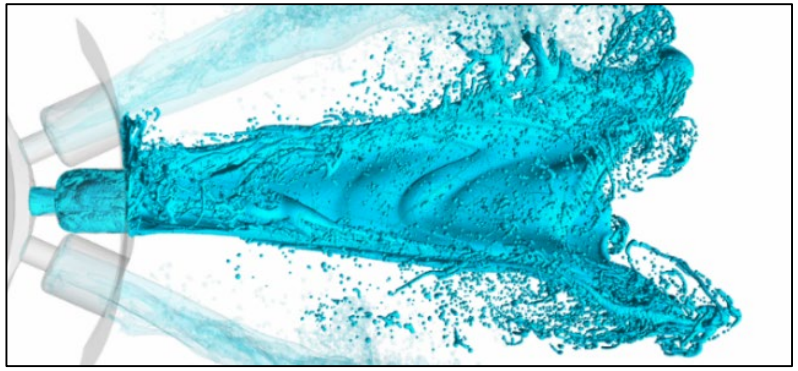
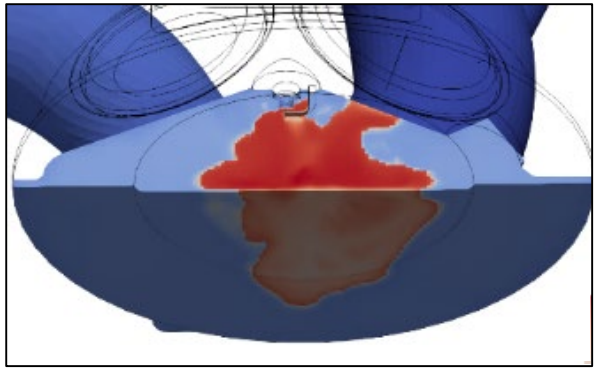
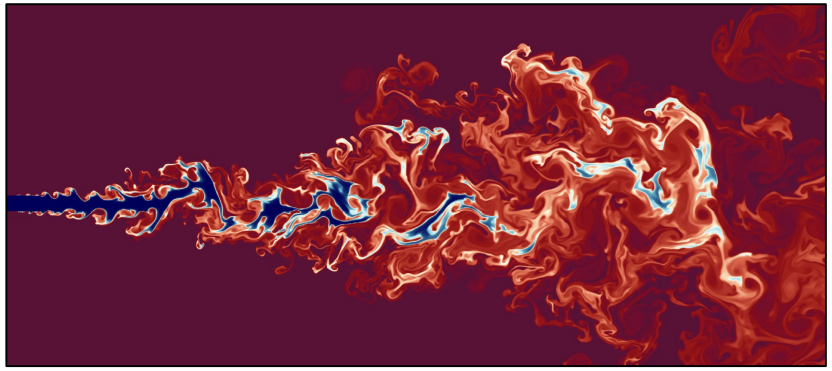




Accelerating multi-phase flow simulations with neural network models in OpenFOAM for fuel-air mixing under transcritical conditions

Nasrin Sahranavardfard, Faniry N. Z. Rahantamialisoa, Michele Battistoni

Dept. of Engineering, University of Perugia, Italy

Outline

- Introduction
  - Background
  - Motivation for NN
- Computational aspects
  - Governing equations and thermophysical models
  - CFD numerical framework
  - Real-fluid data sets and NN model training
  - Coupling between NN models and openFoam
- Results
  - GH2-LOX test case
  - Comparison between NN-version and original-version
  - Performance
- Conclusion
  - Summary and Future work

Background and motivation

Context:

- Increase in energy costs
- Stringent emission policies toward climate neutrality and decarbonization

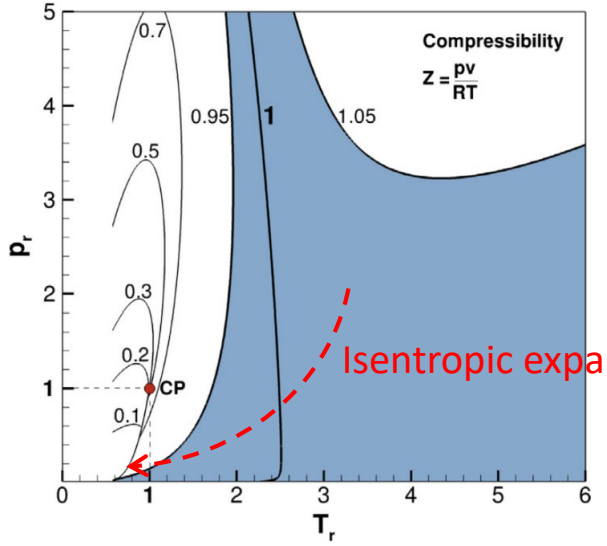
E-fuels including **hydrogen** and hydrogen carriers **(ammonia,..)** are part of the solution

Prospectives:

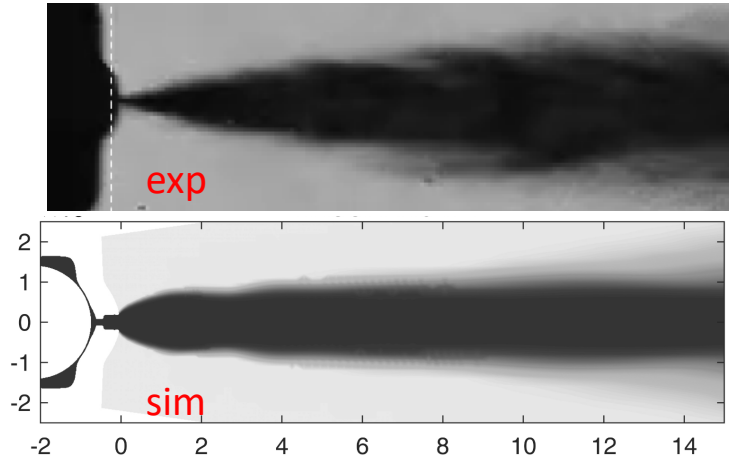
- At high-loads, heavy-duty H2-ICEs performance with DI can be comparable to FCEVs for HD transportation
- Green H2 can become a sustainable/affordable option.

Two-phase flow properties and challenges

- Accurate understanding and prediction capabilities of supercritical, and transcritical jet formation and mixing are important for clean and efficient combustion process.
- Many parameters can have large impacts (inj. pressure, back pressure, temperature, nozzle geometry)
- At high pressure, accounting for real fluids effects may be necessary
- Goal: explorations of the effect of real gas under operating conditions relevant to H2, NH3, other e-fuels, with high pressure, or crossing the two-phase region.



Example of ammonia flash boiling spray



Computational methods: Governing equations

- Mass $\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{U}) = 0$
- Momentum $\frac{\partial \rho \mathbf{U}}{\partial t} + \nabla \cdot (\rho \mathbf{U}\mathbf{U}) = -\nabla p + \nabla \cdot \boldsymbol{\tau}$
- Energy $\frac{\partial \rho h_T}{\partial t} + \nabla \cdot (\rho h_T \mathbf{U}) = \frac{\partial p}{\partial t} + \nabla \cdot (\mathbf{U} \cdot \boldsymbol{\tau}) - \nabla \cdot \mathbf{q}$
- Species $\frac{\partial \rho Y_k}{\partial t} + \nabla \cdot (\rho Y_k \mathbf{U}) = \nabla \cdot \mathbf{J}_k$

$\boldsymbol{\tau}$ = viscous stresses

$\mathbf{q}$ = heat flux due to conduction and species enthalpy flux

$\mathbf{J}_k$ = species diffusion velocities

- coupled with a turbulence model: RANS, LES
- diffuse interface method

Thermodynamics and transport properties

Thermodynamics

- Real-fluid mixture density (PR EOS with non-linear Van der Waals mixing rules)
- Real-fluid caloric properties (departure functions based on PR EOS for the mixture)

Transport

- Real-fluid transport properties (Chung correlation)

$$p(v,T,x_i) = \frac{RT}{v-b_m} - \frac{a_m}{v^2+2b_m v-b_m^2}$$

$$h_m(p,T) - h_{0,m}(p,T) =$$

$$= pv_m - RT + \frac{1}{2b_m\sqrt{2}}\left[a_m - T\frac{\partial a_m}{\partial T}\right]\left[\frac{v_m + b_m(1-\sqrt{2})}{v_m + b_m(1+\sqrt{2})}\right]$$

Mixture coefficients $a_m = \sum_i \sum_j x_i x_j a_{ij}$ $b_m = \sum_i x_i b_i$

Non-linear blends, interactions coeff., *ij* $a_{ij} = (1-k_{ij})\sqrt{a_i a_j}$

Thermo package:

thermoType	
type	hePsiThermo;
mixture	reactingMixture;
transport	chungMix;
thermo	janaf;
energy	sensibleEnthalpy;
equationOfState	mixturePengRobinsonGas;
specie	specie;

- Rahantamialisoa FNZ, et al., 2021, International Conference on Liquid Atomization and Spray Systems ILASS
- Rahantamialisoa FNZ, et al., 2022, *Journal of Physics: Conference Series* Vol. 2385, No. 1, p. 012051. IOP Publishing.

Computational methods: Numerical frameworks

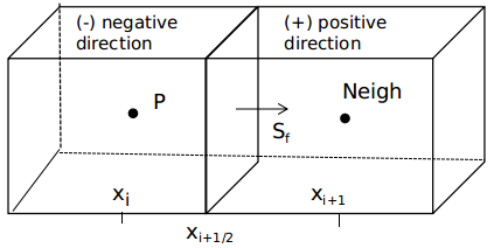
realFluidReactingFoam

- based on **reactingFoam**, with added species transport equations, and real-fluid library
- modified pressure equation to include density changes within the inner loop
- changes in the frequency of updates of other variables (enthalpy, specie)

$$\frac{\partial(\rho_0 - \psi_0 p_0)}{\partial t} + \frac{\partial \psi_0 p}{\partial t} + \frac{\partial}{\partial x_i}\left(\rho \frac{H_P}{A_P}\right) - \frac{\partial}{\partial x_i}\left(\frac{\rho}{A_P}\frac{\partial p^*}{\partial x_i}\right)$$

realFluidHybridFoam

- based on **rhoPimpleCentralFoam**[1], with added species transport equations, and real-fluid library
- with KT/KNP (Kurganov Tadmor / Kurganov Noelle Petrova) schemes for convective terms (flux splitting)

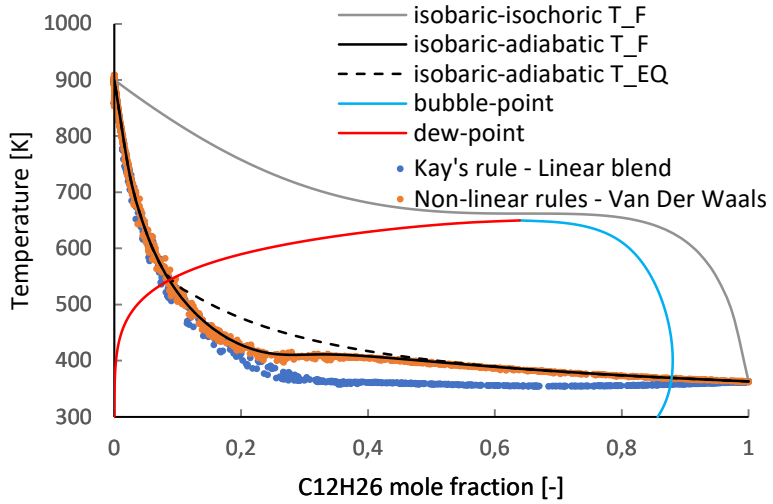


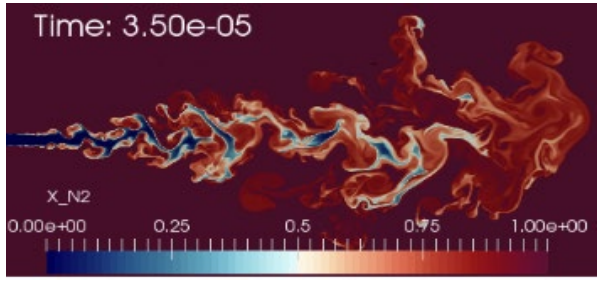
$$\sum_f \phi_f \Psi_f = \sum_f \left[\alpha \phi_{f+} \Psi_{f+} + (1-\alpha)\phi_{f-}\Psi_{f-} + \omega_f(\Psi_{f-} - \Psi_{f+})\right]$$

[1] Kraposhin MV, Banholzer M, Pfitzner M, Marchevsky IK. A hybrid pressure-based solver for nonideal single-phase fluid flows at all speeds. International Journal for Numerical Methods in Fluids. 2018 20;88(2):79-99.

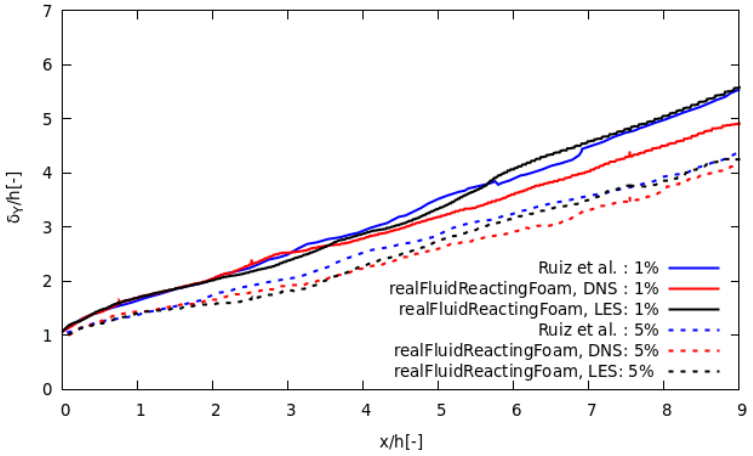
Previous validations

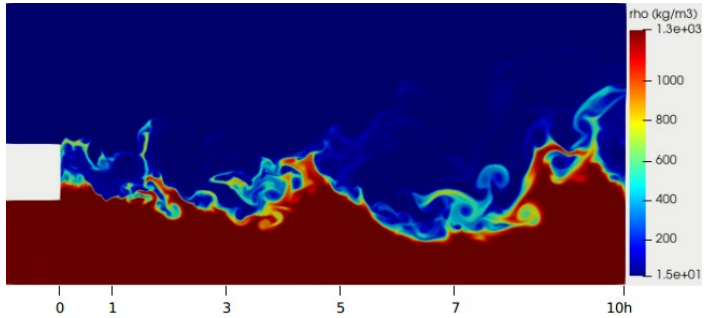
Spray A [1]



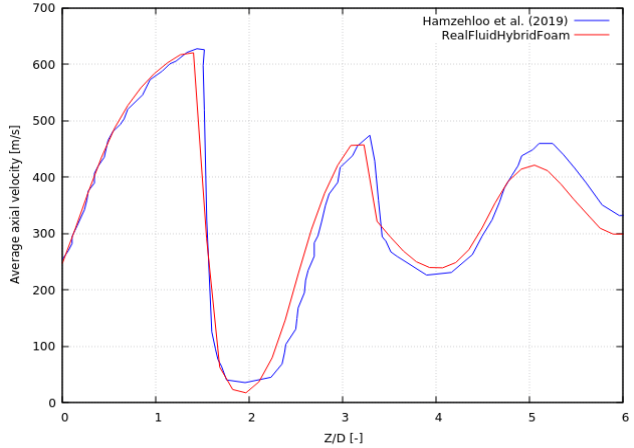


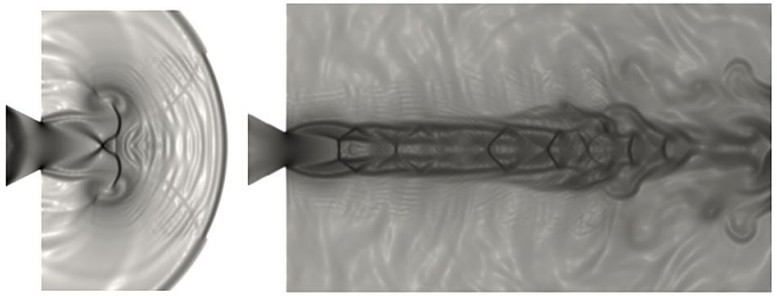
LOX-H2 [2]





H2 jets [3]

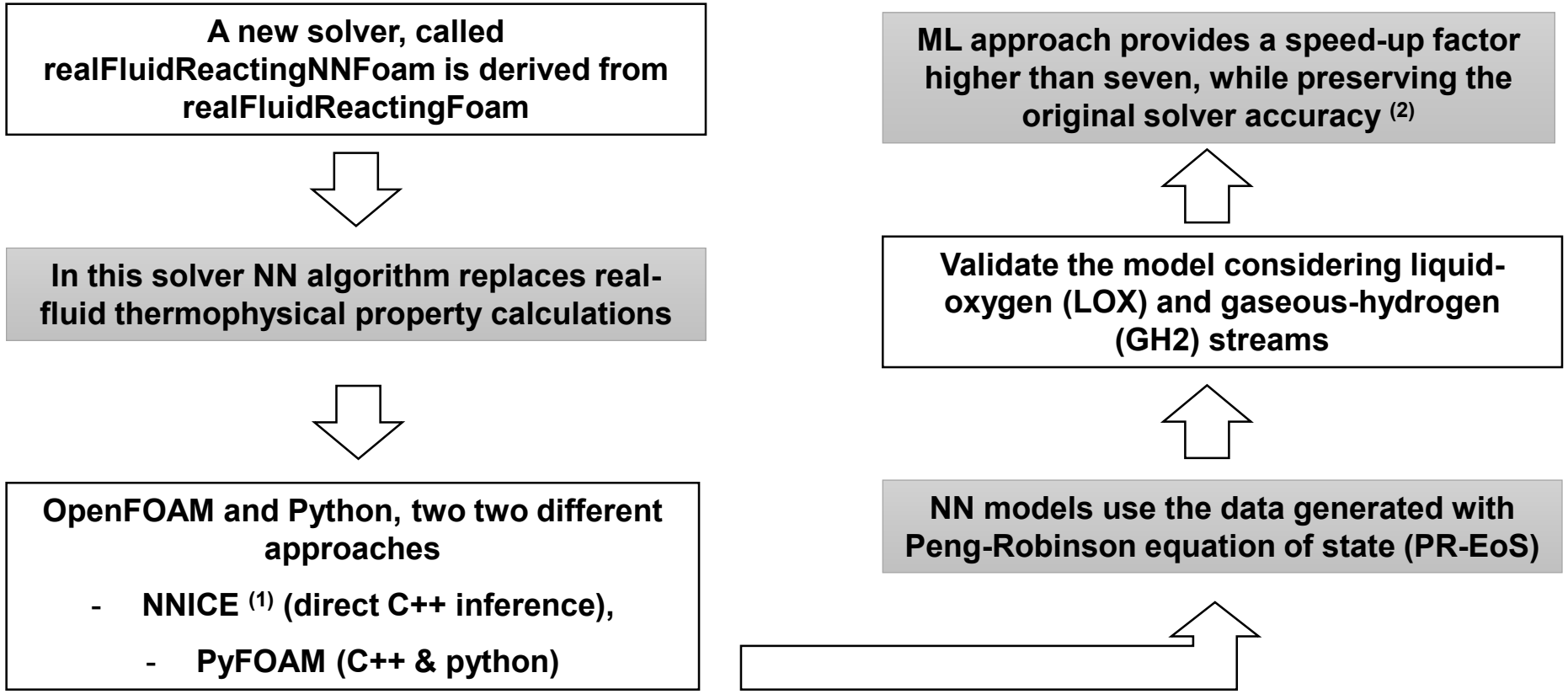




[1] Ningegowda, B.M.; Rahantamialisoa, F.N.Z.; Pandal, A.; Jasak, H.; Im, H.G.; Battistoni, M.; Energies 2020, 13, 5676.
[2] Rahantamialisoa F.N.Z, Pandal A., Ningegowda B.M., Jacopo Z., Nasrin S., H. Jasak, Hong G. Im, Battistoni M., ICLASS 2021.
[3] F Rahantamialisoa, M Battistoni, A Miliozzi, N Sahranavardfard, J Zembi, SAE Technical Paper 2023-01-0312.

Motivation for NN

The purpose is to **reduce computational costs** of multiphase flow simulations using **neural network models** to replace heavy parts related to the calculation of the **thermophysical properties.**

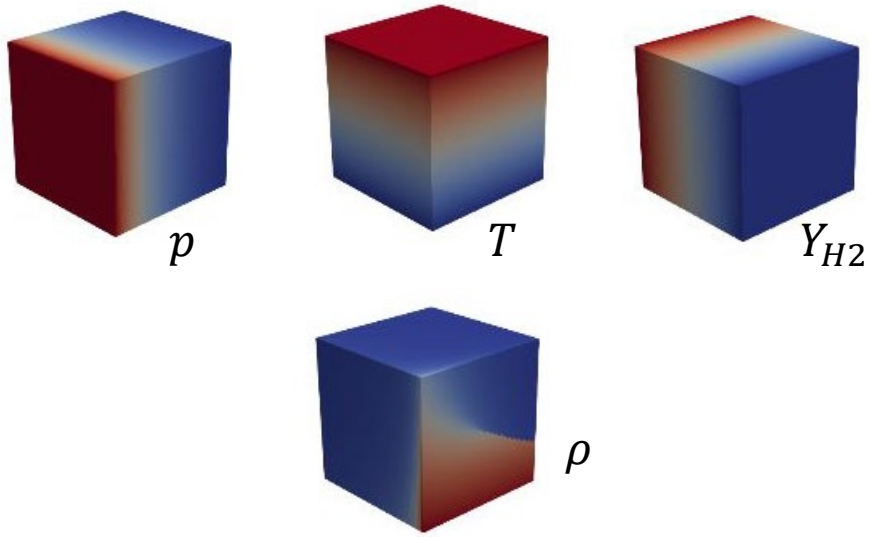


[1] D. Aubagnac (IFP-EN) https://github.com/aubagnacd/NNICE

[2] N. Sahranavardfard, D. Aubagnac-Karkar, C. Habchi, G. Costante, F. N. Z. Rahantamialisoa, and M. Battistoni. “Computation of real-fluid thermophysical properties using a neural network approach implemented in OpenFOAM”, Fluids, MDPI, 2024

Real-fluid Thermophysical Data Generation

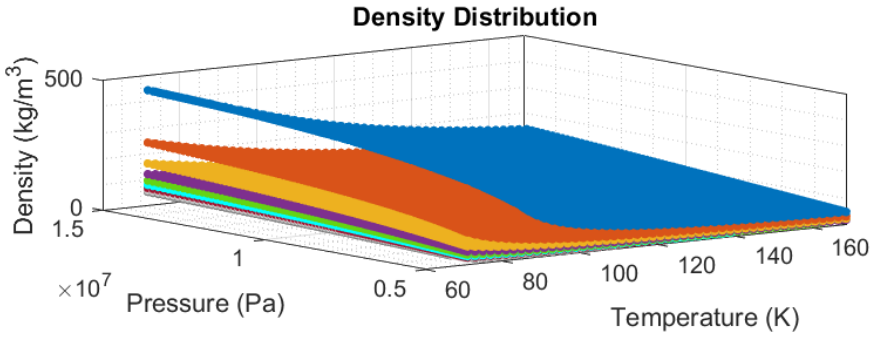
- We have three different inputs, **temperature**, **pressure**, **composition** (hydrogen mass fraction)
- Thermophysical space discretized with **250000** points.
- Data generated through
  - OpenFOAM implemented Peng-Robinson (**realFluidReactingFoam**),
  - or through IFPEN Carnot library

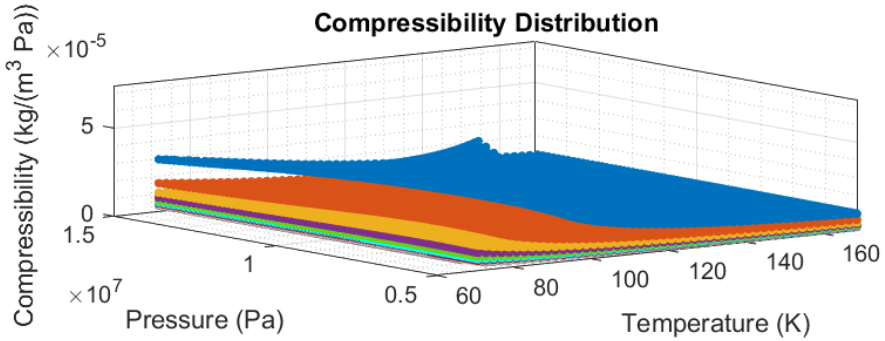


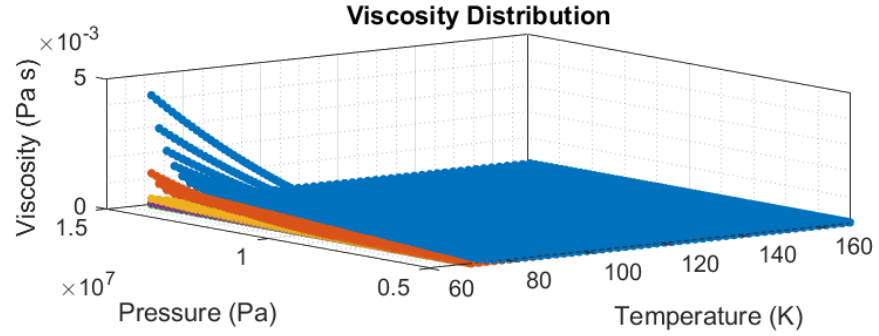
Inputs		Outputs
Variable symbol (code name)	**Definition [units]**	
Y_{H2} (Y)	Fuel mass fraction [-]	all output variables
T (T)	Temperature [K]	all output variables (except T)
p (p)	Pressure [Pa]	all output variables
h (he)	Enthalpy [J/kg]	T

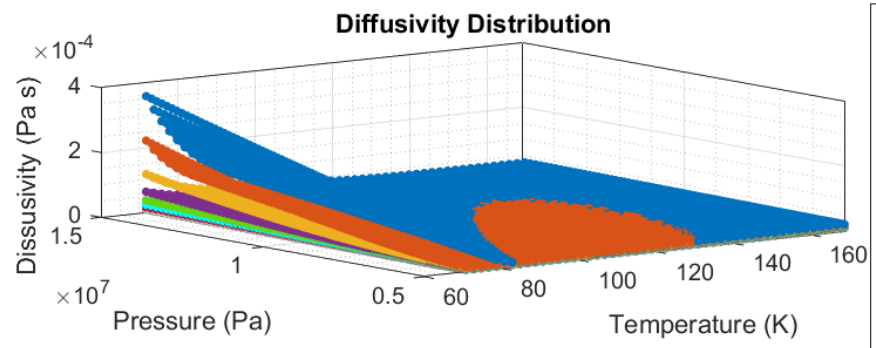
Variable symbol (code name)	**Definition [units]**
ρ (rho)	Density $[kg/m^3]$
μ (mu)	Viscosity $[Pa\ s]$
α $(alpha)$	Thermal diffusivity $[Pa\ s]$
h (he)	Specific enthalpy: $h = sie + p/\rho$ $[J/kg]$
ψ (psi)	Compressibility $\psi = 1/(ZRT)$ $[kg/(m^3 Pa)]$
T (T)	Temperature (K)

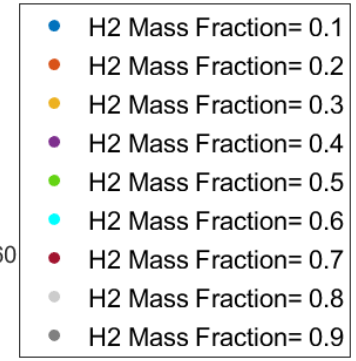
Thermophysical Data Visualization

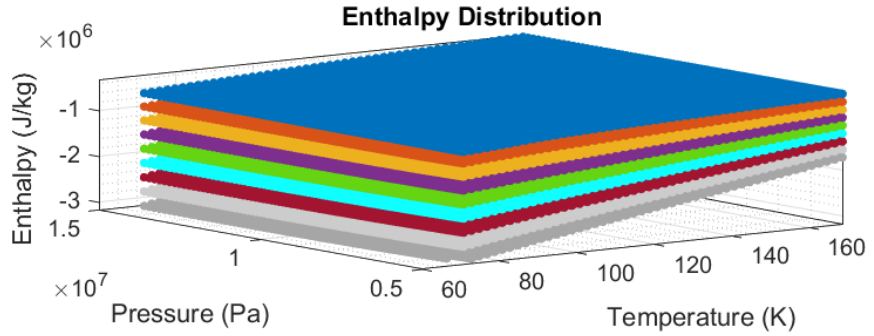


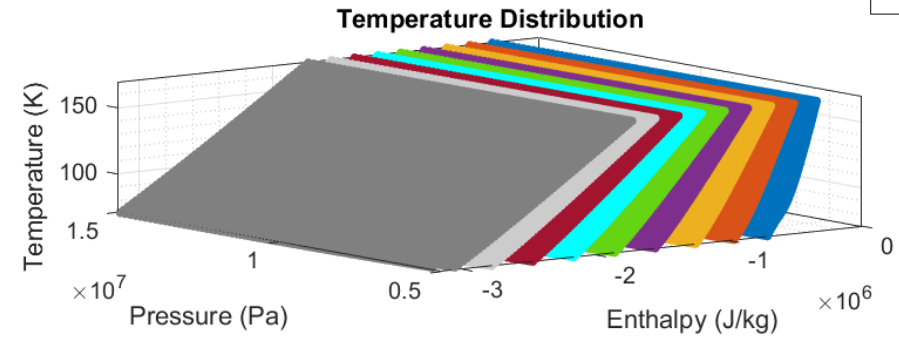




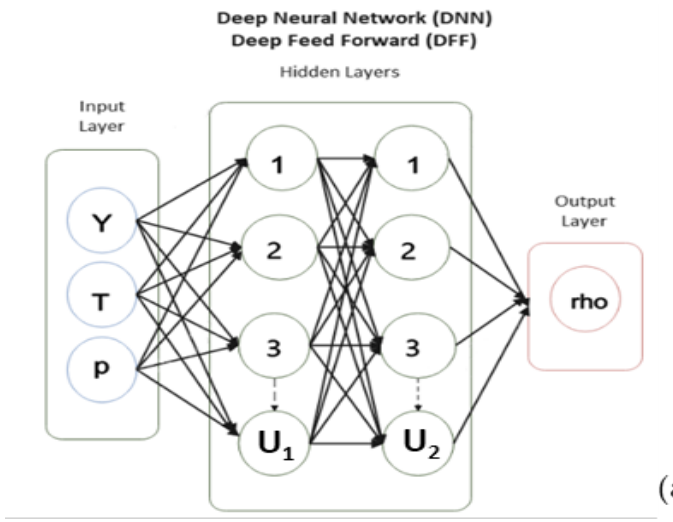




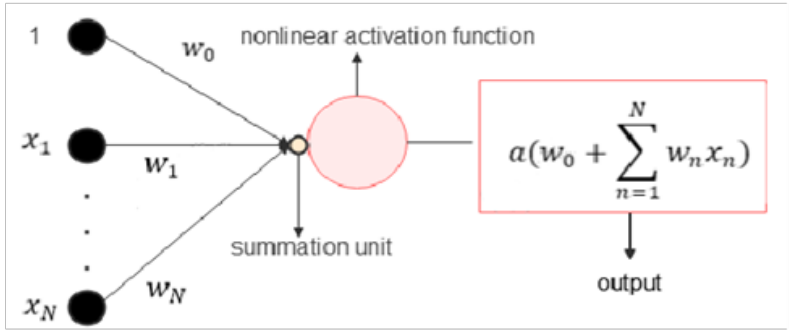




NN model equations



(a)



(b)

$$f_j^{(1)}(\boldsymbol{x}) = \left(v_j(\boldsymbol{x})\right)$$

$$v_j(\boldsymbol{x}) = \boldsymbol{w}_j^{(1)T}\boldsymbol{x} = w_{0,j}^{(1)} + \sum_{n=1}^{N} w_{n,j}^{(1)} x_n, \quad j = 1, \dots, U_1$$

$$\boldsymbol{a}(\boldsymbol{v}) = \boldsymbol{ReLU}(\boldsymbol{v}) = \boldsymbol{max}(\boldsymbol{0}, \boldsymbol{v})$$

$$f_j^{(2)}(\boldsymbol{x}) = a\left(w_{0,j}^{(2)} + \sum_{i=1}^{U_1} w_{i,j}^{(2)} f_i^{(1)}(\boldsymbol{x})\right)$$

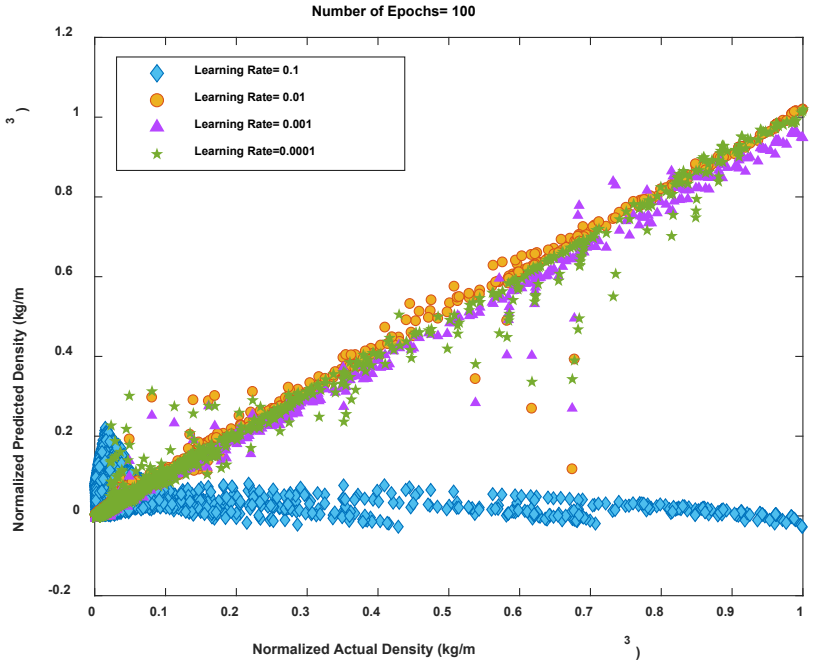
$$f_j^{(L)}(\boldsymbol{x}) = a\left(w_{0,j}^{(L)} + \sum_{i=1}^{U_{L-1}} w_{i,j}^{(L)} f_i^{(L-1)}(\boldsymbol{x})\right)$$

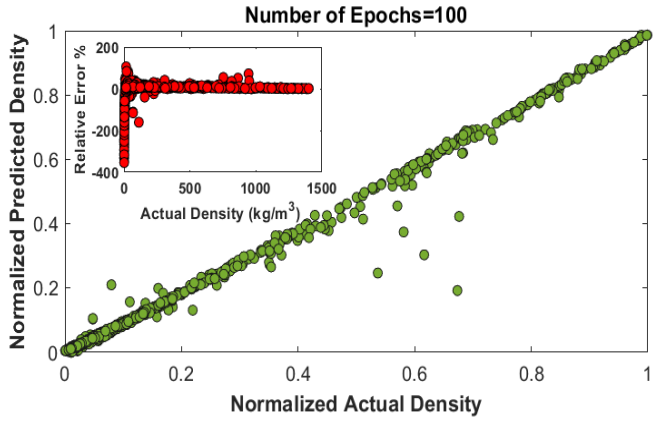
Neural Network algorithm (Python)

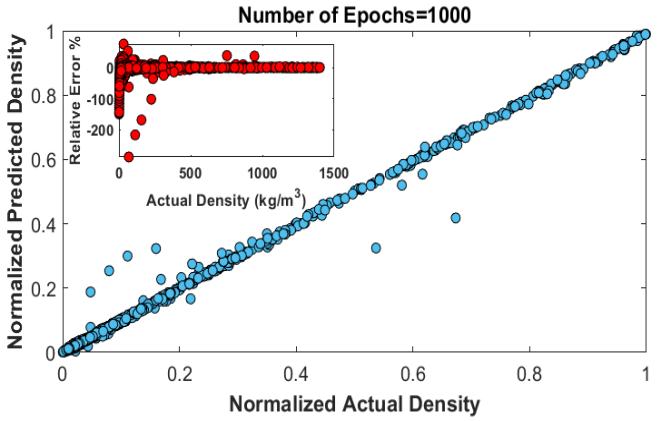
```python
#Import Numpy and Pandas libraries as dependencies
import numpy as np
import pandas as pd

#Read data from a csv file
df = pd.read_csv('file.csv')

#Choose inputs and outputs from imported data
X = df.drop(['variable1'....'variable n'], axis=1)
y = df.drop(['variable1'....'variable n'], axis=1)

#Split train and test dataset, 80 percent as the trainset and 20 pecent as the testset
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)

#Transform the output data of train and test dataset
y_test=np.log(y_test)
y_train=np.log(y_train)

#Scale the input and the output data between 0 and 1
from sklearn.preprocessing import MinMaxScaler
Xscaler=MinMaxScaler(feature_range=(0,1))
X_train=Xscaler.fit_transform(X_train)
X_test=Xscaler.transform(X_test)
Yscaler = MinMaxScaler(feature_range=(0,1))
y_train=Yscaler.fit_transform(y_train)
y_test=Yscaler.transform(y_test)
```

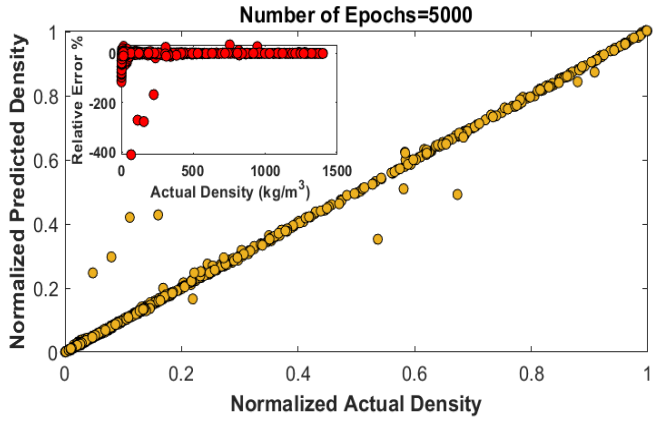
```python

#Import tensorflow and keras libraries, and specify the prefered hyperparameters
import tensorflow as tf
import keras
from keras.models import Sequential
from keras.layers import Dense
model = Sequential()
model.add(Dense(100, input_dim=3, activation='relu'))
model.add(Dense(100, activation='relu'))
model.add(Dense(1, activation='linear'))

#Keras optimization and the model compilation
lr_schedule = tf.keras.optimizers.schedules.ExponentialDecay(
    initial_learning_rate=0.0015,
    decay_steps=10000,
    decay_rate=0.9)
opt = keras.optimizers.Adam(learning_rate=lr_schedule)
model.compile(loss='mse', optimizer=opt, metrics=['mae'])

#Model fit with specification of the number of epochs and the batch size
history = model.fit(X_train, y_train,validation_data = (X_test, y_test), epochs=1000, batch_size=256)
```

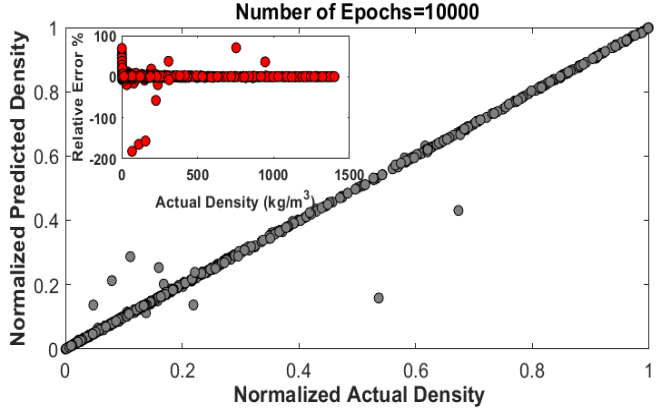
Hyperparameters Importance



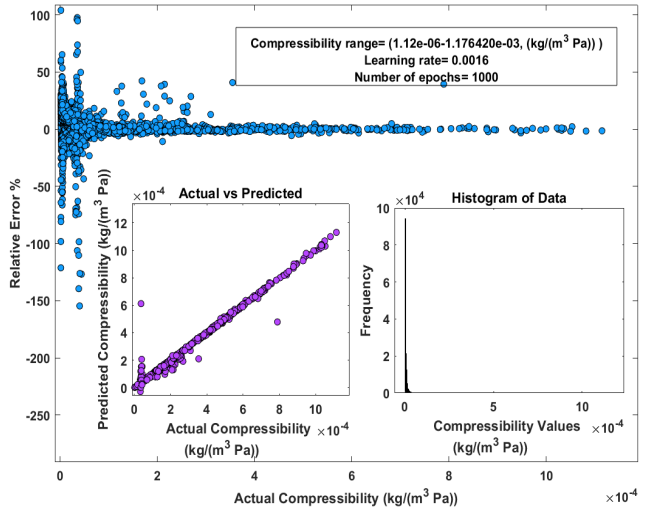


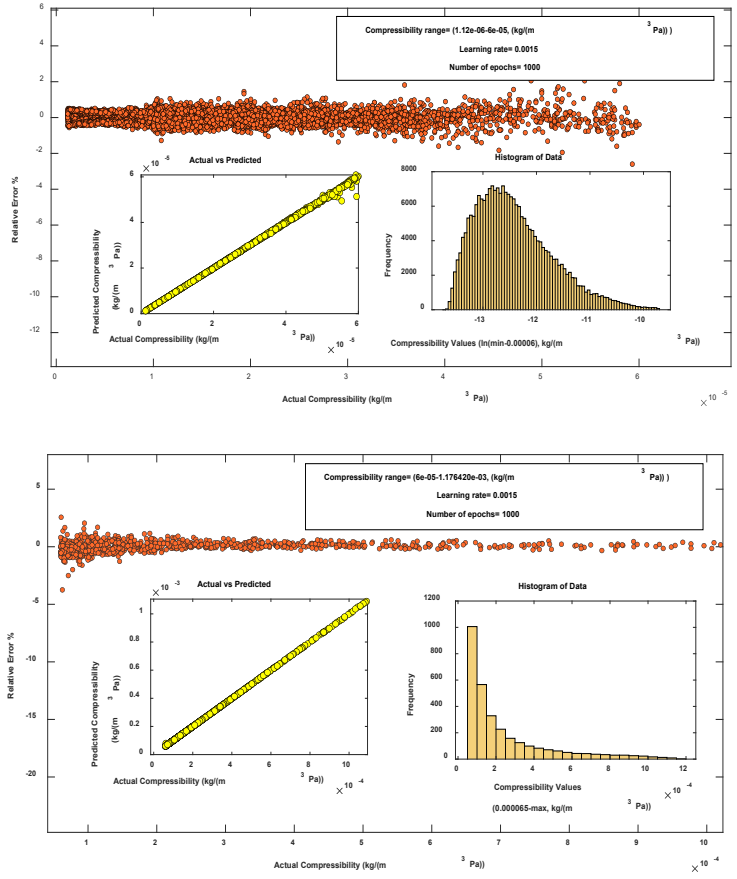


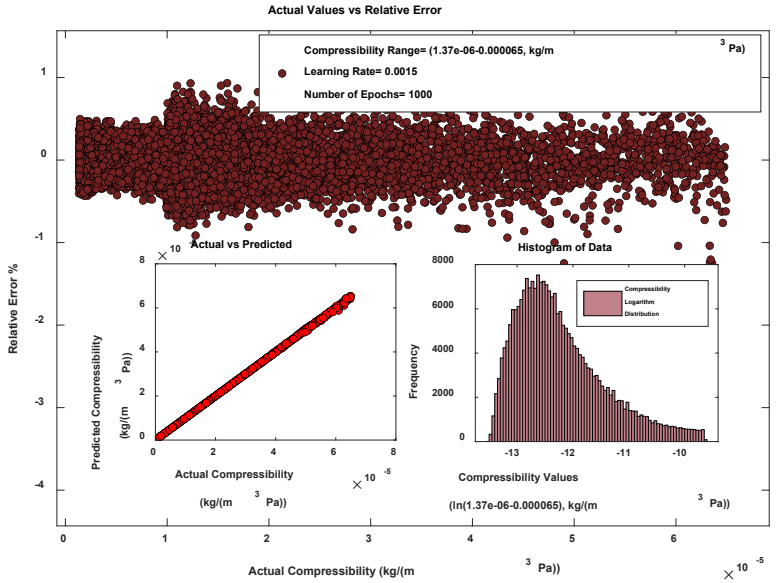




Compressibility Final Training Result

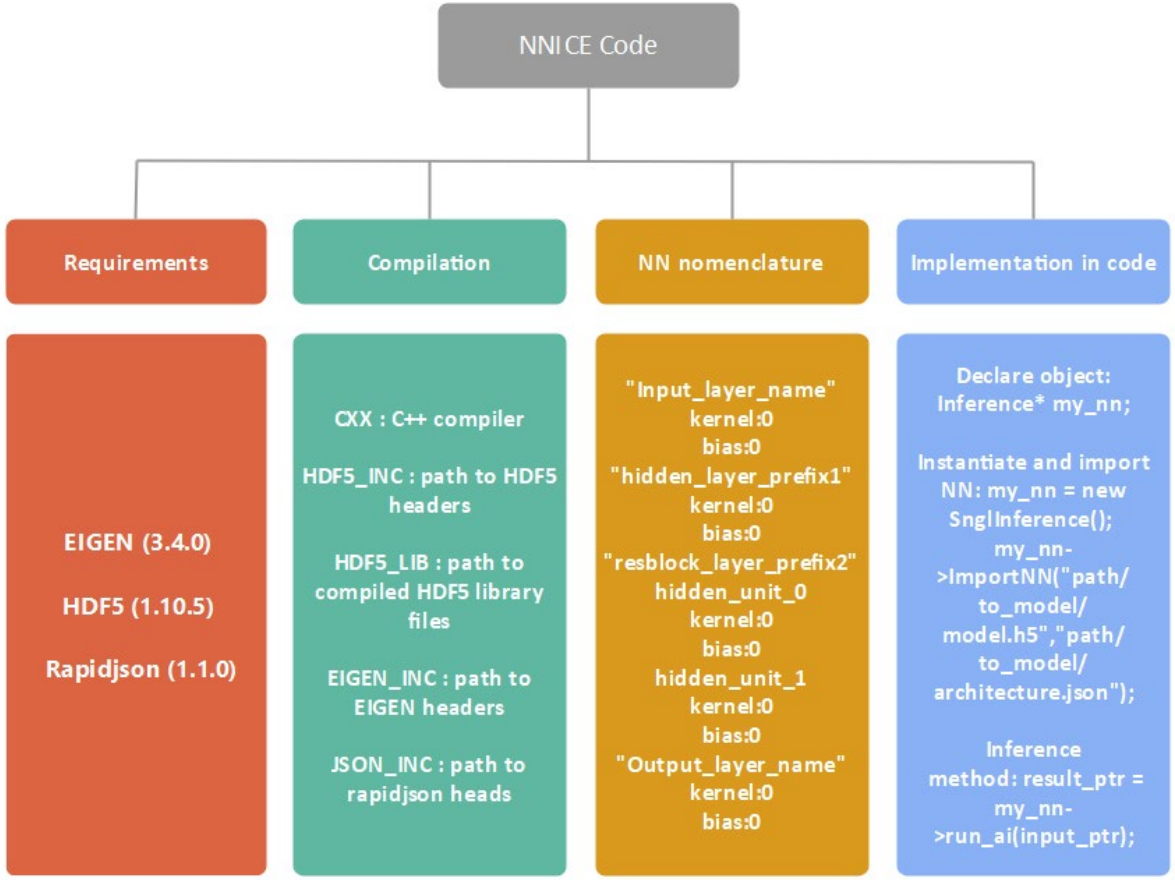


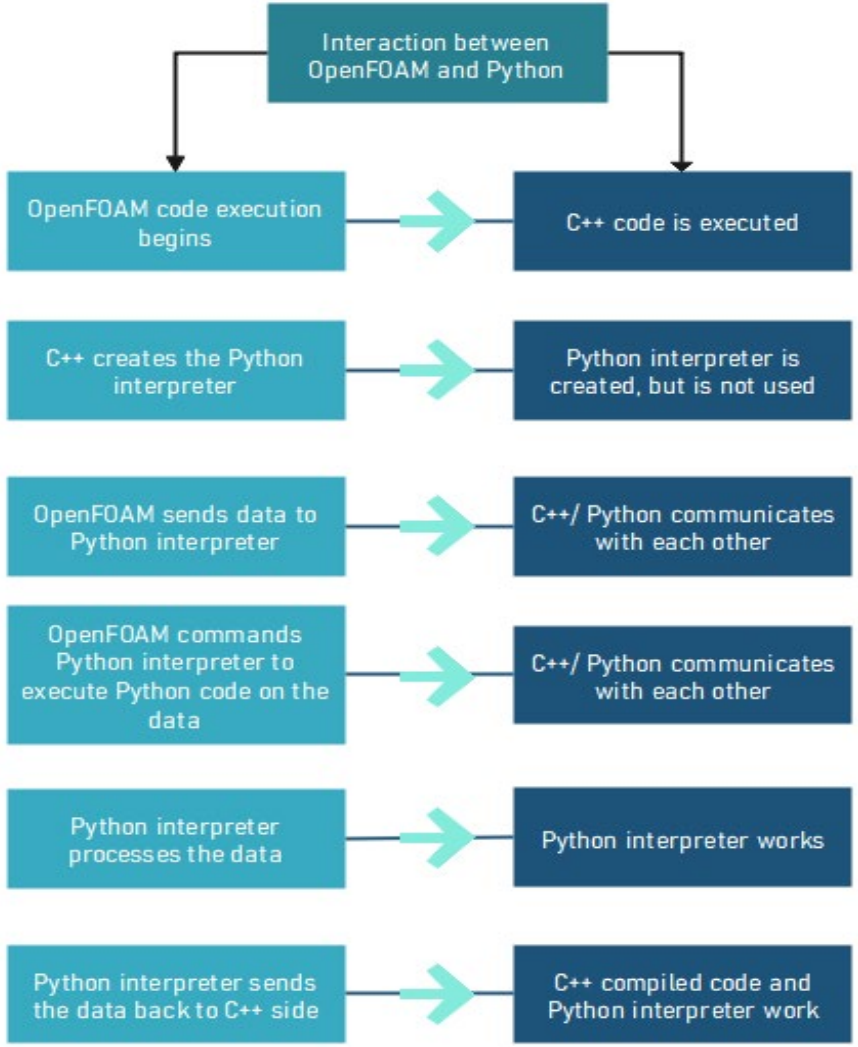




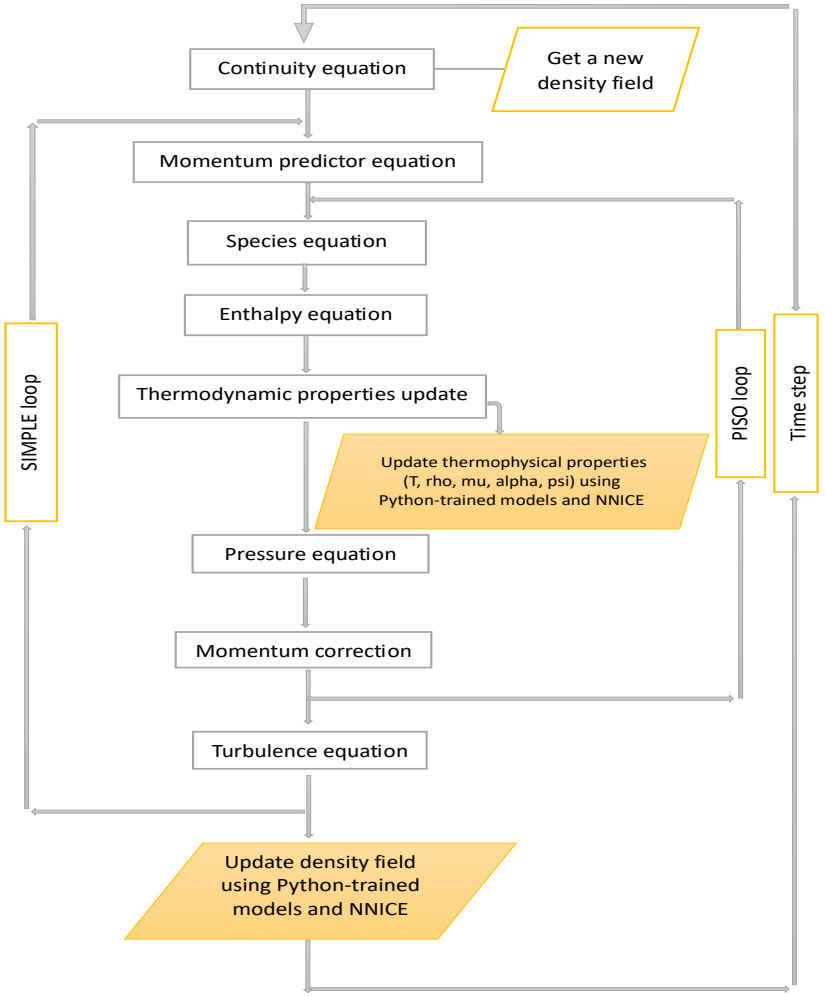
Output (applied min-max-scalar)	Inputs (applied min-max-scalar)	Hidden layer size	Activation function	Solver	Learning-rate-init	Loss metrics	epochs	batch-size
$rho, mu, alpha, he, psi,$	Y_{H2}, T, p	100 × 100	ReLU	ADAM	0.0015	MSE	1000	256
T	Y_{H2}, p, he	100 × 100	ReLU	ADAM	0.0015	MSE	1000	256

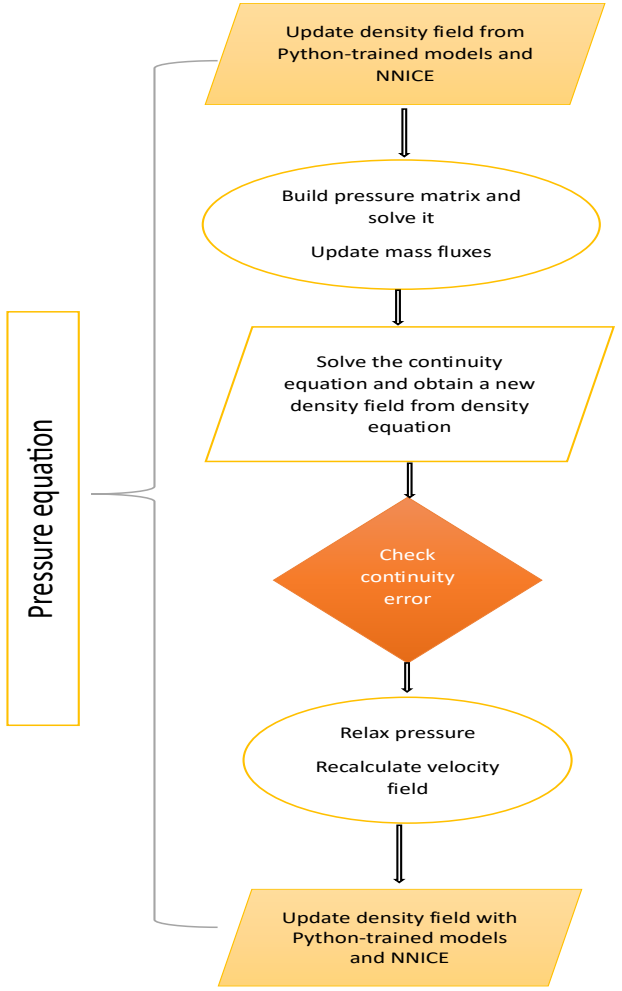
NNICE and PythonFOAM Implementations



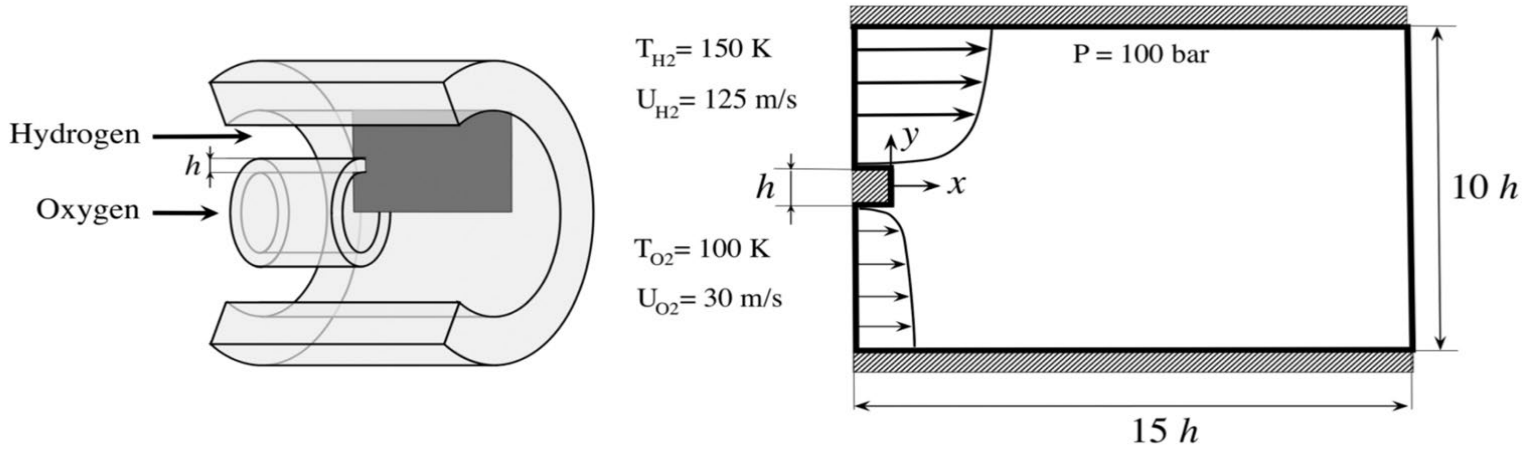


Implementation of the NN models within the PIMPLE loop





GH2-LOX test case: operating condition

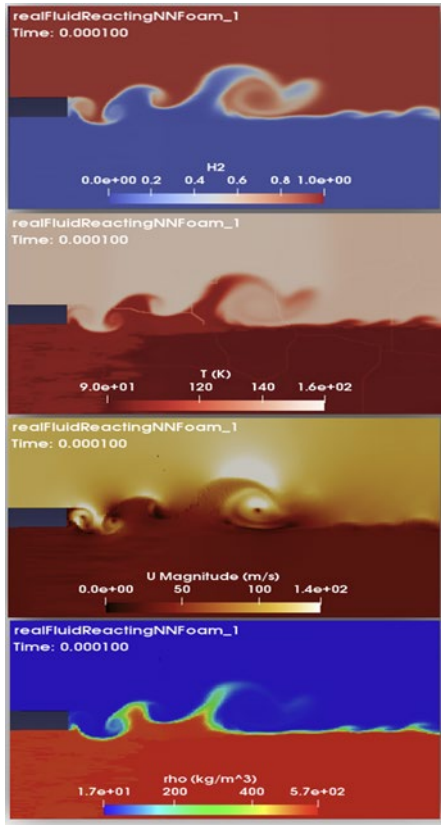


- we analyze a **two-dimensional mixing layer** with an injector lip that divides streams of liquid oxygen **(LOX)** and gaseous hydrogen **(GH2),** both at **supercritical pressure**.
- The simulation setup involves an injector lip with a height **(h) of 0.5 mm**, segregating high-speed gaseous hydrogen (GH2) in the surroundings from dense liquid oxygen (LOX) in the center.
- The area of interest of the computational domain spans **15h** along the axial direction and **10h** in the transversal direction.
- a relatively coarse mesh is employed with a uniform grid spacing of Δ**x/h = Δy/h = 20.**
- At the injector lip, we implement an **adiabatic no-slip** wall condition, while the top and bottom boundaries are treated as **adiabatic slip walls**. Furthermore, we apply a **Dirichlet pressure boundary condition** at the outlet.

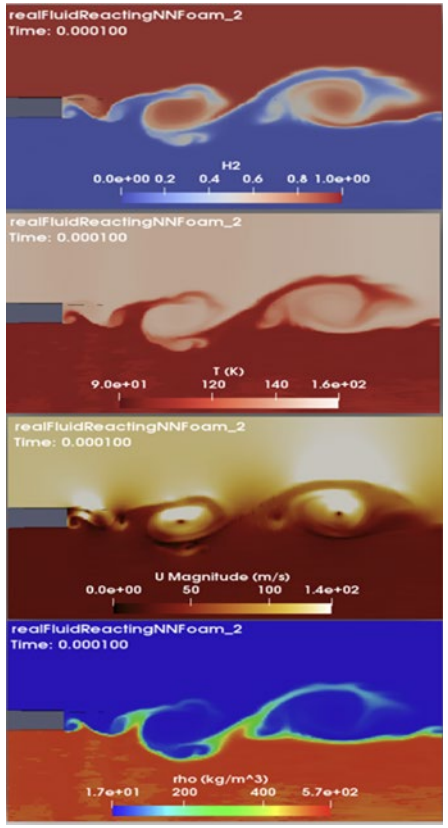
Rahantamialisoa F.N.Z, Pandal A., Ningegowda B.M., Jacopo Z., Nasrin S., H. Jasak, Hong G. Im, Battistoni M., ICLASS 2021.

GH2-LOX - Results and discussion: fields @ t = 0.1 ms

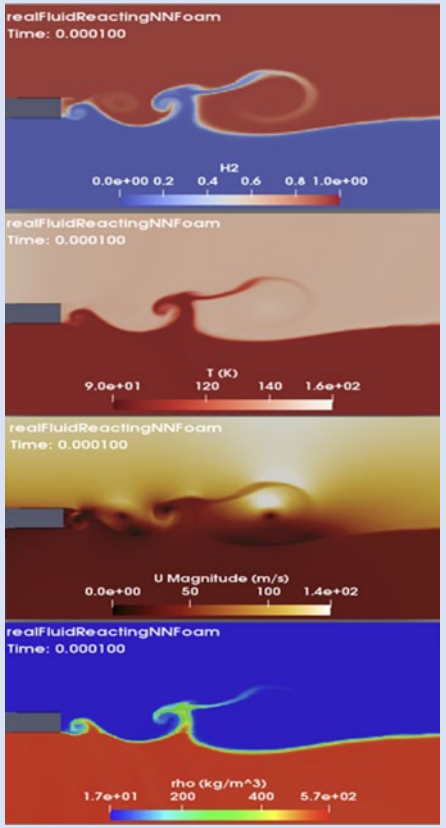
realFluidReactingNNFoam_1
NN_1



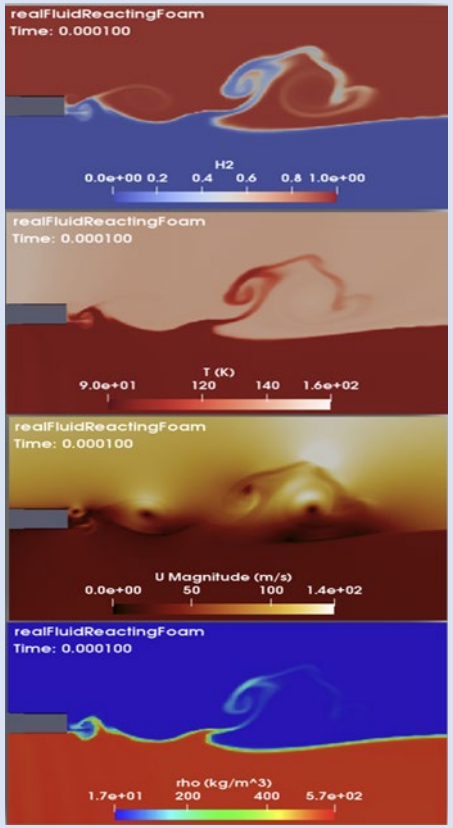
realFluidReactingNNFoam_2
NN_2



realFluidReactingNNFoam



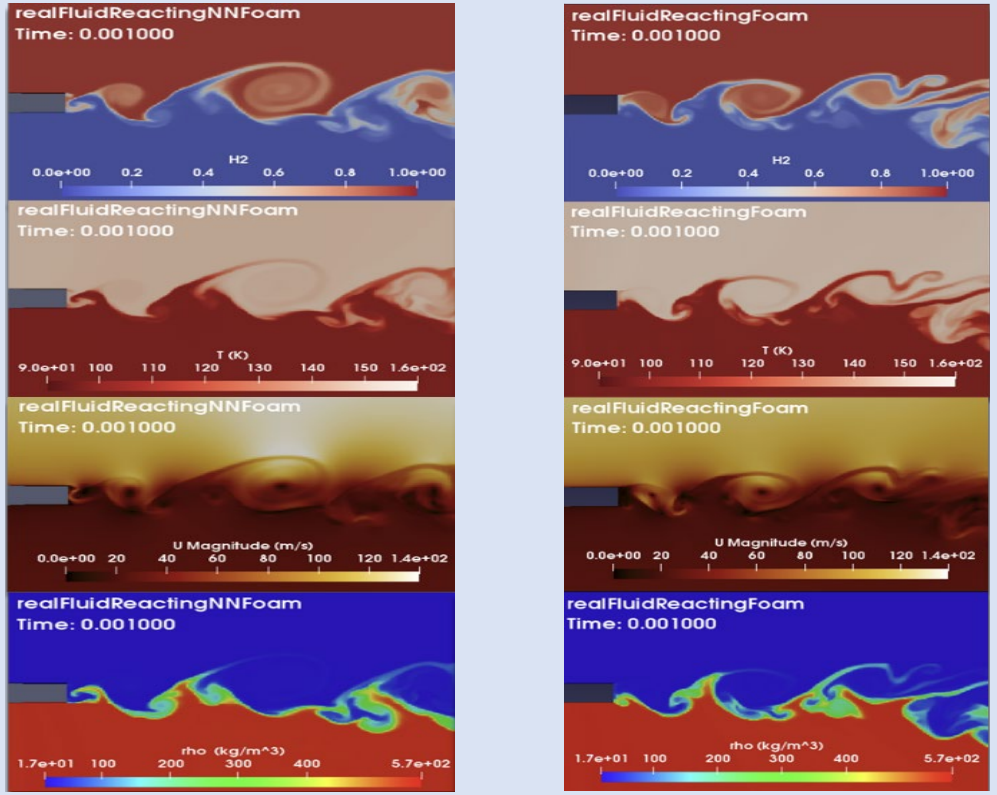
realFluidReactingFoam



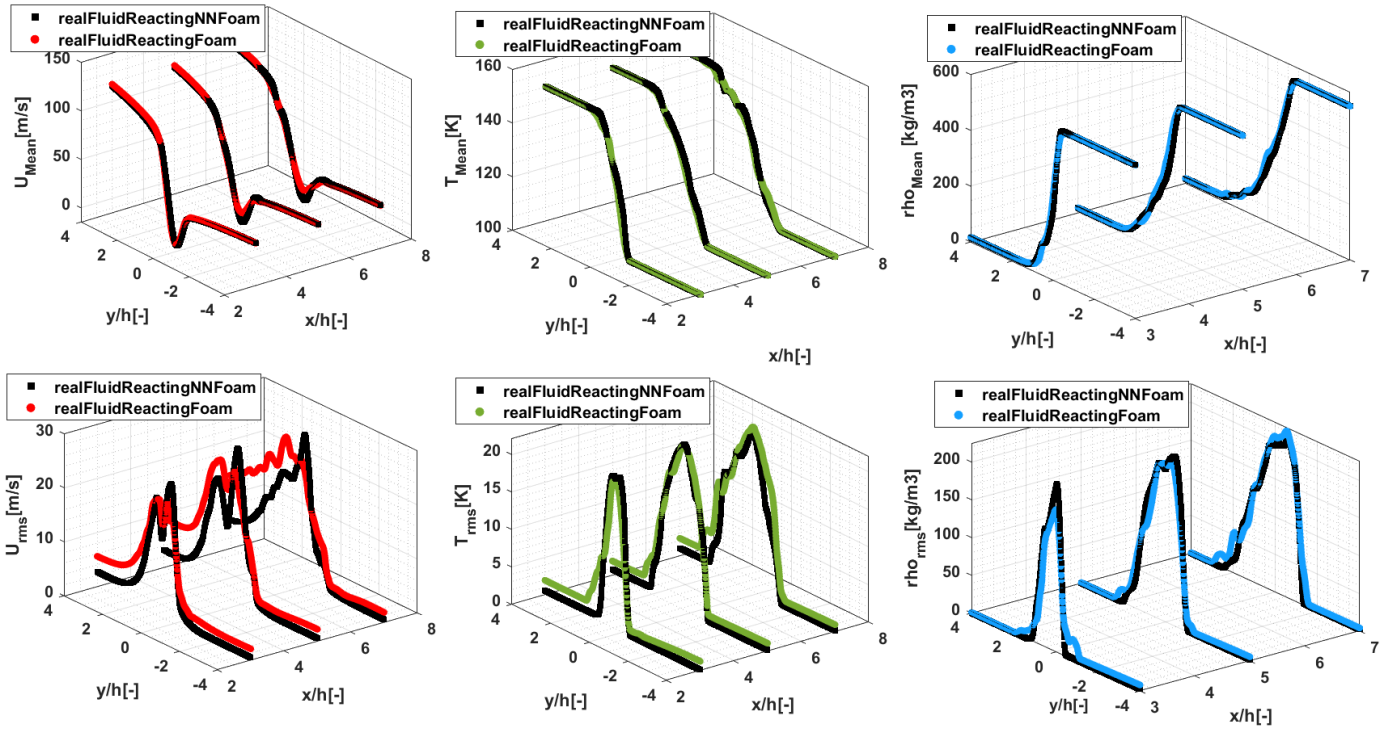
GH2-LOX - Results and discussion: fields @ t = 1 ms and time averages

realFluidReactingNNFoam

realFluidReactingFoam



t = 1 ms



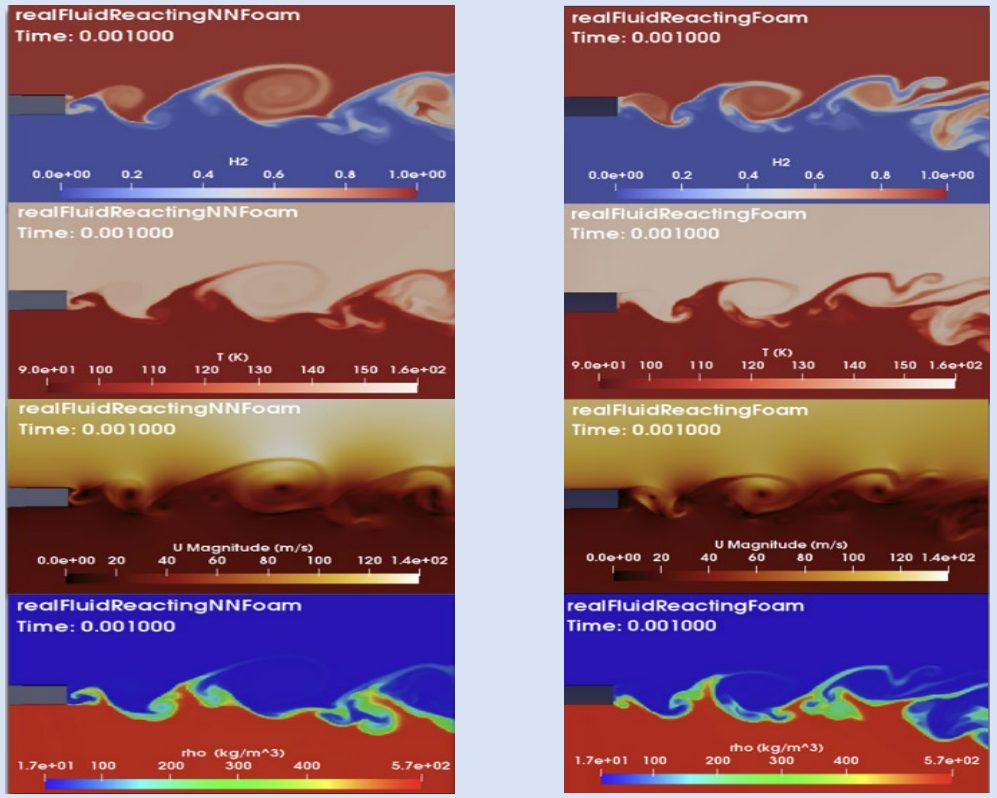
Averages over 4 FTT (1-1.2 ms)

N. Sahranavardfard, D. Aubagnac-Karkar, C. Habchi, G. Costante, F. N. Z. Rahantamialisoa, and M. Battistoni. “Computation of real-fluid thermophysical properties using a neural network approach implemented in OpenFOAM”, Fluids, MDPI, 2024

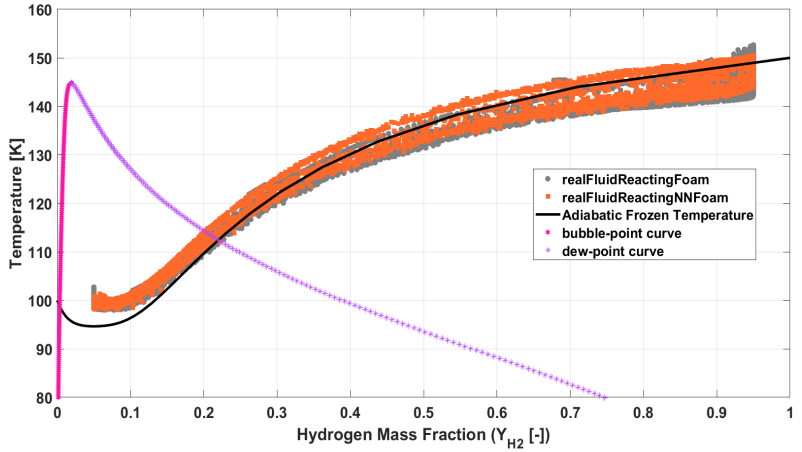
GH2-LOX - Results and discussion: fields @ t = 1 ms and time averages

realFluidReactingNNFoam

realFluidReactingFoam



t = 1 ms



Solver	Total execution			Iteration	
	Time (s)	Speed-up	Total number of Iterations	Time/Iter (s/Iter)	Speed-up
realFluidReactingFoam	511096	1	200550	2.548	1
realFluidReactingNNFoam	179317	2.850	523200	0.343	7.5

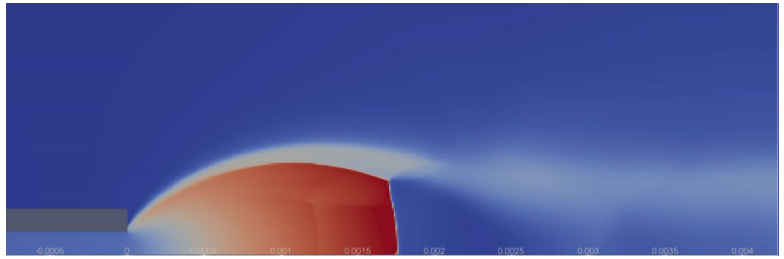
Conclusion: summary and future works

Summary:

- ✓ Exploration of the integration of NN models within the RFM approach for efficiently simulating complex thermodynamic functions in openFoam for a binary mixture under transcritical conditions.
- ✓ realFluidreactingFoam results indicates that the NN model outperforms the original approach in terms of execturion time at the same level of accuracy

Future work

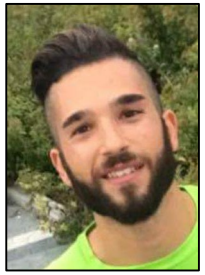
- ✓ The study opens avenues for exploring the full potential of ML techniques in addressing complex problems related to multiphase flows also
- ✓ also with more than just two species
- ✓ or with other highly-volatile fuels (zero and low carbon fuels)



Michele Battistoni
Assoc. Professor

Carlo N. Grimaldi
Full Professor

Jacopo Zembi
Researcher



Federico Ricci
Postdoc

Faniry Rahantamialisoa
Postdoc

Nasrin Sahranavardfard
PhD student

Tamara Gammaidoni
Research Fellow

Filippo Bittoni
PhD student

Thank you