공학석사 학위논문

A Robust and Accurate Reconstruction Method based on Machine Learning for High-Speed Flow Simulation on Unstructured Meshes

비정렬 격자상 초음속 유동 해석을 위한 머신러닝 기반 강건하고 정확한 재구성 기법

2023 년 8 월

서울대학교 대학원
항공우주공학과
주 자 연

A Robust and Accurate Reconstruction Method based on Machine Learning for High-Speed Flow Simulation on Unstructured Meshes

비정렬 격자상 초음속 유동 해석을 위한 머신러닝 기반 강건하고 정확한 재구성 기법

지도교수 김 종 암

이 논문을 공학석사 학위논문으로 제출함

2023 년 8 월

서울대학교 대학원

항공우주공학과

주자연

주자연의 공학석사 학위논문을 인준함

2023 년 8 월

위 원 장	김 규 홍	(인)
부위원장	김 종 암	(인)
위 원	강 명 주	(인)

Abstract

A Robust and Accurate Reconstruction Method based on Machine Learning for High-Speed Flow Simulation on Unstructured Meshes

Joo Jayeon
Aerospace Engineering
The Graduate School
Seoul National University

Developing a robust and accurate shock-capturing method in computational fluid dynamics (CFD) has long been a challenging task, despite extensive research efforts based on mathematical analysis. As a breakthrough, this study proposes a new reconstruction method using a data-driven approach to achieve high levels of robustness and accuracy in multi-dimensional compressible flows based on the finite volume method. The proposed method divides the computational domain into discontinuous and smooth regions using a tree model. Subsequently, fully connected neural network (FCNN) models are trained specifically for each region, allowing for high robustness in capturing shocks in the discontinuous region and high accuracy in

modeling the smooth region.

To train the models in this study, four types of datasets were constructed: one representing discontinuous flows and the other representing smooth flows for quadrilateral and triangular element types. These datasets incorporated a variety of analytic functions, ensuring comprehensive coverage of different flow scenarios. Additionally, suitable input features were defined to enable efficient extension to unstructured meshes, enhancing the method's applicability.

Extensive numerical tests were conducted to validate the robustness and accuracy of the proposed method. This study highlights the potential of data-driven methods in improving the accuracy and robustness of complex flow simulations and presents a promising approach for developing more effective shock-capturing methods in CFD.

Keyword: Computational Fluid Dynamics (CFD), Finite Volume Method (FVM), Machine-Learning (ML), Fully Connected Neural Network (FCNN), Tree Model, Reconstruction Method

Student Number: 2021-22136

Table of Contents

List of Figures

List of Tables

Chapter 1 Introduction

1.1 Research Background

In engineering applications, the finite volume method (FVM) [1], which can be applied regardless of grid topologies, is widely used as a standard. Also, FVM can obtain high-order accuracy by using high-order polynomials in the reconstruction process [2]. However, numerical oscillations can occur in the shock wave region with abrupt flow variable changes when solving supersonic and hypersonic flows with second-order or higher accuracy. These oscillations cause numerical instability. In addition, Godunov's theorem demonstrates that linear and monotonic numerical schemes cannot exceed first-order accuracy [3].

Many shock-capturing methods have been developed based on different stability conditions. These methods aim to eliminate the oscillations while maintaining the desired accuracy.

One such condition is the Total Variation Diminishing (TVD) condition [4], which ensures that the total variation of the solution does not increase over time. However, this approach inevitably results in first-order accuracy in regions of smooth extrema, known as the clipping phenomenon. Another approach is the Total Variation Bounded (TVB) condition [5], which guarantees that the total variation of the solution remains bounded. Representative TVB-based reconstruction methods are the Essentially Non-Oscillatory (ENO) [6] and Weighted Essentially Non-Oscillatory (WENO) [7] methods. However, they require tunable parameters to define the boundness of the total variation. Additionally, since TVD and TVB were developed through one-dimensional analysis, extending the scheme to multi-dimensional problems requires a dimensional splitting method. It still leads to limitations in reconstructing high-order polynomials in irregular meshes. Therefore, in multi-dimensional problems, the maximum principle has been introduced to ensure the positivity and boundedness of the solution. Representative maximum principle-based shock-capturing methods are the Barth and Jespersen limiter [8], Venkatakrishnan limiter [9], and Multi-dimensional Limiting Process (MLP) limiter [10]. The MLP limiter has greatly improved robustness, accuracy, and convergence in high-speed flow by appropriately considering multi-dimensional effects overlooked in developing conventional high-accuracy numerical methods.

However, the methods mentioned above have been developed based on mathematical analysis. Recently, there has been a growing interest in developing shock-capturing methods using data-driven approaches, which offer several advantages. First, these approaches can eliminate user-defined parameters, making the method more objective. Second, they do not need to rely on a single explicit condition, allowing them to combine the advantages of various schemes by training datasets based on multiple conditions. Furthermore, data-driven approaches can be trained using datasets based on analytic functions or exact solutions, thereby improving the accuracy of the solutions.

Examples of data-driven shock-capturing methods are followed. At first, Ray and Hesthaven developed a troubled-cell indicator by training an artificial neural network (ANN) with shock wave data generated using an analytical function. They verified the performance by applying the indicator to the high-order discrete Galerkin method in one/two-dimensional benchmark problems [11],[12]. Yu et al. used ANN to develop a smoothness indicator and combined it with an artificial viscosity technique [13]. Beck et al. treated the solution distribution of each sub-cell as a single image. They applied edge detection techniques to classify cells in the shock region (shock detection) and localize the shock wave location in sub-cells (shock localization). They combined the method with the artificial viscosity technique and verified it through one/two-dimensional benchmark problems [14], [15]. Feng et al. trained an ANN using rigorous solution data of the one-dimensional Burgers equation and extended it to multi-dimensional shock wave flow through dimensional splitting [16], [17]. Additionally, Discacciati et al. developed a universal artificial viscosity model that learns several artificial viscosity methods suitable for flow situations, eliminating the need for parameter adjustment [18].

However, it is important to note that many of these previous studies have primarily focused on using high-order methods rather than the FVM. Additionally, most of these studies have only partially replaced certain aspects of the shock-capturing methods with machine learning techniques.

1.2 Research Objective

The objective of this study is to introduce a novel data-driven reconstruction method based on the FVM. The primary goal is to achieve a data-driven approach that maintains high robustness in discontinuous regions while enhancing accuracy in smooth regions, all using only the MLP stencil.

To achieve this objective, the study incorporates a tree model to effectively distinguish between discontinuous and smooth cells within the computational domain. In addition to the tree model, two fully connected neural network (FCNN) models are employed to compute the cell interface values for each cell. These FCNN models play a crucial role in accurately and robustly reconstructing cell interface values, ensuring high-quality simulations in both discontinuous and smooth regions.

To train these models, the method constructs a discontinuous dataset by using limited values at the cell interface, thereby ensuring robustness in discontinuous regions. Conversely, a smooth dataset is generated by using exact values at the cell interface with a compact stencil, aiming to enhance accuracy in smooth regions. Furthermore, the proposed data-driven reconstruction method is extended to accommodate irregular mixed meshes by defining appropriate input features, allowing the method to handle more complex and diverse flow scenarios.

Finally, through a comprehensive set of numerical tests and application problems, the study aims to rigorously demonstrate the proposed data-driven reconstruction method's robustness and accuracy in handling different flow scenarios. By examining the method's performance in various challenging scenarios, including discontinuous and smooth flow fields, the study seeks to validate the method's potential as an effective and versatile approach for fluid simulations.

Chapter 2 Finite Volume Method

In this section, an overview of finite volume method (FVM) on unstructured meshes are presented.

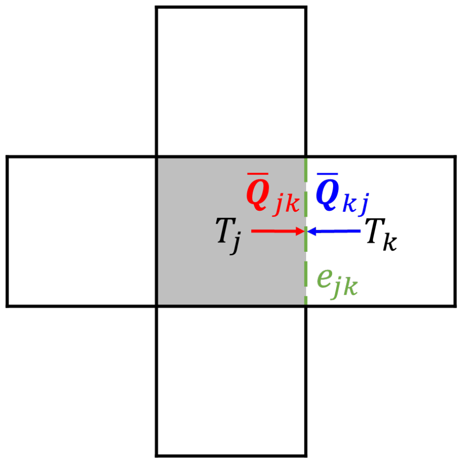


Figure 2.1: Quadrilateral cell and its neighborhood

Consider the multi-dimensional hyperbolic conservation laws given by the partial differential equation:

$$\frac{\partial \mathbf{Q}}{\partial t} + \nabla \cdot \mathbf{F} = 0, \qquad \text{Eq. 2.1}$$

where $\mathbf{Q}$ is the state variable vector and $\mathbf{F}$ is the flux function vector. In FVM, Equation, Eq. 2.1 is integrated over the control volume T_j, resulting in the following Eq. 2.2.

$$\int_{T_j} \frac{\partial \mathbf{Q}}{\partial t} + \int_{\partial T_j} \mathbf{F} \cdot \mathbf{n} dS = 0. \qquad \text{Eq. 2.2}$$

Here, $\mathbf{n}$ represents the outward normal vector. Depending on the location of physical variables, FVM allows for the use of either a cell-centered or cell-vertex approach. In this study, a cell-centered approach is adopted. After numerically approximating the flux function, the semi-discrete form of Eq. 2.3 is obtained:

$$|T_j|\frac{\partial\bar{\mathbf{Q}}_j}{\partial t}+\sum_{e_{jk}\in T_j}\mathbf{H}(\bar{\mathbf{Q}}_{jk},\bar{\mathbf{Q}}_{kj})\,|e_{jk}|. \qquad \text{Eq. 2.3}$$

In this equation, $\bar{\mathbf{Q}}_j$ represents the cell-averaged state vector, while $\bar{\mathbf{Q}}_{jk}$ represents the cell interface state vector in the direction from cell T_j to T_k. $|T_j|$ denotes the area of cell T_j, and e_{jk} represents the edge between T_j and T_k with a length $|e_{jk}|$. The vector $\mathbf{H}(\bar{\mathbf{Q}}_{jk},\bar{\mathbf{Q}}_{kj})$ represents the numerical flux function, and the midpoint rule is applied to calculate the numerical flux.

To achieve high-order accuracy, various reconstruction methods have been developed to compute the cell interface values. One such method, known as the multi-dimensional limiting process (MLP), will be discussed in more detail in Ch. 3. Additionally, in this study, an improved reconstruction method using machine learning will be introduced in Ch. 4.

Chapter 3 Multi-dimensional Limiting Process

The multi-dimensional limiting process (MLP) limiter is a robust and accurate method that incorporates both cell-centered and cell-vertex points, making it effective for non-grid-aligned flow distributions. In this study, the MLP limiter is utilized to generate a discontinuous dataset, ensuring strong robustness. This section provides a concise explanation of the MLP limiter and its implementation steps.

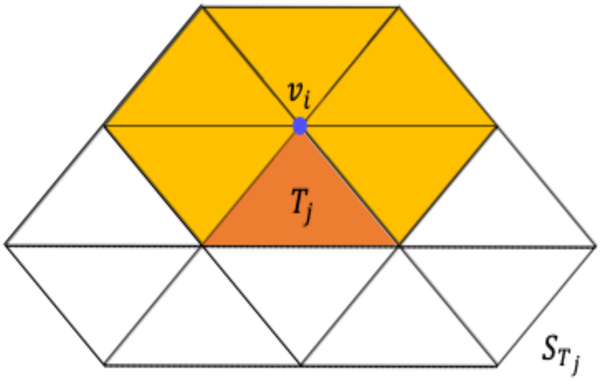


Figure 3.1: MLP stencil of the cell T_j (shaded region: neighboring cells of the vertex v_i)

The condition satisfied by the MLP limiter is expressed by Eq. 3.1, which ensures that the estimated value of the vertex point $(\hat{q}_{v_i})$ lies within the range of the averaged values of neighboring cells.

$$\bar{q}_{neighbor}^{\min} \leq \hat{q}_{v_i} \leq \bar{q}_{neighbor}^{\max}, \ \forall v_i \in \Theta_j, \qquad \text{Eq. 3.1}$$

In this equation, Θ_j represents the set of vertices of the target cell T_j, and v_i represents each vertex. The neighboring cells refer to the cells that share the vertex point, and $\bar{q}_{neighbor}^{min}$ and $\bar{q}_{neighbor}^{max}$ represent the minimum and maximum averaged values of neighboring cells. The estimated value of vertex point $(\hat{q}_{v_i})$ can be calculated by Eq. 3.2.

$$\hat{q}_{v_i} = \bar{q}_j + \phi \nabla q_j \cdot \vec{r}_{v_i,j}. \qquad \text{Eq. 3.2}$$

In this equation, $\bar{q}_j$ represents the averaged value of the target cell T_j, ∇q_j represents the gradient of target cell T_j, and $\vec{r}_{v_i,j}$ represents the distance vector from cell center point of T_j to vertex point v_i. Finally, ϕ represents limiter value. To ensure that the estimated value at the vertex point satisfies the condition mentioned earlier, the MLP limiter is applied to the estimated gradient, restricting its slope, and preventing unphysical oscillations or overshoots in the solution. The range of MLP limiter is then obtained by Eq. 3.1 and Eq. 3.2 as follows.

$$0 \leq \phi \leq \max\left(\frac{\bar{q}_{v_i}^{\min} - \bar{q}_j}{\nabla \bar{q}_j \cdot \vec{r}_{v_i,j}}, \frac{\bar{q}_{v_i}^{\max} - \bar{q}_j}{\nabla \bar{q}_j \cdot \vec{r}_{v_i,j}}\right). \qquad \text{Eq. 3.3}$$

From Eq 3.3, the MLP limiters can be obtained as follows.

$$\phi = \min\begin{cases} \Phi(r_{v_i,j}) & \text{if } \nabla \bar{q}_j \cdot \vec{r}_{v_i,j} < 0 \\ 1 & \text{otherwise} \end{cases}, \qquad \text{Eq. 3.4}$$

where

$$\Phi(r_{v_i,j}) = \max\left(\frac{\bar{q}_{v_i}^{\min} - \bar{q}_j}{\nabla \bar{q}_j \cdot \vec{r}_{v_i,j}}, \frac{\bar{q}_{v_i}^{\max} - \bar{q}_j}{\nabla \bar{q}_j \cdot \vec{r}_{v_i,j}}\right). \qquad \text{Eq. 3.5}$$

Chapter 4 Data-driven Reconstruction Method

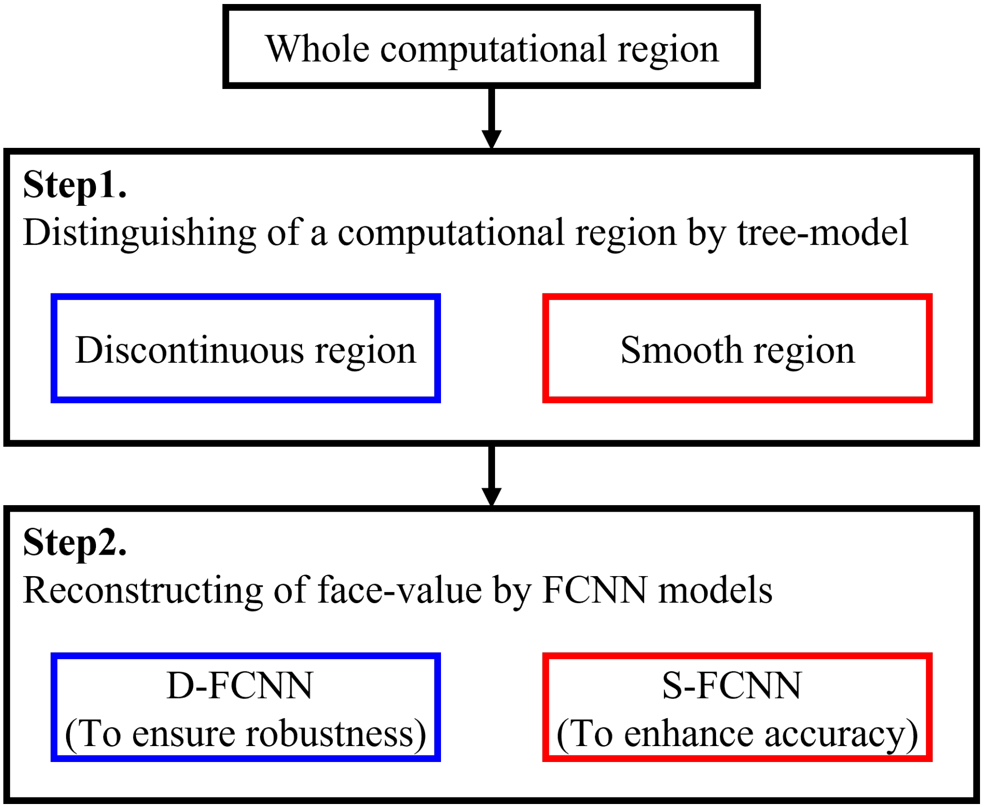


Figure 4.1: Overall process of the data-driven reconstruction method

The primary objective of this study, as discussed in Ch. 1, is to maintain high robustness in discontinuous regions and enhance accuracy in smooth regions using only the MLP stencil. To accomplish this objective, this study employs a two-step procedure, as illustrated in Figure 4.1. In the first step, a tree model is used to distinguish the whole computational domain into two distinct cells: discontinuous cells and smooth cells. In the second step, appropriate fully connected neural network (FCNN) models are employed to reconstruct the interface values of the cells. For robustness in discontinuous cells, the discontinuous FCNN (D-FCNN) model is employed, trained on a dataset representing discontinuous solution distribution. Conversely, the smooth FCNN (S-FCNN) model, trained on a dataset representing smooth solution distribution, is used to achieve enhanced accuracy in smooth regions.

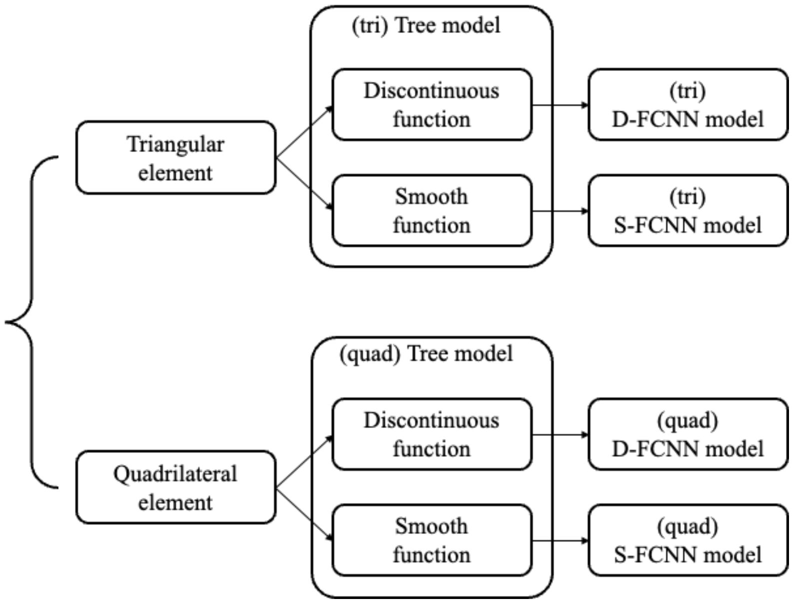


Figure 4.2: Datasets and models used in this study

The datasets and models employed in this study are depicted in Figure 4.2. The primary objective of the study is to develop a method applicable to irregular mixed grids. To achieve this, both triangular and quadrilateral elements are considered. For each type of element, two types of analytic functions are used: one representing discontinuous solution distributions and the other representing smooth solution distributions. This results in four datasets combining the element types with their corresponding solution distribution types.

To address the distinction between function types for each element, two tree models are trained for each element type. These tree models are specifically designed to differentiate between discontinuous and smooth function types.

Additionally, each dataset is associated with a corresponding FCNN model. This ensures that each dataset is paired with an appropriate FCNN model for accurate predictions. Consequently, four FCNN models are employed in this study, aligning with the four datasets.

The following section provides a comprehensive description of the datasets and models utilized in this study.

4.1 Datasets

Before diving into the detailed explanations, let's provide a summary of the data construction and preprocessing in this study:

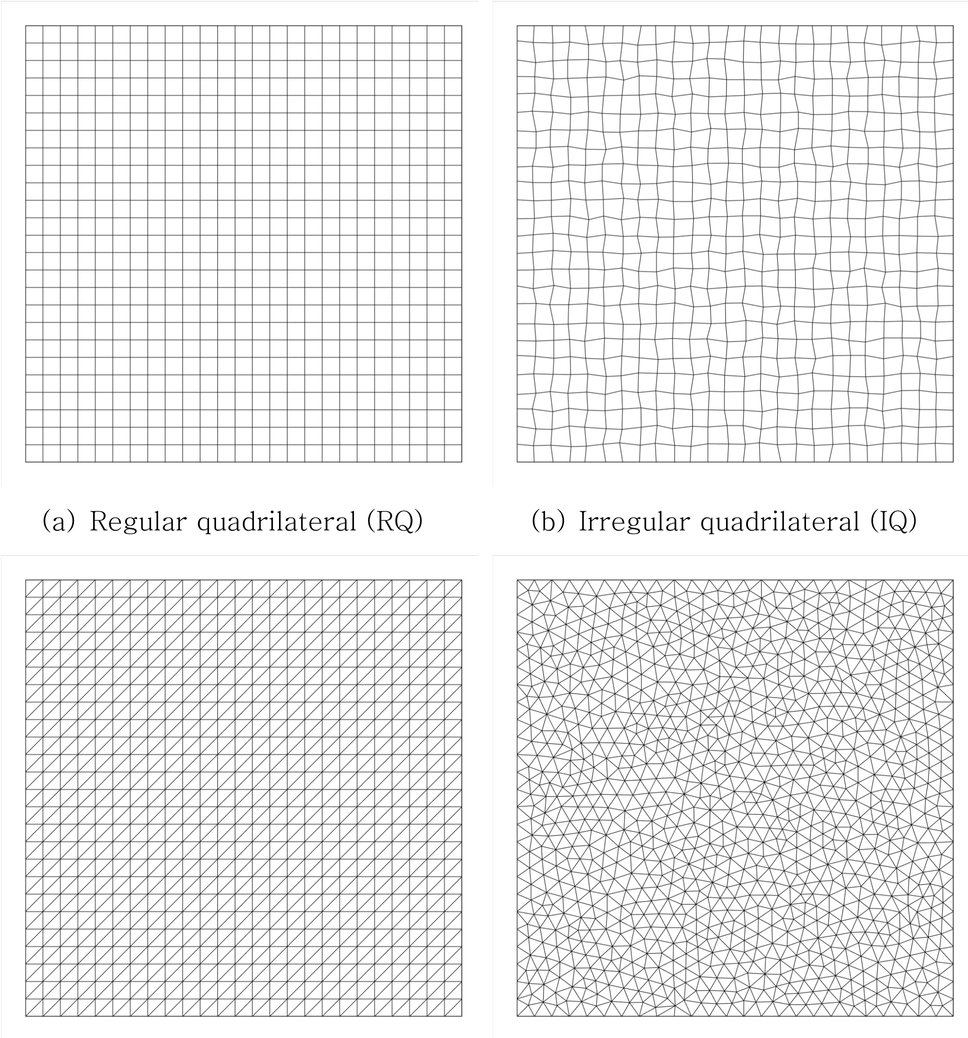
1. Mesh generation: Two types of meshes, regular and irregular, are generated for both quadrilateral and triangular element types.
2. MLP stencil extraction: From the generated mesh, the MLP stencil is extracted.
3. Assignment of analytic function: An analytic function is assigned to represent the desired solution distribution. This function can capture both discontinuous and smooth characteristics.
4. Calculation of flow variables: Flow variables are calculated based on the assigned analytic function. These variables provide essential information for computing the proper input features and label values required for model training.

In the following sections, a detailed explanation of each step will be provided.

4.1.1 Data Construction

The first step to construct data is the generation of the mesh. In this study, two types of elements, quadrilateral and triangular, are considered to ensure the method's extension to irregular mixed meshes. For each element type, the dataset is constructed using two types of meshes: regular and irregular. As a result, four types of meshes are employed: regular quadrilateral (RQ), irregular quadrilateral (IQ), regular triangular (RT), and irregular triangular (IT), as illustrated in Figure 4.3.

The quadrilateral dataset is generated using both RQ and IQ meshes, while the triangular dataset is created using RT and IT meshes. The IT mesh is constructed using the Delaunay algorithm, while the IQ mesh is formed by introducing random perturbations ranging from 0% to 20% to the regular quadrilateral mesh.



(a) Regular quadrilateral (RQ) (b) Irregular quadrilateral (IQ)

(c) Regular triangular (RT) (d) Irregular triangular (IT)

Figure 4.3: Four types of meshes

Once the meshes are generated, the MLP stencils are extracted from each mesh.

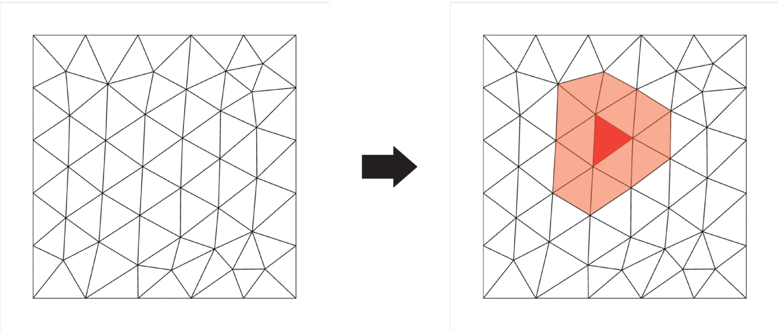


Figure 4.4: MLP stencil extraction

Following that, two distinct types of random functions, namely discontinuous and smooth, are applied to the stencils. This process produces discontinuous and smooth data. The discontinuous data exhibits abrupt changes in the flow field, while the smooth data represents a smoother variation. Once the random functions are assigned, the subsequent process varies based on the type of function.

- Discontinuous data

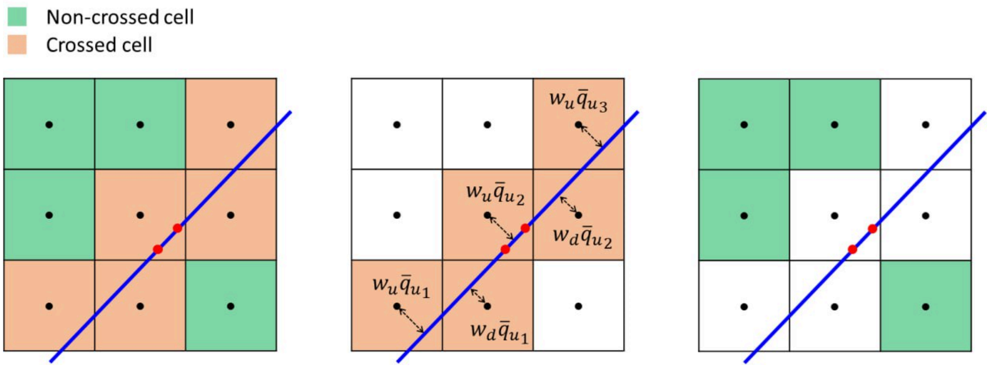
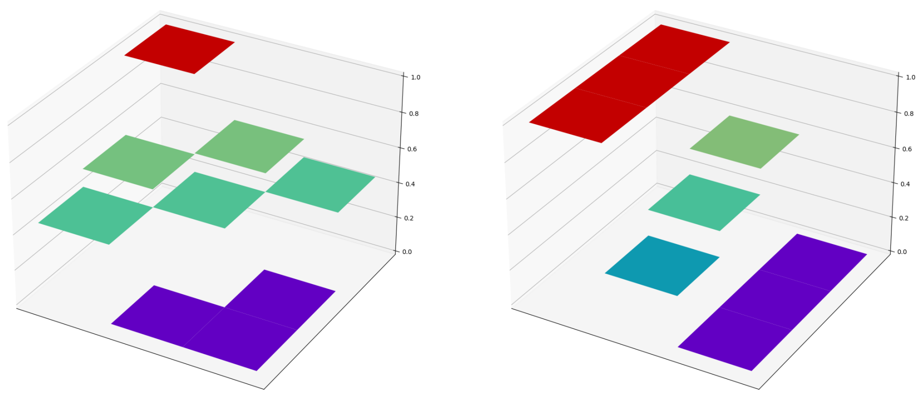


Figure 4.5: Process of assigning cell-averaged values for discontinuous data

1. Two points are randomly selected within the target cell.
2. Then, a line is defined based on these two points, and the cells that the line crosses and the cells that the line does not cross are distinguished.
3. The distance between the cell-centered coordinates and the line is calculated for the crossed cells.
4. Cell-averaged values $(\bar{q})$ are assigned to the crossed cells based on the corresponding distances.
5. The assigned values are multiplied by weight values, w_u for the upward direction and w_d for the downward direction, to create various distributions. w_u and w_d are randomly selected between 0 and 1.

6. The assigned values are scaled to set them within the range of 1 to 3.
7. The assigned values are re-scaled using the formula $(\bar{q}C) + (2 - 2C)$, where C is randomly selected between 0 and 1. This formula sets the assigned values in the range from $(2 - C)$ to $(2 + C)$.
8. For the non-crossed cells, values are assigned in the upward direction from $(2 + C)$ to $(2 + 10C)$ and in the downward direction from $(2 - 10C)$ to $(2 - C)$.

By following this process, the computation is efficient in generating discontinuous distributions, making it a reliable alternative to using mensuration by parts. The examples of the generated data are illustrated in Figure 4.6.



(a) case1　　　(b) case2

Figure 4.6: Examples of discontinuous data (RQ)

- **Smooth data**

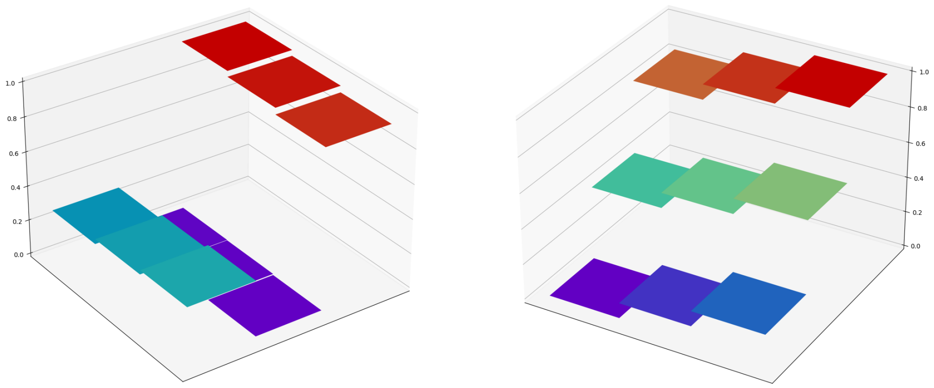
On the other hand, for generating smooth data, the coefficients of the equation given in Eq. 4.1 are randomly selected within a specified range. The range of coefficients is determined through a trial-and-error process to ensure a smooth variation in the data. The specific range of coefficients can be found in Table 4.1.

$$q = a(x - x_p)^2 + b(y - y_p)^2 + c(x - x_p)(y - y_p). \qquad \text{Eq. 4.1}$$

In this equation, x and y represent the coordinates of the cell-center, while x_p and y_p are two randomly selected points within the target cell.

Table 4.1: Range of the coefficients for smooth data

Coefficient	Range
a	$\mathbb{R}[-10,10]$
b	$\mathbb{R}[-10,10]$
c	$\mathbb{R}[-5,5]$



(a) case1 (b) case2

Figure 4.7: Examples of smooth data (RQ)

To compute the inner-face values for training the fully connected neural network (FCNN) model, distinct methods are employed for discontinuous and smooth datasets.

In the case of discontinuous data, the MLP limiter function is employed to ensure robustness. The computation of the MLP limiter value involves calculating the gradient vector of the target cell using the previously assigned values $(\bar{q})$ and the least-square method. The MLP limiter value is then derived following Eq. 3.4 and Eq. 3.5. Subsequently, this MLP limiter value is applied to compute limited reconstructed values at the face center, which are used as the inner-face values for training the FCNN model.

On the other hand, for smooth data, the values extracted directly from the analytic function are used to enhance accuracy. Since smooth data does not require the calculation of gradients or MLP limiter values, the inner-face values for training the FCNN model are simply obtained as the average value along the face. This approach ensures accurate representation of the smooth data for the FCNN training process.

4.1.2 Data Preprocessing

Data preprocessing plays a crucial role in machine learning (ML) by preparing raw data for effective model training. It involves various tasks, such as feature selection, feature ordering, and scaling.

In this study, both tree and fully connected neural network (FCNN) models require fixed-dimensional input features. Therefore, data preprocessing is essential to ensure consistent input dimensions for the models, especially due to the varying number of cells in MLP stencils in unstructured meshes. To achieve this, a compact stencil is extracted from the MLP stencil. This compact stencil includes the target cell and its neighboring cells, forming a subset of the original MLP stencil. The input features are then defined based on this compact stencil.

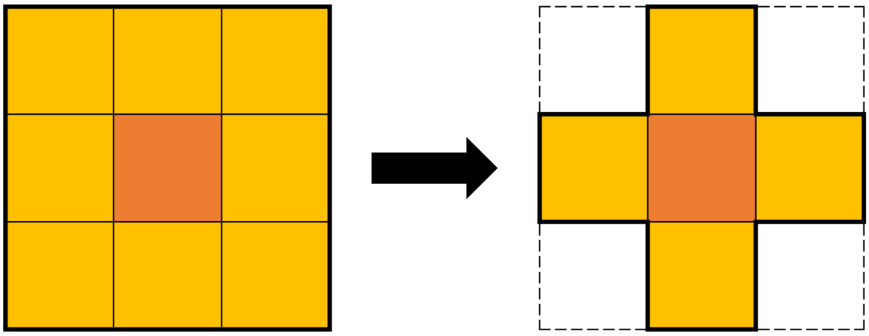


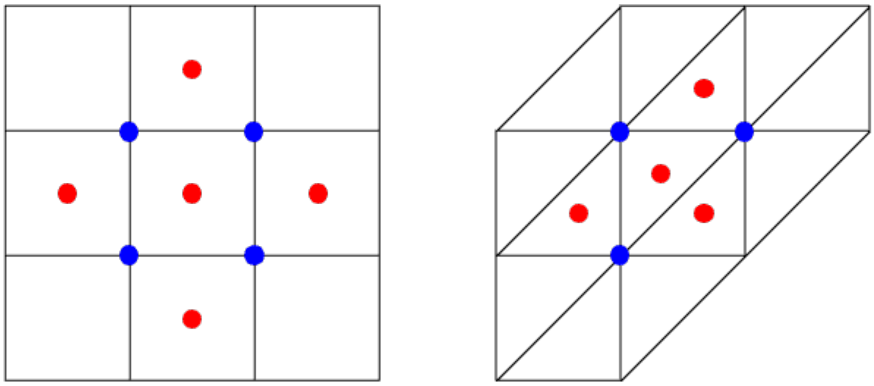
Figure 4.8: Extraction of compact stencil from MLP stencil

The input features consist of two components. Firstly, cell-averaged values $(\bar{q})$ are extracted from the compact stencil, providing essential information about the flow field. Secondly, inverse distance weighted values $(\hat{q})$ are calculated for the vertices of the target cell using Eq. 4.2. These values incorporate both the coordinate information of the target cell and information from the full MLP stencil.

$$\hat{q}_{v_i} = \frac{\sum_{T_k \in S_{T_j}} \bar{q}_k \cdot \frac{1}{|\overrightarrow{r_{v_i k}}|}}{\sum_{T_k \in S_{T_j}} \frac{1}{|\overrightarrow{r_{v_i k}}|}}. \qquad \text{Eq. 4.2}$$

Here, S_{T_j} represents the full MLP stencil of target-cell (T_j), and $|\overrightarrow{r_{v_i k}}|$ represents the distance between cell T_k and vertex v_i.

This extraction process ensures that the input features have a fixed dimension for each element type. For quadrilateral element types, the input features have a dimension of 9, while for triangular element types, the dimension is set to 7.

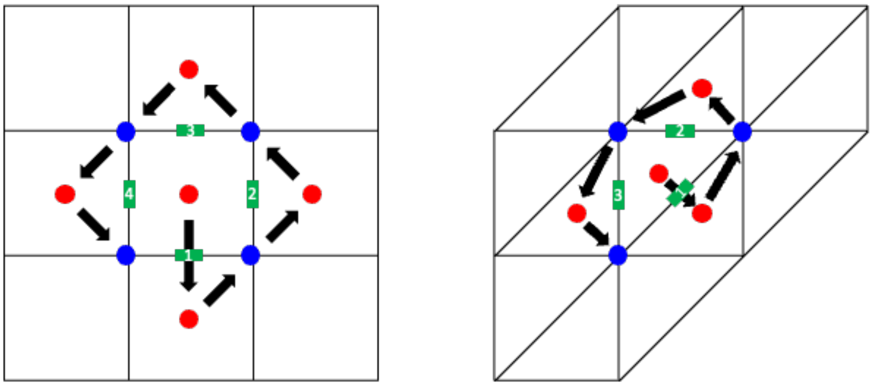


(a) Quadrilateral element (b) Triangular element

Figure 4.9: Extracted input features of MLP stencil

Additionally, to satisfy the rotational symmetry condition, an extra step is taken to order the input features. The target-cell averaged value $(\bar{q}_0)$ is placed first, and then the minimum cell-averaged value is selected. The remaining features are then arranged in a counterclockwise rotation around the target cell. This ordering scheme ensures that the input features maintain rotational symmetry. Furthermore, the label values (inner-face values) are matched with the order of cell-averaged values, guaranteeing that the input features and corresponding label values are correctly aligned and consistent during the training process. The input features $(\boldsymbol{q})$ expressed by Eq. 4.3 and label values are illustrated in Figure 4.10.

$$\begin{cases} \boldsymbol{q}_{quad} = \{\bar{q}_0, \bar{q}_1, \hat{q}_0, \bar{q}_2, \hat{q}_1, \bar{q}_3, \hat{q}_2, \bar{q}_4, \hat{q}_3\} \\ \boldsymbol{q}_{tri} = \{\bar{q}_0, \bar{q}_1, \hat{q}_0, \bar{q}_2, \hat{q}_1, \bar{q}_3, \hat{q}_2\} \end{cases}. \qquad \text{Eq. 4.3}$$



(a) Quadrilateral element (b) Triangular element

Figure 4.10: Order of input-features and label values

Additionally, it is important to normalize the input features. This normalization step involves scaling the features to a range of 0 to 1, considering the minimum and maximum values of the MLP stencil. By normalizing the input features in this manner, the information from the MLP stencil is incorporated into the compact stencil, ensuring that the models can effectively utilize the data.

$$q = \frac{q - \bar{q}_{min}}{\bar{q}_{max} - \bar{q}_{min}}, \qquad \text{Eq. 4.4}$$

where $\bar{q}_{min}$ denotes minimum cell averaged value of the MLP stencil and $\bar{q}_{max}$ denotes maximum cell averaged value of the MLP stencil.

Overall, the data preprocessing steps described in this study ensure that the input features have fixed dimensions, satisfy rotational symmetry, and are normalized appropriately, providing a suitable and standardized format for training the ML models.

The final dataset used to train is summarized in Table 4.2.

Table 4.2: Datasets used in this study

Type of element	Type of function	# of data
Quadrilateral element	Discontinuous function	100,000
	Smooth function	100,000
Triangular element	Discontinuous function	100,000
	Smooth function	100,000

4.2 Tree Models for Shock Indicator

In this study, two tree models are employed as shock indicators to differentiate between discontinuous and smooth cells. One model is designed for triangular element type cells, while the other is tailored for quadrilateral element type cells. In the subsequent section, we provide a comprehensive explanation of the training algorithm employed for the tree models, including the hyperparameter settings and the obtained results.

4.2.1 Decision Tree

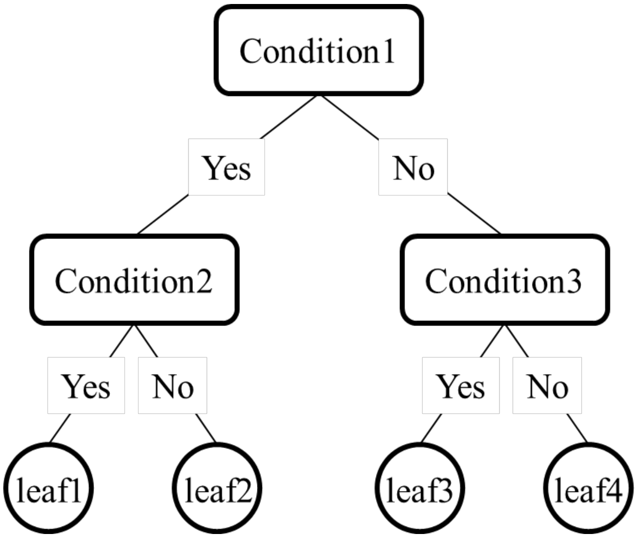


Figure 4.11: Structure of a decision tree (depth = 3)

The decision tree [19] as shown in Figure 4.11 is a widely used machine learning algorithm for classification tasks. It is known for its simplicity and interpretability, as it employs if/else/then rules, making it easy to understand and implement. The algorithm partitions the dataset by conditions that maximize the homogeneity of the resulting groups. The final score which is the output of the tree is assigned to the leaf nodes of the tree.

4.2.2 Ensemble Method

To enhance the accuracy of a tree model, ensemble methods [20], [21] are introduced. Ensemble methods are machine learning techniques that combine multiple individual models to create a more robust and accurate predictive model. The idea behind ensemble methods is that by aggregating the predictions of multiple models, the collective result can outperform any individual model.

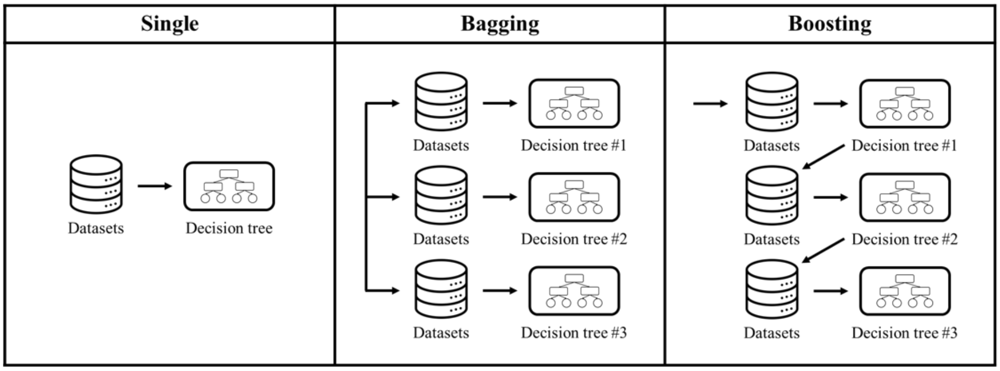


Figure 4.12: Comparison of single method and ensemble methods

There are different types of ensemble methods, but two commonly used ones are:

1. Bagging (Bootstrap Aggregating): In bagging, multiple models are trained on different subsets of the training data, which are randomly sampled with replacement. Each model is trained independently, and the final prediction is made by averaging or voting on the predictions of all the models.
2. Boosting: Boosting is a sequential ensemble method where models are trained iteratively. Each model in the sequence focuses on correcting the mistakes or misclassifications made by the previous models. The final prediction is made by combining the predictions of all the models, often using weighted voting.

Ensemble methods offer several benefits. They can improve the predictive performance and generalization of models by reducing bias and variance. By combining multiple models, ensemble methods can capture different patterns and make more accurate predictions. They are also effective at handling noisy or uncertain data.

In the study, the gradient boosting method (GBM) is selected as the boosting algorithm to build tree models for the shock-indicator.

4.2.3 Gradient Boosting Method (GBM) Algorithm

GBM [22] is a popular and powerful machine learning technique that combines weak models, typically decision trees, to create a strong predictive model. It is an iterative algorithm that sequentially trains models to minimize a loss function. Here, we will describe the algorithm and its key equations.

The gradient boosting algorithm starts by initializing the model with a constant value, which serves as the initial prediction for all instances in the dataset. In each iteration, a weak model, often a decision tree, is trained to predict the negative gradient of the loss function based on the current ensemble's predictions. This negative gradient indicates the direction in which the model needs to be adjusted to minimize the loss, and it can be interpreted as the residual or error.

The prediction of the current ensemble at iteration m is denoted as $F_m(X)$. Here, X represents input features. The negative gradient of the loss function, denoted as r_m, is calculated by the Eq. 4.5.

$$r_m = -\frac{\partial L\big(y, F_m(X)\big)}{\partial F_m(X)}. \qquad \text{Eq. 4.5}$$

Here, $L\big(y, F_m(X)\big)$ represents the loss function, which measures the discrepancy between the true target values y and the current ensemble's predictions $F_m(X)$.

The weak model is trained to fit the negative gradient, acting as a so-called "residual" model. It learns to predict the remaining errors of the ensemble. The weak model is trained on the dataset (X, r_m), where the inputs X are the features and the target values r_m are the negative gradients.

Once the weak model is trained, its predictions are added to the current ensemble:

$$F_{m+1}(X) = F_m(X) + \eta \cdot f_{m+1}(X). \qquad \text{Eq. 4.6}$$

In this equation, η is the learning rate, which controls the contribution of each weak model to the ensemble. It is a hyperparameter that needs to be tuned. The weak model $f_{m+1}(X)$ is scaled by the learning rate η and added to the current ensemble's predictions $F_m(X)$.

This process continues for a predetermined number of iterations or until a convergence criterion is met. The final ensemble prediction is the sum of all weak model predictions:

$$F(X) = F_0(X) + \sum_{i=1}^{M} (\eta \cdot f_i(X)). \qquad \text{Eq. 4.7}$$

Here, M represents the total number of iterations or the number of weak models in the ensemble. The final model $F(X)$ is the output of the gradient boosting algorithm, which can be used to make predictions on new, unseen instances.

4.2.4 Hyperparameter Settings

To train the tree models, the XGBoost library [23] is employed in this study. In the XGBoost, to build the models, the number of trees and the maximum depth of each tree are important hyperparameters that need to be determined. The number of trees refers to the total number of decision trees that will be trained in the ensemble. The maximum depth of each tree defines the maximum number of splits or levels in the tree structure.

Another crucial hyperparameter is the base value, which represents the initial prediction made by the first tree in the ensemble. In addition, the learning rate, another important hyperparameter, controls the contribution of each tree to the final ensemble. A smaller learning rate allows for more conservative updates to the model, while a larger learning rate can lead to faster convergence but may also introduce instability. Moreover, the choice of loss function depends on the specific problem being addressed. In this study, since the tree models are used as classification models, the logistic loss function is selected.

In the Table 4.3, the hyperparameters used in this study are summarized, including the number of trees, maximum depth, base value, and learning rate for both triangular and quadrilateral models. These hyperparameters are carefully chosen through experiments and optimization to achieve the best performance for the specific task at hand.

Table 4.3: Hyperparameter setting of tree models

Hyperparameter	**Value**
# of estimators (trees)	3
Maximum depth	5
Base score	0.5
Learning rate	0.1
Loss function	Logistic

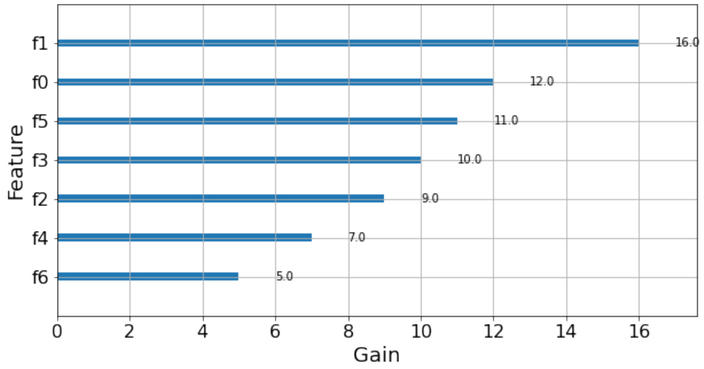
4.2.5 Training Results

The training results are summarized in Table 4.4. The table presents the total accuracy of the models, as well as the accuracy for both the discontinuous data and the smooth data.

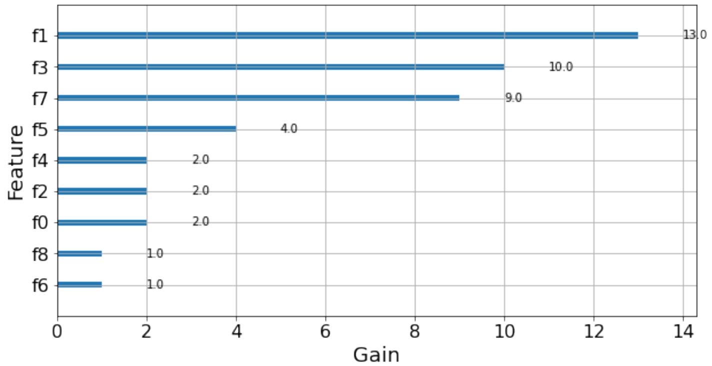
Table 4.4: Results of tree models

Model	Type of Accuracy	Value
Tri-tree model	Total accuracy	98.85%
	Discontinuous data accuracy	99.4%
	Smooth data accuracy	98.2%
Quad-tree model	Total accuracy	99.5%
	Discontinuous data accuracy	99.0%
	Smooth data accuracy	99.9%

Using the tree model, the importance of input features can be determined, which indicates their significance in partitioning the data. The importance of input features was evaluated for both the triangular and quadrilateral datasets, and the results are shown in Figure 4.12. The results reveal that all input features are important, with particular emphasis on the averaged values of face-neighboring cells (f1, f3, f5, f7), which are found to be more influential than the inverse distance weighted values of the vertices of the target-cell (f2, f4, f6, f8).



(a) Triangular dataset

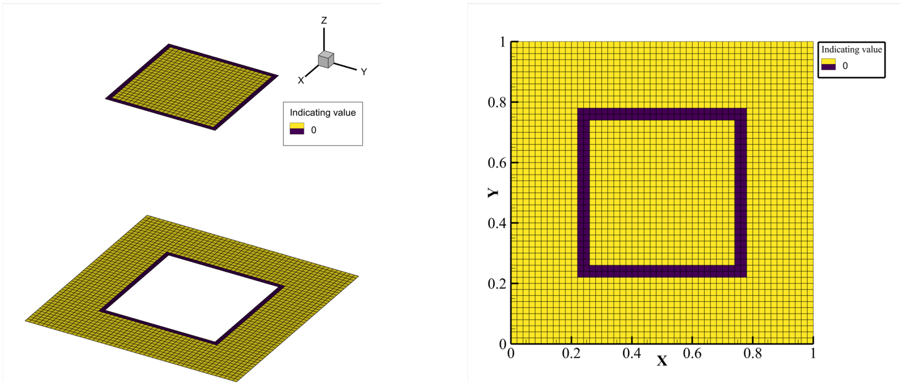


(b) Quadrilateral dataset

Figure 4.13: Feature importance

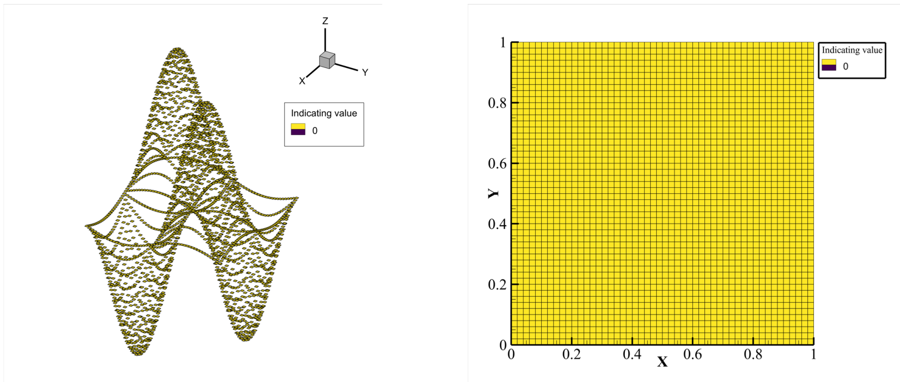
To test the performance of the quadrilateral tree model, different initial conditions are considered, including both discontinuous and smooth wave distributions. The tree mode is then applied to these conditions, and the resulting flow field is analyzed.

Figure 4.14 demonstrates the results for the discontinuous wave case, where the tree model effectively captures the discontinuities in the flow field. This highlights the model's ability to accurately represent the sharp changes in the solution. On the other hand, Figure 4.15 presents the results for the smooth wave case, showcasing the tree model's capability to accurately represent the entire cell as a smooth region.



(a) Solution distribution (b) Indicating value contour

Figure 4.14: Result of square wave problem (50×50)



(a) Solution distribution (b) Indicating value contour

Figure 4.15: Result of double sine wave problem (50×50)

It is important to note that these tests represent simplified problems aimed at testing the performance of the tree model. More advanced and complex tests, along with their results, are discussed in Ch. 5.

4.3 Fully Connected Neural Network Models for Reconstruction

In this study, fully connected neural networks (FCNNs) [24] are employed to develop inner-face regression models. The FCNN models are categorized into four types: tri-D-FCNN, tri-S-FCNN, quad-D-FCNN, and quad-S-FCNN. Each model is specifically trained on a particular dataset, with tri-D-FCNN trained on a triangular discontinuous dataset, tri-S-FCNN trained on a triangular smooth dataset, quad-D-FCNN trained on a quadrilateral discontinuous dataset, and quad-S-FCNN trained on a quadrilateral smooth dataset.

The subsequent section provides comprehensive explanations about the structure of the FCNN models. The model comparative study conducted to establish the criteria for successful training is presented. Additionally, the implementation of a grid search method to determine the optimal size of the FCNN models is discussed. Lastly, the outcomes of training these models are presented.

4.3.1 Fully Connected Neural Network (FCNN)

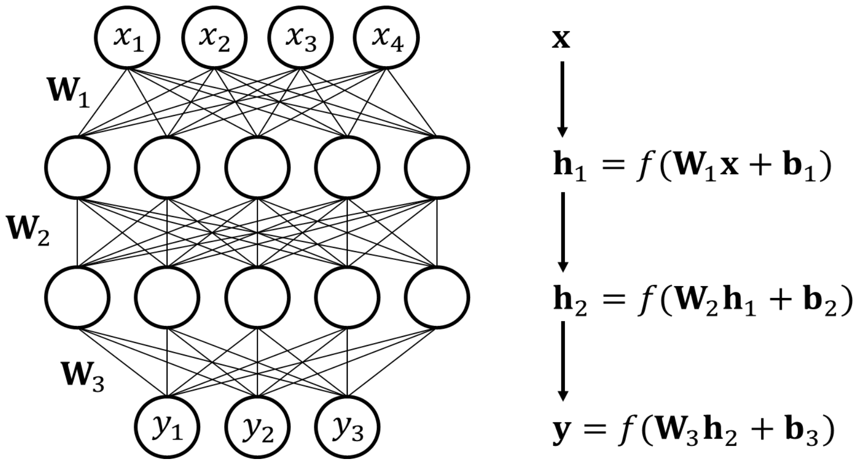


Figure 4.16: Structure of a FCNN model and computation process

A fully connected neural network (FCNN) is an artificial neural network architecture where each neuron in all layers is connected to every neuron in the preceding layer. This connectivity pattern can be visualized in Figure 4.16, where $\mathbf{x}$ represents the feature vector of the input layer, $\mathbf{h}$ represents the feature vector of the hidden layer, $\mathbf{W}$ represents the weight matrix, $\mathbf{b}$ represents the bias vector, and $\mathbf{y}$ represents the feature vector of the output layer. The architecture is well-suited for a variety of tasks due to its ability to capture complex relationships and learn from data.

4.3.2 Model Comparative Study

Before training models, it is crucial to establish that ensures their reliable performance. To determine this criterion, we conduct a model comparative study which is grid convergence test using MLP limiter and three quad-D-FCNN models, each having a distinct mean squared error (MSE) value: order of $10^{-4}, 10^{-5}$, and 10^{-6}.

If the training is successful, the quad-D-FCNN models should exhibit similar convergence rates of L1 and L2 errors, as well as a similar error range, compared to the MLP limiter. This is because the D-FCNN models are trained on the dataset using the MLP limiter. By comparing the performance of the quad-D-FCNN models to the MLP limiter, we can assess the reliability and effectiveness of our trained models.

For the grid convergence test, a linear advection problem of a smooth wave is employed. The initial condition for this test is described by Eq 4.8 and depicted in Figure 4.16. The advection speed vector is set to $(1,0.5)$. In the computation process, the local Lax-Friedrichs flux is selected as the numerical flux scheme. The TVK-RK3 method is employed for time integration.

$$q = \sin(2\pi(x + y)). \qquad \text{Eq. 4.8}$$

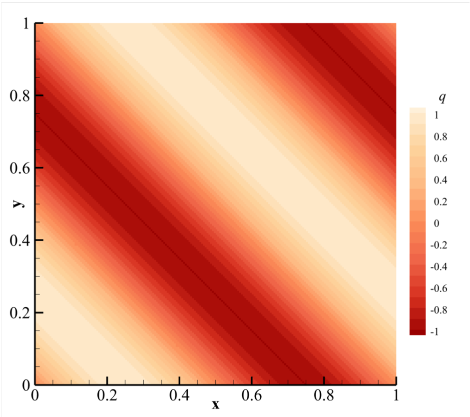
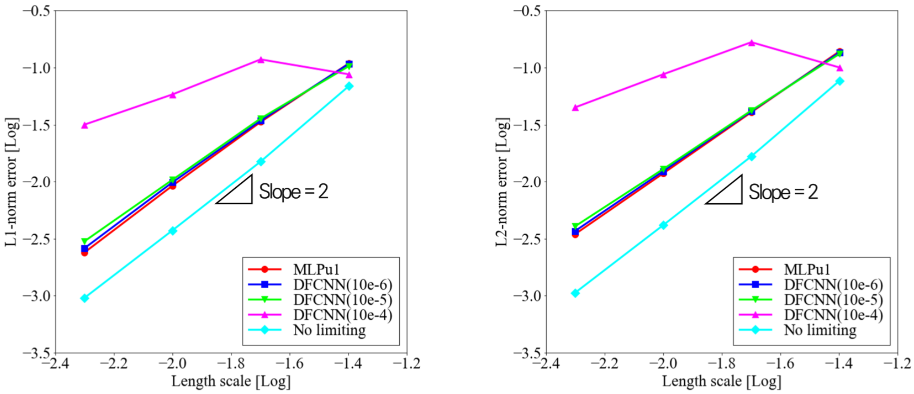


Figure 4.17: Initial condition contour (50×50)



(a) L1 error

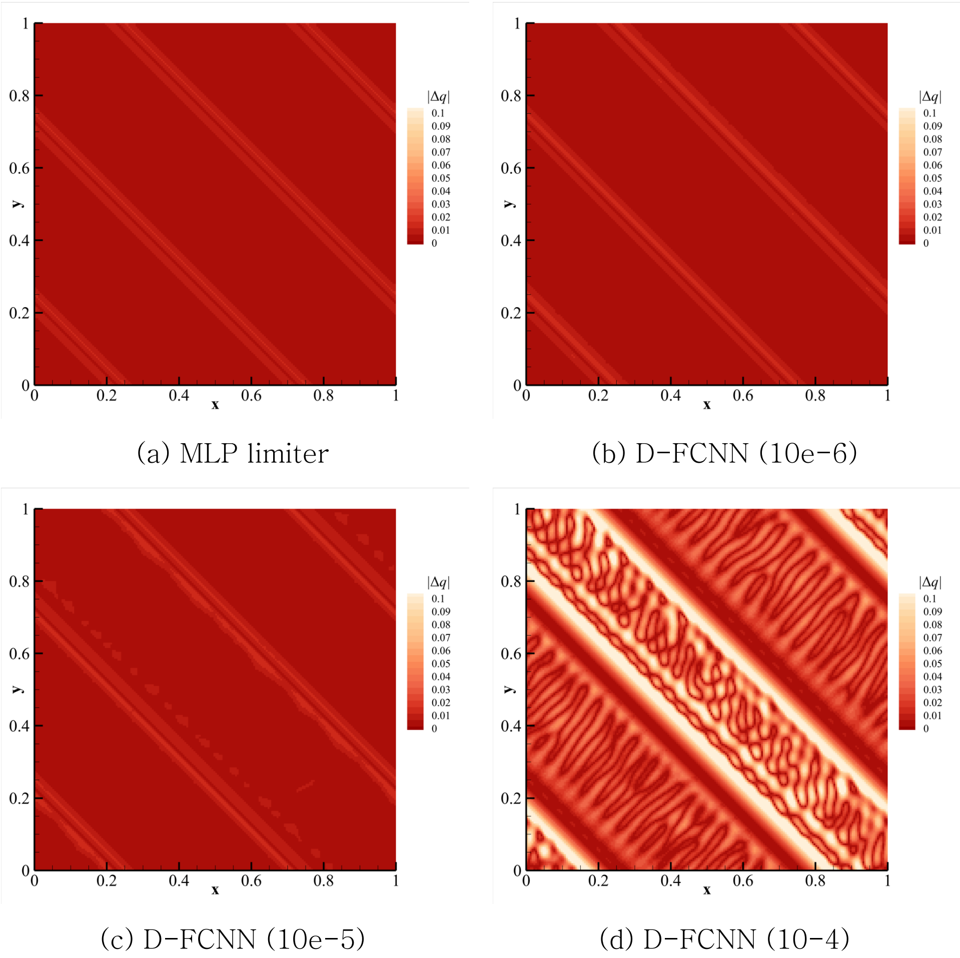
(b) L2 error

Figure 4.18: Convergence rates of L1 and L2 errors

In Figure 4.18, the convergence rates of the L1 and L2 errors are presented. The results indicate that the model with an MSE of 10^{-4} shows an erratic convergence rate. However, the models with an MSE below 10^{-5} demonstrate a consistent second-order convergence rate. This suggests that achieving an MSE of 10^{-4} is insufficient for achieving accurate training.

To further analyze the results, the absolute errors between the initial solution and the advected solution using the MLP limiter and three D-FCNNs are examined. Figure 4.19 illustrates that the absolute error using the MLP limiter is concentrated only in the peak region. Similarly, the D-FCNN with an MSE below 10^{-5} exhibits a similar behavior to the MLP limiter, with localized errors around the peak. However, the absolute error using the model with an MSE of 10^{-4} shows more widespread errors throughout the domain.

Based on the results of the model comparative study, a criterion for successful training has been established: an MSE value below 10^{-5}.



(a) MLP limiter

(b) D-FCNN (10e-6)

(c) D-FCNN (10e-5)

(d) D-FCNN (10-4)

Figure 4.19: Absolute error contour (200×200)

4.3.3 Grid Search

The size of FCNN models directly impacts the computational cost of the computational fluid dynamics (CFD) solver. As each cell requires the FCNN models to perform feed-forward calculations for determining inner-face values, it is essential to minimize the computational cost by selecting the minimum size of the FCNN models.

To determine the minimum size of the FCNN models, a grid search is conducted. The grid search involves varying the number of hidden layers and the number of neurons within those hidden layers in the FCNN models. The objective is to identify the optimal size that satisfies the criteria established through the model comparative study for successful training.

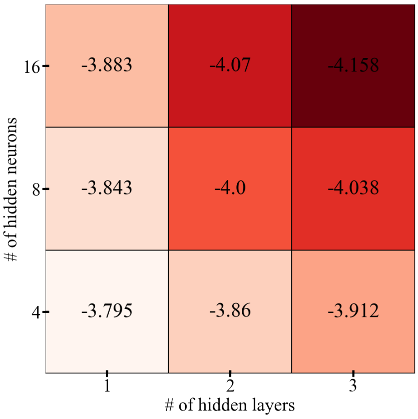
The range of hidden layers explored ranged from 1 to 3, while the number of hidden neurons varied between 4, 8, and 16. During training, the adam optimizer was utilized, and the learning rate was scheduled using CosineAnnealingLR. A batch size of 1,000 was employed, and the maximum number of epochs for the grid search test was set to 2,000. To ensure robustness, each case was tested 5 times. A summary of the hyperparameter settings for this grid search test can be found in Table 4.5.

Table 4.5: Hyperparameter setting for grid search test

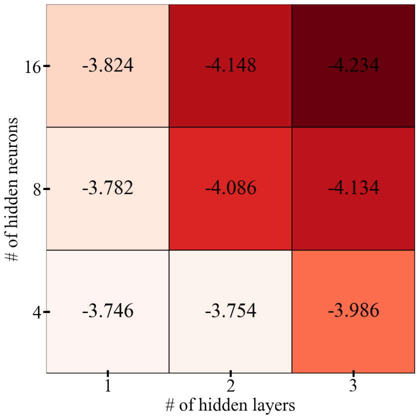
Hyperparameter	**Value**
# of hidden layers	[1,2,3]
# of hidden neurons	[4,8,16]
Optimizer	Adam
Scheduler	CosineAnnealingLR (0.01-0.0001)
Learning rate	0.01
Batch size	1,000
Maximum epoch	2,000

Figure 4.20 illustrates the results of the grid search test conducted for D-FCNNs. The values in the figure represent the MSE (Mean Squared Error) values on a logarithmic scale, indicating the performance of the models on the test dataset.

The findings reveal that for the triangular discontinuous dataset, the optimal D-FCNN size entails 2 hidden layers, each consisting of 16 neurons. Similarly, for the quadrilateral discontinuous dataset, the optimal size comprises 2 hidden layers, with 8 neurons in each layer. These sizes were determined based on their ability to yield low MSE values and effectively capture the desired patterns in the respective datasets.



(a) Triangular dataset



(b) Quadrilateral dataset

Figure 4.20: Results of grid search test (D-FCNN)

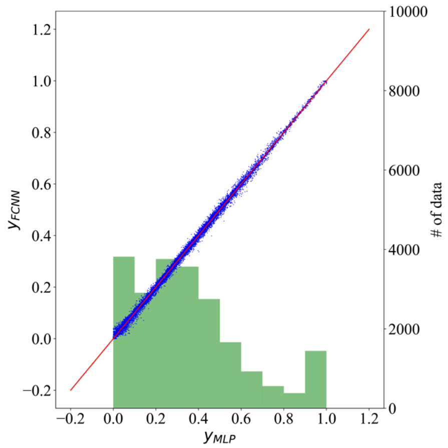
In contrast, for both types of smooth datasets (triangular and quadrilateral), a smaller model size consisting of 1 hidden layer with 8 neurons proved to be sufficient. Consequently, no grid search test was conducted for these cases, as the optimal size was readily identified.

4.3.4 Training Results

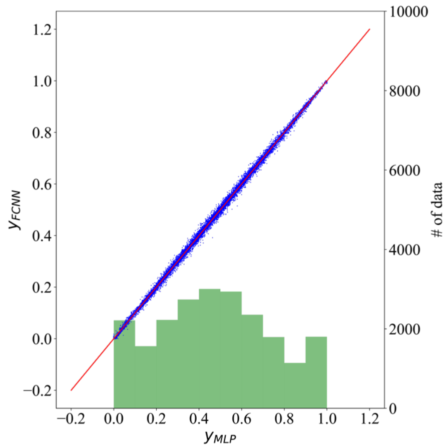
Figure 4.21, Figure 4.22, Figure 4.23, and Figure 4.24 illustrate the training results. The histogram represents the distribution of the label values, giving an overview of the dataset. The red line represents the line where the real value is equal to the predicted value. Additionally, the predicted values are represented by dots, providing a visual representation of how well the model aligns with the actual values.

Table 4.6: Training results of FCNN models

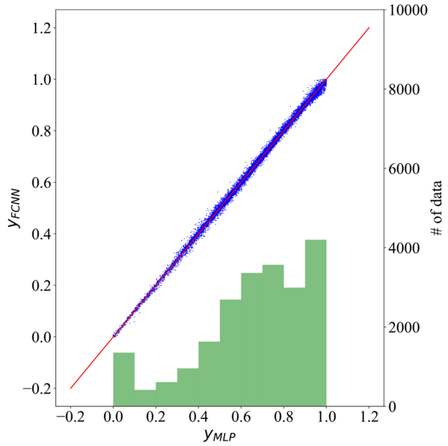
Model	MSE of test dataset
(quad) D-FCNN	7.9×10^{-5}
(quad) S-FCNN	7.1×10^{-5}
(tri) D-FCNN	7.5×10^{-5}
(tri) S-FCNN	7.3×10^{-5}



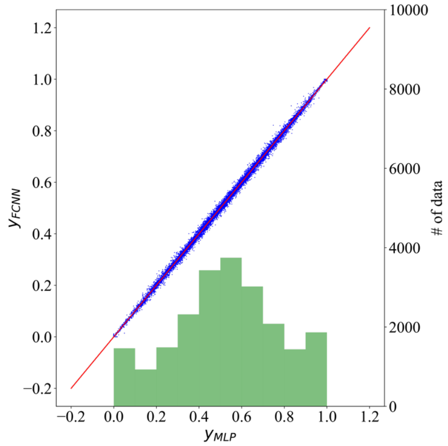
(a) y1



(b) y2

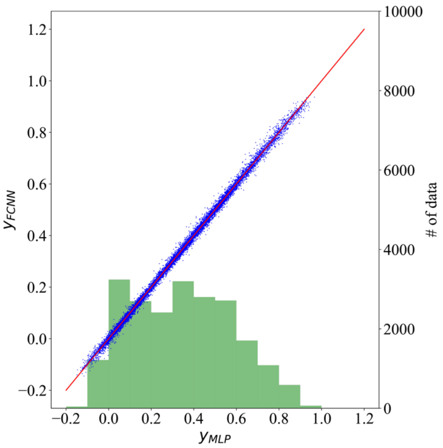


(c) y3

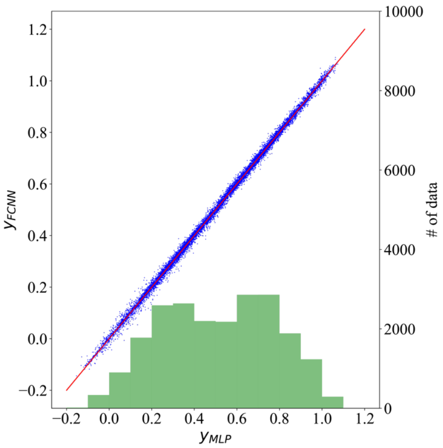


(d) y4

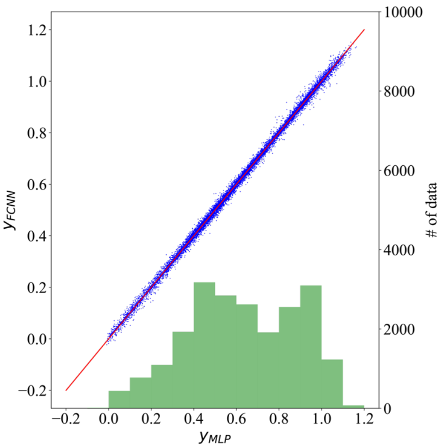
Figure 4.21: Training results of (quad) D-FCNN



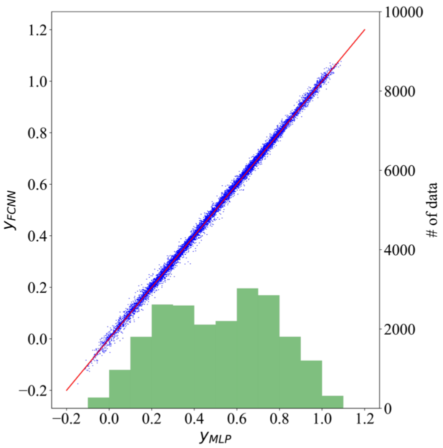
(a) y1



(b) y2

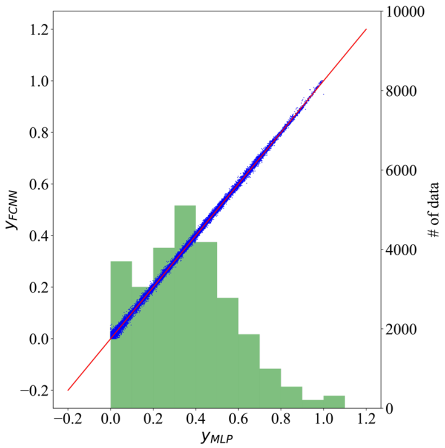


(c) y3

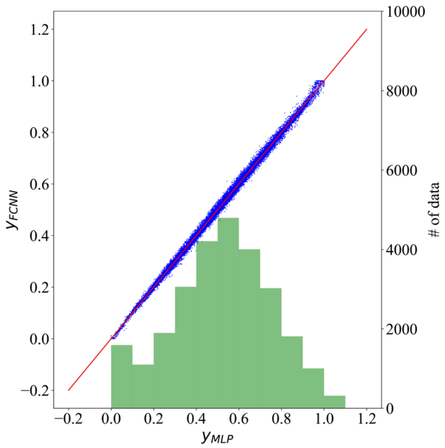


(d) y4

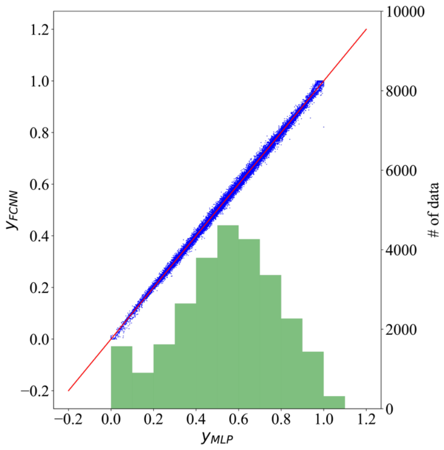
Figure 4.22: Training results of (quad) S-FCNN



(a) y1

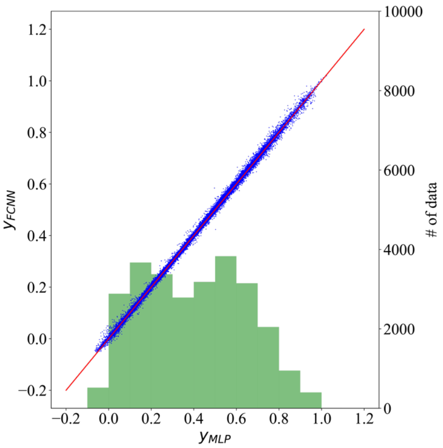


(b) y2

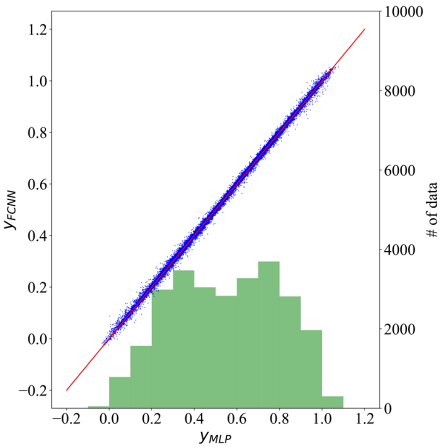


(c) y3

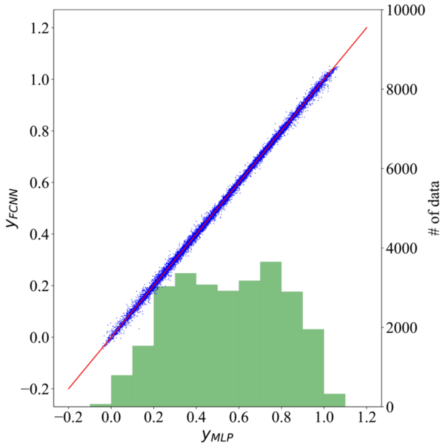
Figure 4.23: Training results of (tri) D-FCNN



(a) y1



(b) y2



(c) y3

Figure 4.24: Training results of (tri) S-FCNN

Chapter 5 Numerical Results

To evaluate the performance of the newly proposed data-driven reconstruction method on unstructured grids, extensive numerical experiments have been conducted. These experiments cover various test cases, including the linear wave problem, shock tube problems, isentropic vortex problem, and other well-known numerical test cases. The accuracy and robustness characteristics of the method are compared with conventional limiters such as MLP-u1 limiter and Barth's limiter. In these experiments, unless stated otherwise, the interpolation of variables is performed using primitive variables. The choice of numerical fluxes includes the local Lax-Friedrichs scheme and Roe-type schemes. For time integration, the third-order TVD Runge-Kutta method is employed.

5.1 Linear Advection Equation

In this section, scalar linear wave problems governed by Eq. 5.1 are considered.

$$q_t + \boldsymbol{a} \cdot \nabla q = 0, \qquad \text{Eq. 5.1}$$

where q is a scalar quantity, $\boldsymbol{a}$ is a constant wave velocity vector of $(1{,}0.5)$, and ∇q is a gradient vector of q. The numerical flux was calculated using a local Lax-Friedrichs (LLF) scheme. The computation domain for all simulations was $[0{,}1] \times [0{,}1]$, and periodic boundary conditions were applied. To assess the performance of both quadrilateral and triangular models, irregular mixed meshes are used for all simulations, with grid resolution of 50×50 as shown in Figure 5.1.

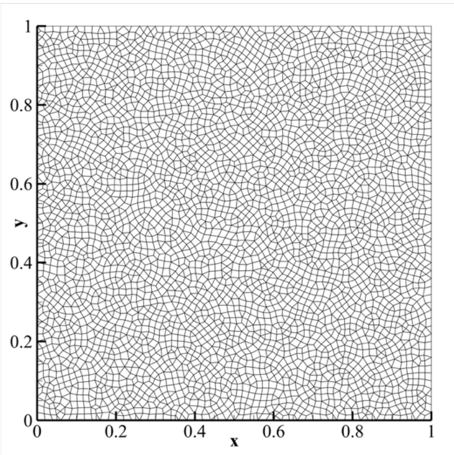


Figure 5.1: Irregular mixed mesh (50×50)

5.1.1 Discontinuous Wave Problems

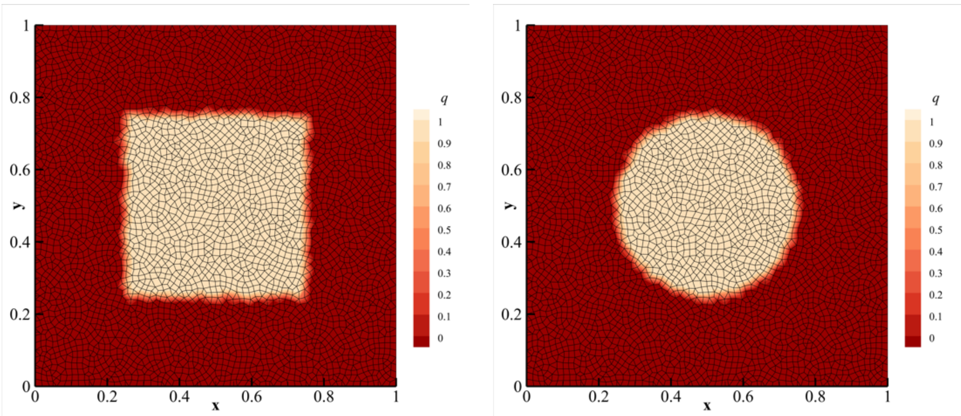
To analyze the oscillatory behavior across a discontinuity, two initial condition problems are considered: a square wave described by Eq. 5.2 and a circular wave described by Eq. 5.3.

1. Square wave problem:

$$q = \begin{cases} 1 & \text{if } 0.25 \leq x, y \leq 0.75 \\ 0 & \text{otherwise} \end{cases} \qquad \text{Eq. 5.2}$$

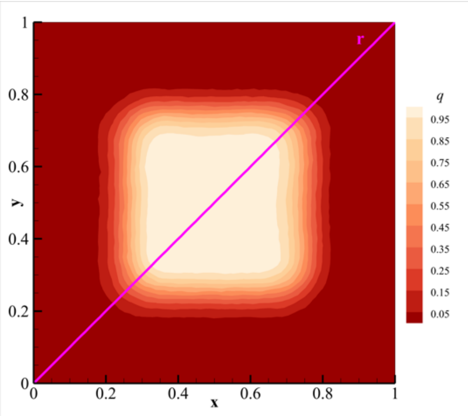
2. Circle wave problem:

$$q = \begin{cases} 1 & \text{if } (x - 0.5)^2 + (y - 0.5)^2 \leq 0.25^2 \\ 0 & \text{otherwise} \end{cases} \qquad \text{Eq. 5.3}$$

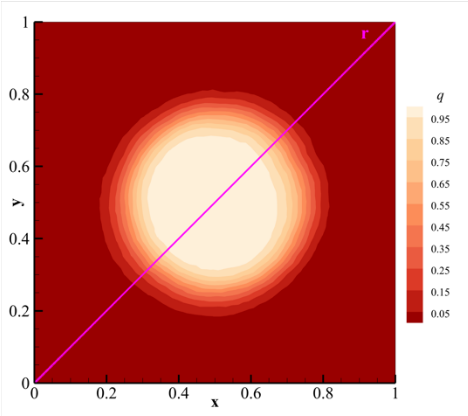


(a) Square wave (b) Circle wave

Figure 5.2: Initial condition of discontinuous wave problems

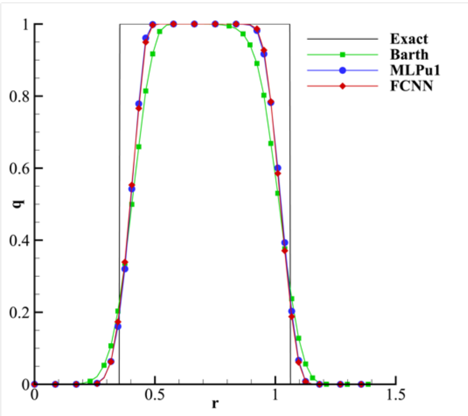


(a) Square wave

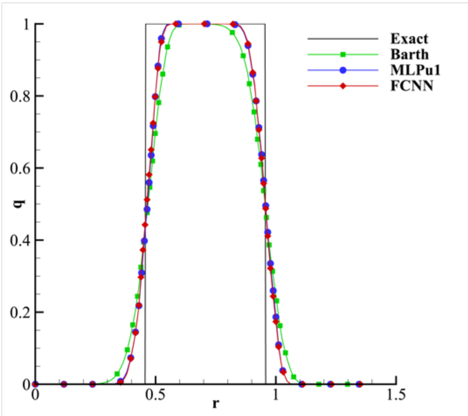


(b) Circle wave

Figure 5.3: Solution contour of discontinuous wave problems of the data-driven reconstruction method



(a) Square wave



(b) Circle wave

Figure 5.4: Solution distributions along the magenta line of discontinuous wave problems

Figure 5.3 and Figure 5.4 illustrate the numerical results of the square wave and circle wave at $t = 2.0$. In particular, Figure 5.3 presents a contour map of the solutions obtained using the data-driven reconstruction method. Figure 5.4 displays a comparison of the solution distributions along the magenta line depicted in Figure 5.3. The results demonstrate that both the Barth & Jespersen limiter and the MLP-u1 limiter, along with the data-driven reconstruction method, produce monotone solutions near the discontinuous profile. However, the MLP-u1 limiter and data-driven method capture the discontinuous region more sharply. These results highlight the successful replication of the robustness observed with the MLP-u1 limiter in discontinuous regions by the data-driven reconstruction method.

5.1.2 Smooth Wave Problems

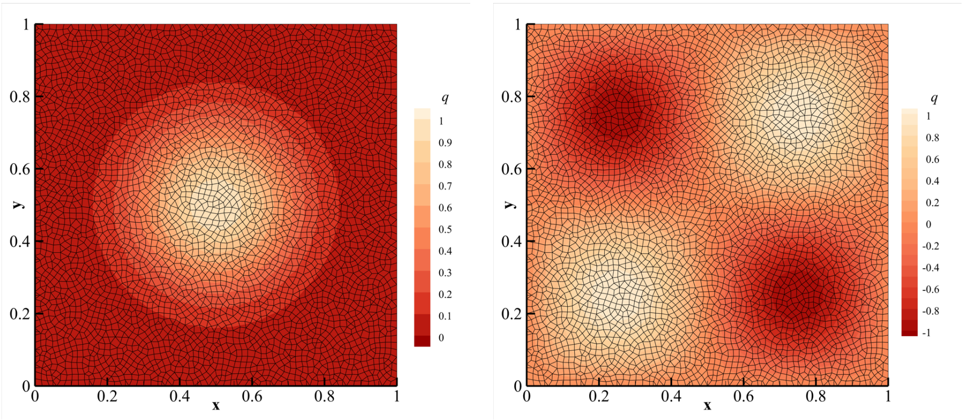
To examine the accuracy of the smooth wave, a Gaussian wave described by Eq. 5.4 and a double sine wave described by Eq. 5.5 are considered.

1. Gaussian wave problem:

$$q = e^{-20(x-0.5)^2(y-0.5)^2} \qquad \text{Eq. 5.4}$$

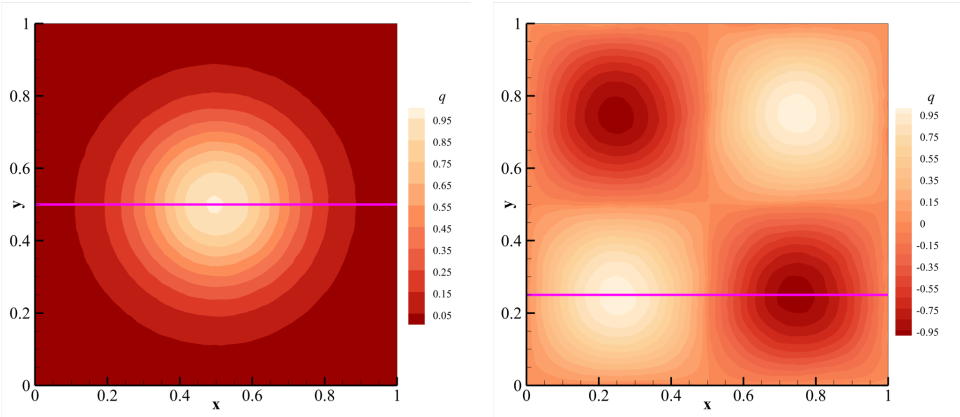
2. Double sine wave problem:

$$q = \sin(2\pi x)\sin(2\pi y) \qquad \text{Eq. 5.5}$$



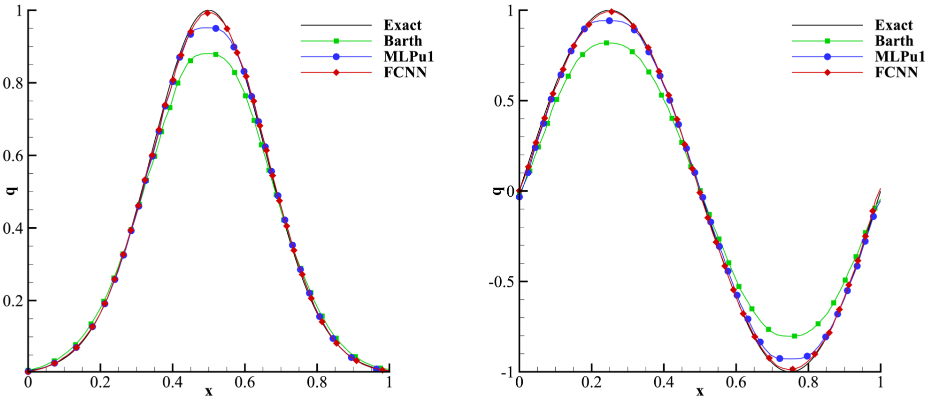
(a) Gaussian wave (b) Double sine wave

Figure 5.5: Initial condition of smooth wave problems



(a) Gaussian wave (b) Double sine wave

Figure 5.6: Solution contour of smooth wave problems of the data-driven reconstruction method



(a) Gaussian wave (b) Double sine wave

Figure 5.7: Solution distributions along the magenta line of smooth wave problems

Figure 5.6 and Figure 5.7 illustrate the numerical results of the Gaussian wave and double sine wave at $t = 2.0$. In Figure 5.6, a contour map of the solutions obtained using the data-driven reconstruction method is presented. Figure 5.7 displays a comparison of the solution distributions along the magenta line depicted in Figure 5.6. The results show that both the Barth & Jespersen limiter and the MLP-u1 limiter exhibit a clipping phenomenon that is diffusive at smooth extrema. However, the data-driven reconstruction method does not show diffusion at smooth extrema. Table 5.1 and Table 5.2 shows the peak values of the results, and as shown in the table, the quantitative values are validated.

Table 5.1: Peak value of Gaussian wave problem at $t = 2.0$

Scheme	Peak value
Barth & Jespersen limiter	0.8801
MLP-u1 limiter	0.9517
Data-driven reconstruction	0.9943

Table 5.2: Peak value of double sine wave problem at $t = 2.0$

Scheme	Peak value	
Barth & Jespersen limiter	-0.8045	0.8211
MLP-u1 limiter	-0.9307	0.9448
Data-driven reconstruction	-0.9917	0.9957

5.2 Shock-tube Problems

These problems are designed to assess the capability of resolving different types of linear and nonlinear waves on unstructured grids. The computational domain covers the range $[0,1]$ in the x-direction and $[0,0.1]$ in the y-direction. The simulation utilizes a grid composed of irregular triangles, as shown in Figure 5.8. This grid configuration consists of a total of 101 vertices in the x-direction and 11 vertices in the y-direction.

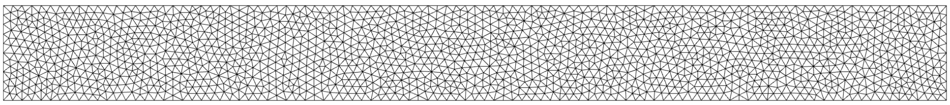


Figure 5.8: Irregular triangular mesh (100×10)

For the calculation of the numerical flux, a local Lax-Friedrichs (LLF) scheme was employed. The horizontal direction boundary conditions were implemented using periodic boundaries, while the vertical direction boundary conditions were applied using supersonic outflow conditions.

Two Riemann-type initial conditions are considered, described by Eq. 5.6 and 5.7.

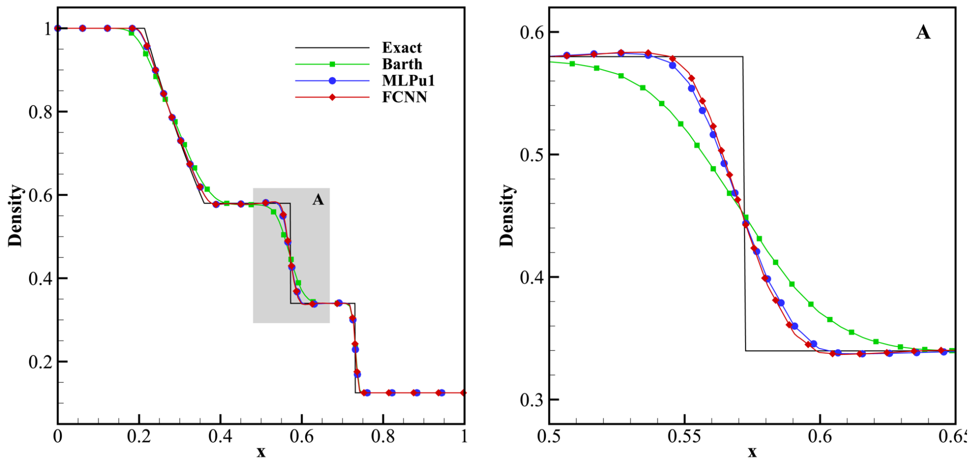
1. Modified Sod problem:

$$(\rho_L, u_L, v_L, p_L) = (1,0,0,1),$$
$$(\rho_R, u_R, v_R, p_R) = (0.125,0,0,0.1) \qquad \text{Eq. 5.6}$$

2. Supersonic expansion problem:

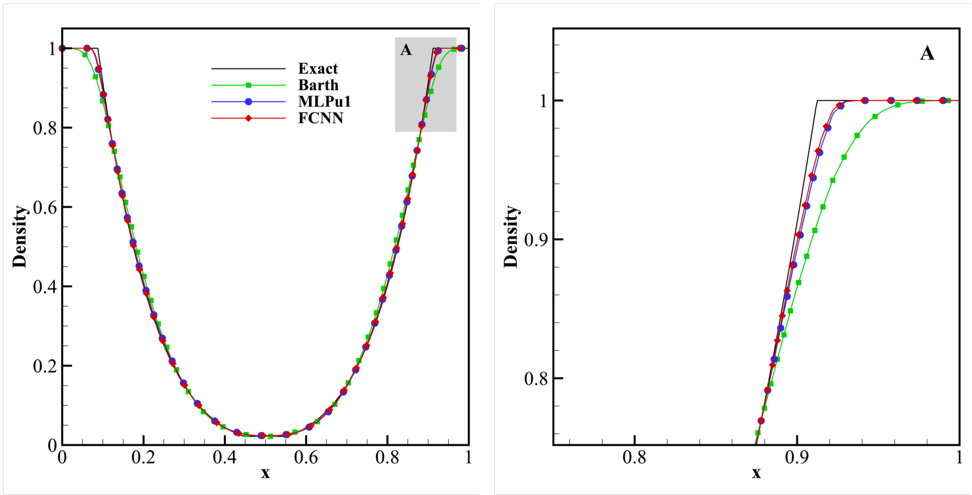
$$(\rho_L, u_L, v_L, p_L) = (1.0,-2.0,0.0,0.4),$$
$$(\rho_R, u_R, v_R, p_R) = (1.0,2.0,0.0,0.4) \qquad \text{Eq. 5.7}$$

Figure 5.9 and Figure 5.10 present the density distribution for each problem. The overall comparison confirms the robustness and accuracy of the data-driven reconstruction method. Notably, in the contact region as shown in Figure 5.9-(b), the method slightly captures the solution more sharply compared to the other limiters. Additionally, the method is tested for the rotational symmetry condition through the supersonic expansion problem, as depicted in Figure 5.10. The results show that the condition is indeed satisfied, which can be attributed to the proper ordering of input features in our method.



(a) Whole region (b) Closed-up view of A region

Figure 5.9: Density distributions along the center line of modified Sod problem



(a) Whole region (b) Closed-up view of A region

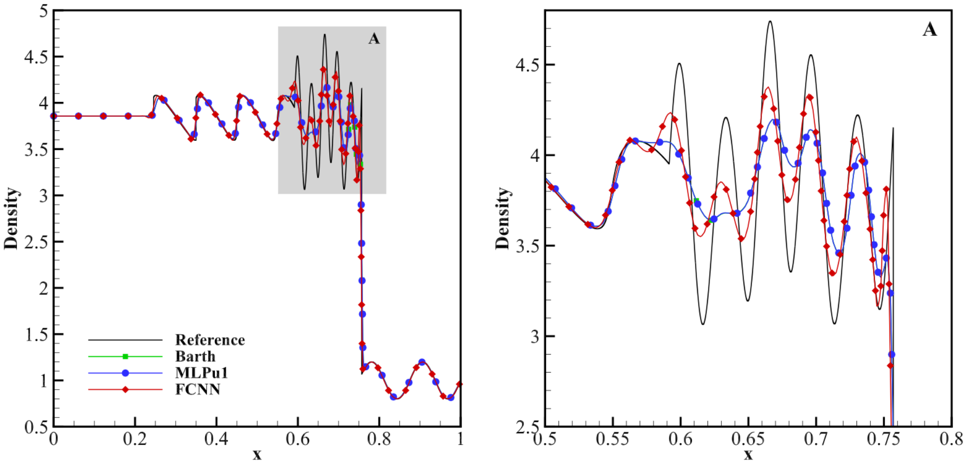
Figure 5.10: Density distributions along the center line of supersonic expansion problem

5.3 Shock and Entropy Wave Interaction Problem

The Shu-Osher problem [25], a standard benchmark problem, is employed to evaluate the performance of high-resolution schemes in handling the interaction between a shock wave and an entropy wave. The computational domain spans $[0,1]$ in the x-direction and $[0,0.04]$ in the y-direction. For this test, two types of grids are used: a regular quadrilateral grid and an irregular triangular grid, both discretized with dimensions of 400×4. The convective numerical flux is computed using the local Lax-Friedrichs (LLF) scheme. This problem is essential to assess the accuracy and robustness of the data-driven reconstruction method in capturing shock and entropy waves.

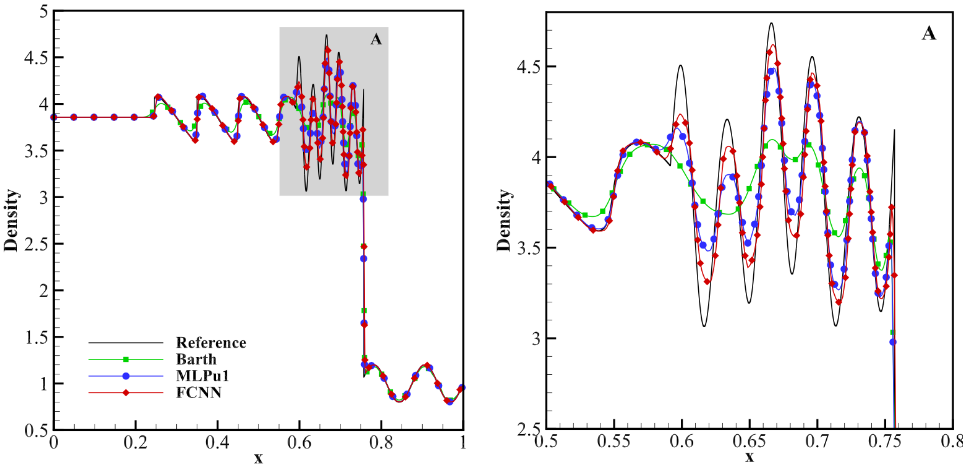
The initial conditions are as follows:

$$(\rho_L, u_L, v_L, p_L) = (3.857143, 2.629369, 0, 10.333333),$$
$$(\rho_R, u_R, v_R, p_R) = (1 + 0.2\sin(16\pi x), 0, 0, 1). \quad \text{Eq. 5.8}$$



(a) Whole region (b) Closed-up view of A region

Figure 5.11: Density distributions along the center line of Shu-Osher problem on RQ mesh



(a) Whole region (b) Closed-up view of A region

Figure 5.12: Density distributions along the center line of Shu-Osher problem on IT mesh

Figure 5.11 and Figure 5.12 display a comparison of the density distributions along the centerline using the Barth & Jespersen limiter and MLP-u1 limiter with the data-driven reconstruction method at $t = 0.178$. The reference solution is obtained using a first-order scheme with fine meshes containing $4{,}000$ points in the x-direction. The results highlight that the data-driven reconstruction method effectively suppresses unwanted oscillations. Specifically, the method accurately resolves the interaction, while the other limiters exhibit diffusivity at smooth extrema. These findings highlight the successful achievement of robustness and accuracy in both the quadrilateral and triangular models.

5.4 Double Mach Reflection Problem

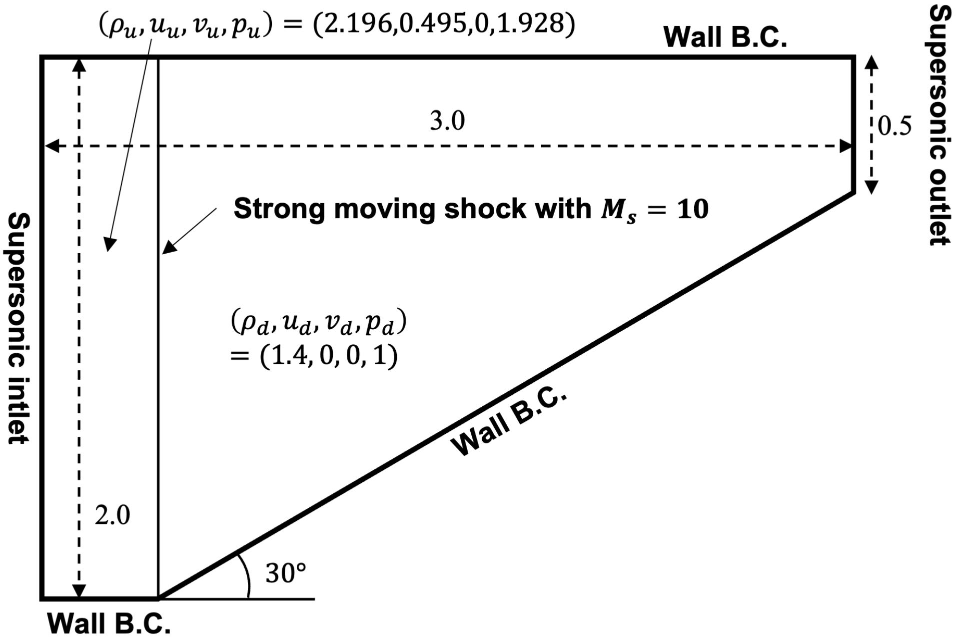
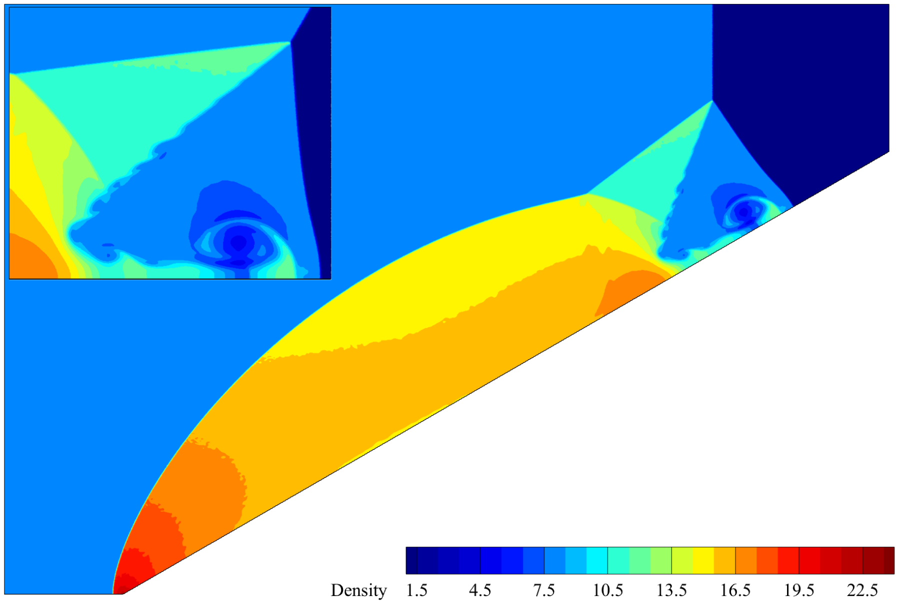


Figure 5.13: Initial/boundary condition of double Mach reflection problem

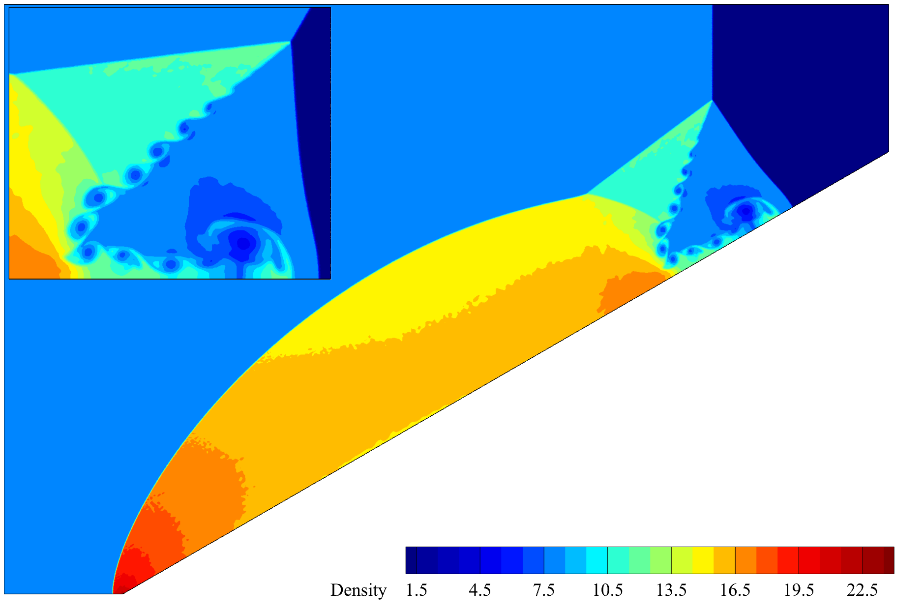
This benchmark problem [26] presents a highly strong moving shock wave with a Mach number of $M_s = 10$ impacting a $30°$ wedge. The intensity of the impinging shock and the presence of a Kelvin-Helmholtz instability along the contact discontinuity lead to intricate and complex flow interactions and phenomena. The collision of the shock wave with the wedge generates a variety of flow features, including turbulent shear layers, vortices, and shock-induced instabilities. As a result, this problem poses significant challenges for numerical simulations and requires robust and accurate shock-capturing methods to effectively capture and resolve these complex flow phenomena.

The downstream condition is defined as $(\rho, u, v, p)_d = (1.4, 0, 0, 1.0)$, and the upstream condition is computed using the moving shock relation with $M_s = 10$. The CFL number used in the simulation is 0.5, and the end time for the simulation is set to 0.2. The numerical flux scheme employed in the simulation is RoeM. The mesh used for the simulation is an irregular triangular (IT) mesh with a grid size of $h = 1/200$.

Figure 5.14 displays the density contour obtained using the MLP-u1 limiter and the data-driven reconstruction method. Both schemes successfully capture shock waves, but the MLP-u1 limiter overly diffuses the Kelvin-Helmholtz instability from the shock triple point. On the other hand, the data-driven reconstruction method effectively captures the small-scale vortices of the Kelvin-Helmholtz instability. Figure 5.15 displays the tagging contour for each primitive variable obtained by the tree model, and the results demonstrate its proper functioning.

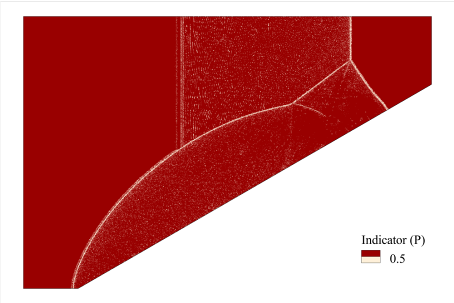


(a) Result of MLP-u1

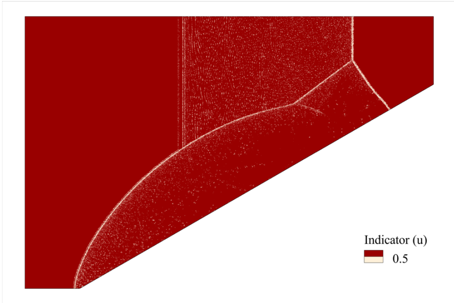


(b) Result of data-driven reconstruction

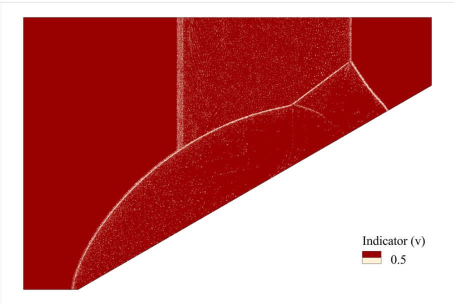
Figure 5.14: Density contour of double Mach reflection problem at $t = 0.2$



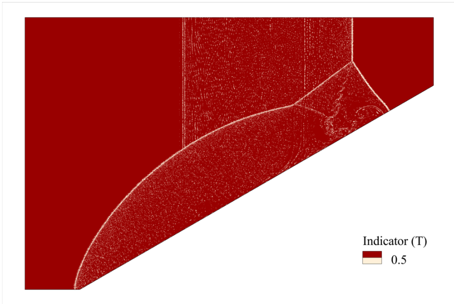
(a) Pressure



(b) Velocity (x component)



(c) Velocity (y component)



(d) Temperature

Figure 5.15: Indicating value contour of double Mach reflection problem at $t = 0.2$

5.5 Shock Wave and Wedge Interaction Problem

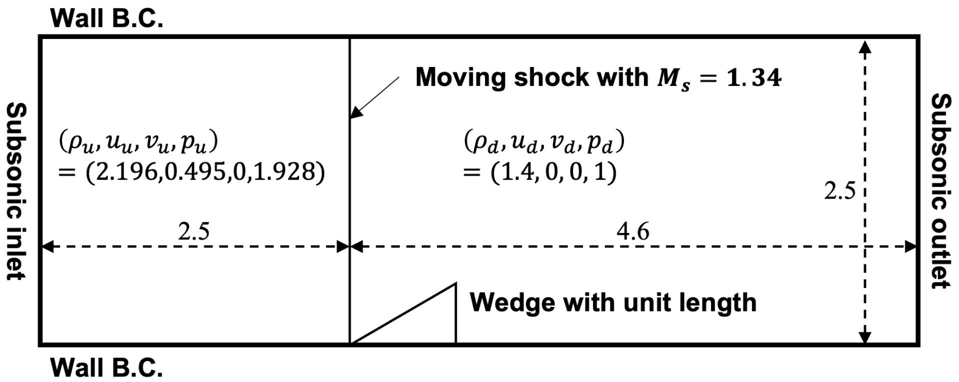
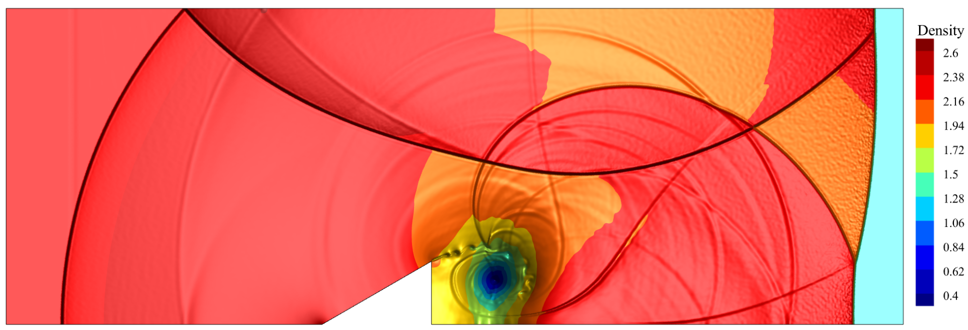


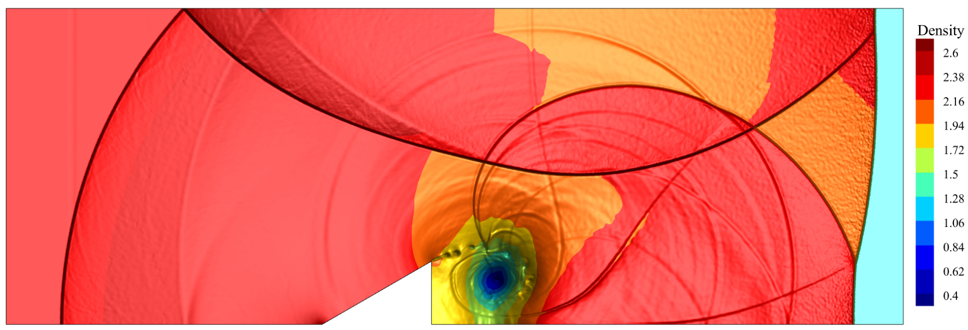
Figure 5.16: Initial/boundary condition of shock wave-wedge interaction problem

This benchmark problem [27] involves a planar moving shock with a Mach number of $M_s = 1.34$ impinging on a $60°$ finite wedge. The impinging shock undergoes reflection and diffraction, generating complex wave structures such as multiple Mach stems, reflected and scattered shocks, slip lines with a series of small vortices, acoustic waves, and shock-vortex interactions. These intricate flow features are ideal for evaluating the shock-capturing performance in high-speed unsteady flows.

The initial downstream condition is given by $(\rho, u, v, p)_d = (1.04, 0, 0, 1.0)$, and the upstream condition is computed from the moving shock relation with $M_s = 1.34$, resulting in $(\rho, u, v, p)_u = (2.196, 0.495, 0, 1.928)$. The numerical simulations are performed using an irregular triangular (IT) mesh with a grid spacing of $h = 1/200$. The CFL number used in the simulation is 0.5, and the end time for the simulation is set to 3.25. The numerical flux scheme employed in the simulation is RoeM.

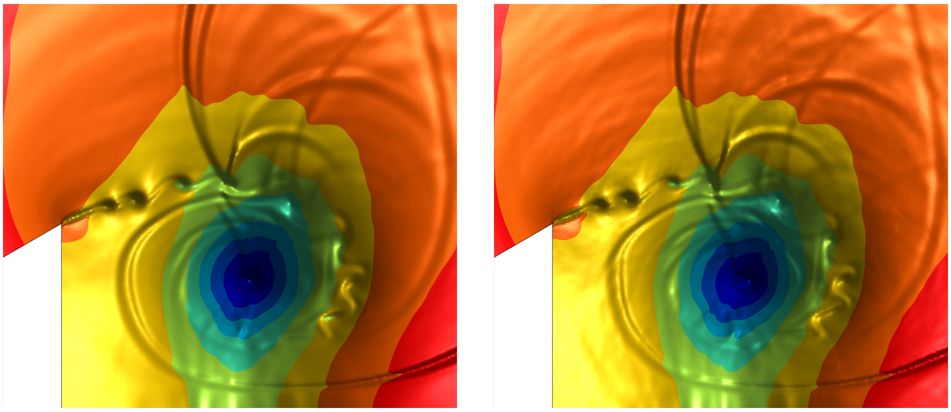


(a) MLP-u1 limiter



(b) Data-driven reconstruction method

Figure 5.17: Density contour of shock wave-wedge interaction problem at $t = 3.25$



(a) MLP-u1 limiter (b) Data-driven reconstruction method

Figure 5.18: Density contour of shock wave-wedge interaction problem at $t = 3.25$ (close-up view near the primary vortex)

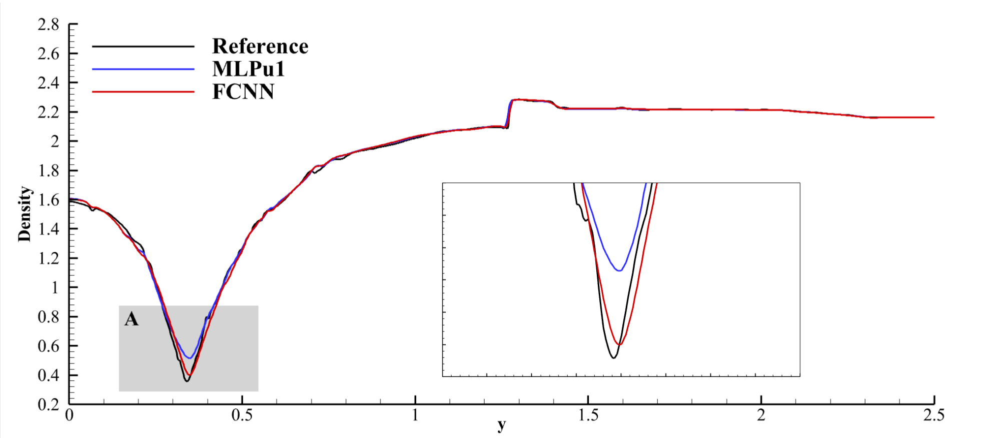


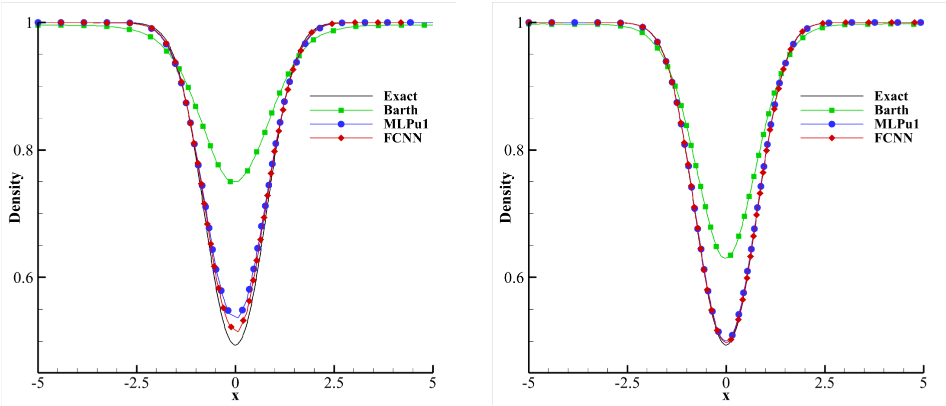
Figure 5.19: Density distributions of shock wave-wedge interaction problem along the primary vortex core at $t = 3.25$

Figure 5.17 and Figure 5.18 display the density contour obtained using the MLP-u1 limiter and the data-driven reconstruction method. Both schemes successfully capture the basic shock structure, and their performance in resolving local flow physics appears to be similar, particularly in terms of the Kelvin-Helmholtz instability from the edge corner and the downstream wave pattern. However, as shown in Figure 5.19, the density distribution along the primary vortex core differs between the two schemes. The MLP-u1 limiter exhibits more diffusion at the vortex core, while the data-driven reconstruction method is more accurate in capturing the vortex core. The reference solution used for comparison is obtained using SPID with a grid spacing of $h = 1/100$.

5.6 Stationary Isentropic Vortex Problem

This benchmark problem [28] aims to assess the accuracy of a numerical scheme in multi-dimensional flows without shock waves and turbulence. The problem involves inviscid vortex flow, where the flow field is inviscid, and the exact solution is the same as the initial condition. The free stream is $\rho_\infty = 1$, $p_\infty = 1$ and $(u_\infty, v_\infty) = (0{,}0)$. The perturbed values are given by $(\delta u, \delta v) = \frac{\epsilon}{2\pi} e^{0.5(1-r^2)}(-\bar{y}, \bar{x})$ and $\delta T = -\frac{(\gamma-1)\epsilon}{8\gamma\pi^2} e^{1-r^2}$. Here, the strength of the vortex is $\epsilon = 5$, and the vortex center is defined as $(\bar{x}, \bar{y}) = (x - x_0, y - y_0)$, where (x_0, y_0) represents the coordinates of the vortex core. In this specific problem, the vortex core is located at $(0{,}0)$. The parameter r^2 is given by $(\bar{x}^2 + \bar{y}^2)$, and γ represents the specific heat ratio. These perturbed values are essential for defining the initial conditions for the inviscid vortex flow problem. By using the equations $\rho = \rho_\infty + \delta\rho, u = u_\infty + \delta u, v = v_\infty + \delta v, T = T_\infty + \delta T$ and the isentropic relation, other physical variables can be obtained.

The computational domain spans $-5 \le x \le 5$ and $-5 \le y \le 5$, and periodic boundary conditions are applied. Irregular triangular meshes are discretized with $h = 1/40$ and $1/80$. The local Lax-Friedrichs flux is employed as the numerical flux for the simulations.



(a) 40×40 (a) 80×80

Figure 5.20: Density distributions of stationary isentropic vortex problem at $t = 10$

Figure 5.20 illustrates the comparison of density distribution across the vortex center. The results demonstrate the accurate characteristic of the data-driven reconstruction method. Particularly, in the coarse mesh, the accuracy is more evident, highlighting the method's effectiveness in capturing the flow features around the vortex core with better precision.

5.7 Viscous Shock-tube Problem

This test case [29] demonstrates the flow structure of a viscous flow involving the interaction between a shock wave, boundary layer, and vortex. The interaction occurs between the viscous boundary layer at the horizontal wall and the reflective shock wave from the vertical wall. Notably, a λ-shock wave and vortices can be observed in this problem.

The computational domain has a width of 1 length unit and a height of 0.5 length unit. The diaphragm is positioned at $x = 0.5$. The initial condition is defined by the following Eq 5.9.

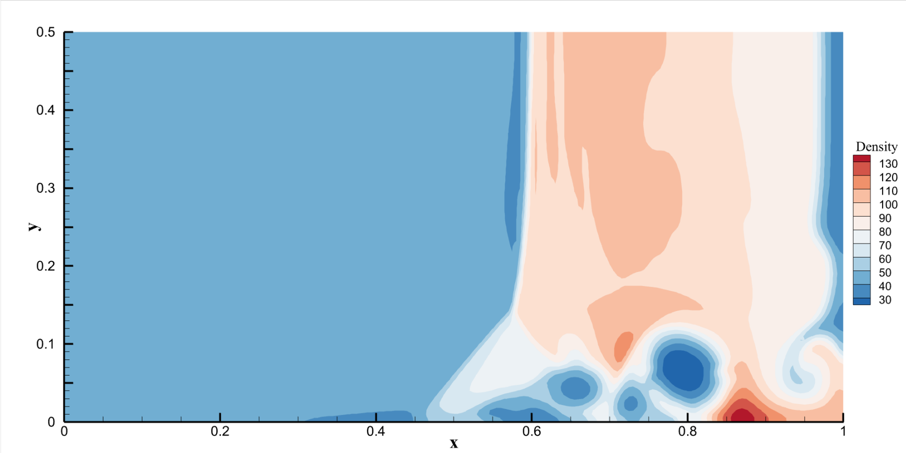
$$\begin{aligned}(\rho_L, u_L, v_L, p_L) &= (120, 0,0,120/\gamma),\\ (\rho_R, u_R, v_R, p_R) &= (1.2,0, 0, 1.2/\gamma).\end{aligned} \qquad \text{Eq. 5.9}$$

The Reynolds number for this case is 200, and the Prandtl number is 0.73. The inviscid flux is calculated using the RoeM scheme, while the discretization of the viscous flux follows the method described in [30]. A third-order accurate TVD Runge-Kutta method is applied to integrate the equations in time with a CFL number of 0.5.

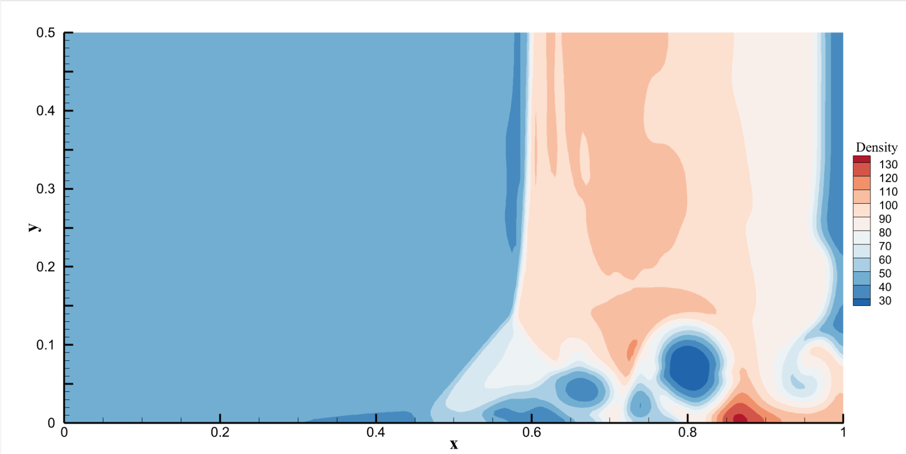
Figure 5.21 shows density contours computed by Barth and MLP limiter and data-driven reconstruction method at $t = 1$. Regular quadrilateral (RQ) grid with grid size $h = 1/200$ is considered. It is evident that the Barth limiter exhibits excessive numerical diffusion, leading to noticeably smeared (or less rotated) shapes of vortices. In comparison, the data-driven reconstruction method shows significantly reduced smearing, even when compared to the MLP limiter.

Table 5.3: Height of the primary vortex of viscous shock-tube problem

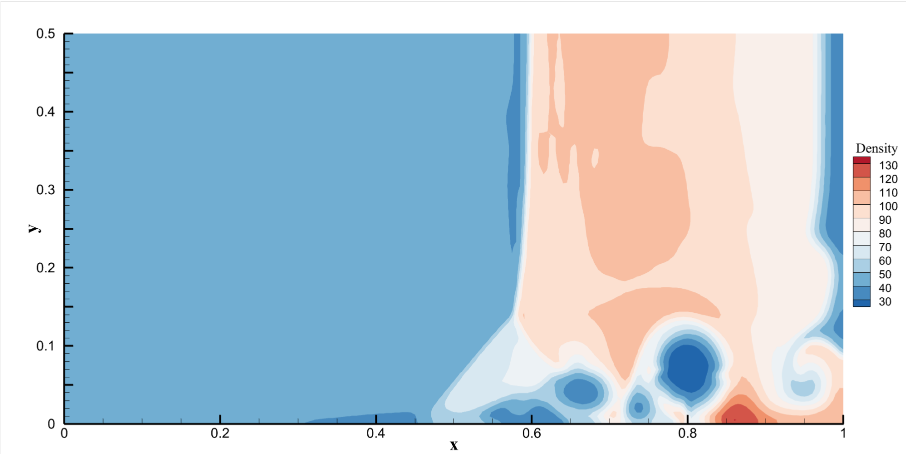
Scheme	Height
Barth & Jespersen limiter	0.127
MLP-u1 limiter	0.140
Data-driven reconstruction	0.143



(a) Barth limiter



(c) MLP limiter



(e) Data-driven reconstruction method

Figure 5.21: Density contour of viscous shock-tube problem at $t = 1$

Chapter 6 Conclusion

This study presents a groundbreaking data-driven approach for designing a new shock-capturing method. It represents the first attempt at developing a data-driven shock-capturing approach tailored for irregular meshes in the context of the Finite Volume Method (FVM). The proposed method possesses unique strengths, achieving both high robustness comparable to the MLP limiter and improved accuracy beyond the MLP limiter when utilizing only the MLP stencil.

The core of the method lies in the adoption of tree models, which effectively distinguish troubled cells, and the application of appropriate fully connected neural network (FCNN) models to reconstruct inner-face values of target cells. To train these models effectively, four distinct datasets are constructed, focusing on different elements and solution distributions. Specifically, the datasets comprise triangular elements with discontinuous or smooth solution distributions and quadrilateral elements exhibiting similar characteristics. The selection of proper input features is crucial for model performance, and the method skillfully utilizes only target-cell and face-neighboring cell information.

A notable advantage of this data-driven approach is its ability to operate without user-defined parameters once the model training is completed. Consequently, it offers objective flow modeling without the need for manual parameter tuning, streamlining the simulation process.

In terms of future work, the method will be extended to handle three-dimensional geometries encompassing various element types, such as tetrahedrons, hexahedrons, prisms, and pyramids. Additionally, efforts will be made to adapt the method to address steady-state problems, enabling efficient simulations of time-independent scenarios. Furthermore, the convergence performance of the method will be thoroughly assessed to ensure its reliability and applicability in various flow scenarios.

Overall, this study showcases a promising direction for advancing shock-capturing methods through the innovative use of data-driven techniques, empowering more accurate and efficient simulations in computational fluid dynamics (CFD).

Bibliography

[1] LeVeque, R. J. (2002). Finite volume methods for hyperbolic problems (Vol. 31). Cambridge university press.

[2] Shi, J., Zhang, Y. T., & Shu, C. W. (2003). Resolution of high order WENO schemes for complicated flow structures. Journal of Computational Physics, 186(2), 690-696.

[3] Godunov, S. K. (1959). A difference scheme for numerical solution of discontinuous solution of hydrodynamic equations. Math. Sbornik, 47, 271-306.

[4] Harten, A. (1997). High resolution schemes for hyperbolic conservation laws. Journal of computational physics, 135(2), 260-278.

[5] Shu, C. W. (1987). TVB uniformly high-order schemes for conservation laws. Mathematics of Computation, 49(179), 105-121.

[6] Shu, C. W., & Osher, S. (1988). Efficient implementation of essentially non-oscillatory shock-capturing schemes. Journal of computational physics, 77(2), 439-471.

[7] Liu, X. D., Osher, S., & Chan, T. (1994). Weighted essentially non-oscillatory schemes. Journal of computational physics, 115(1), 200-212.

[8] Barth, T., & Jespersen, D. (1989, January). The design and application of upwind schemes on unstructured meshes. In 27th Aerospace sciences meeting (p. 366).

[9] Venkatakrishnan, V. (1995). Convergence to steady state solutions of the Euler equations on unstructured grids with limiters. Journal of computational physics, 118(1), 120-130.

[10] Park, J. S., Yoon, S. H., & Kim, C. (2010). Multi-dimensional limiting process for hyperbolic conservation laws on unstructured grids. Journal of Computational Physics, 229(3), 788-812.

[11] Ray, D., & Hesthaven, J. S. (2018). An artificial neural network as a troubled-cell indicator. Journal of computational physics, 367, 166-191.

[12] Ray, D., & Hesthaven, J. S. (2019). Detecting troubled-cells on two-dimensional unstructured grids using a neural network. Journal of Computational Physics, 397, 108845.

[13] Yu, J., Hesthaven, J. S., & Yan, C. (2018). A data-driven shock capturing approach for discontinuous Galekin methods, EPFL.

[14] Beck, A. D., Zeifang, J., Schwarz, A., & Flad, D. G. (2020). A neural network based shock detection and localization approach for discontinuous Galerkin methods. Journal of Computational Physics, 423, 109824.

[15] Zeifang, J., & Beck, A. (2021). A data-driven high order sub-cell artificial viscosity for the discontinuous Galerkin spectral element method. Journal of Computational Physics, 441, 110475.

[16] Feng, Y., Liu, T., & Wang, K. (2020). A characteristic-featured shock wave indicator for conservation laws based on training an artificial neuron. Journal of Scientific Computing, 83, 1-34.

[17] Feng, Y., & Liu, T. (2021). A characteristic-featured shock wave indicator on unstructured grids based on training an artificial neuron. Journal of Computational Physics, 443, 110446.

[18] Discacciati, N., Hesthaven, J. S., & Ray, D. (2020). Controlling oscillations in high-order discontinuous Galerkin schemes using artificial viscosity tuned by neural networks. Journal of Computational Physics, 409, 109304.

[19] Quinlan, J. R. (1986). Induction of decision trees. Machine learning, 1, 81-106.

[20] Breiman, L. (2001). Random forests. Machine learning, 45, 5-32.

[21] Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. Annals of statistics, 1189-1232.

[22] Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. Annals of statistics, 1189-1232.
[23] Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794).
[24] Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. Mathematics of control, signals and systems, 2(4), 303-314.
[25] Shu, C. W., & Osher, S. (1988). Efficient implementation of essentially non-oscillatory shock-capturing schemes. Journal of computational physics, 77(2), 439-471.
[26] Woodward, P., & Colella, P. (1984). The numerical simulation of two-dimensional fluid flow with strong shocks. Journal of computational physics, 54(1), 115-173.
[27] Schardin, H. (1957). High frequency cinematography in the shock tube. The Journal of Photographic Science, 5(2), 17-19.
[28] Hu, C., & Shu, C. W. (1999). Weighted essentially non-oscillatory schemes on triangular meshes. Journal of Computational Physics, 150(1), 97-127.
[29] Daru, V., & Tenaud, C. (2000). Evaluation of TVD high resolution schemes for unsteady viscous shocked flows. Computers & fluids, 30(1), 89-113.
[30] Frink, N. T. (1998). Tetrahedral unstructured Navier-Stokes method for turbulent flows. AIAA journal, 36(11), 1975-1982.

국문 초록

기존의 수학적 배경을 바탕으로 수행된 많은 연구에도 불구하고, 전산유체역학에서 강건하고 정확한 충격파 포착 기법을 개발하는 것은 어려운 과제이다. 이를 해결하기 위하여 본 연구에서는 기존의 수학적 접근 방법이 아닌 새로운 데이터 기반 접근 방법을 사용하여 유한체적법에서의 강건하고 정확한 재구성 기법을 개발하였다. 특히 해당 기법은 유동 영역을 트리 모델을 사용하여 불연속 및 연속 영역으로 분할한 뒤, 각 영역에 대해 알맞은 완전 연결 신경망 모델을 사용하여 불연속 영역에서는 충격파를 강건하게 포착하고 연속 영역에서는 높은 정확도를 확보하고자 한다.

이러한 모델들을 훈련시키기 위해 임의의 해석함수를 활용하여 불연속 유동과 연속 유동을 나타내는 두 가지 유형의 데이터셋을 구축하였으며, 적절한 입력변수를 정의하여 비정렬 격자계에 효율적으로 적용할 수 있도록 하였다. 또한 제안된 방법의 강건성과 정확성을 검증하기 위해 광범위한 수치 시험을 수행하였다.

최종적으로 본 연구는 데이터 기반의 접근 방법을 활용하여 복잡한 유동 해석의 정확성과 강건성을 향상시키는 잠재력을 강조하며, CFD에서 더 효과적인 충격파 포착 방법을 개발하는 데 대한 새로운 가능성을 제시한다.

주요어: 전산유체역학, 유한체적법, 기계학습, 완전 연결 신경망, 트리 모델, 재구성 기법

학번: 2021-22136