Shock Capturing using Neural Networks

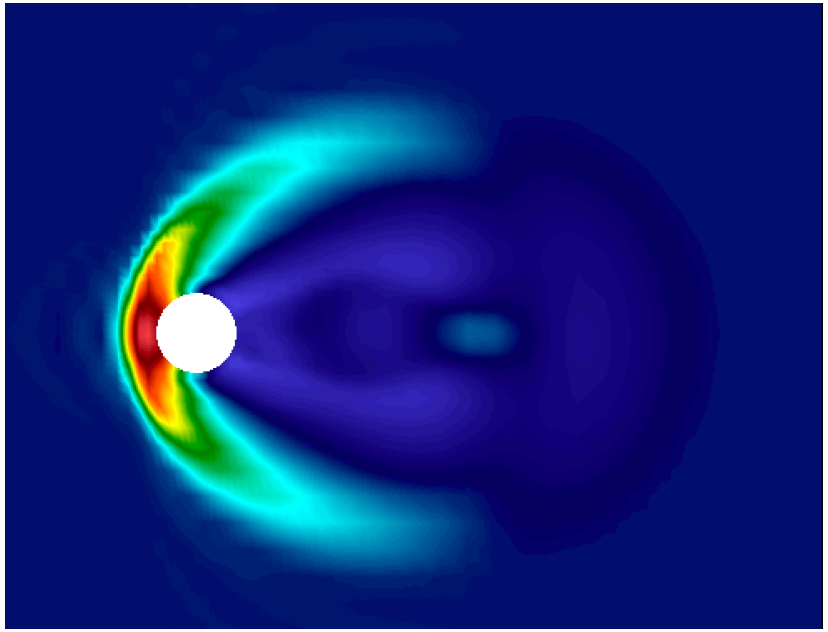
Aaron Larsen
Mentor: Britton Olson

August 27, 2020

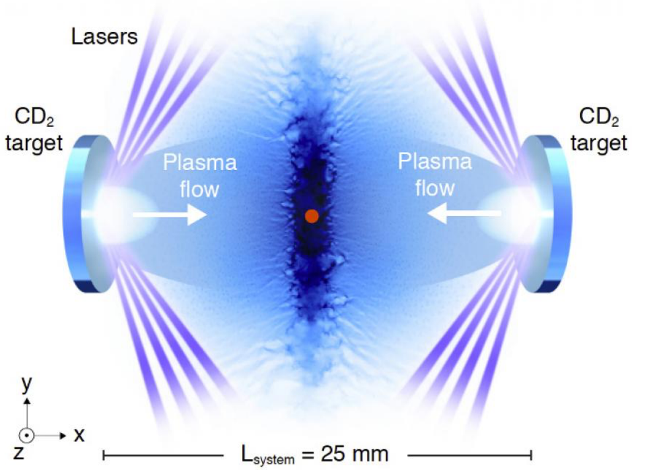
LLNL-PRES-813795
This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC

Introduction: Shock waves occur in many mediums and have a large impact on experimental design

- **Shock waves** are a sharp change in pressure that moves through a medium
- **Shock waves** carry energy, which dissipates at the front of the shock wave
- **Shock waves** have a great impact on a wide variety of engineering and scientific applications

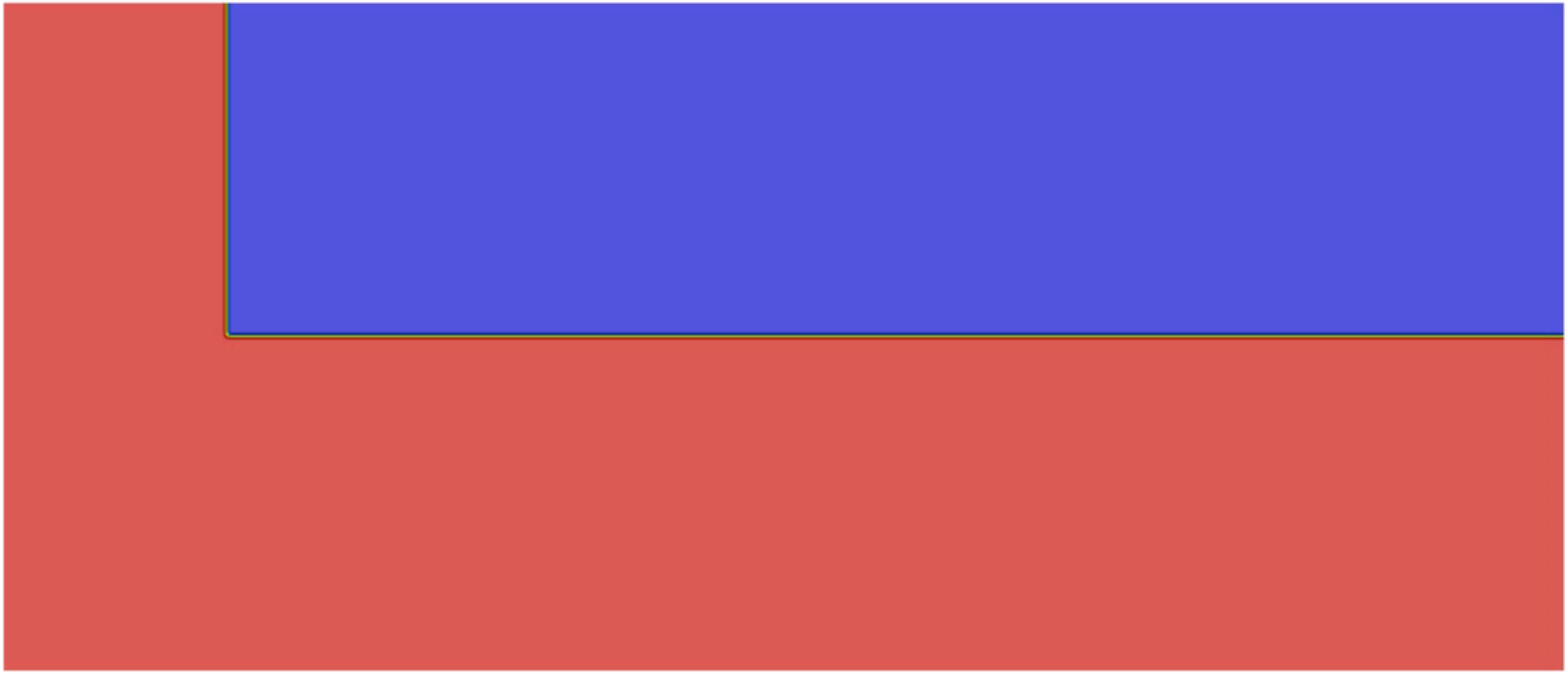

Supersonic flow past a cylinder at Mach 2


Jet flying at supersonic speeds



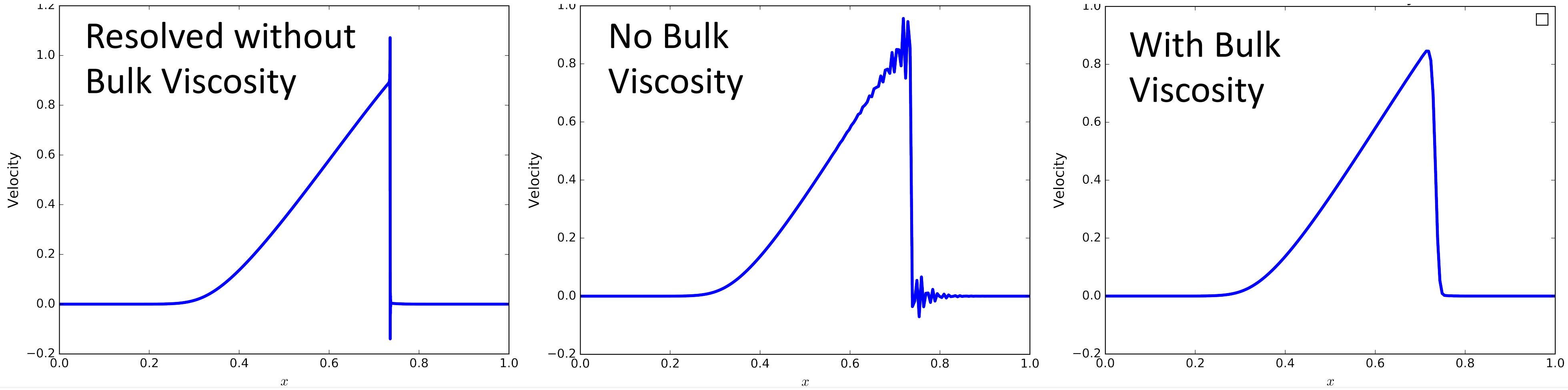
Simulating shock waves in supernova remnants

Introduction: Shock waves occur in many mediums



Introduction: Shock capturing is necessary to resolve shock-dominated problems

- In non-linear hyperbolic PDEs like the Euler equations, shock waves become unresolvable singularities
- Differentiating across a shock leads to Gibbs Oscillations
- Error manifests as oscillations occur because of unresolved features
- **Shock capturing** is used in hydrodynamic simulations to numerically resolve shock waves



Miranda uses a high-order artificial viscosity operator for shock capturing

- **Artificial viscosity (AV)** is a type of shock capturing used in simulations and can be computationally expensive to compute
- AV creates features that are resolved, thus making unresolved shock waves resolved
- Miranda solves the hydrodynamics equations to high-order accuracy in space (10th) and time (4th)
- Calculating AV and other artificial diffusivities in Miranda can account for >50% of the runtime
- AV operator: $\beta^* = C_\beta \rho \overline{\left\| \frac{\partial^r}{\partial x^r} (\nabla \cdot \mathbf{u}) \right\|} \Delta x^{r+2}$

Tools

- TensorFlow and Keras
  - Open source machine learning platform
  - Developed by Google
  - Python package

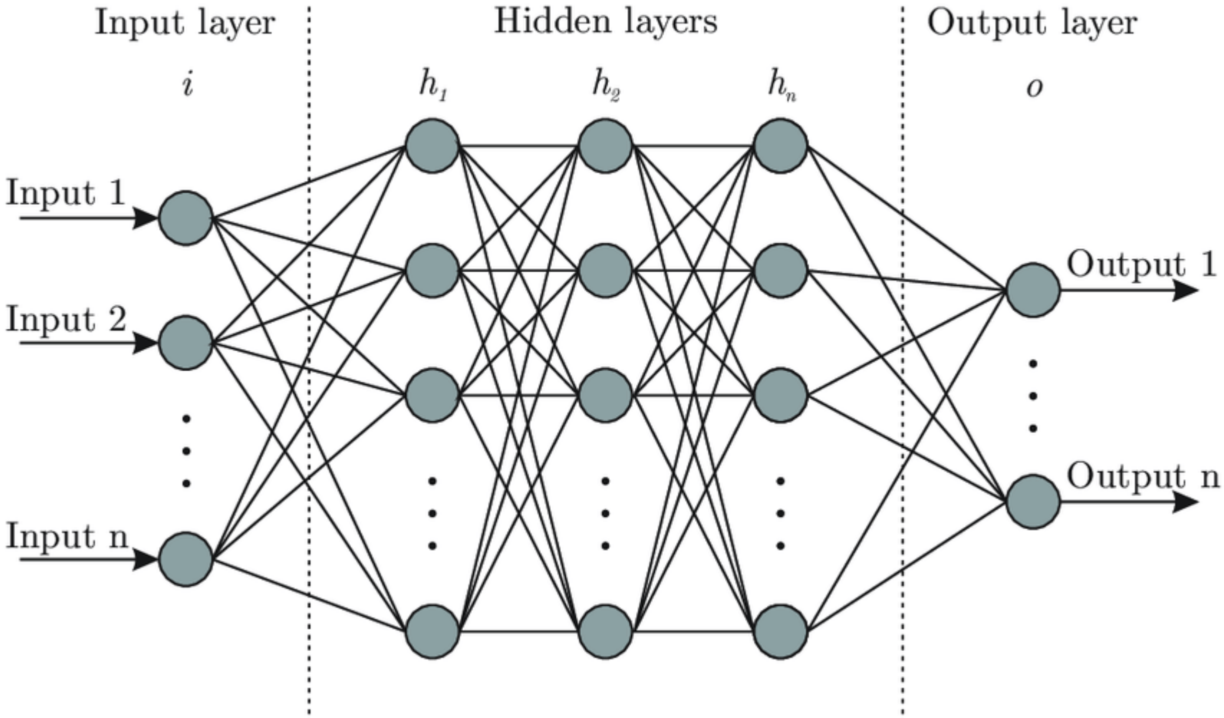
TensorFlow

- Pyranda
  - Mini-App of Miranda
  - Solves PDEs using 4th order Runge-Kutta in time, 10th order finite difference in space
  - Specify equations of motion and initial conditions and grid spacing

- Python
  - Scipy – Analysis via error norms
  - Matplotlib – Visualization

Neural networks can be used as a regression model

- Artificial neural networks (NN) are computer models that mimic the structure of the human brain composed of layers.
- Perceptrons (nodes) compose each layer of the NN
- Each perceptron learns weights in order to maximize the objective function
- These weights are determined through the analysis of training datasets.
- Regression: A NN uses weights learned through training to predict the outputs based on inputs



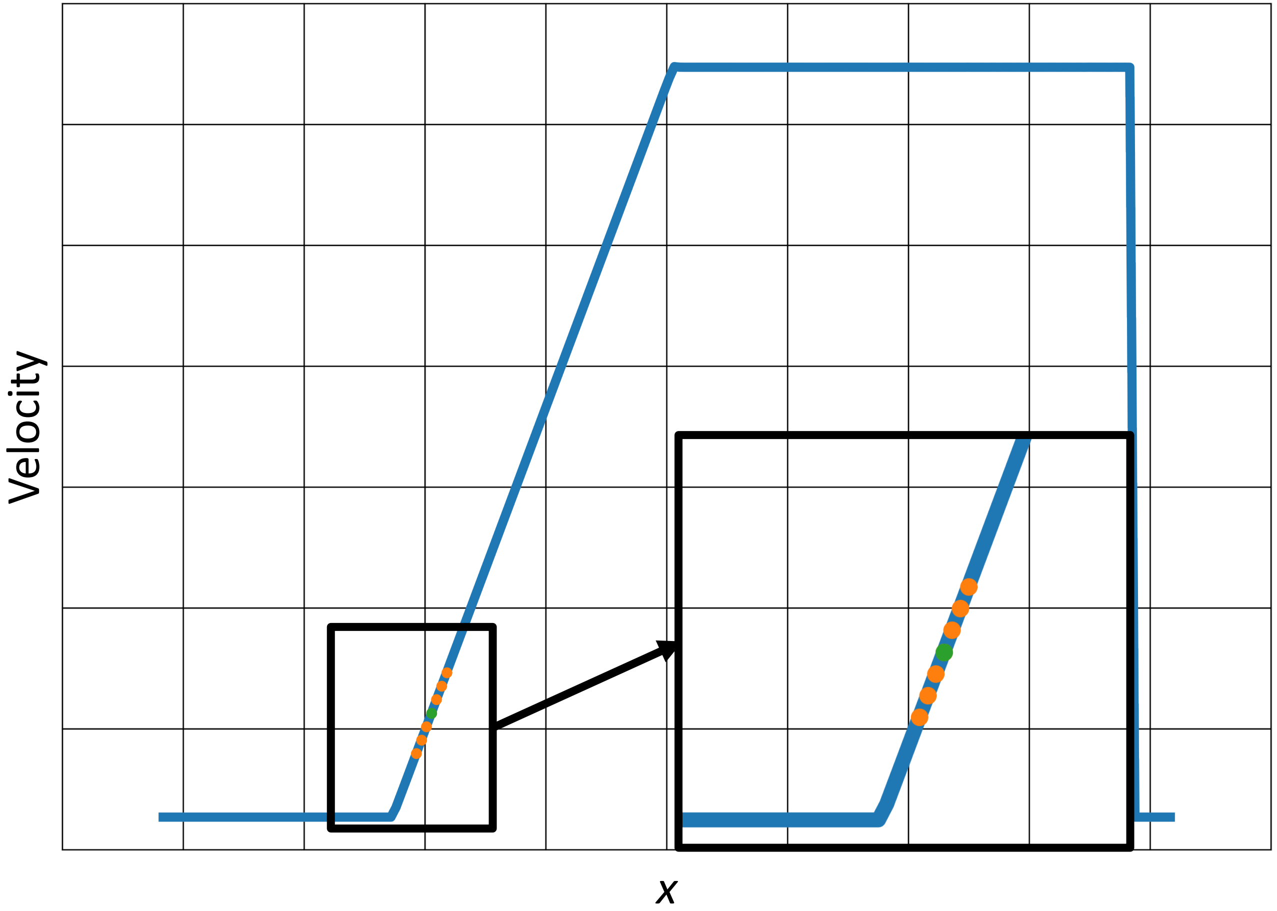
Primary objectives for summer research project

1. Can a NN accurately predict AV?
    - Gather representative training data
    - Train a neural network model
    - Apply the model to shock dominated problems and assess its accuracy

2. Can a NN be optimized to decrease the computational cost of computing AV?

Creating a training dataset with shock-dominated test problems

1. A shock-dominated test problem using the traditional AV operator
2. Use a stencil to make an array of velocity values near the point of interest
   a. Velocity values before and after the point are collected
3. The corresponding AV value for the point of interest is collected
4. $$data[0, j] = \beta[i]$$
   $$data[1:7, j] = u[i-3:i+3]$$



Using nondimensionalization to create a universal model

- Due to different scales of shock-dominated problems, a model trained with a specific Mach number, nondimensionalizing needs to be used
- Velocity
  - $u^* = u/c_s$
- Artificial Viscosity
  - $\beta^* = \beta/\rho/c_s/\Delta x$
- NOTE: In situations where the shock occurs in multiple directions, symmetrical data gathering is needed
  - Using a 1D simulation
    - Collect from left to right
    - Duplicate and flip

Training a neural network to approximate the AV operator

- Software: TensorFlow — Keras
- Neural Network Structure
  — 3 Sequential layers
  — Each layer is dense (all nodes are interconnected)
  — ReLU activation function
  — Loss Function: MSE
- The neural network is reduced to a regression model
- 80% of the dataset collected from the shock-dominated problem was used as training data
- 20% of the dataset was used as validation data
- 100 epochs were used to generate the model

Implementation and Analysis of the neural-network-based AV operator

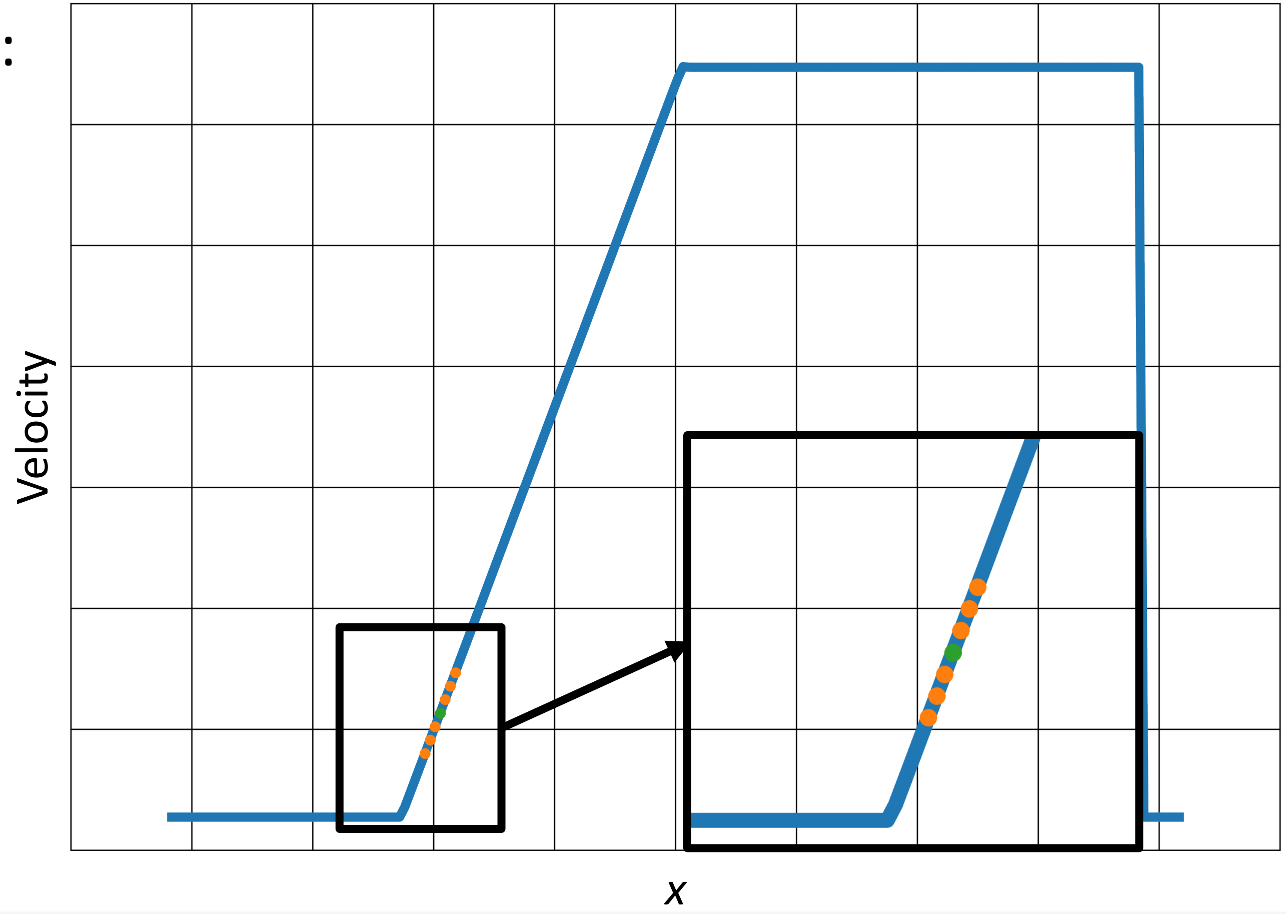
- AV operator: $\beta^* = C_\beta \rho \| \overline{\frac{\partial^r}{\partial x^r} (\nabla \cdot \mathbf{u})} \| \Delta x^{r+2}$ ⟹ NN-AV operator: $\beta^* = NN_{AV}(\mathbf{u})$

- During each step in 4th order Runge-Kutta:
  - Use a stencil to collect an array of velocity values for each point in the domain
    - u[i-3:i+3]
  - Use the NN to predict the AV values

$$\beta_{ML_i} = NN_{AV}(u[i-3:i+3])$$

  - Substitute the predicted AV values from NN model into the simulation

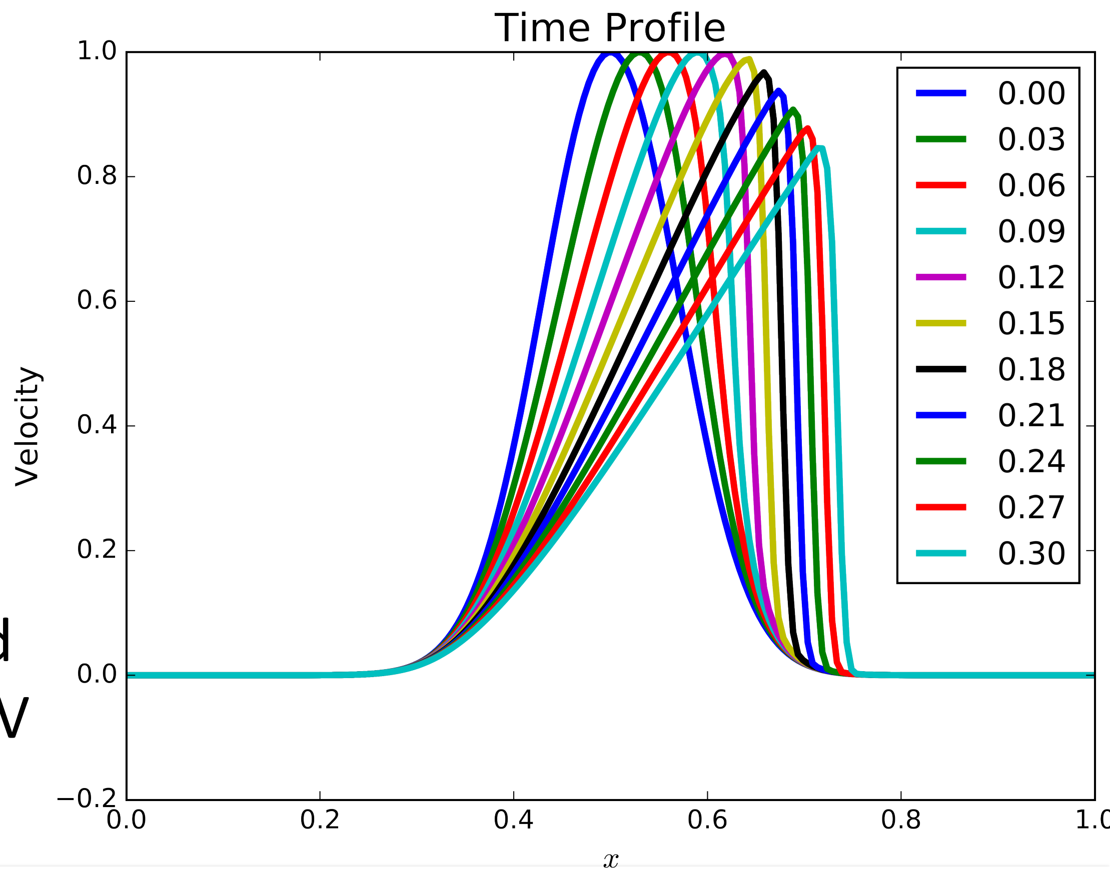
- Analysis
  - Compare NN-AV results and traditional AV data with highly resolved simulations using L_1, L_2, L_∞ errors in density



The Viscous Burgers' Equation

$$\frac{\partial u}{\partial t} = -u\frac{\partial u}{\partial x} + \nu\frac{\partial^2 u}{\partial x^2}$$

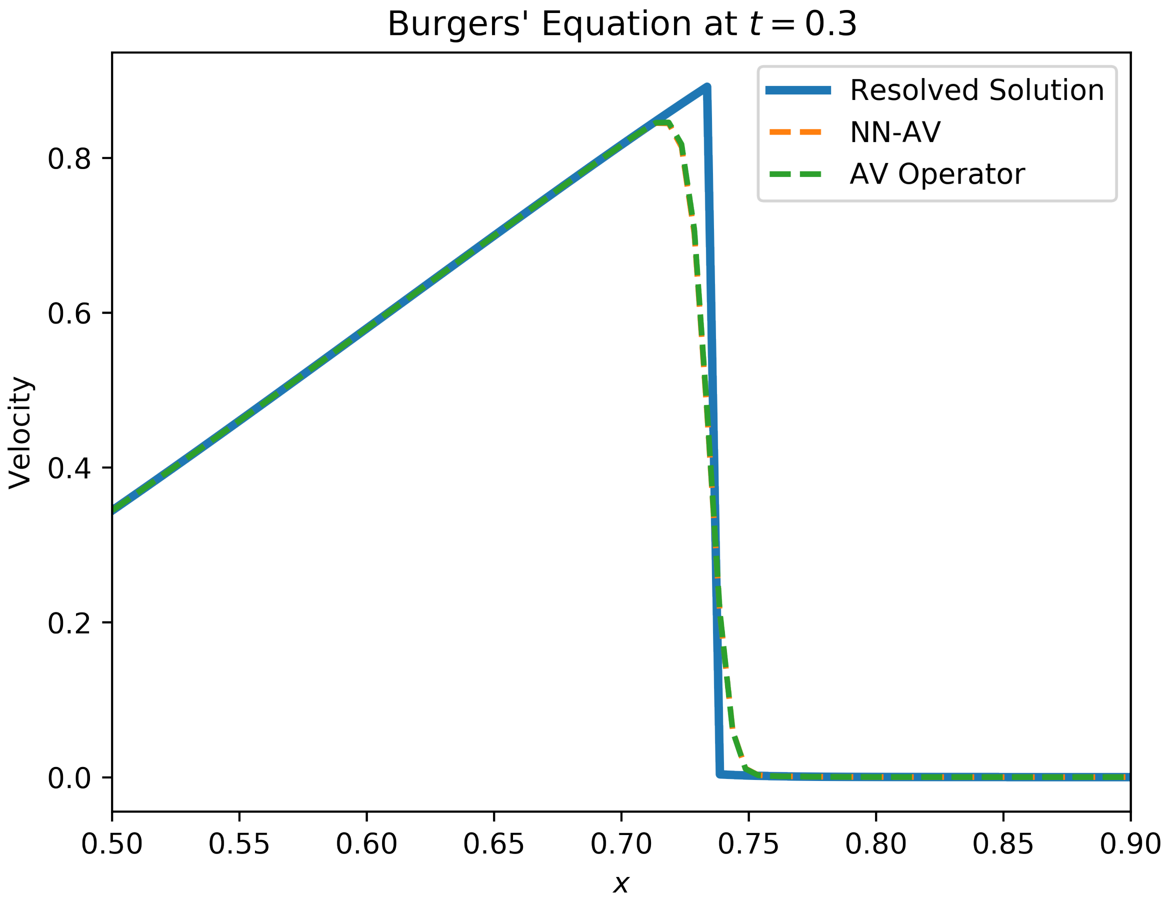
- Single variable hyperbolic PDE that allows for shock waves
- ν is the artificial viscosity term, no physical viscosity is used
- A neural network model was trained on a simple breaking wave
- This model was applied to the same problem and compared with the results from the traditional AV calculation.



Applying the NN-AV to the Viscous Burgers' Equation

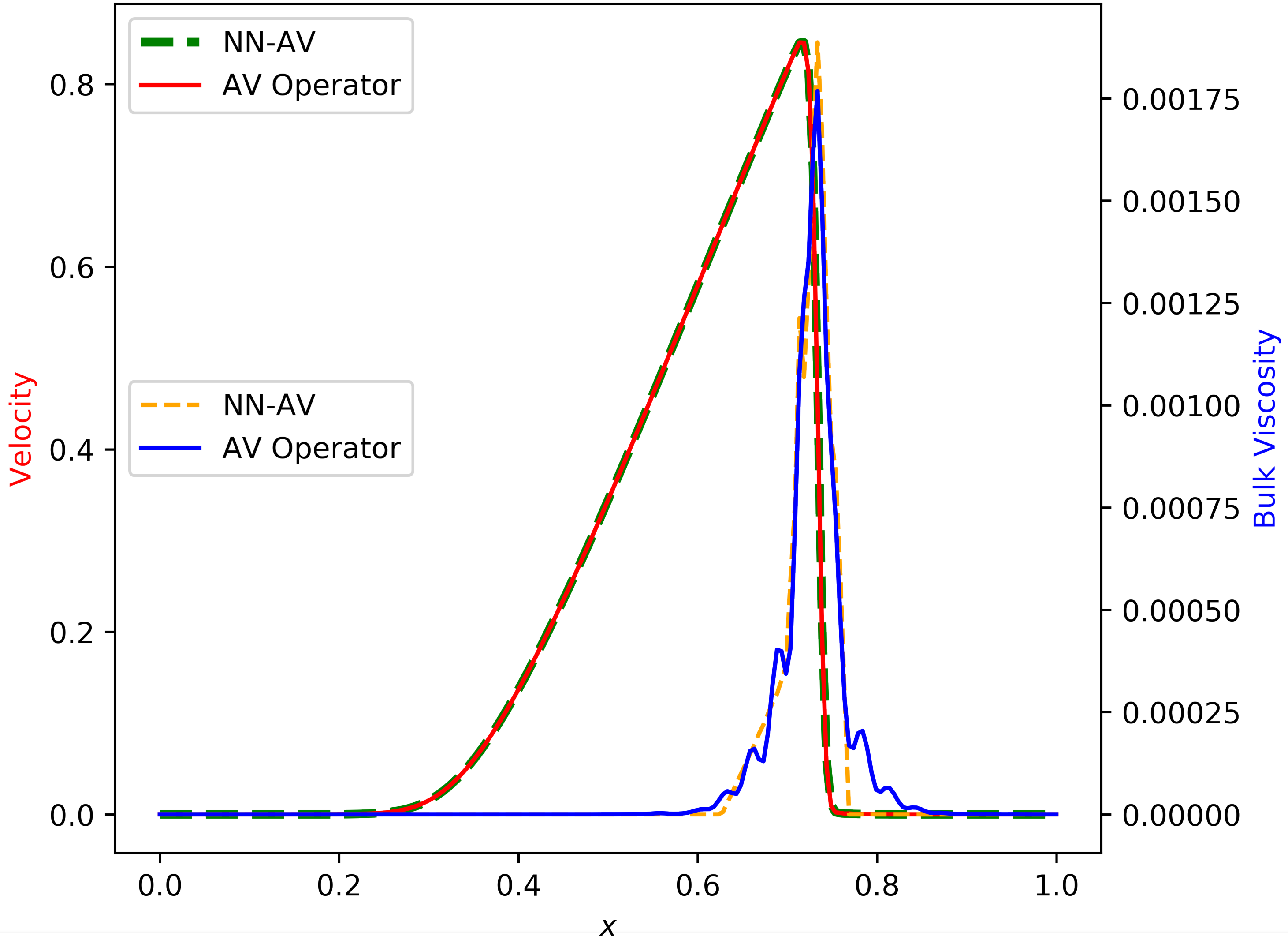
Relative Error in Velocity between AV Operator and NN-AV and Resolved Calculation*				
		L_1	L_2	L_∞
AV Operator	Traditional	2.690e-02	1.471e-02	1.242e-02
	NN-AV	2.659e-02	1.429e-02	1.184e-02

*Resolved calculation was run using the traditional AV operator and 10000 spatial points (50x)



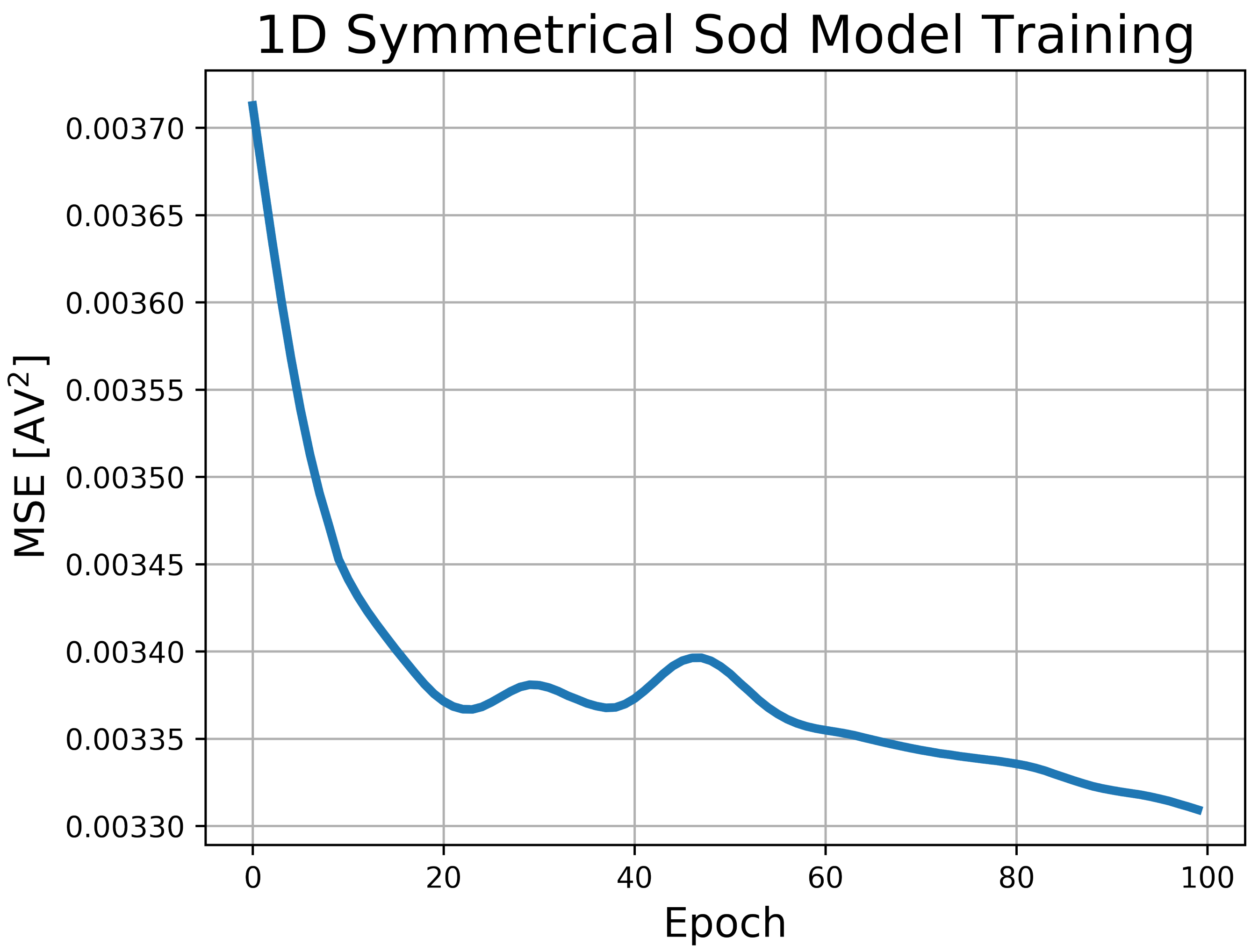
Viscous Burgers' Equation at *t=0.3*

- The NN model follows the same structure as the traditional AV operator
- The NN-AV has the proper scaling
- The shape of NN-AV is slightly different and has smooth discontinuities



Implementing a universal model based on the 1D Sod Shock Tube Problem

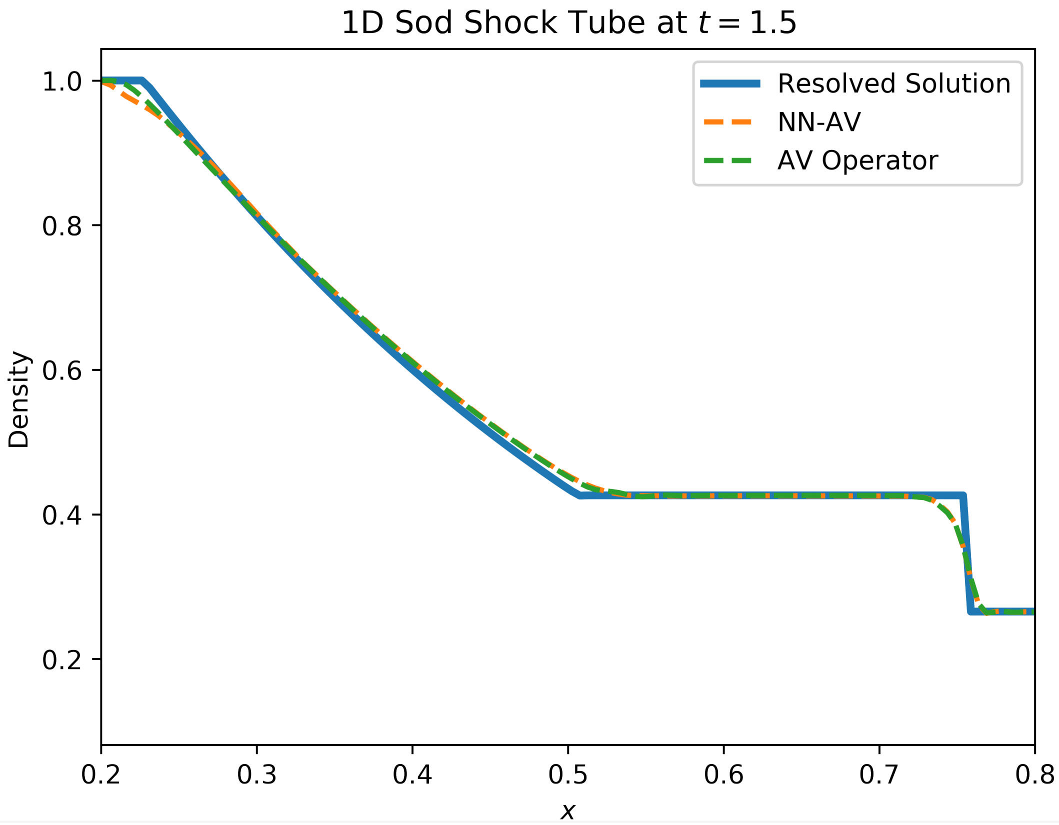
- When training a 1D model, it is biased for shocks in 1 direction
- This can be overcome by using mirroring to get symmetric training data, as though the shock was propagating in both directions
- **Epoch:** The number of iterations that the entire training dataset has been processed
- **MSE:** A loss function used to evaluate accuracy
- These are operations completed by TensorFlow



Applying the 1D Sod Shock Tube model to itself

Relative Error in Density between AV Operator and NN-AV and Resolved Calculation*				
		L_1	L_2	L_∞
AV Operator	Traditional	8.452e-03	1.286e-03	6.035e-04
	NN-AV	9.304e-03	1.338e-03	6.074e-04

*Resolved calculation was run using the traditional AV operator and 10000 spatial points (50x resolution)

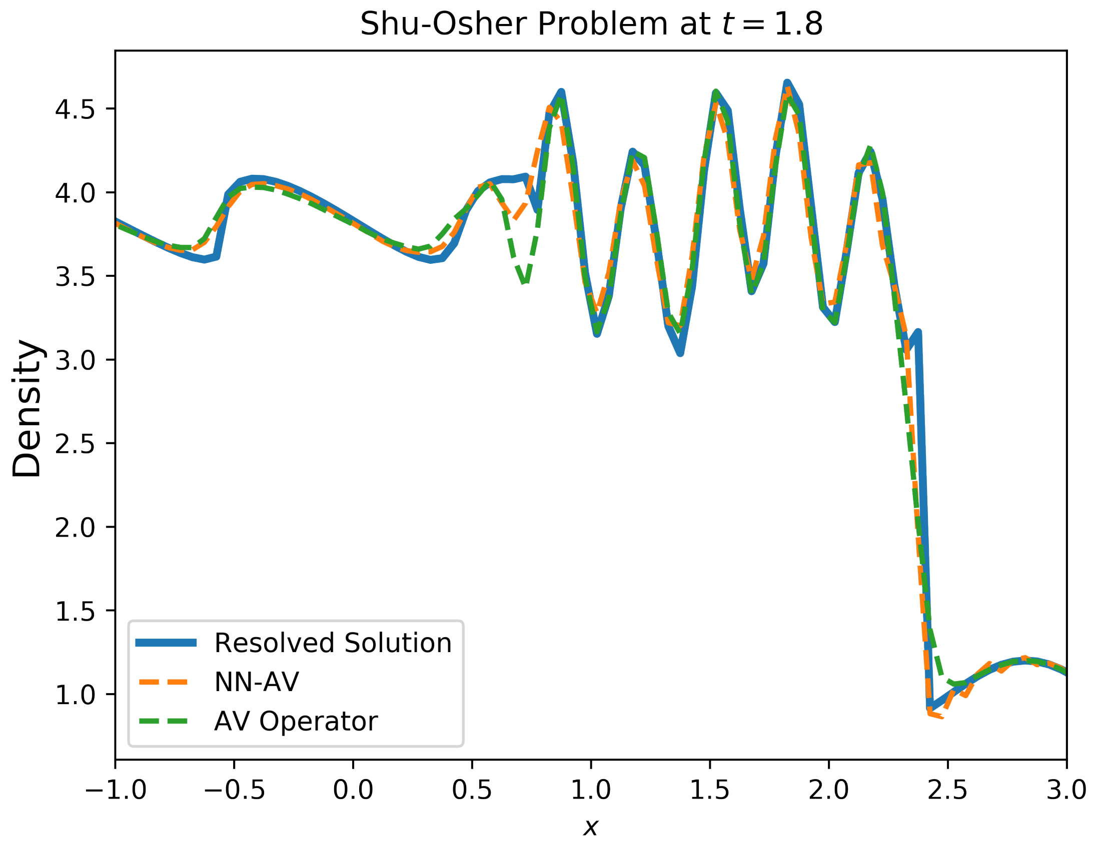


Applying the 1D Sod Shock Tube model to the Shu-Osher Problem

Relative Error in Density between AV Operator and NN-AV and Resolved Calculation*

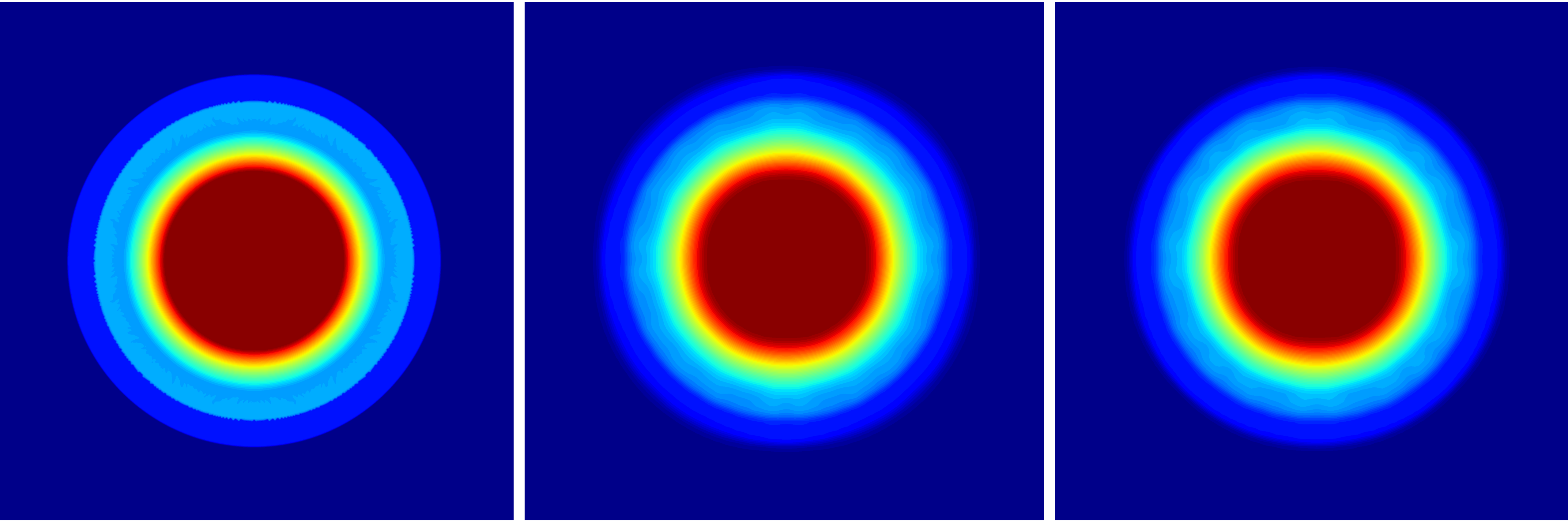
		L_1	L_2	L_∞
AV Operator	Traditional	1.665e-02	2.767e-03	1.998e-03
	NN-AV	1.439e-02	2.618e-03	2.119e-03

*Resolved calculation was run using the traditional AV operator and 10000 spatial points (50x resolution)



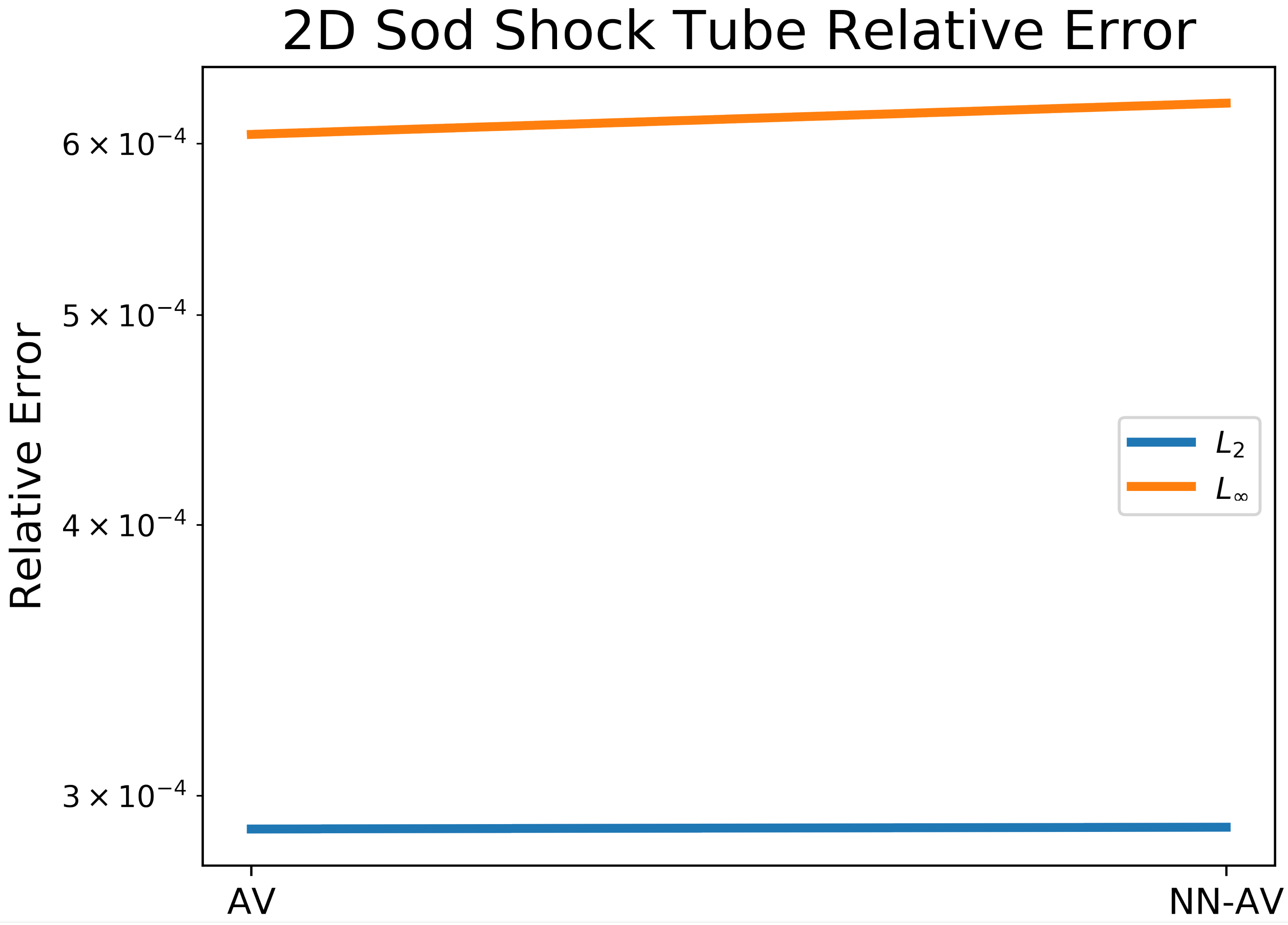
Applying the 1D Sod Shock Tube model to the 2D problem

Density



Resolved Simulation | **NN-AN Operator at t=0.4** | **AV Operator at t=0.4**

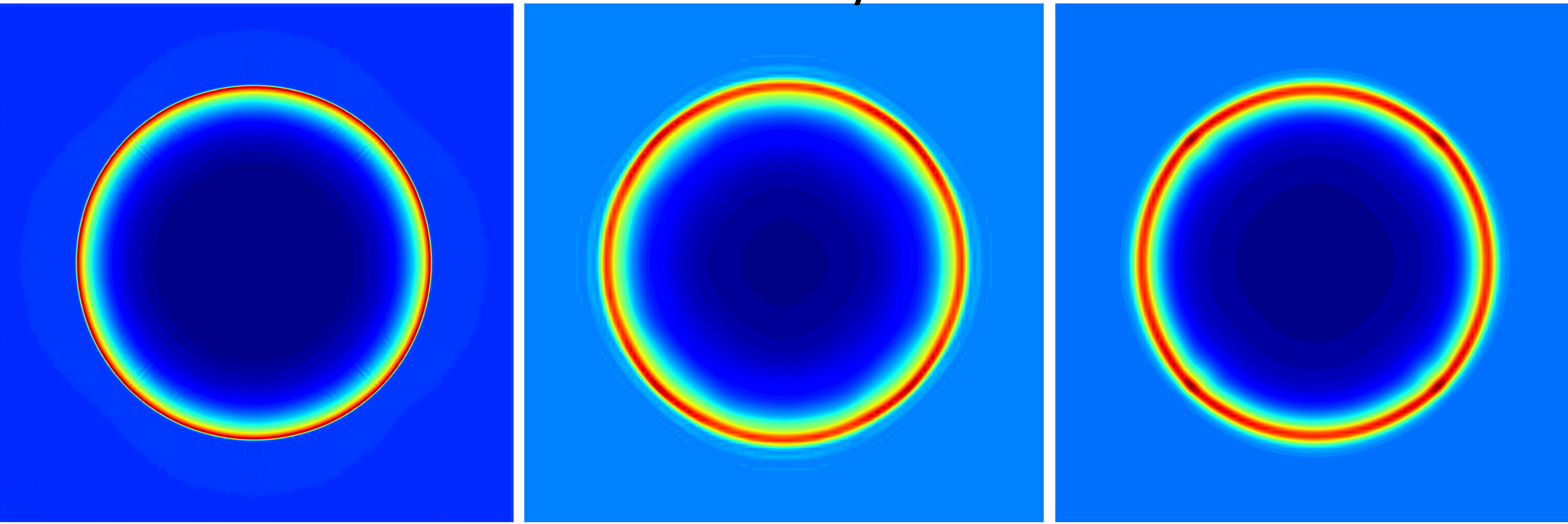
Relative Error in Density between AV Operator and NN-AV and Resolved Calculation* in the 2D Sod Shock Tube



*Resolved calculation was run using the traditional AV operator and 1048576 spatial points (64x resolution)

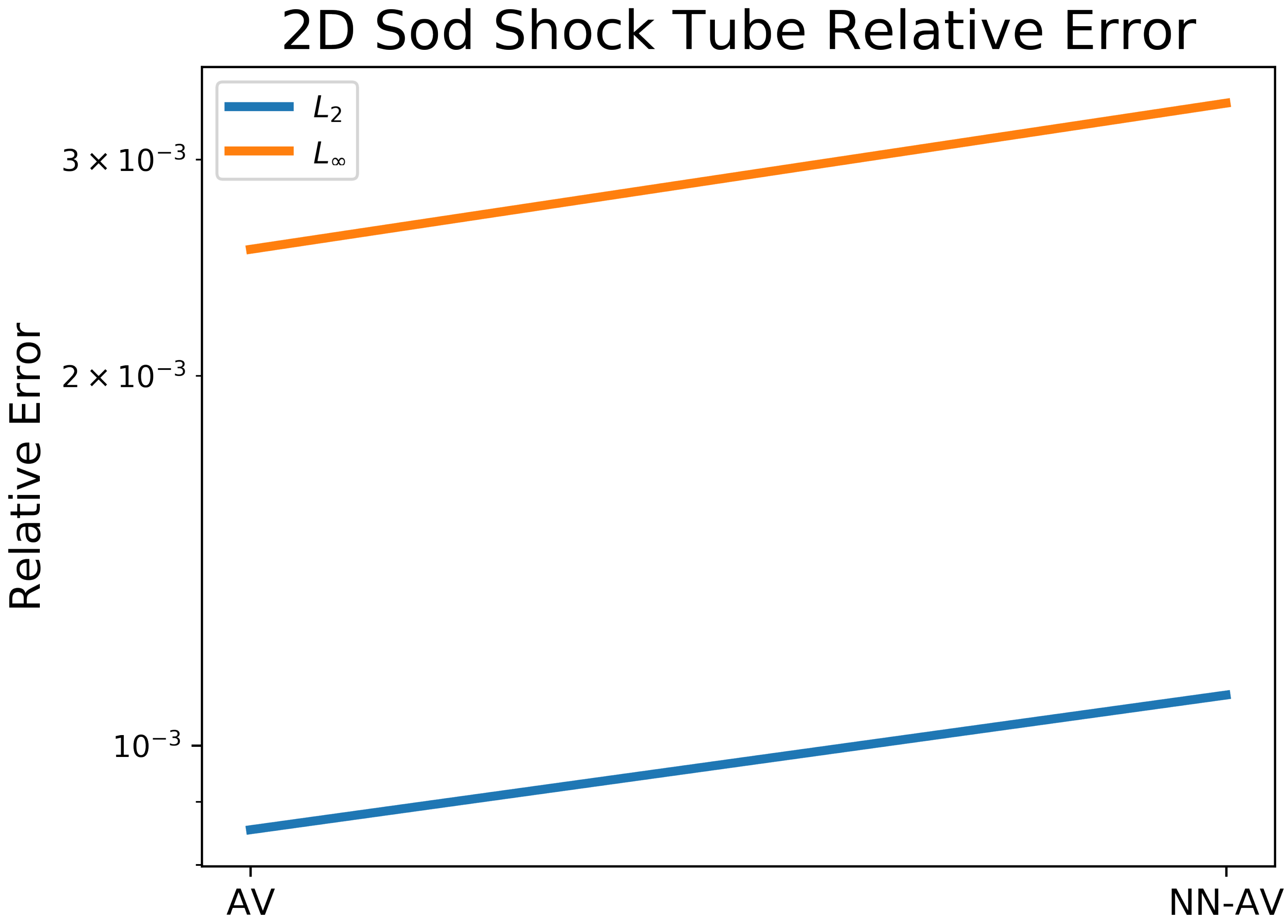
Applying the 1D Sod Shock Tube model to the Sedov Blast Wave

Density



Resolved Simulation **NN-AN Operator at t=0.4** **AV Operator at t=0.4**

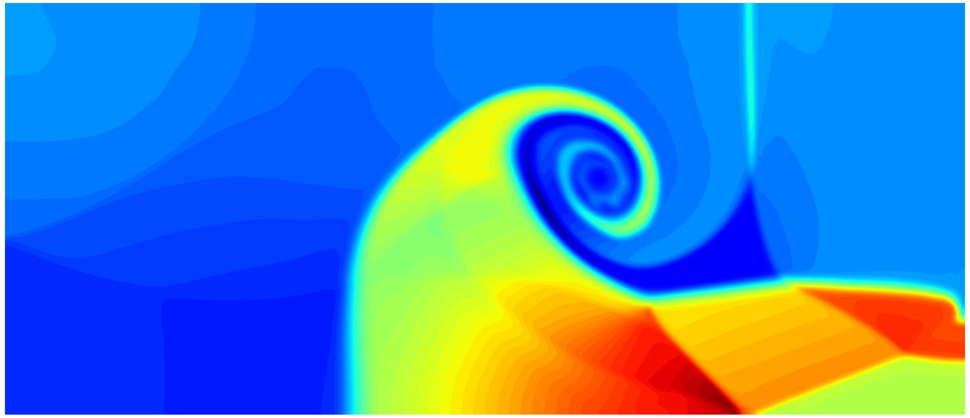
Relative Error in Density between AV Operator and NN-AV and Resolved Calculation* in the Sedov Blast Wave



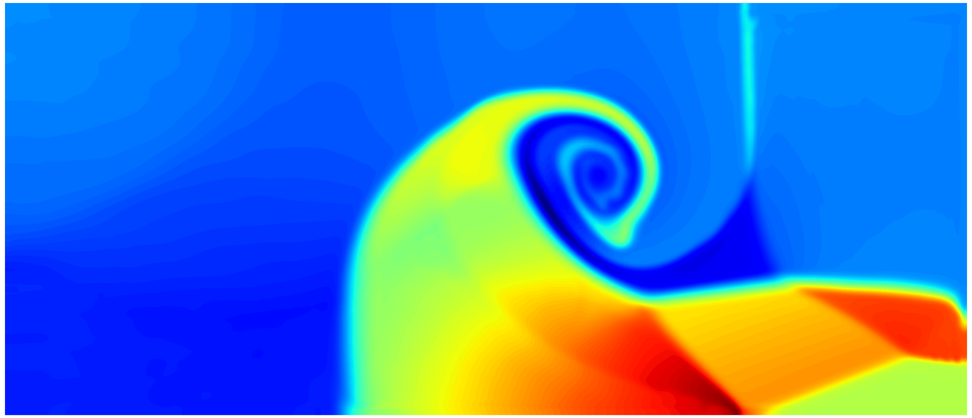
*Resolved calculation was run using the traditional AV operator and 1048576 spatial points (64x resolution)

Applying the 1D Sod Shock Tube model to the Triple Density Problem

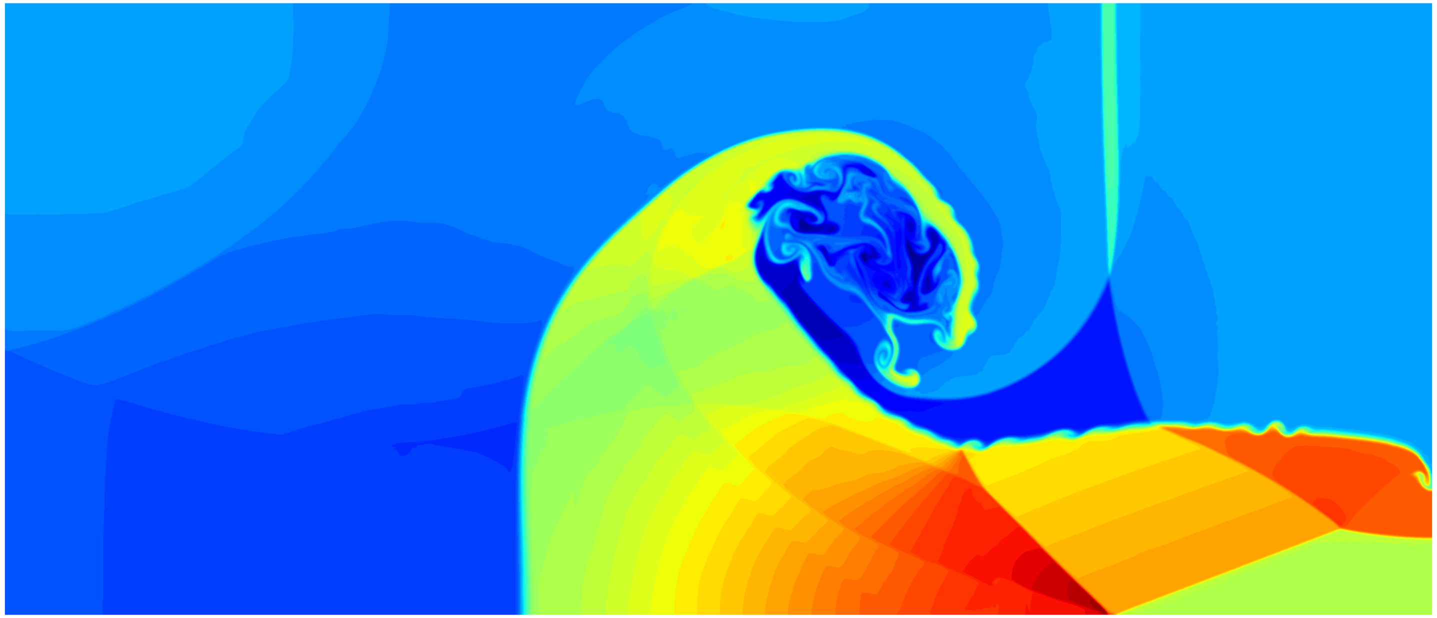
AV Operator at t=0.4



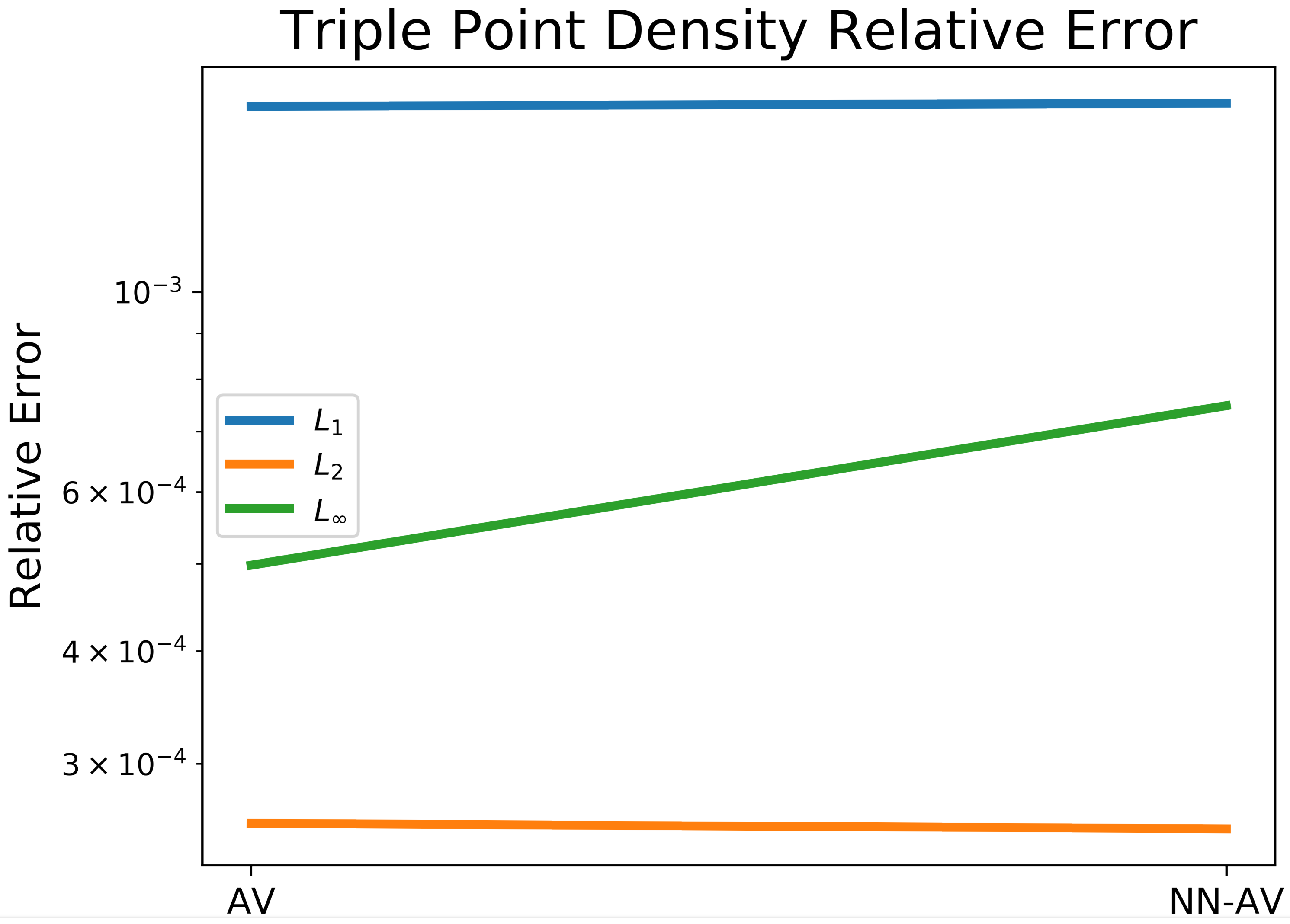
NN-AN Operator at t=0.4



Resolved Simulation

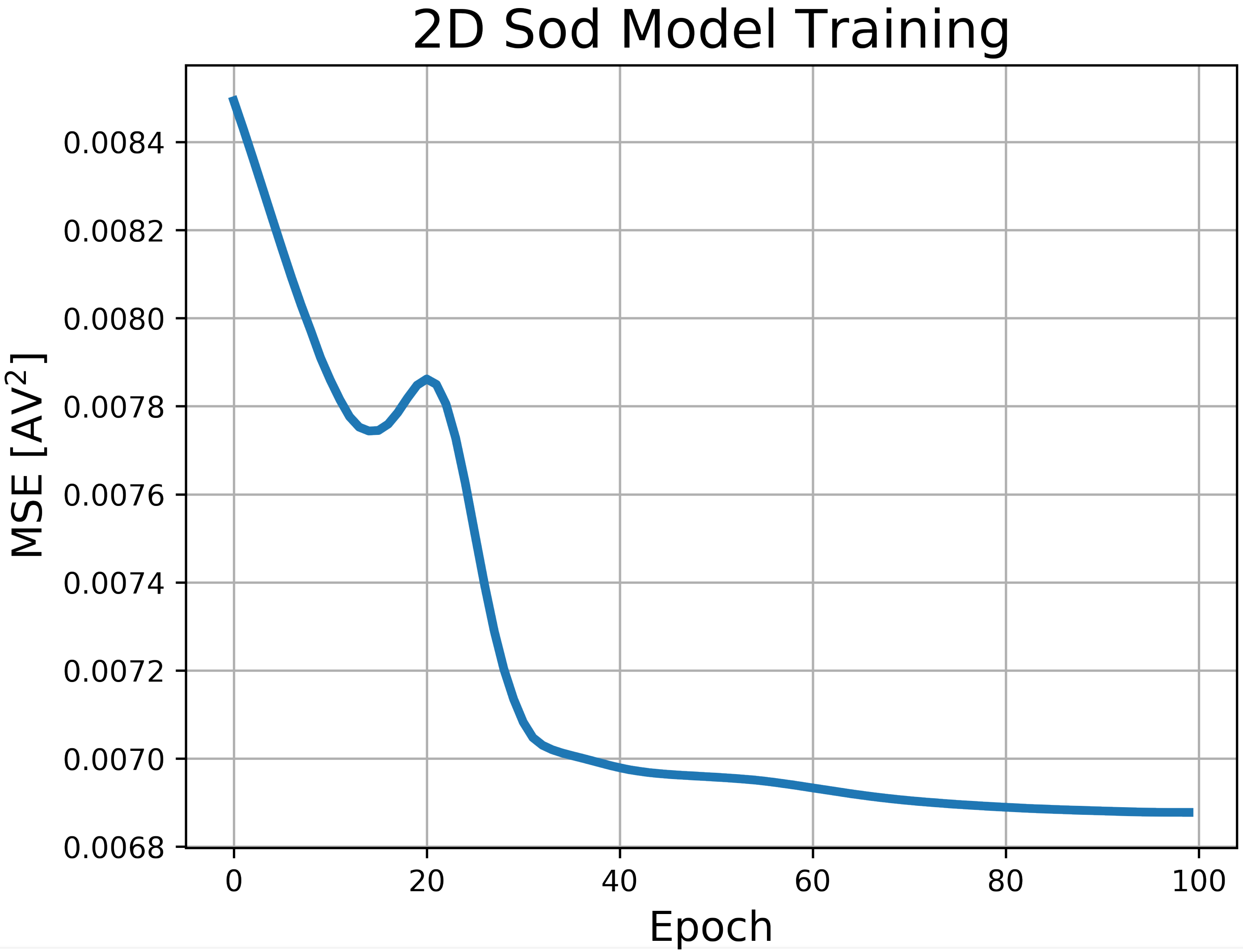


Relative Error in Density between AV Operator and NN-AV and Resolved Calculation* in the Triple Density Problem



*Resolved calculation was run using the traditional AV operator and 840000 spatial points (16x resolution)

Implementing a universal model based on the 2D Sod Shock Tube Problem



A model trained with a 2D shock dominated problem has similar accuracy to that trained with a 1D problem

L_2 Relative Error in Density between AV Operator and NN-AV and Resolved Calculation				
		AV	1D NN-AV	2D NN-AV
Problem	1D Sod	1.286e-03	1.345e-03	1.338e-03
	2D Sod	2.895e-04	2.901e-04	3.192e-04
	Sedov Blast Wave	8.540e-04	1.099e-03	9.812e-04
	Triple Density	1.600e-05	1.582e-05	1.824e-05

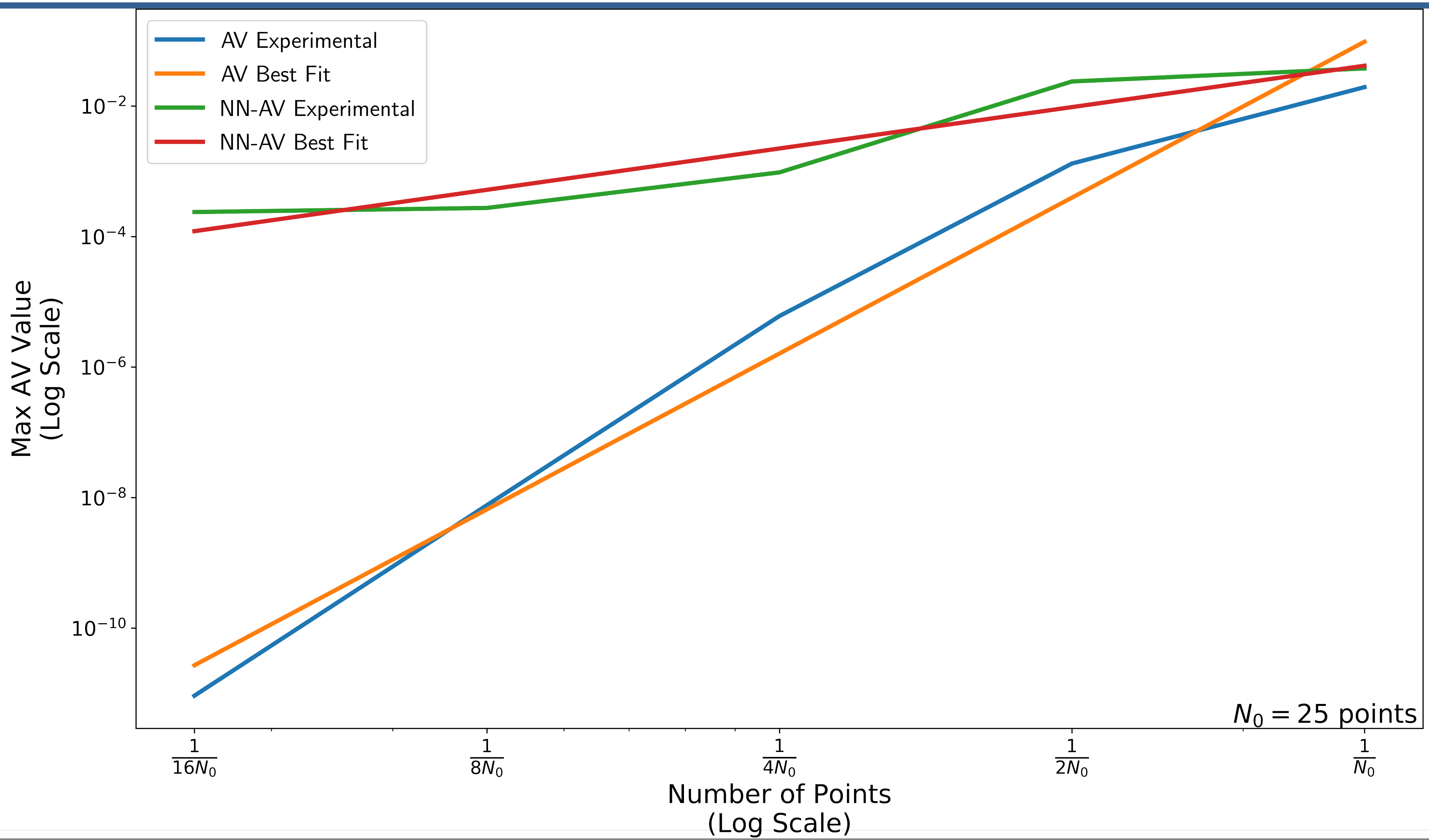
Order of accuracy of traditional AV is high-order by construction

- The NN-AV does not explicitly create a high-order accurate model
- To calculate order of accuracy

$$log(max(AV)) = log(C) + p * log(h)$$

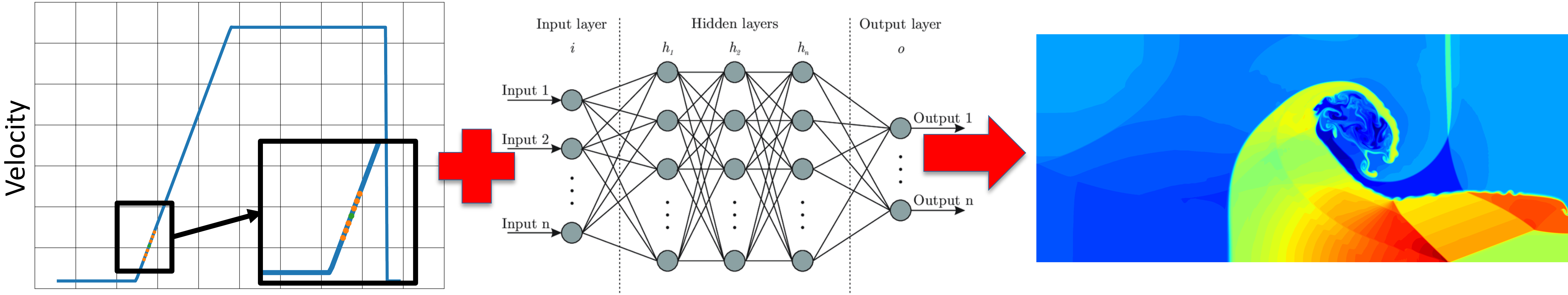
- h : number of points in the domain
- C : constant
- p : order of accuracy
- The max AV value was collected from different resolutions of Burgers' equation before the shock formed using both the traditional AV operator and NN-AV.

AV Operator is 8th order accurate
NN-AV is 2nd order accurate



Conclusion: AV can be modeled accurately using a Neural Network

- To a reasonable degree of accuracy, neural networks can accurately to predict artificial viscosity values in shock dominated problems
- By creating a model using one shock-dominated problem as a training dataset, the model can be used to predict artificial viscosity values in other shock-dominated problems



Future Work

- Decrease runtime
  - Optimize model prediction to decrease computation time needed
  - Optimize variable insertion to Miranda so each step doesn't require calculating AV
- Model improvement
  - Train a NN over mach numbers
  - Make a more universal model
  - Make the model high-order accurate
- Create NN for each artificial diffusivity, including:
  - Thermal conductivity
  - Vorticity
  - Shear viscosity

Lawrence Livermore National Laboratory