Controlling oscillations in high-order Discontinuous Galerkin schemes using artificial viscosity tuned by neural networks

Niccolò Discacciati[a,*], Jan S. Hesthaven[a], Deep Ray[a]

[a]*Institute of Mathematics*
École Polytechnique Fédérale de Lausanne (EPFL)
CH-1015 Lausanne, Switzerland

Abstract

High-order numerical solvers for conservation laws suffer from Gibbs phenomenon close to discontinuities, leading to spurious oscillations and a detrimental effect on the solution accuracy. A possible strategy to reduce it comprises adding a suitable amount of artificial dissipation. Although several viscosity models have been proposed in the literature, their dependence on problem-dependent parameters often limits their performances. Motivated by the objective to construct a universal artificial viscosity method, we propose a new technique based on neural networks, integrated into a Runge-Kutta Discontinuous Galerkin solver. Numerical results are presented to demonstrate the performance of this network-based technique. We show that it is able both to guarantee optimal accuracy for smooth problems, and to accurately detect discontinuities, where dissipation has to be injected. A comparison with some classical models is carried out, showing the superior performance of the network-based model in capturing both complex and fine structures.

Keywords: Conservation laws, Discontinuous Galerkin, Artificial viscosity, Artificial neural networks

1. Introduction

Numerical accuracy is a key property demanded in various fields of computational science, e.g. mechanics or fluid dynamics [38]. This translates to a strict control of the error of the associated discretization schemes, leading to an increasing research activity in the development of high-order methods. The popularity of Discontinuous Galerkin (DG) schemes [19] has significantly increased, due to their high potential in terms of geometric flexibility, high-order accuracy, conservation of physical properties, and parallel efficiency. On the other hand, hyperbolic conservation laws, such as Euler equations or the equations of magnetohydrodynamics, constitute a challenging class of problems, as solutions might be discontinuous even for smooth initial data [32]. In such regions regularity is lost, resulting in the introduction of the Gibbs phenomenon [13], i.e. the emergence of spurious numerical oscillations close to discontinuities. As this leads to inaccurate and even unstable numerical results, high-order schemes need to be corrected to reduce its effect.

*Corresponding author
Email addresses: niccolo.discacciati@epfl.ch (Niccolò Discacciati), jan.hesthaven@epfl.ch (Jan S. Hesthaven), deep.ray@epfl.ch (Deep Ray)

Several approaches have been proposed to tackle this issue [17], among which a popular family is slope limiting [5, 6, 35]. However, limiters can severely affect the global solution accuracy, and the computational cost may increase due to improper parameter selection. An alternative technique is based on weighted essentially non-oscillatory (WENO) reconstruction [30, 40]. Despite maintaining high-order accuracy, the computational cost can remain high. Both these strategies are based on the identification of the troubled cells, i.e. the mesh elements where the solution loses regularity [31].

An alternative family of stabilization methods consists in adding artificial dissipation to the problem. Ideally, a locally varying viscosity can be injected in each cell, retaining high order accuracy in the presence of smooth solutions. In this framework, several approaches have been proposed to choose the amount of artificial viscosity, based on a common theme of constructing a sensor capable of measuring the regularity of the solution. One idea relies on the estimation of the decay rate of the modal coefficients, focusing on the highest modes [29] or using a least-squares approach [21]. Another strategy exploits the entropy production, using the local residual of an entropy equation to estimate the artificial dissipation [14, 41, 39]. Other artificial viscosity models construct the dissipation using different strategies, e.g., first order differential operators [27, 28, 39], the local residual of the equation [15], or the solution jump at the element boundaries [3]. The main drawback of these approaches is their dependence on empirical parameters, the choice of which might have an impact on stability, accuracy, and robustness of the underlying numerical method. Since there is no rule to estimate the optimal values, their tuning is usually done in a problem-dependent, time-consuming way. Therefore, the applicability of these shock capturing models may be limited.

In this work, we propose an alternative technique based on artificial neural networks (ANNs), which can be interpreted as parameter-free, universal black-boxes used to predict a (pseudo-)optimal artificial dissipation. ANNs are computing models which are capable of approximating a function exhibiting high degrees of complexity and nonlinearity. After training them using a given dataset, they are able to predict the output for samples outside the training set. One of the main advantages of ANNs lies in the high computational efficiency, independently of the degree of complexity of the underlying input-output mapping. For this reason, they are widely used in areas like image processing, voice recognition, forecasting, medical diagnosis, etc. Other recent applications include solving partial differential equations [33], reduced order modeling [18] or the identification of troubled cells in limiting-based methods [31].

The main goal of this paper is to explain the design, training and application of ANNs to predict the artificial viscosity. We focus on a simple yet powerful ANN architecture, named multilayer perceptron (MLP), where neurons are grouped in layers, processing data from an input to an output layer. Although the complexity of such nets is rather low, they can be regarded as *universal function approximators* [7, 8]. The (expensive) offline training phase is performed using an appropriately designed dataset. The (cheap) online evaluation of the trained network is carried out at each iteration of the time-integration scheme for the particular problem being solved.

The rest of this paper is structured as follows. In Section 2 we introduce the Runge-Kutta DG formulation, and a few classical artificial viscosity models are discussed. In Section 3, we give an overview of the key ingredients to construct and train the ANNs. The network-based technique is presented, for a one- and a two-dimensional framework, in Sections 4 and 5 respectively, together with numerical results to demonstrate its accuracy. In Section 6 we briefly comment on the computational performances, while some concluding remarks are included in Section 7.

2. Numerical discretization

Let $\Omega \subset \mathbb{R}^d$ be a bounded domain and $T > 0$ be a fixed time instant. Consider the following conservation law

$$\frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot \boldsymbol{f} - \nabla \cdot \boldsymbol{g} = \boldsymbol{0} \qquad \forall\ (\boldsymbol{x}, t) \in \Omega \times [0, T], \tag{2.1}$$

where $\boldsymbol{u} \in \mathbb{R}^n$ is the vector of n conserved variables, $\boldsymbol{f} = \boldsymbol{f}(\boldsymbol{u}) \in \mathbb{R}^{n \times d}$ is the convective flux and $\boldsymbol{g} \in \mathbb{R}^{n \times d}$ is the viscous flux. Throughout this work, $\boldsymbol{g}$ serves only as an artificial diffusion term and takes the following form

$$\boldsymbol{g} = \mu\ \boldsymbol{q}, \quad \boldsymbol{q} = \nabla \boldsymbol{u}, \tag{2.2}$$

where $\mu = \mu(\boldsymbol{u})$ denotes the artificial viscosity coefficient. Equation (2.1) is completed by suitable boundary conditions on $\partial\Omega \times [0, T]$ and an initial condition $\mathbf{u}(\boldsymbol{x}, t = 0) = \mathbf{u}_0(\boldsymbol{x})$ in Ω.

2.1. Runge-Kutta Discontinuous Galerkin discretization

The spatial discretization is performed using a Discontinuous Galerkin (DG) scheme of order $m \geqslant 1$. We recall its main features for a scalar conservation law, with a straightforward extension to systems. After discretizing the domain using K non-overlapping elements as $\Omega = \bigcup_{k=1}^{K} D^k$, the solution u is approximated by

$$u(\boldsymbol{x}, t) \approx u_h(\boldsymbol{x}, t) = \bigoplus_{k=1}^{K} u_h^k(\boldsymbol{x}, t), \tag{2.3}$$

where $u_h^k \in P^m(D^k)$ is a polynomial of degree at most m in the element D^k. The *nodal* representation of the local solution, based on the Lagrange polynomials $\{l_i\}_{i=0}^{N_m}$, is given by

$$u_h^k(x, t) = \sum_{j=0}^{N_m} u_j^k(t) l_j^k(x) = \boldsymbol{u}^k \cdot \boldsymbol{l}^k, \tag{2.4}$$

referring to the nodal values as the local degrees of freedom. In (2.4), $N_m = \binom{m+d}{m}$ is the dimension of the local finite-dimensional space. Choosing $d = 2$ as a prototype case and dropping the superscript k, we define the local mass matrix as

$$\boldsymbol{M}_{ij} = \int_{D^k} l_j(x) l_i(x)\ dx, \quad i, j = 0, \dots N_m - 1, \tag{2.5}$$

the x- and y- advection matrices as

$$\boldsymbol{S}_{ij}^x = \int_{D^k} l_i(x, y) \frac{\partial}{\partial x}(l_j(x, y)), \quad \boldsymbol{S}_{ij}^y = \int_{D^k} l_i(x, y) \frac{\partial}{\partial y}(l_j(x, y)), \quad i, j = 0, \dots N_m - 1, \tag{2.6}$$

and the boundary mass matrix as

$$\boldsymbol{M}_{e,ij}^{\sigma} = \int_{\sigma_e} l_j(x) l_i(x)\ dx, \quad i = 0, \dots N_m - 1, j = 0, \dots N_E - 1, \tag{2.7}$$

where $N_E = m + 1$ denotes the number of points along the e-th edge. Assuming a nodal decomposition for the remaining physical variables, the weak DG formulation becomes

$$\boldsymbol{M}\frac{d\boldsymbol{u}}{dt} - (\boldsymbol{S}^x)^T\boldsymbol{f}^x - (\boldsymbol{S}^y)^T\boldsymbol{f}^y + (\boldsymbol{S}^x)^T\boldsymbol{g}^x + (\boldsymbol{S}^y)^T\boldsymbol{g}^y + \sum_{e=1}^{3}\boldsymbol{M}_e^\sigma\boldsymbol{f}_e^*\cdot\boldsymbol{n}_e - \sum_{e=1}^{3}\boldsymbol{M}_e^\sigma\boldsymbol{g}_e^*\cdot\boldsymbol{n}_e = \boldsymbol{0}, \quad (2.8a)$$

$$\boldsymbol{M}\boldsymbol{q}_x + (\boldsymbol{S}^x)^T\boldsymbol{u} - \sum_{e=1}^{3}\boldsymbol{M}_e^\sigma\boldsymbol{u}_e^* n_e^x = \boldsymbol{0}, \qquad \boldsymbol{M}\boldsymbol{q}_y + (\boldsymbol{S}^y)^T\boldsymbol{u} - \sum_{e=1}^{3}\boldsymbol{M}_e^\sigma\boldsymbol{u}_e^* n_e^y = \boldsymbol{0}. \quad (2.8b)$$

In order to ensure single-valued fluxes at the element boundaries, the physical fluxes are replaced by numerical approximations. Specifically, for $\boldsymbol{g}^*$ and $\boldsymbol{u}^*$ a centered flux is used, while for $\boldsymbol{f}^*$ a Rusanov flux is employed. Equations (2.8a) and (2.8b) are linked by

$$\boldsymbol{g} = \boldsymbol{\mu}\odot\boldsymbol{q}, \quad (2.9)$$

which identifies the pointwise multiplication between the nodal values of μ, collected in the vector $\boldsymbol{\mu}$, and the degrees of freedom of $\boldsymbol{q}$.

The adopted time-integration scheme is the *five-stage fourth-order low-storage explicit Runge-Kutta scheme* [4]. The timestep is adaptively chosen based on the following CFL condition, which takes into account both the convective and the viscous contributions

$$\Delta t = \frac{C}{\max_\Omega|\boldsymbol{f}'(\boldsymbol{u}_h)|\dfrac{m^2}{h} + \max_\Omega(\mu)\dfrac{m^4}{h^2}}. \quad (2.10)$$

Further details on the numerical setup can be found in [19, 10].

2.2. PDE models

In this work, different conservation laws are considered. One-dimensional problems are:

- The linear advection equation, characterized by $f(u) = \beta u$, where $\beta = \beta(x,t)$ is the transport field.
- Burgers' equation, with $f(u) = \frac{u^2}{2}$.
- The Buckley-Leverett problem, which describes the water saturation in a mixture of oil and water [25]. It is defined by choosing the non-convex flux function

$$f(u) = \frac{u^2}{u^2 + 0.5(1-u)^2}. \quad (2.11)$$

- The Euler system, defined by

$$\boldsymbol{u} = \begin{pmatrix}\rho\\ \rho v\\ E\end{pmatrix}, \quad \boldsymbol{f}(\boldsymbol{u}) = \begin{pmatrix}\rho v\\ \rho v^2 + p\\ v(E+p)\end{pmatrix}, \quad (2.12)$$

  where ρ is density, v is velocity, p is pressure and E is energy. The system is closed by the ideal gas law as

$$p = (\gamma-1)\left(E - \frac{1}{2}\rho|v|^2\right), \quad \gamma = \frac{7}{5}. \quad (2.13)$$

In a two-dimensional framework, denoting with $\boldsymbol{e}_x, \boldsymbol{e}_y$ the unit (row) vectors of the canonical basis of $\mathbb{R}^2$, we consider:

- The linear advection equation, defined by $\boldsymbol{f}(u) = f^x \boldsymbol{e}_x + f^y \boldsymbol{e}_y = (\beta^x u, \beta^y u)$.
- A two-dimensional extension of Burgers' equation, with $\boldsymbol{f}(u) = f^x \boldsymbol{e}_x + f^y \boldsymbol{e}_y = (\frac{u^2}{2}, \frac{u^2}{2})$.
- The KPP rotating wave problem [14], characterized by non-convex x and y fluxes

$$\boldsymbol{f}(u) = f^x \boldsymbol{e}_x + f^y \boldsymbol{e}_y = (\sin(u), \cos(u)). \tag{2.14}$$

- The Euler system, defined by

$$\boldsymbol{u} = \begin{pmatrix} \rho \\ \rho v_x \\ \rho v_y \\ E \end{pmatrix}, \quad \boldsymbol{f}(\boldsymbol{u}) = \begin{pmatrix} \rho v_x \\ \rho v_x^2 + p \\ \rho v_x v_y \\ v_x(E+p) \end{pmatrix} \boldsymbol{e}_x + \begin{pmatrix} \rho v_y \\ \rho v_x v_y \\ \rho v_y^2 + p \\ v_y(E+p) \end{pmatrix} \boldsymbol{e}_y, \tag{2.15}$$

  where $\boldsymbol{v} = (v_x, v_y)^T$ is the velocity field, and the pressure is defined in (2.13).

2.3. Artificial viscosity models

We briefly introduce the classical artificial viscosity models. They play a central role in the construction of the training samples, and constitute the primary tool to validate the proposed network-based technique. Unless explicitly required, the subscript h denoting finite-dimensional variables is dropped.

2.3.1. Overview

A first aspect to consider is the spatial *locality*, which results in a different amount of dissipation in each element. Viscosity should be injected only in the vicinity of discontinuities, without altering the inviscid scheme in the presence of smooth solutions. Moreover, an *optimal* amount of viscosity has to be chosen. Large values produce over-dissipative results, and significant features of the solution may not be captured. Conversely, small values might not be sufficient to eliminate the Gibbs oscillations, which might eventually lead to numerical instabilities. The adopted strategy to estimate the viscosity can be summarized as follows:

- In each cell, a maximum amount of viscosity is defined as

$$\mu_{max} = c_{max} \frac{h}{m} \max_{D^k} |\boldsymbol{f}'(\boldsymbol{u})|, \tag{2.16}$$

  where c_{max} is a problem-dependent global constant. Then, a shock sensor, i.e. a quantity which estimates the local smoothness of the solution, is evaluated, based on which a viscosity $\mu_{\mathcal{S}}$ is computed. The models provide a constant value in the given cell. Finally,

$$\mu = \min \{\mu_{\mathcal{S}}, \mu_{max}\} \tag{2.17}$$

  is set in each element.

- A global smoothing is performed for the viscosity [39]. This further reduces the numerical oscillations [2], while enhancing the viscosity sub-cell resolution. Even though sophisticated techniques have been developed [2, 1], in this paper we consider a simple three-step C^0 smoothing:
  1. Consider the set of vertexes and compute the viscosity average at each point across the cells sharing the same node.
  2. Compute the coefficients of the interpolating linear polynomial in each element.
  3. Evaluate this polynomial on the nodal points required by the discretization order m.

In this work, the viscosity is updated once per time step at the beginning of the Runge-Kutta internal loop, but one can consider a less frequent update.

2.3.2. Highest modal decay (MDH) model

This model, presented in [29], uses the decay of the modal coefficients as an indicator for discontinuous solutions. Considering a scalar problem, we rely on the modal expansion of the solution and define the truncated representation $\tilde{u}$ as the expansion containing only the first N_{m-1} terms. The shock sensor is defined as the fraction of energy of u of the highest modes, i.e.

$$S_k = \frac{(u-\tilde{u}, u-\tilde{u})^2_{L^2(D^k)}}{(u,u)^2_{L^2(D^k)}} = \frac{\|u-\tilde{u}\|^2_{L^2(D^k)}}{\|u\|^2_{L^2(D^k)}}. \tag{2.18}$$

Assuming that the modal coefficients satisfy a result analogous to Fourier series, for continuous solutions we expect the indicator S_k to scale as $1/m^4$. Defining $s_k = \log S_k$ and $s_0 = -c_A - 4\log m$, the viscosity coefficient is set as

$$\mu_{MDH} = \mu_{max} \begin{cases} 0 & \text{if } s_k < s_0 - c_\kappa, \\ \frac{1}{2}\sin\left(1 + \frac{\pi(s_k - s_0)}{2c_\kappa}\right) & \text{if } s_0 - c_\kappa \leqslant s_k < s_0 + c_\kappa, \\ 1 & \text{if } s_0 + c_\kappa \leqslant s_k, \end{cases} \tag{2.19}$$

where μ_{max} is defined in (2.16), and $c_A,\ c_\kappa$ are problem-dependent parameters. The extension to the Euler system is carried out by applying this previous framework using a *representative* variable of the problem. Following [29, 39], we choose the density ρ.

2.3.3. Averaged modal decay (MDA) model

This model can be interpreted as an improved version of the MDH technique [21]. In the one-dimensional case, we assume that

$$\log|\hat{u}_j| \simeq \log C - \tau \log j, \quad j = 1, \dots, N_m - 1 = m, \tag{2.20}$$

where both C and τ can be estimated in a least-squares sense by solving

$$\min_{C,\tau} \sum_{j=1}^{m} \left(\log|\check{u}_j| - (\log C - \tau \log j)\right)^2. \tag{2.21}$$

The coefficients $|\check{u}_j|$ in (2.21) are obtained after a modification of the modal coefficients $|\hat{u}_j|$, designed to reintroduce a sense of scale and a monotone decay. Using the optimal decay rate, the viscosity coefficient is defined as

$$\mu_{MDA} = \mu_{max} \begin{cases} 1 & \text{if } \tau < 1, \\ 1 - \frac{\tau - 1}{2} & \text{if } 1 \leqslant \tau < 3, \\ 0 & \text{if } 3 \leqslant \tau, \end{cases} \tag{2.22}$$

where μ_{max} is defined in (2.16). Since this approach is mostly designed for high-order discretization degrees, we employ it for $m \geqslant 3$ only. Th extension to the Euler system is again performed by applying this framework using the density.

In the two-dimensional scenario, a simple yet effective way to extend the previous reasoning is proposed in [39]:

1. Extract the nodal values for a prescribed scalar quantity along each edge e of the element.
2. Apply the one-dimensional approach on each edge, estimating the corresponding decay rates τ_e.
3. Identify the minimum decay rate $\tau = \min_e \tau_e$ and estimate the viscosity using (2.22).

2.3.4. Entropy viscosity (EV) model

The last model we consider is based on the behavior of the entropy [14, 41]. Let $(E(\boldsymbol{u}), \boldsymbol{F}(\boldsymbol{u}))$ be an entropy pair for the inviscid continuous problem [9]. We define

$$\mu_E = c_E \left(\frac{h}{m}\right)^2 B, \tag{2.23}$$

where c_E is an empirical parameter, while B is

$$B = \frac{1}{A} \max\left(\max_{D^k} |R(\boldsymbol{u})|,\ \max_{\partial D^k} |H(\boldsymbol{u})|\right), \tag{2.24}$$

where

$$R = \frac{\partial E}{\partial t} + \nabla \cdot \boldsymbol{F} \simeq \frac{E(\boldsymbol{u}^n) + E(\boldsymbol{u}^{n-1})}{\Delta t} + \frac{\nabla \cdot F(\boldsymbol{u}^n) + \nabla \cdot F(\boldsymbol{u}^{n-1})}{2}, \tag{2.25a}$$

$$H = \left(\frac{h}{m}\right)^{-1} [\![\boldsymbol{F}]\!] \cdot \boldsymbol{n}. \tag{2.25b}$$

The former is the residual of a space-time discretization of the entropy equation, with the time derivative discretized with a Crank-Nicolson scheme. The latter introduces the effect of the jump of the entropy flux across the element boundaries. Finally, the scalar A acts as a normalization, set as

$$A = \max_{\Omega} \left| E - \frac{1}{|\Omega|} \int_{\Omega} E d\Omega \right|. \tag{2.26}$$

Unlike the decay-based models, we explicitly enforce (2.17) as

$$\mu_{EV} = \min\left\{\mu_E, \mu_{max}\right\}. \tag{2.27}$$

3. Artificial neural networks

When constructing a new artificial viscosity model, the main features we seek to reach are:

- estimating a relation which exhibits a *high degree of complexity and nonlinearity*.
- creating a non-parametric, problem-independent 'black box', i.e. *universality*.
- *computational efficiency*.

Artificial Neural Networks (ANNs) are compatible with these requirements, shifting most of the complexity and the computational cost to an offline stage. On the other hand, the online phase primarily consists of low-cost matrix-vector multiplications. Essentially, ANNs are computational models which are able to process information, calibrated by learning from observational data [12, 16]. They are composed of simple units, named neurons, which are gathered together and which handle different signals. The way neurons are arranged and connected defines the topology of the network, i.e. its architecture [22, 12]. Here, we focus on *feedforward* neural networks, where the connections between the nodes do not form cycles and both the input and the output are fixed. Among them, *multilayer perceptrons* (MLPs) are the most common examples, in which the neurons are grouped in layers. The first one, made of N_I source neurons, is the *input layer*. It is followed by $L > 1$ *hidden layers*, made of N_H^l neurons ($l = 1 \dots L$). The *output layer* is made of N_O output neurons. Connections among the neural units are always directed from one layer to the next one, towards the output, as in Figure 1. Therefore, an MLP can be viewed as a map from the input to the output space

$$\boldsymbol{f} : \mathbb{R}^{N_I} \to \mathbb{R}^{N_O}, \quad \boldsymbol{x} \mapsto \boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}; (\boldsymbol{w}, \boldsymbol{b})), \tag{3.1}$$

which depends on multiple parameters, named *weights* and *biases* and denoted by $\boldsymbol{w}$, $\boldsymbol{b}$. The network

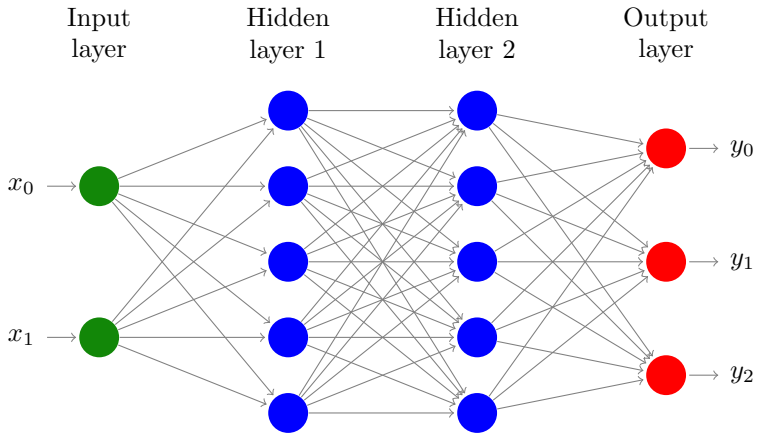


Figure 1: A graphical representation of a MLP with $N_I = 2$ input neurons, $L = 2$ hidden layers made of $N_H^1 = N_H^2 = 5$ neurons and an output layer with $N_O = 3$ neurons.

can be *trained* to accurately predict the responses within a given dataset, called the *training set* $\mathbb{T}$. Several learning strategies are available, among which a popular paradigm is *supervised learning*. The goal of the training algorithm is to minimize a *cost function* $\mathcal{C}$, i.e. a suitable measure of the error between the response of the network and the given target output. In many machine

learning applications, iterative procedures are used, with a single pass over the whole training set called *epoch*. If the network is properly trained, it is able to give good predictions even for input data outside $\mathbb{T}$. This property is known as *generalization*. We refer to [12, 16] for a more detailed description of ANNs.

4. One-dimensional networks

We now adapt MLPs to the artificial viscosity estimation problem. In the spirit of the standard models described in Section 2.3, we predict a local viscosity coefficient and perform a global smoothing afterwards. The procedure is described in the one-dimensional case first, focusing on scalar problems. The extension to systems is treated as in the classical techniques, selecting a representative scalar variable (e.g. density) as the predictor.

4.1. A one-dimensional model

We build a *family* of neural networks with one network for each degree m. This technique exploits all the available degrees of freedom in a mesh element, taking into account all the solution features.

Our approach consists of providing the local nodal values of the solution u as input to the ANN, predicting as output the viscosity coefficient μ. However, the performance of the optimization algorithm and the generalization properties are enhanced if scaled variables are employed. Hence, given the degrees of freedom of the numerical solution, we define the input of the ANN as

$$x_i = \frac{u_i}{U} = \frac{u_i}{\max |u_i| + \delta}, \quad i = 0, \dots, N_m - 1, \tag{4.1}$$

where $\delta = 10^{-8}$ is a small parameter used to avoid division by zero. Thus, each component lies in $[-1, 1]$. Conversely, the output is defined as

$$y_i = \frac{\mu_i}{\mathcal{H}\ \Lambda}, \quad i = 0, \dots, N_m - 1, \tag{4.2}$$

where $\Lambda = \max_{D^k} |f'(u)|$ is the maximum local wave speed, and $\mathcal{H} = h$ is a measure of the mesh size. This procedure is sketched Figure 2. In the construction of the training set, we define the target output as

$$\hat{\boldsymbol{y}} = \bar{y}\ \mathbb{1}_{N_m}, \tag{4.3}$$

where $\bar{y}$ is the scalar coefficient predicted by one of the artificial viscosity models (scaled using (4.2)), and $\mathbb{1}_{N_m}$ is the vector of ones in $\mathbb{R}^{N_m}$. In principle, the output vector $\boldsymbol{y}$ for a general input need not be of the form (4.3). However, we observed that $\boldsymbol{y}$ resembles (4.3), up to a small variance. In this work, we take the scaled viscosity coefficient in each element to be the maximum component of $\boldsymbol{y}$. Finally, this coefficient is rescaled using (4.2), and then smoothened as depicted in Figure 3.

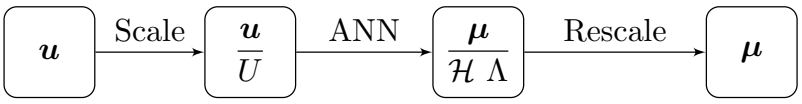


Figure 2: A graphical representation of the strategy used to build the network.

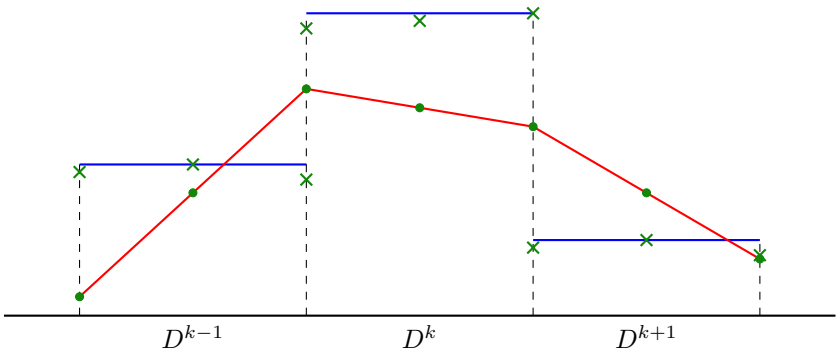


Figure 3: A graphical representation of the viscosity profile before and after smoothing, with $m = 2$ in a one-dimensional framework. The green crosses denote the pointwise values obtained with the network. The blue lines denote the element-wise constant values. The continuous red line represents the viscosity profile after the smoothing process. The green dots represent the final nodal viscosity values at the quadrature points.

Dealing with a regression problem, a natural choice to measure the prediction error is the Euclidean norm. We also add a *regularization* term to the cost function [23], to avoid *overfitting* the training set. Indeed, it may happen that the model performs very well on the training set but fails to generalize well to other data. A simple strategy consists in a penalization for the weights [12], hence

$$\mathcal{C} = \mathcal{C}_{err} + \mathcal{C}_{reg} = \frac{1}{2N_{\mathbb{T}}} \sum_{i=1}^{N_{\mathbb{T}}} \|\boldsymbol{y}_i - \hat{\boldsymbol{y}}_i\|_{l^2}^2 + \frac{\beta}{2} \|\boldsymbol{w}\|_{l^2}^2 = \frac{1}{2N_{\mathbb{T}}} \sum_{i=1}^{N_{\mathbb{T}}} \sum_{j=0}^{N_O-1} (y_{ij} - \hat{y}_{ij})^2 + \frac{\beta}{2} \sum_{(i,j)\in\mathcal{N}} w_{ij}^2, \quad (4.4)$$

where β is a non-negative coefficient set to 10^{-5}.

The same activation functions are chosen for all the neurons belonging to the same layer. For the hidden units, we use the *leaky rectified linear unit (leaky ReLU)* function [26]

$$f_{LReLU}(t) = \max\{x, 0\} - \alpha \max\{-x, 0\} = \begin{cases} x & \text{if } x \geqslant 0, \\ \alpha x & \text{if } x < 0, \end{cases} \quad (4.5)$$

where α is a (small) nonnegative coefficient, here set to 10^{-3}. In addition to a fast computation of its derivative, it avoids the problem of vanishing gradients and *dying neurons* [31, 26]. However, the leaky ReLU is not a viable option for the output layer. Since the output of the network is proportional to the dissipation coefficient, negative values are not admissible. Even though all the target outputs satisfy the non-negativity constraint, there is no guarantee that this property is preserved outside the training set, unless explicitly enforced. Using the ReLU function, i.e. (4.5) with $\alpha = 0$, the vanishing gradient problem still emerges. Hence, for the output layer we adopt the *softplus (SP)* function [11]

$$f_{SP}(t) = \ln(1 + e^x), \quad (4.6)$$

which can be interpreted as a smooth positive approximation of the ReLU.

The choice of the number of hidden layers and neurons therein has a considerable influence on the performances of the model, especially in terms of accuracy and computational efficiency. Within our framework, a good choice is given by $L = 5$ hidden layers made of $N_H^l = 10$ $(l = 1, \dots, L)$

neurons, independently of m. The adopted minimization algorithm is the *Adam optimizer* [20] with a learning rate of $\eta = 10^{-3}$, enhanced with mini-batch optimization. The procedure is run for a prescribed number of epochs $N_{epochs} = \mathcal{O}(10^3)$. A validation set $\mathbb{V}$, generated in a way similar to the training set, is also used to guarantee that overfitting does not occur.

4.2. Training and validation sets

The training and validation sets are constructed as follows:

1. Select a specific problem (flux function, initial condition, mesh size, polynomial degree, etc.) to be solved.
2. Determine the *best* artificial viscosity model. To do this, we analyze each technique (MDH, MDA, EV) separately and choose the parameters which give the best solution at the final time. Next, we compare the obtained results and pick the optimal model. The optimization is performed according to the following criteria:
   - The amplitude of overshoots and undershoots with respect to a reference solution must be reduced. The comparison is done visually, and it is benchmarked by numerically evaluating their magnitude.
   - The injected dissipation must maintain the solution features intact and avoid over-dissipation. In most of the cases, this is evaluated visually.

   Unless an exact solution is available, the reference solution is computed with one of the viscosity models using high-order polynomials and a very fine mesh. Here, we assume that the optimal model and parameters do not change in time.
3. Repeat steps 1 and 2 for different grid spacings h and initial conditions u_0, capturing spatial variability and different solution features (e.g. smooth solutions, points of non-differentiability, jumps).
4. Repeat step 3 for different conservation laws, in order to make the network capable of tracking, and possibly distinguishing between, shocks, rarefaction and contact waves. We consider Burgers' and the linear advection equations.
5. Repeat step 4 for different degrees m.

This procedure is followed by a post-processing phase:

- Create a *balanced* dataset. Simulations on finer meshes provide a larger number of samples in the set, due to an increasing number of elements and a smaller time step. Thus, a few samples have to be removed, in order to have roughly an equal contribution from all the simulations.
- Check for data *consistency*. If two (scaled) samples have the same input, then the corresponding output must be the same. If this is not the case, the training algorithm may fail to converge properly. In order to address this issue, we replace the inconsistent output values with their average.

Ultimately, the training and validation sets are generated. For each m, we randomly shuffle all post-processed samples. Then, we assign the first 70% of the data to the training and the remaining 30% to the validation set.

As a concrete example, consider $m = 4$ and Burgers' equation. Here, we select different initial conditions $u_0(x)$ and final simulation times T. For a fixed problem, multiple values of h are considered. In Table 1 we list the simulations, reporting the number of samples collected after the post-processing phase. The test cases are generated by considering the following initial conditions:

u_0	Ω	T	h	Best model	Nb. samples
u_0^1	$[0,2]$	0.15	2/40	MDH: $c_A = 2.2,\ c_\kappa = 0.2,\ c_{max} = 0.5$	10480
			2/80	MDH: $c_A = 2.2,\ c_\kappa = 0.2,\ c_{max} = 0.5$	10480
			2/120	MDH: $c_A = 2.4,\ c_\kappa = 0.2,\ c_{max} = 0.4$	10560
			2/200	MDH: $c_A = 2.2,\ c_\kappa = 0.2,\ c_{max} = 0.5$	10600
u_0^2	$[0,1]$	0.08	1/40	EV: $c_E = 1.2,\ c_{max} = 0.4$	7040
			1/80	EV: $c_E = 1.2,\ c_{max} = 0.4$	7040
			1/120	EV: $c_E = 1.0,\ c_{max} = 0.4$	7280
u_0^3	$[0,2]$	0.07	2/40	EV: $c_E = 1.8,\ c_{max} = 0.5$	10414
			2/80	EV: $c_E = 1.4,\ c_{max} = 0.5$	10560
			2/120	EV: $c_E = 1.5,\ c_{max} = 0.4$	10600
			2/200	EV: $c_E = 1.0,\ c_{max} = 0.6$	10667
u_0^4	$[0,1]$	0.07	1/40	EV: $c_E = 2.0,\ c_{max} = 0.8$	10827
			1/80	EV: $c_E = 1.8,\ c_{max} = 0.8$	10827
			1/120	EV: $c_E = 1.6,\ c_{max} = 0.8$	9740
u_0^5	$[0,1]$	0.03	1/40	MDH: $c_A = 2.5,\ c_\kappa = 0.3,\ c_{max} = 0.6$	11627
			1/80	MDH: $c_A = 2.0,\ c_\kappa = 0.3,\ c_{max} = 0.6$	11627
			1/120	MDH: $c_A = 2.0,\ c_\kappa = 0.3,\ c_{max} = 0.4$	10480
u_0^6	$[0,1]$	0.3	1/40	EV: $c_E = 1.5,\ c_{max} = 0.5$	13880
			1/80	EV: $c_E = 1.5,\ c_{max} = 0.6$	13920
			1/120	EV: $c_E = 1.5,\ c_{max} = 0.4$	12540
u_0^7	$[0,1]$	0.08	1/40	MDA: $c_{max} = 0.6$	7440
			1/80	MDA: $c_{max} = 0.6$	7440
			1/120	MDA: $c_{max} = 0.5$	6720

Table 1: Example on dataset generation, Burgers' equation, $m = 4$.

- $u_0^1(x) = 0.5 + 1 \cdot \mathbb{1}_{[0.1,0.25]}(x) + (x - 0.5) \cdot \mathbb{1}_{[0.5,1]}(x) + \sqrt{\frac{1}{4} - (x-1)^2} \cdot \mathbb{1}_{[1,1.5]}(x)$.
- $u_0^2(x) = -e^{-400(x-0.5)^2} \cdot \mathbb{1}_{[0.3,0.7]}(x)$.
- $u_0^3(x) = 20\ (0.5 - |x - 0.5|) \cdot \mathbb{1}_{[0,1]}(x)$.
- $u_0^4(x) = 10 \cdot \mathbb{1}_{[0,0.2]}(x) + 6 \cdot \mathbb{1}_{[0.2,0.4]}(x) + 0 \cdot \mathbb{1}_{[0.4,0.6]}(x) - 4 \cdot \mathbb{1}_{[0.6,1]}(x)$.
- $u_0^5(x) = \beta + (\alpha - \beta) \cdot \mathbb{1}_{[0.25,0.75]}(x)$, with $\alpha \in \{5, -1, 3\}$ and $\beta \in \{0, -3, -2\}$.
- $u_0^6(x) = \sin(2\pi x)$.
- $u_0^7(x) = \sin(4\pi x) \cdot \mathbb{1}_{[0.25,0.5]}(x) + \sin(8\pi x) \cdot \mathbb{1}_{[0.5,0.75]}(x)$.

4.3. An improved version

The technique described in Section 4.1 provides satisfactory results in the presence of discontinuous solutions, but performs poorly for smooth problems. Indeed, in the second scenario the injected viscosity should have the optimal scaling h^{m+1} [19]. This is not observed, since the scaling $\mathcal{H} = h$ in (4.2) adds an m-1 dissipation coefficient, preventing high-order accuracy for smooth solutions. Although the ANN recognizes smooth regions by adding low dissipation there, it never predicts a zero viscosity. Thus, the linear scaling forces a sub-optimal accuracy. To avoid such a behavior, an ideal strategy could use a troubled-cell indicator, like in [31].

However, we seek to show the potential of the model without relying on external tools. Since we can identify the scaling h to be the major source for low-order accuracy, we seek a factor satisfying

$$\mathcal{H} = \begin{cases} \mathcal{O}(h^{m+1}) & \text{if } u \text{ is smooth,} \\ \mathcal{O}(h) & \text{otherwise.} \end{cases} \tag{4.7}$$

A good parameter-free option is based on the solution jump. In particular, one may pick

$$\tilde{\mathcal{H}} = \max_{e \in \partial D^k} (|[\![u]\!]_e|) = \max \left(|u^-(x_{k-1}^r) - u^+(x_k^l)|, |u^-(x_k^r) - u^+(x_{k+1}^l)| \right). \tag{4.8}$$

The requirement (4.7) is then satisfied in the presence of smooth solutions, but a scaling of $\mathcal{O}(1)$ is observed close to shocks. The adopted scaling is thus defined as

$$\mathcal{H} = \min \left\{ \tilde{\mathcal{H}}, h \right\}. \tag{4.9}$$

4.4. Numerical results

In the following, we present extensive numerical results, for both scalar equations and systems, to demonstrate the capabilities of the ANN-based approach. We show that the networks outperform the standard models, as the latter are based on empirically tunable parameters. To enhance this, we keep the parameter values constant across all the simulations. The selected values, reported in Table 2, are chosen as a trade-off between accuracy and stability for all test cases [39]. Furthermore, we present the results with ANNs trained for $m = 1, 2, 3, 4$, although networks for higher degrees can be trained in a similar manner.

Model	1D problems
EV	$c_E = 1,\ c_{max} = 0.5$
MDH	$c_A = 2.5,\ c_\kappa = 0.2,\ c_{max} = 0.5$
MDA	$c_{max} = 1$

Table 2: Parameter values for the standard artificial viscosity models for 1D problems.

4.4.1. A smooth problem

The first test assesses the ability of the viscosity models to maintain the expected accuracy for a smooth solution. We consider the linear advection problem with a constant transport field $\beta = 1$. We choose $\Omega = [0, 1]$,

$$u_0(x) = 2 + \sin(2\pi x), \tag{4.10}$$

a CFL constant $C = 0.1$, $T = 0.2$ and periodic boundary conditions. The validation of the scheme relies on the $L^2(\Omega)$-norm of the discretization error vector evaluated at the final time

$$\epsilon = \|u_h(\cdot, T) - u(\cdot, T)\|_{L^2(\Omega)}\,. \tag{4.11}$$

Theoretical results for inviscid hyperbolic problems [19, 17] suggest that

$$\epsilon \leqslant C_1(m) h^{m+1}\,(1 + C_2(m)T) \sim h^{m+1} \tag{4.12}$$

if a Rusanov flux is used and u is sufficiently smooth. However, we are adding a dissipative term which might negatively impact on the accuracy. We report the convergence results in Table 3. As expected, the inviscid scheme achieves the optimal convergence rate. The EV scheme guarantees high accuracy as well, even though the error magnitude is larger when compared to the inviscid problem. A sub-optimal rate appears to be present for high orders ($m = 4$), possibly due to the Crank-Nicolson discretization of the time derivative in (2.25a) [41]. For low degrees ($m = 1$), the MDH model shows an order less than one. This is caused by the linear scaling with h, which prevents higher rates of convergence. For high degrees ($m \geqslant 2$), the shock sensor correctly detects the solution as regular and does not add dissipation, i.e. the inviscid scheme is retained. A similar reasoning holds for the MDA model, which is applied for $m \geqslant 3$ only. The ANN-based model guarantees optimal accuracy for all the orders of approximation. A slight irregularity from the optimal rate is observed for linear basis functions, particularly with coarse meshes, possibly due to an insufficient resolution. In most of the cases, the error magnitude is very similar to the inviscid scheme. Thus, at least for smooth problems, the scaling (4.9) guarantees the expected accuracy. We note that, even though the use of a centered flux for $\boldsymbol{g}^*$ and $\boldsymbol{u}^*$ in (2.8) can lead to an even/odd pattern of the convergence rates [19], this is not observed in our experiment.

4.4.2. Burgers' equation: a compound wave

We test the ability of our model to capture shocks, adding a spatially localized viscosity. A good starting point is provided by Burgers' equation. Since it was used as a prototype problem to generate the training set, the generalization properties of the network can be tested by considering a domain, initial condition and grid spacing not included in the generation of the dataset. Set $\Omega = [-4, 4]$, $K = 200$, $T = 0.4$ and $C = 0.1$. The chosen intial condition (Figure 4) consists of both

m	K	Inviscid		EV		MDH		MDA		ANN	
		ϵ	p	ϵ	p	ϵ	p	ϵ	p	ϵ	p
1	10	1.3386e−2	-	3.1848e−1	-	1.9595e−1	-			4.8874e−2	-
	20	3.3576e−3	1.99	8.9574e−2	1.83	1.1561e−1	0.76			1.1627e−2	2.07
	40	8.3953e−4	2.00	1.3176e−2	2.77	7.1512e−2	0.69			3.4402e−3	1.76
	80	2.0987e−4	2.00	2.1677e−3	2.60	4.2553e−2	0.75			5.7626e−4	2.58
	160	5.2465e−5	2.00	3.0662e−4	2.82	2.4201e−2	0.81			5.2756e−5	3.45
	320	1.3116e−5	2.00	4.1720e−5	2.88	1.3184e−2	0.88			1.3126e−5	2.00
2	10	1.0519e−3	-	6.2039e−2	-	1.0519e−3	-			1.0586e−3	-
	20	1.3298e−4	2.98	3.8533e−3	4.00	1.3298e−4	2.98			1.3383e−4	3.02
	40	1.6664e−5	3.00	2.9406e−4	3.71	1.6664e−5	3.00			1.6684e−5	3.00
	80	2.0844e−6	3.00	1.9935e−5	3.88	2.0844e−6	3.00			2.0854e−6	3.00
	160	2.6059e−7	3.00	1.3089e−6	3.92	2.6059e−7	3.00			2.6069e−7	3.00
	320	3.2575e−8	3.00	8.7767e−8	3.81	3.2575e−8	3.00			3.2587e−8	3.00
3	10	3.1021e−5	-	3.4710e−4	-	3.1021e−5	-	8.3751e−2	-	3.1263e−5	-
	20	2.2845e−6	3.76	1.3637e−5	4.67	2.2845e−6	3.76	4.3327e−2	0.95	2.2864e−6	3.77
	40	1.5260e−7	3.90	4.9179e−7	4.79	1.5260e−7	3.90	2.2260e−2	0.96	1.5268e−7	3.90
	80	9.3750e−9	4.02	1.7742e−8	4.79	9.3750e−9	4.02	1.1304e−2	0.99	9.3778e−9	4.03
	160	5.8609e−10	4.00	7.5682e−10	4.55	5.8609e−10	4.00	5.6980e−3	0.99	5.8621e−10	4.00
	320	3.6631e−11	4.00	4.0744e−11	4.22	3.6631e−11	4.00	2.8609e−3	0.99	3.6687e−11	4.00
4	10	9.9474e−7	-	1.4982e−4	-	9.9474e−7	-	9.9474e−7	-	9.9572e−7	-
	20	3.1481e−8	4.98	3.6170e−6	5.37	3.1481e−8	4.98	3.1481e−8	4.98	3.1487e−8	4.98
	40	1.0073e−9	4.97	1.1971e−7	4.92	1.0073e−9	4.97	1.0073e−9	4.97	1.0075e−9	4.97
	80	3.3036e−11	4.93	5.7002e−9	4.39	3.3036e−11	4.93	3.3036e−11	4.93	3.3040e−11	4.93
	160	1.0925e−12	4.92	3.3235e−10	4.10	1.0925e−12	4.92	1.0925e−12	4.92	1.0927e−12	4.92

Table 3: L^2 errors ϵ and estimated rate of convergence p in the inviscid case and with the artificial viscosity models for the linear advection problem.

smooth and discontinuous data [31]

$$u_0(x) = \begin{cases} \sin(\pi x) & \text{if } 1 \leqslant |x| \leqslant 4, \\ 3 & \text{if } -1 < x \leqslant -0.5 \text{ or } 0 < x \leqslant 0.5, \\ 1 & \text{if } -0.5 < x < 0, \\ 2 & \text{if } 0.5 < x \leqslant 1, \end{cases} \tag{4.13}$$

while the problem is completed with periodic boundary conditions. As the solution evolves, shocks

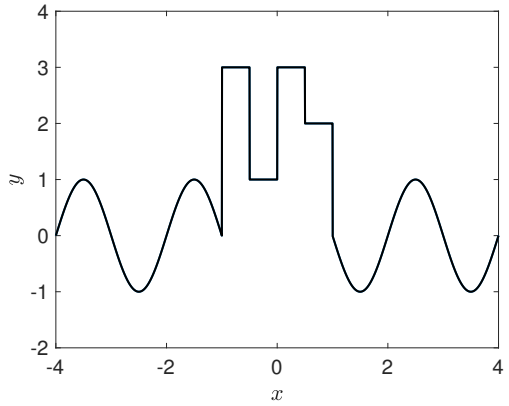
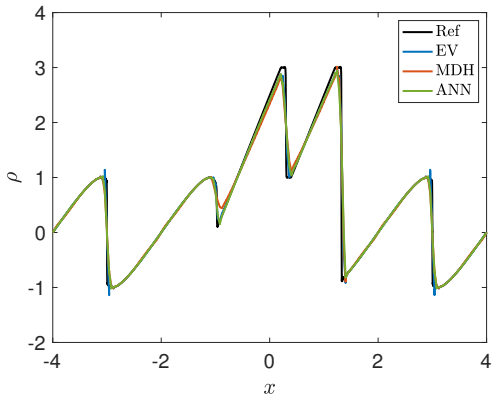


Figure 4: Initial condition for the compound wave problem, Burgers' equation.

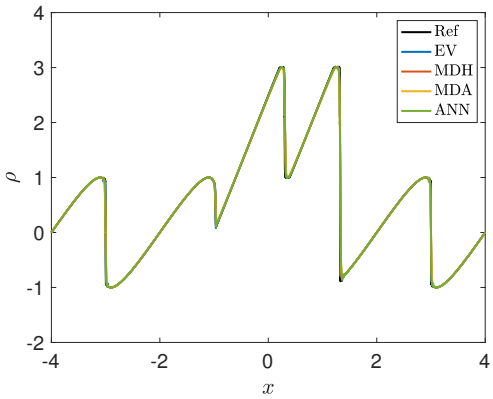
and rarefaction waves develop and interact. For the sake of simplicity, in Figure 5 we report the results only for $m = 1$ and $m = 4$, as prototype cases for low and high degrees. From a global perspective, all the models provide similar solutions. Looking at Figure 5(c) and Figure 5(e), we note that the EV model does not accurately smoothen the oscillations around $x = -3$, while the MDH produces an overshoot close to $x = 1$. The ANN solves both issues, reducing the amplitude of the wiggles while resolving the shocks. This reasoning is confirmed by looking at the temporal history of the artificial viscosity, shown in Figure 6. All techniques capture the regions where discontinuities develop. The EV model adds a slightly less localized viscosity, especially for small times, while the shocks around $x = \pm 3$ are not smoothed enough, resulting in larger oscillations in these regions. Both the decay-based models inject more viscosity, with the MDA appearing to be the most dissipative one. The ANN retains the good properties of the standard models, adding sufficient viscosity close to the shocks without resulting in an over-dissipative solution. For all cases, we observe an oscillatory behavior of μ with respect to time. Indeed, if the solution is sufficiently smooth, all models seek to avoid to inject dissipation. This will lead to oscillations, and once their amplitude becomes large, viscosity is added. This phenomenon is enhanced, close to the moving shocks, by the scaling (4.9).

4.4.3. Buckley-Leverett problem

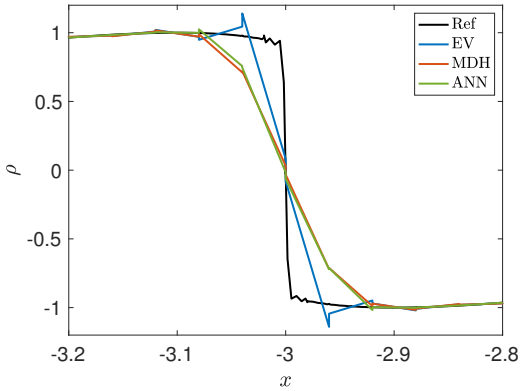
The main goal of this test is to demonstrate the performances of the ANN technique for flux functions not included in the training set. Following [31], we consider the Buckley-Leverett problem



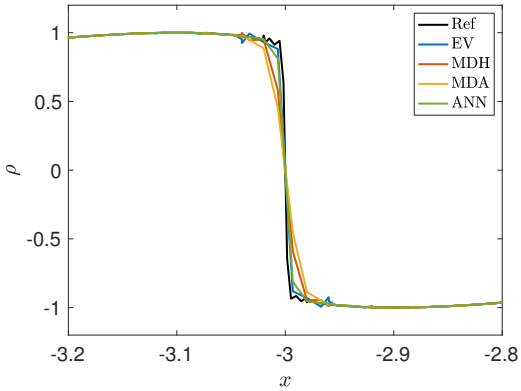
(a) Overall result, $m = 1$.



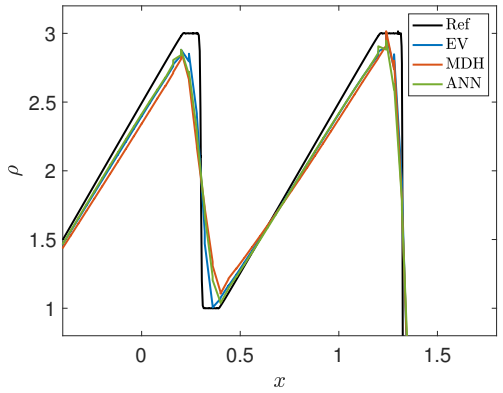
(b) Overall result, $m = 4$.



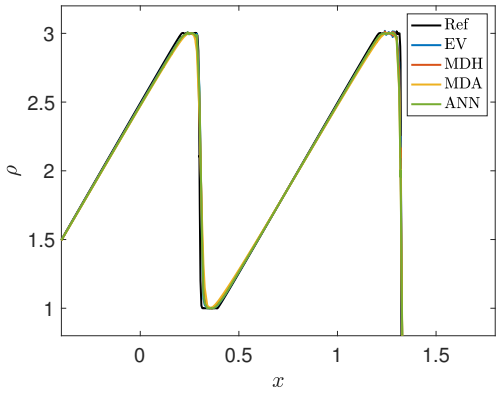
(c) Zoom close to the first shock, $m = 1$.



(d) Zoom close to the first shock, $m = 4$.

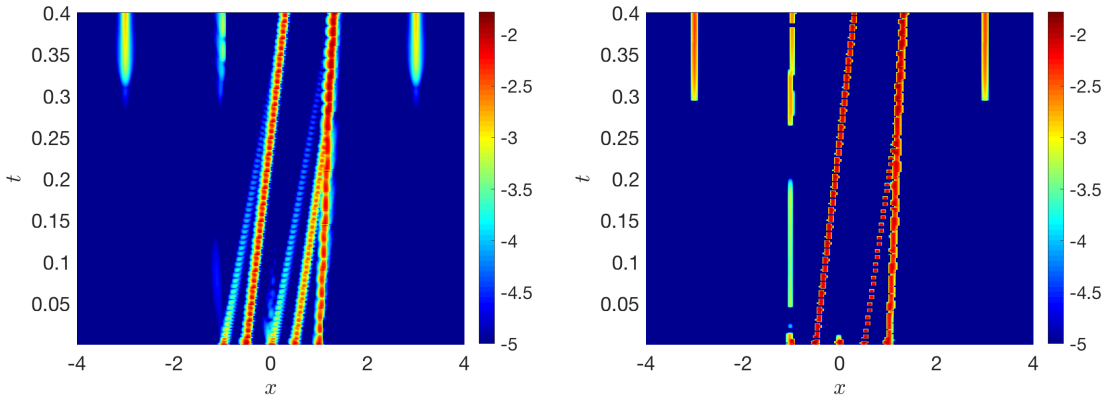


(e) Zoom close to the middle region, $m = 1$.



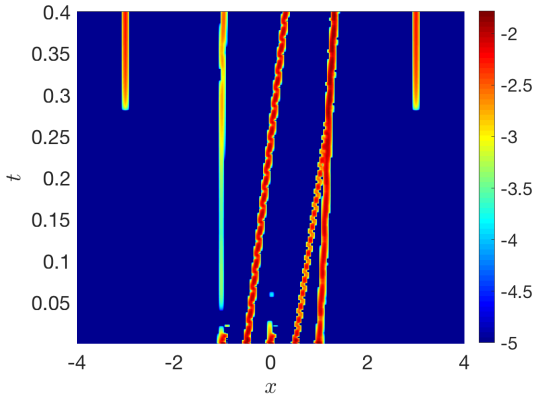
(f) Zoom close to the middle region, $m = 4$.

Figure 5: Numerical results for the compound wave problem, Burgers' equation.

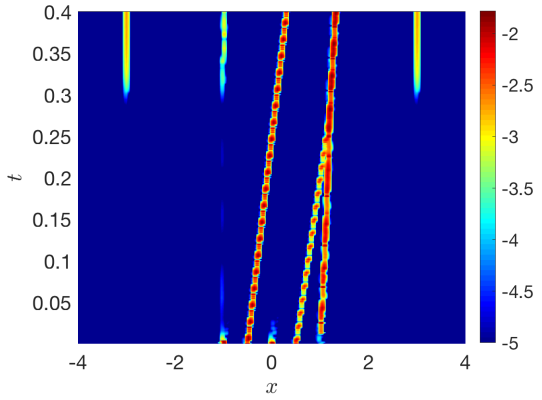


(a) EV

(b) MDH



(c) MDA



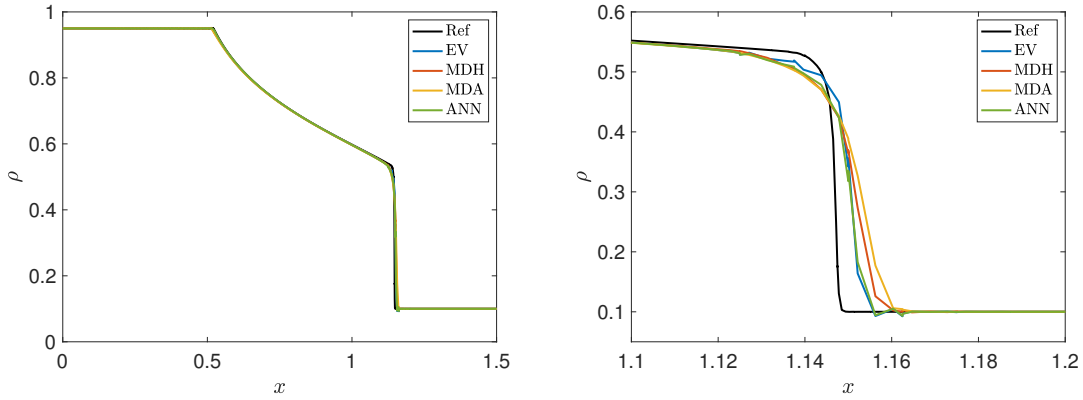
(d) ANN

Figure 6: Temporal history of the logarithm of the artificial viscosity for the compound wave problem, Burgers' equation, $m = 4$.

(2.11), setting $\Omega = [0, 1.5]$, $K = 120$, $T = 0.4$ and $C = 1$. The problem is completed by choosing

$$u_0(x) = \begin{cases} 0.95 & \text{if } 0 \leqslant x < 0.5, \\ 0.1 & \text{if } 0.5 \leqslant x \leqslant 1.5, \end{cases} \tag{4.14}$$

and constant consistent Dirichlet boundary conditions. The initial condition evolves into a compound wave consisting of a shock and a rarefaction wave [25]. The results are shown in Figure 7, while the temporal evolution of the artificial viscosity is reported in Figure 8, both generated with $m = 4$. Most of the comments related to problem 4.4.2 remain valid, with all the models able to



(a) Overall result.

(b) Zoom close to the shock.

Figure 7: Numerical results for the Buckley-Leverett problem, $m = 4$.

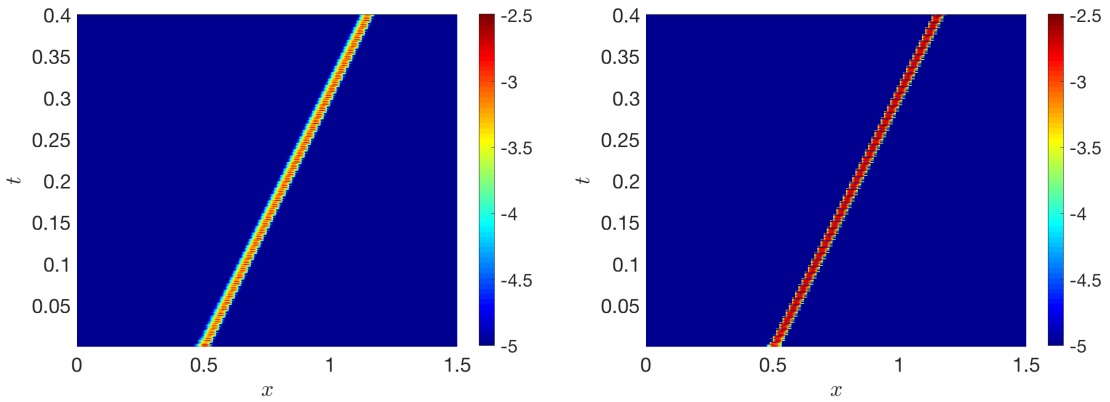
suppress the oscillations. Both the EV and the ANN approaches resolve the shock well, but the former leads to a slightly more oscillatory profile. All four models inject viscosity near the shock only, leaving the inviscid scheme unaltered everywhere else.

4.4.4. Sod shock-tube problem

We now consider a couple of test cases for the Euler equations, seeking to show the potential of the network-based technique for systems of conservation laws. We begin with the Sod shock-tube problem [37]. We select $\Omega = [0, 1]$, $K = 100$, $T = 0.2$, $C = 0.2$ and the initial state

$$(\rho, v, p)_0 = \begin{cases} (1, 0, 1) & \text{if } 0 \leqslant x \leqslant 0.5, \\ (0.125, 1, 0.1) & \text{if } 0.5 < x \leqslant 1, \end{cases} \tag{4.15}$$

completing the problem with constant Dirichlet boundary conditions. Three different waves are generated. A left-moving rarefaction fan, where no dissipation should be added, and right-moving contact and shock waves. Viscosity should be continuously injected in the region close to the shock, while less dissipation should be added where the contact develops. As prototype cases, we rely once more on $m = 1$ and $m = 4$, which result in the profiles shown in Figure 9. The temporal history of the artificial viscosity is reported, for the latter case only, in Figure 10. For both degrees, it appears that the EV model outperforms the others, as the parameters are optimal for this particular test



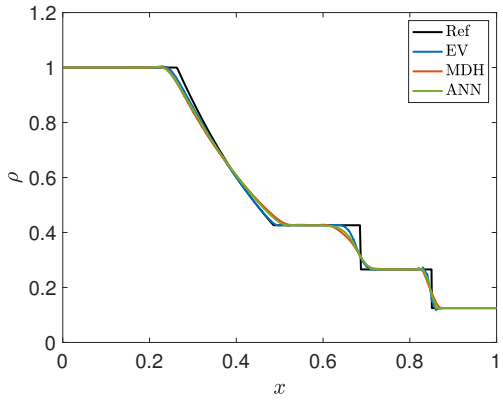
(a) EV

(b) MDH

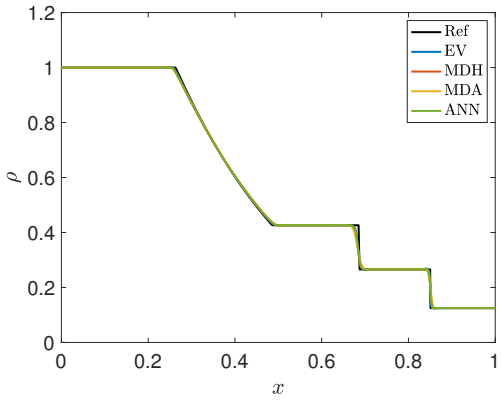
(c) MDA

(d) ANN

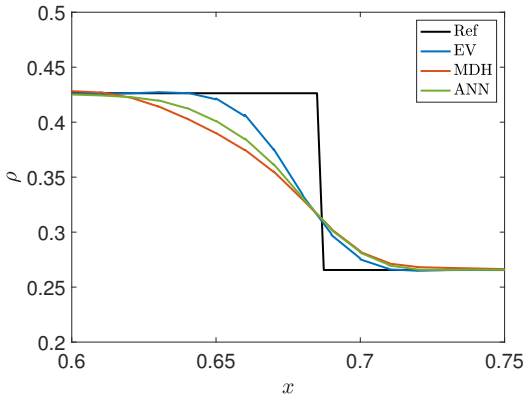
Figure 8: Temporal history of the logarithm of the artificial viscosity for the Buckley-Leverett problem, $m = 4$.



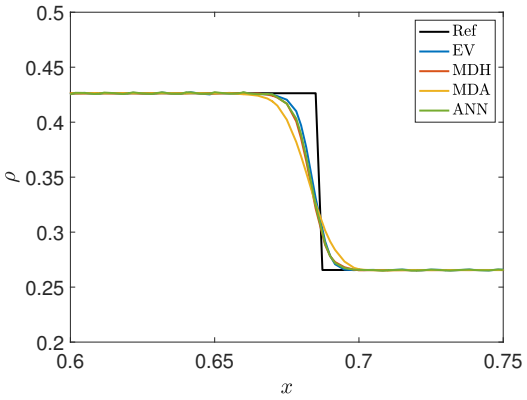
(a) Overall result, $m = 1$.



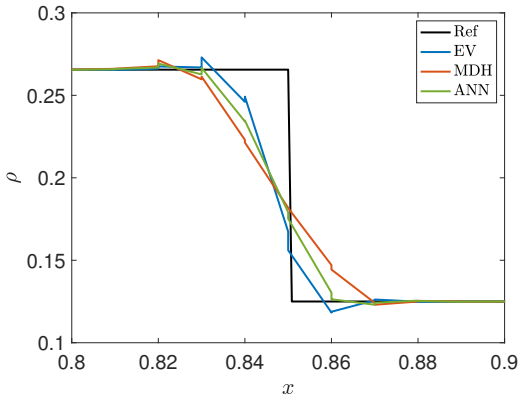
(b) Overall result, $m = 4$.



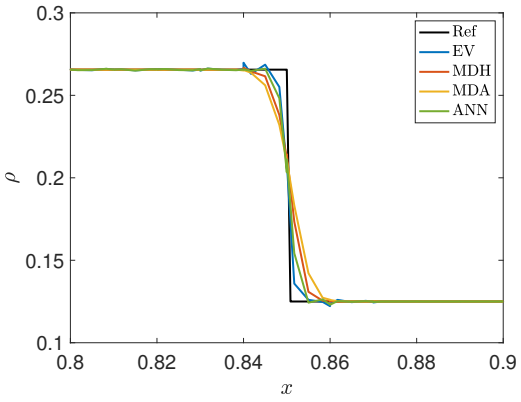
(c) Zoom close to the contact wave, $m = 1$.



(d) Zoom close to the contact wave, $m = 4$.

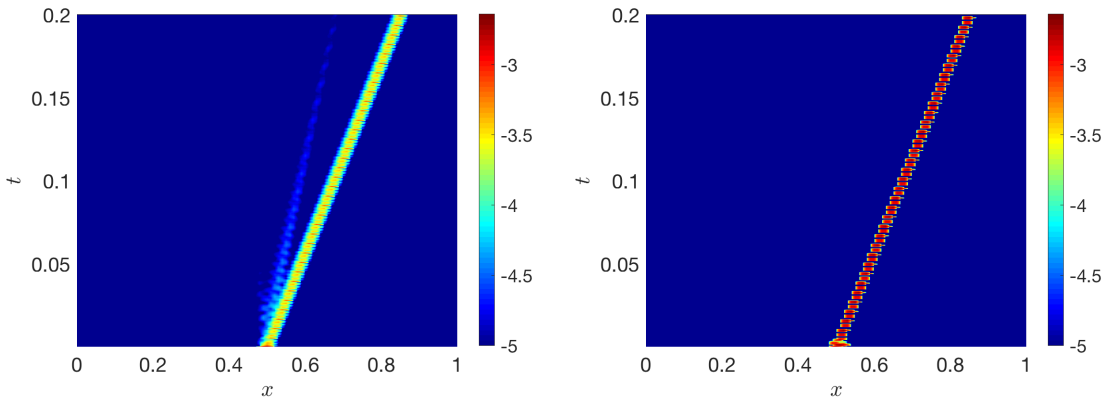


(e) Zoom close to the shock wave, $m = 1$.



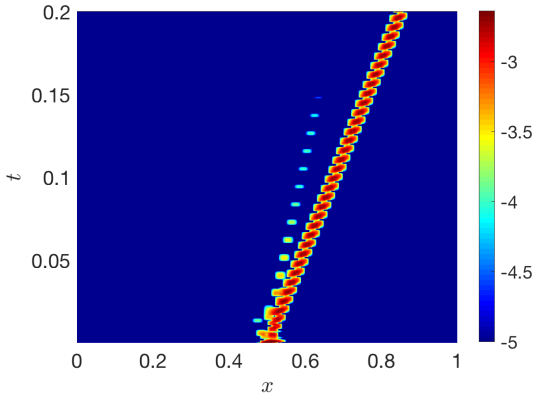
(f) Zoom close to the shock wave, $m = 4$.

Figure 9: Numerical results for the Sod problem (density).

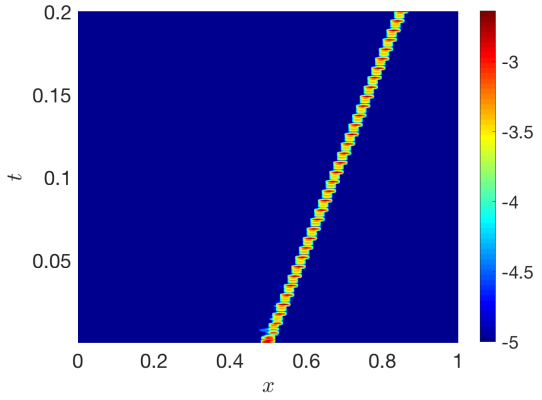


(a) EV

(b) MDH



(c) MDA



(d) ANN

Figure 10: Temporal history of the logarithm of the artificial viscosity for the Sod problem, $m = 4$.

case. Both the contact and the shock are better resolved, even though the corresponding solution has a slightly larger overshoot close to the shock. The ANN reduces its amplitude, and it is able to resolve the shock better than the decay-based models. The contact is also captured, with better performances as the polynomial degree increases. Initially, dissipation is injected close to the discontinuity, while, as time evolves, it is enough to track the position of the shock and add viscosity in that region only. This is the behavior observed with the MDH and ANN. On the other hand, both the EV and the MDA models add a small amount of dissipation close to the contact wave, even though it eventually vanishes.

4.4.5. Shu-Osher problem

The second test case is the Shu-Osher problem [36]. Consider $\Omega = [-5, 5]$, $K = 200$, $T = 1.8$ and $C = 0.2$. The initial condition is defined as

$$(\rho, v, p)_0 = \begin{cases} (3.857143, 2.629369, 10.333333) & \text{if } -5 \leqslant x \leqslant -4, \\ (1 + 0.5\sin(5x), 0, 1) & \text{if } -4 < x \leqslant 5, \end{cases} \tag{4.16}$$

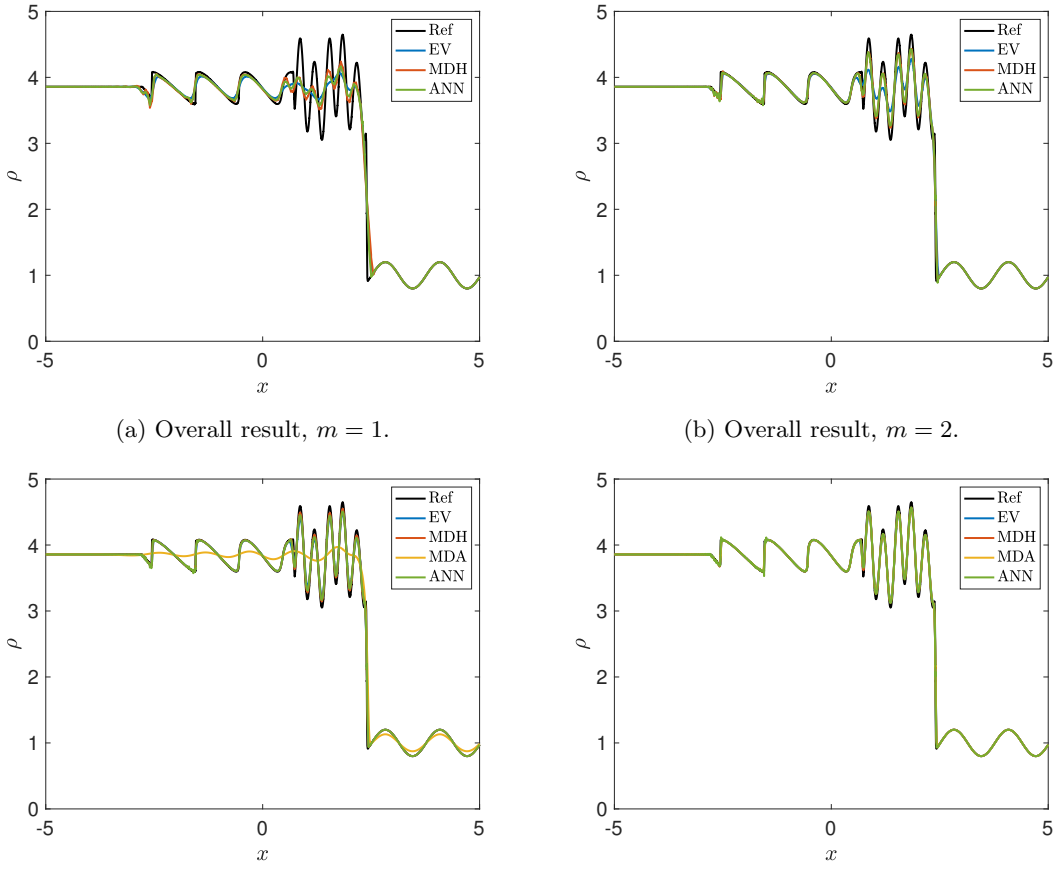
with Dirichlet (resp. Neumann) conditions at the left (resp. right) boundary. A combination of smooth and discontinuous data is present, making this case well suited to test the capability to capture both shocks and physical oscillations. The results for $m = 1, 2, 3, 4$ are shown in Figures 11 and 12, with the viscosity profile for $m = 4$ reported in Figure 13. The overall solution quality improves as m is increased, so that the injected dissipation does not destroy the benefits of choosing higher discretization orders. The network-based model shows performances similar to the best among the classical techniques for each discretization degree. For low orders, it clearly outperforms the EV model. Even though this is less evident, this phenomenon is present even for higher orders. The results obtained with the ANN and MDH models look similar, while the MDA does not perform well. Choosing $m = 4$, the ANN appears to be the best technique in terms of resolving the physical fluctuations. Our observations are validated by the analysis of the artificial viscosity profile. The EV model is the most dissipative and adds a less localized viscosity. The MDH, MDA and ANN have similar profiles, keeping track of both strong and weak shocks.

5. Two-dimensional networks

The technique described in Section 4 can be extended to higher spatial dimensions, keeping the spirit of one-dimensional problems.

5.1. A two-dimensional model

As in Section 4.1, we construct a family of networks, where input and output are the scaled nodal values of the solution and the artificial viscosity, respectively. Even though the network architecture, activations and cost functions are not altered, the number of hidden neurons has to be re-tuned. Since the input dimension N_m increases significantly, we noted that having 10 neurons in each layer is no longer sufficient. A good compromise between computational performances and accuracy is found with $N_H^l = 20$ neurons per hidden layer. Note that the network depth L is kept equal to 5. The generation procedure for training and validation sets mimics the one-dimensional one. The main difference is that a single mesh is used, in order to control computational performances and memory issues. To capture spatial variability, a fine enough unstructured grid is used. In Table 4 we show a concrete example for the case $m = 4$, generated using the two-dimensional Burgers' equation. The corresponding initial conditions are:



(a) Overall result, $m = 1$.

(b) Overall result, $m = 2$.

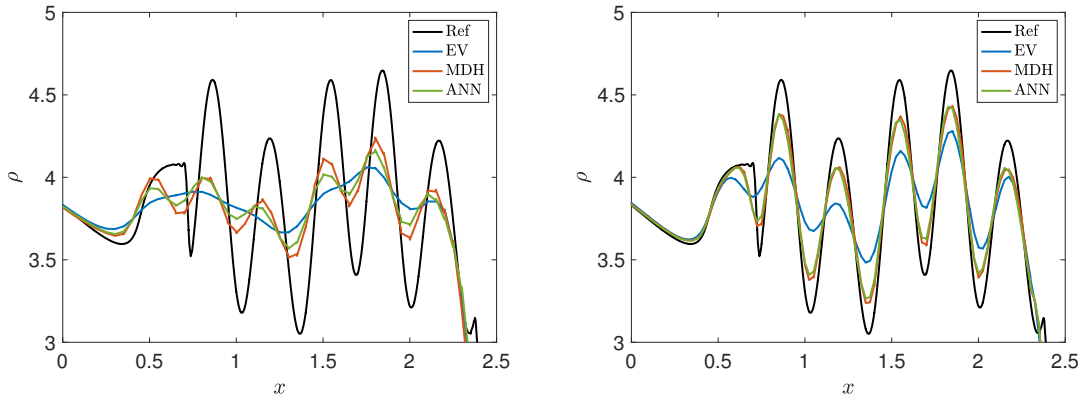
(c) Overall result, $m = 3$.

(d) Overall result, $m = 4$.

Figure 11: Numerical results for the Shu-Osher problem (density).

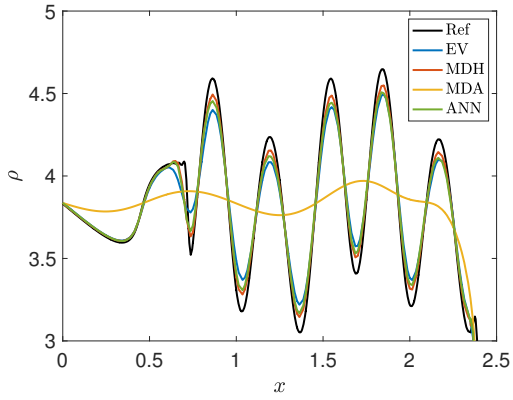
u_0	Ω	T	Best model	Nb. samples
u_0^1	$[0,1]^2$	0.12	MDH: $c_A = 2.0,\ c_\kappa = 0.2,\ c_{max} = 0.5$	4153666
u_0^2	$[0,1]^2$	0.12	EV: $c_E = 1.0,\ c_{max} = 0.4$	3715358
u_0^3	$[0,1]^2$	0.06	EV: $c_E = 1.5,\ c_{max} = 0.2$	815098
u_0^4	$[0,1]^2$	0.12	MDH: $c_A = 2.0,\ c_\kappa = 0.4,\ c_{max} = 1$	1841660

Table 4: Example on dataset generation, two-dimensional Burgers' equation, $m = 4$.

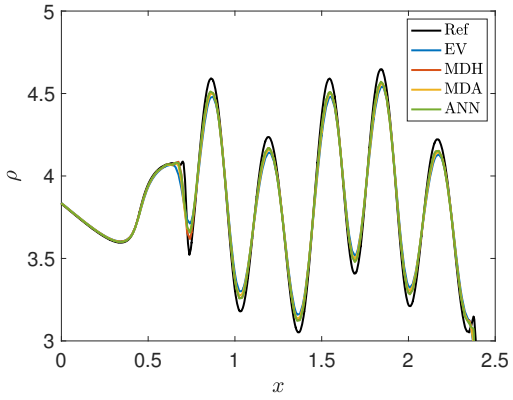


(a) $m = 1$.

(b) $m = 2$.

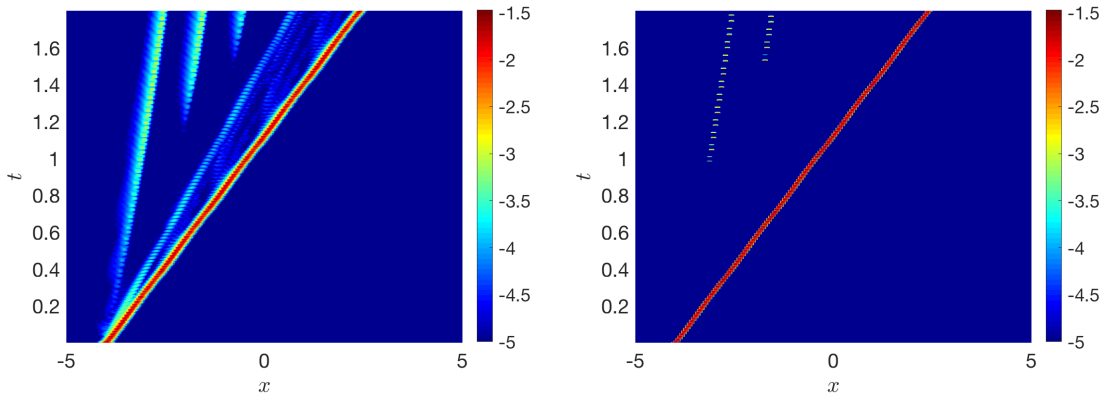


(c) $m = 3$.



(d) $m = 4$.

Figure 12: Numerical results for the Shu-Osher problem (density). Zoom close to the fluctuations.



(a) EV

(b) MDH

(c) MDA

(d) ANN

Figure 13: Temporal history of the logarithm of the artificial viscosity for the Shu-Osher problem, $m = 4$.

- $u_0^1(x,y) = \sin(\pi\omega x)\sin(\pi\omega y)$, with $\omega \in \{2,4\}$.
- $u_0^2(x,y) = 1 \cdot \mathbb{1}_{[0,0.5]}(x)\mathbb{1}_{[0.5,1]}(y) + 2 \cdot \mathbb{1}_{[0.5,0.5]}(x)\mathbb{1}_{[0,0.5]}(y) + 1 \cdot \mathbb{1}_{[0.5,1]}(x)\mathbb{1}_{[0,0.5]}(y)$.
- $u_0^3(x,y) = 3 + 6(0.25 - |y - 0.5|) \cdot \mathbb{1}_{[0.25,0.75]}(y)$.
- $u_0^4(x,y) = u_0^4(r) = -2\left(1 + \cos\left(\frac{\pi}{0.5}r\right)\right) \cdot \mathbb{1}_{[0,0.5]}(r)$, where $r = \sqrt{(x-0.5)^2 + (y-0.5)^2}$.

Finally, the modified scaling proposed in Section 4.3 is easily ported to a two-dimensional scenario, by taking into account all the boundary quadrature points.

5.2. Numerical results

We now present a few numerical results for two-dimensional problems. In this framework, the differences among the models are enhanced, since different combinations of one-dimensional waves lead to more complex structures that have to be captured by the models. The ANN model is always superior to the others, unless specific values for the parameters are chosen. To better demonstrate the potential of the network, we fix the parameters for the classical artificial viscosity models, as reported in Table 5. As in the one-dimensional framework, we consider $m = 1, 2, 3, 4$. For the test cases considered below, the domains are discretized using two types of meshes, defined as follows:

- Mesh *S-N*. It is obtained by considering a structured quadrilateral grid with N elements on each edge, and dividing each element into two triangles. Thus, the mesh will have $K = 2 \cdot N^2$ triangular elements.
- Mesh *U-q*. It is an unstructured triangular mesh with characteristic grid size q.

Model	2D scalar	2D Euler
EV	$c_E = 1,\ c_{max} = 0.25$	$c_E = 1,\ c_{max} = 0.5$
MDH	$c_A = 2,\ c_\kappa = 0.4,\ c_{max} = 0.8$	$c_A = 2,\ c_\kappa = 0.2,\ c_{max} = 0.8$
MDA	$c_{max} = 0.8$	$c_{max} = 0.5$

Table 5: Parameter values for the standard artificial viscosity models for 2D problems.

5.2.1. A smooth problem

We first validate the techniques by considering a smooth problem, i.e., the linear advection equation with constant transport field $\boldsymbol{\beta} = (\beta^x, \beta^y) = (1,1)$. We choose $\Omega = [0,1]^2$, the initial condition

$$u_0(x,y) = 1 + \sin(2\pi x)\sin(2\pi y), \tag{5.1}$$

periodic boundary conditions, and $T = 0.2$. The convergence analysis leads to the results reported in Tables 6 and 7 for structured and unstructured meshes, respectively. The corresponding CFL constants have been chosen equal to $C = 0.8$ and $C = 0.4$. The results are consistent with their one-dimensional counterpart, and most of the comments in Section 4.4.1 remain valid. The inviscid scheme guarantees optimal accuracy, as well as the EV model. On structured meshes, the decay-based models are first-order accurate for low resolution, while the inviscid scheme is retained for higher polynomial orders. Conversely, on unstructured meshes the MDH model guarantees the optimal accuracy even at low degrees, while the qualitative behavior of the MDA does not change. The ANN technique achieves the optimal convergence rate, with errors comparable to the inviscid scheme independently of the mesh type.

m	Mesh S-	Inviscid		EV		MDH		MDA		ANN	
		ϵ	p	ϵ	p	ϵ	p	ϵ	p	ϵ	p
1	10	1.8672e−2	-	3.0321e−1	-	4.1350e−1	-			1.1511e−1	-
	20	4.5400e−3	2.04	1.1735e−1	1.37	2.9138e−1	0.51			2.9692e−2	1.95
	40	1.1194e−3	2.02	1.3438e−2	3.13	1.5561e−2	0.91			6.1154e−3	2.28
	80	2.7864e−4	2.01	1.7744e−3	2.92	8.1373e−2	0.94			1.0747e−3	2.51
	160	6.9578e−5	2.00	2.3654e−4	2.91	4.1920e−2	0.96			1.6979e−4	2.66
2	10	2.0727e−3	-	2.6468e−1	-	2.1721e−1	-			7.6748e−3	-
	20	2.7604e−4	2.91	8.9372e−3	4.89	9.1063e−2	1.25			5.7189e−4	3.75
	40	3.4781e−5	2.99	5.6395e−4	3.97	4.0360e−2	1.17			3.9325e−5	3.86
	80	4.3580e−6	3.00	3.7920e−5	3.89	1.7564e−2	1.20			4.4154e−6	3.15
	160	5.4508e−7	3.00	2.4749e−6	3.94	7.4911e−3	1.23			5.4733e−7	3.01
3	10	1.2818e−4	-	2.7373e−2	-	3.4605e−2	-	1.0003e−1	-	1.4388e−4	-
	20	8.0155e−6	4.00	4.6557e−4	5.88	8.0155e−6	12.07	5.3495e−2	0.90	8.1953e−6	4.13
	40	5.2651e−7	3.93	1.6802e−5	4.79	5.2651e−7	3.93	3.2831e−2	0.70	5.3040e−7	3.95
	80	3.2941e−8	4.00	5.7068e−7	4.88	3.2941e−8	4.00	1.9288e−2	0.77	3.3049e−8	4.00
4	10	1.1961e−5	-	4.2204e−3	-	1.3695e−2	-	1.1961e−5	-	1.2135e−5	-
	20	3.5969e−7	5.06	1.2057e−5	8.45	3.5969e−7	15.22	3.5969e−7	5.06	3.6035e−7	5.07
	40	1.0980e−8	5.03	2.1698e−7	5.80	1.0980e−8	5.03	1.0980e−8	5.03	1.0989e−8	5.04
	80	3.4094e−10	5.01	3.9194e−9	5.79	3.4094e−10	5.01	3.4094e−8	5.01	3.4107e−10	5.00

Table 6: L^2 errors ϵ and estimated rate of convergence p in the inviscid case and with the artificial viscosity models for the linear advection problem using structured meshes.

m	Mesh U-	Inviscid		EV		MDH		MDA		ANN	
		ϵ	p	ϵ	p	ϵ	p	ϵ	p	ϵ	p
1	0.1	1.5764e−2	-	3.4709e−1	-	1.3597e−1	-			7.8710e−2	-
	0.05	3.5448e−3	2.15	5.4757e−2	2.83	4.8203e−2	1.50			1.9643e−2	2.00
	0.025	8.5458e−4	2.05	6.3683e−3	2.84	1.0718e−2	2.17			3.9005e−3	2.33
	0.0125	2.1797e−4	1.97	8.9245e−4	3.10	1.9222e−3	2.47			6.0793e−4	2.68
	0.00625	5.444e−5	2.00	1.2547e−4	2.66	3.0148e−4	2.67			1.0746e−4	2.50
2	0.1	1.4611e−3	-	3.6159e−2	-	5.7999e−2	-			4.4728e−3	-
	0.05	1.9270e−4	2.92	1.7627e−3	4.36	1.1351e−2	2.35			2.5448e−4	4.14
	0.025	2.3932e−5	3.01	1.0981e−4	4.00	1.8357e−3	2.63			2.4715e−5	3.36
	0.0125	3.0338e−6	2.98	7.2953e−6	3.79	2.7075e−4	2.76			3.0451e−6	3.02
	0.00625	3.9030e−7	2.96	6.2576e−7	3.66	4.0006e−5	2.76			3.9114e−7	2.96
3	0.1	9.5672e−5	-	5.5512e−3	-	9.5661e−5	-	1.0251e−1	-	1.0110e−4	-
	0.05	5.1961e−6	4.20	5.4090e−5	6.68	5.1961e−6	4.20	5.0981e−2	1.01	5.3520e−6	4.24
	0.025	3.1269e−7	4.05	1.6869e−6	5.00	3.1269e−7	3.05	2.5970e−2	0.97	3.1613e−7	4.08
	0.0125	1.9776e−8	3.98	6.2160e−8	4.76	1.9776e−8	3.98	1.3447e−2	0.95	1.9892e−8	3.99
4	0.1	4.3367e−6	-	1.3564e−4	-	4.3367e−6	-	4.3367e−6	-	4.4617e−6	-
	0.05	1.5175e−7	4.84	1.9874e−6	6.09	1.5175e−7	4.84	1.5175e−7	4.84	1.5263e−7	4.87
	0.025	4.2212e−9	5.17	3.2082e−8	5.95	4.2212e−9	5.17	4.2212e−9	5.17	4.2332e−9	5.17
	0.0125	1.4308e−10	4.88	6.9971e−10	5.52	1.4308e−10	4.88	1.4308e−10	4.88	1.4332e−10	4.88

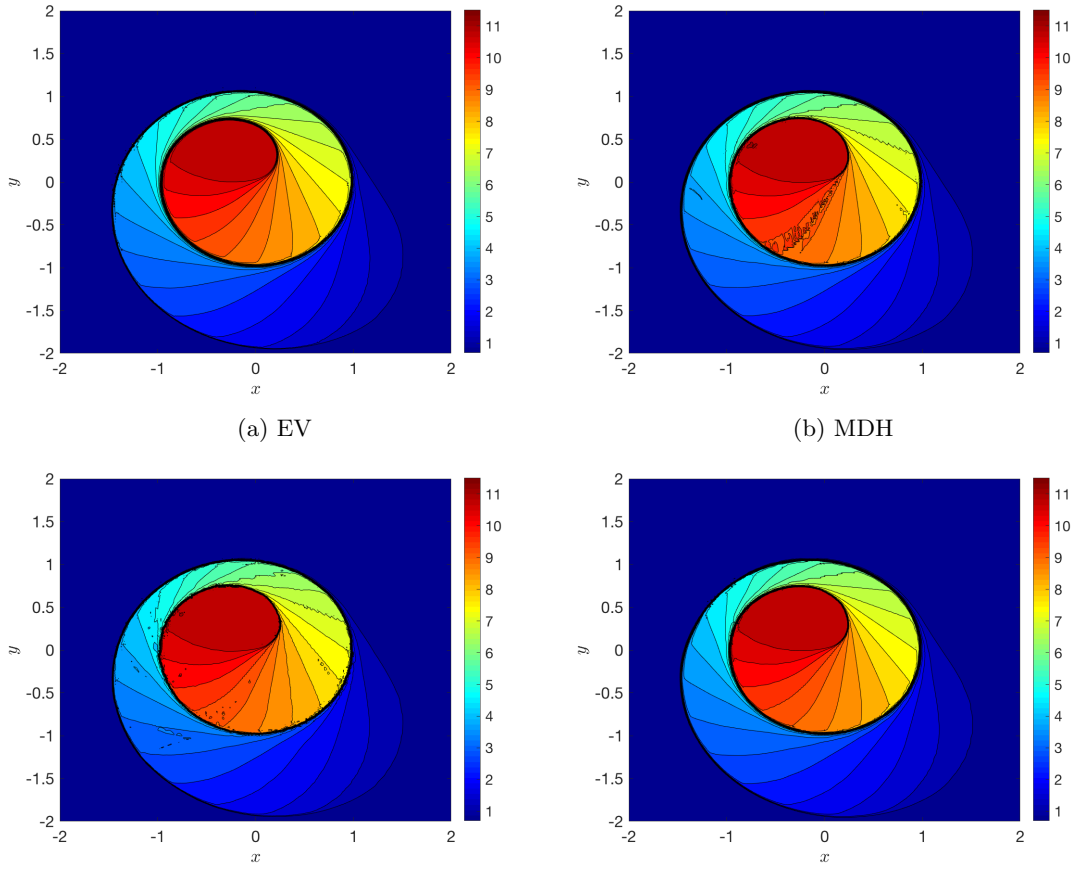
Table 7: L^2 errors ϵ and estimated rate of convergence p in the inviscid case and with the artificial viscosity models for the linear advection problem using unstructured meshes.

5.2.2. KPP rotating wave problem

A few tests were run using the two-dimensional extension of Burgers' equation. The results are as expected and are not reported. Instead, we consider the KPP rotating wave problem, with the flux function defined in (2.14). We select $\Omega = [-2, 2]^2$, $T = 1$ and $C = 0.8$. Choosing the initial condition as

$$u_0(x, y) = \begin{cases} 3.5\pi & \text{if } x^2 + y^2 < 1, \\ 0.25\pi & \text{otherwise,} \end{cases} \tag{5.2}$$

and periodic boundary conditions, a two-dimensional composite wave structure is present [14]. Simulations have been run using the S-120 and U-0.04 meshes, leading to the results reported in Figure 14 and Figure 15, respectively. The degree $m = 4$ is used in both cases. The models yielding

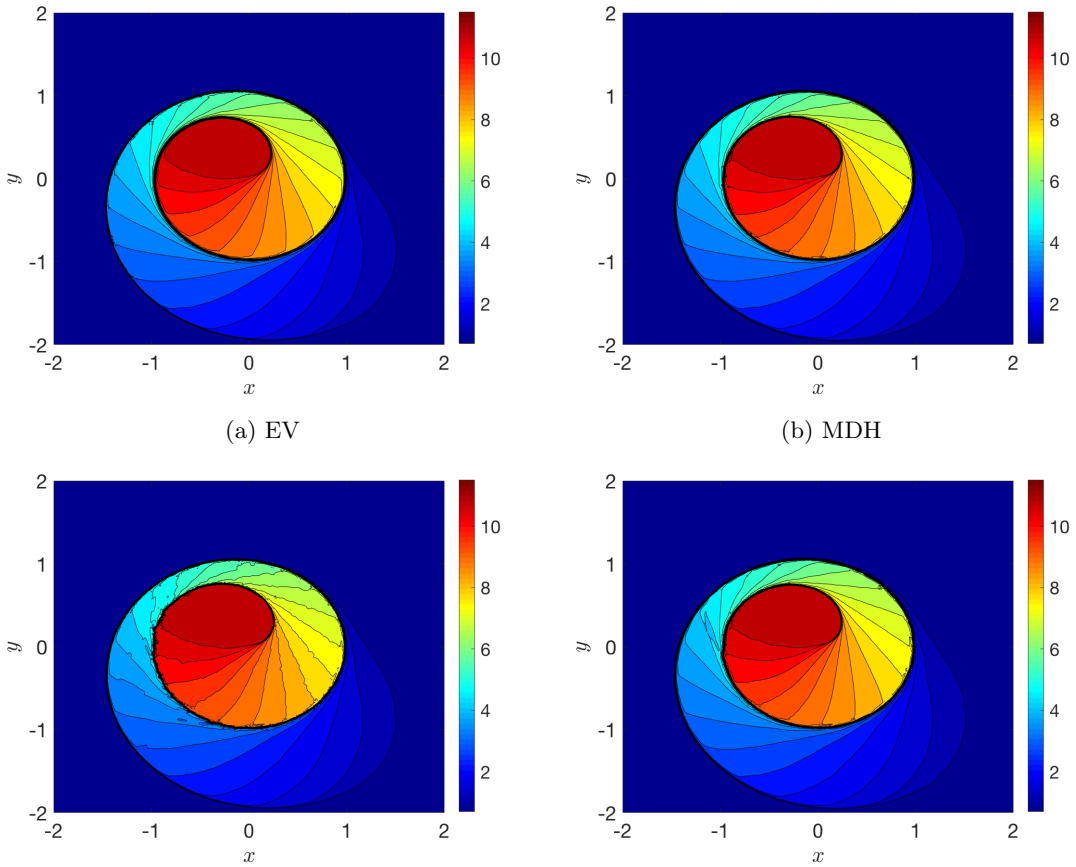


(a) EV (b) MDH

(c) MDA (d) ANN

Figure 14: Numerical results for the KPP problem on a structured mesh, $m = 4$. Thirty equally spaced contours from 0.7 to 11.5.

the most accurate results are the EV and the ANN, which result in similar solution profiles using both the structured and the unstructured mesh. In the second case, the EV approach slightly



(a) EV

(b) MDH

(c) MDA

(d) ANN

Figure 15: Numerical results for the KPP problem on an unstructured mesh, $m = 4$. Thirty equally spaced contours from 0.7 to 11.5.

outperforms the network. The MDH model results in a too oscillatory solution using the structured mesh, while the results are comparable with the EV and ANN methods on the unstructured grid. The MDA is not well suited for this problem, as a lot of wiggles are present.

5.2.3. Euler equations: Case 4

We now consider two-dimensional Riemann problems for the Euler system. They are described in detail in [24, 34]. In all the following test cases the physical domain is $[0,1]^2$, enlarged to $[-1,2]^2$ to avoid that the physical solution is contaminated by boundary effects. The domain is discretized using the mesh S-120 and periodic boundary conditions are applied. The results, shown in the physical domain only, are obtained using different CFL constants as m is varied, setting $C = 0.2, 0.6, 0.6, 1.5$ for $m = 1, 2, 3, 4$ respectively.

We first consider a test case where only shock waves are present. We set $T = 0.25$ and

$$(\rho, v_x, v_y, p)_0 = \begin{cases} (1.1, 0, 0, 1.1) & \text{if } 0.5 < x,\ 0.5 < y, \\ (0.5065, 0.8939, 0, 0.35) & \text{if } x < 0.5,\ 0.5 < y, \\ (1.1, 0.8939, 0.8939, 1.1) & \text{if } x < 0.5,\ y < 0.5, \\ (0.5065, 0, 0.8939, 0.35) & \text{if } 0.5 < x,\ y < 0.5. \end{cases} \tag{5.3}$$

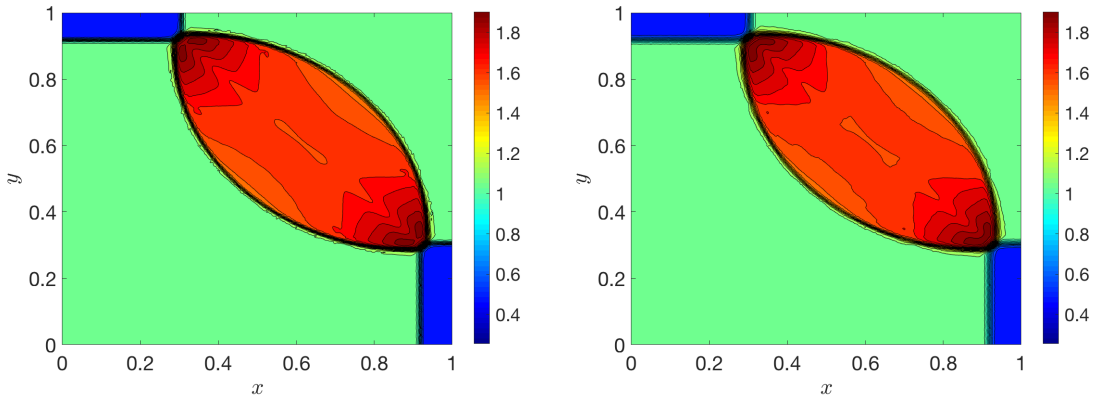
The results are reported in Figure 16 for $m = 4$. The MDA is the least dissipative model, and the corresponding solution creates spurious wiggles. On the other hand, the MDH model is the most dissipative one, as certified by the lower resolution of the shocks. Both the EV and the ANN model are superior. The latter appears to capture the fine structures in the high-density regions more accurately.

5.2.4. Euler equations: Case 12

This problem is characterized by the presence of both contact waves and shocks. We set $T = 0.25$ and

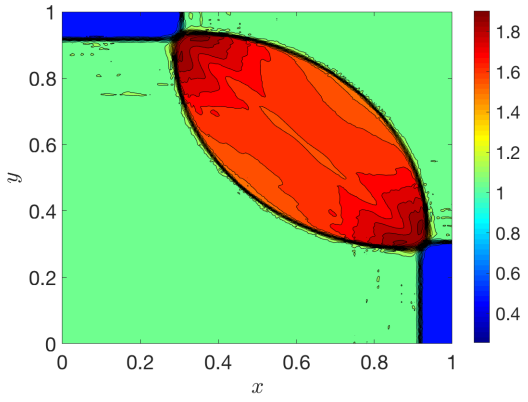
$$(\rho, v_x, v_y, p)_0 = \begin{cases} (0.5313, 0, 0, 0.4) & \text{if } 0.5 < x,\ 0.5 < y, \\ (1, 0.7276, 0, 1) & \text{if } x < 0.5,\ 0.5 < y, \\ (0.8, 0, 0, 1) & \text{if } x < 0.5,\ y < 0.5, \\ (1, 0, 0.7276, 1) & \text{if } 0.5 < x,\ y < 0.5. \end{cases} \tag{5.4}$$

The results are reported in Figures 17, 18, 19 and 20 for degrees $m = 1, 2, 3, 4$ respectively. A simple yet effective way to illustrate the robustness of a model is to evaluate how well the fine structures close to $(x, y) = (0.5, 0.5)$ are captured. For low degrees, the resolution is too low to fully resolve the solution, with all models oversmoothing the solution around the center of the domain. Increasing the order of approximation, the potential of the DG scheme is triggered. The EV model does not capture well the contact wave, adding too much dissipation in that region. The MDH model exhibits larger smoothing close to the shock waves, while the MDA model produces an either too dissipative ($m = 3$) or too oscillatory ($m = 4$) solution. The ANN is able to capture the central structure, the contact waves and the shocks, making it the best performing model for this test case.

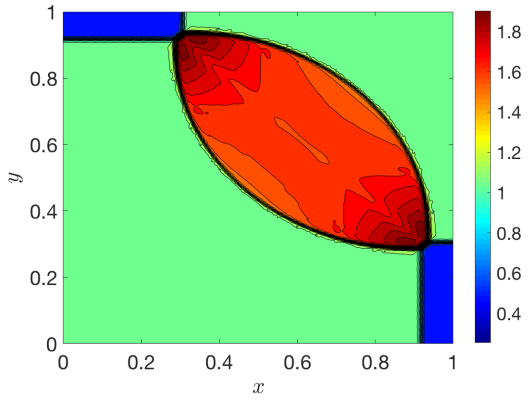


(a) EV

(b) MDH

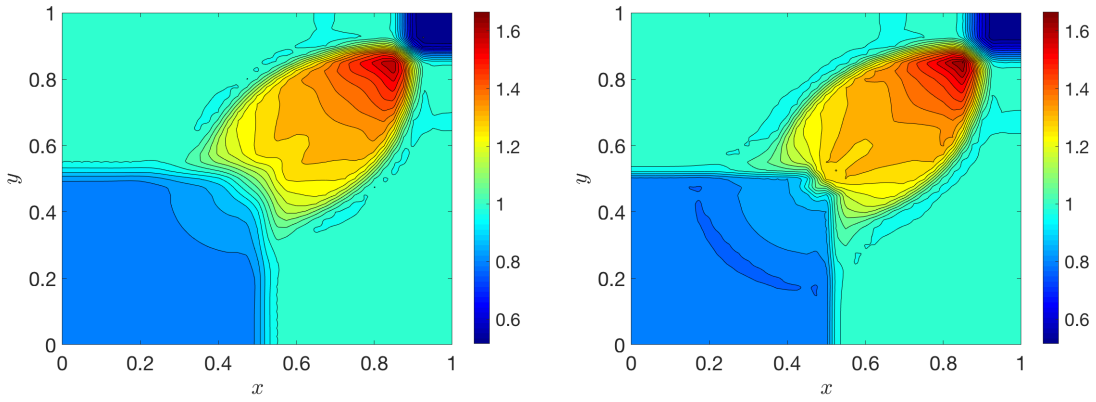


(c) MDA



(d) ANN

Figure 16: Numerical results for the Riemann problem configuration 4, $m = 4$ (density). Thirty equally spaced contours from 0.255 to 1.9.

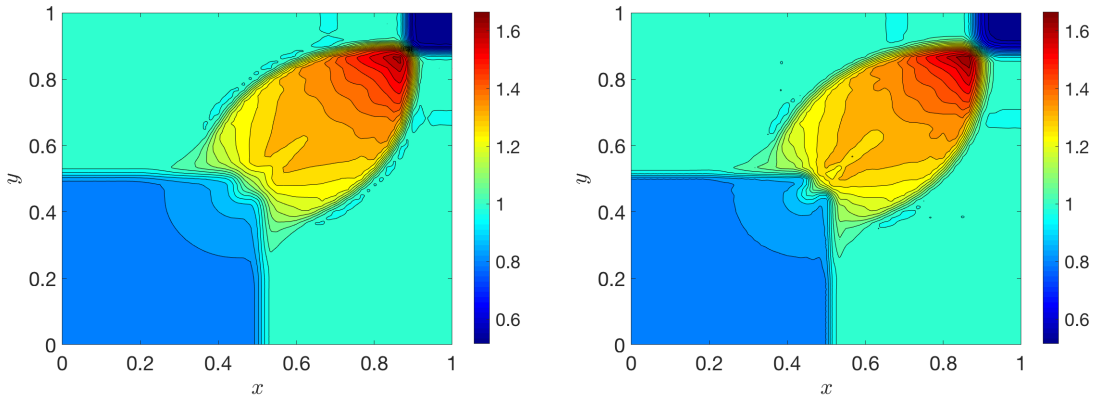


(a) EV

(b) MDH

(c) ANN

Figure 17: Numerical results for the Riemann problem configuration 12, $m = 1$ (density). Thirty equally spaced contours from 0.515 to 1.665.

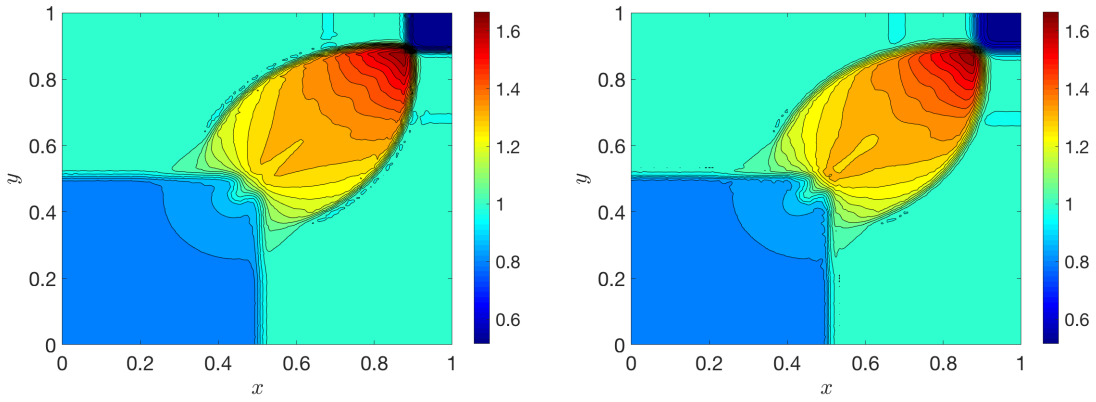


(a) EV

(b) MDH

(c) ANN

Figure 18: Numerical results for the Riemann problem configuration 12, $m = 2$ (density). Thirty equally spaced contours from 0.515 to 1.665.



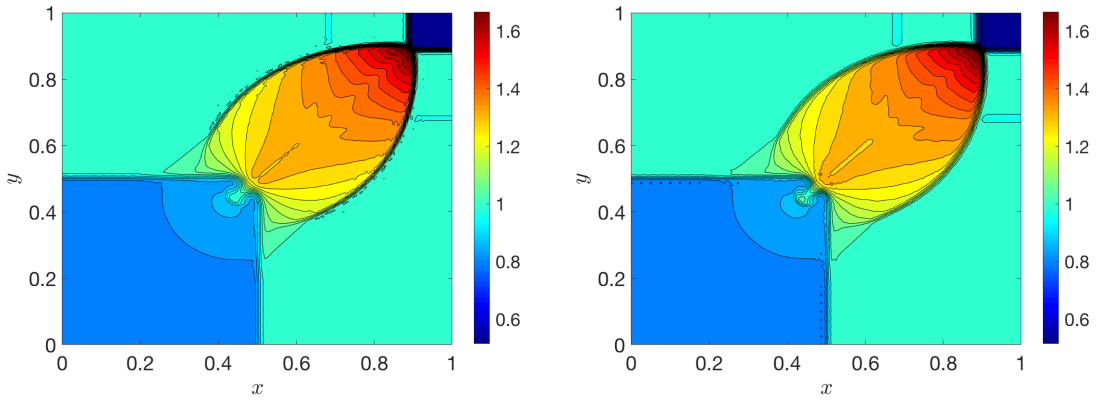
(a) EV

(b) MDH

(c) MDA

(d) ANN

Figure 19: Numerical results for the Riemann problem configuration 12, $m = 3$ (density). Thirty equally spaced contours from 0.515 to 1.665.



(a) EV

(b) MDH

(c) MDA

(d) ANN

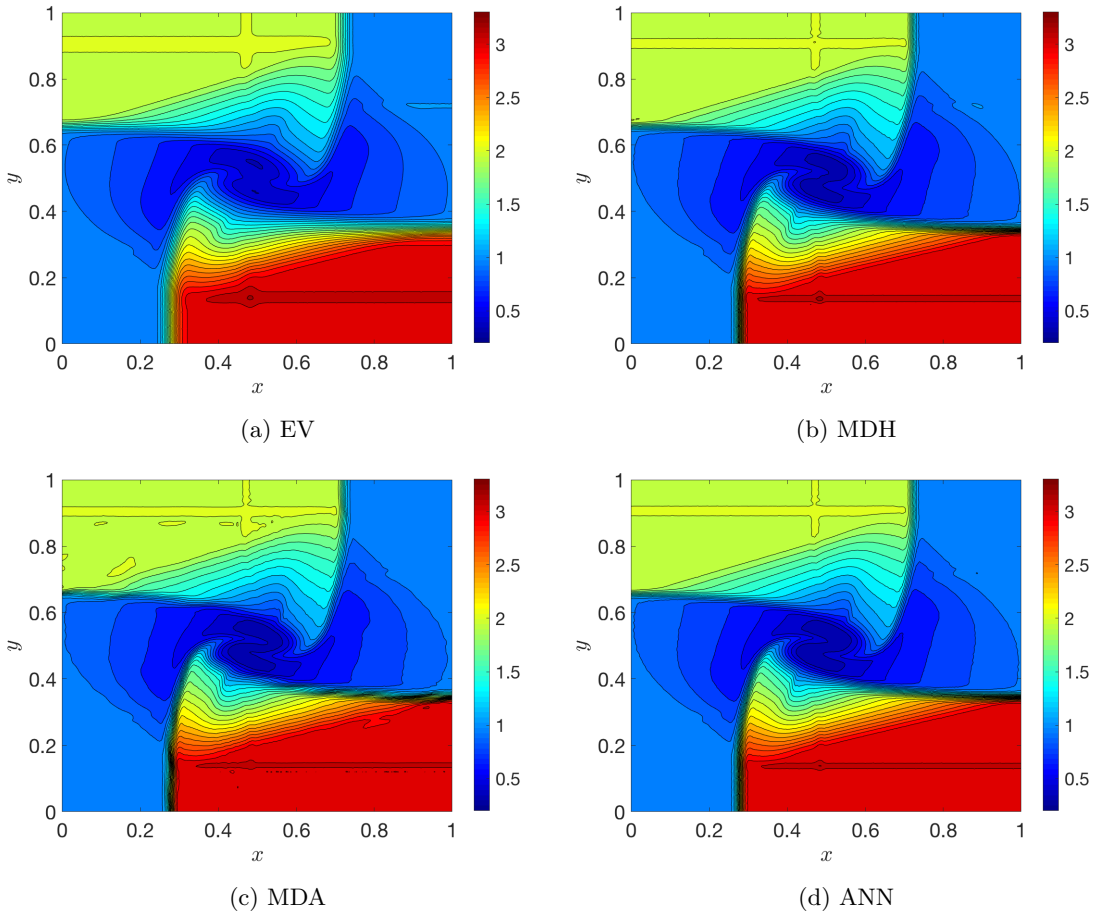
Figure 20: Numerical results for the Riemann problem configuration 12, $m = 4$ (density). Thirty equally spaced contours from 0.515 to 1.665.

5.2.5. Euler equations: Case 6

The final test exhibits four contact waves. We set $T = 0.3$ and

$$(\rho, v_x, v_y, p)_0 = \begin{cases} (1, 0.75, -0.5, 1) & \text{if } 0.5 < x,\ 0.5 < y, \\ (2, 0.75, 0.5, 1) & \text{if } x < 0.5,\ 0.5 < y, \\ (1, -0.75, 0.5, 1) & \text{if } x < 0.5,\ y < 0.5, \\ (3, -0.75, -0.5, 1) & \text{if } 0.5 < x,\ y < 0.5, \end{cases} \tag{5.5}$$

leading to the solution profiles reported in Figure 21 for $m = 4$. Consistently with the results for



(a) EV (b) MDH

(c) MDA (d) ANN

Figure 21: Numerical results for the Riemann problem configuration 6, $m = 4$ (density). Thirty equally spaced contours from 0.2 to 3.3.

the Configuration 12 (Section 5.2.4), the EV model tends to overdissipate the solution. For the parameters we selected, the model is not able to capture the contact waves well. More generally, it is not well suited for complex flow structures [39]. Conversely, the ANN is able to resolve all the features. The decay-based models give results comparable to the network-based one.

6. Performance evaluation

One of the main advantages of the ANNs lies in the computational cost. The offline phase, consisting of dataset generation and network training, is potentially expensive. However, it is done only once and can be ignored in the performance analysis. The online part, which is integrated in the time-advancing loop, is computationally cheap. It consists of matrix-vector multiplications and evaluations of the activation functions.

For the analysed test cases, the simulation time per timestep is comparable among all models. Indeed, the computational cost required by the assembly of the right-hand-side of the equation is much larger than the cost of the viscosity estimation procedure, which is done only once per time iteration. On the other hand, the total execution time can exhibit significant variations, especially in two-dimensional problems. Following (2.10), the time step is chosen adaptively and depends on both the maximum viscosity and the maximum wave speed of the problem. A trade-off between their values is present, leading to different time steps. It appears that the decay-based models are less efficient than the EV or ANN-based schemes. To provide a more quantitative analysis, we select a couple of test cases among the ones presented in Sections 4.4 and 5.2. The simulations have been run on *Matlab* using a 3.2 GHz Intel Core i5 processor.

For one-dimensional problems, we show the results of the Shu-Osher problem (Section 4.4.5), to which we refer for the numerical setup. Table 8 reports the results for different models and degrees m. The cost per timestep is slightly different among the models, due to the different viscosity

Model	$m=1$			$m=2$			$m=3$			$m=4$		
	Time	Steps	TpS	Time	Steps	TpS	Time	Steps	TpS	Time	Steps	TpS
EV	2.83	839	3.37e−3	11.20	3387	3.31e−3	25.60	7688	3.30e−3	49.17	13704	3.59e−3
MDH	2.72	847	3.21e−3	10.72	3414	3.14e−3	24.50	7716	3.18e−3	49.22	13734	3.58e−3
MDA							24.73	7491	3.30e−3	50.82	13721	3.70e−3
ANN	2.91	843	3.45e−3	11.51	3408	3.38e−3	26.12	7703	3.39e−3	52.06	13735	3.79e−3

Table 8: Computational times, number of timesteps and execution time of a single timestep TpS for the Shu-Osher problem. Both the total time and the time per timestep are expressed in seconds.

estimation procedures. The ANN is generally the slowest, but the differences with the other models are minimal. No significant variations are present even when the discretization order is varied. The total computational time is essentially independent of the chosen model, once the degree is fixed. Therefore, for one-dimensional problems, we can conclude that the computational performances among the models are comparable.

A prototype test for two-dimensional cases is the Configuration 12 of the Riemann problems for the Euler system, described in Section 5.2.4. The results of the performance analysis are reported in Table 9. As expected, the cost per timestep is approximately constant among the models, while it increases with m. Unlike the one-dimensional problems, the total computational time is very different. The MDH and MDA seem to become less efficient for high orders, since a larger number of timesteps is required. The ANN technique is computationally fast, with a performance comparable to the EV model. For low orders, the network-based approach requires the largest timestep.

Model	$m=1$			$m=2$			$m=3$			$m=4$		
	Time	Steps	TpS	Time	Steps	TpS	Time	Steps	TpS	Time	Steps	TpS
EV	268	174	1.54	516	228	2.27	1886	597	3.16	2938	696	4.23
MDH	281	186	1.51	721	331	2.18	3543	1162	3.05	6483	1516	4.27
MDA							3103	1000	3.10	6263	1470	4.26
ANN	236	152	1.55	476	213	2.24	2013	648	3.11	3522	822	4.29

Table 9: Computational times, number of timesteps and execution time of a single timestep TpS for the 2D Riemann problem (configuration 12). Both the total time and the time per timestep are expressed in seconds.

7. Conclusion

In this work we consider the problem of estimating an optimal artificial viscosity amount in high-order numerical solvers for conservation laws. Seeking to avoid a parameter-dependent viscosity estimator, we propose a new approach based on artificial neural networks. They are trained in an offline process using a robust dataset, constructed by collecting data from simulations run using the classical models with optimal parameters. The online evaluation is then integrated in the Runge-Kutta time-advancing loop.

The proposed technique has been successfully tested on both scalar equations and systems of conservation laws. Despite training the network by means of rather simple problems, it provides good generalization properties. The model guarantees optimal accuracy for smooth problems, as well as good shock-capturing properties, and recognizes regions where numerical oscillations have to be dissipated. It has been shown that the classical artificial viscosity models might fail to produce accurate results, unless optimal parameters are chosen. The proposed technique is shown to be among the best models for each test case, adding an optimal amount of dissipation. This is more evident in a two-dimensional framework, with the network being able to capture complex configurations, fine structures, and multidimensional waves. The computational cost of the network-based model is comparable with the other approaches.

Future work might investigate the applications of the proposed technique to more domain-specific test cases, such as turbulence and Large Eddy Simulations, where an efficient spatial filter is needed.

References

[1] Hesam Abbassi, Farzad Mashayek, and Gustaaf B. Jacobs. Shock capturing with entropy-based artificial viscosity for staggered grid discontinuous spectral element method. *Computers and Fluids*, 2014.

[2] Garrett E. Barter and David L. Darmofal. Shock capturing with PDE-based artificial viscosity for DGFEM: Part I. Formulation. *Journal of Computational Physics*, 229(5):1810–1827, 2010.

[3] Francesco Bassi, Andrea Crivellini, Antonio Ghidoni, and Stefano Rebay. High-order discontinuous Galerkin discretization of transonic turbulent flows. *47th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, (January), 2009.

[4] Mark H Carpenter and a Kennedy. Fourth-Order Kutta Schemes. *Nasa Technical Memorandum*, 109112:1–26, 1994.

[5] Bernardo Cockburn, San Yih Lin, and Chi Wang Shu. TVB runge-kutta local projection discontinuous galerkin finite element method for conservation laws III: One-dimensional systems. *Journal of Computational Physics*, 84(1):90–113, 1989.

[6] Bernardo Cockburn and Chi-Wang Shu. The Runge–Kutta Discontinuous Galerkin Method for Conservation Laws V. *Journal of Computational Physics*, 141(2):199–224, 1998.

[7] George Cybenko. Continuous Valued Neural Networks with Two Hidden Layers are Sufficient. Technical report, Department of Computer Science, Tufts University, 1988.

[8] George Cybenko. Approximations by superpositions of sigmoidal functions. *Approximation Theory and its Applications*, 9(3):17–28, 1989.

[9] Constantine M. Dafermos. *Hyperbolic Conservation Laws in Continuum Physics*. Springer-Verlag Berlin Heidelberg, 2010.

[10] Niccolò Discacciati. Controlling oscillations in high-order schemes using neural networks. Master's thesis, Politecnico di Milano and École Polytechnique Fédérale de Lausanne, 2018.

[11] Charles Dugas, Yoshua Bengio, François Bélisle, Claude Nadeau, and René Garcia. Incorporating second-order functional knowledge for better option pricing. *Advances in Neural Information Processing Systems*, (January 2002):472–478, 2001.

[12] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[13] David Gottlieb and Chi-Wang Shu. On the Gibbs Phenomenon and Its Resolution. *SIAM Review*, 39(4):644–668, 1997.

[14] Jean Luc Guermond, Richard Pasquetti, and Bojan Popov. Entropy viscosity method for nonlinear conservation laws. *Journal of Computational Physics*, 230(11):4248–4267, 2011.

[15] Ralf Hartmann. Adaptive discontinuous Galerkin methods with shock-capturing for the compressible NavierStokes equations. *International Journal for Numerical Methods in Fluids*, 51(9-10):1131–1156, 2006.

[16] Simon Haykin. A comprehensive foundation. *Neural Networks*, 2(2004):41, 2004.

[17] Jan S. Hesthaven. *Numerical Methods for Conservation Laws: From Analysis to Algorithms*. SIAM Publishing, 2018.

[18] Jan S. Hesthaven and Stefano Ubbiali. Non-intrusive reduced order modeling of nonlinear problems using neural networks. *Journal of Computational Physics*, 363:55–78, 2018.

[19] Jan S. Hesthaven and Tim Warburton. *Nodal discontinuous Galerkin methods*, volume 54 TS - C. 2008.

[20] Diederik Kingma and Jimmy Ba. Adam: a method for stochastic optimization. *CoRR*, pages 1–13, 2014.

[21] Andreas Klöckner, Tim Warburton, and Jan S. Hesthaven. Viscous Shock Capturing in a Time-Explicit Discontinuous Galerkin Method. *Mathematical Modeling of Natural Phenomena*, 6(3):57–83, 2011.

[22] David Kriesel. A Brief Introduction to Neural Networks, 2007.

[23] Anders Krogh and John A. Hertz. A Simple Weight Decay Can Improve Generalization. *Advances in Neural Information Processing Systems*, 4:950–957, 1992.

[24] Alexander Kurganov and Eitan Tadmor. Solution of Two-dimensional Riemann Problems for Gas Dynamics without Riemann Problem Solvers. *Numerical Methods for Partial Differential Equations*, 18(5):584–608, 2002.

[25] Randall J. LeVeque. *Finite Volume Methods for Hyperbolic Problems*. 2002.

[26] Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng. Rectifier Nonlinearities Improve Neural Network Acoustic Models. *Proceedings of the 30 th International Conference on Machine Learning*, 28:6, 2013.

[27] Ali Mani, Johan Larsson, and Parviz Moin. Suitability of artificial bulk viscosity for large-eddy simulation of turbulent flows with shocks. *Journal of Computational Physics*, 228(19):7368–7374, 2009.

[28] David Moro, Ngoc Cuong Nguyen, and Jaime Peraire. Dilation-based shock capturing for high-order methods. *International Journal for Numerical Methods in Fluids*, 82(7):398–416, 2016.

[29] Per-Olof Persson and Jaime Peraire. Sub-Cell Shock Capturing for Discontinuous Galerkin Methods. In *44th AIAA Aerospace Sciences Meeting and Exhibit*, 2006.

[30] Jianxian Qiu and Chi-Wang Shu. Runge-Kutta Discontinuous Galerkin Method Using WENO Limiters. *SIAM Journal on Scientific Computing*, 26(3):907–929, 2005.

[31] Deep Ray and Jan S. Hesthaven. An artificial neural network as a troubled-cell indicator. *Journal of Computational Physics*, 367:166–191, 2018.

[32] Boris L. Rozhdestvenskii. Discontinuous solutions of hyperbolic systems of quasilinear equations. *Russ. Math. Surv.*, 15(53), 1960.

[33] Keith Rudd and Silvia Ferrari. A constrained integration (CINT) approach to solving partial differential equations using artificial neural networks. *Neurocomputing*, 155:277–285, 2015.

[34] Carsten W. Schulz-Rinne. Classification of the Riemann problem for two-dimensional gas dynamics. *SIAM J. Math. Anal.*, 24(1):76–88, 1993.

[35] Chi-Wang Shu. Discontinuous Galerkin Methods: General Approach and Stability. *Numerical solution of partial differential equations*, pages 149—-202, 2009.

[36] Chi-Wang Shu and Stanley Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes, II. *Journal of Computational Physics*, 89:32–78, 1989.

[37] Gary A. Sod. A Survey of Several Finite Difference Methods for Systems of Nonlinear Hyperbolic Conservation Laws. *Journal of Computational Physics*, 27:1–31, 1978.

[38] Z. J. Wang, Krzysztof Fidkowski, Rémi Abgrall, Francesco Bassi, Doru Caraeni, Andrew Cary, Herman Deconinck, Ralf Hartmann, Koen Hillewaert, Hung T. Huynh, Norbert Kroll, Georg May, Per Olof Persson, Bram van Leer, and Miguel Visbal. High-order CFD methods: Current status and perspective, 2013.

[39] Jian Yu and Jan S. Hesthaven. A comparative study of shock capturing models for the discontinuous Galerkin method. *EPFL-Article*, 231188:1–42, 2017.

[40] Jun Zhu, Jianxian Qiu, Chi Wang Shu, and Michael Dumbser. Runge-Kutta discontinuous Galerkin method using WENO limiters II: Unstructured meshes. *Journal of Computational Physics*, 227(9):4330–4353, 2008.

[41] Valentin Zingan, Jean Luc Guermond, Jim Morel, and Bojan Popov. Implementation of the entropy viscosity method with the discontinuous Galerkin method. *Computer Methods in Applied Mechanics and Engineering*, 253:479–490, 2013.