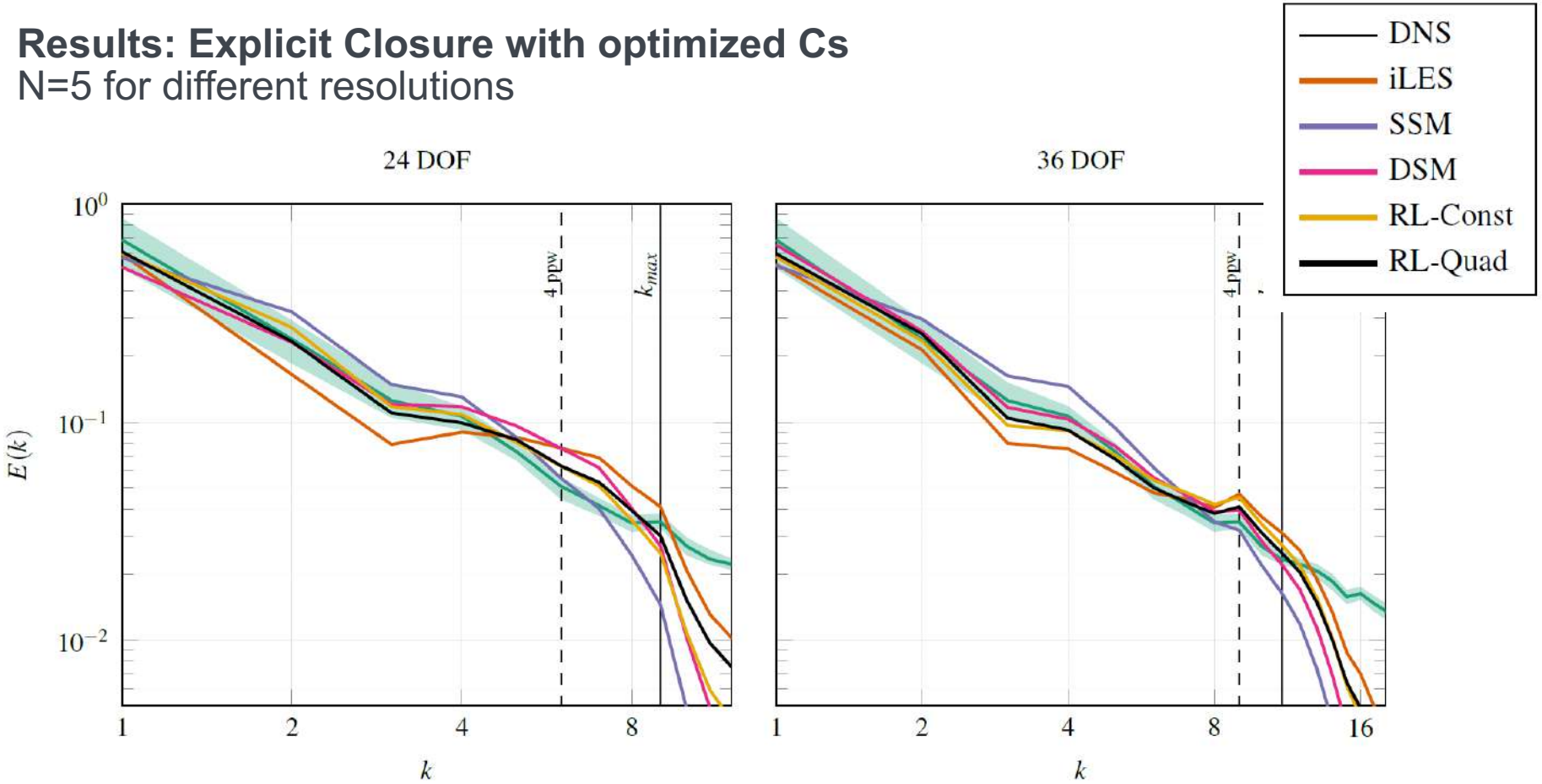


# Results: Explicit Closure with optimized Cs
N=5 for different resolutions



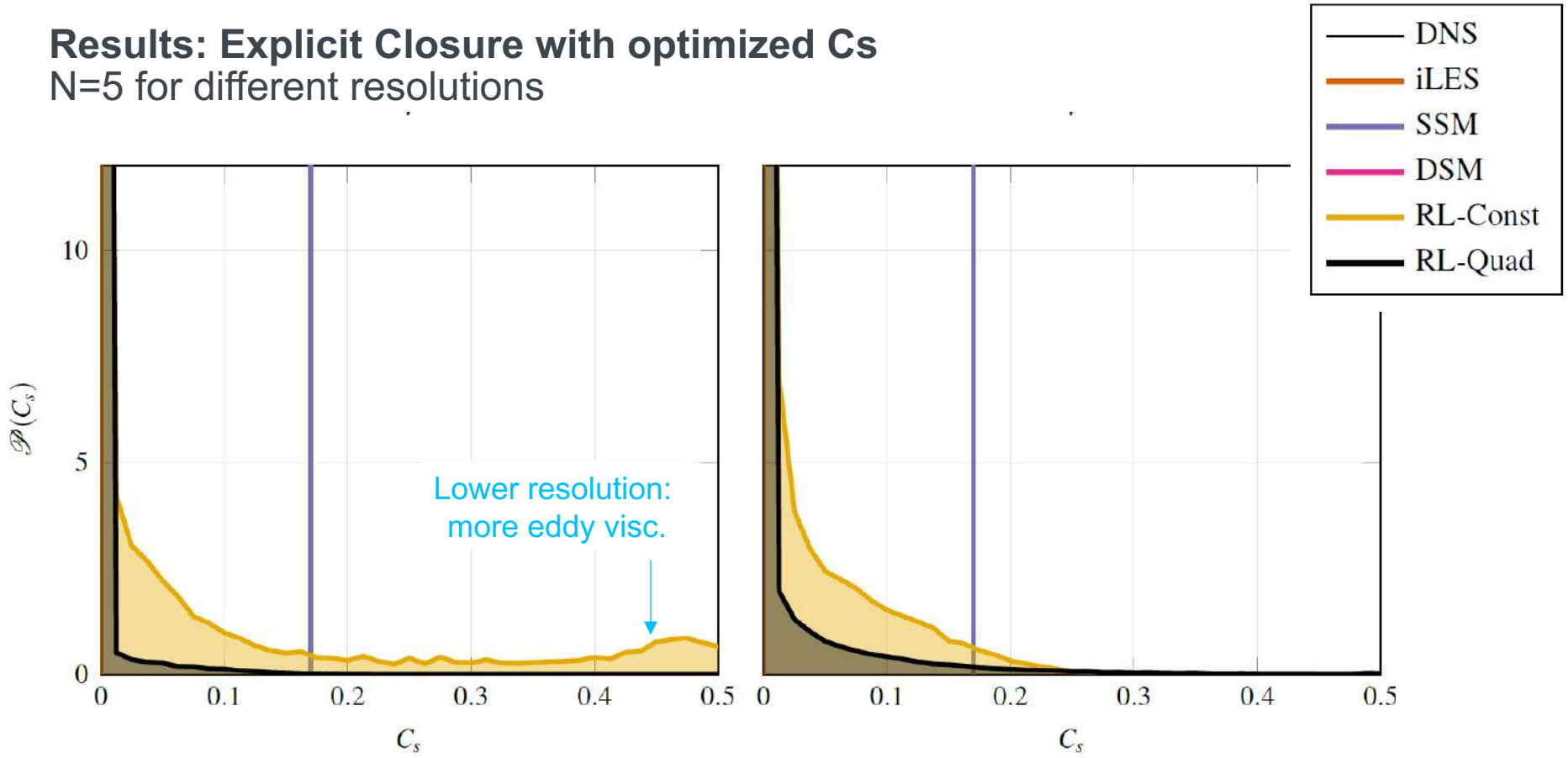
Results: Explicit Closure with optimized Cs

N=5 for different resolutions



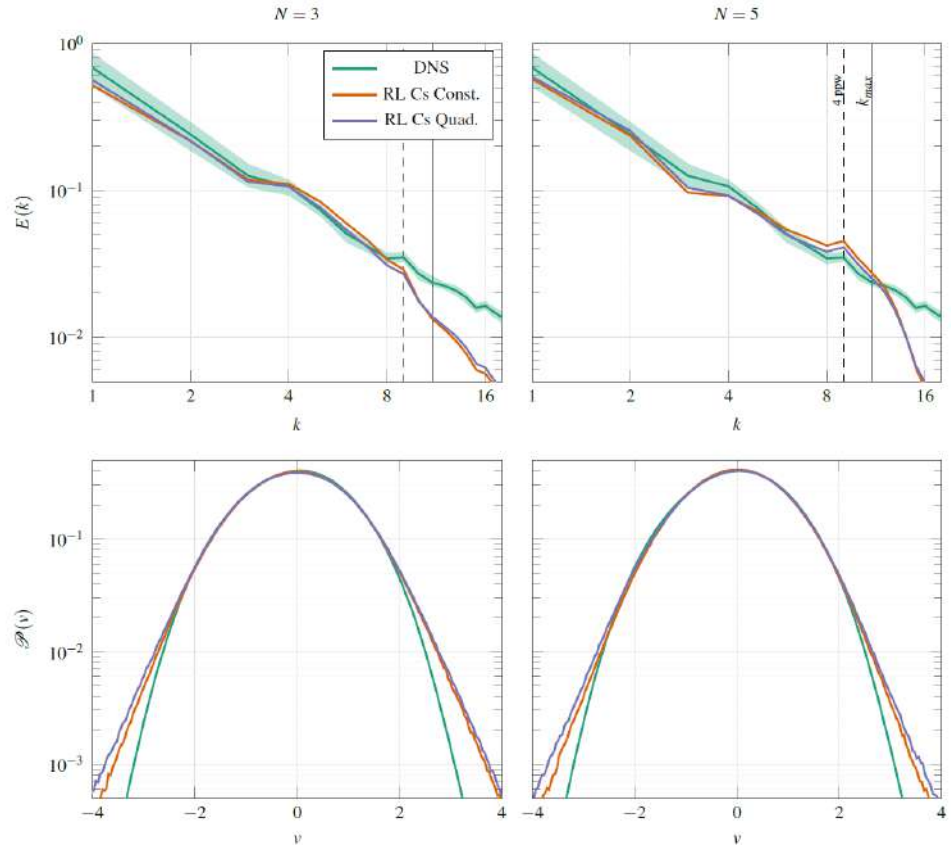
Results: Explicit Closure with optimized Cs

N=5 for different resolutions



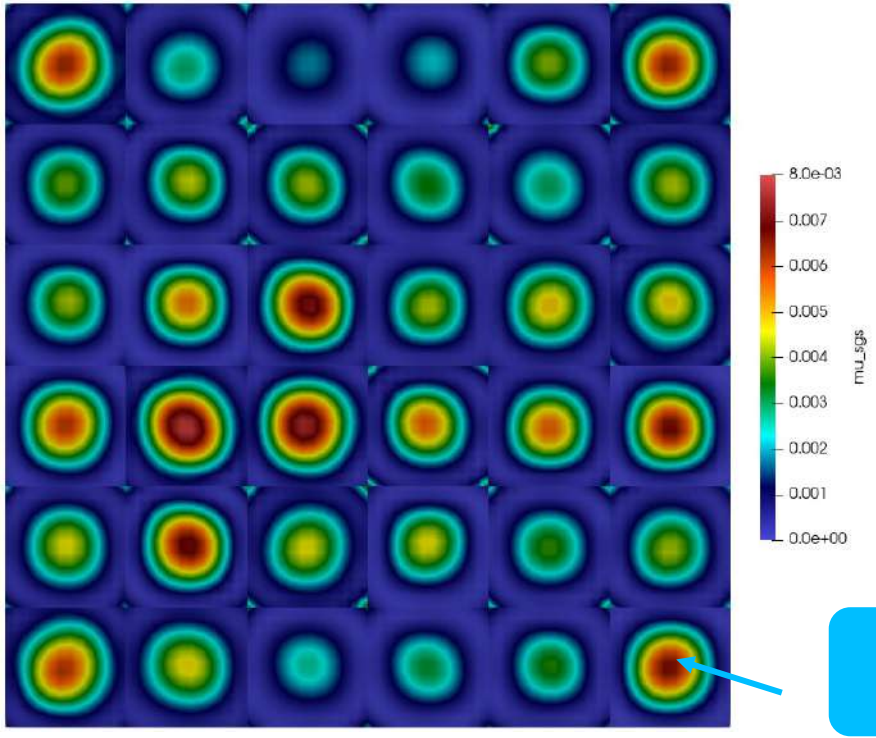
Results: Explicit Closure with optimized Cs

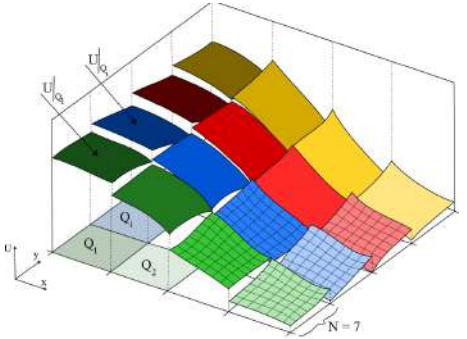
Different operators, same resolution

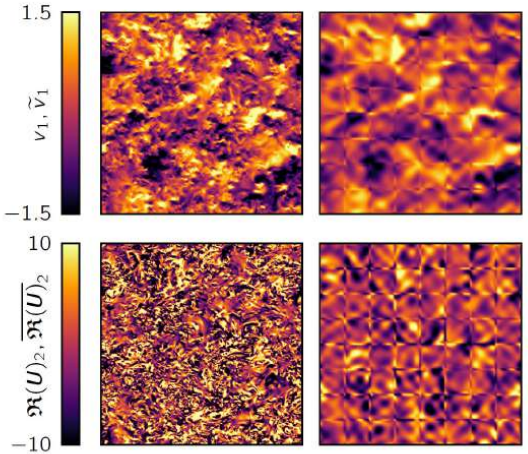


Results: Explicit Closure with optimized Cs

Mean Eddy viscosity field for Cs2





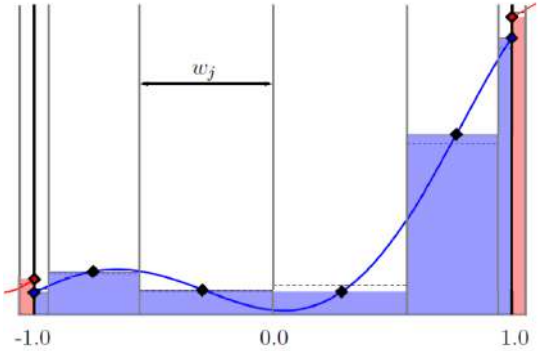


The discretization / induced filter footprint!

A model that accounts for the error of the discretization.

Two modelling scenarios

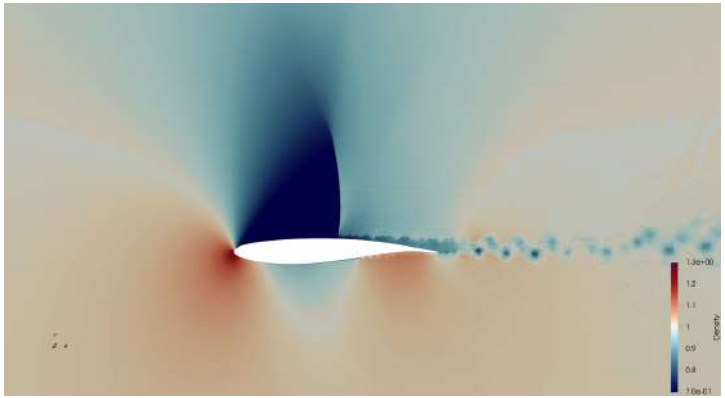
Implicit and explicit SGS modeling: Optimal Operator Blending

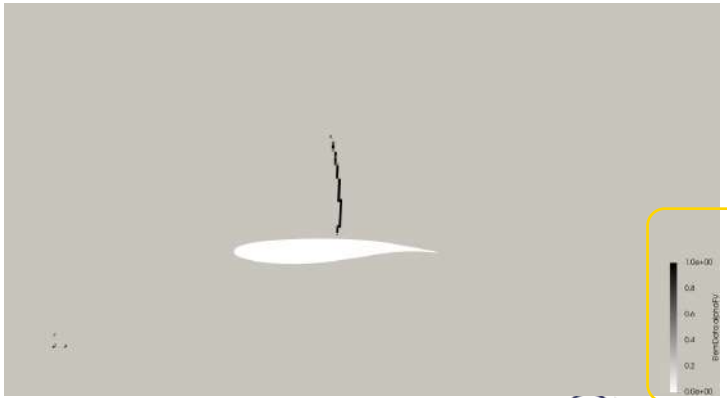


$$\underline{\dot{u}}^{BL} = \underline{\dot{u}}^{DG}\,(1-\alpha) + \underline{\dot{u}}^{FV}\,\alpha$$

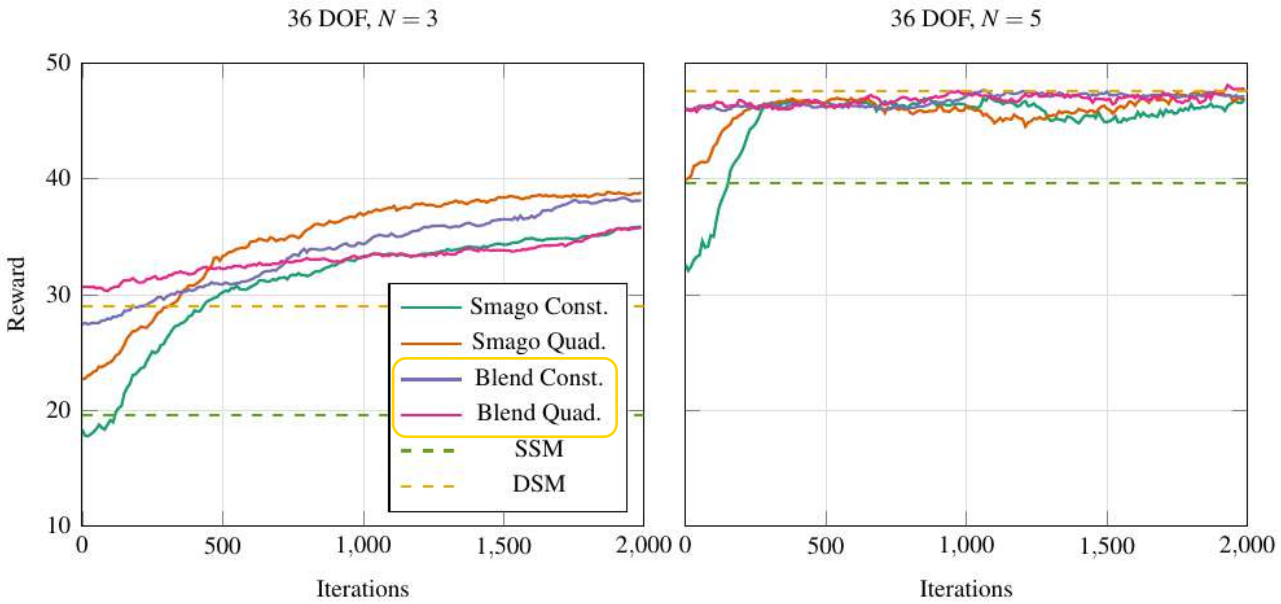
A0: For each grid element q_{rst} with $r,s,t \in [1, nElems] : \alpha_s^q = \alpha(t)$

A2: For each grid element q_{rst} with $r,s,t \in [1, nElems] : \alpha_s^q = \sum_{i,j,k=0}^{2} a_i(t) L_i(x) L_j(y) L_k(z)$



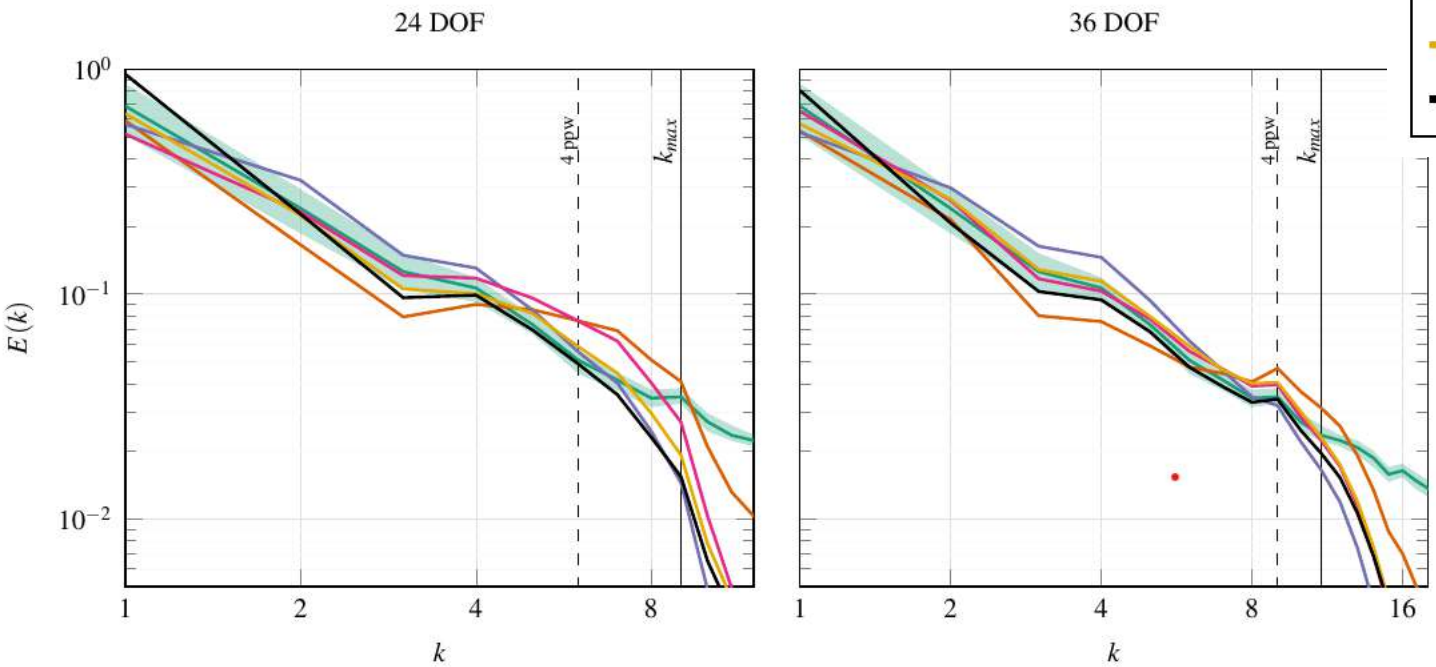


Results: Implicit Closure with optimized Blending



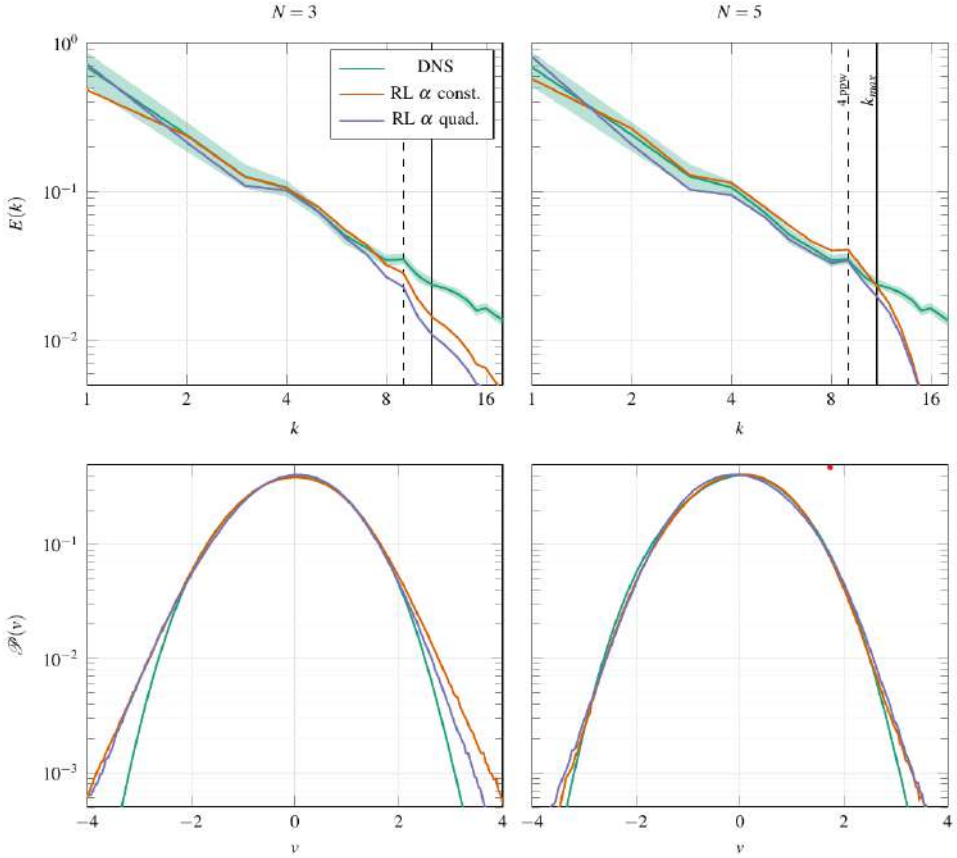
Results: Implicit Closure with optimized Blending

N=5 for different resolutions

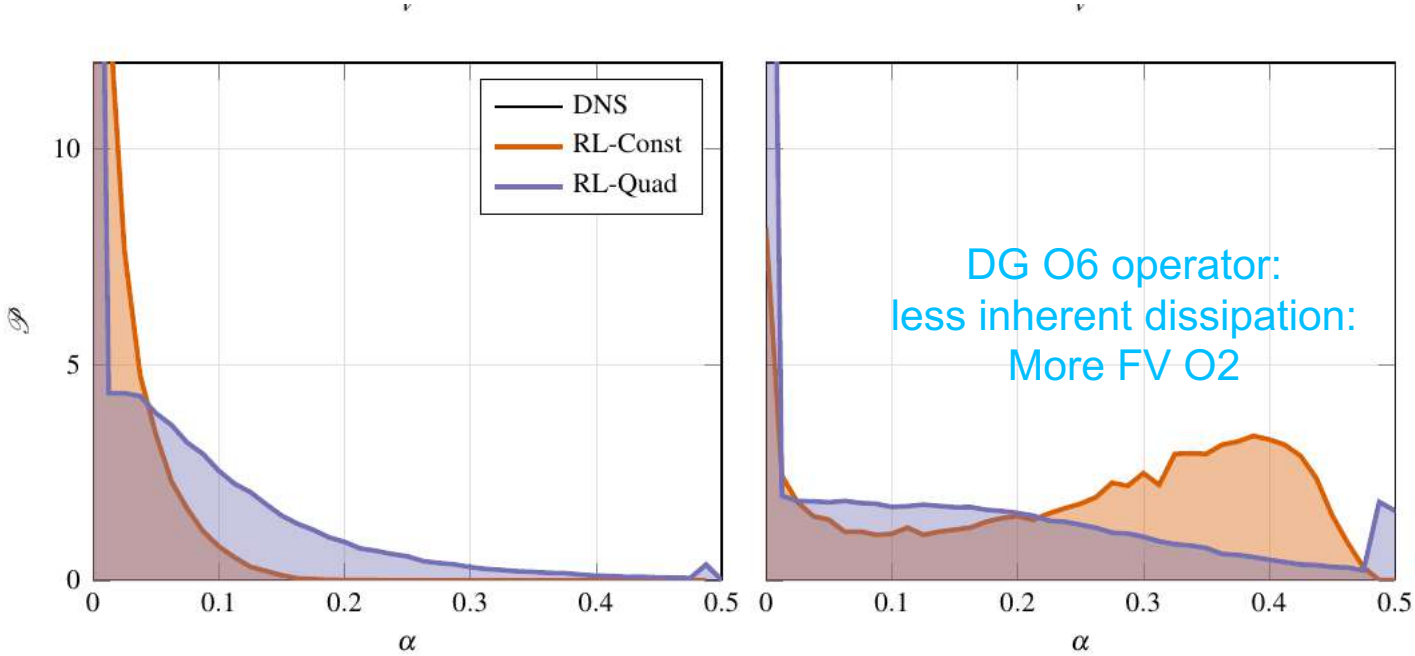


Results: Implicit Closure with optimized Blending

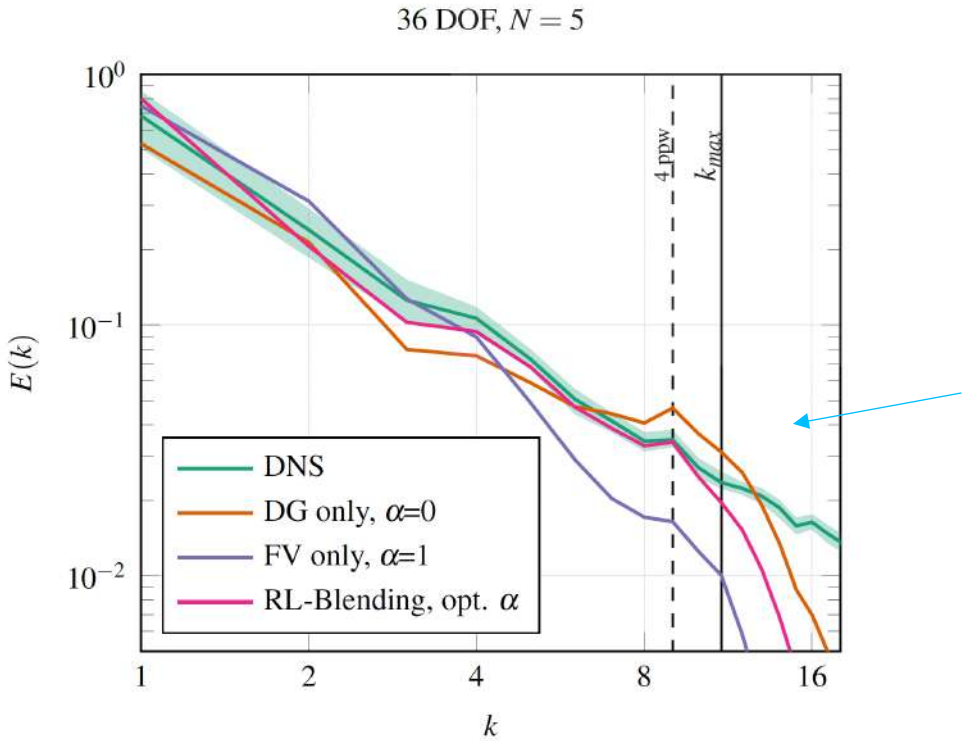
Different operators, same resolution



Results: Implicit Closure with optimized Blending
Different operators, same resolution



Results: Implicit Closure with optimized Blending



DG MILES: implicit LES model through non-linear combination of HO and TVD scheme

An “intelligent” LES solver?

Summary

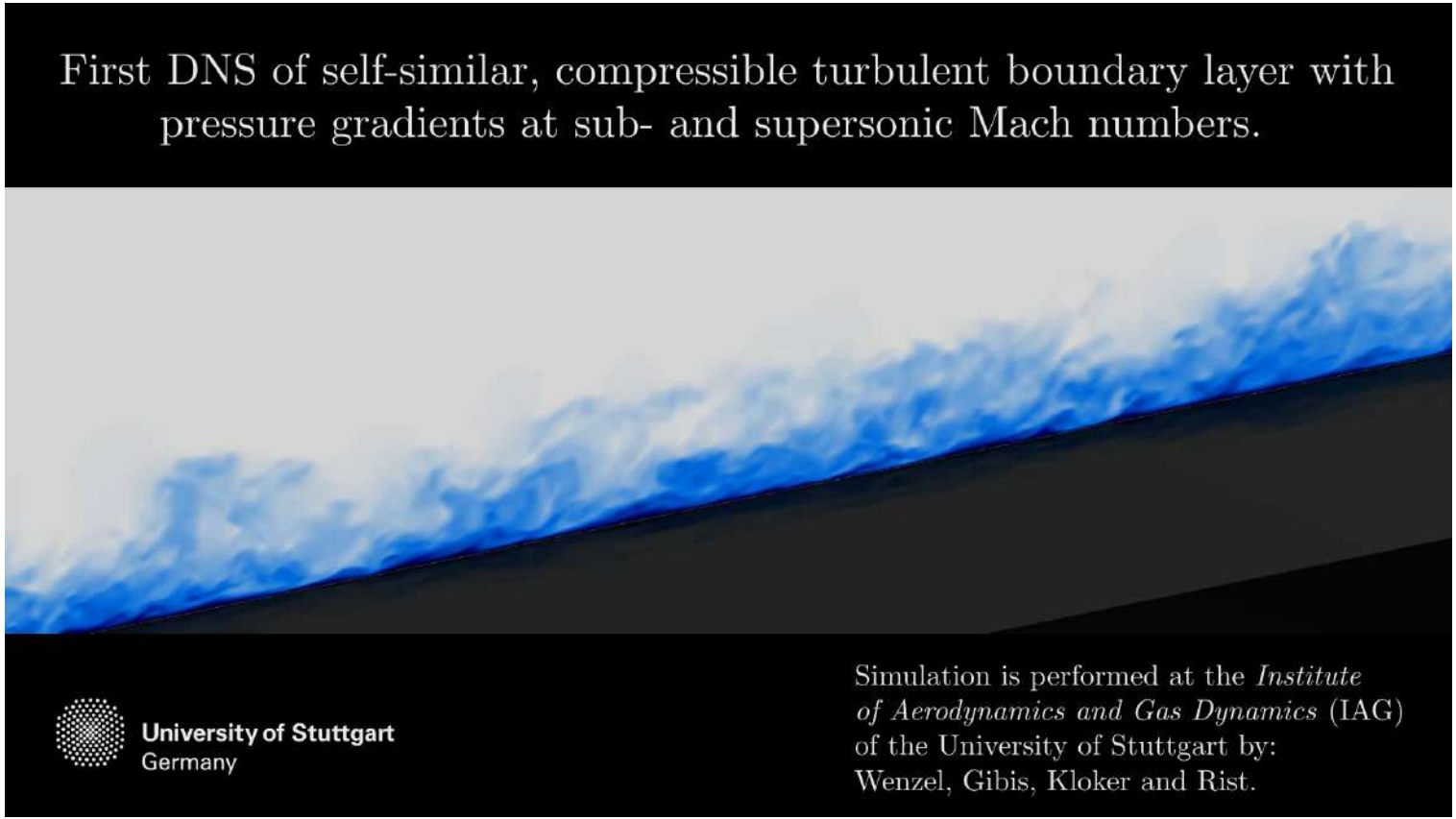
Food for thought

- Multiscale problems require multi-X numerical schemes
- SL works great if one can define good data a priori
- The classical LES formulation omits the challenges of “real life” LES: C1…C3
- Reinforcement learning with a PDE solver in the loop: Versatile, inclusive optimization environment for flow control and modelling
  - RL does not need fine scale data (like DNS), but statistics. Training happens on the coarse (LES) level: Huge computational savings
  - RL allows to optimize and discover new closures, it is not a model itself: All the knowledge about a “good” model is still applicable
  - RL is an alternative to adjoints or differentiable solvers: It approximates gradients of the cost function through sampling
  - RL is ML: it has hyperparameters and we lack theory in parts

Our publications on data-driven CFD

- Journal of Computational Physics 398, 108910
- GAMM-Mitteilungen 44
- Journal of Computational Physics 313, 1-12
- Journal of Computational Physics, 423, 109824
- Electronic Transactions on Numerical Analysis, 56, 117–137
- International Journal of Heat and Fluid Flow 99, 109094
- Journal of Theoretical and Computational Acoustics 27, 1850044
- Journal of Computational Physics 441, 110475
- SIAM Journal on Scientific Computing 42 (4), B1067–B1091
- Journal of Computational Science 65, 101884
- Journal of Computational Physics 408, 109303
- Software Impacts 14, 100422
- Computers & Fluids 228, 105039
- Wear 508, 204476
- Journal of Open Source Software 8 (82), 4683
- PAMM 23 (1), e202200207
- Toward Discretization-Consistent Closure Schemes for Large Eddy Simulation Using Reinforcement Learning **arXiv:2309.06260v1**

What do we need LES for?



NRG

University of Stuttgart
Institute of Aerodynamics and Gas Dynamics

NRG

Thank you!

Andrea Beck

e-mail beck@iag.uni-stuttgart.de
phone +49 711 685-60218
www.iag.uni-stuttgart.de
numericsresearchgroup.org

- Kurz, Marius, Philipp Offenhäuser, and Andrea Beck. "Deep reinforcement learning for turbulence modeling in large eddy simulations." *International Journal of Heat and Fluid Flow* 99 (2023): 109094.
- Kurz, Marius, et al. "Relexi—A scalable open source reinforcement learning framework for high-performance computing." *Software Impacts* 14 (2022): 100422.
- Mossier, Pascal, Andrea Beck, and Claus-Dieter Munz. "A p-adaptive discontinuous Galerkin method with hp-shock capturing." *Journal of Scientific Computing* 91.1 (2022): 4.
- Schwarz, Anna, et al. "A Reinforcement Learning Based Slope Limiter for Second-Order Finite-Volume Schemes." *Proceedings in Applied Mathematics and Mechanics* 7 (2022): 10.
- Kopper, Patrick, et al. "Boundary-Layer Dynamics in Wall-Resolved LES Across Multiple Turbine Stages." *AIAA Journal* 59.12 (2021): 5225-5237.
- Dürrwächter, Jakob, et al. "A high-order stochastic Galerkin code for the compressible Euler and Navier-Stokes equations." *Computers & Fluids* 228 (2021): 105039.
- Beck, Andrea, and Marius Kurz. "A perspective on machine learning methods in turbulence modeling." *GAMM-Mitteilungen* 44.1 (2021): e202100002.

NRG