Machine learning approach to computational fluid dynamics

Intern: Bing-Jyun Tsao  Mentor: Shirui Luo
NCSA SPIN, University of Illinois at Urbana-Champaign

Introduction
In computational fluid dynamics (CFD), the numerical simulations of complex dynamical systems are often computationally expensive. Wide research in large-eddy simulations (LES) proposed several models to recover the unrepresented physics found in higher fidelity simulations. Instead of using existing models, this project utilizes machine learning (ML) to train deep neural networks to recover the unrepresented physics in the original simulation while retaining the efficiency in computation.

Direct Numerical Simulation (DNS)
To find the velocity field \( u \) and the pressure \( p \) for a fluid flow, a direct numerical solve of the Navier-Stokes (NS) equation is implemented.

\[
\frac{\partial u_i}{\partial t} + \frac{\partial}{\partial x_j} \left( u_i u_j \right) = -\frac{1}{\rho} \frac{\partial p}{\partial x_i} + \mu \frac{\partial}{\partial x_j} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{2}{3} \frac{\partial u_k}{\partial x_k} \delta_{ij} \right)
\]

with the divergence-free constraint:

\[
\frac{\partial u_i}{\partial x_i} = 0
\]

(\( \rho \) is the density, \( \mu \) is the viscosity, \( \delta_{ij} \) is the Kronecker delta)

The equations are solved using Chorin’s fractional step scheme with the fourth order Runge-Kutta (RK4) method as the forwarding method and the biconjugate gradient stabilized (BiCGSTAB) method as the Poisson solver for pressure.

Issue with down-sampling:
When down-sampling a simulation to a lower resolution grid, additional term in the equation requires closure, and some physical features that depend on the scale of the simulation are unrepresented.

Large-Eddy Simulation (LES)
Large-Eddy simulation adds in a sub-grid scale (SGS) stress tensor \( \tau_{ij} \) into the model for closure:

\[
\frac{\partial \bar{u}_i}{\partial t} + \frac{\partial \bar{u}_i \bar{u}_j}{\partial x_j} = -\frac{1}{\rho} \frac{\partial \bar{p}}{\partial x_i} + \frac{\mu}{\rho} \frac{\partial}{\partial x_j} \left( \frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_j}{\partial x_i} - \frac{2}{3} \frac{\partial \bar{u}_k}{\partial x_k} \delta_{ij} \right) + \frac{\partial \tau_{ij}}{\partial x_j}
\]

(\( \bar{u}_i \) is the filtered velocity, \( \bar{p} \) is the filtered pressure)

In this project, the Smagorinsky model is used for the SGS stress tensor.

Machine Learning (ML)
Instead of using existing model for the SGS tensor, the ML model trains a deep neural network to implement the unrepresented physics \( h_g \) in addition to the original equation \( f_u \):

\[
\frac{\partial u}{\partial t} = f_u(u, u_x, u_xx) + h_g(u, u_x, u_xx)
\]

(\( \nu \) is the scenario parameter, \( \theta \) is the neural network parameter)

The neural network is trained by calculating the gradient of the loss function using the adjoint method. A velocity correction method is also used to eliminate discretization error when down-sampling the grid.

PyFlow
PyFlow is a Python-based platform that produces computational fluid dynamics simulations on local environments as well as supercomputers.

Results
The result shows the simulations from three models DNS, LES, and ML in a \( 32^3 \) grid with a target data (labeled DS) downsampled from a simulation with \( 128^3 \) grid.

The plot on the left shows the velocity kinetic energy comparison (N = 32) between DNS and ML.

The plot on the right shows the turbulence kinetic energy decay of all three models.

Future
- Improve ML model to exceed the performance of the LES model
- Run longer simulation to find convergence for the ML model
- Test ML model on different initial conditions

References