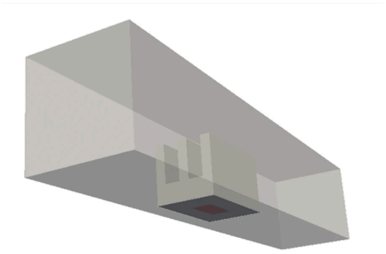
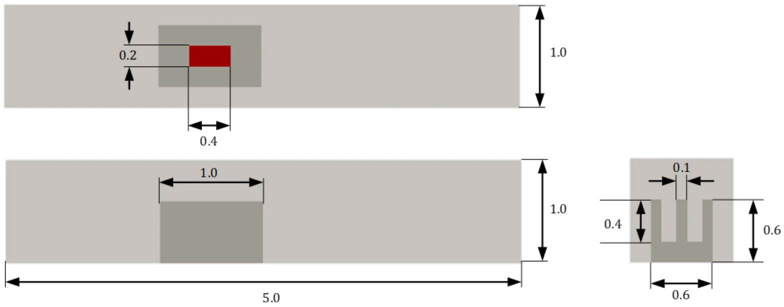
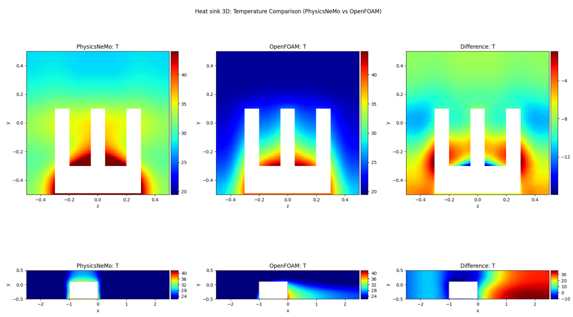
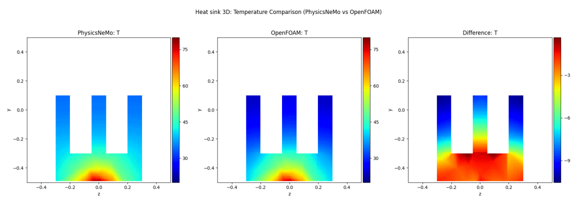
NVIDIA PhysicsNeMo (Modulus) Part 5: Conjugate Heat Transfer









https://docs.nvidia.com/deeplearning/modulus/modulus-sym/user_guide/advanced/conjugate_heat_transfer.html

https://catalog.ngc.nvidia.com/orgs/nvidia/teams/modulus/resources/modulus_sym_examples_supplemental_materials

https://github.com/NVIDIA/physicsnemo-sym

https://github.com/AI-ME-Ben/NVIDIA-Modulus-Reproduce/tree/main

Part 1 : Problem description

The geometry for a 3-fin heat sink placed inside a channel is shown in Fig. 144. The inlet to the channel is at $1\ m/s$. The pressure at the outlet is specified as 0 Pa. All the other surfaces of the geometry are treated as no-slip walls.

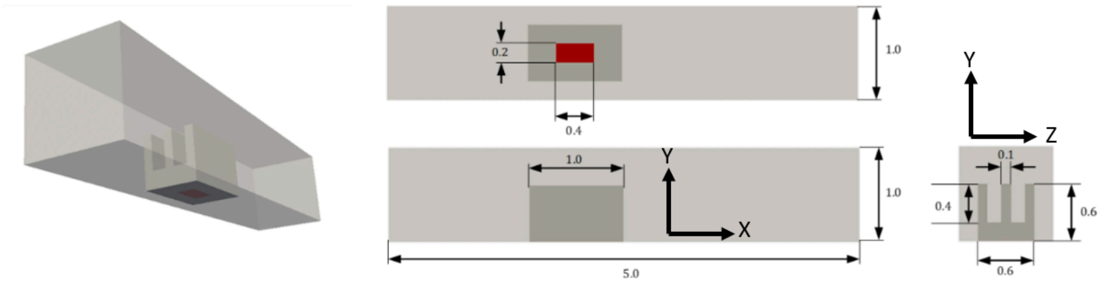


Fig. 144 *Three fin heat sink geometry (All dimensions in m)*

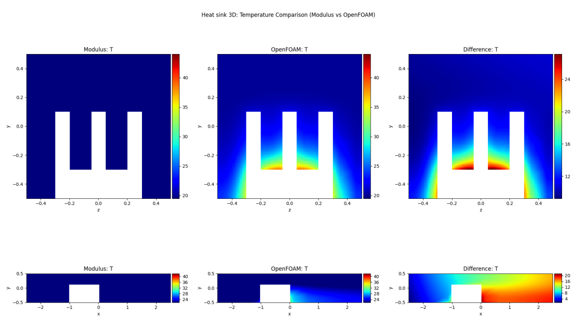
The inlet is at 273.15 K. The channel walls are adiabatic. The heat sink has a heat source of $0.2 \times 0.4\ m$ at the bottom of the heat sink situated centrally on the bottom surface. The heat source generates heat such that the temperature gradient on the source surface is 360 K/m in the normal direction. Conjugate heat transfer takes place between the fluid-solid contact surface.

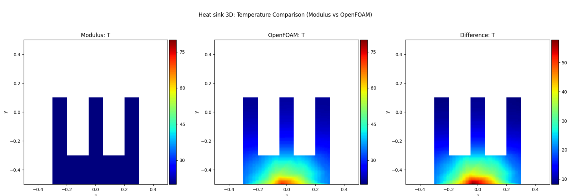
Table 5 Fluid and Solid Properties

Property	Fluid	Solid
Kinematic Viscosity (m^2/s)	0.02	NA
Thermal Diffusivity (m^2/s)	0.02	0.0625
Thermal Conductivity $(W/m.K)$	1.0	5.0

Material	Bulk Conductivity (W/mK)
Silver, Pure	418.0
Copper 11000	388.0
Aluminum 6061 T6	167.0
Zinc, Pure	112.2
Iron, Cast	55.0
Solder, 60% Tin	50.0
Titanium	15.6
ThermalGrease,T660	0.90
Fiberglass	0.040
Air, stp	0.025

Part 2 : Implementation





https://catalog.ngc.nvidia.com/orgs/nvidia/teams/physicsnemo/containers/physicsnemo

1. docker run --shm-size=1g --ulimit memlock=-1 --ulimit stack=67108864 --runtime nvidia -p 8888:8888 -p 7007:7007 --memory=16g -it nvcr.io/nvidia/physicsnemo/physicsnemo:25.03 bash -c "jupyter notebook --ip=0.0.0.0 --port=8888 --no-browser --allow-root"

```
https://github.com/NVIDIA/physicsnemo-sym
```

https://github.com/AI-ME-Ben/NVIDIA-Modulus-Reproduce/tree/main

https://catalog.ngc.nvidia.com/orgs/nvidia/teams/modulus/resources/modulus_sym_examples_supplemental_materials

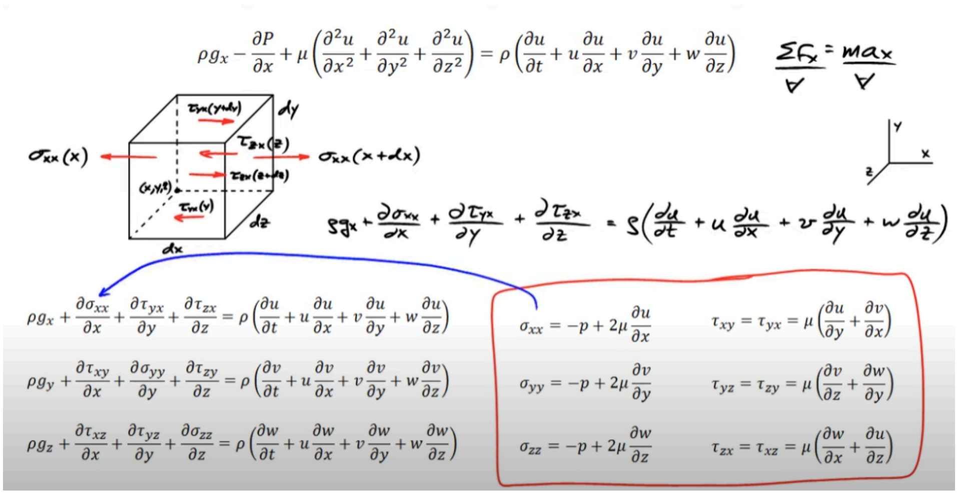
three_fin_flow.py
three_fin_geometry.py
three_fin_thermal.py

Part 3 : Physics equation

1. Modified Navier-Stokes equation → Flow

```python
# make navier stokes equations
if cfg.custom.turbulent:
    ze = ZeroEquation(nu=0.002, dim=3, time=False, max_distance=0.5)
    ns = NavierStokes(nu=ze.equations["nu"], rho=1.0, dim=3, time=False)
    navier_stokes_nodes = ns.make_nodes() + ze.make_nodes()
else:
    ns = NavierStokes(nu=0.01, rho=1.0, dim=3, time=False)
    navier_stokes_nodes = ns.make_nodes()
normal_dot_vel = NormalDotVec()
```

solve u, p



Navier-Stokes Equations with Zero Equation Turbulence Model

The standard **Navier-Stokes equations** with viscosity ν are:

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\frac{1}{\rho}\nabla p + \nabla \cdot (\nu_{\text{eff}} \nabla \mathbf{u}) + \mathbf{f}$$

https://www.youtube.com/watch?v=NjoMoH51UZc

https://docs.nvidia.com/deeplearning/modulus/modulus-sym/user_guide/foundational/zero_eq_turbulence.html

https://www.youtube.com/watch?v=zf0jU25uWqo&t=4s

2. Advection diffusion equation → Heat

```
# make thermal equations
ad = AdvectionDiffusion(T="theta_f", rho=1.0, D=0.02, dim=3, time=False)
dif = Diffusion(T="theta_s", D=0.0625, dim=3, time=False)
dif_inteface = DiffusionInterface("theta_f", "theta_s", 1.0, 5.0, dim=3, time=False)
f_grad = GradNormal("theta_f", dim=3, time=False)
s_grad = GradNormal("theta_s", dim=3, time=False)
```

known u, solve C

4. Interpretation of the Terms

$$\underbrace{\frac{\partial C}{\partial t}}_{\text{Time variation}} + \underbrace{\mathbf{u} \cdot \nabla C}_{\text{Advection}} = \underbrace{D\nabla^2 C}_{\text{Diffusion}} + \underbrace{S}_{\text{Source/Sink}}$$

- **Time variation** ($\frac{\partial C}{\partial t}$): Change in C over time.
- **Advection** ($\mathbf{u} \cdot \nabla C$): Transport of C by the velocity field.
- **Diffusion** ($D\nabla^2 C$): Spreading of C due to random molecular motion.
- **Source/Sink** (S): External sources (e.g., chemical reactions, heat generation).

General Form of the Diffusion Equation

The **diffusion equation** is given by:

$$\frac{\partial C}{\partial t} = D\nabla^2 C + Q$$

where:

- $C(x, y, z, t)$ is the concentration (or temperature, or other dependent variable) at position (x, y, z) and time t.
- D is the **diffusion coefficient** (or **thermal diffusivity** for heat transfer problems).
- ∇^2 is the **Laplacian operator**, which in Cartesian coordinates is:

$$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$$

- $Q(x, y, z, t)$ is a **source term**, representing external sources or sinks (such as chemical reactions, heating sources, etc.).

Diffusion Interface Boundary Conditions

1. Dirichlet Condition (Continuity of Variable)

- Ensures the variable (e.g., temperature or concentration) is continuous across the interface.

$$T_1 = T_2 \quad \text{at the interface}$$

2. Neumann Condition (Flux Conservation)

- Ensures the flux is continuous across the interface, preserving physical conservation laws.

$$D_1 \nabla T_1 \cdot \mathbf{n} = D_2 \nabla T_2 \cdot \mathbf{n}$$

Key Takeaways:

✅ No sudden jumps in the variable.
✅ Flux must be conserved at the interface.
✅ Critical for heat transfer, diffusion, and multi-material simulations.

three_fin_thermal.py

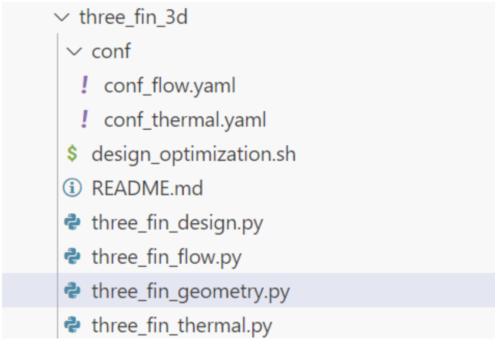
https://www.youtube.com/watch?v=4EVKEImJtsg

https://www.youtube.com/watch?v=8ew_ZRNkRWU&t=219s

https://en.wikipedia.org/wiki/Convection%E2%80%93diffusion_equation

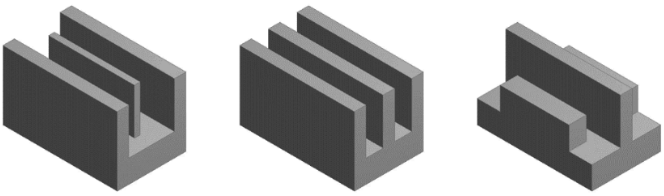
---

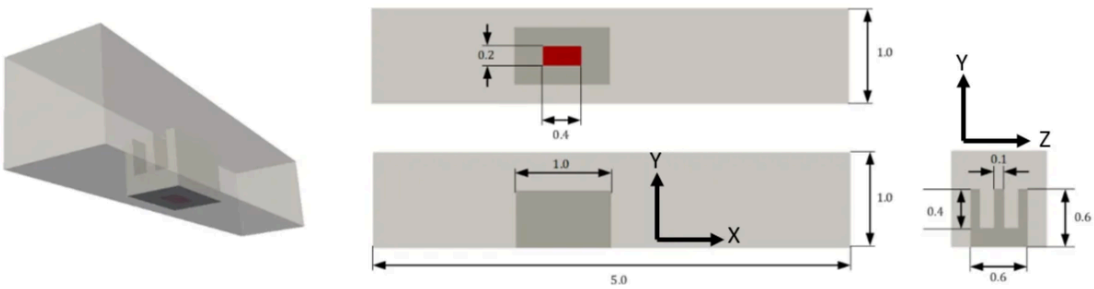
Part 4 : Code walkthrough

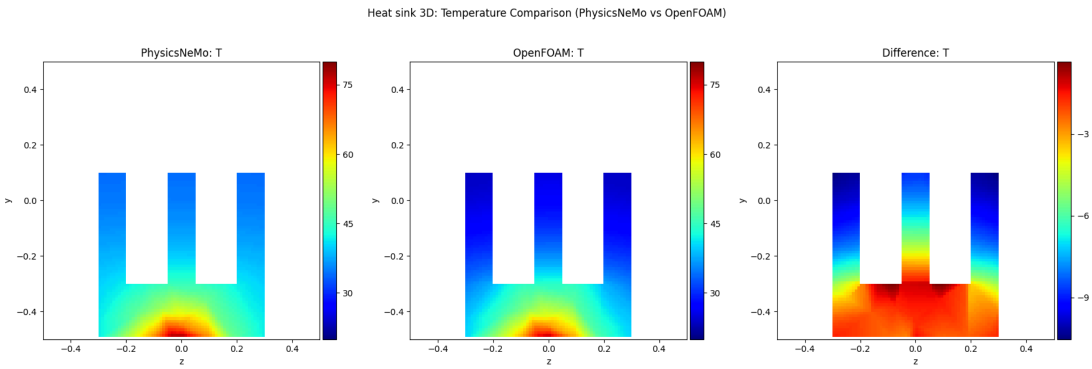


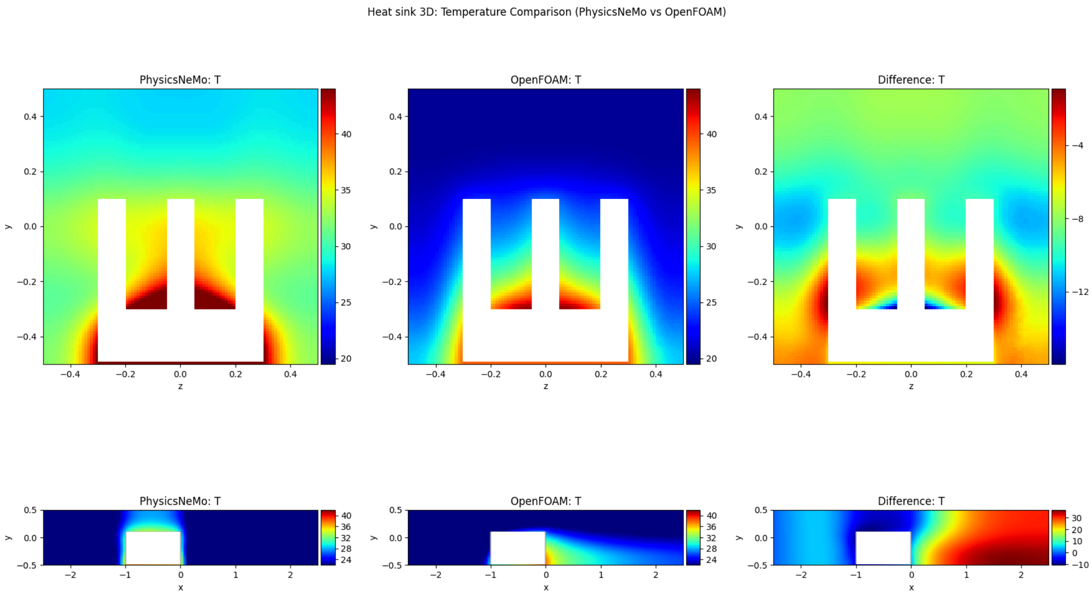
https://docs.nvidia.com/deeplearning/physicsnemo/physicsnemo-sym/user_guide/advanced/parametrized_simulations.html

$$
\begin{aligned}
h_{centralfin} &= (0.0, 0.6), \\
h_{sidefins} &= (0.0, 0.6), \\
l_{centralfin} &= (0.5, 1.0) \\
l_{sidefins} &= (0.5, 1.0) \\
t_{centralfin} &= (0.05, 0.15) \\
t_{sidefins} &= (0.05, 0.15)
\end{aligned}
$$








Heat sink 3D: Temperature Comparison (PhysicsNeMo vs OpenFOAM)
PhysicsNeMo: T
OpenFOAM: T
Difference: T
PhysicsNeMo: T
OpenFOAM: T
Difference: T