Physics-ML 開発のためのフレームワーク NVIDIA Modulus の紹介

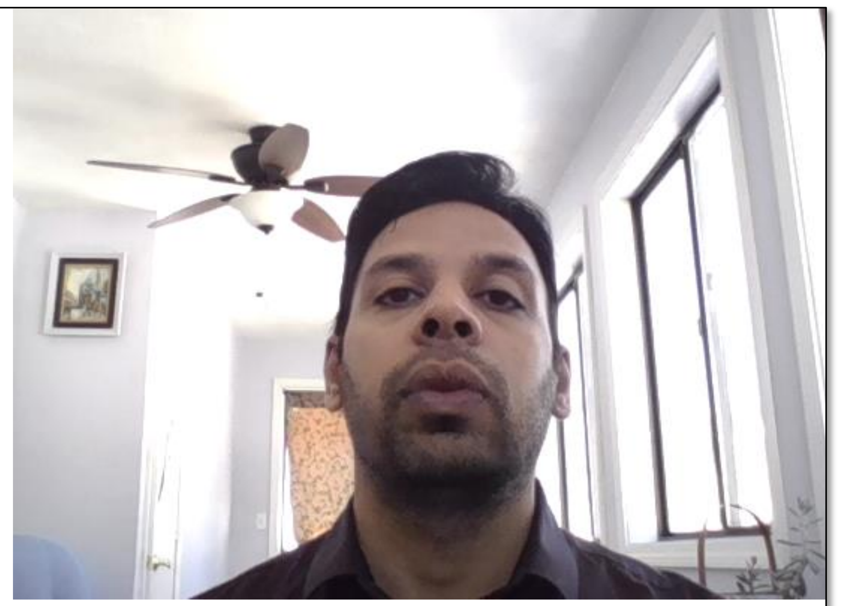
Naruhiko Tan, Solution Architect

Agenda

- What is NVIDIA Modulus
- Use cases
- What's new and online resources

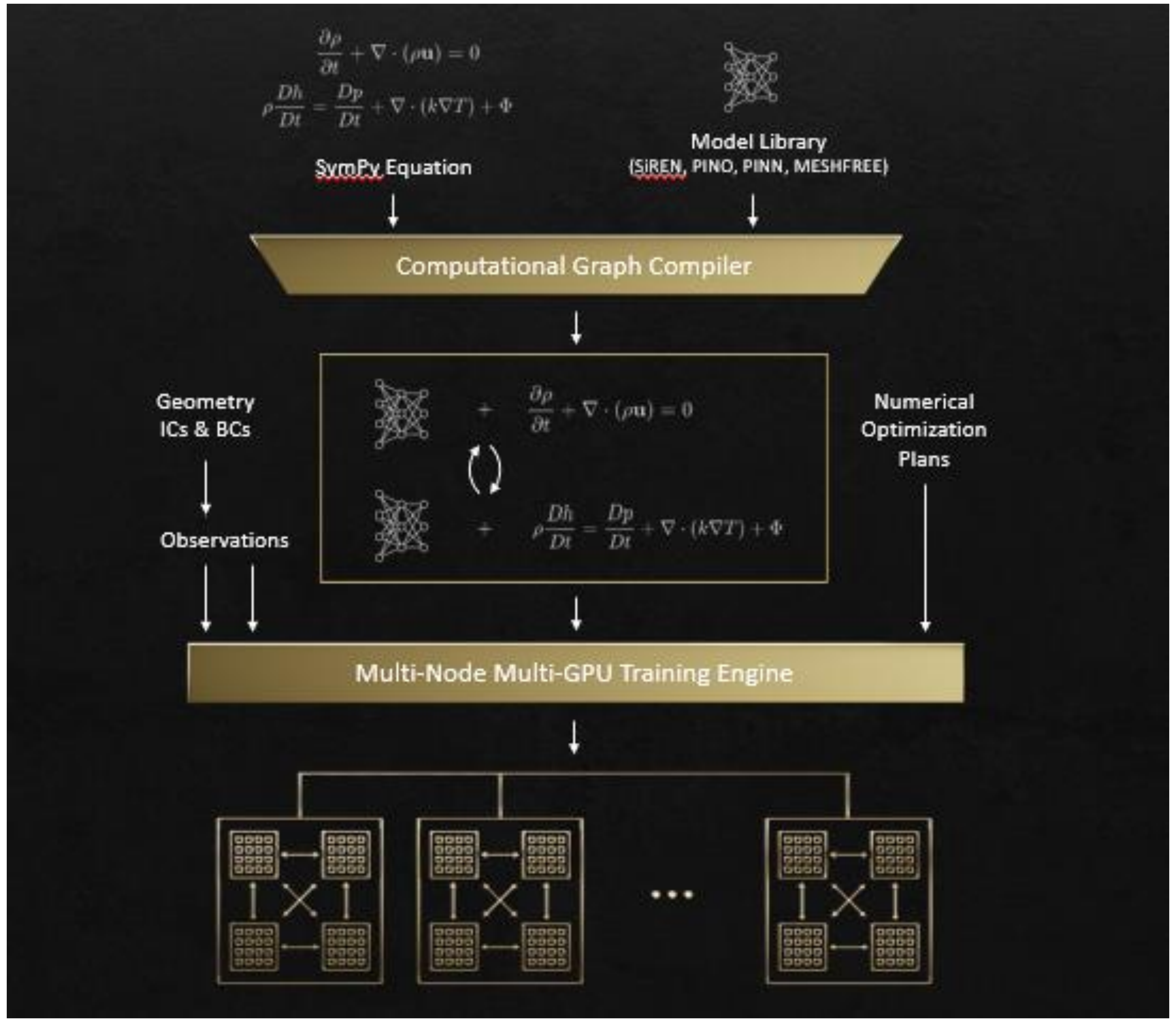
What is NVIDIA Modulus

Getting started with ai for engineering simulations using modulus on rescale platform [S42087]



NVIDIA MODULUS

Framework for Developing Physics ML Models for Digital Twins



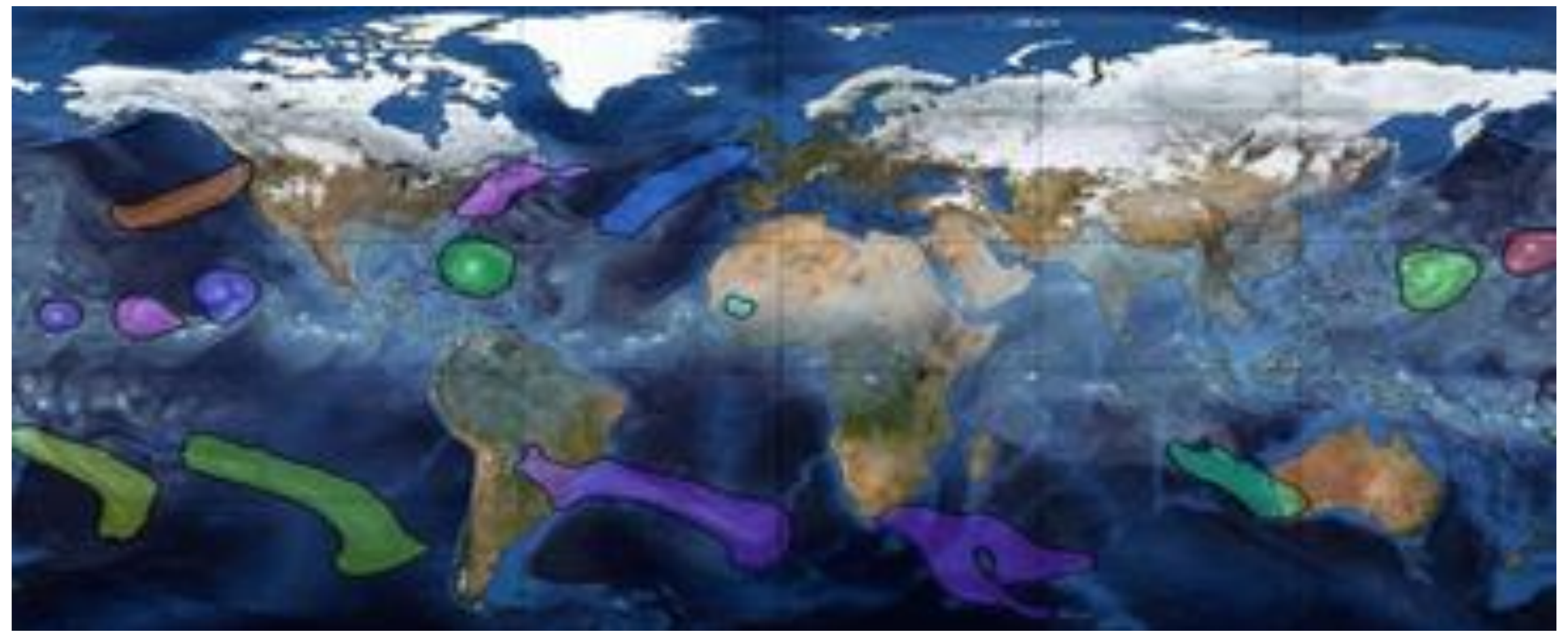
- Use simulation and observation data and governing physics equations to generate a robust surrogate model
- Generalizes parameterized domain and physics to encapsulate multiple configurations/scenarios in the trained model
- Builds a Physics ML model/digital twin to iterate on the design/operating space
- Forward simulation, inverse and data assimilation problems
- What its not? Not a Solver, Not a Simulation platform

NVIDIA

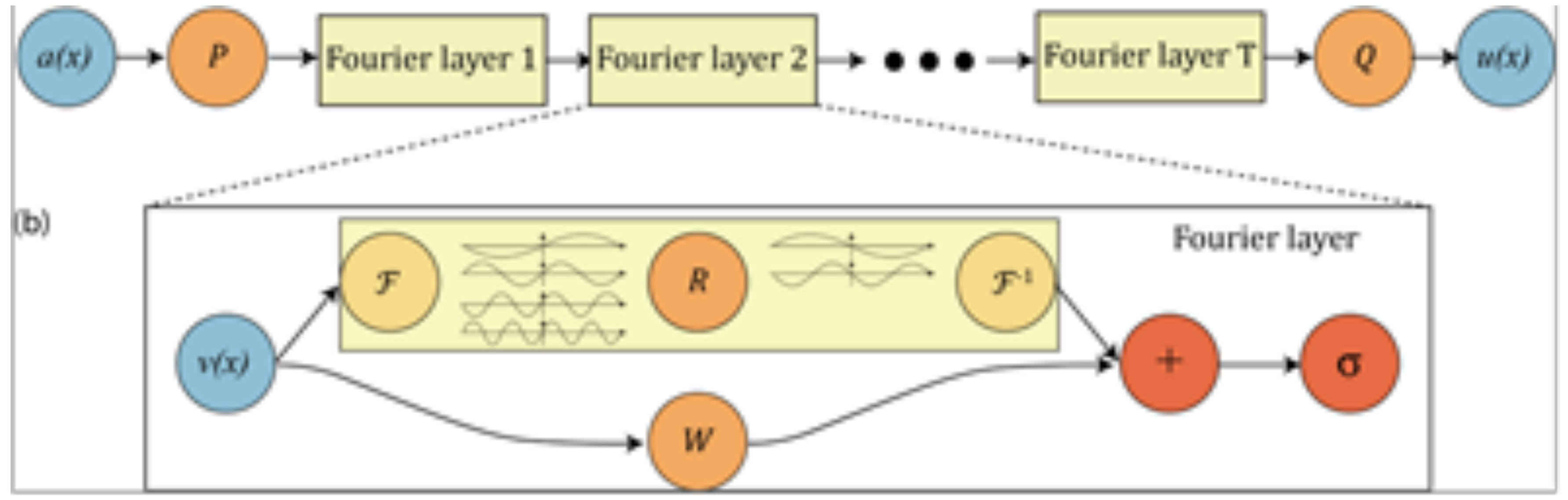
[Developing digital twins for weather, climate, and energy [S41823]]

Physics-ML categorization

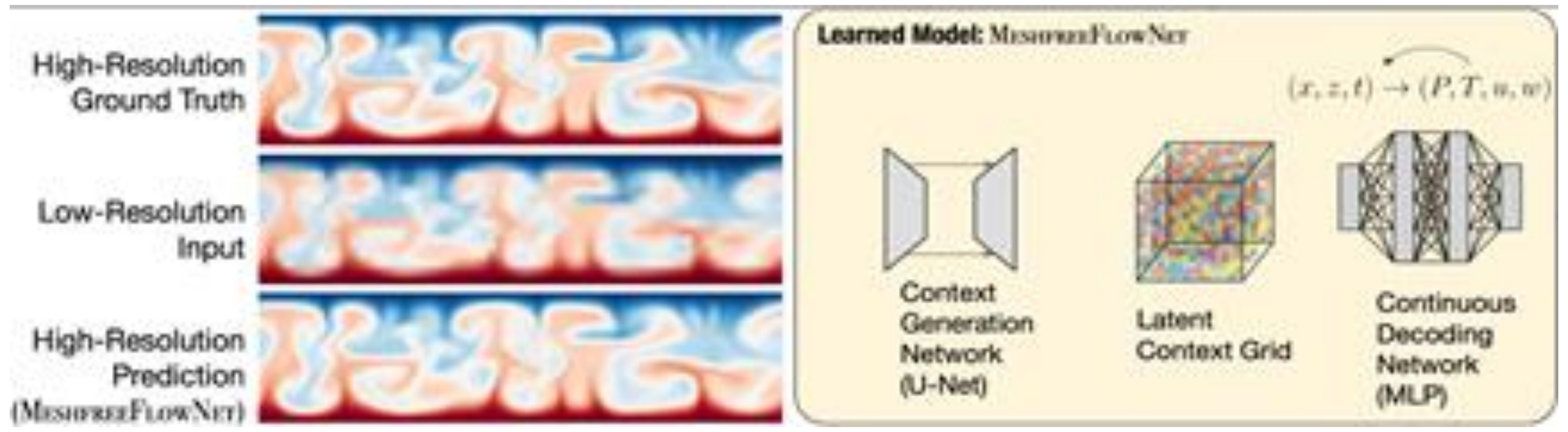
Fully data driven



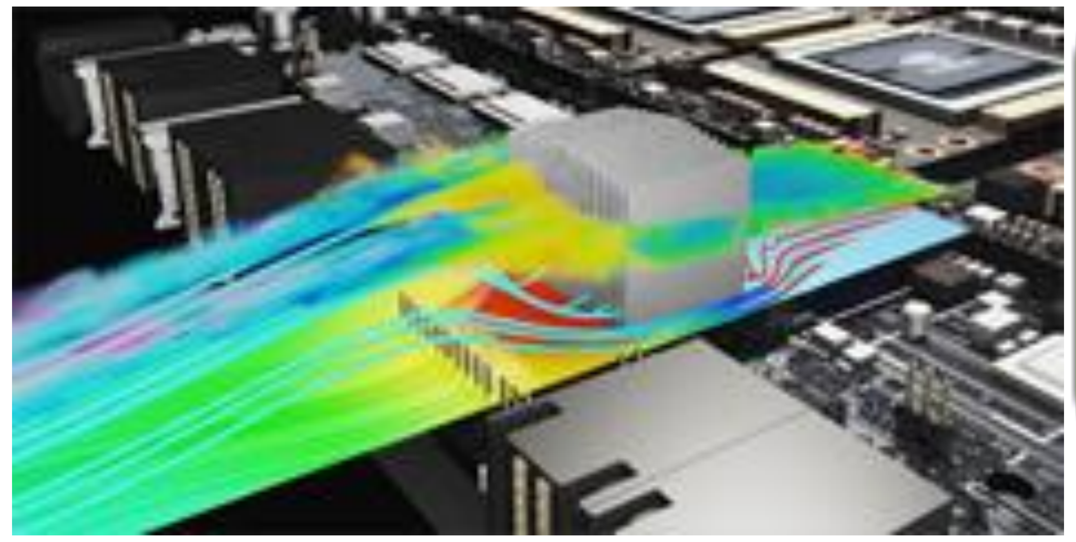
Inductive bias

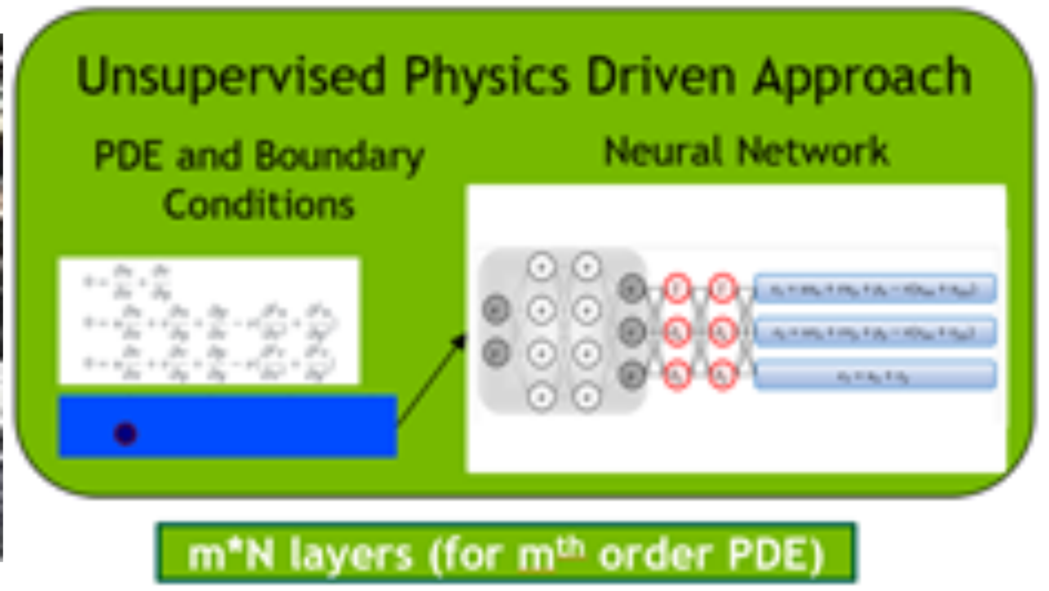


Physics constrained



Fully physics driven





Data

Physics

NVIDIA Modulus – A Framework

Key concepts

Specifying geometry of the domain

- STL or Constructive Solid Geometry (CSG)
- Specify sampling policy
- Specify parameterization

Using ground truth data in Modulus

- Observed data or simulation data
- Use only the governing equations with no data
- Use only data
- Use both
  - Consistent with first principles
  - Faster convergence

Network architectures – Curated networks

- Fourier Features (FN), Sinusoidal Representation (SiReNs), Modified Fourier Features (mFN)
- Fourier Neural Operator, Adaptive Fourier Neural Operator
- Fully Connected (FC)
- Deep Galerkin Method (DGM)
- Modified Highway Networks
- Multiplicative Filter Networks

and more…

Case Study on Developing Digital Twins for the Power Industry using Modulus and Omniverse [S41671]

NVIDIA MODULUS - A FRAMEWORK

Key Concepts

Specify governing equations

- Symbolic equations
- Integral form or weak form
- extensible

$$\mathbf{P} : \begin{cases} \frac{\delta^2 u}{\delta x^2}(x) = f(x), \\ u(0) = u(1) = 0, \end{cases}$$

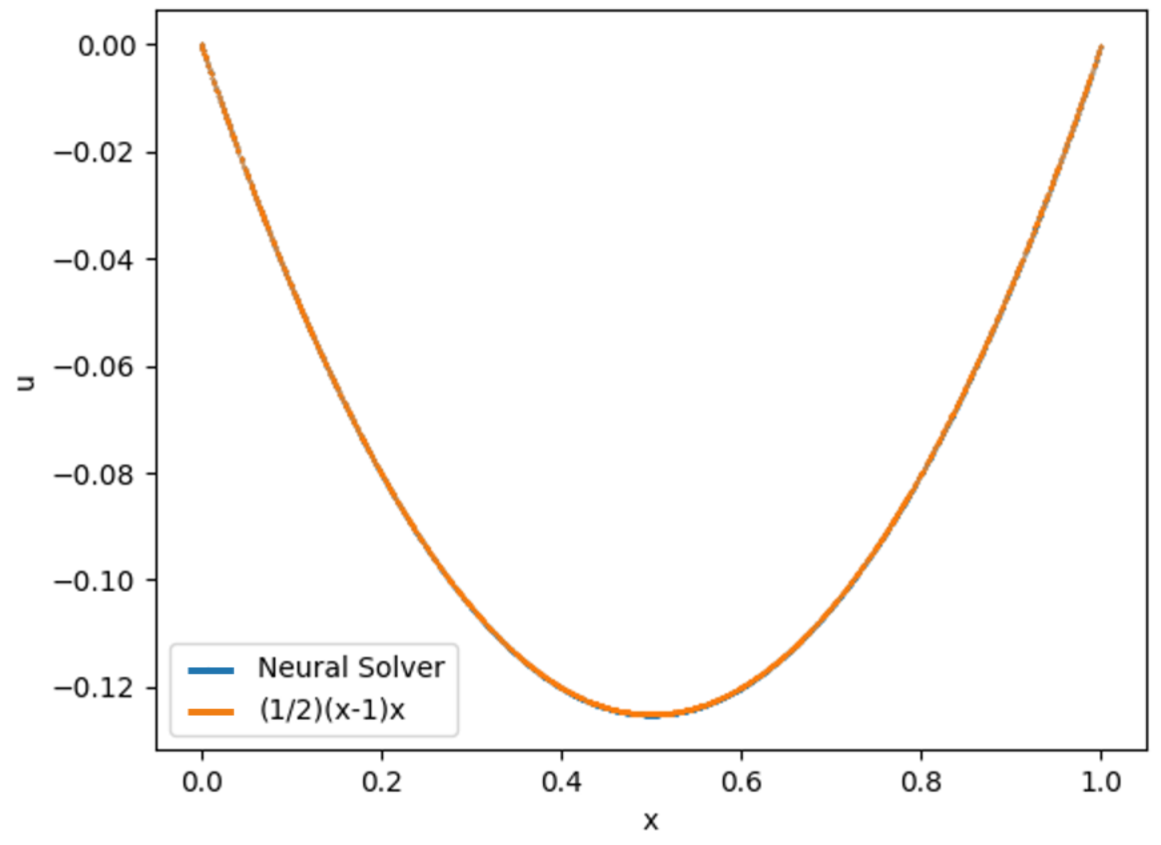
$$L_{residual} = \int_0^1 \left(\frac{\delta^2 u_{net}}{\delta x^2}(x) - f(x) \right)^2 dx$$

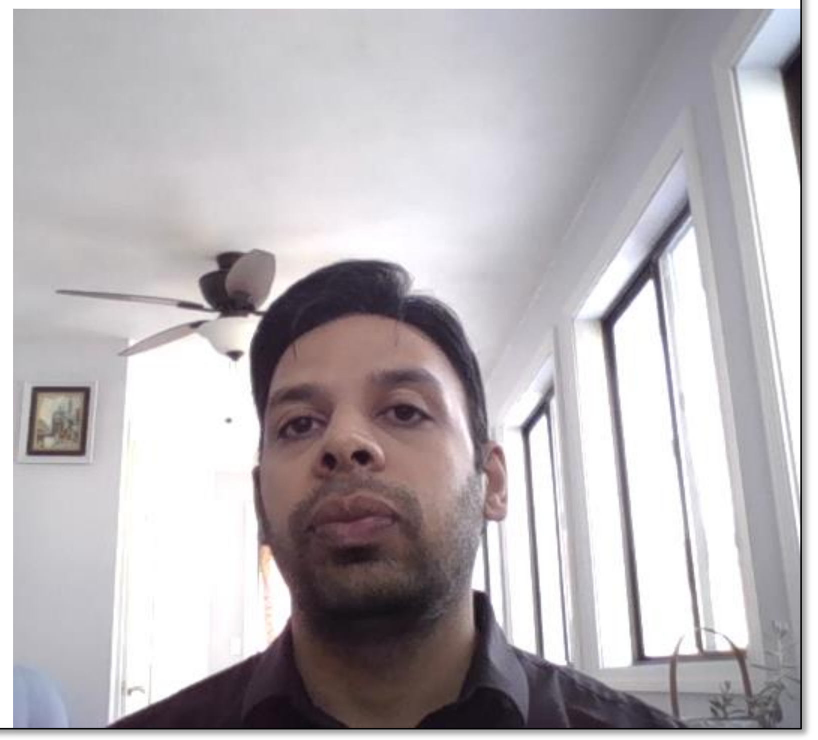
$$L_{residual} = \frac{1}{N} \sum_{i=0}^{N} \left(\frac{\delta^2 u_{net}}{\delta x^2}(x_i) - f(x_i) \right)^2$$

$$L_{BC} = u_{net}(0)^2 + u_{net}(1)^2$$

Loss function minimization

- Satisfy the PDE
- Boundary conditions





NVIDIA

Case Study on Developing Digital Twins for the Power Industry using Modulus and Omniverse [S41671]

NVIDIA MODULUS - A FRAMEWORK

Key Concepts

Specify governing equations

- Symbolic equations
- Integral form or weak form
- extensible

$$\mathbf{P}: \begin{cases} \frac{\delta^2 u}{\delta x^2}(x) = f(x), \\ u(0) = u(l) = 0, \end{cases}$$

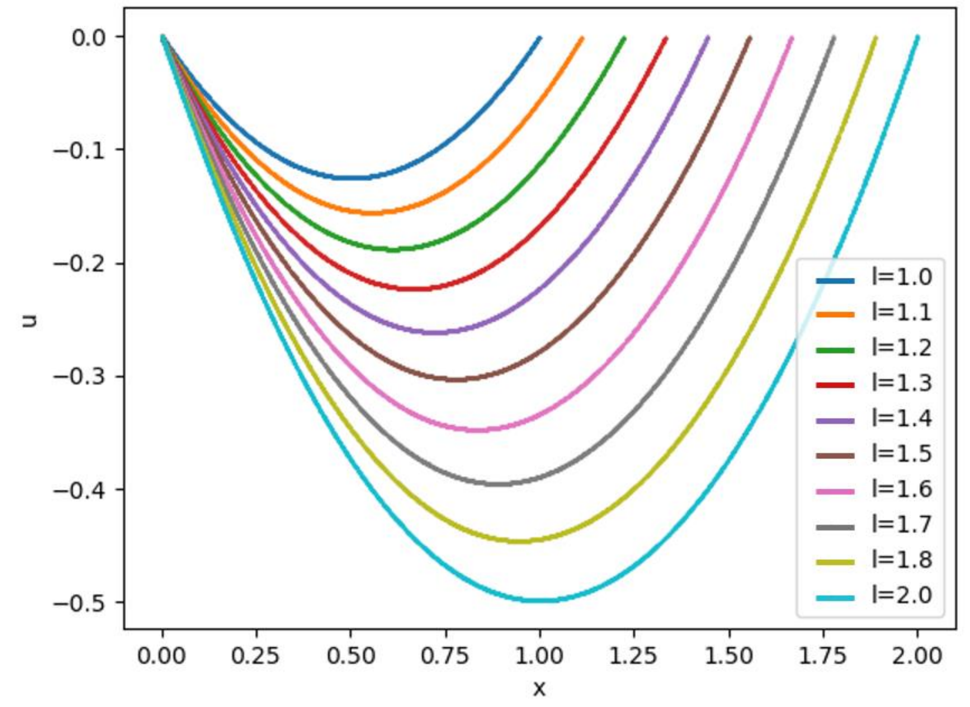
$$L_{residual} = \int_1^2 \int_0^l \left(\frac{\delta^2 u_{net}}{\delta x^2}(x, l) - f(x) \right)^2 dx dl \approx \left(\int_1^2 \int_0^l dx dl \right) \frac{1}{N} \sum_{i=0}^{N} \left(\frac{\delta^2 u_{net}}{\delta x^2}(x_i, l_i) - f(x_i) \right)^2$$

$$L_{BC} = \int_1^2 (u_{net}(0, l))^2 + (u_{net}(l, l) dl \approx \left(\int_1^2 dl \right) \frac{1}{N} \sum_{i=0}^{N} (u_{net}(0, l_i))^2 + (u_{net}(l_i, l_i))^2$$

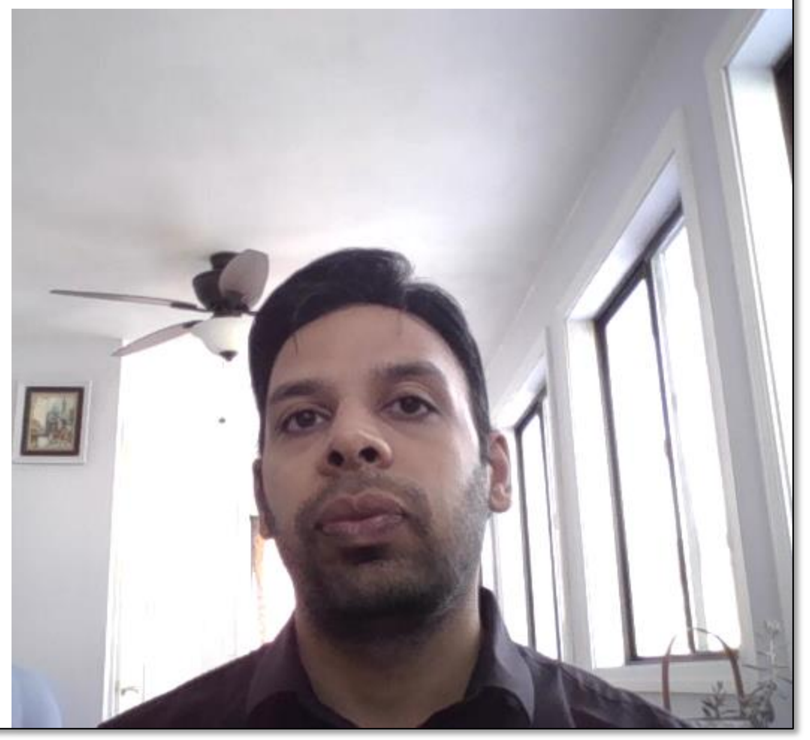
Loss function minimization

- Satisfy the PDE
- Boundary conditions

Explicit Parameterization



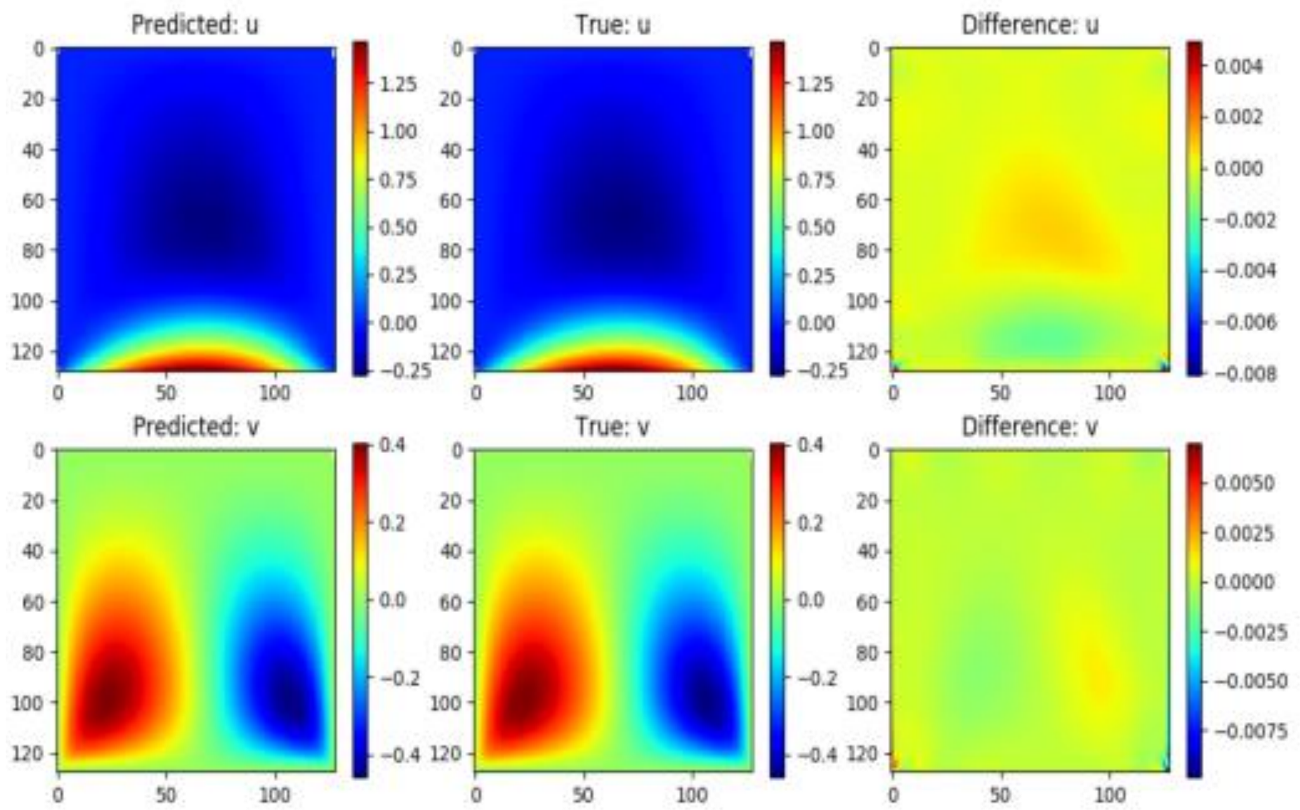
S41823: Towards developing digital twins for weather, climate, and energy



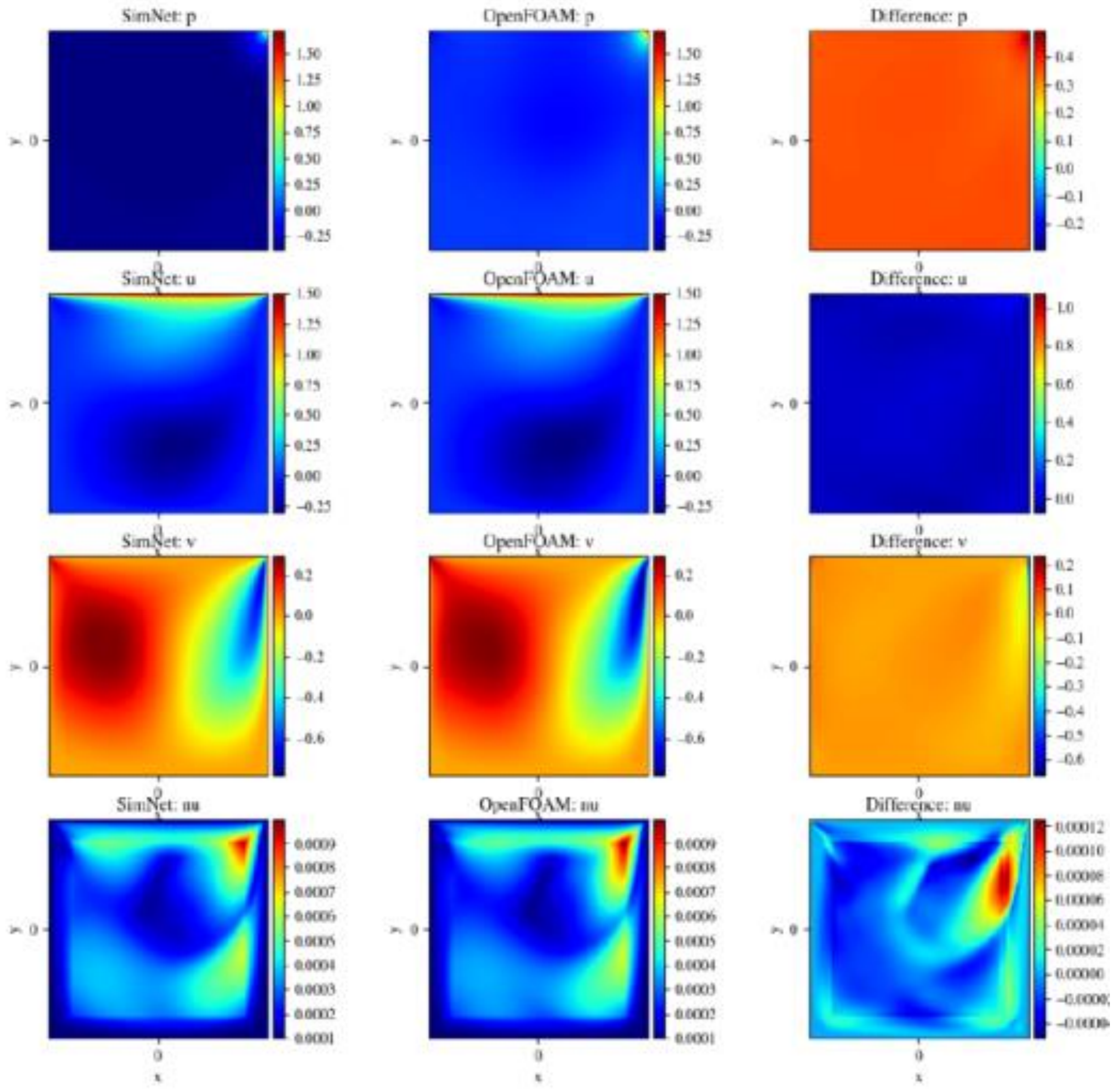
NVIDIA

Modulus Framework - Verification

CFD

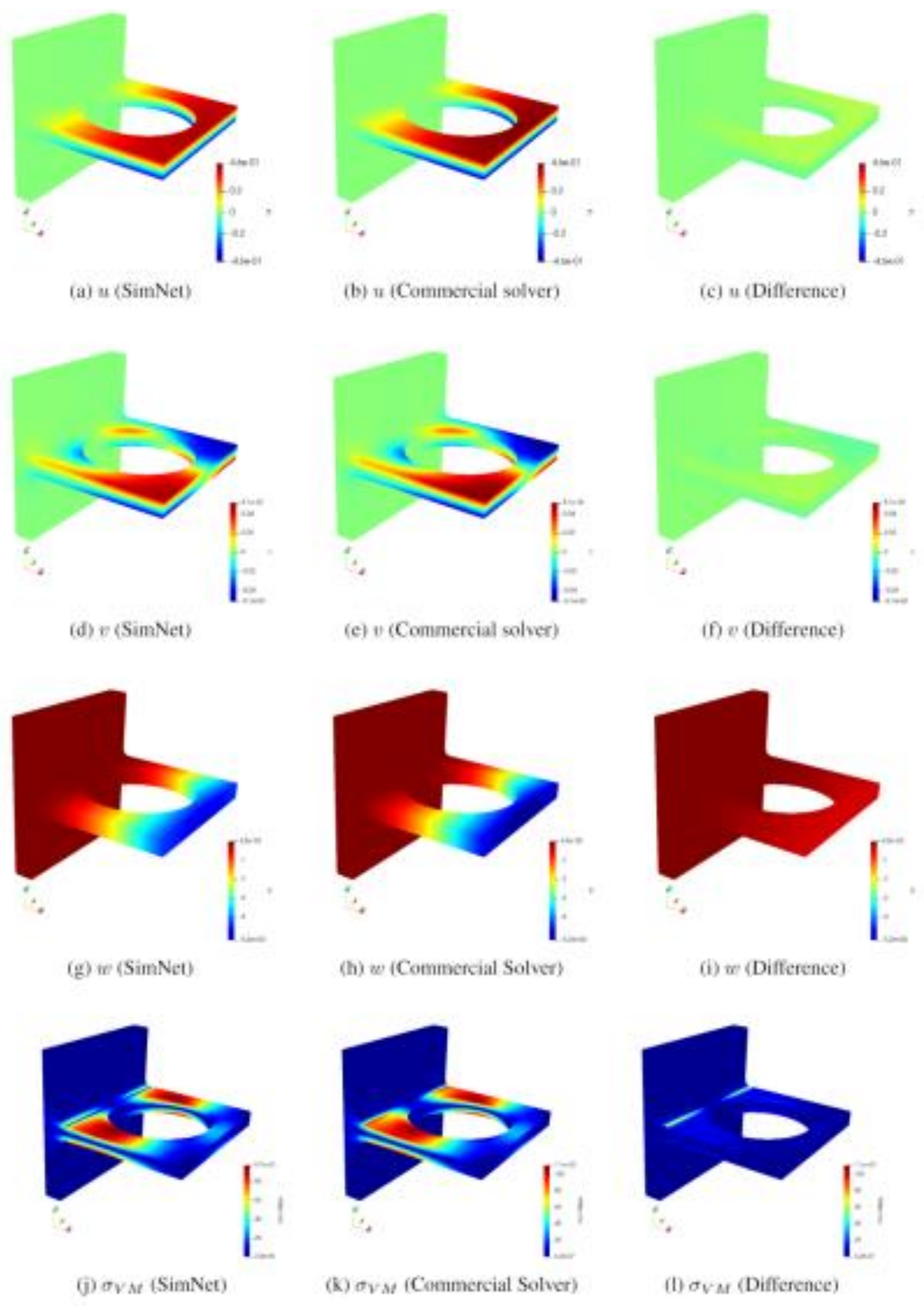


Laminar



Turbulent

Solid Mechanics

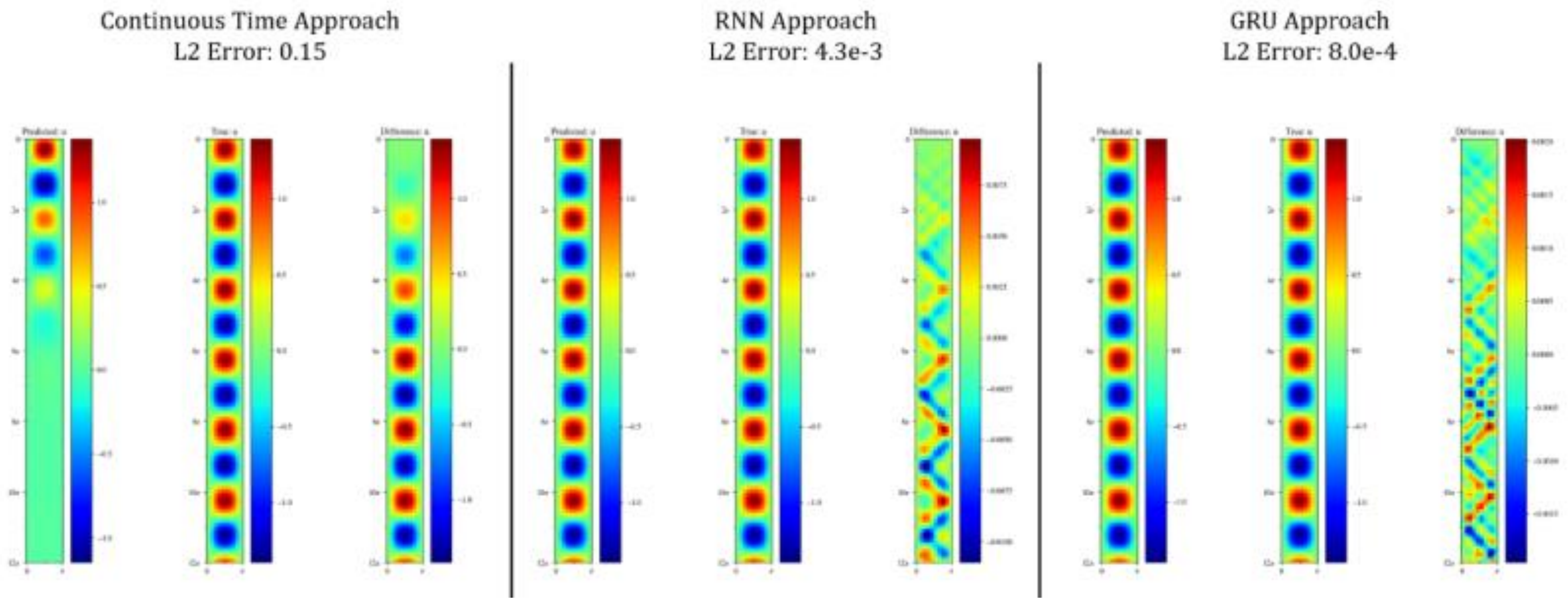


Equilibrium: $\sigma_{ji,j} + f_i = 0,$

Stress-Strain: $\sigma_{ij} = \lambda \epsilon_{kk} \delta_{ij} + 2\mu \epsilon_{ij},$

Strain-Displacement: $\epsilon_{ij} = \frac{1}{2}\left(u_{i,j} + u_{j,i}\right).$

Acoustics



$$
\begin{aligned}
u_{tt} &= c^2 u_{xx} \\
u(0,t) &= 0, \\
u(\pi,t) &= 0, \\
u(x,0) &= \sin(x), \\
u_t(x,0) &= \sin(x).
\end{aligned}
$$

Modulus Framework - Verification

Electromagnetics

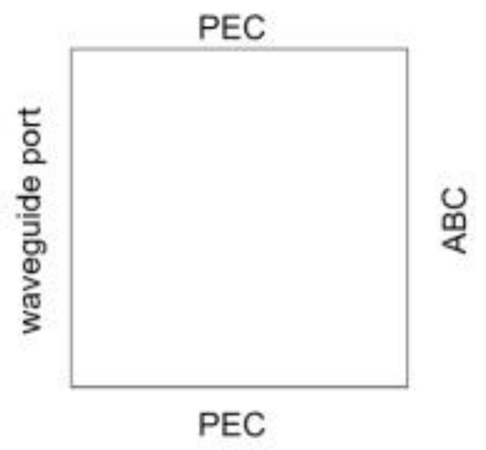


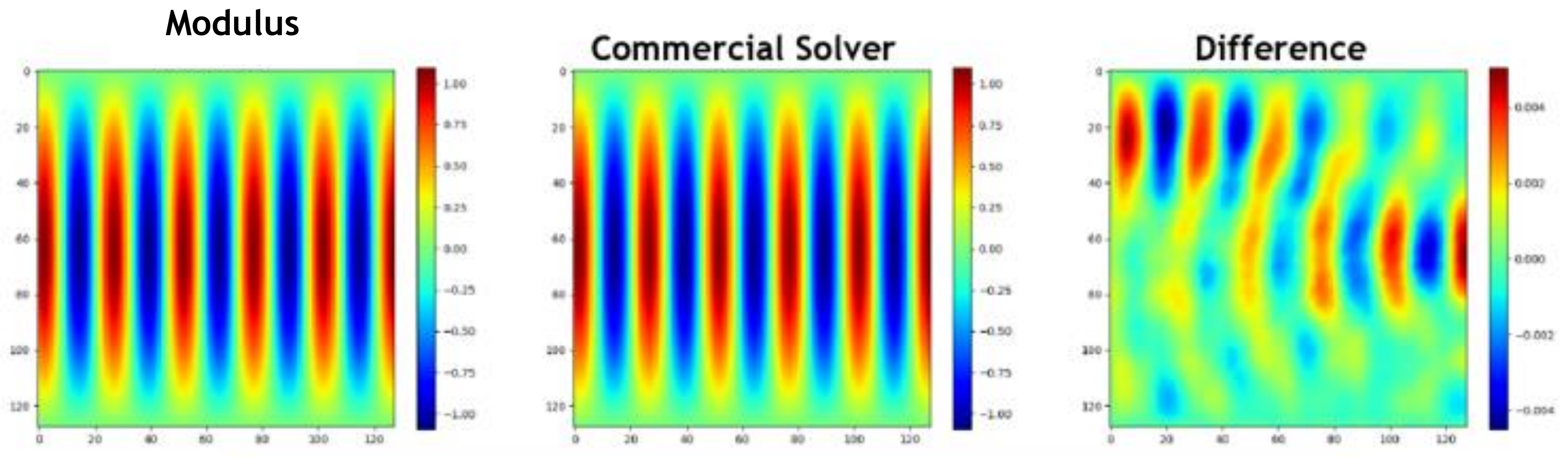
Figure 42: Domain of 2D waveguide

In this example we will solve this waveguide problem by transverse-magnetic (TM_z) mode, so that our unknown variable is $E_z(x, y)$. The governing equation in Ω is

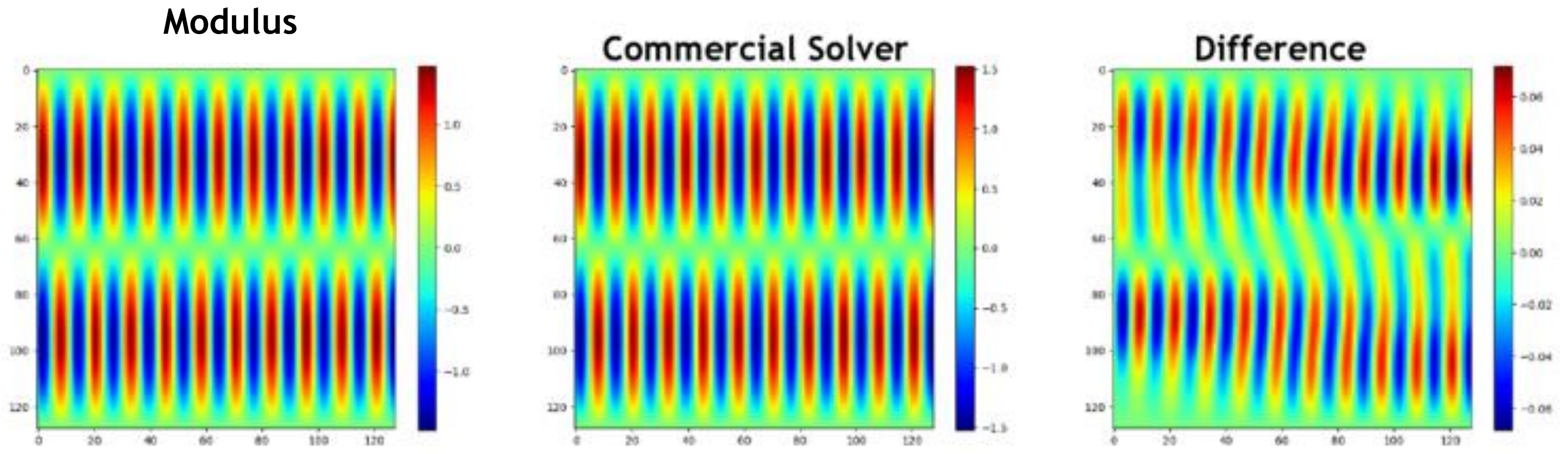
$$\Delta E_z(x, y) + k^2 E_z(x, y) = 0,$$

where k is the wavenumber. Notice in 2D scalar case, the PEC and ABC will be simplified in the following form, respectively:

$$E_z(x, y) = 0 \text{ on top and bottom boundaries, } \frac{\partial E_z}{\partial y} = 0 \text{ on right boundary.}$$



Wave Number = 16, Eigen Mode = 1

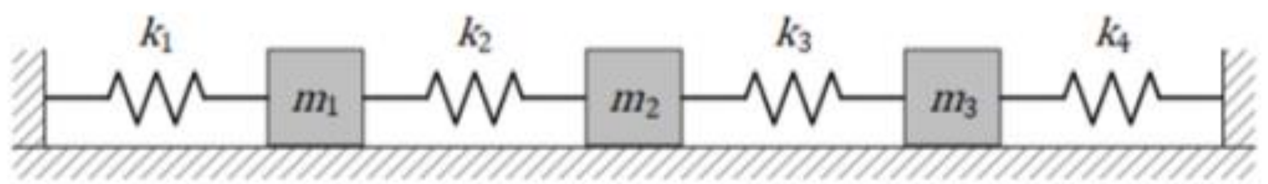


Wave Number = 32, Eigen Mode = 2

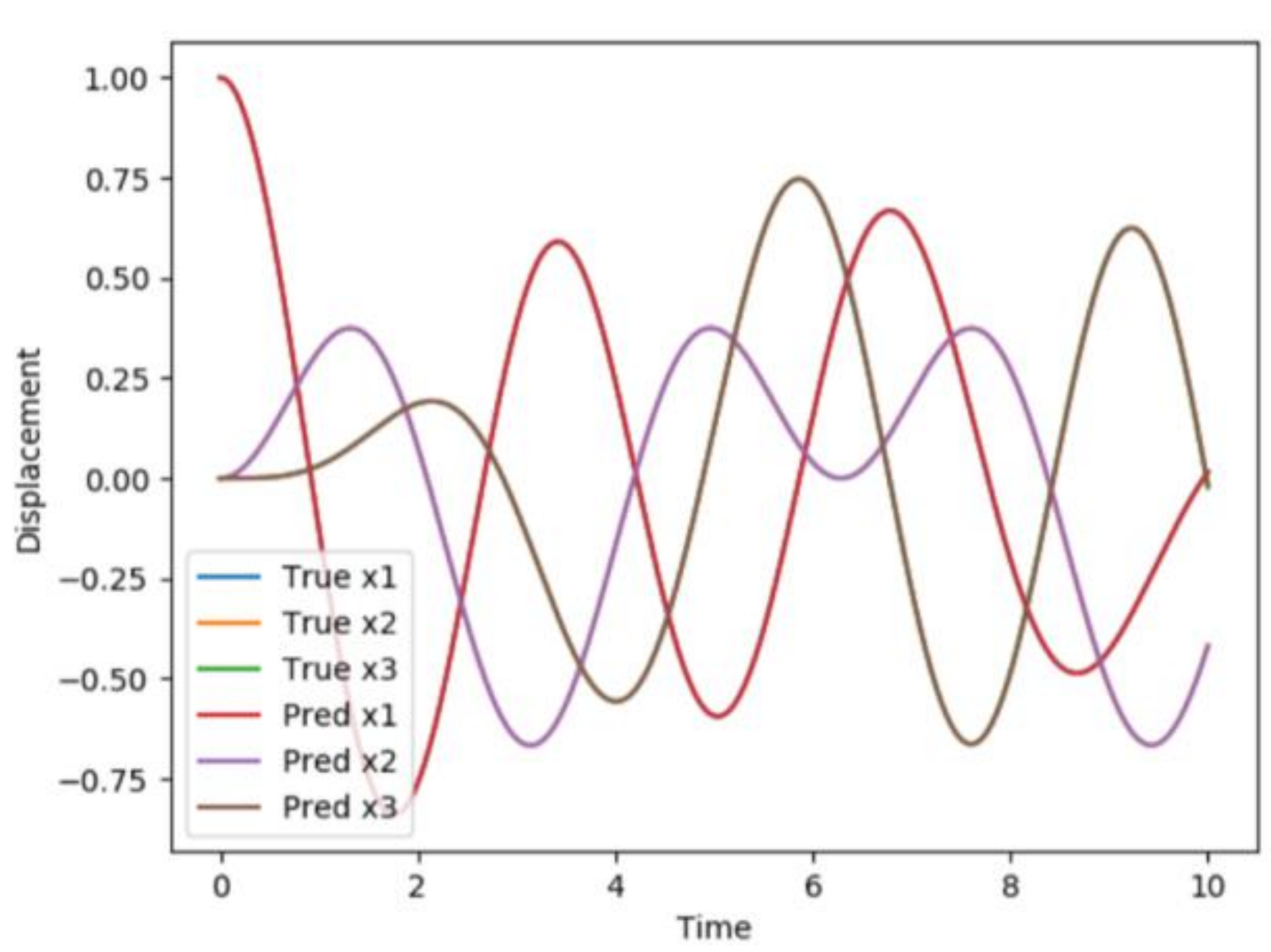
Vibrations

$$
\begin{aligned}
m_1 x_1''(t) &= -k_1 x_1(t) + k_2(x_2(t) - x_1(t)),\\
m_2 x_2''(t) &= -k_2(x_2(t) - x_1(t)) + k_3(x_3(t) - x_2(t)),\\
m_3 x_3''(t) &= -k_3(x_3(t) - x_2(t)) - k_4 x_3(t).
\end{aligned}
$$

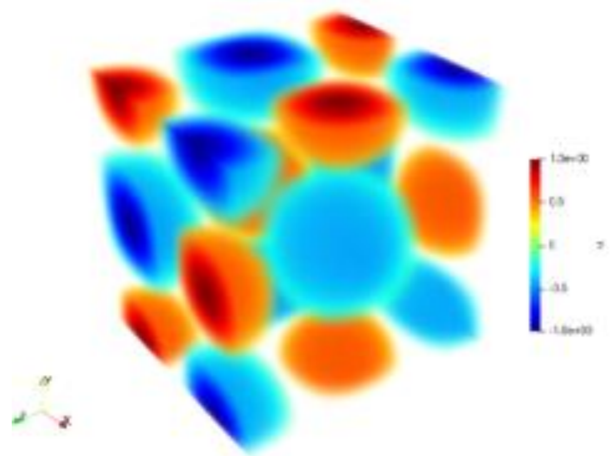
$$
\begin{aligned}
[m_1, m_2, m_3] &= [1, 1, 1],\\
[k_1, k_2, k_3, k_4] &= [2, 1, 1, 2],\\
[x_1(0), x_2(0), x_3(0)] &= [1, 0, 0],\\
[x_1'(0), x_2'(0), x_3'(0)] &= [0, 0, 0].
\end{aligned}
$$



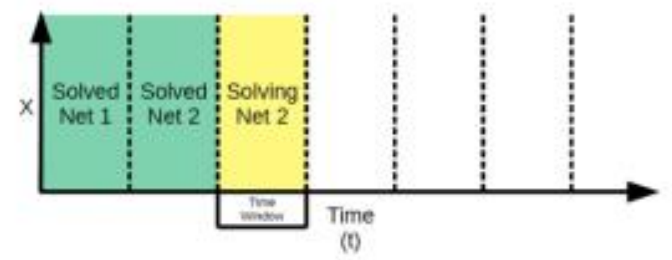
Three masses connected by four springs on a friction-less surface



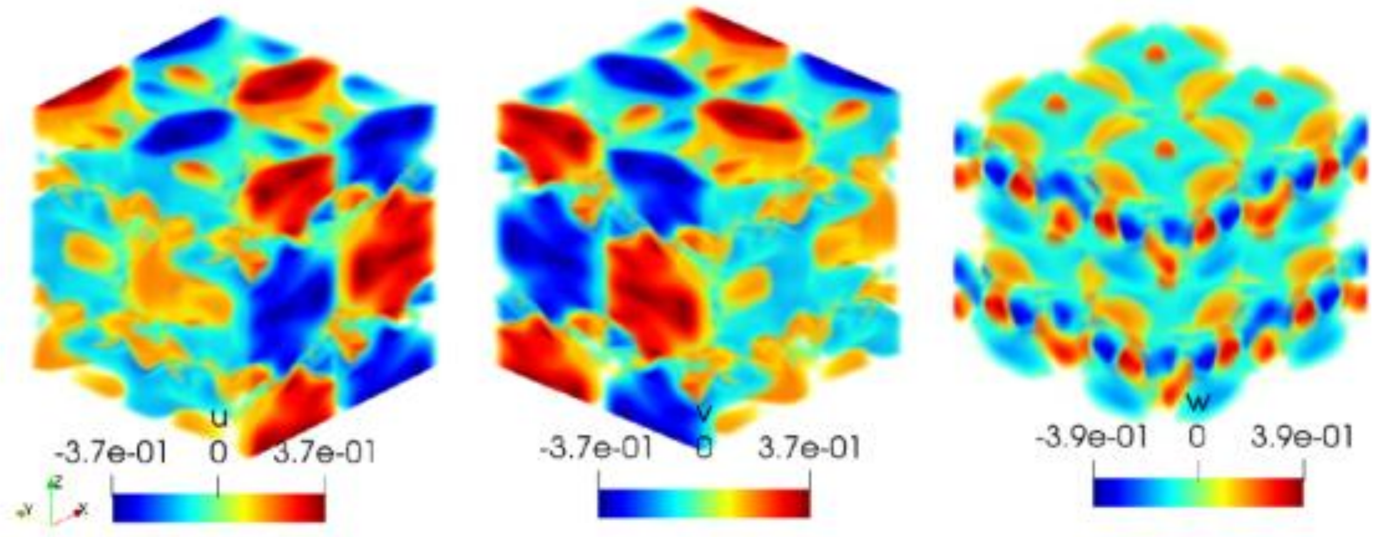
Turbulence

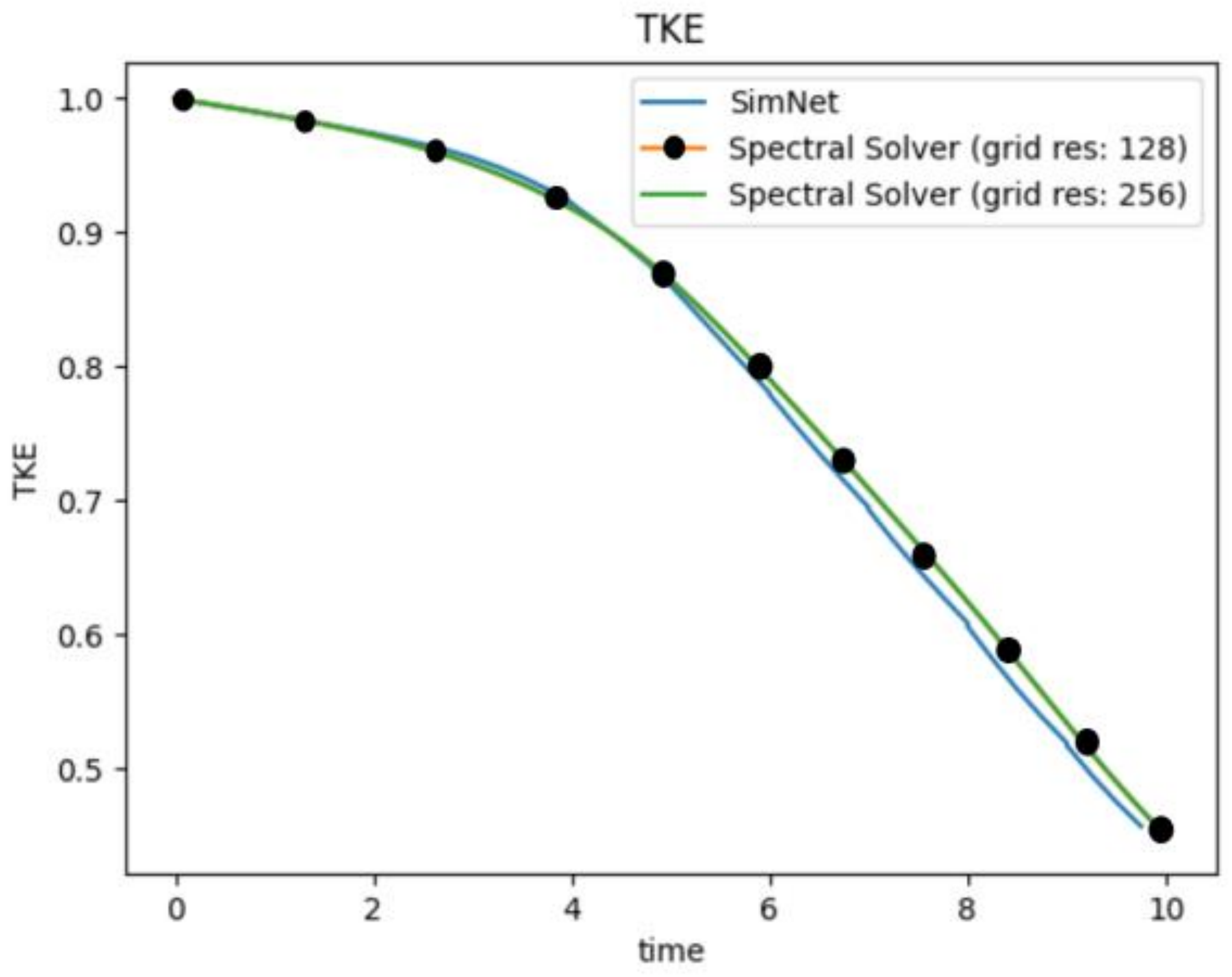


Taylor-Green vortex initial conditions.



Moving Time Window Method.

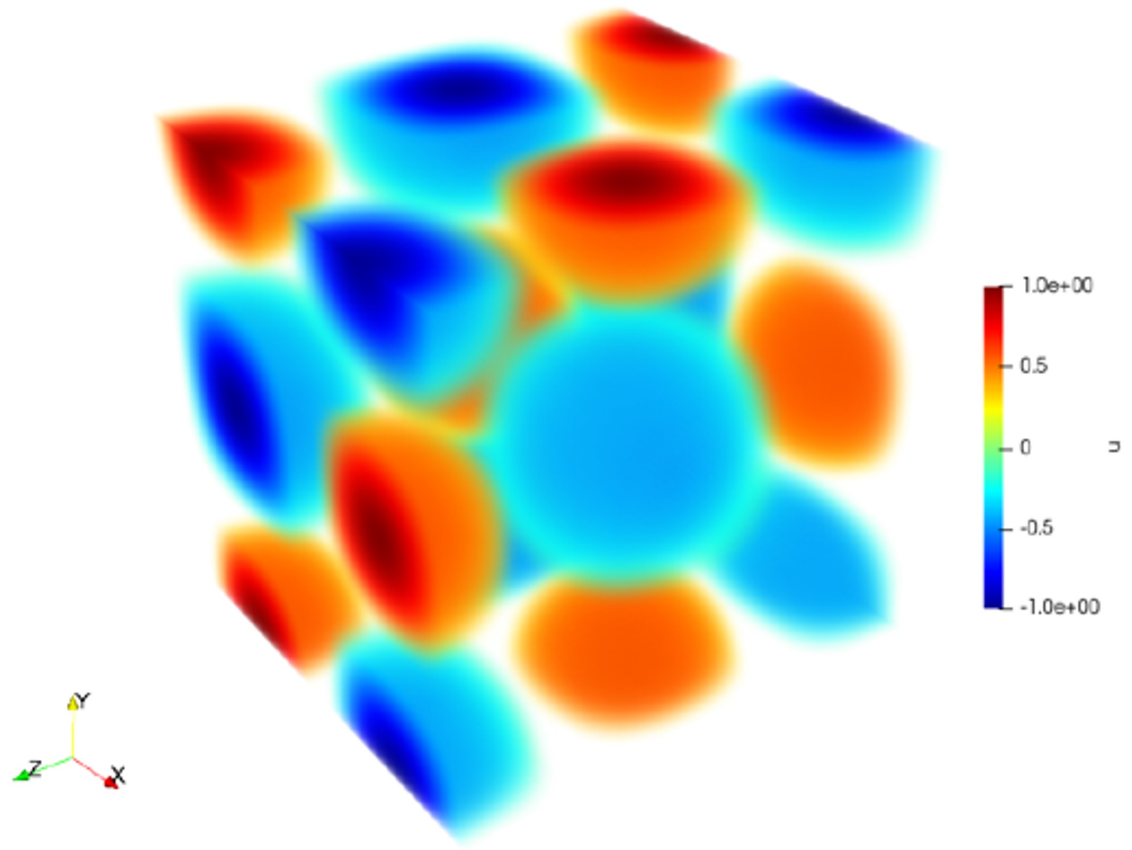




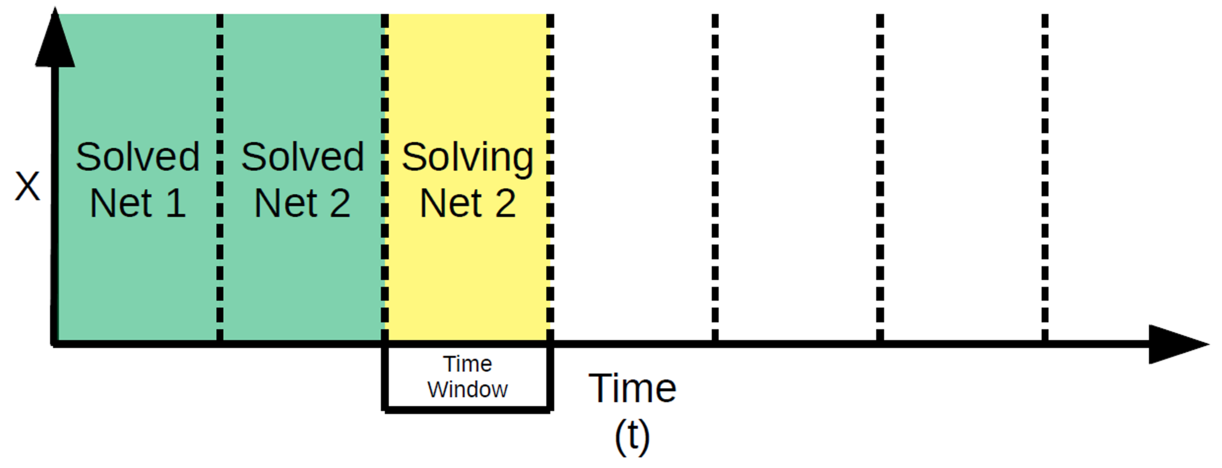
Taylor-Green Trubulent kinetic energy decay.

Modulus Framework - Verification

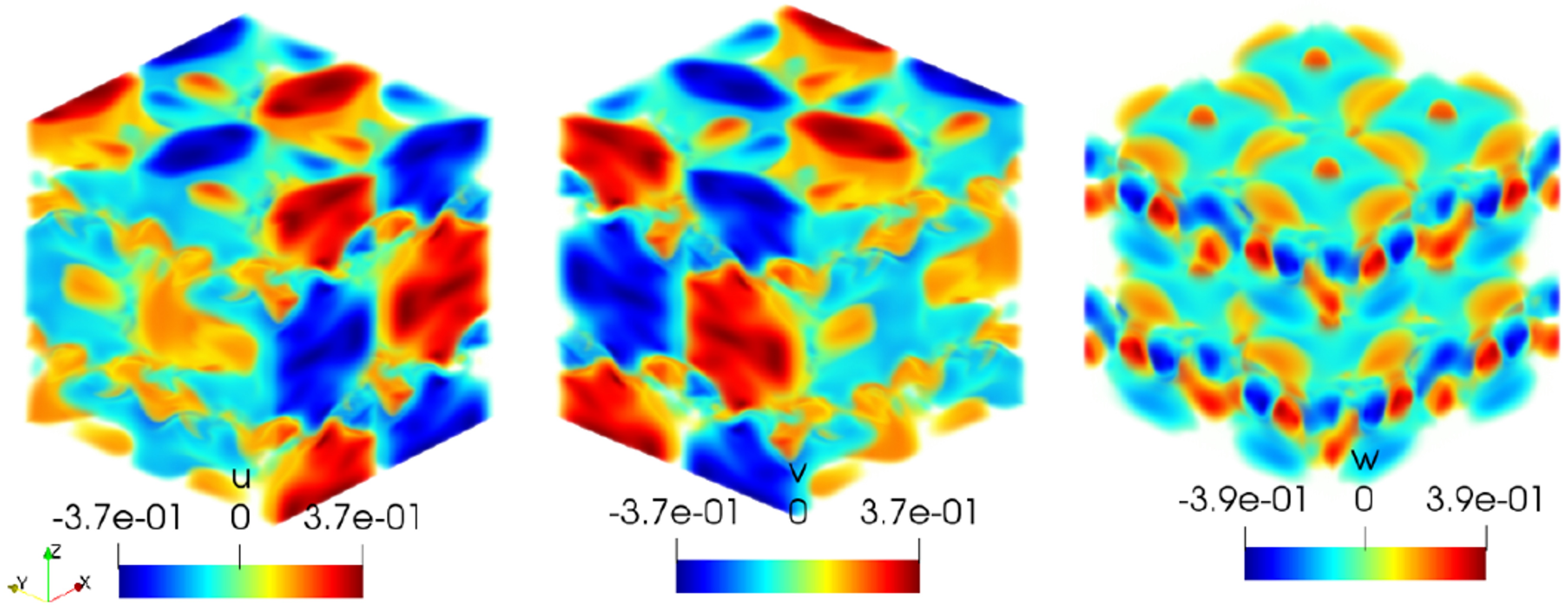
Taylor-Green Vortex Decay



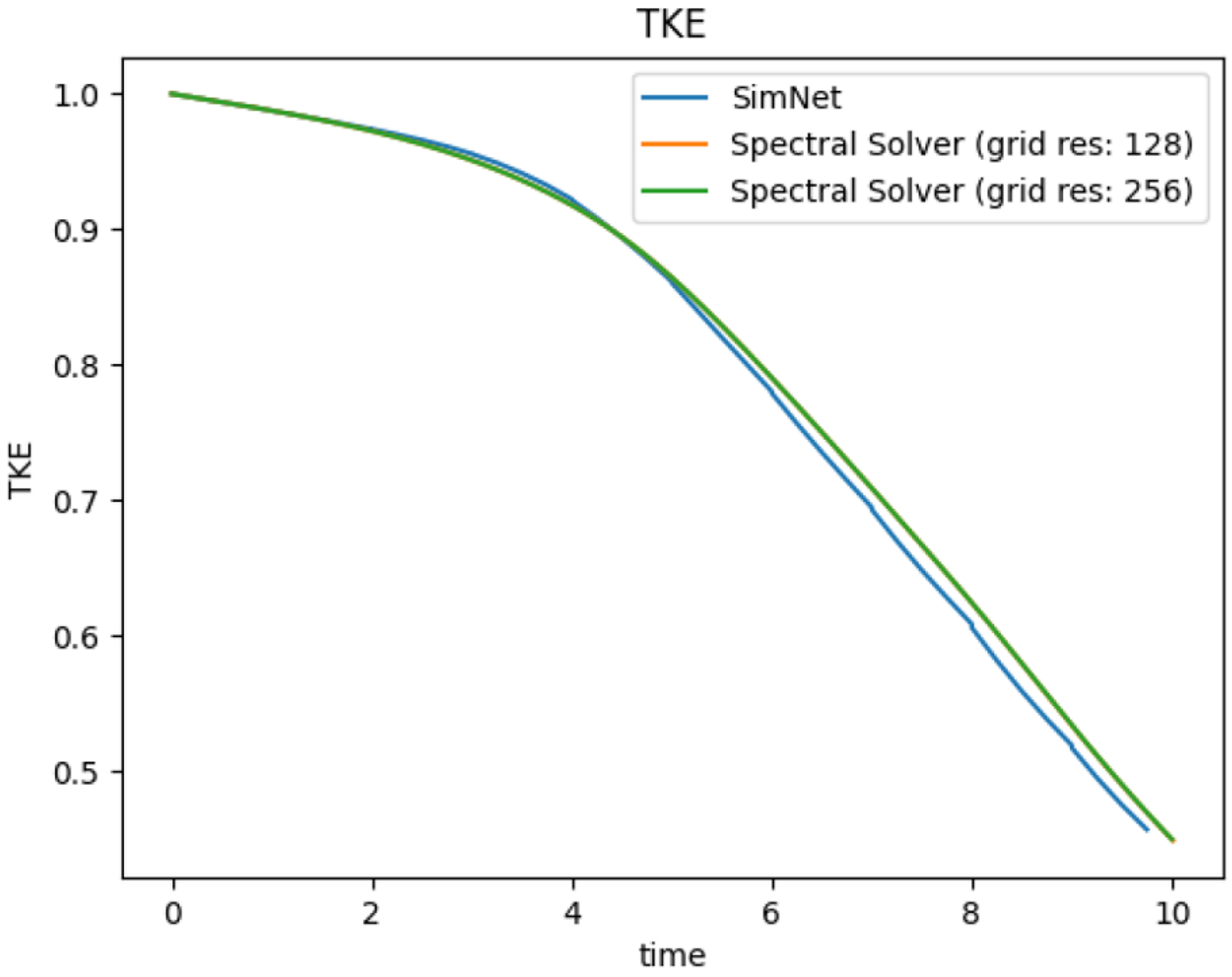
Taylor-Green vortex initial conditions.



Moving Time Window Method.



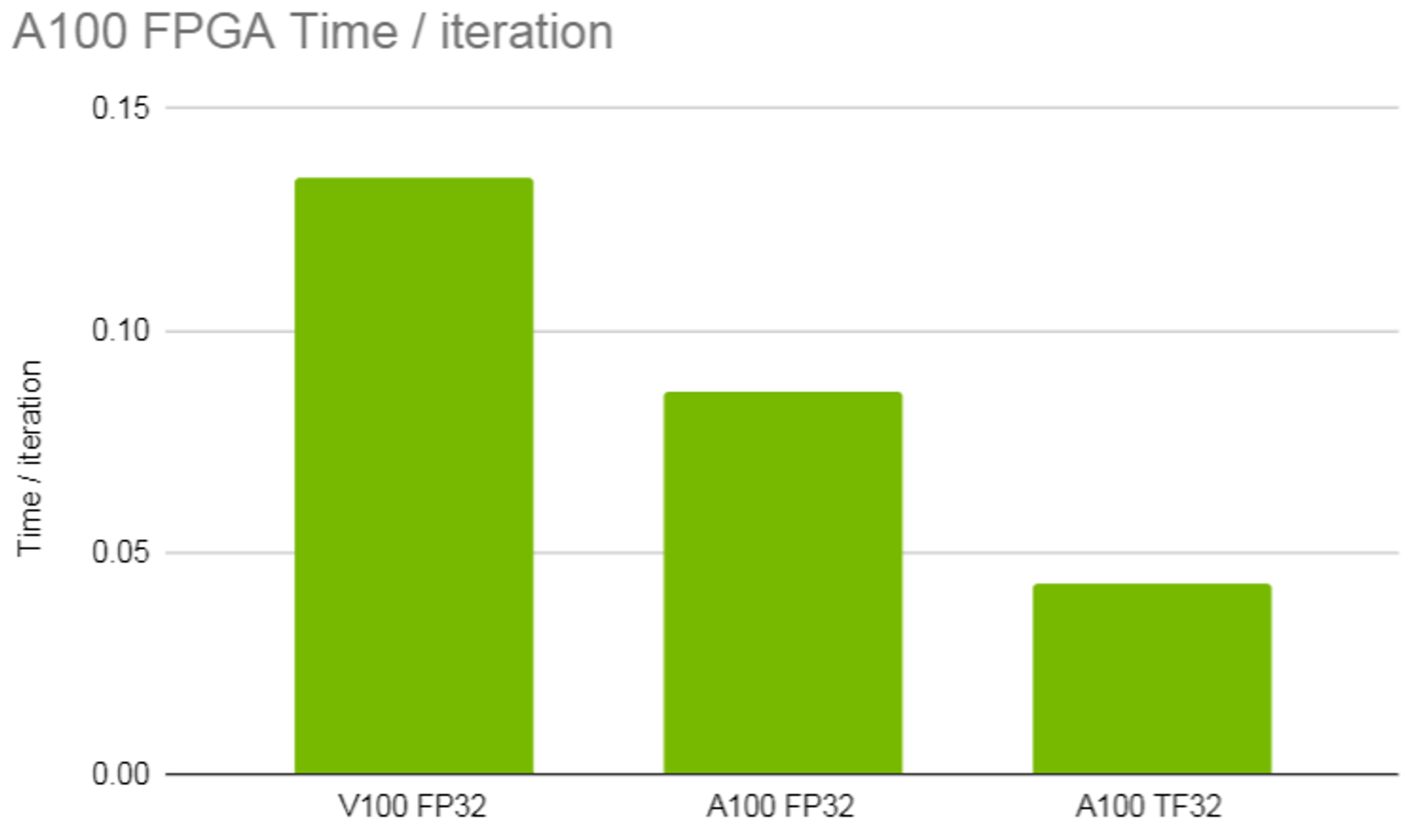
Taylor-Green vortex at time 15.0.



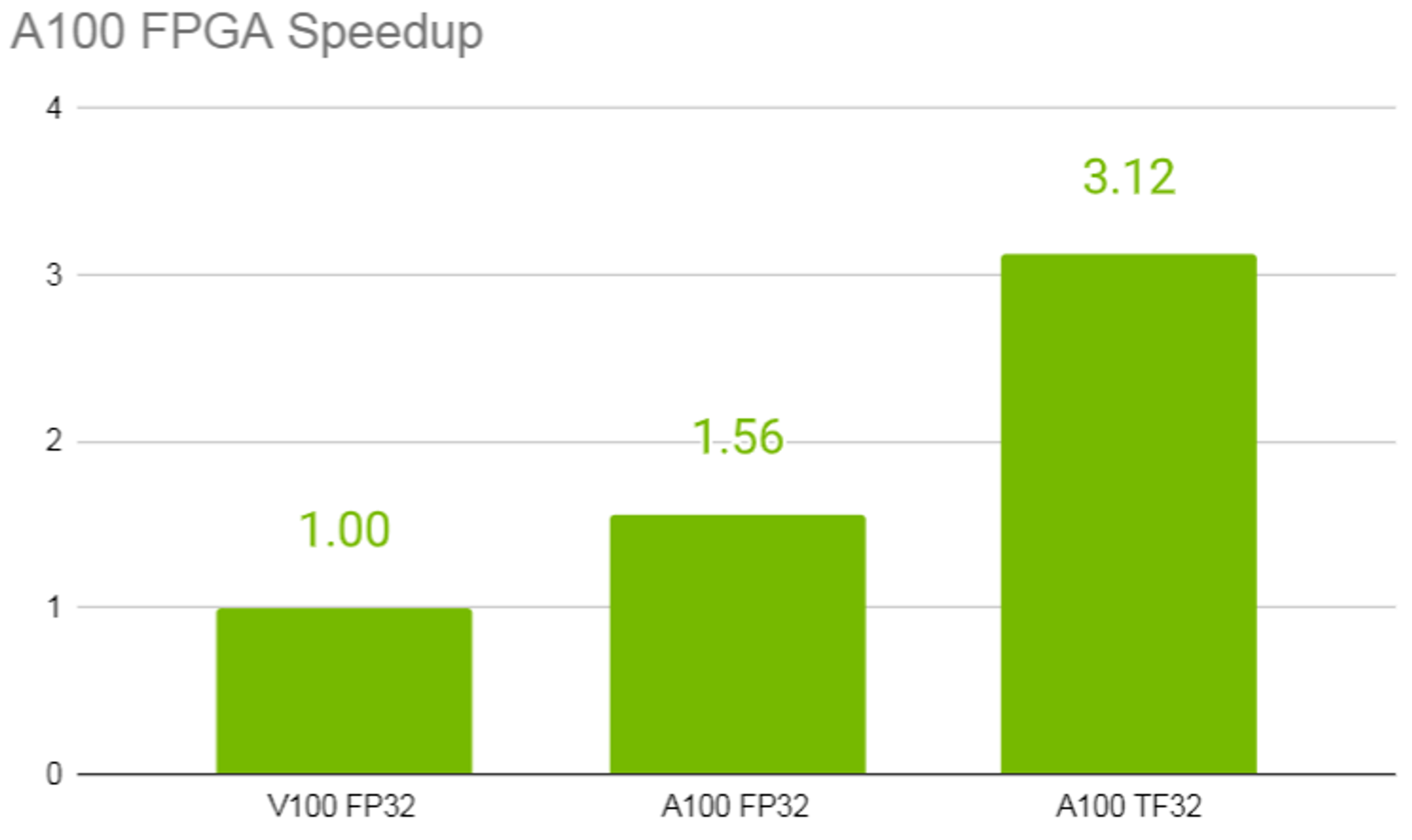
Taylor-Green Trubulent kinetic energy decay.

Modulus Framework - Performance

SINGLE GPU: Tensor Core Speed-Up for PDEs

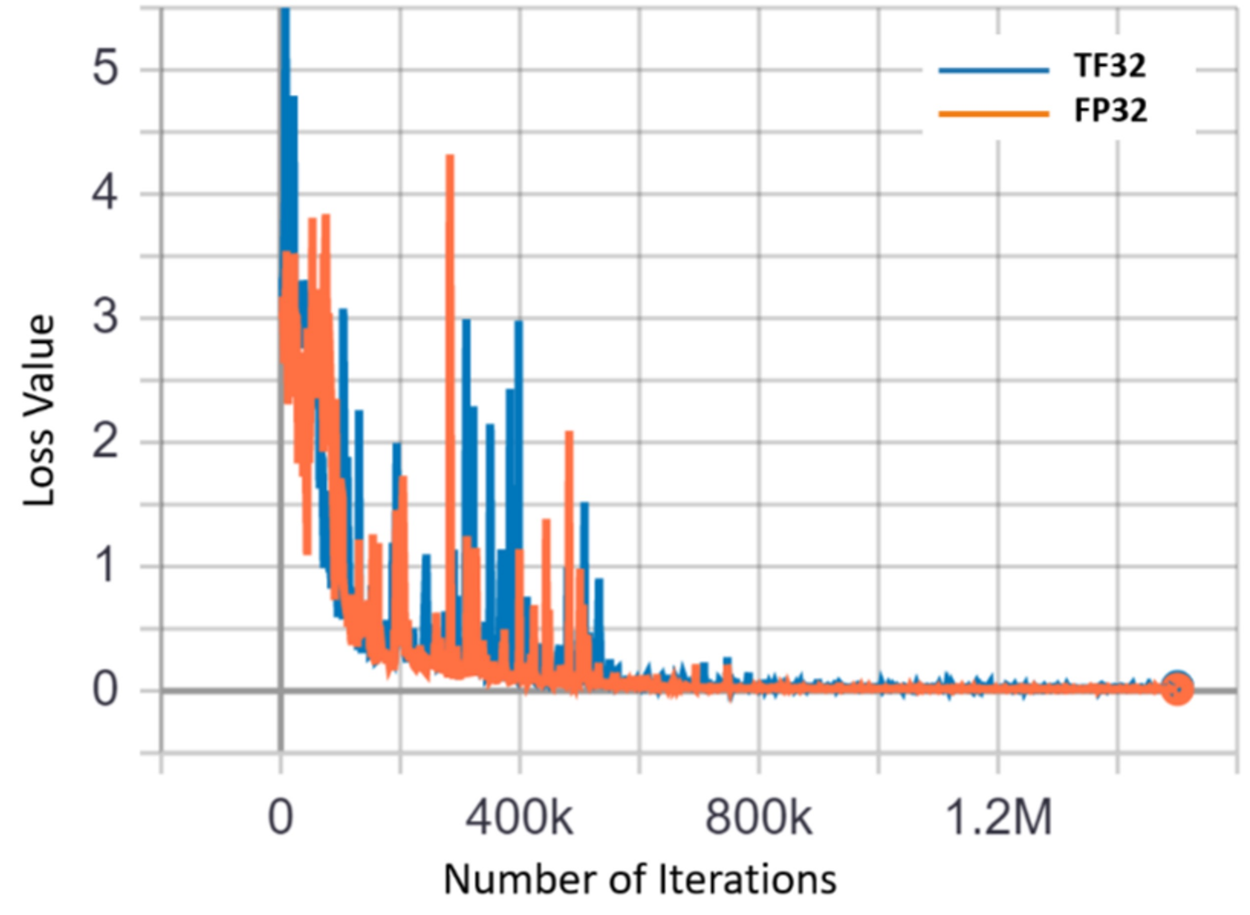


(a) Time per iteration



(b) Speed-up

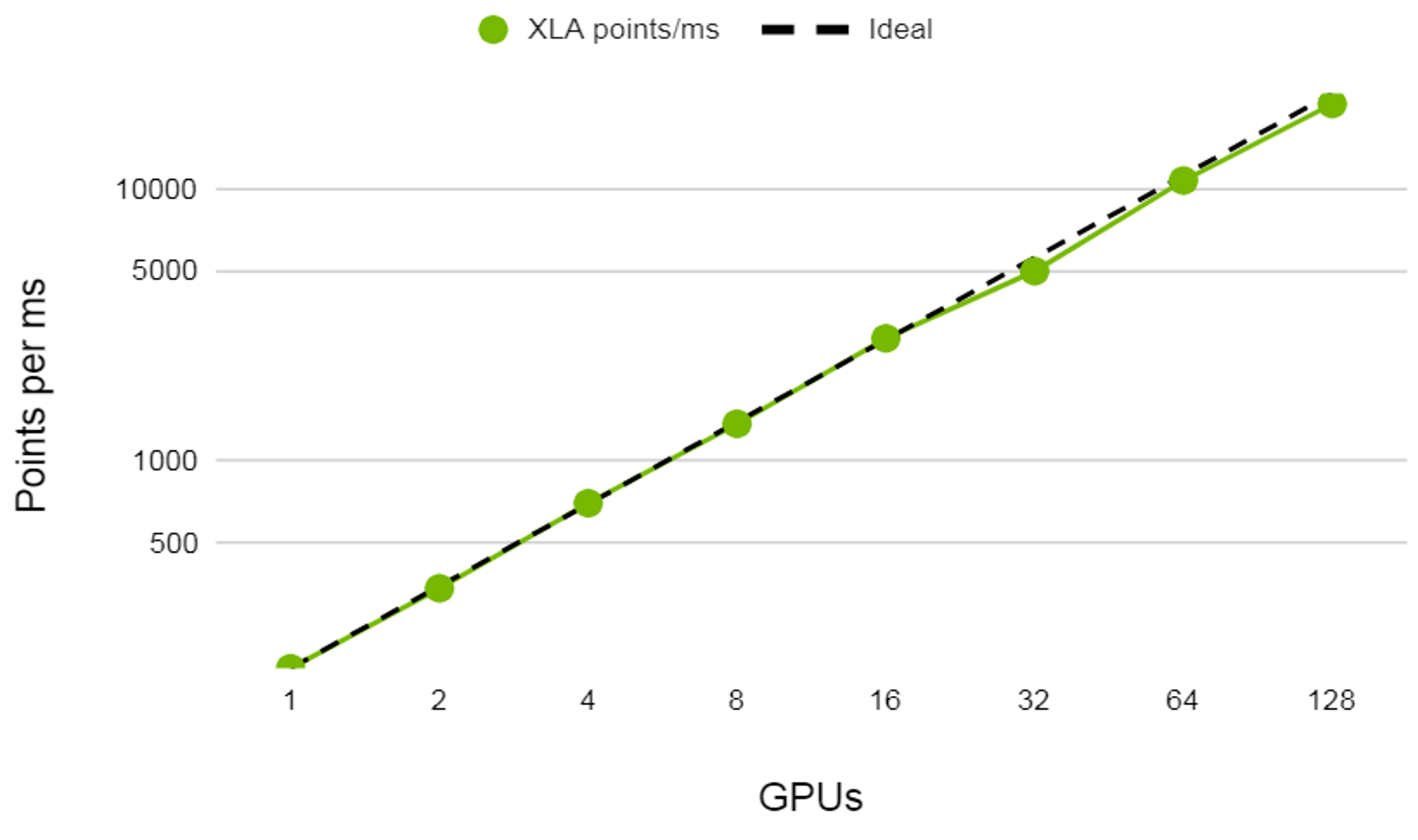
Case Description	$P_{drop}\ (Pa)$	Compute Time (hrs)
SimNet: Fully Connected Networks with FP32	29.24	86.9
SimNet: Fully Connected Networks with TF32	29.13	39.5



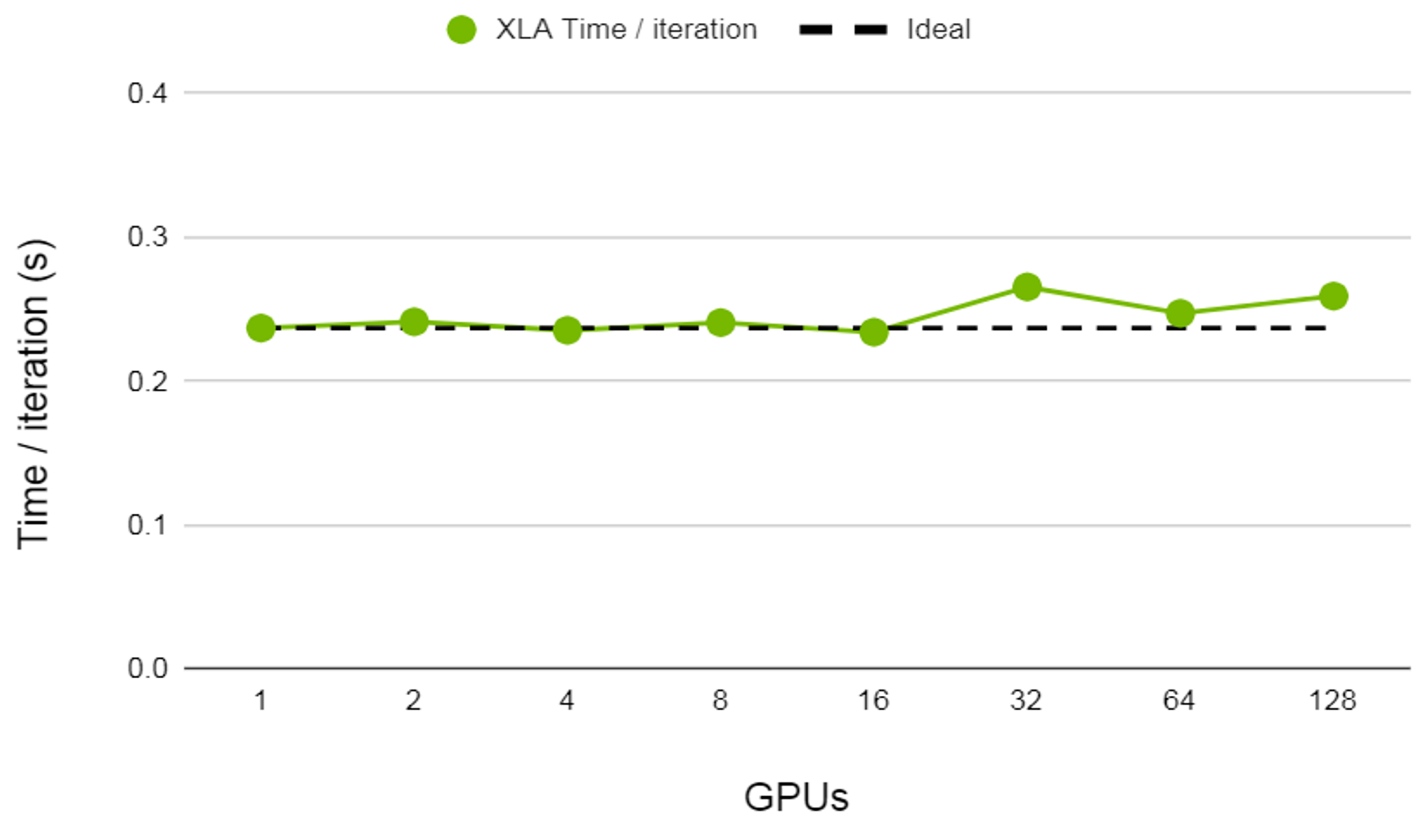
A100 FP32 vs. TF32: Results, Compute Time, Loss

Modulus Framework - Performance

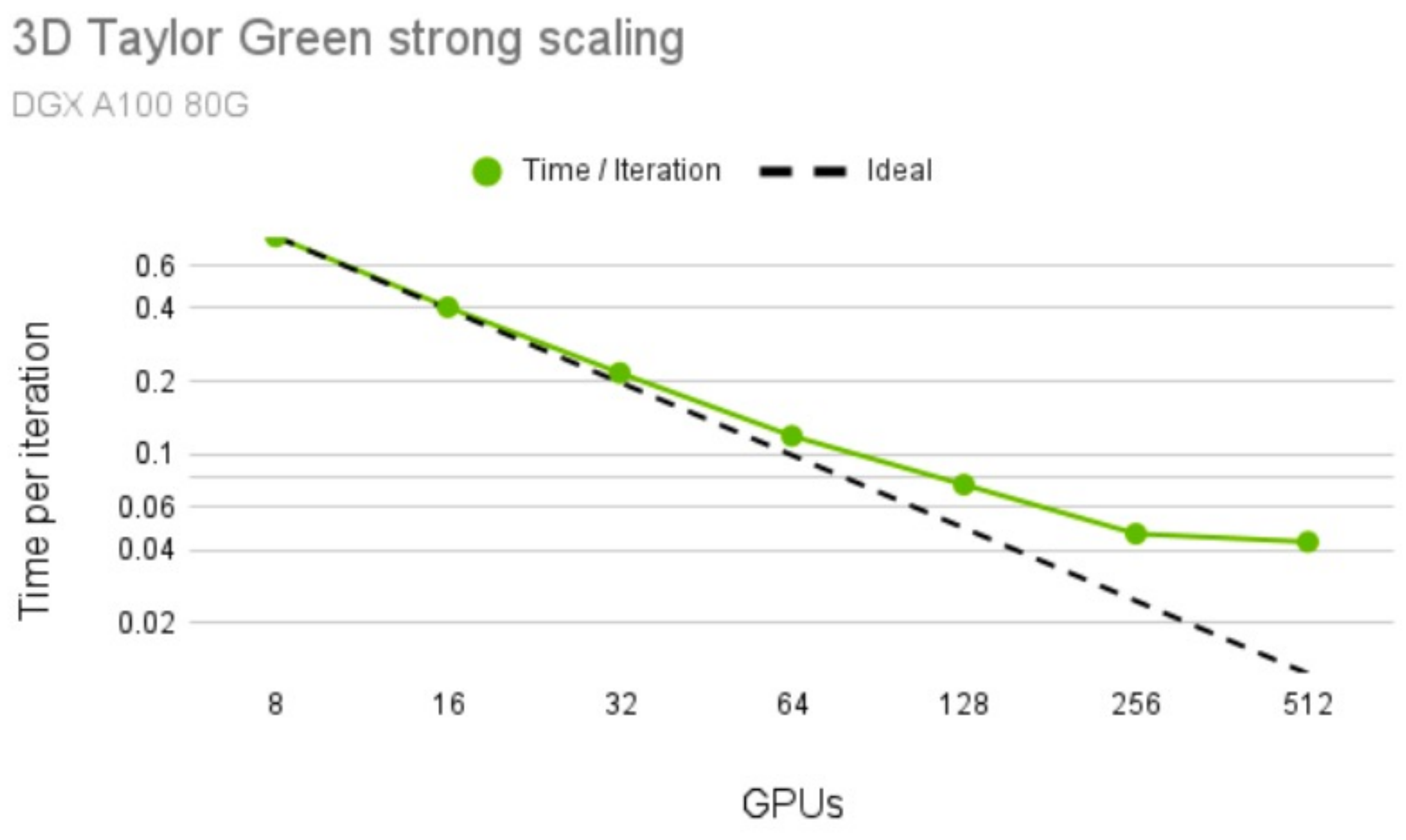
MULTI-GPU: Node Scalability

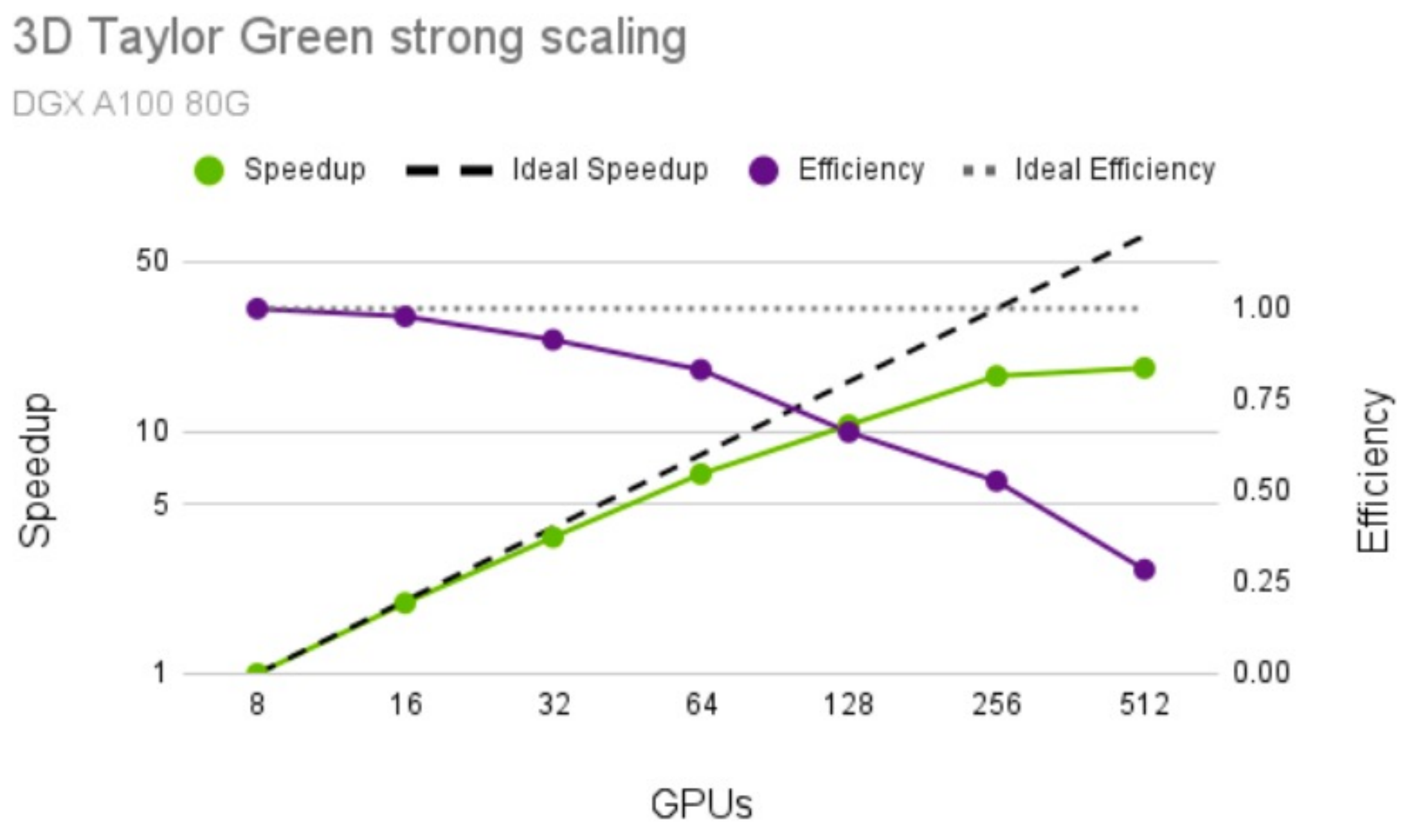


(a) Points per ms

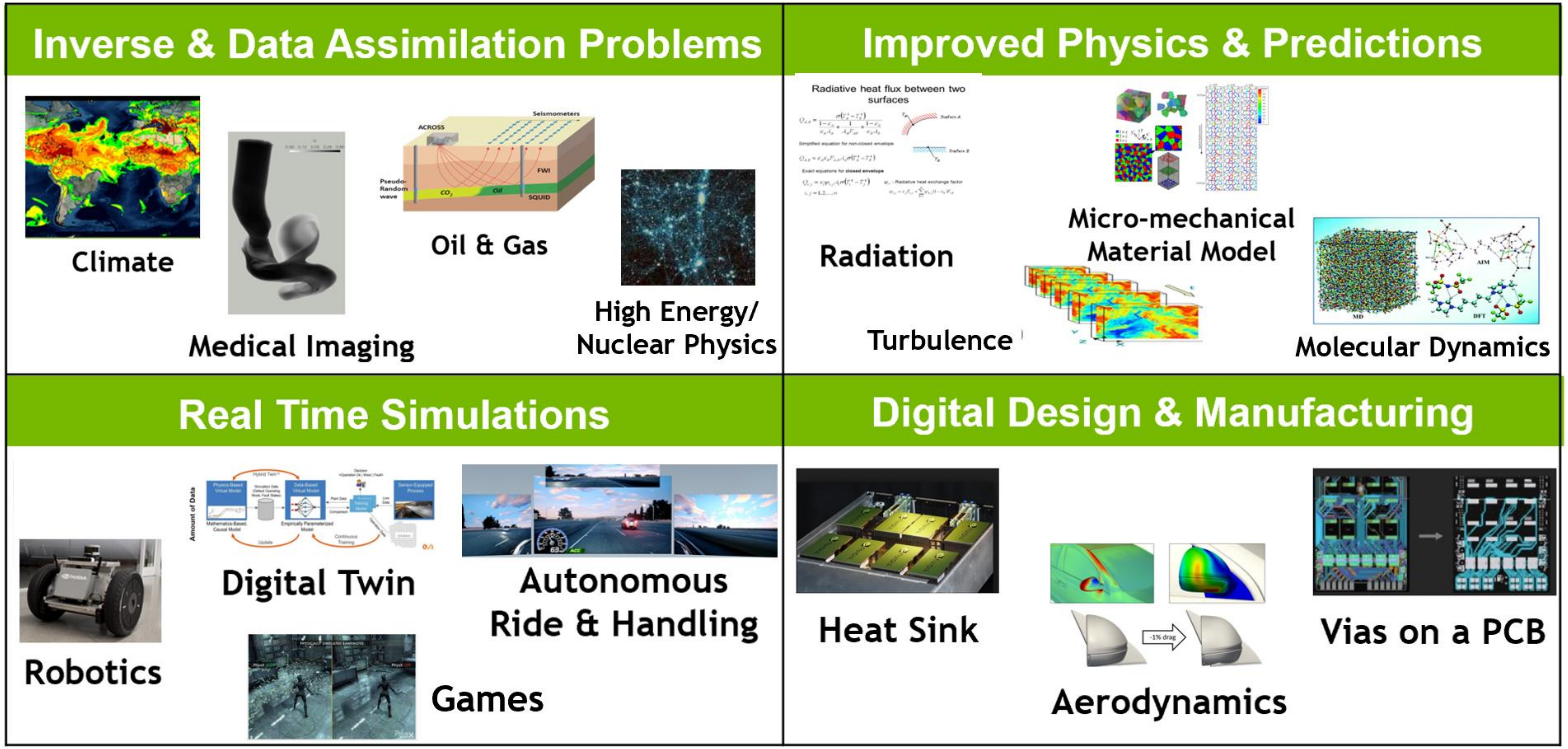


(b) Time per iteration





AI in Science and Engineering



Physics & Data - No Traditional Solver

Physics - Traditional Solver (Speed is a limitation)

Use cases

- **Reacting flow in industrial scale boiler**
- **Fluid accelerated corrosion of heat recovery steam generator**

Developing Digital Twins for Energy Applications using Modulus [S41325]

PINN for reacting flows

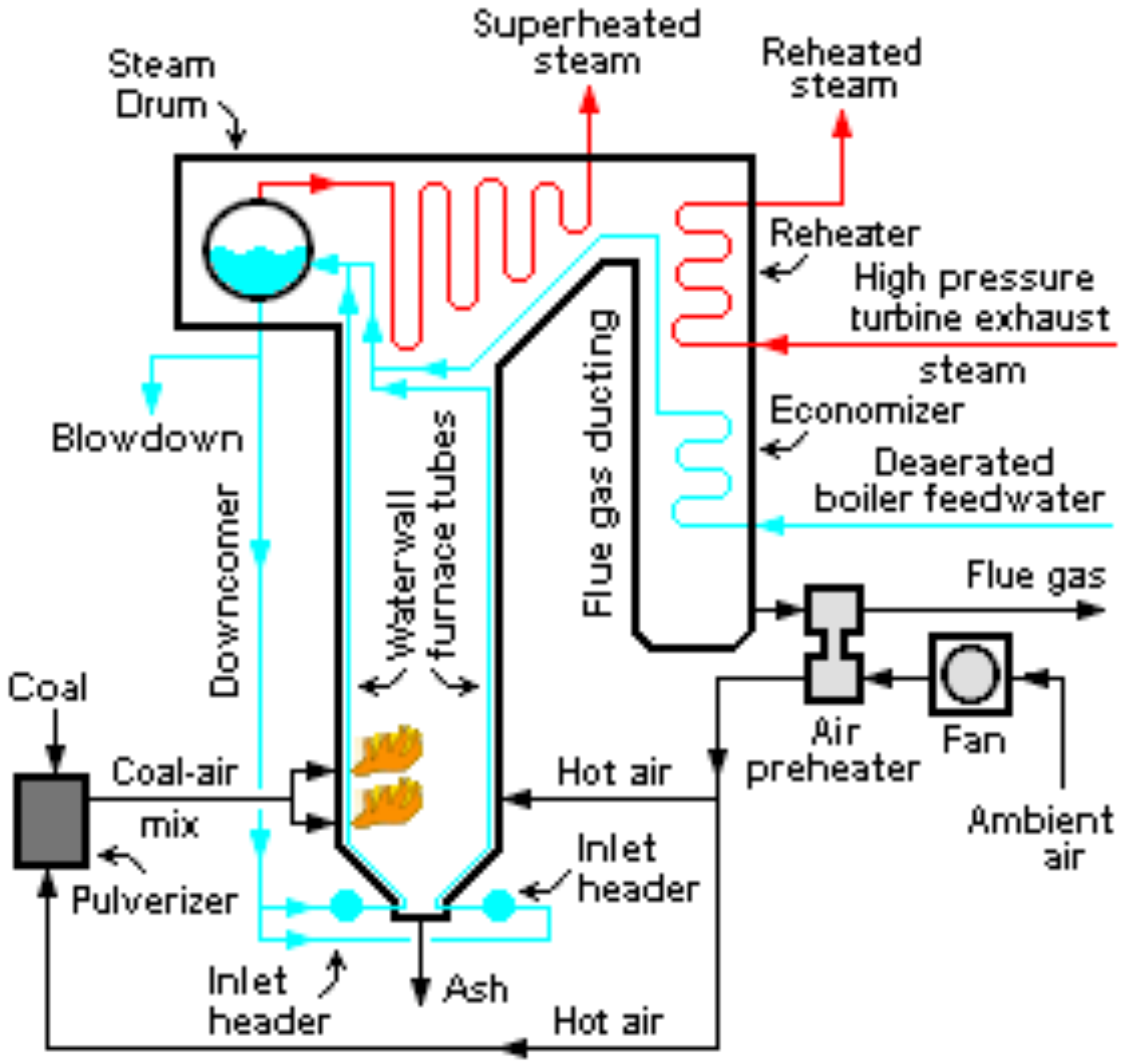
Formulation and PINN vs CFD

- Aim: Create a digital twin of an industrial scale boiler
- Simplified methane oxidation
- Implemented reacting flow transport equations for kinetics-controlled combustion
- No requirement for training data

★ Single PINN model for a range of input conditions

★ Fidelity and accuracy comparable to CFD

★ Trained PINN can provide near-instantaneous inference for any input condition



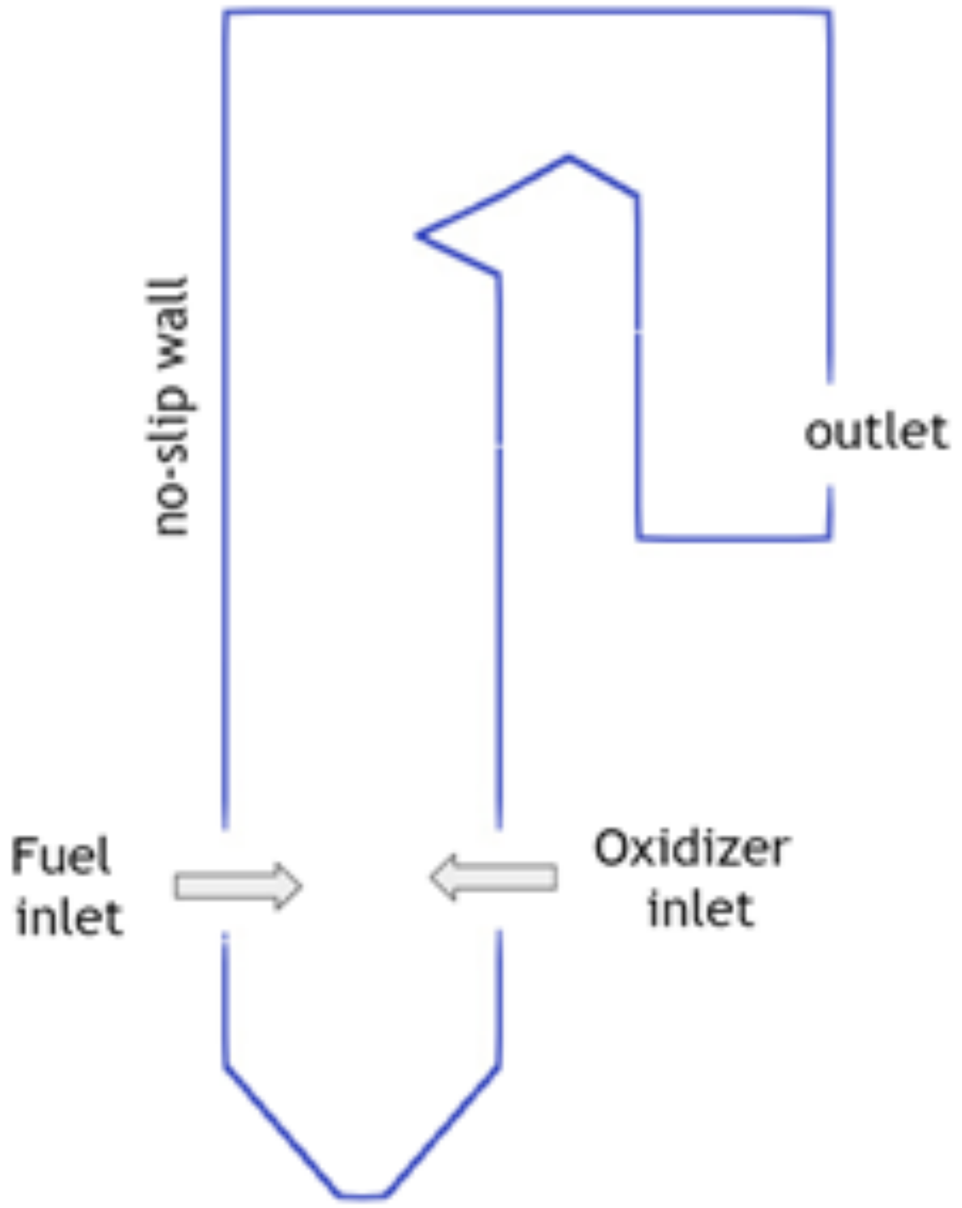


Figure source: https://commons.wikimedia.org/wiki/File:Steam_Generator.png

Developing Digital Twins for Energy Applications using Modulus [S41325]

Towards a reacting flow solver

NATIONAL ENERGY TECHNOLOGY LABORATORY

Governing equations: Strongly coupled PDEs

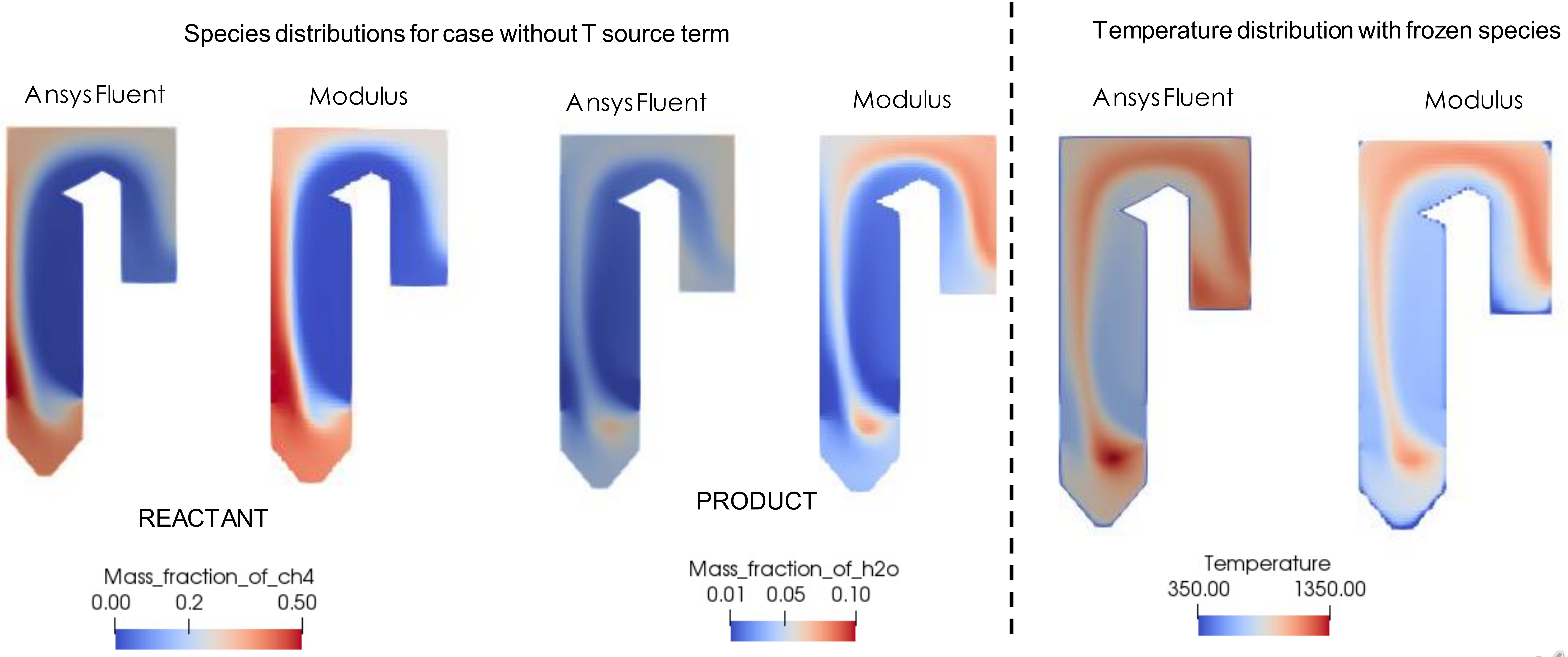
• Single species • Turbulent • Steady state • 2D	• **Multiple species** • Turbulent • Steady state • 2D	• Multiple species • **Reacting** • Turbulent • Steady state • 2D	• Multiple species • Reacting • Turbulent • **Transient** • 2D	• Multiple species • Reacting • Turbulent • Transient • **3D**

- Continuity: $\frac{\partial \rho}{\partial t} + \frac{\partial(\rho u_i)}{\partial x_i} = 0$
- Species mass fraction: $\rho \frac{\partial Y_k}{\partial t} + \rho u_i \frac{\partial Y_k}{\partial x_i} = \dot{\omega}_k + \frac{\partial}{\partial x_i}\left(\rho D_k \frac{\partial Y_k}{\partial x_i}\right)$
- Momentum: $\frac{\partial(\rho u_i)}{\partial t} + \frac{\partial(\rho u_i u_j)}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \frac{\partial \tau_{ij}}{\partial x_j}$
- Temperature: $\frac{\partial T}{\partial t} + \frac{\partial}{\partial x_i}(u_j T) = \frac{\dot{\omega}_T}{\rho c_p} + \frac{\partial}{\partial x_i}\left(\alpha \frac{\partial T}{\partial x_i}\right)$
- Kinetics-controlled single step irreversible reaction: $CH_4 + 2(O_2 + 3.76N_2) \rightarrow CO_2 + 2H_2O + 7.52N_2$
- Species source/sink terms: $\dot{\omega}_{CH4} = -MW_{CH4} k_f \left(\frac{\rho Y_{CH4}}{MW_{CH4}}\right) \left(\frac{\rho Y_{O2}}{MW_{O2}}\right)$ etc
- Temperature source term: $\dot{\omega}_T = -\sum_{k=1}^{N} h_k \dot{\omega}_k = -\sum_{k=1}^{N} h_{sk} \dot{\omega}_k - \sum_{k=1}^{N} \Delta h_{f,k}^0 \dot{\omega}_k$

Developing Digital Twins for Energy Applications using Modulus [S41325]

Simplified systems

PINN vs CFD



Resolving the issue with large T-source

Handling large T-source

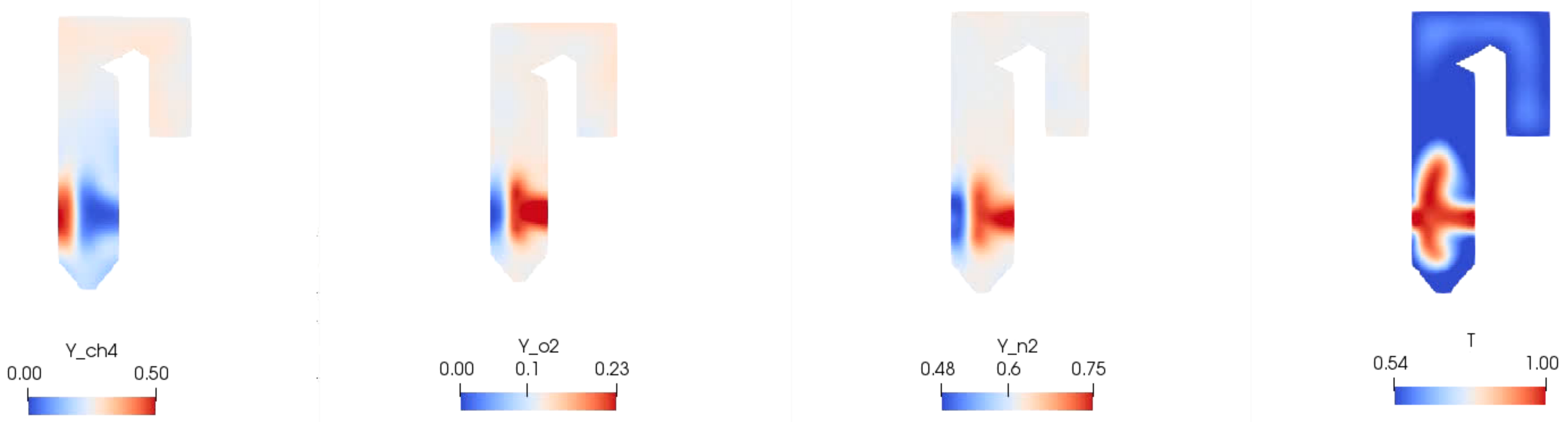
- T-source dominates the Y-source
- This can lead to imbalances between the backpropagated gradients

A) Gradient normalization approach

- Attempts to remove the dominance of any component of the global loss function
- Dynamically assigns weights to different constraints

B) Transient approach

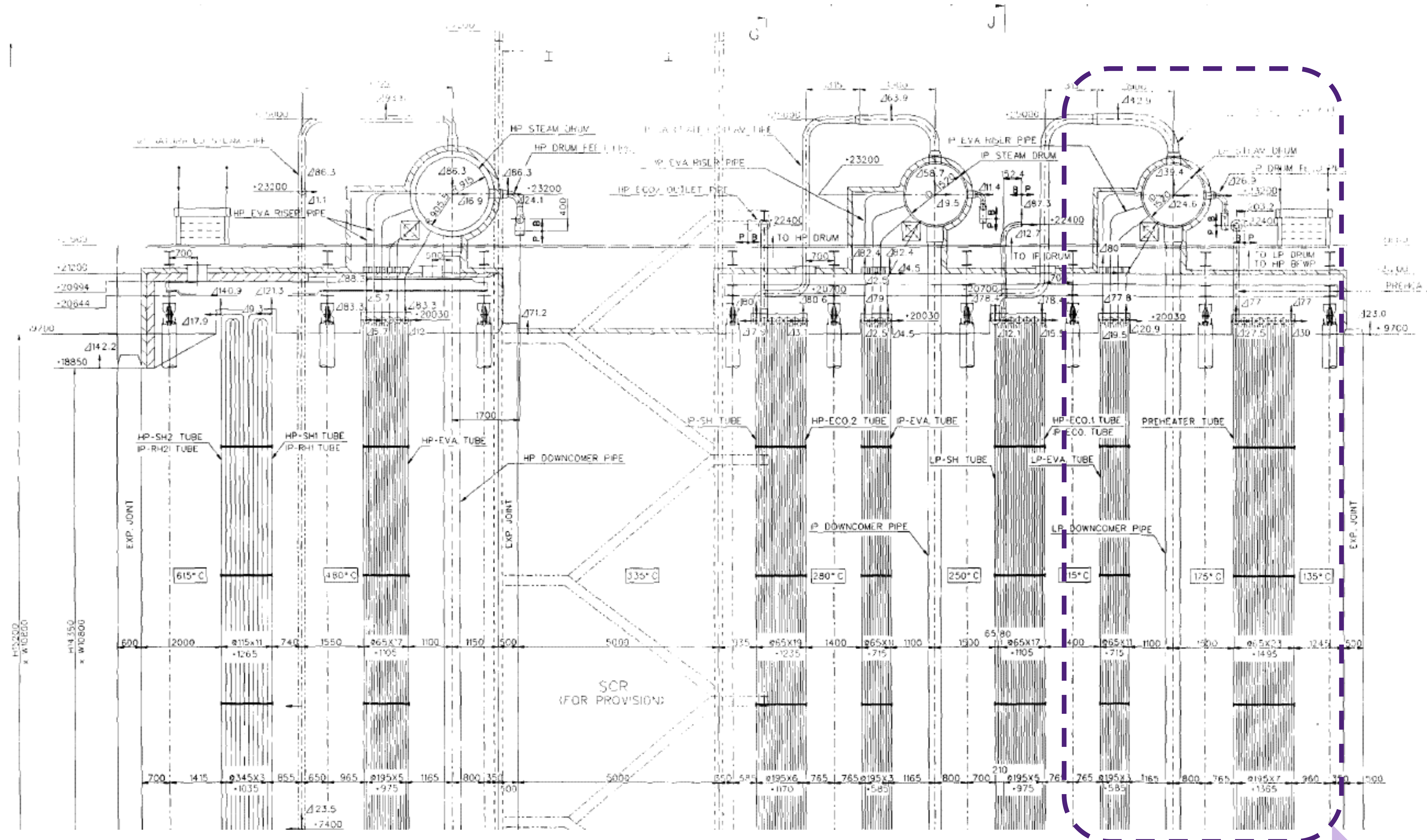
- Handles large source terms by learning the change between states instead of learning everything at once
- Uses a moving time window approach



Use cases

- Reacting flow in industrial scale boiler
- Fluid accelerated corrosion of heat recovery steam generator

A typical Heat Recovery Steam Generator and some challenges for Digitalization



Exhaust gas flow and temperatur

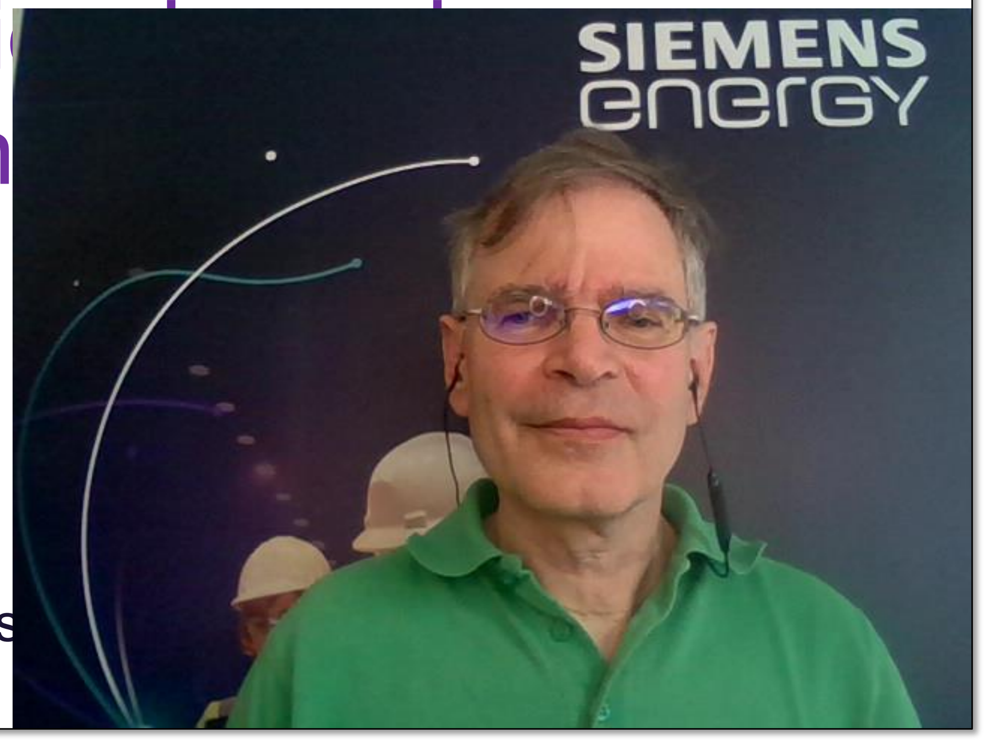
Low Pressor Evaporator

- Mix of steam and liquid changes with operations
- Area of evaporation changes with operations

➔Point of corrosion difficult to predict

But

- Thousands of pipes, no sensors
- Fluid dynami geometry, on available



The Fluid Accelerated Corrosion Workflow

Current process

Geometry : 2D ⟶ 3D
Full 3D geometry from 2D drawing

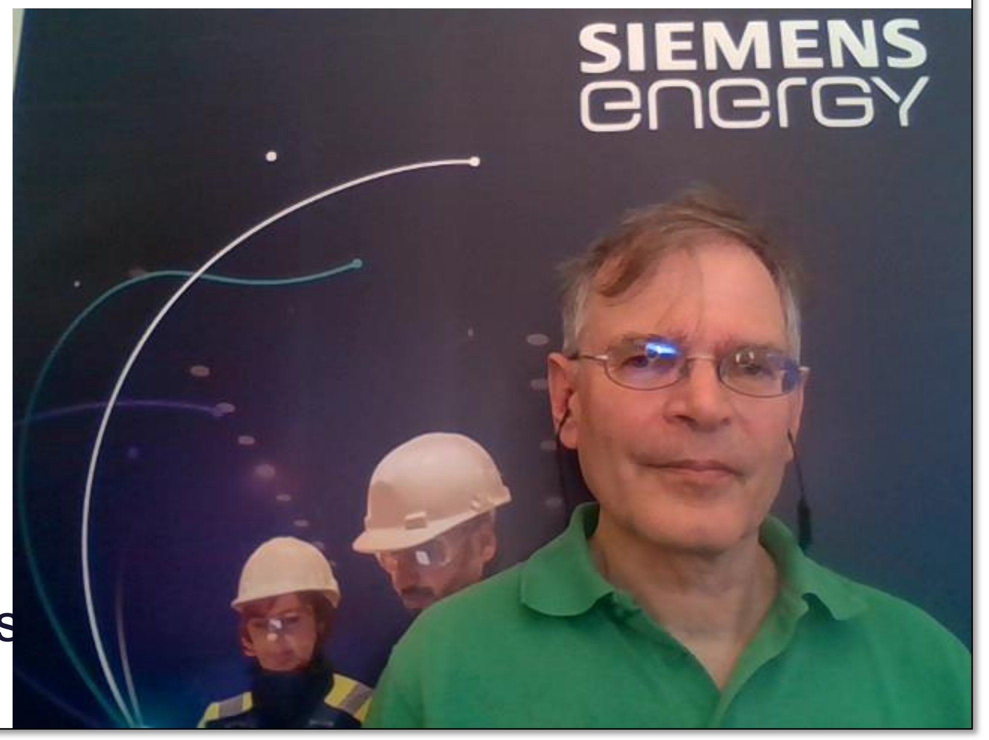
CFD Model preparation
Geometry preparation for CFD, mesh generation (possible iterations)

3D CFD operating conditions
Different operating conditions for creating reduced order model response surface

Better process?

The challenge of the current approach:

- Detailed simulation take months to prepare for one plant
- Costs exceed customer benefits
- Simplified data driven approaches do not determine the risk level in an acceptable manner



Case Study on Developing Digital Twins for the Power Industry using Modulus and Omniverse [S41671]

HRSG WORKFLOW WITH MODULUS

Current process

Geometry : 2D ⟶ 3D

Full 3D geometry from 2D drawing

CFD Model preparation

Geometry preparation for CFD, mesh generation (possible iterations)

3D CFD operating conditions

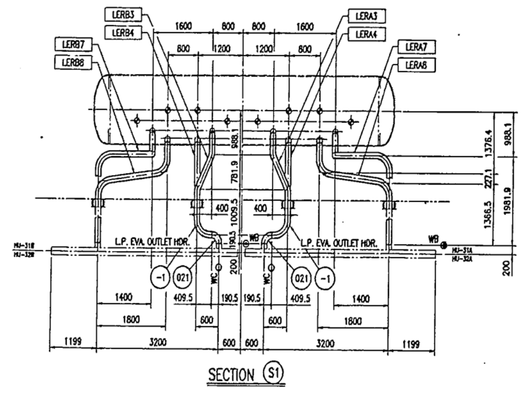
Different operating conditions for creating reduced order model response surface

---

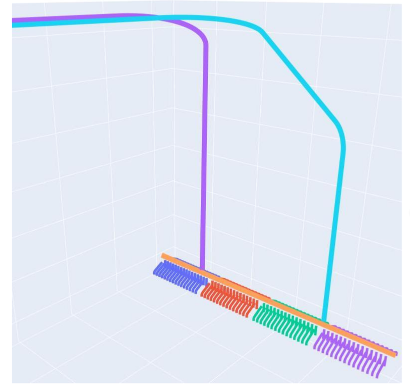
New suggested process

Geometry : 2D ⟶ 3D

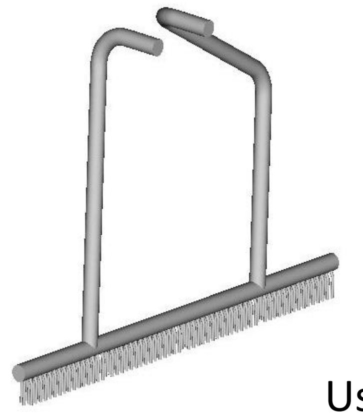
2D drawing



Centerline geometry

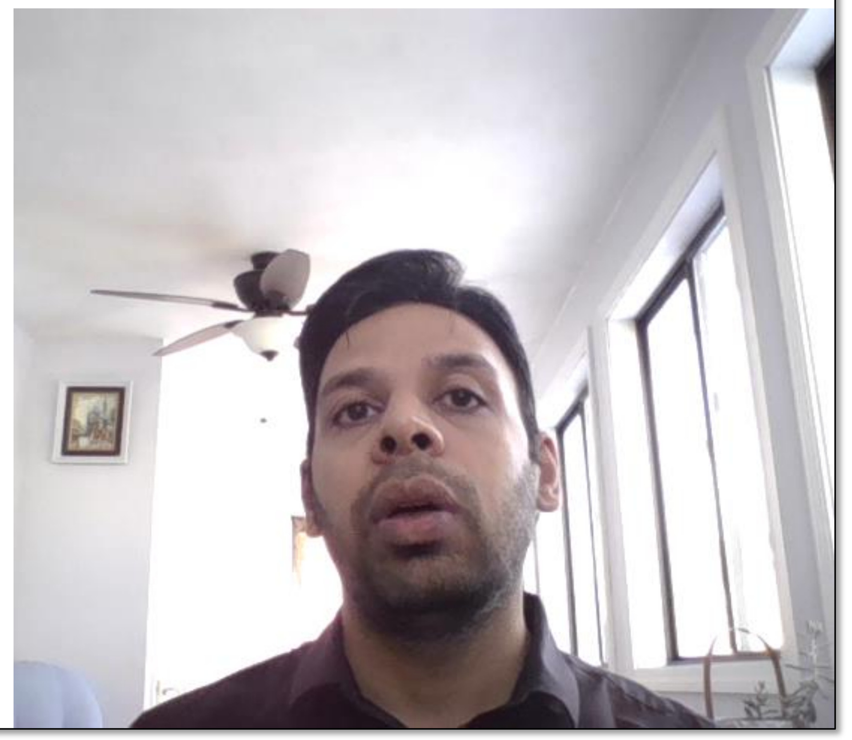


3D geometry for Fluids



< 0.5 hr

Using standard CAD tools



Case Study on Developing Digital Twins for the Power Industry using Modulus and Omniverse [S41671]

HRSG WORKFLOW WITH MODULUS

Current process

Geometry : 2D ⟶ 3D
Full 3D geometry from 2D drawing

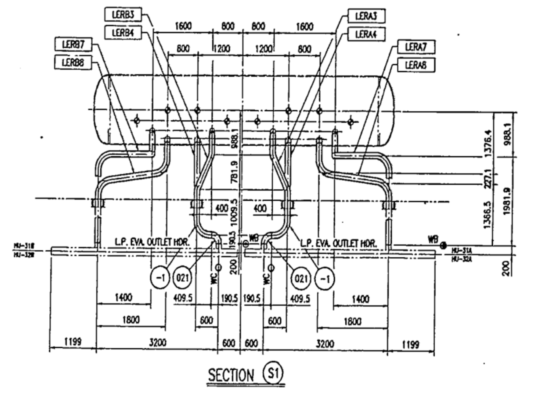
CFD Model preparation
Geometry preparation for CFD, mesh generation (possible iterations)

3D CFD operating conditions
Different operating conditions for creating reduced order model response surface

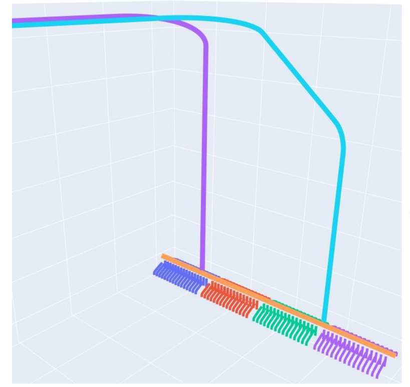
New suggested process

Geometry : 2D ⟶ 3D

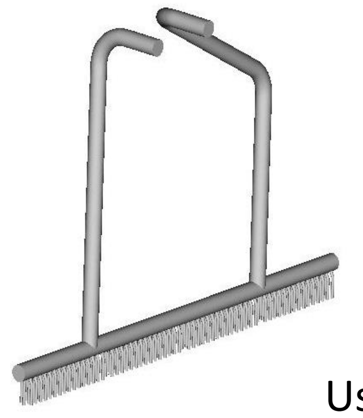
2D drawing



Centerline geometry



3D geometry for Fluids

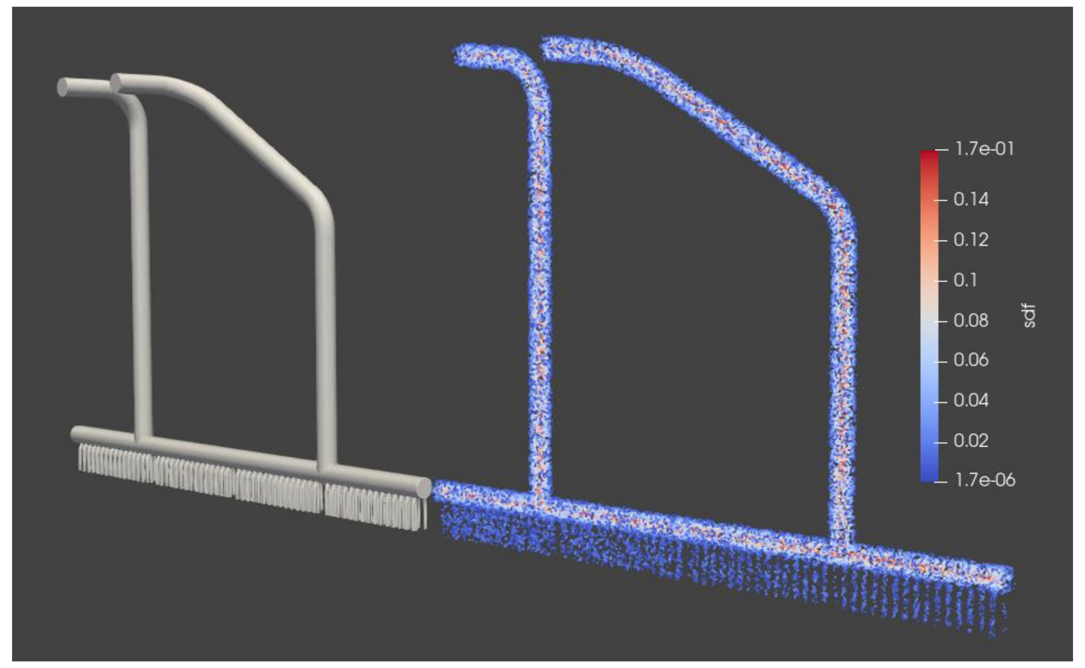


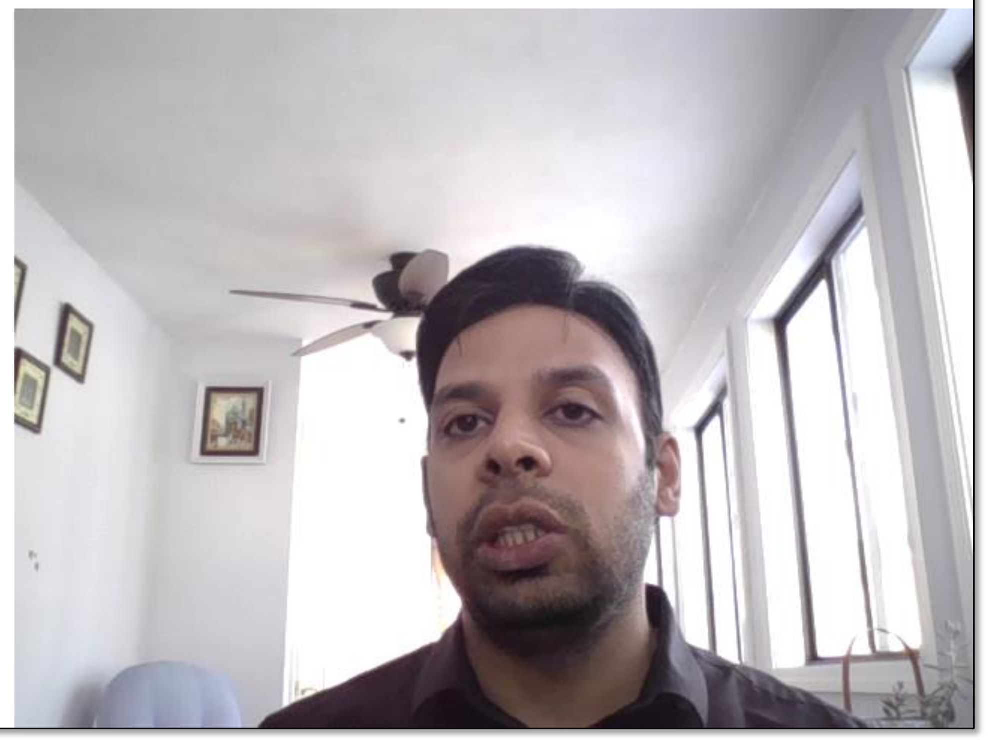
Using standard CAD tools

< 0.5 hr

Model Training

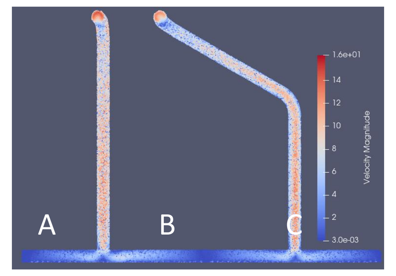
- Mesh free, fast point cloud generation
- Incompressible NS eqs
- Fourier feature neural network
- Parameterized input velocity



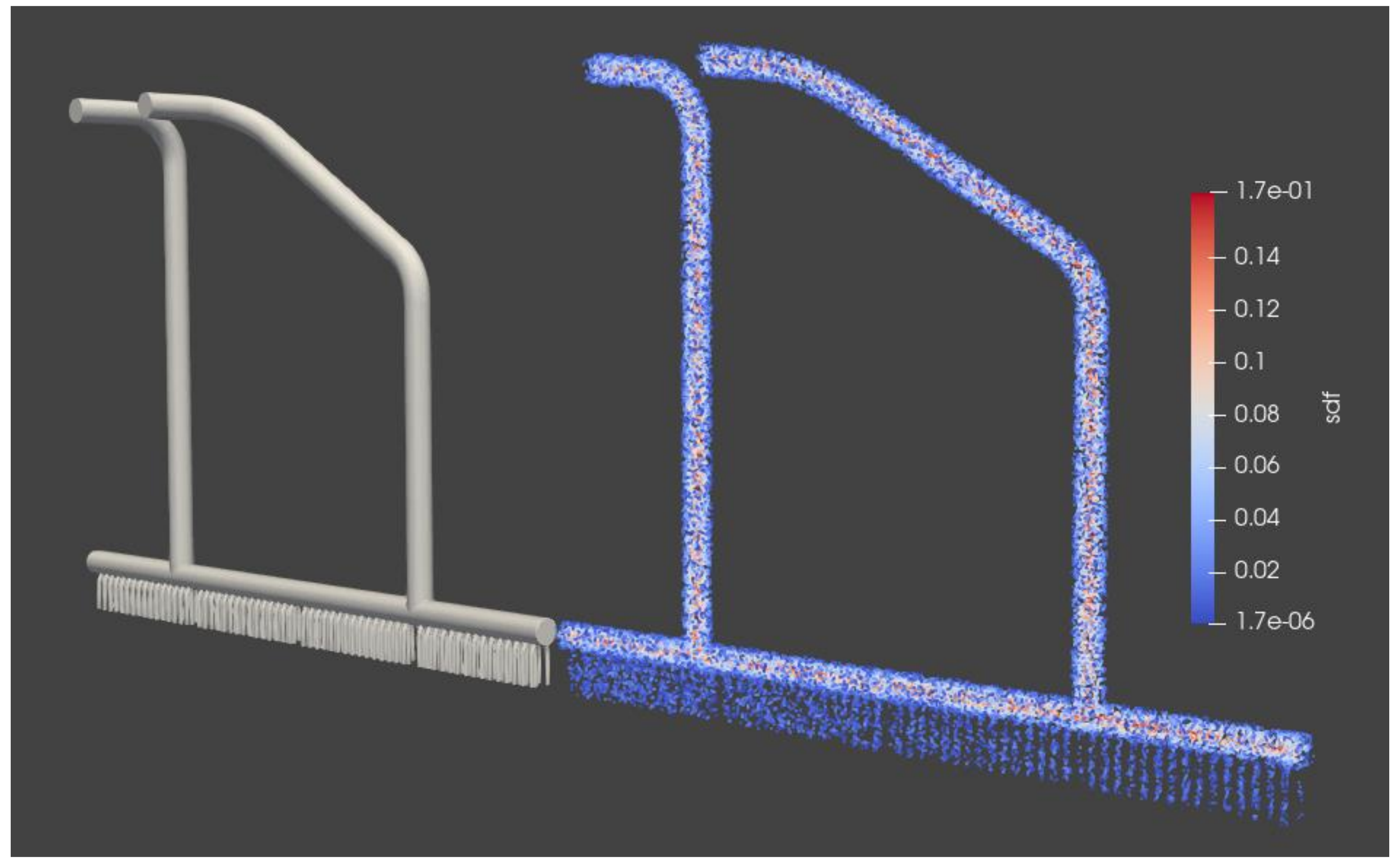


Case Study on Developing Digital Twins for the Power Industry using Modulus and Omniverse [S41671]

HRSG WORKFLOW WITH MODULUS




```
from modulus.mesh_utils.mesh import Mesh
from modulus.plot_utils.vtk import var_to_vtk
...
# read stl files to make point cloud
mesh_path = './stl/'
prefix = 'Mesh_up_LP0_'

interior_mesh = Mesh.from_stl(mesh_path + prefix + 'pipes.stl')
noslip_mesh = Mesh.from_stl(mesh_path + prefix + 'pipes_noslip.stl',airtight=False)

# sample points in interior
interior_points = interior_mesh.sample_interior(64000, compute_distance_field=True)
var_to_vtk(interior_points, 'interior_points')
```



PySDF library: Modulus' Tessellated Geometry module, Optix for Sampling and SDF

Symbolic definition of equations

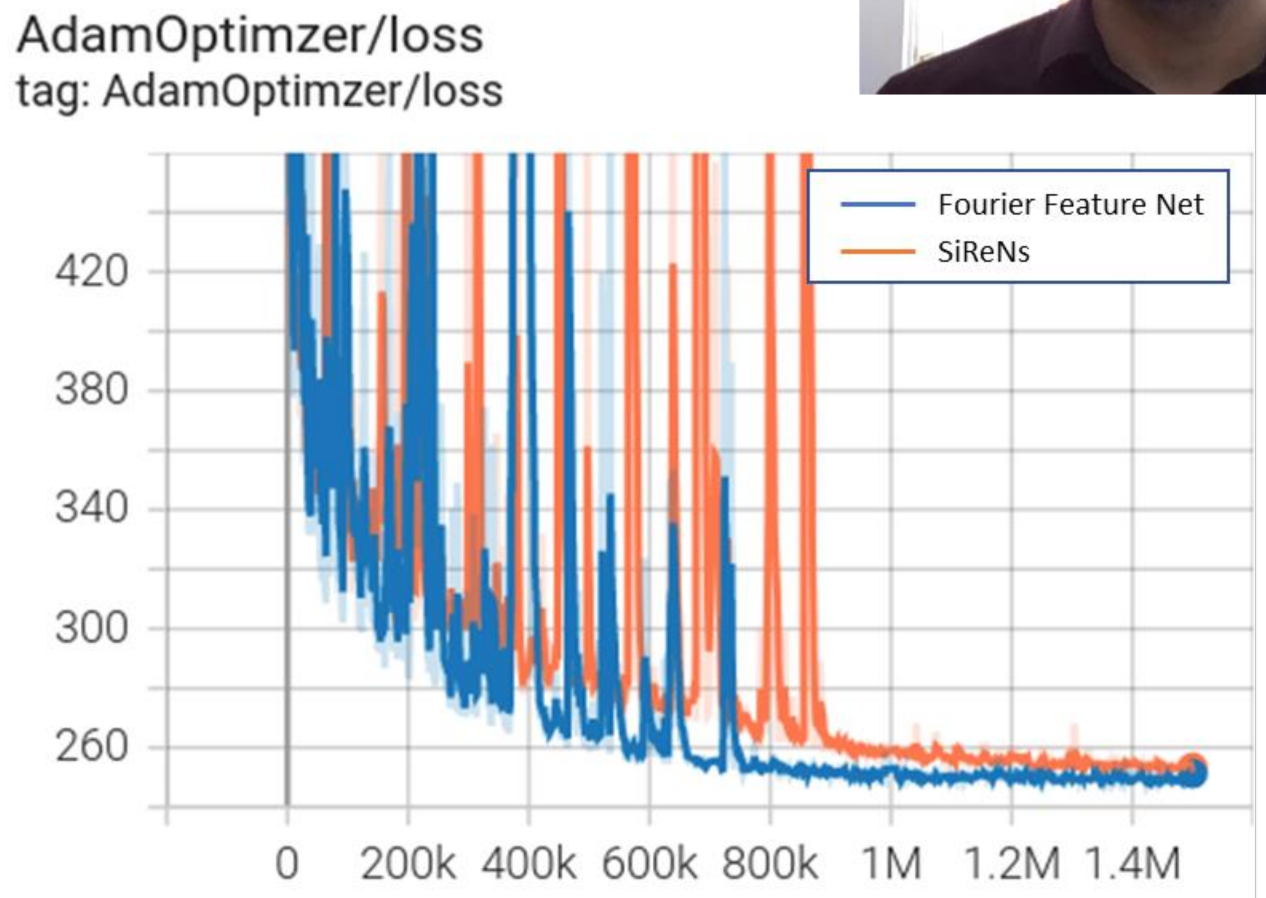
$$\nabla \cdot \mathbf{u} = 0$$
$$\rho \mathbf{u} \cdot \nabla \mathbf{u} = -\nabla p + \mu \nabla^2 \mathbf{u}$$

```
...
        self.equations["continuity"] = (
            rho.diff(t) + (rho * u).diff(x) + (rho * v).diff(y) + (rho * w).diff(z)
        )
        self.equations["momentum_x"] = (
            (rho * u).diff(t)
            + (
                u * ((rho * u).diff(x))
                + v * ((rho * u).diff(y))
                + w * ((rho * u).diff(z))
                + rho * u * (curl)
            )
            + p.diff(x)
            - (-2 / 3 * mu * (curl)).diff(x)
            - (mu * u.diff(x)).diff(x)
            - (mu * u.diff(y)).diff(y)
            - (mu * u.diff(z)).diff(z)
            - (mu * (curl).diff(x))
        )
...
```

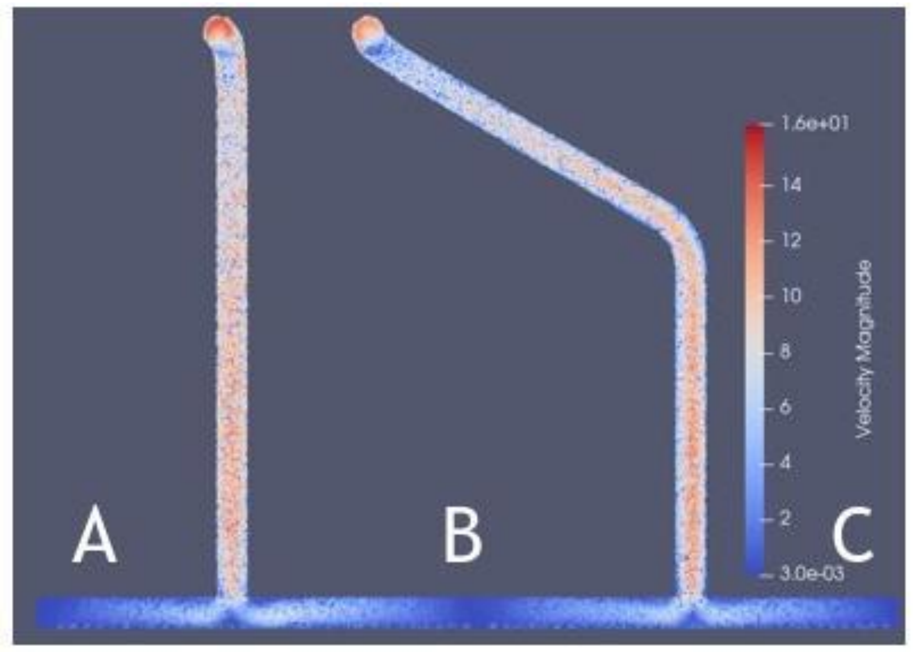
Defining network architecture

```
...
class HRSGSolver(Solver):
    train_domain = HRSGTrain
    monitor_domain = HRSGMonitor
    arch = FourierNetArch   <----- Fourier Feature Network

    def __init__(self, **config):
        super(HRSGSolver, self).__init__(**config)

        self.arch.frequencies = ('axis', [i for i in range(20)])
...
```

$$Loss = L_{PDE} + L_{BC}$$

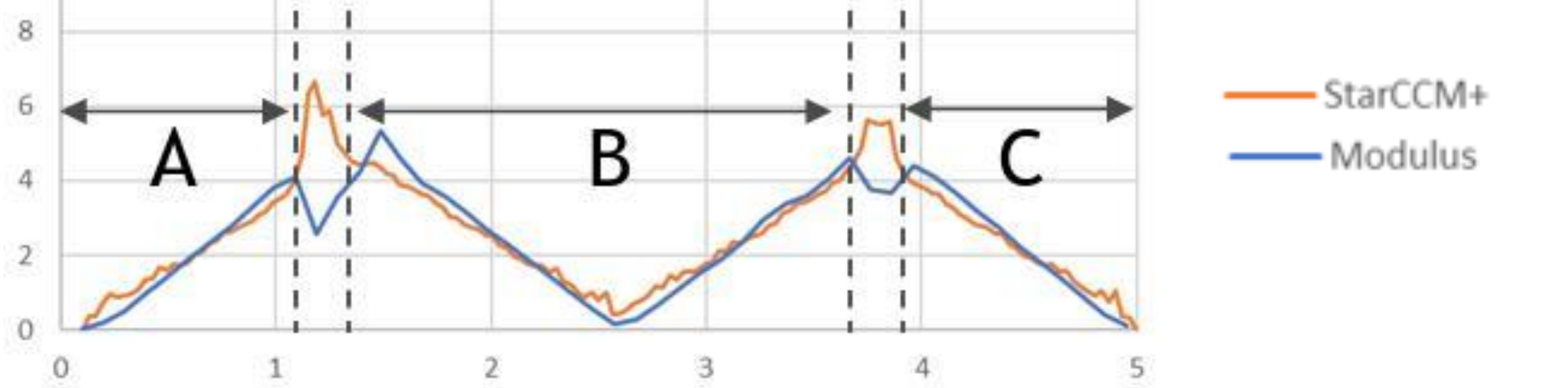


Hyperparameter tuning and training



Verification and Validation

Good match (in A,B,C):



NVIDIA

Case Study on Developing Digital Twins for the Power Industry using Modulus and Omniverse [S41671]

HRSG WORKFLOW WITH MODULUS

Current process

Geometry : 2D ⟶ 3D

Full 3D geometry from 2D drawing

CFD Model preparation

Geometry preparation for CFD, mesh generation (possible iterations)

3D CFD operating conditions

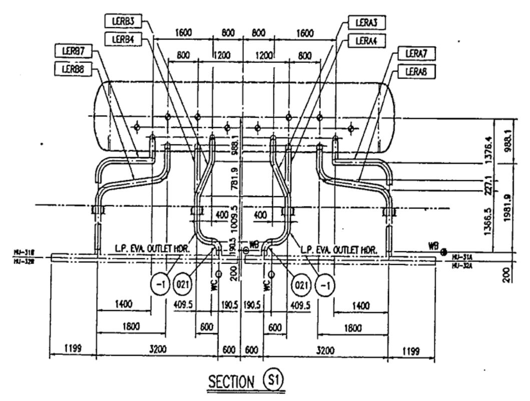
Different operating conditions for creating reduced order model response surface

---

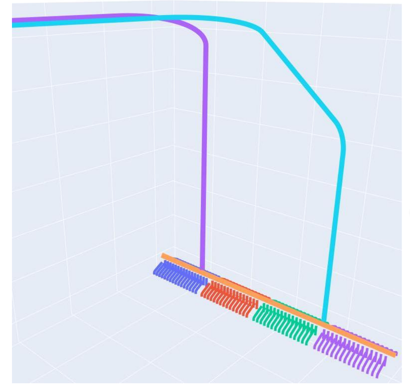
Suggested process

Geometry : 2D ⟶ 3D

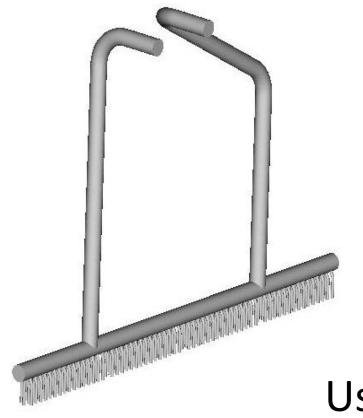
2D drawing



Centerline geometry



3D geometry for Fluids

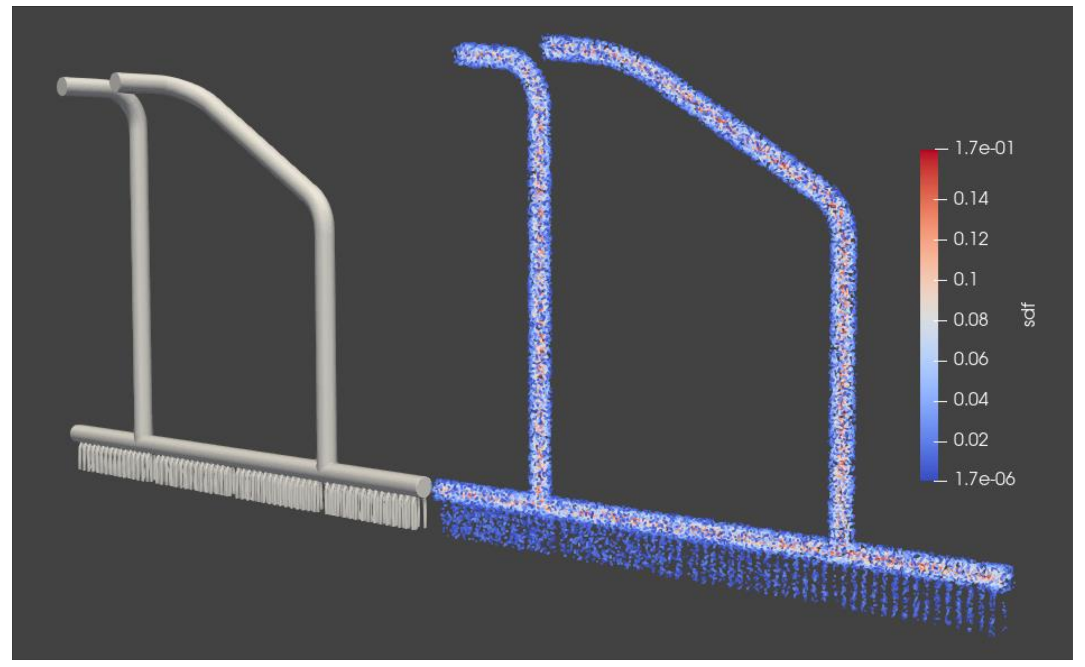


Using standard CAD tools

< 0.5 hr

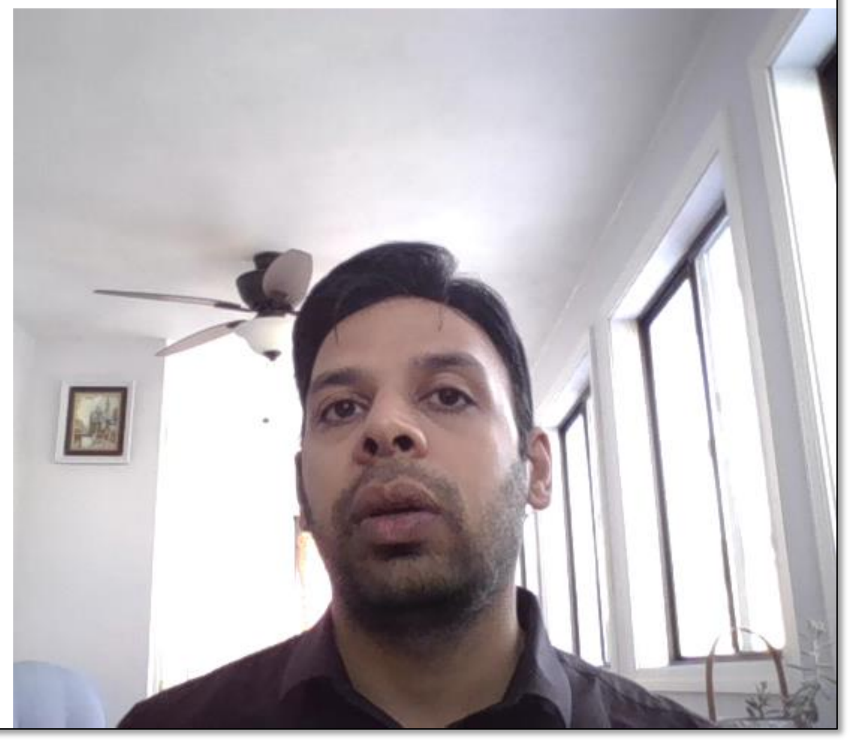
Model Training

- Mesh free, fast point cloud generation
- Incompressible NS eqs
- Fourier feature neural network
- Parameterized input velocity



Infer new scenario

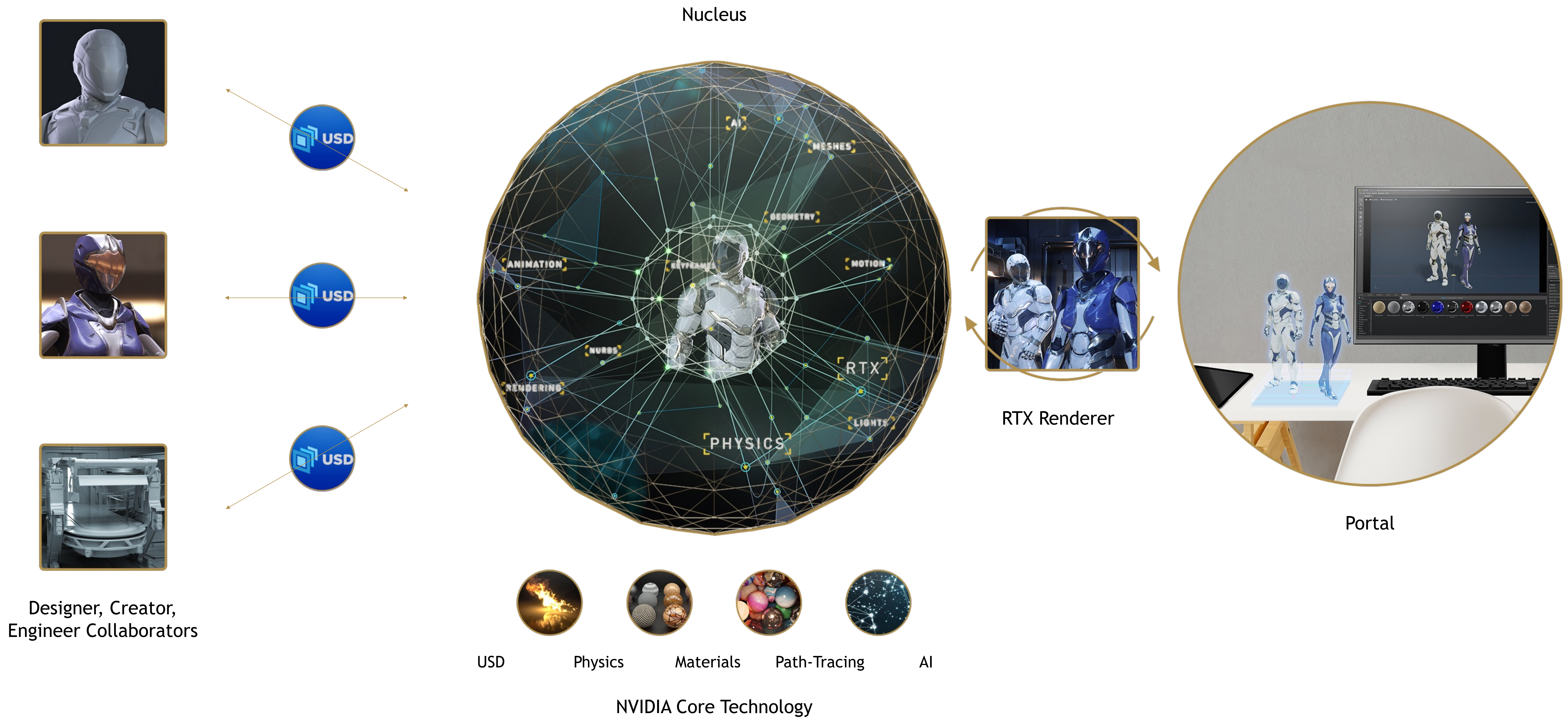
- Order of 10,000x speed up per scenario
- Order of seconds inference time vs 8 hr per CFD run



What’s New and Online Resources

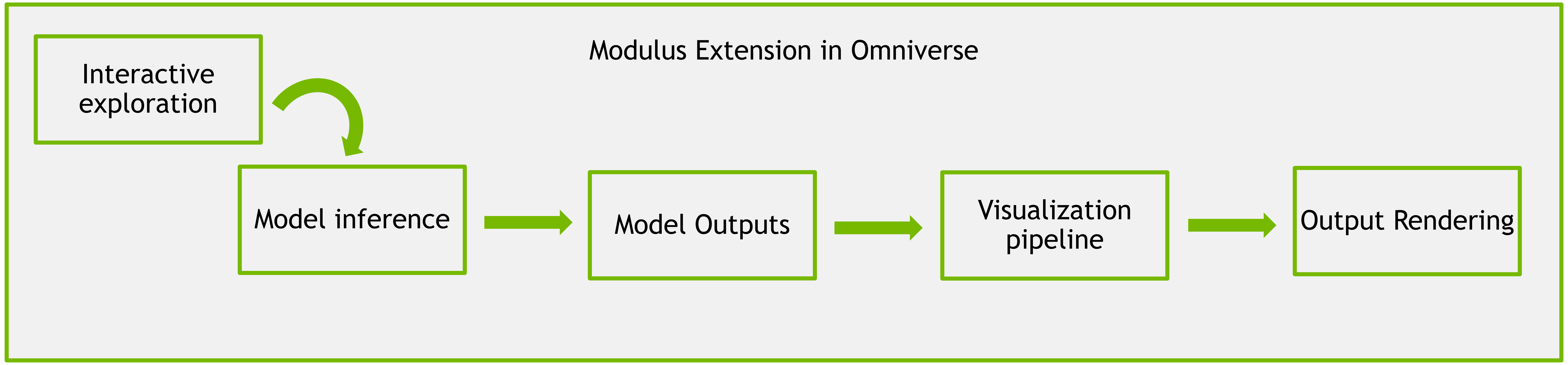
NVIDIA Omnivers Enterprise

Platform for Creating and Connecting Virtual Worlds



Case Study on Developing Digital Twins for the Power Industry using Modulus and Omniverse [S41671]

MODULUS – OMNIVERSE INTEGRATION



- Modulus Omniverse extension:
  - enables importing outputs of Modulus trained model into a visualization pipeline for common output scenarios ex: streamlines, iso-surface
  - provides an interface that enables interactive exploration of design variables/parameters to infer new system be

NVIDIA

New Features in NVIDIA Modulus v22.03

- New network architectures
  - Fourie Neural Operator
  - Physics Informed Neural Operator
  - Adaptive Fourier Neural Operator
  - Deep-O Net
- Modeling Enhancements
  - 2-eqn. Turbulence model
    - $k-\epsilon,\ \ k-\omega$ models with standard and Launder-Spalding wall functions
  - Exact boundary condition imposition
- Training features
  - Support for new optimizers
    - 30+ optimizers
  - New algorithms for loss balancing
    - Grad Norm, Relative Loss Balancing with Random Lookback, and Soft Adapt
  - Sobolev (gradient-enhanced) training
  - Hydra Config
  - Post-processing

New Features in NVIDIA Modulus v22.07

- New network architectures
  - Generalized DeepONet architecture
  - FourCastNet
- Training features
  - L1-L2 Loss Decaying
- Performance Enhancements
  - Meshless Finite Differentiation
  - Dataset Refactor
  - Tiny CUDA NN
  - CUDA Graphs
  - Geometry Module Refactor

Online Resource

NVIDIA Modulus Product Page

https://developer.nvidia.com/modulus

User Guide

https://docs.nvidia.com/deeplearning/modulus/index.html

NVIDIA Modulus Resource Center

https://resources.nvidia.com/l/en-us-modulus-pathfactory-explore-page

Physics Machine Learning のためのフレームワーク、NVIDIA Modulus 最新事情

https://medium.com/nvidiajapan/physics-machine-learning-%E3%81%AE%E3%81%9F%E3%82%81%E3%81%AE%E3%83%95%E3%83%AC%E3%83%BC%E3%83%A0%E3%83%AF%E3%83%BC%E3%82%AF-nvidia-modulus-%E6%9C%80%E6%96%B0%E4%BA%8B%E6%83%85-2734ea6b5ad4

NVIDIA GTC 2022

参加登録はこちらから

NVIDIA®