

Machine Learning Applications in CFD (9 credits, 2-0-3-0)

2026-27, Sem II

The course treats machine learning as a toolset for CFD, not a replacement for it. You will build surrogates and closures on data generated in OpenFOAM, inspect them in ParaView, and test them against the solver they are meant to serve.

Item	Detail
Level	Open elective
Instructor	Prof. Santanu De (ME)
Prerequisite	A first course in CFD (ME630 or equivalent). Working knowledge of Python, basic linear algebra, and probability is necessary.
Contact hours	26 lectures of 50 minutes (two per week) and 13 labs of 3 hours (one per week), over a 13-week semester
Software	OpenFOAM, ParaView, and Python with PyTorch and scikit-learn
Computing	IITK HPC cluster. OpenFOAM runs in parallel with MPI for all data generation, and model training is CPU-optimized

What you will be able to do

By the end of the course you will be able to:

1. Formulate CFD, turbulence, combustion and multiphase-flow problems (closure modeling, flow reconstruction, stochastic combustion, atomization and spray modeling) as machine learning problems.
2. Generate ML-ready training data from OpenFOAM simulations run on an HPC cluster, and embed trained models back into OpenFOAM as custom closures (`RASModel`, LES subgrid-scale model).
3. Implement and train standard architectures (MLP, CNN, RNN/LSTM, autoencoders, PINNs, DeepONet/FNO) on OpenFOAM-generated data.
4. Apply machine learning to stochastic turbulent combustion (transported-PDF manifold learning, chemistry surrogates, instability-precursor forecasting) and to atomization (droplet statistics from interface-resolved data, droplet drag and deformation, secondary breakup and collision closures).
5. Quantify and propagate aleatoric and epistemic uncertainty in ML-CFD surrogates, and detect when a surrogate is evaluated outside its training regime.
6. Carry out a peer-review-style ML-CFD research project on your own, from OpenFOAM data generation through trained models and error bounds to a defended result.

Course modules

The course moves from the foundations of scientific machine learning through turbulence, combustion and atomization to neural operators and uncertainty quantification. Tentative syllabus is provided below.

Module	Sessions	What it covers
1. Foundations of Scientific Machine Learning	1–4	Automatic differentiation, multilayer perceptrons and backpropagation, optimization and loss design including physics-constrained losses, CNN and RNN/LSTM architectures
2. Dimensionality Reduction and Flow Reconstruction	5–7	Proper orthogonal decomposition, convolutional autoencoders, super-resolution from coarse-grid CFD or sparse PIV data
3. ML for Turbulence Modeling: RANS and LES	8–11	Data-driven Reynolds-stress correction with tensor-basis networks, a-priori and a-posteriori testing, embedding a closure in OpenFOAM, CNN and GAN subgrid-scale models
4. ML for Stochastic Turbulent Combustion	12–15	Manifold learning for transported-PDF solvers, neural surrogates for stiff chemistry compared with ISAT, LSTM forecasting of thermoacoustic and lean-blowout precursors
5. ML in Multiphase Flow: Atomization and Spray Modeling	16–18	The atomization modeling chain, droplet statistics learned from interface-resolved data, secondary breakup, droplet drag and deformation, collision and coalescence closures
6. Physics-Informed Neural Networks and Neural Operators	19–22	Physical residuals in the loss, a worked 1D Burgers' example, DeepONet, the Fourier neural operator
7. Uncertainty Quantification in ML-Assisted CFD	23–26	Aleatoric and epistemic uncertainty, deep ensembles, polynomial chaos expansions, out-of-distribution detection, course synthesis

Week-by-week schedule (Tentative)

These are tentative class/lab schedules. There can be deviation from this course plan. Each week has two 50-minute lectures and one 3-hour lab.

Week	Sessions	Lectures	Lab
1	1–2	Course overview and the modeling bottlenecks of classical CFD, automatic differentiation; multilayer perceptrons,	Environment setup (OpenFOAM MPI build, HPC job submission, PyTorch, PyVista and ParaView); automatic

Week	Sessions	Lectures	Lab
		forward pass and backpropagation	differentiation on a 1D conservation law
2	3–4	Optimization (SGD, Adam, L-BFGS) and loss design, including physics-constrained losses; CNN and RNN/LSTM architectures	MLP and backpropagation from scratch on an OpenFOAM-generated dataset; a short CNN and RNN exercise
3	5–6	Proper orthogonal decomposition of CFD snapshots; convolutional autoencoders for flow-field compression	POD on OpenFOAM transient snapshots (icoFoam, pisoFoam) generated on the HPC cluster
4	7–8	Super-resolution from coarse-grid CFD or sparse PIV data; RANS and LES recap, and the tensor-basis neural network for Reynolds-stress correction	Convolutional autoencoder for non-linear modal decomposition; super-resolution from coarse to fine OpenFOAM fields
5	9–10	A-priori and a-posteriori testing, and embedding a trained closure in OpenFOAM as a custom RASModel; LES subgrid-scale modeling with CNNs	Train a tensor-basis Reynolds-stress closure against DNS/LES data, embed it in OpenFOAM, and compare a-priori and a-posteriori behavior
6	11–12	LES subgrid-scale modeling with GANs; recap of transported PDFs, SDEs, Lagrangian Monte Carlo and micromixing	Train a CNN or GAN subgrid-scale model and test it inside an OpenFOAM LES run
7	13–14	Manifold learning for transported-PDF solvers; neural surrogates for stiff chemistry and flamelet libraries, ISAT versus neural chemistry	Run a transported-PDF (ESF, Lagrangian Monte Carlo) reacting-flow case and examine the thermochemical state manifold in ParaView
8	15–16	LSTM forecasting of thermoacoustic and lean-blowout precursors; atomization as a multiscale modeling problem	Train a manifold-mapping or neural-chemistry surrogate and couple it into the transported-PDF solver
9	17–18	Learning primary atomization from interface-resolved data; secondary breakup, droplet drag and collisions in the Lagrangian spray	Train an RNN/LSTM instability-precursor forecaster on pressure and heat-release time series
10	19–20	Physics-informed neural networks: residuals in the loss, boundary and initial conditions; the 1D Burgers' example,	Train a droplet drag and deformation surrogate on interface-resolved OpenFOAM data

Week	Sessions	Lectures	Lab
		PINNs versus classical solvers	
11	21–22	DeepONet; the Fourier neural operator and real-time parametric prediction	Train an atomization closure: a drop-size distribution from a VOF jet, or a collision and breakup sampler
12	23–24	Aleatoric and epistemic uncertainty; deep ensembles and polynomial chaos expansions	Implement a PINN for 1D Burgers'; train a DeepONet or FNO on a parametric family of OpenFOAM solutions
13	25–26	Out-of-distribution detection; confidence intervals and variance maps from HPC-parallel ensembles; course synthesis	UQ capstone: deep ensembles, polynomial chaos and out-of-distribution detection on a surrogate combustor model; project work session

Labs and computing

Every lab generates or analyzes its data through OpenFOAM and ParaView on the IITK HPC cluster, and each lecture topic is followed by a hands-on implementation. Lab 1 sets up your environment: the OpenFOAM MPI build, job submission on the cluster, PyTorch, and PyVista with ParaView.

- **OpenFOAM** is the only CFD solver in the course and runs in parallel with MPI for every dataset. Across the labs you will use `simpleFoam`, `pimpleFoam`, `icoFoam` and `pisoFoam`, `chemFoam`, `reactingFoam`, a volume-of-fluid solver for the atomization labs, the `forces` function object, and `blockMesh` and `snappyHexMesh`.
- **ParaView** is the main tool for visualization and field extraction, used together with the PyVista Python bindings to turn fields into ML-ready tensors.
- **Python with PyTorch** is used for all model definition, automatic differentiation and training, CPU-optimized on the cluster's compute nodes.
- **scikit-learn** provides classical machine learning utilities and baselines.
- A **polynomial-chaos and UQ toolkit**, such as `chaospy`, is used in Module 7.

Course Evaluation

The end-term project is mandatory for all students. A student who fails to complete this, will get “F” grade. Attendance in all lab classes are mandatory.

Weightage: Midsem (30%), Lab Assignments (30%), End term project (40%)

Malpractice and plagiarism will lead to de-registration and will be reported to the competent authority.

Reading

Reading comes in two parts: textbooks, with the chapters that match each module, and research papers grouped by module.

Textbooks

Chapters from following books will be used in this course. Apart from these books, relevant journal articles and other reading materials will be shared on different modules.

I. Goodfellow, Y. Bengio, A. Courville, Deep Learning (MIT Press, 2016)

S. L. Brunton, J. N. Kutz, Data-Driven Science and Engineering (Cambridge University Press)

S. B. Pope, Turbulent Flows (Cambridge University Press, 2000)

F. Moukalled, L. Mangani, M. Darwish, The Finite Volume Method in Computational Fluid Dynamics (Springer, 2016)

H. K. Versteeg, W. Malalasekera, An Introduction to Computational Fluid Dynamics: The Finite Volume Method (Longman, 1st ed., 1995)