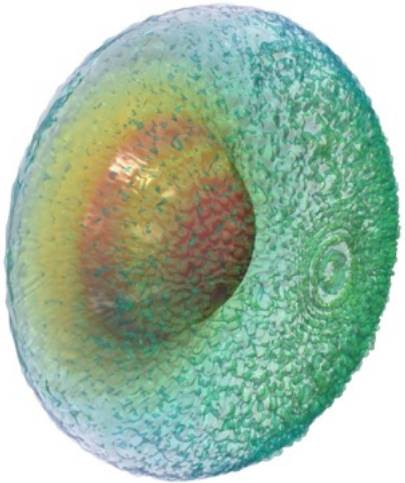
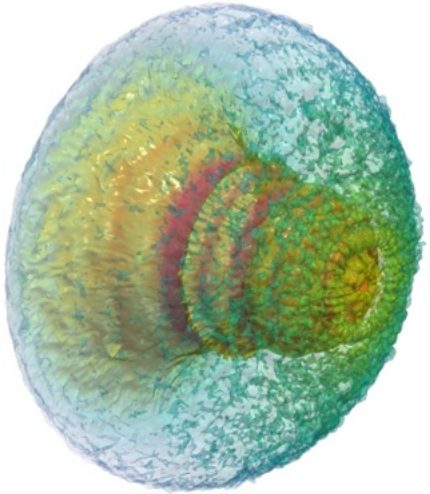
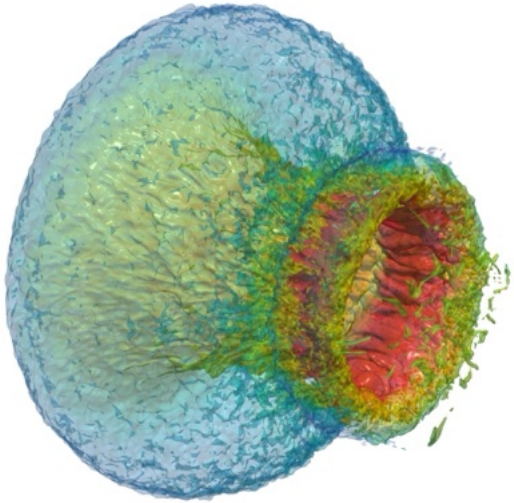
Toward Machine-learned Discretizations & Differentiable CFD for Compressible Two-phase Flows





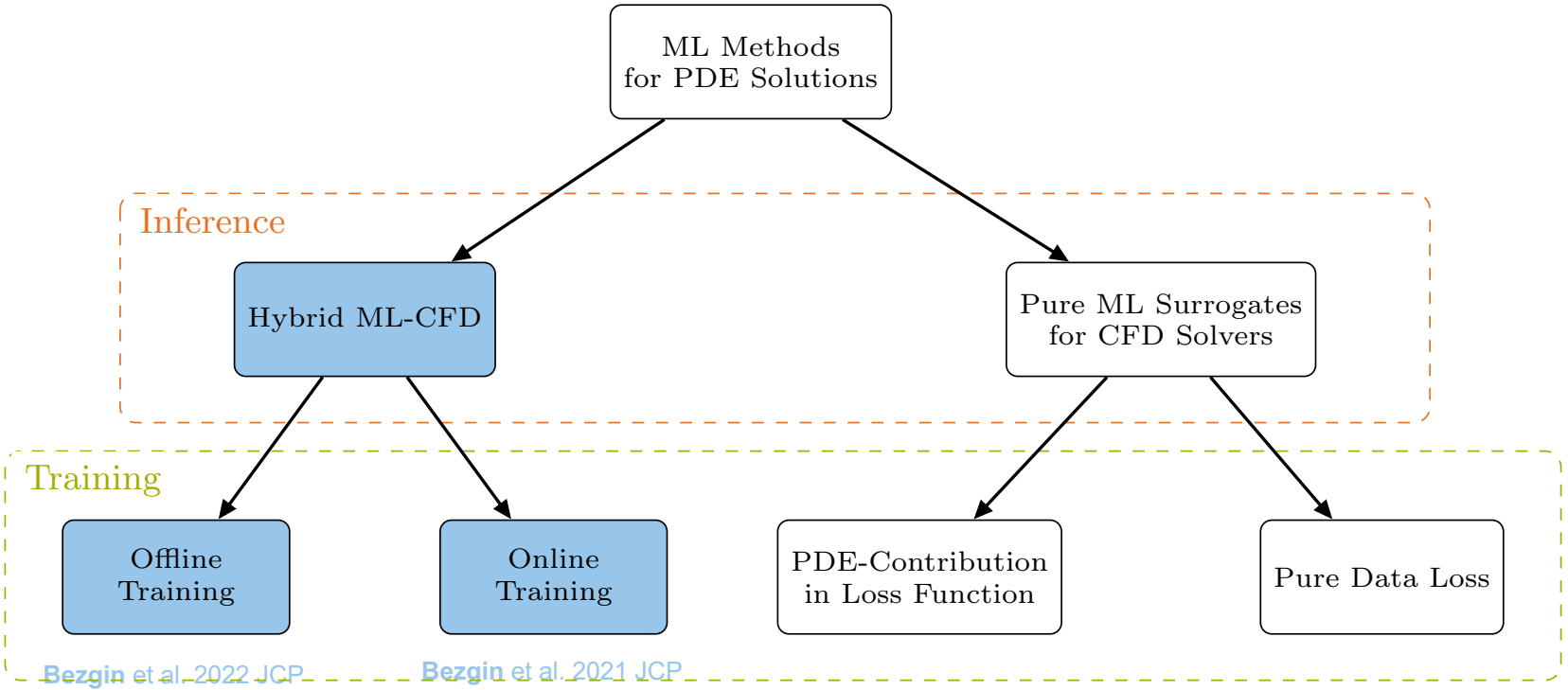


Deniz A. Bezgin

ERCOFTAC – 20th Da Vinci Competition

October 9, 2025

ML methods for solving PDEs



Bezgin et al. 2022 JCP
B., **Bezgin**, A. 2022 JCP

Bezgin et al. 2021 JCP
Bezgin et al. 2023 CPC
Bezgin et al. 2025 CPC
Bezgin et al. 2025 JCP
B., **Bezgin** et al. 2025 PRF

Online Training of Non-linear Discretizations

Online training of non-linear discretizations

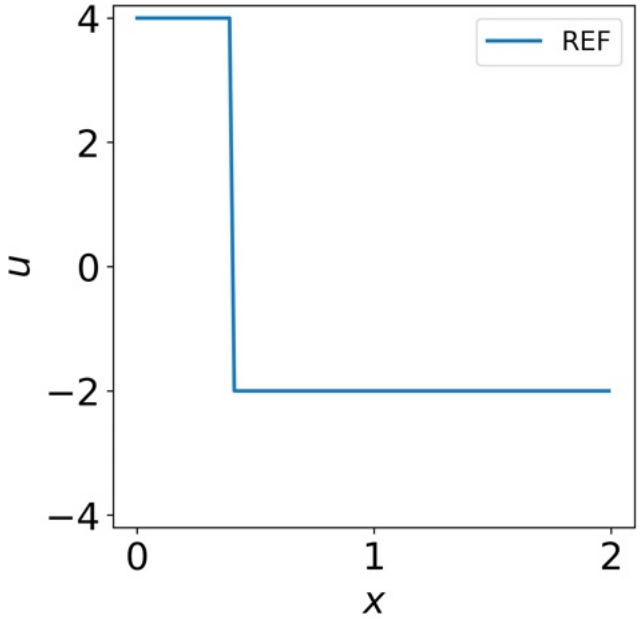
Model equation: Scalar cubic conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial u^3}{\partial x} = 0$$

$$\frac{\partial u}{\partial t} + \frac{\partial u^3}{\partial x} = \epsilon \frac{\partial^2 u}{\partial x^2} + \delta\epsilon^2 \frac{\partial^3 u}{\partial x^3}; \epsilon \to 0$$

Admits nonclassical (undercompressive) shocks.

Conventional schemes fail to capture nonclassical shocks due to a **mismatch of the truncation error and the small-scale terms in the PDE**.



Can we use online training to match the truncation error of a data-driven discretization with the correct small-scale terms in the PDE?

Data-driven discretization

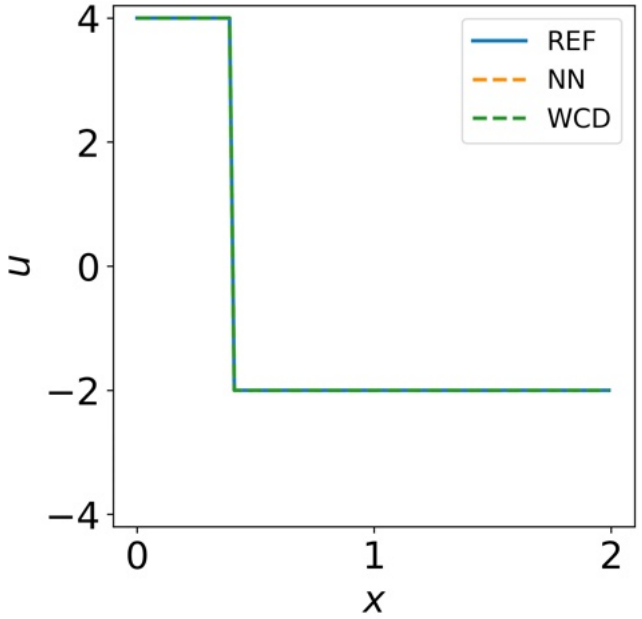
$$\tilde{F}_{N,i+1/2} = F\left(\tilde{u}^{-}_{i+1/2}\right) - \tilde{\delta}^{\mathrm{NN}}_{i+1/2}(c\Delta x)^2 u_{xx}\left(x_{i+1/2}\right)$$

$$\tilde{u}^{-}_{i+1/2} = \sum_{k=1}^{3}\sum_{r=0}^{k-1} \omega^{-,\mathrm{NN}}_{k,r} p^{-}_{k,r}$$

$$u_{xx}\left(x_{i+1/2}\right) = \frac{\bar{u}_{i-1} - \bar{u}_i - \bar{u}_{i+1} + \bar{u}_{i+2}}{2\Delta x^2}$$

Training

- Online training (end-to-end optimization) using an **automatically differentiable 1D solver**
- Analytical solutions of Riemann problems
- Training horizon: $n_{int} = 7$ integration steps



Can we use online training to match the truncation error of a data-driven discretization with the correct small-scale terms in the PDE?

Data-driven discretization

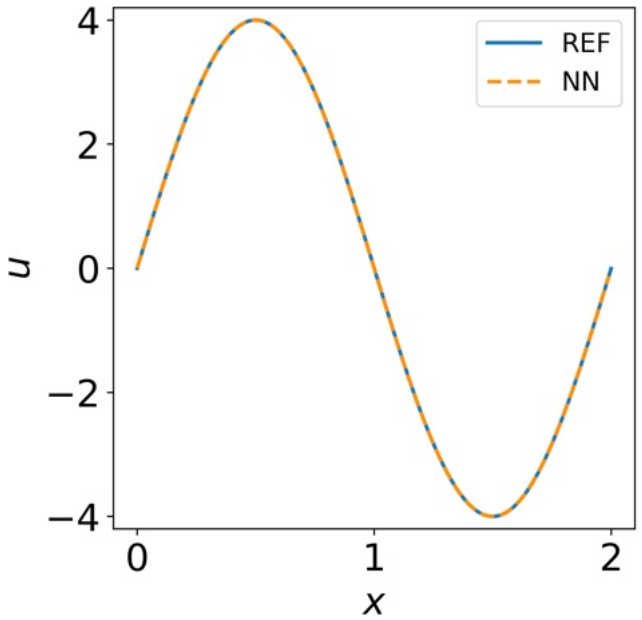
$$\tilde{F}_{N,i+1/2} = F\left(\tilde{u}^{-}_{i+1/2}\right) - \tilde{\delta}^{\mathrm{NN}}_{i+1/2}(c\Delta x)^2 u_{xx}\left(x_{i+1/2}\right)$$

$$\tilde{u}^{-}_{i+1/2} = \sum_{k=1}^{3}\sum_{r=0}^{k-1} \omega^{-,\mathrm{NN}}_{k,r} p^{-}_{k,r}$$

$$u_{xx}\left(x_{i+1/2}\right) = \frac{\bar{u}_{i-1} - \bar{u}_i - \bar{u}_{i+1} + \bar{u}_{i+2}}{2\Delta x^2}$$

Training

- Online training (end-to-end optimization) using an **automatically differentiable 1D solver**
- Analytical solutions of Riemann problems
- Training horizon: $n_{int} = 7$ integration steps



Can we use online training to match the truncation error of a data-driven discretization with the correct small-scale terms in the PDE?

Data-driven discretization

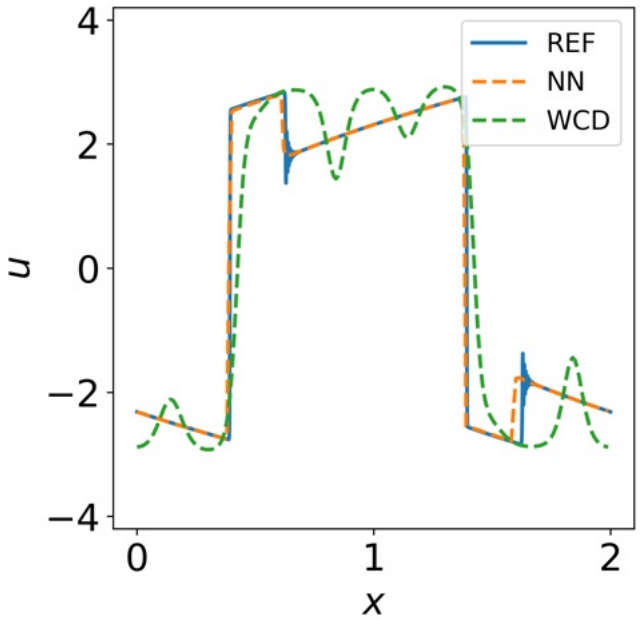
$$\tilde{F}_{N,i+1/2} = F\left(\tilde{u}^{-}_{i+1/2}\right) - \tilde{\delta}^{\mathrm{NN}}_{i+1/2}(c\Delta x)^2 u_{xx}\left(x_{i+1/2}\right)$$

$$\tilde{u}^{-}_{i+1/2} = \sum_{k=1}^{3}\sum_{r=0}^{k-1} \omega^{-,\mathrm{NN}}_{k,r} p^{-}_{k,r}$$

$$u_{xx}\left(x_{i+1/2}\right) = \frac{\bar{u}_{i-1} - \bar{u}_i - \bar{u}_{i+1} + \bar{u}_{i+2}}{2\Delta x^2}$$

Training

- Online training (end-to-end optimization) using an **automatically differentiable 1D solver**
- Analytical solutions of Riemann problems
- Training horizon: $n_{int} = 7$ integration steps

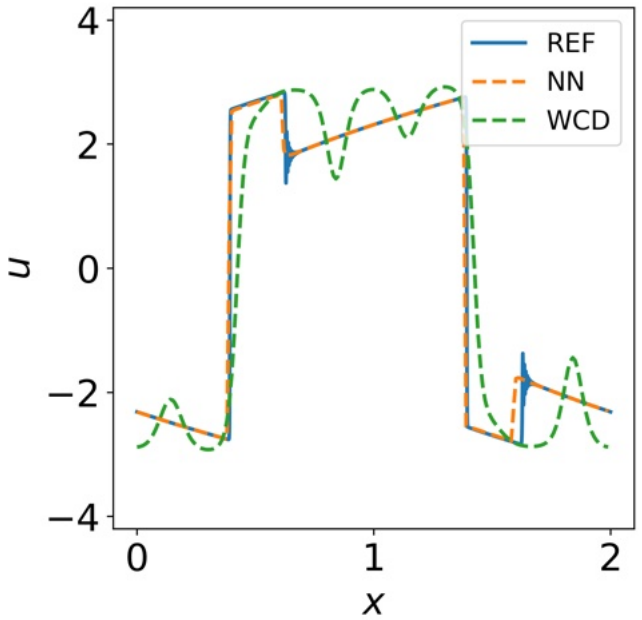


Can we use online training to match the truncation error of a data-driven discretization with the correct small-scale terms in the PDE?

Online Training

- Neural network model observes PDE dynamics
- Discretized PDE → **Inductive bias**
- Generalization to out-of-sample configurations
- Longer training horizon generally enhance stability

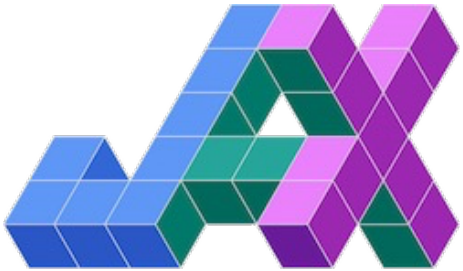
→ Fully-differentiable CFD solver for compressible single- & two-phase flows!



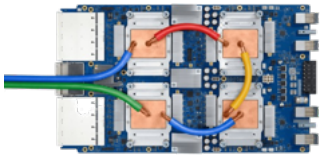
JAX-Fluids: Differentiable CFD for Compressible Single- and Two-phase Flows

JAX-Fluids: Differentiable CFD for Compressible Flows

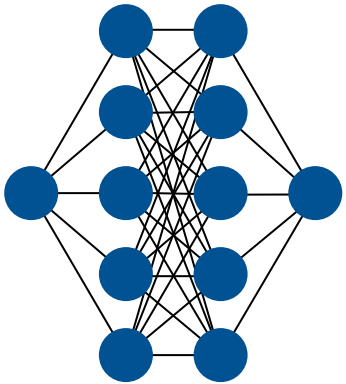
High-level language for **rapid prototyping** & seamless integration with **ML pipelines**



HPC capability → runs on different XLA accelerators out of the box



Automatic differentiation gradients for the solution of **inverse problems**

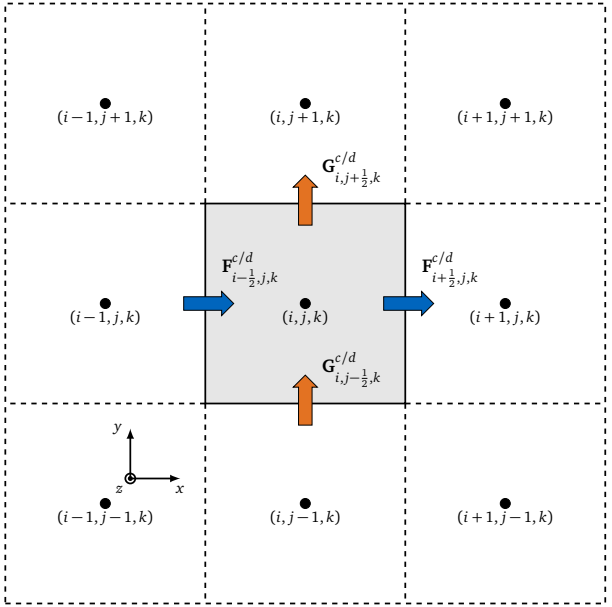


$\partial \mathcal{L} / \partial \theta$

JAX-Fluids is a Finite-Volume Solver

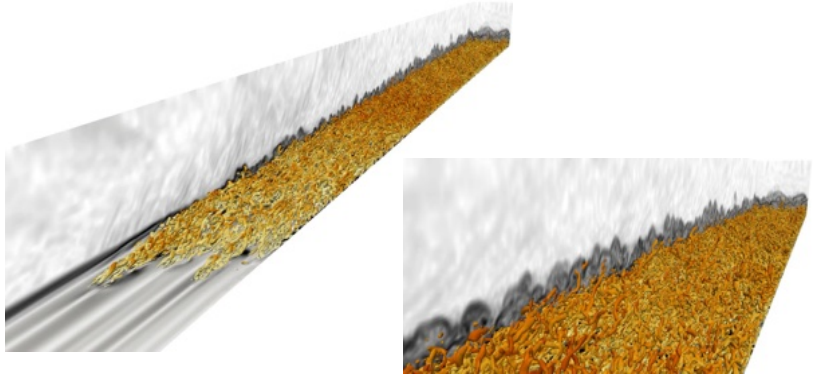
- High-order Godunov-type **finite-volume solver**
- WENO/TENO **shock-capturing** + Approximate Riemann solvers
- Cartesian meshes + Cut-cell based **Immersed boundary method**
- Two-phase models: Sharp interface **level-set model** & five-equation **diffuse-interface model**
- **JAX-based parallelization** → computation of automatic differentiation gradients across GPUs

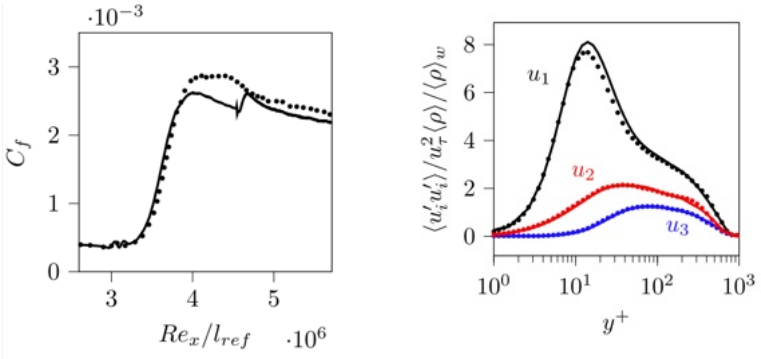
$$\partial_t \bar{\boldsymbol{U}}_{i,j} = -\frac{1}{\Delta x}\left[\left(\boldsymbol{F}^c_{i+1/2,j} - \boldsymbol{F}^d_{i+1/2,j}\right) - \left(\boldsymbol{F}^c_{i-1/2,j} - \boldsymbol{F}^d_{i-1/2,j}\right)\right] - \frac{1}{\Delta y}\left[\left(\boldsymbol{G}^c_{i,j+1/2} - \boldsymbol{G}^d_{i,j+1/2}\right) - \left(\boldsymbol{G}^c_{i,j-1/2} - \boldsymbol{G}^d_{i,j-1/2}\right)\right] + \bar{\boldsymbol{S}}_{i,j}$$



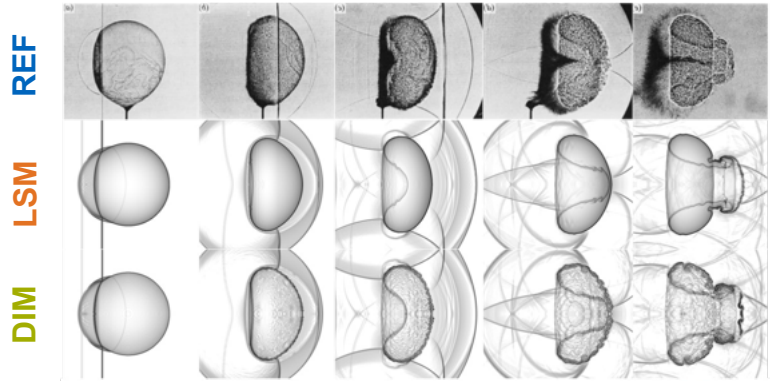
Verification on Canonical Single- & Two-phase Flows

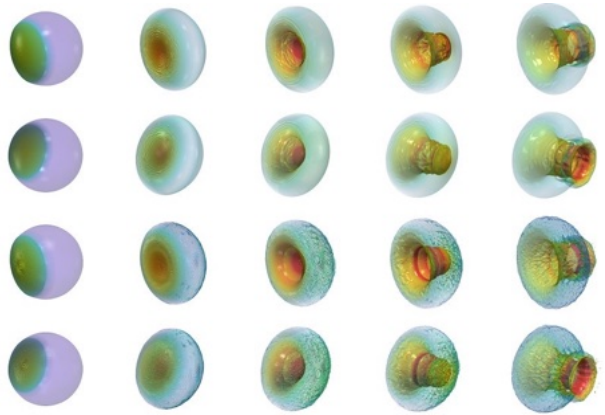
Compressible boundary layer Pirozolli POF 2004





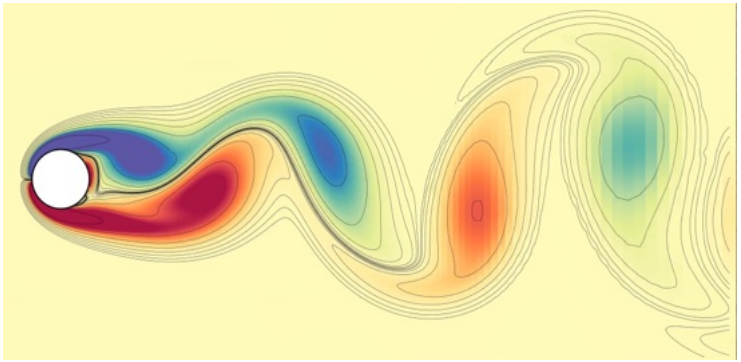
Air-helium shock bubble Haas et al. JFM 1987





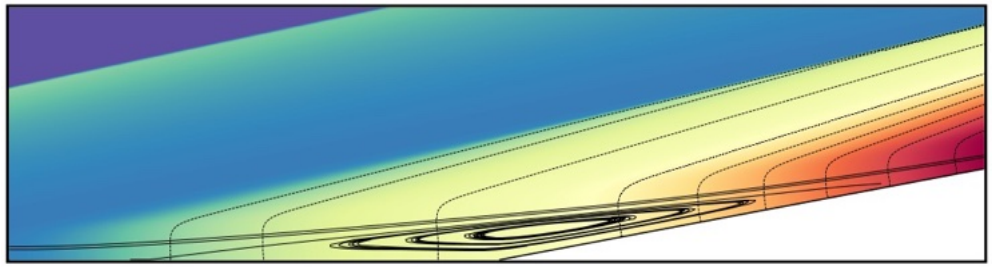
Verification of the Immersed Boundary Method

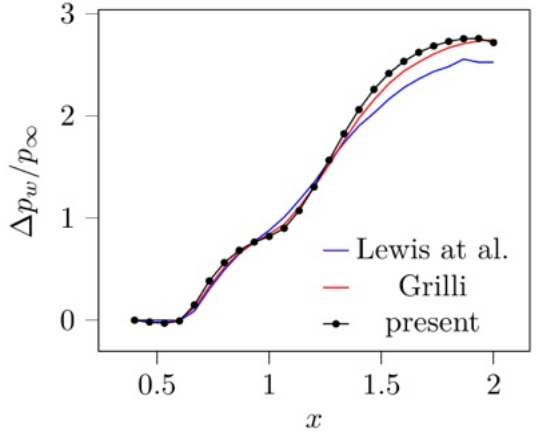
Laminar flow around a cylinder at Re = 100



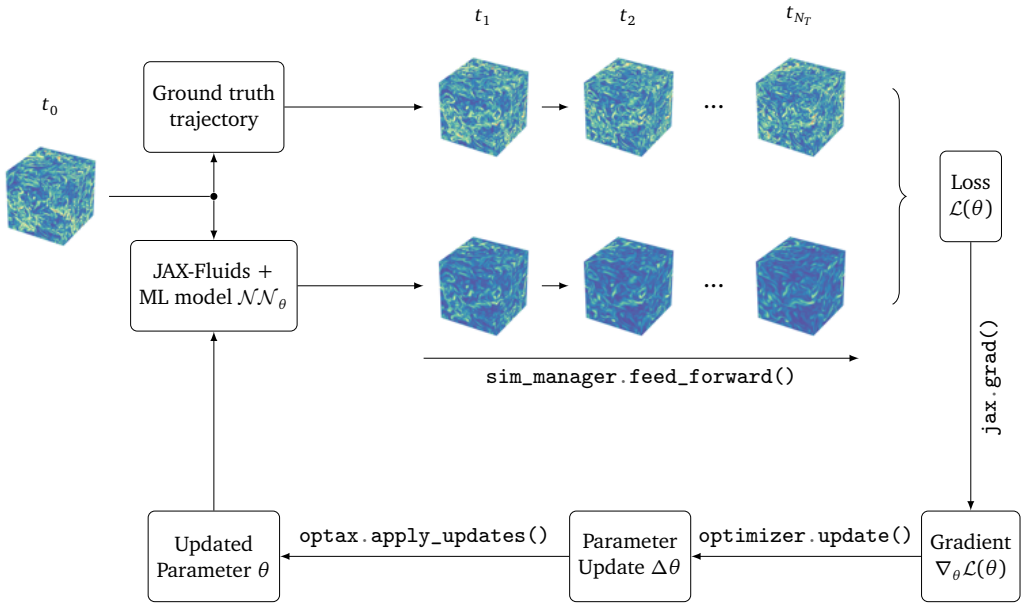
	St	$\overline{C}_D$	C_L^{max}
Linnick et al. [12]	0.166	1.34	0.33
Vanna et al.[9]	0.160	1.32	0.22
Grilli [5]	0.166	1.38	0.33
Meyer at al. [2]	0.165	1.26	0.34
present	0.162	1.37	0.34

Compression corner at Mach = 6.06





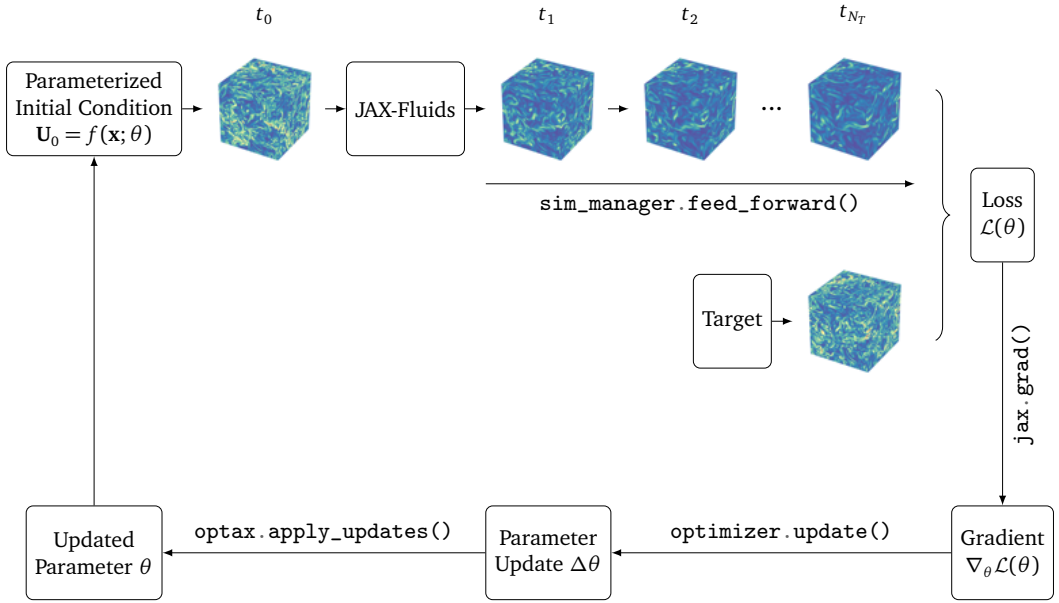
Differentiable CFD can be used for online training of numerical models & discretizations…



- Data-driven model is **embedded into the CFD solver**
- Model is **applied recursively** over the course of a simulation and is optimized **end-to-end**
- Example: Online training of a turbulence model

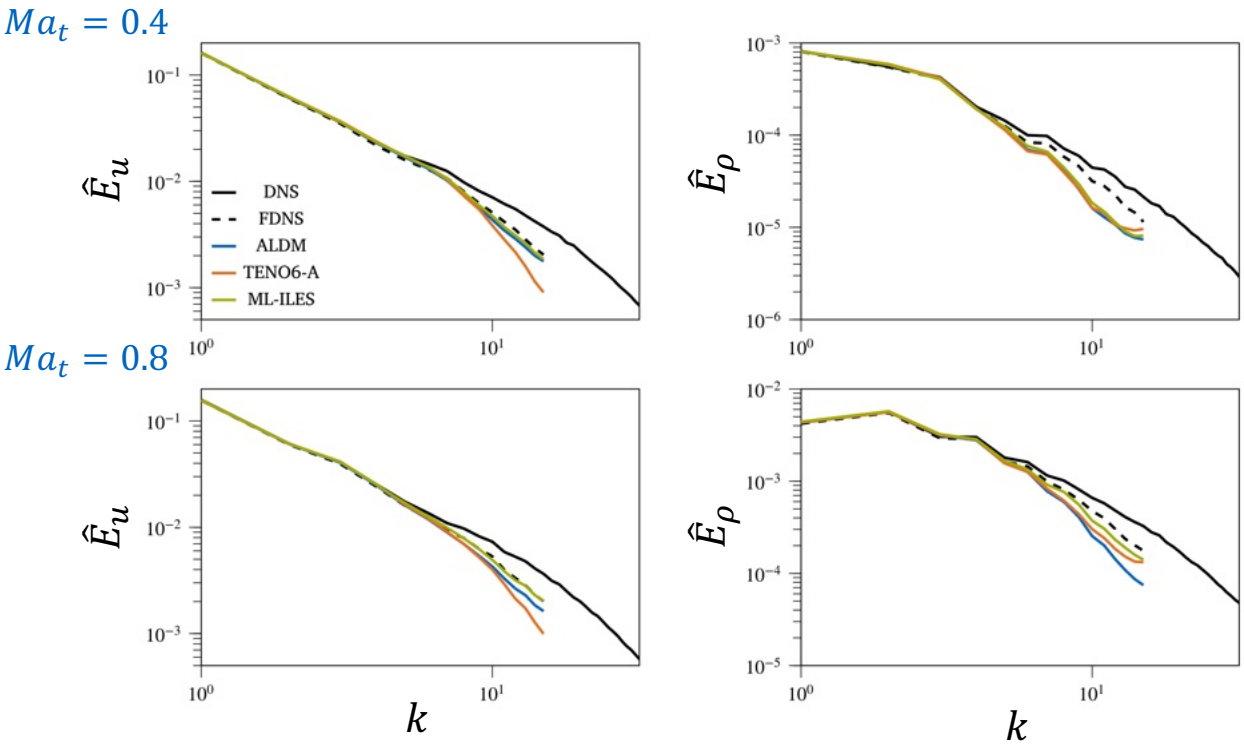
… but also for the solution of inverse problems

- **Parameterized initial condition** is integrated forward in time
- Loss is typically computed using observables of the spatio-temporal trajectory
- Example: Shape optimization → **Drag as QoI** to be minimized

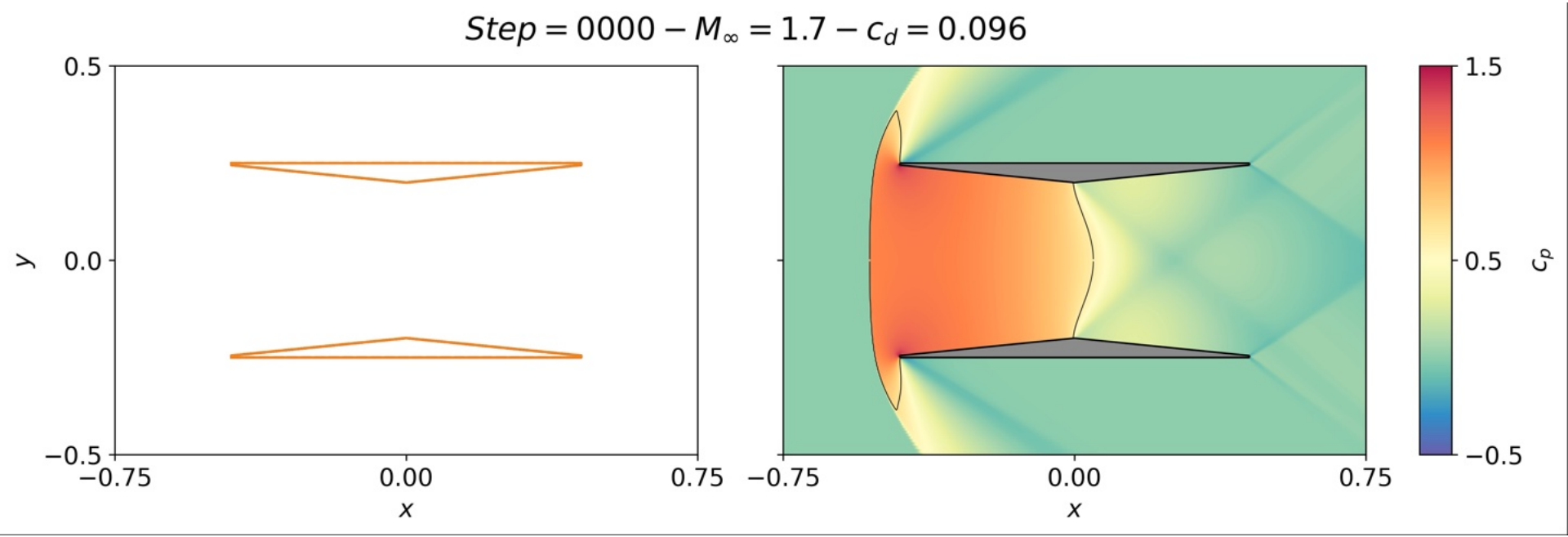


Online training for Machine-learned Implicit LES

- A standard HLLC flux function is combined with a **neural network-based interface reconstruction**.
- The data-driven discretization is trained on **coarse-grained HIT data** $Ma_t \in \{0.4, 0.6, 0.8\}$, $Re_\tau \approx 200$
- The model is trained online using JAX-Fluids.



Parametric Shape Optimization by Automatic Differentiation Adjoints



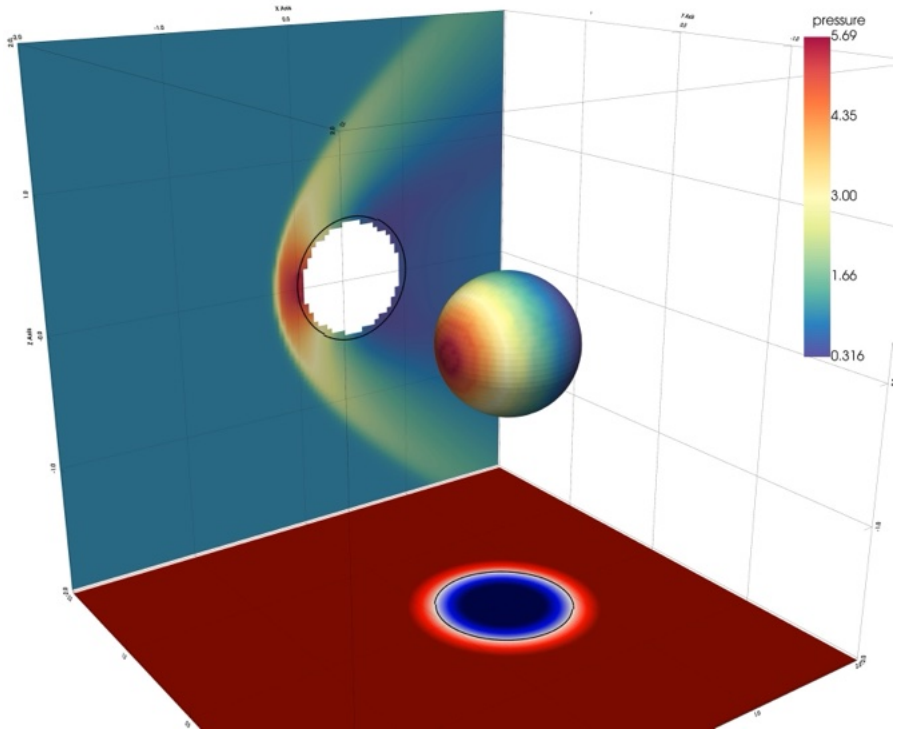
Optimization by **black-box gradient** or **automatic differentiation Jacobians for adjoint equations.**

Free-form Shape Optimization by Automatic Differentiation Adjoints

- We want to optimize the shape of an immersed solid in **supersonic flow** in order to reduce its drag.

- QoI is the drag; but the loss takes into account the initial volume of the solid

$$\mathcal{L} = C_d + (V - V_0)^2$$



Conclusion & Outlook

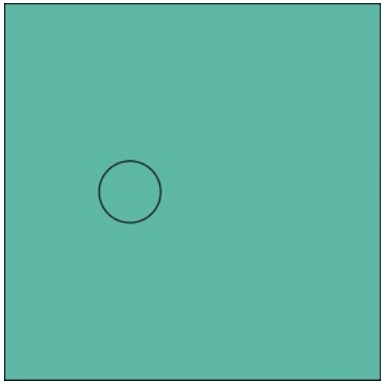
Online training

Gradients are backpropagated through the solver → Models are PDE-informed, improving generalization and long-term stability

JAX-Fluids: Differentiable CFD

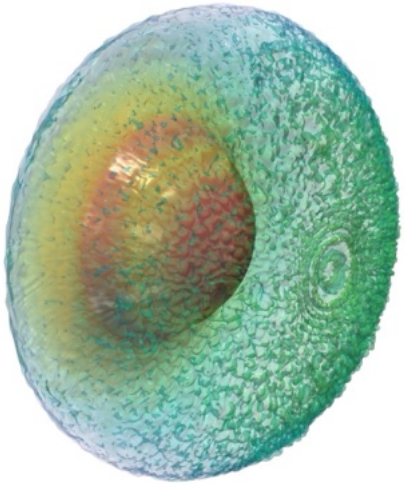
HPC-capable differentiable CFD solver for compressible single- and two-phase flows
→ Model training & inverse problems by AD gradients

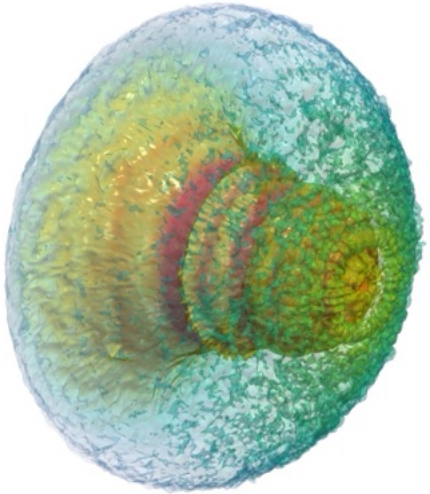
Towards a Differentiable Multiphysics Simulator

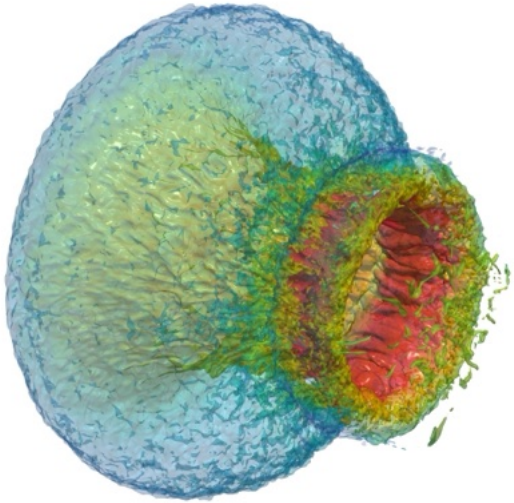


Discretized equations are a powerful inductive bias, and differentiable CFD opens novel avenues for data-driven model development and inverse problems.

Toward Machine-learned Discretizations & Differentiable CFD for Compressible Two-phase Flows







Deniz A. Bezgin

ERCOFTAC – 20th Da Vinci Competition

October 9, 2025