Surrogate Modeling of Fluids for Real-Time Control-Based Feedback

Sean Ruskowski[1], Jose Parra[1], Andrew Hembree[1], Wilson Kerr[1]

Arizona State University, Tempe, Arizona, 85281, United States of America

Direct numerical simulations and approximate fluid flow modeling are computationally cumbersome to simulate and analyze using traditional numerical solvers. Traditionally, the partial differential equations (PDEs) are solved via an iteratively learned mapping between the governing equations and their discretized Euclidean space. In this study, we explore the Fourier Neural Operator (FNO) framework as a surrogate modeling technique for the Navier-Stokes equations, where we aim to approximate the complex operator problem in higher dimensional space using the Fourier transform and conventional machine learning techniques. The FNO efficiently maps infinite-dimensional input-output spaces and spatial-independent phenomena with reduced computational overhead. This approach offers a cost-effective solution for modeling complex fluid dynamics problems. Our FNO architecture integrates a temporal pointwise convolution layer, Fourier layers, and spatial pointwise convolution layers to capture spatiotemporal phenomena on a local and global level. The FNO architecture was benchmarked on a problem involving a transient two-phase flow within a mixing chamber, where the FNO was tasked with predicting several field parameters: velocity, pressure, density, and temperature, with respect to time. Though the model currently results in a noisy prediction, it manages to semi-accurately predict transient properties while significantly reducing computational overhead.

1 Introduction

In recent years, advancements in deep learning have sparked interest in applying data-driven methods to computational science and engineering. This shift has given rise to scientific machine learning, a field focused on solving domain-specific data challenges by merging traditional numerical methods with the predictive capabilities, interpretability, and domain knowledge offered by surrogate models. Neural networks are particularly valued for their ability to approximate functions in high-dimension spaces, driving the development of new techniques to extract physical laws and domain specific trends from numerical and experimental data.

1.1 Fourier Neural Operator

The Fourier neural operator is a framework for learning relationships between infinite-dimensional spaces based on a finite set of observed input-output pairs. More specifically, let $\mathcal{A}$ and $\mathcal{U}$ be input-output observations in a normed high dimensional vector subspace, or *Banach space*, and $\mathcal{G}$ a model that maps $\mathcal{A}$ onto $\mathcal{U}$. The goal is approximate $\mathcal{G}$ using a neural network $\mathcal{G}_\Theta$, trained on data sampled from a probability distribution over $\mathcal{A}$. Here $\mathcal{G}_\Theta$ is a neural network approximation of the form

$$\mathcal{G}_\Theta \coloneqq \mathcal{P} \circ \sigma_T(\mathcal{K}_T) \circ \cdots \circ \sigma_1(\mathcal{K}_1) \circ \mathcal{Q}$$

where $\mathcal{Q}$ lifts the input to a higher-dimension representation, $\mathcal{P}$ projects it back to the target space, and σ_T an activation function acting component-wise on the output of some kernel operator $\mathcal{K}_T$.

Sample Fourier Neural Operator Architecture

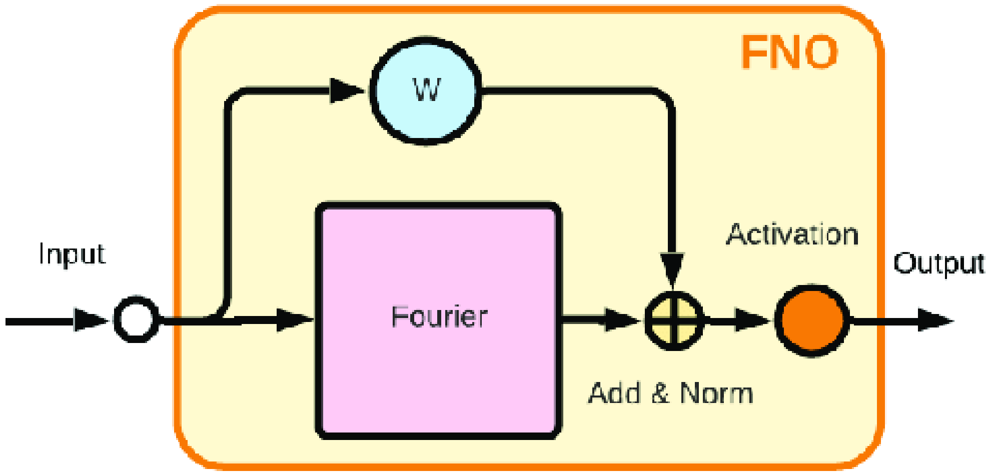

Figure 1. Fourier Neural Operator architecture. The input data is input into a Fourier layer, though the bias circumvents such layer. Within the Fourier layer, lifting operators are included. Once passing the Fourier layer, the data is normalized, passes the activation function, and projected to the output.

A key innovation of the FNO is its ability to parameterize the kernel operator into the Fourier domain. By decomposing the spatial mapping into its frequency components, the model can more efficiently capture far-field dependencies. Upon this transformation, learnable spectral weights are updated, and the kernel operator is returned to the spatial domain. By combining the Fourier spectral layer with traditional projecting and lifting operators and activation function, the FNO can leverage scientific machine learning techniques to efficiently model near-field and far-field flow interactions within a computationally efficient framework, and potentially provide significant generalized approximations independent of numerical discretization and resolution effects.

1.2 Navier Stokes Equations.

The Navier-Stokes equations are a set of partial differential equations, bounded by physical continuities, and are fundamental in fluid mechanics for their utility in simulating the macroscopic flow of fluid substances. This can be expressed as

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) = \mu\nabla^2\mathbf{u} - \frac{2}{3}\mu\nabla(\nabla\cdot\mathbf{u}) + \nabla(\xi(\nabla\cdot\mathbf{u}) - p) + \rho\boldsymbol{f}$$

where ρ is the density, t the time, $\mathbf{v}$ is the velocity vector, p the pressure, μ is the dynamic viscosity, and $\boldsymbol{f}$ containing the body forces.

1.3 Navier-Stokes Solution Operator

Implicitly, the NS equations describe a mapping from input spaces, such as initial and boundary conditions, forcing functions, and thermodynamic properties, to output spaces comprising density, pressure, and energy fields over time. Such mapping, known as the *Navier-Stokes solution operator* – denoted as $\mathcal{N}$, can be expressed as $\mathcal{N}: \mathcal{A} \to \mathcal{U}$ such that N_Θ can take the form

$$\mathcal{N}_\Theta := \mathcal{P} \circ \sigma_L(\mathcal{K}_L) \circ \cdots \circ \sigma_1(\mathcal{K}_1) \circ \mathcal{Q}$$

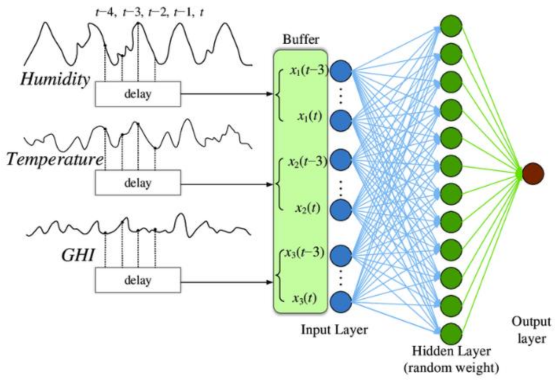

Figure 2. Navier Stokes solution operator architecture. Multiple spatial field inputs, including humidity, temperature, and GHI, with respect to the temporal quantity are input into the model. Within the model, the data goes through arbitrary lifting operators and activation functions, until being projected to the output space.

Through spectral representation, this data-driven approach can directly learn the trajectory of $\mathcal{N}$, enabling rapid predictions of the Navier-Stokes equations in domains that are traditionally cost prohibitive when solved within a conventional numerical manner. In this work, we explore the application of the Fourier Neural Operator framework for surrogate modeling in real-time plant systems. We aim to establish a theoretical framework in solving deterministic industrial problems by integrating data-driven surrogate models into real-time fluid dynamics applications.

2 Background

In this section we provide the necessary background for this paper. First we describe the Fourier Neural Operator and its formulation. We discuss our approach to the FNO and our formulation of our benchmark problem. Additionally, we discuss the mapping of the Navier-Stokes solution operator onto its surrogate via the Fourier Neural Operator.

2.1 Fourier Neural Operator

Scientific machine learning is a field concerned with the study of domain-specific data and interpreting, generalizing, and predicting unseen physical outputs. One such concept is the Fourier Neural Operator (FNO), which aims to approximate the solution of operator-based problems by constructing a surrogate model that maps input $u(x)$ to output $v(x)$. The FNO leverages the Fourier transform to represent functions in the frequency domain, decomposing $u(x)$ into its frequency components

$$\hat{u}(\xi) = F[u](\xi) \equiv \int_{\Omega} u(x) e^{-i2\pi\xi\cdot x}\, \mathrm{d}x \tag{1.1}$$

where ξ represents the frequency components and $\hat{u}(\xi)$ are the Fourier coefficients. This spectral representation allows $\mathcal{G}_\Theta$ to efficiently encode $\mathcal{G}$ on a global scale, thus ensuring the conservation of far-field and near-field interactions potentiated within a given solution operator problem. In the frequency-domain, the mapping is parameterized by learned filter $W_k(\xi)$ via

$$\hat{v}(\xi) = W_k(\xi) \cdot \hat{u}(\xi) \tag{1.2}$$

The filtered data is returned to the spatial domain using the inverse Fourier transform

$$v(x) = F^{-1}[\hat{v}](x) \equiv \int_{\Omega} \hat{v}\,(\xi) e^{i2\pi\xi\cdot x}\, \mathrm{d}\xi \tag{1.1}$$

Upon performing the inverse Fourier transform, a bias term $b_k(x)$ is added to the output, and an activation function σ is applied to output $v(x)$. Thus, a Fourier layer can be expressed as

$$v_{k+1}(x) = \sigma\left(F^{-1}[W_k(\xi) \cdot F[v_k](\xi)] + b_k(x)\right) \tag{1.3}$$

2.2 Benchmark Problem

To evaluate the performance of our FNO framework, we conducted a transient fluid simulation of laminar, immiscible water and oil within a mixing chamber. The flow field was resolved using the Navier-Stokes equations, while the transport of the fluid primitives was modeled using a Volume of Fluid (VOF) approach. Periodic boundary conditions were applied at the inlets, and the outlet was defined using a continuity-enforcing constraint. Rotating disks were incorporated within the chamber to promote mixing, with their boundaries handled using the immersed boundary method. This setup enabled a comprehensive assessment of the framework's ability to capture complex transient fluid phenomena.

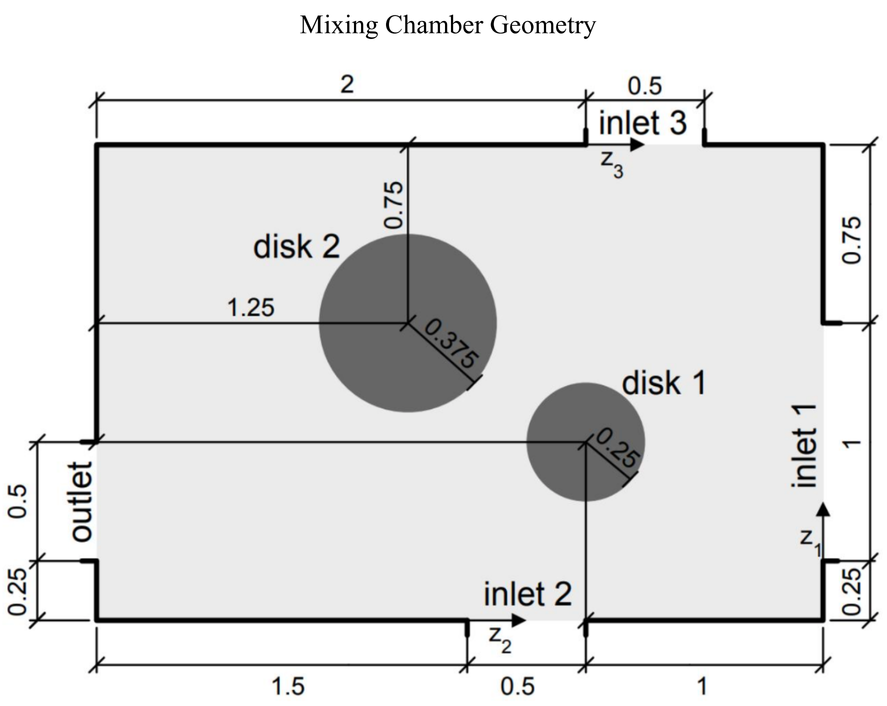

Fig. 3 Mixing Chamber Geometry. The inlet, outlet, and disk boundary conditions are shown, as well as the overall geometry of the benchmark problem. It can be implied that the disks, inlets, and outlets, as well as multiple fluid species will generate complex fluid phenomena, serving to provide a stress test foundation for our Fourier Neural Operator architecture.

2.3 Navier-Stokes Equations

The simulation of fluid flow is grounded in the Navier-Stokes (NS) equations, a set of conservation laws – principally, the laws of mass, momentum, and energy – applied over a defined spatial region, or *control volume* (CV). In this approach, fluxes of extensive properties are evaluated, ensuring that they satisfy conservation principles within the CV. This yields a generalized form of the continuity equation for any conserved quantity, denoted as

$$\frac{\partial \phi}{\partial t} + \nabla \cdot (\phi \mathbf{u}) = S(\phi) \tag{1.4}$$

For mass conservation, $\phi = \rho$ and the source/sink term $S(\phi) = 0$, thus

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \tag{1.5}$$

For momentum conservation, $\phi = \rho \boldsymbol{v}$, and the source term $S(\phi) = \sum \boldsymbol{f}$ arise due to surface effects (pressure and viscous stresses) and body forces (e.g., gravity). The momentum conservation is given by

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot (\rho \mathbf{u}\mathbf{u}) = -\nabla p + \nabla \cdot \boldsymbol{\tau} \tag{1.6}$$

Here, τ represents the viscous stress force. Assuming stress can be linearly related between velocity gradients, isotropic terms are fully isotropic and deviatoric terms are fully anisotropic, the viscous stress tensor can be represented as

$$\tau = \mu(\nabla \mathbf{u} + (\nabla \mathbf{u})^T) + \left(\xi - \frac{2}{3}\right)(\nabla \cdot \mathbf{u})\mathbf{I} \tag{1.7}$$

with μ being the dynamic viscosity and ξ being the bulk viscosity. Coupling the divergence operator in Eq. (1.6) into Eq. (1.5) and converting the Navier-Stokes equation into its incompressible form results in

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\frac{1}{\rho}\nabla p + \frac{1}{Re}\nabla^2 \mathbf{u} \tag{1.8}$$

$$\nabla \cdot \mathbf{u} = 0 \tag{1.9}$$

where Re is the dimensionless ratio between inertial and viscous forces, and Eq. (1.9) is the divergence free constraint required to enforce incompressibility.

2.4 Implicit Navier Stokes Solve

We begin by solving the incompressible Navier-Stokes equation in a conventional numerical framework. An operator-splitting approach was employed to decouple the governing equations into their constituent physical processes, splitting the equations into convective, diffusive, and stress terms. The Volume of Fluid method was employed to track the transport of the immiscible fluid phases in the mixing chamber, using a volume fraction field, Y, to indicate the fraction of each cell occupied by the respective fluid species.

To enforce the incompressibility constraint, the stress terms were decomposed into their viscous and inviscid components. These terms were subsequently coupled through Helmholtz – Hodge decomposition, which iteratively projects the velocity field onto a divergence-free space. This decomposition ensures the final coupled pressure-velocity field satisfies the incompressibility constraint.

The convective terms were discretized in space using the fifth-order Weighted Essentially Non-Oscillatory (WENO-5) scheme. The WENO-5 scheme adaptively adjusts node weights based on local smoothness parameters, preserving high-order accuracy in smooth regions while minimizing numerical dissipation in areas with discontinuities. The diffusive terms were solved using the Crank-Nicolson scheme, where the diffusive terms were discretized in space using the second-order trapezoidal rule.

The inviscid and diffusive terms were spatially discretized using second-order central differencing, while convective terms were discretized using fifth-order upwind differencing. For the volume fraction field, the inviscid, diffusive, and convective terms were integrated in time using the third-order Total Variation Diminishing Runge-Kutta (TVD-RK3) scheme. This scheme begins with an initial prediction step based on the third-order Runge-Kutta method, followed by two successive correction steps to enforce the Total Variation Diminishing property, ensuring stability and suppressing numerical dissipation. For the primitive fields, the inviscid, diffusive, and convective terms were integrated in time using the Adams-Bashforth method. This method utilizes the second-order Euler's method to advance the solution while maintaining temporal accuracy.

The immersed boundary method was employed to model the rotating disks inside the mixing chamber. The boundaries of the rotating disks were represented through marker points that described the disks portion and motion. These marker points were used to enforce the no-slip condition by correcting the velocity field to match the rotational motion of the disks. This approach allowed for the inclusion of complex moving boundaries within the grid while preserving the equidistant cartesian grid required for the FNO.

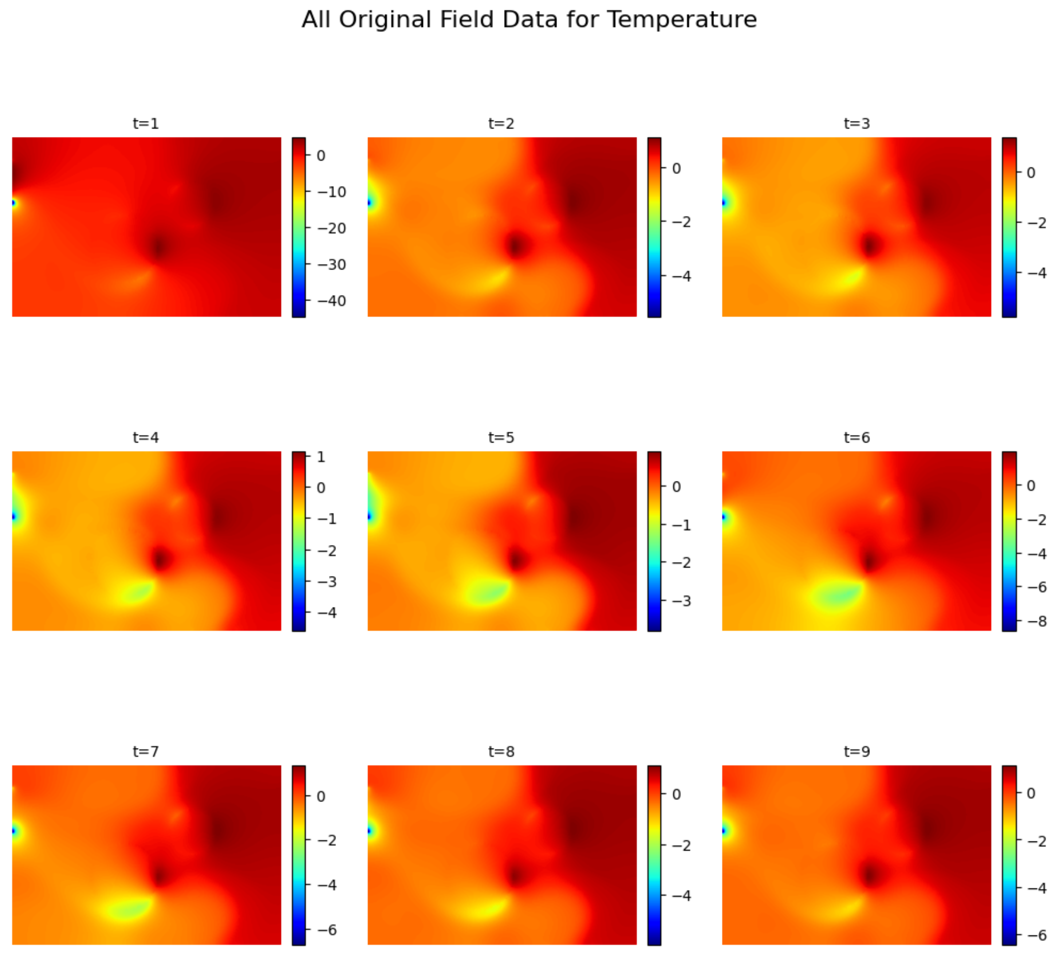

Figure 4. All original spatiotemporal data for temperature. We show the normalized temperature distribution calculated via the Navier-Stokes equations and appropriate equations of state. Each subplot represents a different time step, and the color represents temperature magnitude.

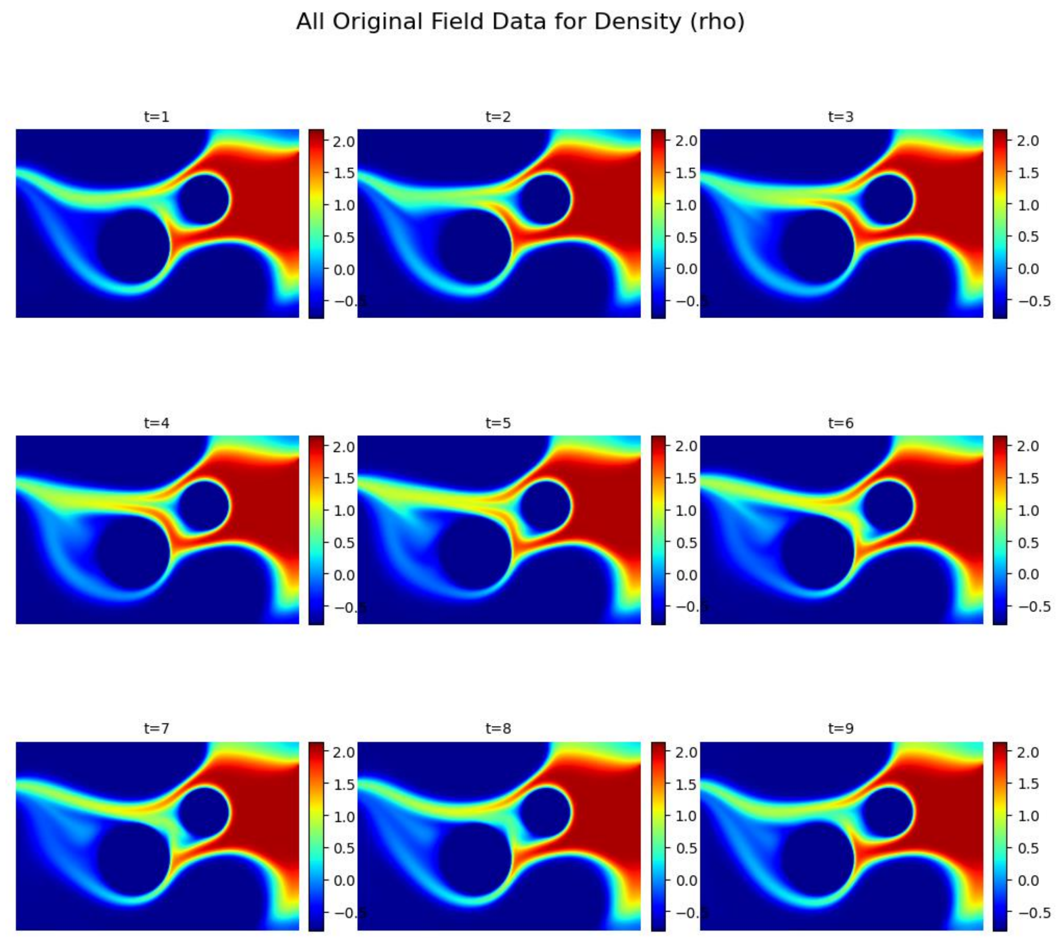

Figure 5. All original spatiotemporal data for density. We show the normalized density distribution calculated via the Navier-Stokes equations. Each subplot represents a different time step, and the color represents temperature magnitude.

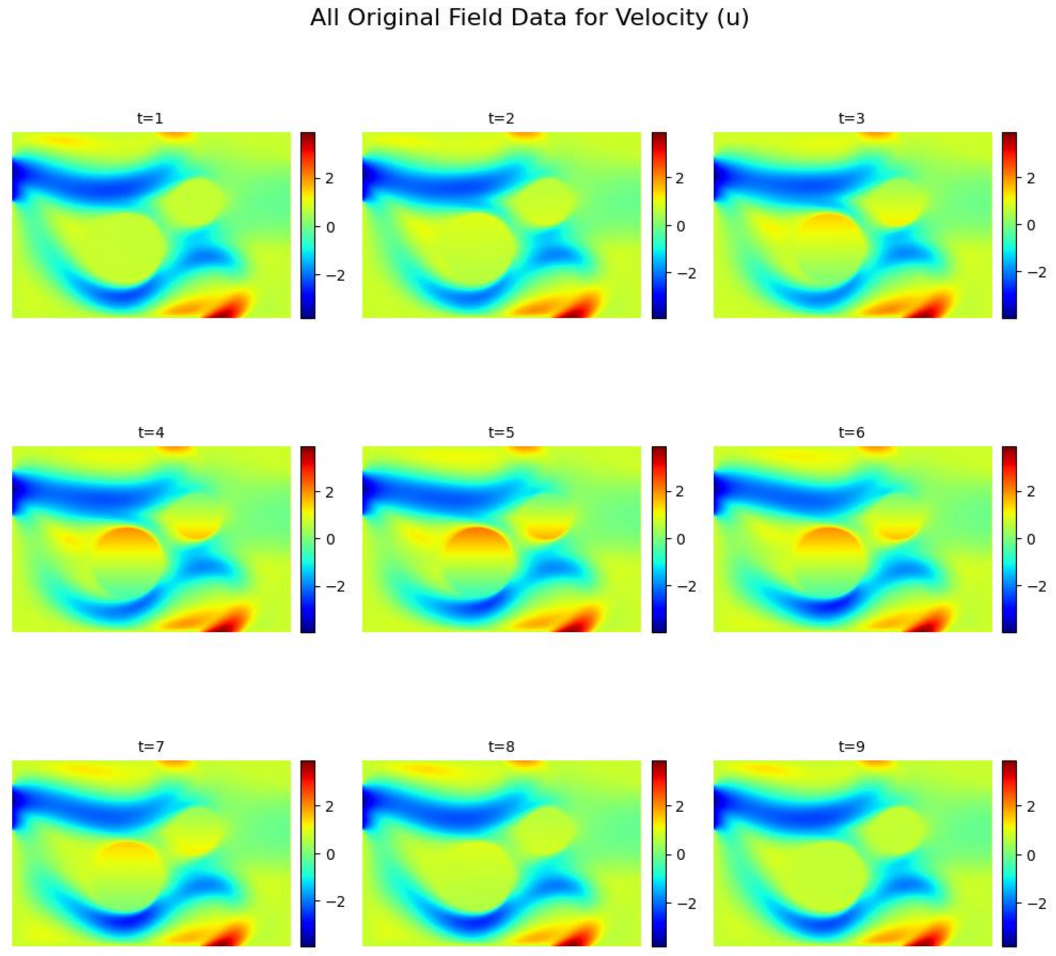

Figure 6. All spatiotemporal data for the velocity in the x direction. Figure 6 represents velocity profile projected from the pressure, that being oil and water. This is the original data solved by the Navier-Stokes and with numerical methods. Each subplot represents a velocity profile for the fluid flow at different time steps for the u direction.

3 Model Architecture

The proposed model is built upon the Fourier Neural Operator framework which enables efficient learning of fluid dynamics problems through spatiotemporal mappings. The model architecture begins by processing an input tensor: [B, T, C, H, W], where B is the batch size, T is the temporal dimension, C is the channel containing the number of field quantities, H is the spatial height dimension, and W is the spatial width dimension. Input channel C contains five quantities: x-direction velocity $\mathbf{u}$, y-direction velocity $\mathbf{v}$, density, pressure, and temperature, with each quantity stored in a 2D array indexed with respect to the field's spatial dimension. The temporal dimension T is projected onto these fields, with the third dimension respecting the field data at that respective temporal quantity.

Temporal Convolution Layer

Let $T \rightarrow C$ represent the field variable quantities at its respective temporal step. The spatiotemporal information $T \rightarrow C$ is processed using a sliding window approach, where an input $T_{window} \rightarrow C$ is used to forecast an output of $T_{horizon} \to C$. The window input is passed through a pointwise temporal convolutional layer, which applies a 1D convolution along the temporal axis. This operation pointwise convolves the input temporal sequence $T_{window} \to C$ into a forecasted sequence $T_{horizon} \to C$, capturing temporal dependencies while preserving the spatial structure. The sliding window then shifts $T \to C$ forward by an index and the process is repeated iteratively until the forecasted temporal range spans the entire sequence T_{max}. The resulting tensor is reshaped and passed through a fully connected layer.

Fourier Neural Operator

The spatial features are refined through four series of FNO blocks, each composed of a spectral layer, a convolution layer, an activation function, and a fully connected layer. The spectral layer applies a 2D Fourier Transform to project the spatial data into the frequency domain, where the model learns a set of spectral weights. These weights are controlled by the hyperparameter modes, which suppress local features by filtering out high frequency modes. The modified frequency representation is then transformed back into the spatial domain using the inverse Fourier transform. The output from the spectral layer is passed to a convolution layer, which applies a pointwise convolution across the spatial domain. This operation captures localized patterns and low-frequency effects that are not explicitly addressed within the spectral layer. The unified representation is then passed through a Gaussian Error Linear Unit (GELU) activation function such as to enhance the model's capacity to learn complex spatiotemporal relations input into the model. The processed features are then input into a fully connected layer which connects to another FNO block: or in the case of the final FNO block, the output layer.

4 Training

Given $\mathcal{N}_\Theta \coloneqq \mathcal{P} \circ \sigma_L(\mathcal{K}_L) \circ \cdots \circ \sigma_1(\mathcal{K}_1) \circ \mathcal{Q}$, the goal with training is to find a $\mathcal{N}_\Theta$ that maps $\mathcal{N}_\Theta$ onto $\mathcal{N}$. This is typically done by taking the norm of output $\mathcal{N}_\Theta$ with respect to input $\mathcal{N}$ and differentiating this norm with respect to Θ. This differential serves as a high-dimensional convex line search for an optimizer to transverse, where the model is fully fit once this gradient reaches a minimum. To calculate this differential and norm, backpropagation and the mean square error (MSE), respectively, were used.

4.1 Training Data and Batching

The training data was generated via our benchmark problem, with density, velocity, pressure, and temperature fields stored with respect to space and time. The fields were stored in a 386 by 258 arrays where each index of the array represented cartesian location, equidistant from the corresponding index array position. Nine times cales were stored, corresponding to the simulation times 1 – 9, with 1 second increments. Given the scale of the data, the data was fit into a single batch for training.

4.2 Optimizer

We trained our model using the AdamW optimizer. The AdamW optimizer is an extension of the classical Stochastic Gradient Descent (SGD) optimizer which incorporates the moving mean, $\hat{\boldsymbol{m}}_t$, and variance, $\hat{\boldsymbol{v}}_t$, of the gradient for line search normalization, and decoupled weight decay, λ, normalization. The AdamW algorithm is given by:

Algorithm 1 The AdamW method algorithm

```
1: given $\beta_1, \beta_2, \epsilon, \lambda$
2: initialize $t = 0, \Theta_t = \mathcal{W}, m_t = 0, v_t = 0$
3: repeat
4:     $t = t + 1$
5:     $\nabla \mathcal{N}_\Theta(\Theta_{t-1})$
6:     $\boldsymbol{g}_t = \nabla \mathcal{N}_\Theta(t-1) + \lambda \Theta_{t-1}$
7:     $\boldsymbol{m}_t = \beta_1 \boldsymbol{m}_{t-1} + (1-\beta_1)\boldsymbol{g}_t$
8:     $\boldsymbol{v}_t = \beta_2 \boldsymbol{v}_{t-1} + (1-\beta_2)\boldsymbol{g}_t^2$
```

```
9:     $\hat{\boldsymbol{m}}_t = \boldsymbol{m}_t/(1-\beta_1^t)$
10:    $\hat{\boldsymbol{v}}_t = \boldsymbol{v}_t/(1-\beta_2^t)$
11:    $\eta_t = SetSchedule(t)$
12:    $\Theta_t = \Theta_{t-1} - \eta_t\big(\hat{\boldsymbol{m}}_t/(\sqrt{\boldsymbol{v}_t} + \epsilon) + \lambda\Theta_{t-1}\big)$
13: until stopping criterion is met
14: return $\Theta_t$
```

4.3 Hyperparameters

The hyperparameters were $\beta = [0.9,0.95]$, $\epsilon = 1e-8$, and $\lambda = 0.01$. Additionally, sixteen spectral modes were used within our spectral filters. For the learning rate, cosine annealing with warmup was used, where the initial warmup learning rate was $\eta_t = 4e-5$, the max learning rate was $\eta_t = 4e-6$, and the minimum learning rate was $\eta_t = 1e-5$. One-hundred warmup epochs were used, and five-hundred total epochs were used in training. The learning rate scheduler can be represented via:

$$\eta_t = \eta_{init} + (\eta_{max} - \eta_{init})\,\frac{t}{t_{warmup}} \tag{1.10}$$

$$\eta_t = \eta_{min} + \frac{1}{2}(\eta_{max} - \eta_{min})\left[1 + \cos\left(\pi\,\frac{t - t_{warmup}}{T - t_{warmup}}\right)\right] \tag{1.11}$$

To initialize the weights and biases within the network, uniform Xavier Initialization was used, ensuring that the variance of activations remain approximately the same across all layers of the network. This can be represented via

$$\mathcal{W} \sim \mathcal{U}\left(-\frac{\sqrt{6}}{\sqrt{\mathrm{n_{in} + n_o ut}}}, \frac{\sqrt{6}}{\sqrt{\mathrm{n_{in} + n_o ut}}}\right) \tag{1.12}$$

where n_{in} are the input neurons and n_{out} are the output neurons. To prevent destabilization from large updates, element-wise gradient clipping was applied, with large gradients clipped to a value of 1.0.

5 Results

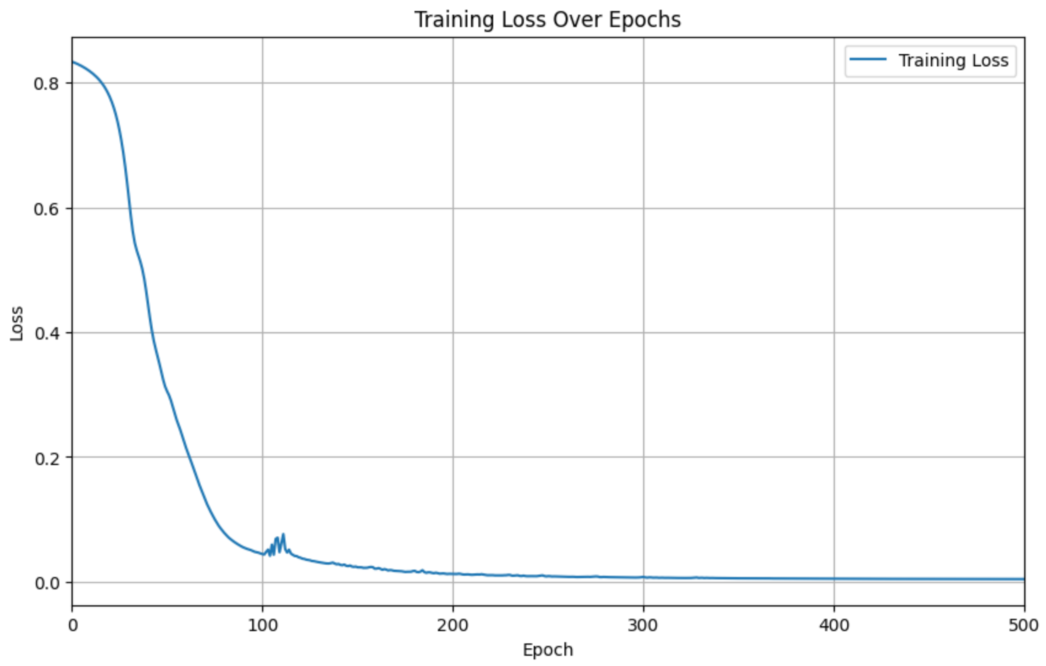

Figure 7. Training Loss Model using AdamW. This is the Loss vs epochs using a warmup scheduler then a cosine annealing learning rate.

Figure 7 shows the model's loss with respect to epoch count. Initially, the loss is high, and slowly declines. As the learning rate warms up, the loss declines more steeply. The loss slightly reverses when the optimizer initially switches to the cosine annealing learning rate. This is likely because of the dramatic switch in learning rates, with the learning rate going from a steep increase to a gradual decline. Extending beyond this, the loss gradually decreases until the max epoch count, reaching a near zero loss.

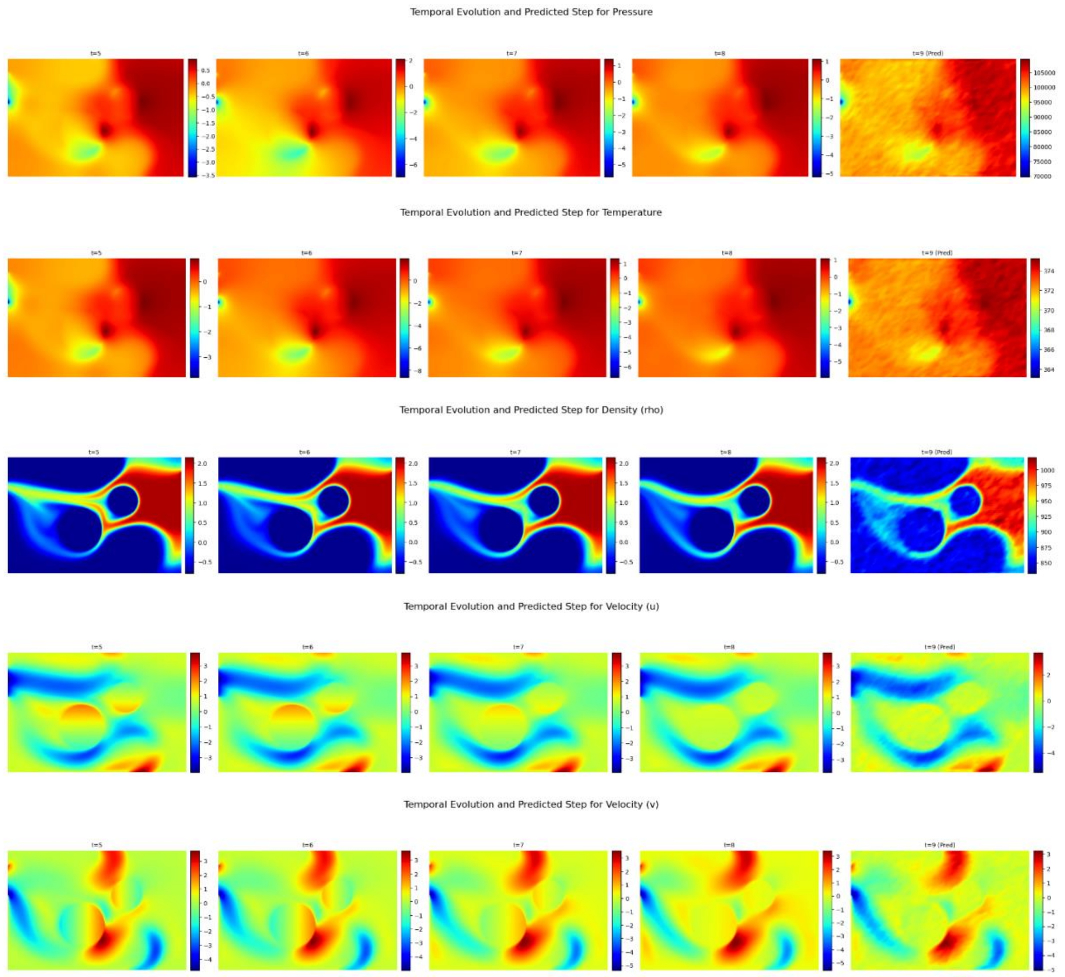

Figure 8. Temporal and predicted evolution of field quantities. The field variables within the first four columns of the subplots are the numerically solved quantities. The fifth column contains the quantities predicted via the Navier-Stokes operator.

The results of the Fourier Neural Operator (FNO) model predictions for our fluid flow within the mixing chamber show the model's ability to capture large-scale dynamics of physical parameters, such as: pressure, temperature, density, and velocity components. The square in each visualization row shows the flows properties true state for the first four columns while the last column shows the model's predicted state. The last column shows a resemblance when compared to the true models. The FNO captures global interactions and long-range dependencies effectively, though it may undermine localized and lower frequency modes in the model. Another reason for this blurriness could be due to the model potentially being overfitted. Overall, the FNO model successfully reproduces flow behaviors, pressure fields, thermal diffusion, and realistic density and velocity distributions. This shows that the model can be used if optimized. The results validate the effectiveness of our FNO in handling high-dimension operators, with manageable computation requirements.

6 Conclusion

In this study, we have been able to show the potential of the Fourier Neural Operator framework for surrogate modeling in real-time fluid dynamics applications. With the efficiency and scalability of the FNO, complex transient fluid behaviors can be significantly improved with less computational overhead. The benchmark problem involving a two-phase flow in a mixing chamber has successfully tested the capability of the framework to approximate the Navier-Stokes solution operator both accurately and efficiently.

Our results suggest that the integration of data-driven surrogate models with physics-based methodologies provides an promising avenue to solve challenges related to computational efforts within fluid dynamics. The training process, optimized by advanced techniques like AdamW and a warmup cosine learning rate schedule made sure the stability and convergence of the model were maintained. Success in the framework's use for predicting important fluid properties, such as velocity, pressure, density, and temperature, ensures versatility and applicability in diverse industrial and scientific contexts.

Future work will focus on extending this framework toward more complex geometries, multi-physics scenarios, and higher Reynold's numbers in order to further push the envelope. Additionally, it could be further improved by including uncertainty quantification and adaptive learning techniques in order to make the model even more robust and reliable for real-time feedback systems

7 Acknowledgments

We acknowledge the foundational work of Zongyi Li, Nikola Kovachki, Kamyar, Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar for their introduction to the Fourier Neural Operator.

8 References

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention Is All You Need," 31st Conference on Neural Information Processing Systems (NeurIPS 2017), Long Beach, CA, USA, 2017. Available: https://arxiv.org/abs/1706.03762

J. H. Ferziger and M. Peric, *Computational Methods for Fluid Dynamics*, 3rd ed., Springer, Berlin, 2002.

P. Moin, *Fundamentals of Engineering Numerical Analysis*, 2nd ed., Cambridge University Press, Cambridge, U.K., 2010.

9 Appendix

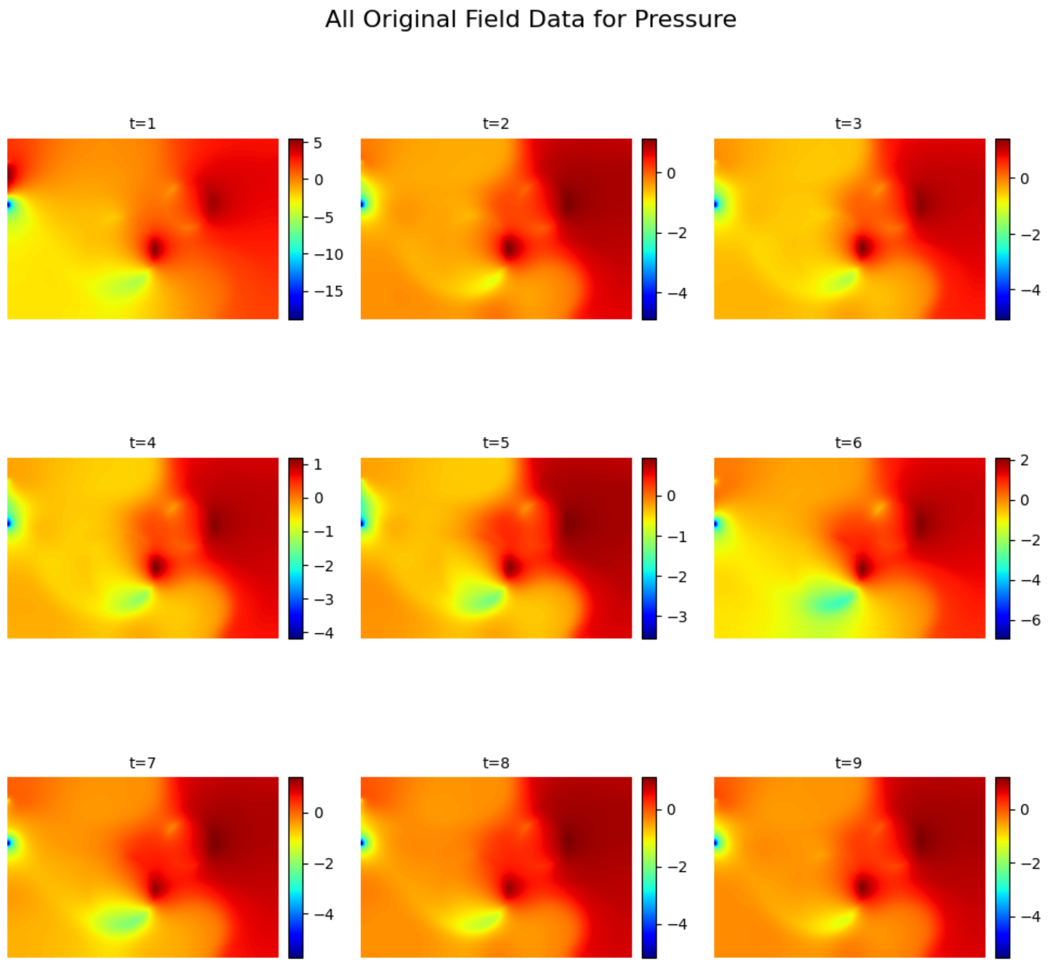

Fig 9. All original spatiotemporal data for temperature. We show the normalized pressure distribution calculated via the Navier-Stokes equations and appropriate equations of state. Each subplot represents a different time step, and the color represents pressures magnitude.

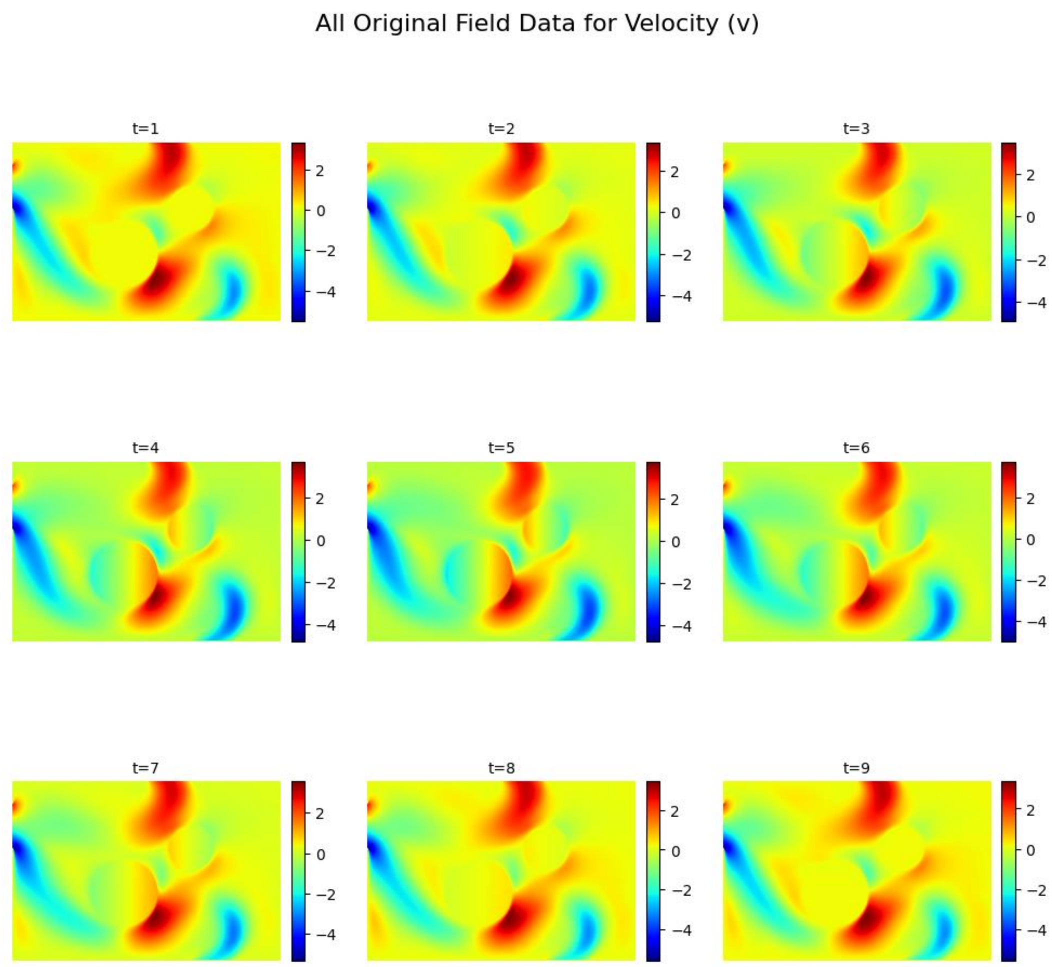

Figure 10. All spatiotemporal data for the velocity in the x direction. Figure 6 represents velocity profile projected from the pressure, that being oil and water. This is the original data solve by the Navier-Stokes and with numerical methods. Each subplot represents a velocity profile for the fluid flow at different time steps for the v direction.