Journal of Computational Physics 423 (2020) 109824

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

A neural network based shock detection and localization approach for discontinuous Galerkin methods

Andrea D. Beck [a,*,1], Jonas Zeifang [a], Anna Schwarz [a], David G. Flad [b]

[a] Institute of Aerodynamics and Gas Dynamics, University of Stuttgart, Stuttgart, Germany
[b] NASA Ames Research Center, Moffett Field, CA, USA

ARTICLE INFO

Article history:
Received 20 January 2020
Received in revised form 22 August 2020
Accepted 30 August 2020
Available online 9 September 2020

Keywords:
Shock indicator
Shock capturing
High order schemes
Machine learning
Neural networks
Discontinuous Galerkin

ABSTRACT

The stable and accurate approximation of discontinuities such as shocks on a finite computational mesh is a challenging task. Detection of shocks or strong discontinuities in the flow solution is typically achieved through a priori troubled cell indicators, which guide the subsequent action of an appropriate shock capturing mechanism. Arriving at a stable and accurate solution often requires empirically based parameter tuning and adjustments of the indicator settings to the discretization and solution at hand. In this work, we propose to separate the task of shock detection and shock capturing more strongly and aim to develop a shock indicator that is robust, accurate, requires minimal user input and is suitable for high order element-based methods like discontinuous Galerkin and flux reconstruction methods. The novel indicator is learned from analytical data through a supervised learning strategy; its input is given by the high order solution field, its output is an element-local map of the shock position. We use state of the art methods from edge detection in image analysis based on deep convolutional multiscale networks and deep supervision to train the indicators. The resulting networks are then used as black box indicators, showing their robustness and accuracy on well established canonical testcases. All simulations are run ab initio using the developed indicators, showing that they provide also stability during the strongly transient phases. In particular for high order schemes with large cells and considerable inner-cell resolution capabilities, we demonstrate how the additional accurate prediction of the position of the shock front can be exploited to guide inner-element shock capturing strategies.

1. Introduction

1.1. Problem definition and state of the art

The occurrence of strong compression waves in compressible fluid flow is a well-known and characteristic phenomenon [8]. Mathematically, these shocks are an expression of the non-linearity of the underlying governing equations (e.g. the Euler equations), which can produce discontinuous solution even for initially smooth data [56]. This possibility for discontinuities often clashes with the discretization strategy for the numerical approximation of these flows, where a (at

* Corresponding author.
E-mail address: beck@iag.uni-stuttgart.de (A.D. Beck).
[1] A. Beck and J. Zeifang agree to share first authorship.

https://doi.org/10.1016/j.jcp.2020.109824
0021-9991/

least locally) smooth solution representation is sought. Discretization strategies with ansatz spaces that include discontinuities like low order finite volume (FV) schemes are naturally better equipped to handle shocks, leading to the dominance of FV and related schemes for compressible aerodynamics, but encounter the same mismatch for orders greater than one [35].

Based on the rationale that the solution is predominantly smooth except for limited regions, high order discretizations are often favored and have gained considerable interest for practical applications. These schemes provide low approximation errors and excellent scale resolving capabilities, leading to efficient methods in particular for smooth problems [68]. In regions with discontinuities, these schemes then encounter spurious oscillations, often called Gibb's instability [14], which can lead to a loss of stability and accuracy. Avoiding instabilities and treating these solutions in a numerically stable and accurate way is the task of the so-called shock capturing methods. A conceptually different approach, labeled shock-fitting, is significantly less widespread due to its complexities on unstructured grids [46].

Depending on the baseline discretization scheme chosen, a number of different shock capturing methods exist, for example, for finite volume discretizations, a TVD limiting strategy or an essentially non-oscillatory (W)ENO reconstruction are typical [49,63,62,36]. For the class of flux reconstruction (FR), spectral difference (SD) and discontinuous Galerkin (DG) schemes which are the focus of this work, the methods typically fall into two categories: In the first category, once a shock is detected, a local diffusion operator of prescribed strength is added to regularize the solution. This strategy is often called the "artificial viscosity shock capturing" method [50,42,13,3,48,16]. In the second category, the discretization operator is switched *locally* to a more or guaranteed robust one, for example, through h-refinement and/or p-coarsening or a reconstruction procedure. By appropriate limiting, this operator may then additionally ensure the boundedness of the solution. We will refer to these methods rather broadly as "h/p-shock capturing" in the following. Examples of the combination of high order discretization schemes with these shock capturing methods can, for example, be found in [7,6,73,52,12,51,1,75]. Before applying any of these strategies however, the occurrence and location of a shock must be detected. This, in itself, is a non-trivial task and is typically achieved through *parameter-dependent indicator functions* which are based on physical considerations [22], modal smoothness estimates [20,25,48] or image detection ideas [61]. So at least in theory, in this scenario, the shock detection task can be decoupled from the numerical solution stabilization mechanisms. However, in practical applications, this is most often not the case. Instead, the parameters of the indicator, which informs the local shock capturing action of the scheme, are tuned to provide a stable simulation result. Thus, the indicator not primarily signals a discontinuity but is tuned to *predict* the occurrence of an underresolved state that will likely lead to an invalid solution of the discretized PDE in an a priori fashion. In this way, the "correct" parameter settings for each indicator become dependent not only on the underlying flow field but also heavily on the discretization and its robustness. This dilemma is reflected in the fact that indicators used in this setting are often labeled *troubled cell* indicators, where the potential trouble is not the existence of a sharp gradient or discontinuity in the solution but the *failure of the current discretization to deal with it robustly*. In our presented approach here, we return to the original separation of tasks.

A posteriori methods are less frequently applied, a recent example can be found in [12]. Here, the occurrence of an unstable solution in troubled cells is detected after the corresponding time step. The solution is then rolled back to a previous, stable state, and a more robust discretization, typically a lower order scheme, is chosen to recompute the solution in a recursive manner until a stable solution is achieved. Within this method, the number of roll-back steps and the shock detection criteria are tuneable parameters. Their optimal values are not known a priori and need to be guided by the user.

As can be surmised from the discussion so far, the detection (or prediction) of troubled cells from the solution of the discretized PDE usually contains at least some level of empiricism and parameter tuning. This somewhat unsatisfactory state results from the complexity of the problem due to the inherent non-linearities in the equation itself and its discretization. Given the rise of the interest in data-driven methods in fluid mechanics with applications to a range of non-linear problems (e.g. subspace model design, active control [4,53]), it is not surprising that these methods have also been investigated for the problem at hand. Before presenting our approach, we thus briefly survey the existing literature on shock localization and solution stabilization with data-based approaches and machine learning techniques in the next section.

1.2. Shock detection and capturing with data-based methods

In this section, we present a brief summary of shock detection and localization methods with a focus on data-based approaches. The initial concept of shock detection based on image analysis as well as application examples of a classical Sobel filtering can be found in [58,67]. In [37], the authors propose a staged approach, where step edges are located through an approximation of second order derivatives of the pressure field. In the regions of interest, a shock function is then evaluated, which incorporates physical considerations like Mach number and pressure gradients. Contours of density gradient or normal Mach number was also used successfully in [47,40]; a collection of results of image-based methods can be found in [71]. A recent example of shock detection based on modern neural network architectures is presented in [44], where the network was trained to predict the Schlieren value from local velocity strain data; an improved method with higher detection accuracy was shown in [39]. While the previous examples only consider shock detection, there have also been some efforts to combine data-based methods for combined shock detection / shock capturing, i.e. as troubled cell indicators. In [15], the authors classify a troubled cell as one in which the unlimited solution deviates by a user-specified metric from one limited by the methods proposed in [28]. They then train a multilayer perceptron (MLP) as a classifier to predict the occurrence of troubled cells for 1D advection and Euler equations. While their network is capable of essentially mimicking the underlying limiter from [28], the number of neurons and the associated cost of the network are substantial.

In [54,55], the authors develop an MLP based classifier to distinguish smooth and non-smooth regions of the flow and show its robustness by applying it to a range of test problems. An extension of the classifier towards the prediction of an artifical viscosity is presented in [74]. Recently, Morgan et al. also proposed an MLP based shock detector for a high order finite element scheme [45].

1.3. Motivation and outline

In the approach considered here, we return to the original separation of tasks for shock capturing: shock detection and solution stabilization are considered two different, *independent* entities, and we make the choice of treating them as such. For the second task, we rely on a guaranteed stable approximation and employ a robust finite volume formulation with TVD limiters on inner-element sub-cells [65,27], but note that other solution stabilization methods, e.g. artificial viscosity based ones, are equally possible. For the purpose of this work, we will treat this shock capturing method as a black box with fixed settings and not attempt any parameter tuning. We will instead focus on the first task: Reliably and accurately detecting the occurrence of shocks or steep gradients in an a priori manner based on the current solution alone without the need for parameter adjustment. In order to do this, we propose a novel strategy for developing shock location sensors based on ideas from machine learning, in particular edge detection. We follow the *supervised learning* paradigm, where a (computationally expensive) offline training process based on labeled data produces a (cheap) approximation to the underlying mapping between input and output data. In the context of edge detection, we interpret the flow solution data within one grid element as the image and a possible shock or strong gradient as an edge to be detected and localized within this image. Thus, the input data into our method consists of the conservative solution variables in a given mesh cell, the output is whether a shock is present or not and where it is located. The mapping from input to output data itself is expressed as a convolutional neural network (CNN). The approach we propose is particularly attractive for high order schemes like DG, FR or SD schemes, as it takes advantage of the full high order information contained in a single grid element. The presented novel indicators offer the advantage of being parameter-free and do not require tuning by the user. We will demonstrate their applicability to a range of canonical flow problems. In addition, our results will show that they are highly insensitive towards changes in grid resolution. Finally, we demonstrate that with a slight modification of the network training, the resulting indicators can now be used not only to flag cells that contain shocks but also to locate the shock precisely within a cell - this local information can then inform the subsequent shock capturing strategy and help target it precisely to where it is needed. Again, this information is particularly beneficial for high order schemes with typically rather large grid cells, where a localized shock capturing on the scale of the smallest resolved length scale (roughly estimated in 1D as $\Delta x/p$, where Δx is the length of the considered cell and p is the degree of the polynomial ansatz) is desirable. While we demonstrate the potential benefits of this approach in the context of the h/p-shock capturing with the chosen local finite volume based solution stabilization method, it can readily be transferred to other approaches and could for example inform a local artificial viscosity approach or guide element-local r-adaptation.

For the remainder of this paper, we refer to the task of detection whether a shock exists in a grid cell or not as “shock detection” and the corresponding developed method as ANNSI (artificial neural network based shock indicator); the task of locating the exact position of the shock front within an element is called “shock localization” and our proposed method ANNSL (artificial neural network based shock localization).

We will proceed as follows: In Sec. 2, we introduce the numerical method employed in this work and the present shock capturing method used later in a black box manner. We also review two classical troubled cell indicators which serve as reference for the evaluation of the ANNSI results. In Sec. 3, a short summary on neural networks is given, we define the training data and the network architectures used for the ANNSI and ANNSL indicators, and we close the section by reporting the results of the training process. In the following section (Sec. 4), we apply the novel ANNSI indicator to well known test problems and compare the results with two classical troubled cell indicators. Once we have established the accuracy and robustness of the ANNSI indicator, we focus on its in-build extension to localize shocks within the grid elements. In Sec. 5, we therefore apply the ANNSL indicator to several test problems and show how the information about the inner cell shock position can be exploited. Finally, in Sec. 6, we conclude and give an outlook.

2. Numerical method

2.1. Governing equations

In this work, we consider the usual compressible Euler equations of fluid dynamics which are given by

$$\underbrace{\partial_t \begin{pmatrix} \rho \\ \rho \mathbf{u} \\ E \end{pmatrix}}_{=:\mathbf{w}} + \nabla \cdot \underbrace{\begin{pmatrix} \rho \mathbf{u} \\ \rho \mathbf{u} \otimes \mathbf{u} + p\, \mathbf{I} \\ \mathbf{u}\,(E+p) \end{pmatrix}}_{=:\mathbf{F}(\mathbf{w})} = 0, \tag{1}$$

where ρ denotes density, $\mathbf{u}$ velocity, E energy density and the pressure p is computed via the equation of state for a perfect gas

$$p(\rho, \rho\mathbf{u}, E) := (\gamma - 1)\left(E - \frac{1}{2}\rho\|\mathbf{u}\|_2^2\right),$$

with $\gamma = 1.4$ being the isentropic expansion coefficient.

2.2. High order discontinuous Galerkin spectral element method

We discretize the equations (Eq. (1)) with a Discontinuous Galerkin Spectral Element Method (DGSEM) [26,18,27], offering the possibility of choosing arbitrary high approximation order. DGSEM is based on the weak formulation for a piecewise smooth function $\mathbf{w}_h$

$$\frac{\partial}{\partial t}\int_E \mathbf{w}_h\,\phi(\mathbf{x})\,d\mathbf{x} + \oint_{\partial E}\mathbf{F}_n^*\phi(\mathbf{x})\,ds - \int_E \mathbf{F}(\mathbf{w}_h)\cdot\nabla_x\phi(\mathbf{x})\,d\mathbf{x} = 0, \tag{2}$$

on every element E for every polynomial test function $\phi(\mathbf{x})$ of degree $\mathcal{N}$. Note that all elements and thus Eq. (2) are mapped onto a reference space $[-1,1]^d$. DGSEM uses an inner-element solution representation based on the tensor product of nodal Lagrange basis functions and couples neighboring elements weakly through a numerical flux $\mathbf{F}_n^*$. Choosing the same $\mathcal{N}+1$ Legendre-Gauss points for quadrature and interpolation reduces the number of required operations per degree of freedom. With the restriction to tensor-product elements, the total number of degrees of freedom per grid cell results in $(\mathcal{N}+1)^d$. The final discrete d-dimensional spatial operator can be constructed through multiple one-dimensional operations. For the remainder of this text, we limit our discussion to 2D problems, i.e. $d = 2$. According to the method of lines, integration in time is then achieved by any suitable explicit or implicit method. For further details the reader is referred to [27], [5] and [18]; the full code framework is available as open source software.[2]

2.3. Shock capturing for the DGSEM

As stated earlier, high order methods tend to produce oscillations near discontinuities, which conflict with the locally smooth approximation space. Thus, a form of stabilization or regularization has to be introduced. In this work, we apply a finite volume sub-cell scheme [64] at elements which contain a discontinuity. In these cells, the polynomial solution representation with $(\mathcal{N}+1)^d$ degrees of freedom is projected onto $(\mathcal{N}+1)^d$ equidistantly distributed piecewise constant sub-cells, i.e. an equivalent FV solution representation is constructed. The projection is realized through the Vandermonde matrix $\mathbf{V}_{\mathrm{FV}}$. For each sub-cell, a second order TVD finite volume scheme is applied, and the coupling between discontinuous Galerkin elements and finite volume elements is achieved in a natural way via numerical flux functions. This type of shock capturing introduces a stable representation of discontinuities through the reduction of the order of the solution representation (p-adaptation), while at the same time preventing a significant loss in resolution by increasing the local element numbers through the introduction of sub-cells (h-adaptation). If a cell no longer contains a discontinuity, it is switched back to the DG representation by multiplication with $\mathbf{V}_{\mathrm{FV}}^{-1}$. Note that with regards to the discussion in Sec. 1.1 and 1.3, we treat this shock capturing scheme as a black box and use it as is for the remainder of this work.

2.4. Shock indicators

To indicate when to switch from/to the polynomial representation of the discontinuous Galerkin method to/from the piecewise constant finite volume representation, a shock detection function is required. Besides the novel shock indicator introduced later in Sec. 3, a wide range of indicators exits which can be used to identify elements containing a discontinuity, see e.g. [10,11,20,22,48] and the discussion in Sec. 1.1. Typically the indicators rely on the definition of a user-defined value against which the calculated indicator value can be compared. Due to the interactions of solution and discretization, these values are specific to a certain setup. For our calculations, we define an upper and a lower threshold introducing a delay in the switching procedure to avoid an oscillatory behavior if the indicator value almost matches the threshold. This means that a cell is only switched to the finite volume representation if the indicator value is greater than the upper threshold $\mathcal{I}_{\mathrm{upper}}$ and is switched back to the polynomial representation if the indicator value is below the lower threshold $\mathcal{I}_{\mathrm{low}}$. Fig. 1 depicts a typical evolution of the indicator and its thresholds.

We compare our ANNSI indicator against two well-known formulations from literature:

- Indicator based on a modal smoothness analysis in analogy to [20,48] taken from [64]:
  Given the solution in modal polynomial space, this indicator evaluates the relative contribution of the highest modes for the definition of the indicator value $\mathcal{I}_{\mathrm{modal}}$.

[2] www.flexi-project.org, GNU GPL v3.0.

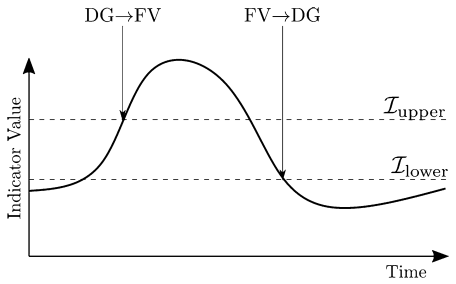


Fig. 1. Indicator based switching for shock capturing in the hybrid DG/FV sub-cell method.

$$\mathcal{I}_{\text{modal}} = \log_{10}\max\left\{\frac{([\mathbf{w}_{\text{modal}}]_i^i, [\mathbf{w}_{\text{modal}}]_i^i)_{L_2}}{([\mathbf{w}_{\text{modal}}]_0^i, [\mathbf{w}_{\text{modal}}]_0^i)_{L_2}}, i = \mathcal{N}-2, \mathcal{N}-1, \mathcal{N}\right\},$$

with $(\cdot,\cdot)_{L_2}$ being the L_2 inner product and the truncation to modes between a and b being

$$[\mathbf{w}_{\text{modal}}]_a^b = \sum_{i=n(a)-1}^{n(b)-1} \mathbf{w}_{\text{modal},i}\psi_i,$$

where $n(b)$ denotes the total number of the Legendre basis functions ψ_i for the polynomial degree b.

- Indicator based on the jump inside an element inspired by [22] given in [64]:
  For two dimensional simulations the indicator value can be calculated as

$$\mathcal{I}_{\text{jump}} = \frac{1}{V_E}\sum_{i,j=0}^{\mathcal{N}} \frac{w_{\min,ij} - 2w_{ij} + w_{\max,ij}}{w_{\min,ij} + 2w_{ij} + w_{\max,ij}} V_{ij},$$

where V_E and V_{ij} denote the volume of the element and the finite volume sub-cell with the index i, j, respectively. With w being a user defined variable of $\mathbf{w}$ e.g. pressure or density, $w_{\min,ij}$ is defined as $w_{\min,ij} = \min(w_{i\pm id, j\pm jd}, d = 1, 2)$, denoting the minimal value of the chosen variable neighboring the current degree of freedom. $w_{\max,ij}$ is defined in an analogous way.

3. Neural networks design, data generation and training

As outlined in Sec. 1.3, our goal is to develop shock detection and localization indicators through methods from supervised machine learning, in particular image-based edge detection methods. Details on these methods and their training process will be given in this section. Before discussing the details, it is helpful to outline the steps: For the sake of completeness, we briefly introduce general neural networks in Sec. 3.1. For the specific task of edge detection, we then discuss the holistically nested edge detection (HED) method in Sec. 3.2 which improved the state of the art when it was introduced in 2015 and is still considered among the best edge detection methods, and present our designed variant of HED. We discuss how we employ these networks as indicators for both shock detection and shock localization. The data set for the supervised learning of variants suitable for DG methods is presented in Sec. 3.3, followed by information on the training process and the achieved results in Sec. 3.4.

3.1. Function approximation via neural networks

The number of possible artificial neural network (ANN) architectures is large and currently growing [66]. We will restrict our discussion herein to the common forms of multilayer perceptrons and convolutional networks. In their most general form, ANNs are multivariate compound functions that map an input vector $\mathbf{X}$ to an output vector $\mathbf{Y}$. The mapping itself consists of a sequence of linear and non-linear functions with adjustable parameters. The common visual representation of a neural network as a directed acyclic graph (DAC) encodes the way in which these functions are stacked together, see Fig. 2 for an example of an MLP network. Here, arrows between two adjacent layers of neurons represent the entries of the linear weight matrices, with subsequent non-linear activation functions applied to each neuron before feeding into the next layer. In supervised learning, the weights of the network are adjusted through incremental learning from labeled training data, where the approximation quality of the neural network is evaluated by a cost function. The gradient of the cost function w.r.t. the parameters then guides the optimization process to determine the best functional fit to the data. Under mild assumptions on the underlying mapping and the architecture of the network, neural networks with a single or more hidden layers containing a finite number of neurons, can be shown to be universal function approximators [2,9,19,41]. For more details on neural networks, their properties and the training process in particular for deep learning, we refer the

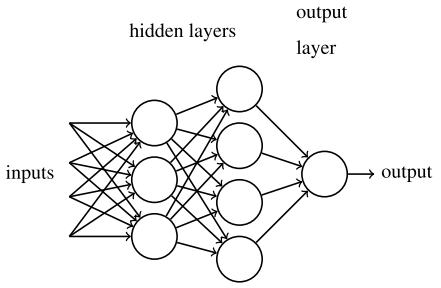


Fig. 2. DAC of a classical MLP neural network.

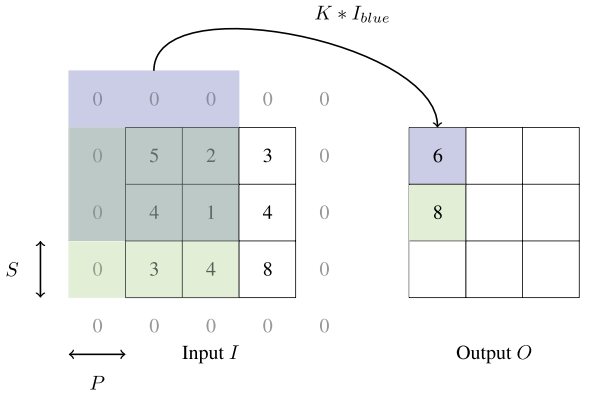


Fig. 3. Illustration of a convolutional operation. The filter kernel K is of size 3×3, the colors indicate the corresponding filter input and output neurons. For demonstration purposes, the kernel is chosen the identity matrix, and for completeness stride (S) and padding (P) are chosen as $S = P = 1$. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

reader to [17,59]. An alternative architecture to the multilayer perceptrons with dense layer connections are the so-called convolutional neural networks. They were originally developed for computer vision tasks [32,33,29]. This architecture retains the layer structure but replaces the dense matrix connections between the individual layers by a local discrete convolution of the input $I \in \mathbb{R}^{i_h \times i_w}$ with a filter $K \in \mathbb{R}^{k_h \times k_w}$ (see Fig. 3)

$$K * I(i,j) := \sum_{m=-k_h}^{k_h} \sum_{n=-k_w}^{k_w} K(m,n) I(i + S \cdot m - P, j + S \cdot n - P), \tag{3}$$

for $i, j = 1, \cdots, (i_{h,w} - k_{h,w} + P + S)/S$ with padding P and stride S. The padding describes the width of an artificially added zero layer around the image and the stride defines the step size when moving the filter kernel. This local convolution is equivalent to the componentwise inner product of kernel and input: $K * I = tr(KI^T) = tr(IK^T)$ Due to the locality of the kernel matrix, each neuron in a layer is only connected to a local region of the input, called receptive field. The entries of these filter kernels K are then the parameters to be optimized. These trainable, local filter functions equip the CNNs with some favorable properties over the classical MLPs. First, the sparseness of the inter-layer connection greatly reduces the amount of parameters, making training of large networks feasible. Secondly, due to the local convolution, they are invariant to spatially shifted inputs. As one can interpret a CNN as a method of finding a suitable reduced order basis based on data snapshots, it is not surprising that CNNs have shown to be particularly powerful when the underlying data has a hierarchical structure, as for example in image recognition and segmentation [30] or turbulence modeling [4].

3.2. Neural network architecture

Due to these favorable properties of CNNs, they are widely used in edge detection (and other image-based tasks). As we interpret the task of shock detection and localization as one from this field, our network design is inspired by the holistically-nested edge detection (HED) network from [72]. We use the same architecture for both tasks, i.e. the network design for ANNSI and ANNSL is *identical*. Differences between ANNSI and ANNSL exist only in the training data and will be discussed later on.

3.2.1. Edge detection through HED

The detection of the position and shape of a shock wave within an element is akin to edge detection in images in that a steep gradient or discontinuity separates the original image into distinct regions. We thus use ideas from this area to

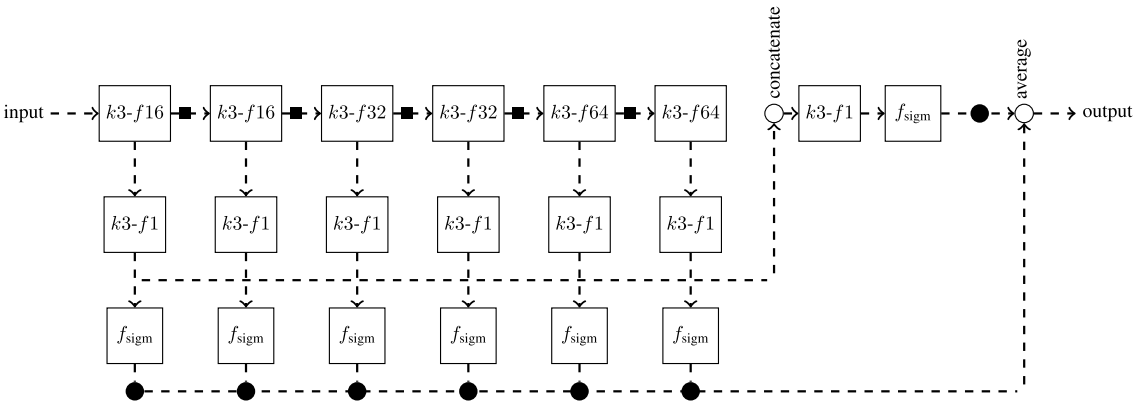


Fig. 4. Network architecture used for the shock detection (ANNSI) and localization (ANNSL). Each convolution layer has a kernel size k and f channels. Layers with $f = 16, 32, 64$ use LReLU as activation function; in layers with $f = 1$ no activation function is used. Black dots symbolize positions where the output is taken for the evaluation of the loss function C and squares symbolize batch normalization. The concatenate operator takes all side outputs as channels to create a convolution-like layer (without learnable parameters), which is followed by a convolutional layer and a sigmoid function. The average function computes the mean of this output and all side outputs (with activation function).

generate a network capable of localizing shocks in the 2D "image" made up by the solution points in each element. One particularly successful approach that has advanced the state of the art on reference data set is the HED algorithm proposed in [72]. It has a range of properties that make it suitable for the task at hand. The first one is indicated by the "holistic" attribute, which refers to the fact that HED outputs an image of the same size as the input, with the edges marked on it. This is directly useful for the solution representation and the inner-element interpolation points that exist in element-based high order discretizations. The second feature, covered by the term "nested", refers to the architecture of the HED network. First of all, it is based on a convolutional neural network design which makes it shift-invariant, which is clearly desirable for shock localization. Although the kernel size is limited in each layer of these networks and thus the corresponding neuron does not encounter the full input picture, the total receptive field of each layer increases along the DAC, as the region of influence (seen by a neuron through all the layer) increases. Thus, different layers act on different scales of the input in a local-to-global manner. This multiscale property of CNNs is used in HED by also considering the side outputs of each layer in the cost function (and thus their parameters in the network training), recognizing that an edge should be scale-invariant. Each layer is thus thought of as a singleton network with the task of locating the edge at the given scale. This notion in HED was inspired by deeply-supervised nets [34]. In addition to supervision, the predictions of the sideoutputs are combined in a weighted manner and fused with the overall network output, again taking into account the scale invariance of edges. Although HED incorporates these powerful features, its actual network design can easily be derived from standard deep CNNs and is easy to describe and implement.

3.2.2. ANNSI/ANNSL architecture

While the network design is inspired by HED, our task at hand differs from the intended applications in terms of scale: While in edge detection in images, the input image is typically of size $\mathcal{O}(100)^2$ pixels (disregarding the color channels; they represent the RGB colors of the image) or even larger, in our intended application, it is of size $(\mathcal{N}+1)^2$, which, for typical $\mathcal{N} < 10$. Thus, the network design itself has been adjusted to reflect this much smaller input space.

An overview of the network architecture used in this work is provided in Fig. 4. At its core, the upper lane through the network is a standard deep CNN architecture. The side outputs (middle and lower lane) compute the output predictions of each layer of the first lane, i.e. so-called edge maps with the same size as the input image. Information from all the edge maps is fused with the final output, thus taking advantage of the scale invariance of edges.

For the specific network used in our context and shown in Fig. 4, we use the following technical details: Our network is build of multiple convolutional layers with an increasing number of channels over the DAC to extract more complex information. A channel is a convolutional layer (in a set or stack of these layers) which has a unique filter but shares the same receptive field with the other channels. In layers with $f = 16, 32, 64$ channels, we use a *leaky Rectified Linear Unit* [43] (LReLU)

$$f_{\mathrm{LReLU}}(z) = \max(0, z) + \alpha \min(0, z) \quad \text{with} \quad \alpha = 0.2,$$

as activation function, whereas in layers with $f = 1$ no activation function is applied as they are connected to a *sigmoid* function

$$f_{\mathrm{sigm}}(z) = \frac{1}{1 + e^{-z}}.$$

As common, we use batch normalization [21] to enhance the training process and use padding $P = 1$ and stride $S = 1$ to keep the size of the forward and side outputs constant for the different layers, which is required for the "holistically-nested"

approach and hence all outputs have the same size as the input image, i.e. $(N+1)^2$. The filter kernel size is chosen to be 3, which we found to work best for the limited size of the input image. In total, the network used has 72477 trainable parameters.

3.3. Training data

For the training of our neural network, we stick to the *supervised learning* strategy, see e.g. [31]. Under the assumption that there exists a functional dependency $\mathbf{f}:\mathbf{X}\mapsto\mathbf{Y}$ of the input vector $\mathbf{X}$ and the true desired output $\mathbf{Y}$, this paradigm states that one tries to find an approximation $\hat{\mathbf{f}}:\mathbf{X}\mapsto\hat{\mathbf{Y}}$, such that the difference between the truth $\mathbf{Y}$ and the predicted output $\hat{\mathbf{Y}}$ is minimal. The metric for this difference is called cost function $C:=C(\mathbf{Y},\hat{\mathbf{Y}})$, and a single evaluation of $\hat{\mathbf{f}}:\mathbf{X}\mapsto\hat{\mathbf{Y}}$ given the current mapping encoded in the network is called a forward pass. During the backward pass, the gradient of the cost function w.r.t. all the parameters of the network is computed, and the weights are updated following a gradient descent optimization [57,69]. Often, this weight update is computed not for single training samples (or all of them), but for so-called *mini batches* [21]. The size of these mini batches is chosen empirically and should provide a compromise between convergence rate, stability, memory demands and accuracy; one training iteration over all training samples is called an epoch. The magnitude of the weight increment depends on a scaling parameter termed *learning rate*. Since the sought mapping is typically non-convex with respect to the parameters of the network, neural networks are considerably data-hungry in their training process.

3.3.1. Training data for ANNSI

For the training of a shock detector, this requires the definition of a sufficiently broad spectrum of input images $\mathbf{X}$ and a corresponding label indicating the true desired output image $\mathbf{Y}$ for each date. Note that since the network architecture is of convolutional type, the input into the network are not restricted to vectors. On the contrary, 2D (or higher dimensional volumetric data) are naturally suited as inputs and outputs of these nets. Thus, with our chosen limitation to 2D problems, the input consists of square images with $(\mathcal{N}+1)^2$ pixels, where each pixel corresponds to a nodal degree of freedom of the DG approximation. In other words, in true image-based analysis fashion, we interpret the solution within an element as an image and the solution points as its pixels.

Inspired by [54,55], we use a set of smooth and non-smooth analytical functions to represent possible solutions of a simulation with the DGSEM and define on the analytic level if and where a shock is present in the current grid cell. We map all physical elements to reference space in $[-1,1]^2$, thus, for training, all inputs and outputs are given on the *same reference square*. The analytical functions given in Table 6 in Appendix A are evaluated on Cartesian meshes covering the interval $[-1,1]^2$ with n_e^2 elements. A class label of 0 refers to a no-shock solution, a label of 1 to a shock solution. In each of the n_e^2 elements, the function is approximated by the tensorproduct basis with $(\mathcal{N}+1)^2$ solution points located at the Legendre Gauss nodes as described above. For the case of $n_e=1$, the domain coincides with the reference element $[-1,1]^2$, for other choices of n_e, the resulting solution is mapped back to the reference element. This provides an efficient and easy way to introduce the effect of different spatial resolutions into the training process.

With this information, we define the true desired output $\mathbf{Y}$, which is a so-called *binary shock edge map*, having the same size as the input image $\mathbf{X}$. The entries of the edge map are 1 for a degree of freedom / solution point being located at the discontinuity, otherwise it is labeled as 0. However, the analytical position of the shock does not generally coincide with the location of the solution points, instead, it will typically pass between two adjacent points. In this case, the two solution points adjacent to the shock front per each coordinate direction are given the 1 label. This is akin to the definition of a shock bounding box on a sub-element level. Thus, labeled true data $\mathbf{Y}$ for each input $\mathbf{X}$ consists of the binary edge shock map of size $(\mathcal{N}+1)^2$, where each entry is in $[0,1]$ to indicate that pixel being associated with a shock front or not. A class label (element contains a shock or not) can be derived from this for the whole element by 1 if any$(\mathbf{Y})=1$, else 0. However, as the outputs of the network are connected to a sigmoid function, the predicted output $\hat{\mathbf{Y}}$ ranges from 0 to 1. A threshold of 0.5 is chosen to obtain a binary shock edge map, i.e. the output is set to 1 if $\hat{\mathbf{Y}}>0.5$ and 0 otherwise.

The total amount of data chosen from the different classes are summarized in Table 1. The number of training cells with and without shocks are approximately balanced to enhance training. Note since the input tensors are dependent on the choice of the polynomial degree and to take the different spatial resolution capabilities into account, we choose to train a specific network for each $\mathcal{N}$.

Remark 1. In the presented approach, the shock detection per element is achieved by training the network to predict the binary edge map within an element and decide that a shock is present if any of the pixels in the edge map show 1. This might seems as a waste of information and training, as the predicted matrix is condensed to a single bit per cell. We choose this route for a number of reasons:

- As discussed in Sec. 1.2, a number of publications report useful and accurate shock detectors based on MLP architectures. In our investigations however, the results obtained for shock detection with the ANNSI approach showed greater accuracy and robustness than training an element-wise classifier directly. In addition, we are also interested in the inner-cell shock location, see next item.

Table 1
Number of samples chosen from the different analytical functions in the training and validation sets for the polynomial degrees a) $\mathcal{N} = 5$ and b) $\mathcal{N} = 9$ for the shock detector ANNSI.

#	Training set		Validation set	
	class 0	class 1	class 0	class 1
1	20311	0	2046	0
2	66288	0	6638	0
3	6402	0	619	0
4	51	37266	6	3758
5	2377	16297	247	1618
6	102	32308	9	3206
7	59	18551	8	1853
	95590	104422	9573	11530

(a) $N = 5$

#	Training set		Validation set	
	class 0	class 1	class 0	class 1
1	20400	0	2388	0
2	66400	0	7289	0
3	6400	0	1195	0
4	40	40023	95	6494
5	2067	15660	251	2184
6	116	31094	21	4206
7	80	18520	2	2938
	95503	105297	11712	15822

(b) $N = 9$

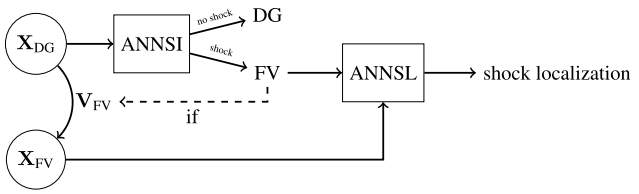


Fig. 5. Schematic overview of the workflow for shock localization at each timestep.

- While we choose to condense the edge map to a single value, its full information is available during the computation. For shock capturing schemes which do not change the localization of the solution nodes, this information can directly be used for shock localization. Such schemes are e.g. artificial viscosity approaches (see e.g. [3]) or sub-cell shock capturing on Legendre-Gauss nodes [65]. Thus, our choice of input and output data provides flexibility in using the full edge map information in ways that fit the chosen shock capturing approach.
- It avoids having to develop two different neural network architectures, thereby reducing the complexity of the approach.

3.3.2. Training data for shock localization (ANNSL)

As discussed in the previous section, the ANNSI network outputs a prediction of a shock map within each element, where the pixels correspond to the position of the Legendre Gauss solution points of degree $\mathcal{N}$ that host the DG solution polynomial. In theory, based on this information, any shock capturing scheme could be guided to improve the inner cell treatment of shocks. In the following, we discuss our strategy for exploiting this information in the context of our hybrid DGSEM / FV scheme: Keeping the network architecture and input functions fixed, we re-train the network but shift the position of the input and output pixels from $(\mathcal{N}+1)^2$ Legendre Gauss nodes to $(\mathcal{N}+1)^2$ equispaced nodes. This just requires evaluating the functions from Table 6 on the new node distribution. The reason for this shift lies in a peculiarity of the chosen shock capturing strategy described in Sec. 2.3: The sub-cell finite volume scheme is formulated on an equispaced grid, not the Legendre Gauss grid. With increasing degree $\mathcal{N}$, the discrepancy between the equidistant and Legendre Gauss points increases, and local correspondence is lost. This makes transferring positional information from the FV to the DG grid and vice versa less accurate and robust. In other words, locating the shock on the DG grid is not very useful for shock capturing on the equispaced grid. For this reason, we train the ANNSL network directly on the FV grid.

In summary, our strategy for shock detection and localization works as follows: First, all cells, including elements with FV sub-cell representation, are treated as DG cells with the corresponding solution points, and the ANNSI network (trained on Legendre Gauss nodes) acts as a shock detector, i.e. ANNSI decides whether a cell is a troubled cell or not. Cells that are marked by ANNSI as shock-free remain DG elements for the next timestep and are not considered by the ANNSL network. Those that are marked as containing shocks are treated by the FV shock-capturing scheme: The solution is projected onto the FV sub-cell space using the Vandermonde matrix $\mathbf{V}_{FV}$. Then, the ANNSL network is used to predict the shock location within those cells. This procedure is repeated in the next timestep and all following. Fig. 5 provides a schematic overview of this process.

We also note that it is possible to directly apply the ANNSL to the FV representation in all elements of the domain without pre-selection by ANNSI: This gives slightly less robust, however comparable results but requires the computational effort of computing the projection onto the FV space in all cells at each timestep. Since this approach however makes the initial ANNSI network superfluous, it can result in an overall speed-up. Since our main focus and motivation in this work is on accuracy and robustness of the approach and not so much on efficiency, we did not investigate this in greater detail.

To summarize our training rationale, we train two identical networks for the prediction of a binary edge map on two different sets of input data: the DG solution points (ANNSI) and equispaced FV (ANNSL) solution points. Both networks thus provide information about the inner-cell shock position by marking the associated nodes / pixels. The choice to retrain the network on equispaced points is solely driven by the specific shock capturing scheme employed here. We choose to use the

Table 2
Number of samples chosen from the different analytical functions in the training and validation sets for the polynomial degree $\mathcal{N} = 9$ for the shock localization with ANNSL.

#	Training set		Validation set	
	class 0	class 1	class 0	class 1
1	21327	0	4253	0
2	69190	0	13769	0
3	6706	0	1345	0
4	4248	35364	861	7115
5	2281	17239	469	3386
6	80	33858	17	6922
7	253	19454	37	3826
	104085	105915	20751	21249

Table 3
Parameters for the training of the ANNSI and ANNSL network.

Training optimizer	Adam [24]
Mini-batch size	$m = 500$
Epochs	$n_{\text{epoch}} = 120$
Learning rate	LR = 0.01
	reduced every 15th epoch by factor 0.5

ANNSI network to pre-select elements containing shocks to show its capabilities as a pure shock detection scheme as well as to demonstrate the flexibility of our method with regards to the underlying properties of the discretization. The exact shock location on the FV grid is then conducted by the ANNSL net, which has been trained with suitable input data. The training data is summarized in Table 2, using approximately the same number of samples per class as for the ANNSI net in Table 1.

3.4. Neural network training

As discussed in Sec. 3.2.1, the HED architecture evaluates the edge prediction on different scales through the incorporation of side outputs into the cost function. Quantities that contribute to the cost function are marked as black dots in Fig. 4. Following [38], an adapted weighted cross-entropy function is chosen

$$C = -\frac{1}{m}\sum_{m}\sum_{l=1}^{L}\left[\frac{1}{S_Y}\sum_{s=1}^{S_Y}\Lambda \mathbf{Y}_s \ln(\hat{\mathbf{Y}}_s^l) + \lambda(1-\Lambda)(1-\mathbf{Y}_s)\ln(1-\hat{\mathbf{Y}}_s^l)\right]_m, \tag{4}$$

with L being the number of input layers to the loss function (black dots in Fig. 4), S_Y being the size of the output vector, the weighting factor $\lambda = 1.1$ and Λ being defined as

$$\Lambda = \frac{\#\text{total number of labels with class 1}}{\#\text{total number of labels}},$$

for the mini-batch size m. We start the training with an initial guess for the degrees of freedom of the neural network: All biases are chosen to be zero and the weights are initialized with randomly drawn numbers from a normal distribution with zero mean and standard deviation of $\sigma = 0.01$, except for the last layer where the standard deviation of the normal distribution is chosen to be $\sigma = 0.2$, following [38]. The remaining parameters for the training process are the same for ANNSI and ANNSL and are summarized in Table 3. We exemplary report the results for ANNSI trained for $\mathcal{N} = 5$ and $\mathcal{N} = 9$ to cover both a medium and a high order scheme, as well as for ANNSL with $\mathcal{N} = 9$. The influence of the training procedure on the loss evaluated with Eq. (4) is visualized in Fig. 6. It shows a monotone decreasing validation loss, indicating that no overfitting takes place. In order to quantify the validation accuracy of the trained networks, we choose the F_1 score, a custom metric for binary classifiers which incorporates true/false positives and negatives into the metric. The ANNSI networks reach a F_1 score of 96.461% for $\mathcal{N} = 5$ and 96.679% for $\mathcal{N} = 9$. With the ANNSL network for $\mathcal{N} = 9$ a F_1 score of 95.028% is reached. During training, we achieved F_1 scores of over 90% consistently with much smaller networks. We choose to focus on the best network with respect to the F_1 score in this work only, to evaluate the full potential of our approach.

4. Results: shock detection

With the ANNSI networks for $\mathcal{N} = 5$ and $\mathcal{N} = 9$ from Sec. 3.4 in place, we now apply the trained shock indicators to a range of 2D shock problems. We first describe the selected flow cases briefly in Sec. 4.1. With these definitions out

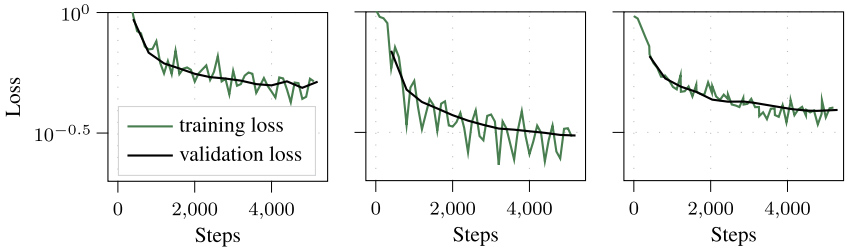


Fig. 6. Training and validation loss during the training of ANNSI (a/b) and ANNSL (c) network.

Table 4
Initial conditions of (ρ, v_1, v_2, p) in the four quadrants of the domain of the two-dimensional Riemann problems [60].

#	Configuration 4	Configuration 6	Configuration 12
1	(1.1, 0, 0, 1.1)	(1, 0, 0, 1)	(0.5313, 0, 0, 0.4)
2	(0.5065, 0.8939, 0, 0.35)	(0.5, -0.8708, 0, 0.3636)	(1, 0.7276, 0, 1)
3	(1.1, 0.8939, 0.8939, 1.1)	(1, -0.8708, 0.7977, 1)	(0.8, 0, 0, 1)
4	(0.5065, 0, 0.8939, 0.35)	(0.5, 0, 0.7977, 0.3636)	(1, 0, 0.7276, 1)
t_{end}	0.25	0.2	0.25

of the way, we then discuss the results for the ANNSI approach and contrast it against the established indicators from Sec. 2.4. In this section, we focus on the shock detection aspect of the proposed indicator, i.e. the marking of grid elements where shocks likely reside using the ANNSI network only. The aspect of shock localization within an element will be discussed in Sec. 5. For all computations presented herein, we rely on the discretization scheme, including the TVD shock capturing method, presented in Sec. 2.2 and 2.3. The polynomial ansatz degree of the scheme is either $\mathcal{N} = 5$ and $\mathcal{N} = 9$ as indicated; the scheme is a standard DGSEM collocation formulation on Legendre-Gauss points, with a Roe or HLLE scheme for the numerical flux functions of both DG and FV elements. A minmod limiter ensures the TVD property of the linear FV reconstruction polynomials. The solution is advanced in time by an explicit high order Runge-Kutta scheme [23]; the typical CFL condition limits the allowable global timestep [27]. The density is chosen as an input quantity into ANNSI.

All simulations with ANNSI or ANNSL shown in this manuscript are run ab initio, i.e. from the given initial conditions at time $t_{start} = 0.0$ to the indicated t_{end}, using the proposed indicators only. While we focus on discussing the solution quality at a given time typical for the selected test case, this shows that ANNSI and ANNSL approaches are capable of dealing with highly transient situations without the need for any special treatment.

4.1. Test cases

4.1.1. 2D Riemann problems

In [60], the definition of the 1D Riemann (RP) problem for the Euler system is extended to two dimensions. The authors identify 16 distinct configurations in which constant states in the four domain quadrants are separated initially by an elementary wave and propose them as canonical test cases. We choose three configurations (4,6,12), which among them include the three typical wave phenomena, shocks, rarefactions and contact discontinuities. The initial conditions and end times of the RPs are given in Table 4; they define the solution in the four quadrants of the domain (numbered from the upper right quadrant counterclockwise). The boundary conditions on all domain faces are set to the exact solutions.

Fig. 7 shows the resulting density contours for the three configurations as a reference, computed on a fine grid with the numerical scheme described above. Note that along the contact discontinuities in configuration 12, a Kelvin-Helmholtz instability triggers the roll-up of vortical structures. For the comparisons of the shock indicators later on, we define a baseline regular Cartesian mesh M_{RP} with 50×50 elements for all configurations.

4.1.2. Double Mach reflection problem

This testcase consists of a strong shock at Mach 10 impinging on a wedge at an oblique angle of 30°. Our setup is consistent with the description in [70], with slip-wall boundary conditions at the top and bottom, a Dirichlet inflow on the left and a supersonic outflow at the right hand side of the domain. Fig. 8 shows the reference solution at $t_{\text{end}} = 0.2$ computed on a fine grid. We define a coarse baseline grid M_{DMR} with 49×12 elements for later use in this study. The numerical flux is approximated by a Roe scheme.

4.1.3. Forward facing step problem

This test setup, introduced in [70] describes the flow over a forward facing step (FFS) with a Mach number of $\text{Ma} = 3.0$. The expected flow phenomena are mainly a developing bow shock in front of the step and the reflections of this shock

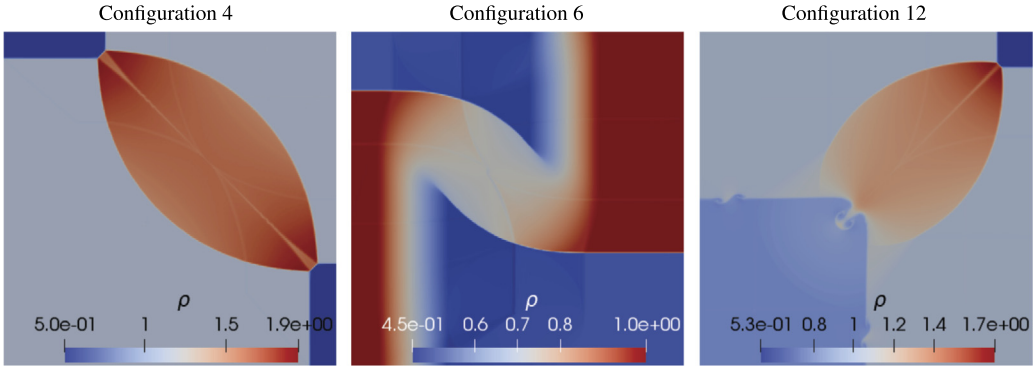


Fig. 7. Reference solution for the density contours of configuration 4, 6 and 12 of the two-dimensional Riemann problem. Computed on a 100×100 mesh with a polynomial degree of $\mathcal{N} = 5$.

Fig. 8. Reference solution of the density of DMR computed on a 96×24 mesh at $t_{end} = 0.2$.

Fig. 9. Reference solution of the density of the FFS at $t_{end} = 4$.

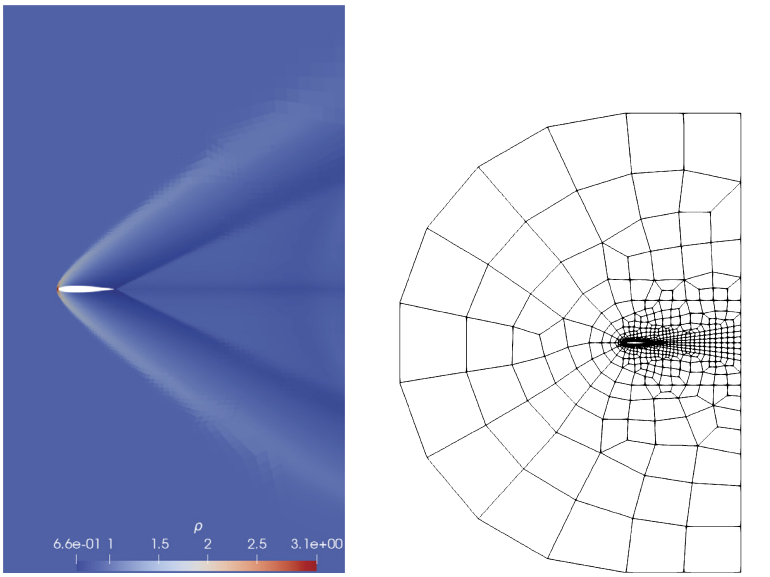
from the upper and lower wall. Inflow and outflow are described with Dirichlet boundary conditions, other boundaries are modeled as walls. In Fig. 9 the density at $t_{\text{end}} = 4$ is visualized for a calculation with 50 elements in the x-direction and 25 and 20 elements in the two blocks in y-direction using $\mathcal{N} = 9$ and an HLLE Riemann solver. For more details on the used mesh see also Fig. 20.

4.1.4. 2D NACA 0012

As a final testcase, we consider the flow over a NACA 0012 airfoil at $Ma = 2.0, AoA = 0°$ on an unstructured grid, depicted in Fig. 10. This setup results in a detached, curved bow shock and a tail shock. The solution is advanced in time from an initial freestream condition until a steady state is achieved. The mesh is coarse, without symmetries and does not take a priori knowledge of the shock front position, strength and shape into account, resulting in a deliberately "bad" mesh for this task. This setup was generated to test the shock indicators in a more realistic setting for high order methods than in the previous test cases, where the shock location might be uncertain or rapidly moving, grid refinement is not available or too costly, and large grid elements are the norm. Fig. 10 shows the computational grid and the flow solution.

4.2. ANNSI results for $\mathcal{N} = 5$

We first consider the results for the $\mathcal{N} = 5$ computations and networks, and discuss selected testcases from Sec. 4.1 and contrast the results from ANNSI and the troubled cell indicators from Sec. 2.4. It is important to underline that while for the modal and jump indicators $\mathcal{I}_{\text{modal}}$ and $\mathcal{I}_{\text{jump}}$ the respective parameters defining the thresholds need to be set by the



(a) Density at $t_{end} = 8.0$. (b) Unstructured mesh used for the computation.

Fig. 10. Density (left) and mesh (right) of the NACA0012 profile.

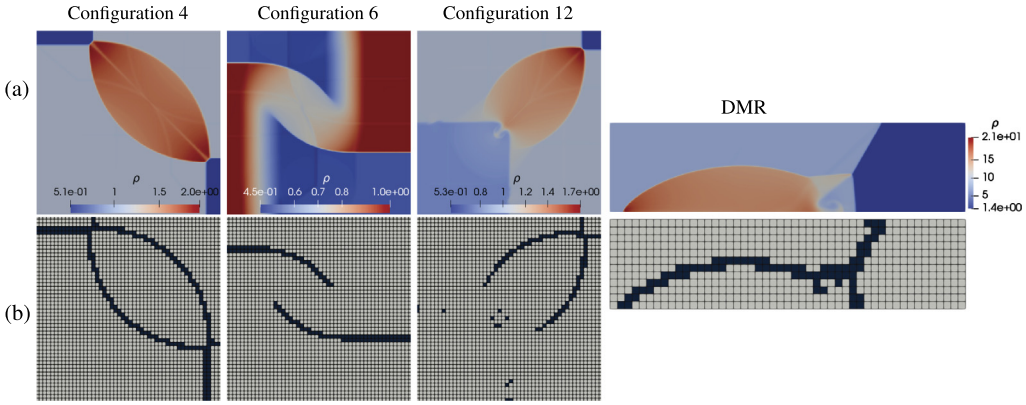


Fig. 11. $\mathcal{N} = 5$ results: Flagged cells (dark color) of ANNSI for configuration 4 (left), 6 (left of middle) and 12 (right of middle) of the two-dimensional Riemann problems and for the Double Mach Reflection (right). The Riemann problems are calculated on the baseline mesh M_{RP} with 50×50 mesh and the DMR on the M_{DMR} mesh with 49×12 elements.

user and possibly adapted for each situation, the ANNSI prediction of the neural network established in Sec. 3 is used here 'as is' throughout this chapter, without the need (or possibility) of user intervention or parameter adjustment.

In a first step, we apply the ANNSI to a range of test problems to establish its suitability, robustness and accuracy for the intended purpose.

Fig. 11 shows the resulting flow fields in the top row and the flagged grid elements in the bottom row for the Riemann problems and the DMR. All simulations using ANNSI are stable and the resulting flow fields are qualitatively as expected. The flagged cells also coincide with the shock locations, the shock fronts are unbroken and well-captured. For the Riemann problem configuration 12, the contact discontinuity lines are partially detected, which is likely due to the roll up of the shear layer. However, as expected, outside of the vortex regions, the contact discontinuity is not marked by ANNSI.

Based on these initial findings that the ANNSI method delivers a useful and robust shock detection, we now compare its predictions against the $\mathcal{I}_{\text{modal}}$ and $\mathcal{I}_{\text{jump}}$ indicators. As pointed out above, the upper and lower detection thresholds of these indicators are tuneable parameters and no strict quantitative guidelines exist that can help fix them. Instead, the task of the user is to find a setting mainly through trial and error that results in sufficient robustness but do not trigger excessive solution stabilization mechanisms, which can either be computationally expensive or unduly influence smooth regions of the solution - or both. Beyond the academic examples considered here, such parameter variations however are often too costly and cumbersome, and thus often a parameter set based on previous experience is chosen. In order to now compare this approach with the one proposed here, we hand-tuned the $\mathcal{I}_{\text{modal}}$ and $\mathcal{I}_{\text{jump}}$ indicators collectively on the four test

Table 5
Settings for $\mathcal{I}_{\text{modal}}$ and $\mathcal{I}_{\text{jump}}$, collectively tuned for the RP and DMR cases.

#	$\mathcal{I}_{\text{modal}}$	$\mathcal{I}_{\text{jump}}$
$\mathcal{I}_{\text{upper}}$	-4.5	0.012
$\mathcal{I}_{\text{lower}}$	-4.7	0.01

cases from Fig. 11 (Riemann problems 4,6,12 and DMR) under the following conditions: For each of the two indicators, the optimal parameter settings must provide a stable simulation of all four test cases and must not result in excessive flagging of cells outside of the expected regions. Our approach resulted in the settings listed in Table 5.

Remark 2. The approach of hand-tuning these indicators based on the criteria specified above is somewhat unsatisfactory as it is more or less subjective. However, this is commonly used in practical applications with a priori troubled cell indicators for shock capturing and again reflects the challenges of predicting the solution update of the discretized PDE based on the current solution alone.

Remark 3. Tuning the parameters for each of the cases individually would likely have lead to improved results in some cases. However, our goal here is not to establish the superiority of any of the indicators over the other ones but to compare them qualitatively in a typical setting.

We compare these results with the ANNSI cases in Fig. 12, where we have repeated the relevant results from Fig. 11 as the first column. All three indicators yield comparable results, flagging shock locations but not expansion waves. The ANNSI predictions typically flag two adjacent cells across a shock, while in particular the jump indicator shows a more refined picture. On the other hand, the shock fronts detected by ANNSI are continuous and unbroken in contrast to the $\mathcal{I}_{\text{modal}}$ and $\mathcal{I}_{\text{jump}}$ results, where "holes" in the shock fronts appear. Applying the same indicators to the NACA0012 case leads to a stable simulation in all cases; the results for the flagged cells are shown in Fig. 13. The ANNSI driven method performs well on this suboptimal grid, yielding continuous shock fronts and a more conservative estimate than the other two. We note that as the computational grid is not symmetric w.r.t. the x-axis, so we also do not expect a necessarily symmetric flagging of the shock fronts.

The results presented in this section can thus be summarized as: The developed ANNSI method is robust, suitable for structured and unstructured grids and of comparable accuracy to other well-tuned indicators.

4.3. ANNSI results for $\mathcal{N} = 9$

In this section we apply the ANNSI network to the same 2D problems as in the previous section. We perform the calculations with a polynomial degree of $\mathcal{N} = 9$ to show the invariance of the performance with respect to the polynomial degree of the ansatz functions.

In Fig. 14 we repeat the calculations from Fig. 11 with $\mathcal{N} = 9$ instead of $\mathcal{N} = 5$. A very similar behavior can be observed: again all simulations using the ANNSI are stable and the resulting flow fields are qualitatively as expected. The shock localization is correctly recognized and captured with the finite volume sub-cell scheme. Differences can be seen at the contact discontinuities in RP12 which are partially detected by the indicator. We also compared the ANNSI results against those for the selected problems in a repetition of Fig. 12. We found the results from $\mathcal{N} = 5$ transferable to the $\mathcal{N} = 9$ case.

4.4. Influence of grid resolution

Having established the properties of the ANNSI approach on the baseline grids, we now investigate the influence of grid resolution changes by either halving or doubling the grid cells for the $\mathcal{N} = 5$ case per direction. Since the training data for ANNSI was constructed in reference space, we expect its results to be robust against grid resolution changes. For $\mathcal{I}_{\text{jump}}$ and $\mathcal{I}_{\text{modal}}$, we again use the found parameter settings from Sec. 4.2.

Fig. 15 and 16 repeat the investigations for the coarse and fine grids. Missing plots indicate that the simulation was not stable. The coarse grid results confirm our assumption that the ANNSI approach is rather robust against a resolution drop. Comparing Fig. 15 and Fig. 12, the shock position and extension are predicted in a consistent manner. On the fine grid in Fig. 16, the ANNSI results converge and become sharper as desirable. In order to give the reader an impression of the flow field achieved with ANNSI and a direct visual comparison of the results, we have shown the solution of the Riemann problems on the fine grid for the indicators considered in Fig. 22. As expected from the comparable distribution of troubled cells (cf. Fig. 16), the solution contours for all three cases are visually very similar. In particular, the ANNSI indicator does not introduce excessive smearing and yields sharp shock fronts.

Remark 4. Note that it is certainly possible to achieve stable results for all testcases with the $\mathcal{I}_{\text{jump}}$ and $\mathcal{I}_{\text{modal}}$ indicators under the given conditions (see e.g. [64,65]), but this would require retuning the threshold parameters. Since our focus here

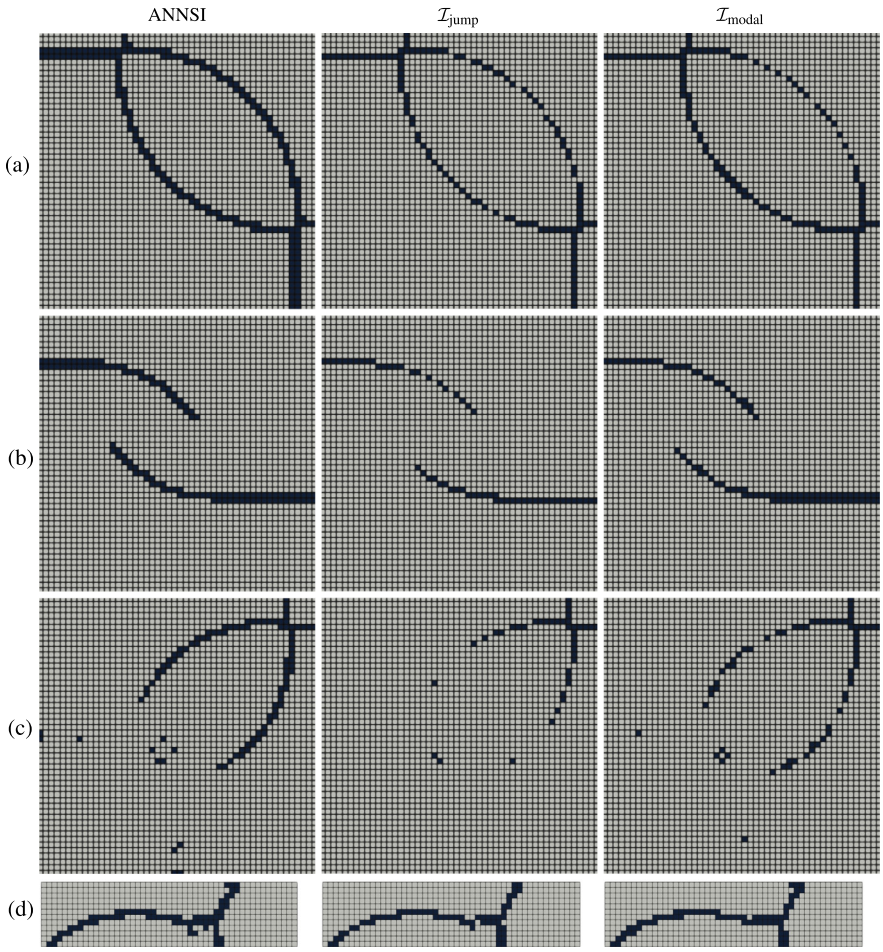


Fig. 12. $\mathcal{N} = 5$ results: Flagged cells (dark color) of ANNSI (left), the jump (middle) and the modal indicator (right) for configuration 4 (a), 6 (b) and 12 (c) of the two-dimensional Riemann problems and for the Double Mach Reflection (d). The Riemann problems are calculated on the baseline mesh M_{RP} with 50×50 mesh and the DMR on the M_{DMR} mesh with 49×12 elements.

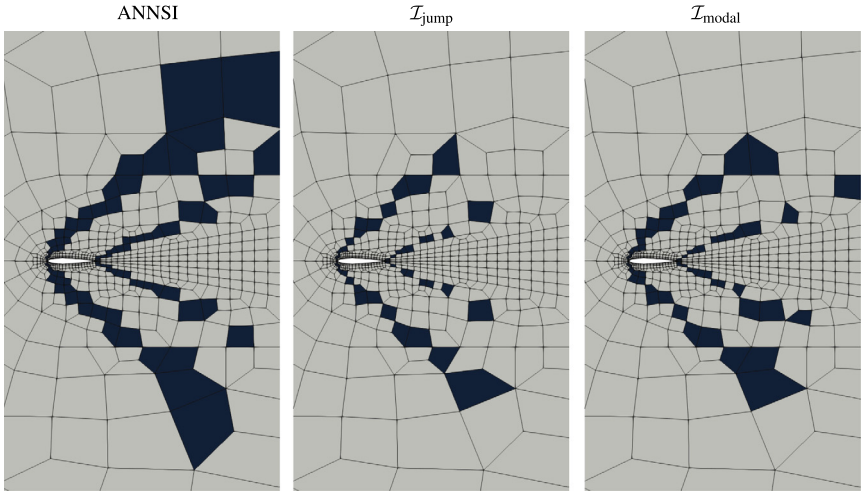


Fig. 13. $\mathcal{N} = 5$ results: Flagged cells (dark color) of ANNSI (left), the jump (middle) and the modal indicator (right) for $Ma = 2.0$ NACA0012 flow.

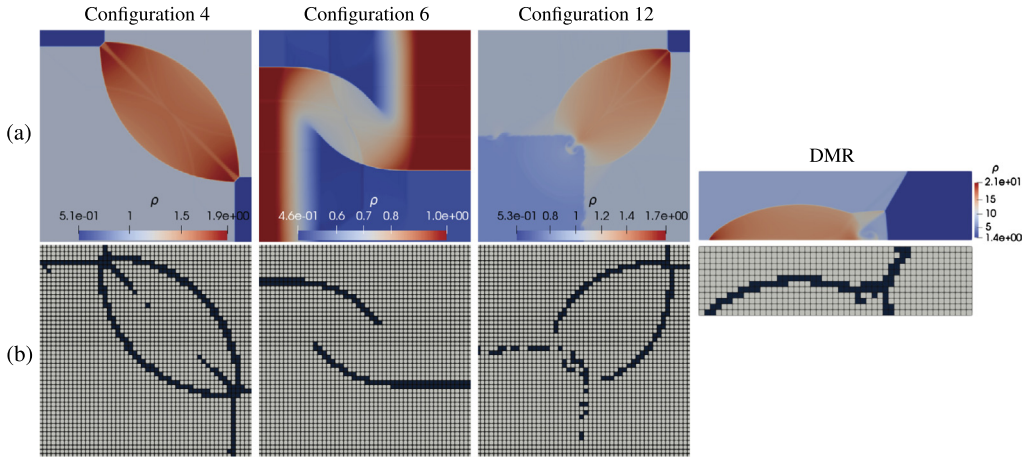


Fig. 14. $\mathcal{N} = 9$ results: Flagged cells (dark color) of ANNSI for configuration 4 (a), 6 (b) and 12 (c) of the two-dimensional Riemann problems and for the Double Mach Reflection (d). The Riemann problems are calculated on a mesh with half the resolution in each direction (25×25 elements) and the DMR on a mesh with 24×6 elements.

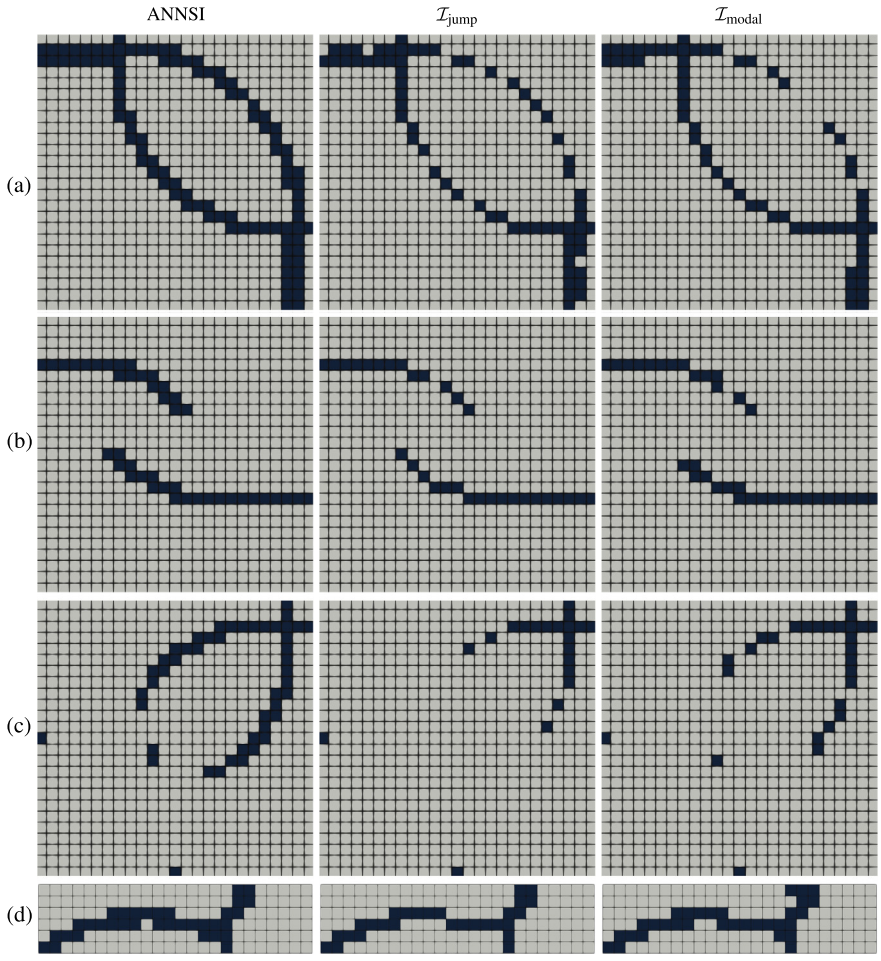


Fig. 15. $\mathcal{N} = 5$ results, coarse mesh: Flagged cells (dark color) of ANNSI (left), the jump (middle) and the modal indicator (right) for configuration 4 (a), 6 (b) and 12 (c) of the two-dimensional Riemann problems and for the Double Mach Reflection (d). The Riemann problems are calculated on a mesh with half the resolution in each direction (25×25 elements) and the DMR on a mesh with 24×6 elements.

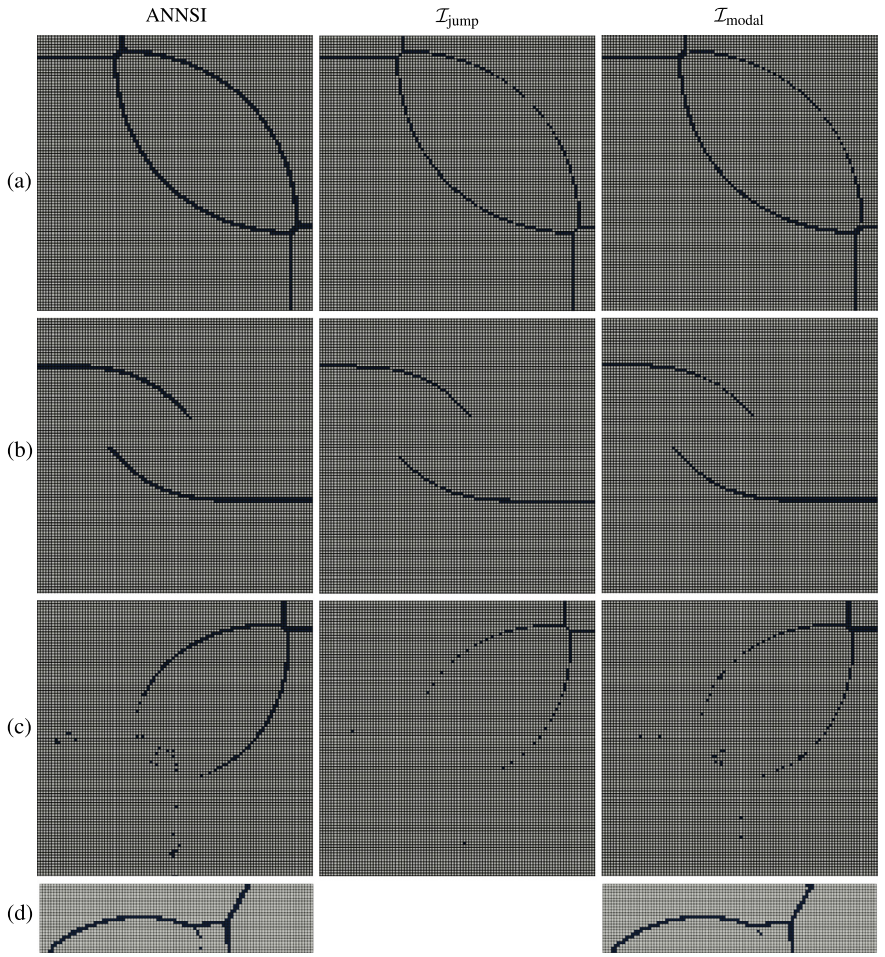


Fig. 16. $\mathcal{N} = 5$ results, fine mesh: Flagged cells (dark color) of ANNSI (left), the jump (middle) and the modal indicator (right) for configuration 4 (a), 6 (b) and 12 (c) of the two-dimensional Riemann problems and for the Double Mach Reflection (d). The Riemann problems are calculated on a mesh with twice the resolution in each direction (100×100 elements) and the DMR on a mesh with 96×24 elements.

is precisely on investigating these sensitivities with regards to the resolution, we did not adjust the parameters of $\mathcal{I}_{\text{jump}}$ and $\mathcal{I}_{\text{modal}}$.

Summarizing the findings in this section, the developed ANNSI approach results in robust solutions for the problems considered. It compares well against other solution-based indicators in terms of accuracy and does not excessively flag elements outside of shock regions and can deal robustly with the initial solution transients and shock movement. It has shown to be insensitive to grid resolution and consistently flags the same regions / features, thus working as intended as a discretization independent shock detector, and not as a troubled cell indicator. While results of comparable or possibly better quality for specific cases can be achieved by parameter studies and tuning with other a priori indicators, once trained, ANNSI is parameter-free. It can be argued that the time spent training the neural network should be weighted against the time one might invest in parameter tuning, but with large scale application with huge computational costs in mind, it is certainly more economic to do invest the costs “offline” before the actual computation. This aspect will become even more important in the next section, where we discuss the results of the ANNSL extension of ANNSI, where a localization of the shock front within an element is proposed.

Remark 5. Considering the performance during runtime, the evaluation time of the ANNSI indicator is approximately 1.5 orders of magnitude higher than for traditional indicators. Note that the performance of the method can certainly by improved, however, in this work, we are primarily interested in building the fundamentals for an accurate and robust shock indicator in the context of high order methods, rather than on efficiency. An identifiable large bottleneck in the performance is the coupling of the CPU-run discretization scheme with a GPU-run neural network serving architecture. Improvements regarding runtime performance can be made by reducing the complexity of the equipped network architecture, by applying a pruning strategy, by using specialized hardware or by combining the ANN-approach with a GPU-based code.

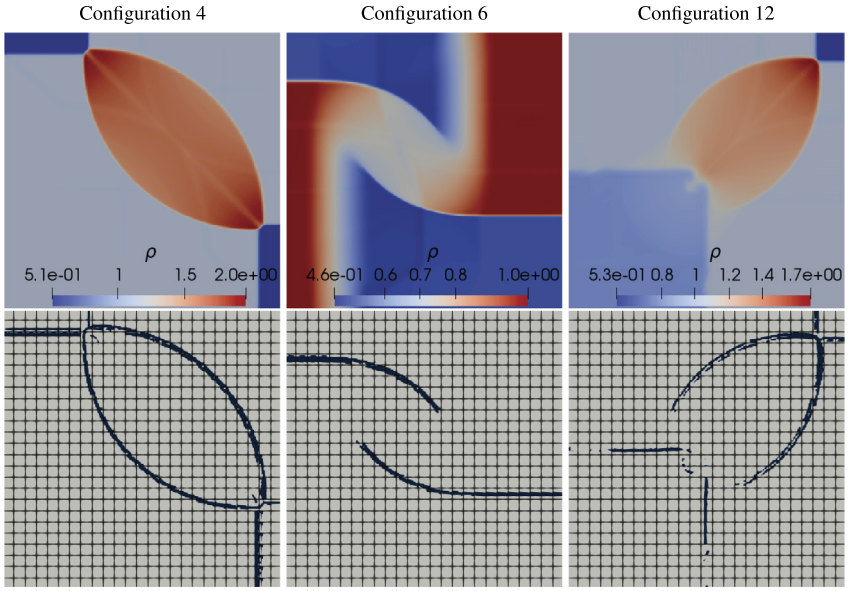


Fig. 17. Density (top) and binary shock edge map obtained with the ANNSL network with the underlying grid of size 25×25 (bottom) of RP configuration 4 (left), 6 (middle) and 12 (right).

5. Results: shock localization

In this section, we illustrate how the information about the shock location inside an element generated by the ANNSL network can be exploited. First, the test cases from the previous section are used to illustrate the capabilities of the shock localization procedure. Next, the ANNSL indicator is applied to a new testcase and the possible exploitation of the shock localization in an adaptive mesh refinement (AMR) framework is illustrated. All ANNSL computations are again started from ab initio, with ANNSL being active in every timestep. The procedure is according to Fig. 5: First, ANNSI is used as a shock detector, then ANNSL takes care of the shock localization within the cells marked by ANNSI.

5.1. Shock localization

In the previous section, we have shown that the ANNSI shock detector performs well in terms of robustness, accuracy and insensitivity to spatial resolution changes for a range of classical test problems. Here, we use those applications again to illustrate the capabilities of the ANNSL network of identifying the shock position inside an element, i.e. of creating an inner-element bounding box around the shock front and flagging the solution points adjacent to the front. Since this is particularly useful for high order discretizations such as $\mathcal{N} = 9$, we will focus our investigation on this case, using the network trained in Sec. 3.4 on the data from Sec. 3.3.2. Fig. 17 and Fig. 18 apply the ANNSL network to the established test cases. The binary edge map is shown alongside the flow solution, with dark colors indicating that a pixel / solution point is considered to be in the direct vicinity of or on the shock front. One can observe that with the proposed indicator, a very sharp identification of discontinuities is possible. The identification is generally continuous within a cell (no holes or bumps in the shock fronts) and consistent across element boundaries - neighboring element “agree” in the identification of the fronts. All of these properties are desirable in edge detection, however, many more simple edge detection algorithms struggle with one or more of them [72]. For the intentionally “bad” grid in Fig. 18, the ANNSL prediction also works very well on coarse cells. The infrequently occurring ‘two-stripes’ patterns (see e.g. lower region of the bow shock) are due to the discrete resolution of a shock: on the discrete level, it consists of two regions with kinks at the foot and head of the shock and an almost linear part in between. As the linear part is not distinguishable from other smooth solutions, it is not recognized as a shock, and only the shock edges are recognized as parts of a shock.

In Fig. 19, we analyze the convergence behavior of the shock map with decreasing element size. We observe that the shock localization is stable with respect to the grid size, and the edge map predictions become successively more refined. This feature is an important prerequisite for successful h/p/r-adaption strategies to improve accuracy at the shock. In the next section, we discuss a possibility for taking advantage of this knowledge during the simulation.

5.2. Neural network informed H-adaptation

We use the flow over a forward facing step problem on a regular Cartesian mesh defined in Sec. 4.1.3 to illustrate how the information about the inner-cell shock position can be exploited during the calculation to improve the accuracy of the shock capturing approach. We restrict the discussion to a three-level h-refinement strategy in which a donor cell on the

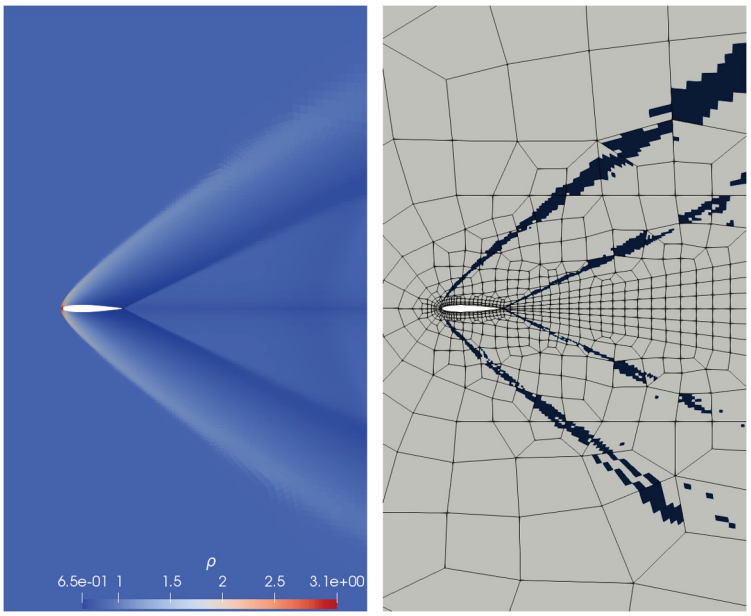


Fig. 18. Density (left) and binary shock edge map obtained with the ANNSL network with the underlying grid (right) of the NACA 0012 profile.

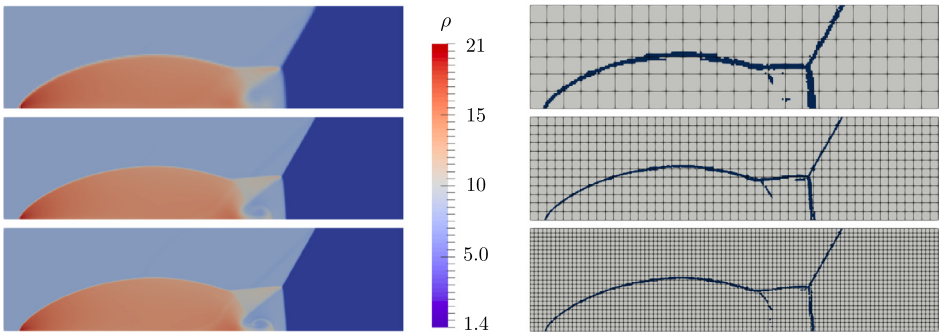


Fig. 19. Density (left) and binary shock edge map obtained with the ANNSL network with the underlying grid (right) of the DMR computed on meshes of size 24×6 (upper), 49×12 (middle) and 96×24 (lower).

original grid can be halved in any coordinate direction separately or in both at the same time, leading to either 2 or 4 new child cells replacing the original element (anisotropic or isotropic split). This process can be repeated twice, so three levels of grid elements can be created. The original regular grid is shown in the top left column in Fig. 20. We conduct all simulations with a polynomial degree of $\mathcal{N} = 9$ and the ANNSL network from the previous sections and visualize the results at $t_{\text{end}} = 4$.

The left column of Fig. 20 shows the solution on the baseline mesh, which is direction-wise equidistant. The corresponding shock localization indicator and the resulting density field are shown below, again, the ANNSL approach leads to a sharp and consistent prediction of the front locations, confirming the findings from Sec. 5.1. From the binary edge map, two pieces of information can now be extracted: First, the shock width is characterized by the amount of flagged solution points / pixels within one element. Secondly, the orientation of the shock front is available. Several methods to take advantage of this information to inform an h-refinement are possible; we choose a very simple strategy for demonstration purposes here which already produces remarkable results: We define an indicator for anisotropic mesh refinement which exploits both information from the binary edge map

$$\mathcal{I}^{\text{dir}}_{\text{meshref}} = \sum_{i=0}^{\mathcal{N}} \sum_{s=1}^{r_i^{\text{dir}}-1} \left(1.7^{s-1} + 1.7^{s}\right), \tag{5}$$

independently for both the x and y-directions, with r_i^{dir} being the number of solution points with class 1 of layer i in the corresponding x or y-direction, i.e. we count the number of flagged pixels line-wise. After evaluating the indicator we apply the mesh refinement in each direction independently. Last, a constant interpolation on the new grid is done. We

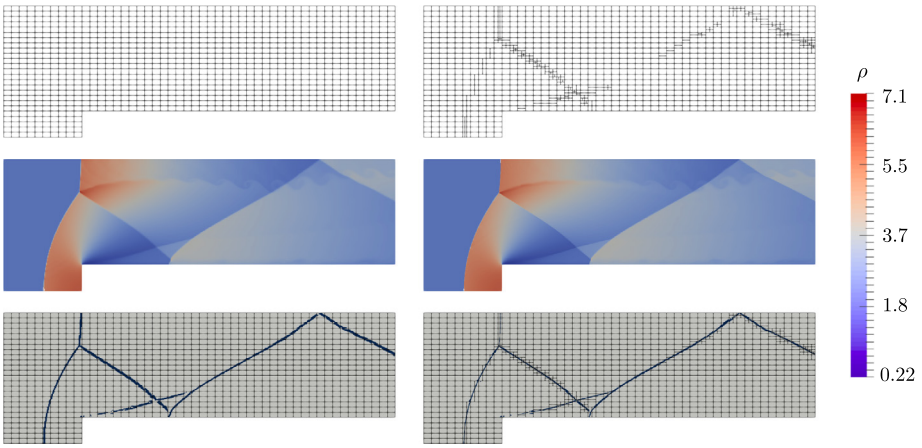


Fig. 20. Computational grid (upper), density (middle) and binary shock edge map obtained with the ANNSL network with the underlying grid (bottom) of the FFS computed on a direction-wise equidistant mesh (left) and on a h-refined version of it (right).

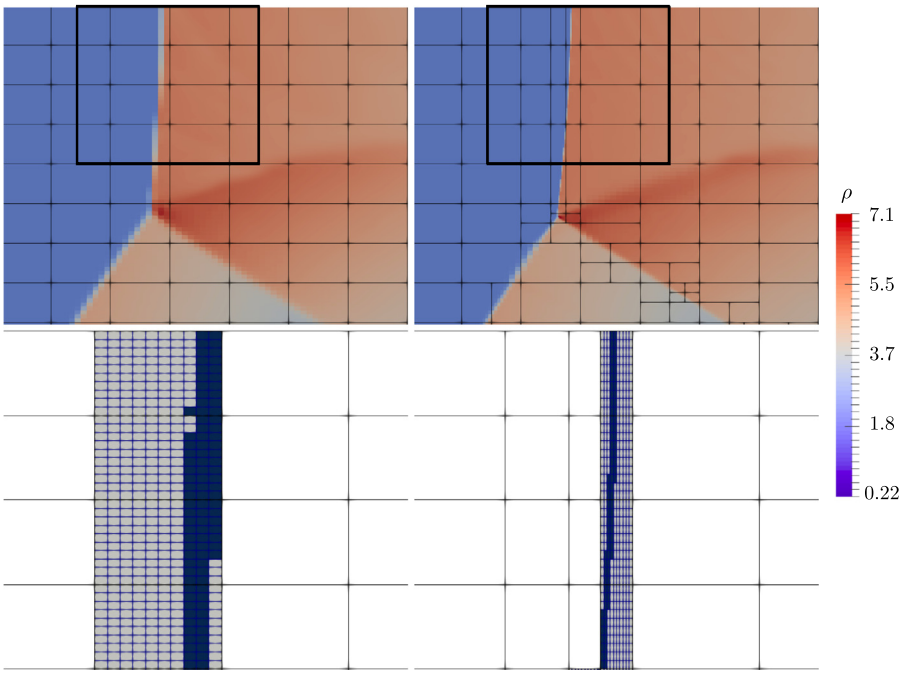


Fig. 21. Top row: Zoom to density and underlying mesh at upper shock crossing point for FFS with direction-wise equidistant grid (left) and anisotropic refined mesh (right). The region shown in the bottom row is marked by the black rectangle. Bottom row: Detailed view of edge map shock predictions. The grid cells are denoted by black lines. Cells without shocks have a white background. In cells in which a shock is detected, the finite volume subgrid is shown, dark colors indicate the shock location prediction.

apply the indicator again on the refined solution, choosing the empirically determined threshold values $\mathcal{I}_{\text{meshref}}^{\text{threshold}} = 33$ and $\mathcal{I}_{\text{meshref}}^{\text{threshold}} = 172$ for the first and the second refinement. These three levels of grid elements provide a sharp resolution of the shocks. Note that during the refinement it is taken care that only non-conforming interfaces with a 2:1 ratio occur. Details on DGSEM and the finite volume sub-cell method on meshes with non-conforming interfaces can be found in [27,64]. Fig. 20 (right column) shows the resulting mesh, the results on the refined mesh and the binary edge map, illustrating the sharper resolution of the shocks due to the refined mesh. Fig. 21 (top row) provides a zoomed-in view on the upper crossing point of the shocks, showing both the density field and the associated mesh. It is easy to recognize how the refinement informed by ANNSL has improved the sharpness of the shock while avoiding unnecessary refinement of cells or along directions. Also, the refinement criterion proposed in Eq. (5) has guided the selection between anisotropic and isotropic splits, which additionally helps to keep the number of newly introduced cells and thus computational cost low. In the bottom row of Fig. 21, the computational grid and the binary edge map in this region are depicted. Cells of the computational grid are denoted by black lines; in elements with white backgrounds, no shocks have been detected and the DG representation of

the solution is used. In cells with a light blue background, shocks are present, and the FV representation of the solution on equispaced grid cells is active. In FV cells / pixels of the binary edge map marked with a dark color, the ANNSL method has predicted the occurrence of a shock. As discussed before, the network has been designed to mark the two adjacent pixels to the shock front per direction, i.e. to provide a bounding box of the discontinuity on the scale of smallest resolved length scale within a DG element. The bottom plots in Fig. 21 shows that ANNSL method successfully provides such a bounding box along the shock front, given a consistent and sharp inner-cell shock localization on both the original (left) and refined grid (right).

Concluding, this example illustrates how the information obtained by the novel indicator for shock localization can be used in an anisotropic adaptive mesh refinement framework to improve the resolution of the shock. Due to the separation of shock detection and shock capturing in our approach, the indicator can be transferred to other discretization schemes and can then help to inform other shock capturing methods.

6. Conclusion and outlook

The stable approximation of shocks or very steep gradients on a finite computational grid is a challenging task due to the non-linear interactions of solution behavior and discretization. In the most commonly used approach, parameterized a priori indicators are used to not only detect the occurrence of a shock within an element, but also to implicitly predict the stability of the discretized PDE solution at the next time step. This subtle redefinition of tasks is often expressed by the use of the term “troubled cell” indicators. The interactions of solution, discretization and indicator parameters introduces the need for parameter tuning and user experience, and considerable time may be spent to find an optimal indicator setting.

In this work, we have proposed a novel indicator based on edge detection methods from machine learning and computer vision. The indicator is developed, tested and applied in a high order discontinuous Galerkin setting, but it can easily be transferred to related discretization schemes. In order to make this indicator (at least more) universal and parameter-independent, we have returned to the original concept of separating the tasks of detection a shock and of its stable and accurate numerical approximation. The latter one is based on our hybrid finite volume / discontinuous Galerkin scheme and is treated as a black box for the purpose of this investigation. The task of detecting and localizing a shock is performed by neural networks designed for edge detection and works on the basis of considering the inner element solution akin to an image and the nodal degrees of freedom as its pixels. The neural networks are trained on analytical smooth and non-smooth functions to provide a binary edge map in each element that bounds the shock fronts in between two solution points. This information can then be used to flag an element containing a shock in the sense of a detector or to guide an inner-cell shock capturing method.

This feature is particularly attractive for high order schemes in non-trivial geometries and for complex flows, where the location of the shocks might not be obvious a priori (so the grid can be generated accordingly) or where shock movement occurs. Due to the typically large grid cells in use with these schemes, locating the shock front accurately and reliably within an element is useful to guide h/p/r-refinement methods, artificial viscosity based approaches, enriched basis functions or other methods to stably approximate the solution.

We see our work here as a step towards this goal. Our shock detection indicator has shown to be as robust and accurate as two established shock indicators, without the need for user intervention. It has also proven to be robust against changes in spatial resolution and the choice of different Riemann solvers for the DG method illustrates the insensitivity of the indicator to such a variation of the discretization. The novel shock localization algorithm is capable of reliably bounding the shock front within a high order element, even on coarse grids. Its location prediction behaves consistently when refining the grid, opening the way for targeted, multilevel grid refinement. We have demonstrated how to use the information provided by the indicator to improve solution quality considerably by local h-refinement.

Based on the current work, a number of possible routes can be investigated in the future: The extension to 3D is a first natural step. Here, a tensorproduct expansion of the training data is an obvious choice. Incorporating more physical information into the training, i.e. through training on e.g. the density and pressure fields can extend the methods towards identifying other flow features. This might become particularly interesting for the notoriously difficult shock-turbulence interaction problems. Also, beyond the binary edge map, predicting a shock strength or other properties can be useful for informing the shock capturing method. Improving network performance and its integration in existing solvers are important technical issues to be addressed.

It can be argued that the costs of computing the neural network based indicator are certainly higher than those of established formulations, so the natural question is to ask if it is worth it. While we see our work as a proof of concept and do not directly tackle this issue here, answering this question in the future will depend on more than just the cost per evaluation: For one, the presented indicators can help to reduce the human and computational effort of parameter tuning. Even more significant is the potential for saving effort and improving solution accuracy through exploitation of the precise shock position, in particular in 3D for high order discretizations. Also to be taken into account are moves towards integration of specialized hardware for machine learning applications in HPC systems.

CRediT authorship contribution statement

Andrea D. Beck: Conceptualization, Funding acquisition, Methodology, Project administration, Software, Supervision, Validation, Writing - original draft, Writing - review & editing. **Jonas Zeifang:** Conceptualization, Investigation, Methodology, Project administration, Software, Supervision, Validation, Visualization, Writing - original draft, Writing - review & editing. **Anna Schwarz:** Data curation, Investigation, Methodology, Software, Validation, Visualization, Writing - original draft, Writing - review & editing. **David G. Flad:** Methodology, Validation, Writing - original draft, Writing - review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The research presented in this paper was funded in parts by Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy - EXC 2075 - 390740016 and by the DFG GRK 2160/1: DROPIT Droplet Interaction Technologies. The authors gratefully acknowledge the support and the computing time on "Hazel Hen" provided by the HLRS through the project "hpcdg".

Appendix A. Generation of 1D training data

In the following, we detail the generation of training samples for the different analytical functions with $\varepsilon = 0.1$, chosen from 7 families given below. Here, u_i denotes the vector of all values in a single training sample, i.e. an element.

- No. 1: Linear functions, initialized on three meshes $n_e = 1, 10, 20$. The probability of choosing each of the meshes is $p_e = 0.5, 0.3, 0.2$, respectively.
- No. 2: Superposition of oscillations with different amplitudes and frequencies, initialized on meshes with $n_e = 1, 10, 20$ and probabilities of $p_e = 0.3, 0.4, 0.3$. Note that $f_{\text{Nyquist}} = \mathcal{N}/2$ approximates the maximum possible frequency which can be resolved by the Lagrange polynomials.
- No. 3: Exponential functions in both directions, initialized on $n_e = 10, 20$ with probabilities $p_e = 0.6, 0.4$. No function evaluation with $n_e = 1$ is done in order not to indicate too steep gradients with class 0.
- No. 4: Four constant values separated with straight and curved lines, initialized on $n_e = 1$. The assignment to class 1 is done if the maximum jump height fulfills the condition $|\max(|u_i|) - \min(u_i)| > \varepsilon \max(|u_i|)$ and $\max(|u_i|) > 0.01$ holds. Degrees of freedom neighboring or being directly at the discontinuity are labeled as 1. The parameter d is chosen to give a ratio of straight to curved separation lines of 7:3.
- No. 5: Magnitude function with linear gradients connected by kinks, initialized on $n_e = 1$. The assignment to class 1 is done if the condition $a > \varepsilon \max(|u_i|)$ for the gradient a and $\max(|u_i|) > 0.01$ for the maximum value holds and is set for degrees of freedom neighboring or being directly at the discontinuity.
- No. 6: Linear or constant states connected by kinks, initialized on $n_e = 1$. The assignment to class 1 is done if the maximum gradient fulfills $\max(b_1, b_2) > \varepsilon \max(|u_i|)$ and $\max(|u_i|) > 0.01$ holds and is set for degrees of freedom neighboring or being directly at the discontinuity.
- No. 7: Decaying high frequency oscillations in analogy to Gibb's instability, initialized on $n_e = 1$. The assignment to class 1 is done if the maximum difference in the element fulfills the condition $|\max(|u_i|) - \min(u_i)| > 0.2 \max(|u_i|)$ and $\max(|u_i|) > 0.01$ holds. Then degrees of freedom having the highest absolute value in the element are labeled as 1.

Table 6 summarizes the function families and their parameters. After the generation of the training samples, the inputs are normalized to enhance the convergence of the training process. We first shift the data $\mathbf{X} = (u_1, \cdots, u_{n_k})$ with the number of input samples n_k

$$u_i = \begin{cases} u_i - \min_x u_i & \text{if } \min_x u_i < 0, \\ u_i & \text{otherwise,} \end{cases}$$

and then scale them to the interval $\mathcal{I} = [0, 1]$

$$u_i = \begin{cases} \frac{u_i}{\max_x |u_i|} & \text{if } \max_x |u_i| > 1, \\ u_i & \text{otherwise.} \end{cases}$$

This enables to train with data in the positive interval $\mathcal{I} = [0, 1]$ while retaining the information about relative differences in the solution. To obtain training samples from the analytical functions, they are initialized on Cartesian meshes covering the interval $[-1, 1]^2$ with n_e^2 elements. To account for under resolved phenomena, we use a polynomial degree in each

Table 6
Functions used to generate the training and validation sets on the interval $[-1,1]^2$. Note that $\mathcal{U}[\mathcal{I}]$ denotes the uniform distribution on the interval $\mathcal{I}$ and $\mathcal{N}[\mu, \sqrt{\sigma^2}]$ denotes the normal distribution with mean μ and variance σ^2.

#	$u(x,y)$	Parameters	Class
1	$ax+by$	$a,b \in \mathcal{N}[0,0.2]$	0
2	$\sum_{k=1}^{N_f} a_k sin(k\pi x)+$ $b_k cos(k\pi y)+c$	$a_k, b_k \in \mathcal{U}[-0.5,0.5]$, $c \in \mathcal{U}[0,1]$, $N_f = 1,..,f_{\text{Nyquist}}$	0
3	$exp(a_1[(x-a_2)^2+(y-a_3)^2])+$ $exp(a_4[(x-a_5)^2+(y-a_6)^2])$	$a_i \in \mathcal{U}[-1,1]$	0
4	4 values a_i in 4 sections defined by $y-y_0 = m(x-x_0)^d$ $y-y_0 = -1/m(x-x_0)^d$	$a_i \in \mathcal{U}[0,1]$, $m \in \mathcal{U}[0,10]$, $x_0, y_0 \in \mathcal{U}[-1,1]$, $d = \{1,2\}$	0,1
5	$a\lvert(y-y_0)-m(x-x_0)\rvert+c$	$a \in \mathcal{N}[0,0.4]$, $m \in \mathcal{U}[-2,2]$, $x_0, y_0 \in \mathcal{U}[-1,1]$, $c \in \mathcal{U}[0,1]$	0,1

6	**if** $a_1 > 0$ **then**
	$a_2 \max(0, b_1(x-x_0))$
	$+a_3 \max(0, b_2(y-y_0))+c$
	else
	$a_2 \max(0, b_1(x-x_0)+$
	$b_2(y-y_0))+c$
		$a_i = \{-1,1\}$,
		$b_i \in \mathcal{N}[0,0.6]$
		$x_0, y_0 \in \mathcal{U}[-0.6,0.6]$,
		$c \in \mathcal{U}[0,1]$
			0,1

7	$a_1 \sin(f_{\text{Nyquist}}\pi(x-x_0))$ $\exp(a_3(x-x_0))$ $+a_2 \cos(f_{\text{Nyquist}}\pi(y-y_0))$ $\exp(a_3(y-y_0))+c$	$c \in \mathcal{U}[0,1]$, $a_1, a_2 \in \mathcal{N}[0,0.4]$, $x_0, y_0 \in \mathcal{U}[-1,1]$, $a_3 \in \mathcal{U}[-2,2]$	0,1

grid cell of $2\mathcal{N}$ for the evaluation of the analytical functions. Next, the analytical functions given in the Lagrange basis representation of degree $2\mathcal{N}$ are projected onto the Lagrange basis representation of degree $\mathcal{N}$, which is used for the simulation with DGSEM. Summing up, the input image $\mathbf{X}$ is build up with the normalized nodal values of the Lagrange polynomials of degree $\mathcal{N}$ resulting in $(\mathcal{N}+1)^2$ values. The true output $\mathbf{Y}$ is the binary edge map.

Appendix B. Comparsion of solution quality

The following section has been added by suggestion the reviewer.

In Fig. 22, we compare the density results for the three indicators considered and the Riemann problems from Sec. 4.1.1 on the fine mesh. All results are in very good visual agreement; the results for ANNSI are on par with those of the classical indicators. We note that we achieve the same general results for the other test cases considered in this paper, as well as for the different meshes investigated.

References

[1] D.S. Balsara, C. Altmann, C.-D. Munz, M. Dumbser, A sub-cell based indicator for troubled zones in RKDG schemes and a novel class of hybrid RKDG+HWENO schemes, J. Comput. Phys. 226 (1) (2007) 586–620.
[2] A.R. Barron, Universal approximation bounds for superpositions of a sigmoidal function, IEEE Trans. Inf. Theory 39 (3) (1993) 930–945.
[3] G.E. Barter, D.L. Darmofal, Shock capturing with PDE-based artificial viscosity for DGFEM: Part I. Formulation, J. Comput. Phys. 229 (5) (2010) 1810–1827.
[4] A. Beck, D. Flad, C.-D. Munz, Deep neural networks for data-driven LES closure models, J. Comput. Phys. 398 (2019) 108910.
[5] A.D. Beck, T. Bolemann, D. Flad, H. Frank, G.J. Gassner, F. Hindenlang, C.-D. Munz, High-order discontinuous Galerkin spectral element methods for transitional and turbulent flow simulations, Int. J. Numer. Methods Fluids 76 (8) (2014) 522–548.
[6] A. Burbeau, P. Sagaut, C.-H. Bruneau, A problem-independent limiter for high-order Runge–Kutta discontinuous Galerkin methods, J. Comput. Phys. 169 (1) (2001) 111–150.
[7] B. Cockburn, S. Hou, C.-W. Shu, The Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws. IV: the multidimensional case, Math. Comput. 54 (190) (1990) 545–581.
[8] R. Courant, K.O. Friedrichs, Supersonic Flow and Shock Waves, vol. 21, Springer Science & Business Media, 1999.
[9] G. Cybenko, Approximation by superpositions of a sigmoidal function, Math. Control Signals Syst. 2 (4) (1989) 303–314.
[10] V. Dolejší, M. Feistauer, C. Schwab, On some aspects of the discontinuous Galerkin finite element method for conservation laws, Math. Comput. Simul. 61 (3–6) (2003) 333–346.
[11] F. Ducros, V. Ferrand, F. Nicoud, C. Weber, D. Darracq, C. Gacherieu, T. Poinsot, Large-eddy simulation of the shock/turbulence interaction, J. Comput. Phys. 152 (2) (1999) 517–549.

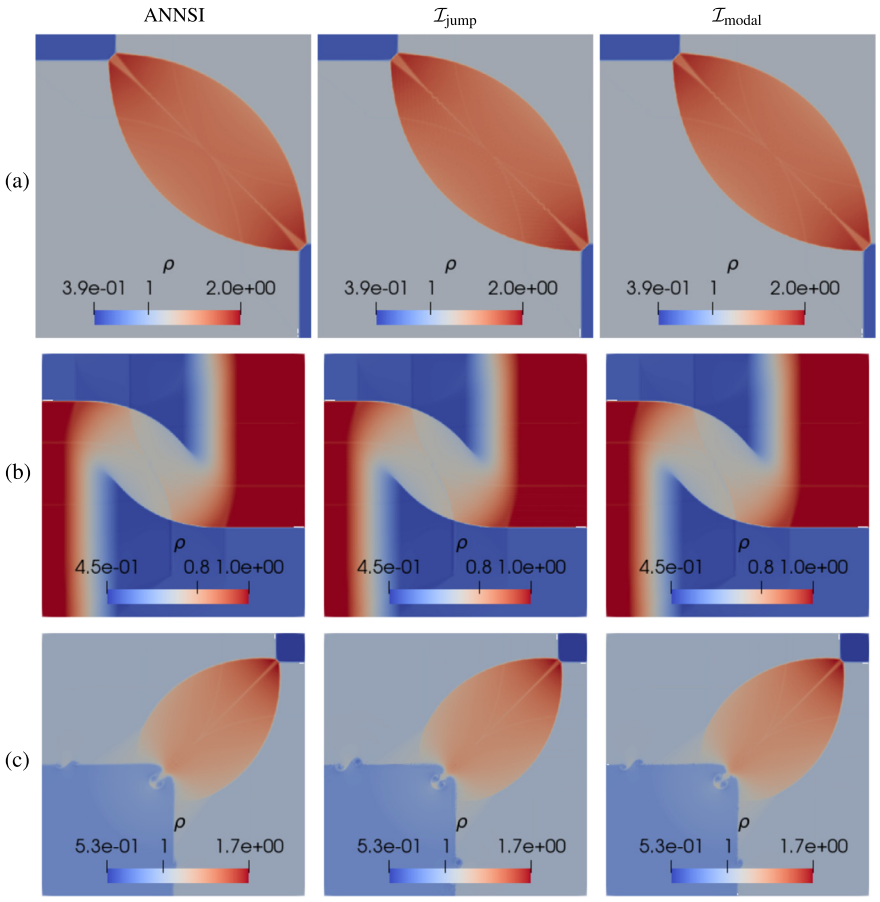


Fig. 22. $\mathcal{N} = 5$ results, fine mesh: Density contours computed with ANNSI (left), the jump (middle) and the modal indicator (right) for configuration 4 (a), 6 (b) and 12 (c) of the two-dimensional Riemann problems. The Riemann problems are calculated on a mesh with twice the resolution in each direction (100×100 elements).

[12] M. Dumbser, O. Zanotti, R. Loubère, S. Diot, A posteriori subcell limiting of the discontinuous Galerkin finite element method for hyperbolic conservation laws, J. Comput. Phys. 278 (2014) 47–75.
[13] M. Feistauer, V. Kučera, J. Prokopová, Discontinuous Galerkin solution of compressible flow in time-dependent domains, Math. Comput. Simul. 80 (8) (2010) 1612–1623.
[14] D. Gottlieb, C.-W. Shu, On the Gibbs phenomenon and its resolution, SIAM Rev. 39 (4) (1997) 644–668.
[15] M.M. Han Veiga, R. Abgrall, Towards a general stabilisation method for conservation laws using a multilayer perceptron neural network: 1D scalar and system of equations, in: ECCM - ECFD 2018 6th European Conference on Computational Mechanics (Solids, Structures and Coupled Problems) 7th European Conference on Computational Fluid Dynamics, Glasgow, United Kingdom, June 2018.
[16] R. Hartmann, P. Houston, Adaptive discontinuous Galerkin finite element methods for the compressible Euler equations, J. Comput. Phys. 183 (2) (2002) 508–532.
[17] S. Haykin, Neural Networks: A Comprehensive Foundation, Prentice Hall PTR, 1994.
[18] F. Hindenlang, G.J. Gassner, C. Altmann, A. Beck, M. Staudenmaier, C.-D. Munz, Explicit discontinuous Galerkin methods for unsteady problems, Comput. Fluids 61 (2012) 86–93.
[19] K. Hornik, Approximation capabilities of multilayer feedforward networks, Neural Netw. 4 (2) (1991) 251–257.
[20] A. Huerta, E. Casoni, J. Peraire, A simple shock-capturing technique for high-order discontinuous Galerkin methods, Int. J. Numer. Methods Fluids 69 (10) (2012) 1614–1632.
[21] S. Ioffe, C. Szegedy, Batch normalization: accelerating deep network training by reducing internal covariate shift, arXiv preprint, arXiv:1502.03167, 2015.
[22] A. Jameson, W. Schmidt, E. Turkel, Numerical solution of the Euler equations by finite volume methods using Runge Kutta time stepping schemes, in: 14th Fluid and Plasma Dynamics Conference, 1981, p. 1259.
[23] C.A. Kennedy, M.H. Carpenter, R.M. Lewis, Low-storage explicit Runge–Kutta schemes for the compressible Navier–Stokes equations, Appl. Numer. Math. 35 (3) (2000) 177–219.
[24] D.P. Kingma, J. Ba, Adam: a method for stochastic optimization, arXiv preprint, arXiv:1412.6980, 2014.
[25] A. Klöckner, T. Warburton, J.S. Hesthaven, Viscous shock capturing in a time-explicit discontinuous Galerkin method, Math. Model. Nat. Phenom. 6 (3) (2011) 57–83.
[26] D.A. Kopriva, Implementing Spectral Methods for Partial Differential Equations: Algorithms for Scientists and Engineers, Springer Science & Business Media, 2009.
[27] N. Krais, A. Beck, T. Bolemann, H. Frank, D. Flad, G. Gassner, F. Hindenlang, M. Hoffmann, T. Kuhn, M. Sonntag, et al., Flexi: a high order discontinuous Galerkin framework for hyperbolic-parabolic conservation laws, arXiv preprint, arXiv:1910.02858, 2019.
[28] L. Krivodonova, Limiters for high-order discontinuous Galerkin methods, J. Comput. Phys. 226 (1) (2007) 879–896.

[29] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.
[30] Y. LeCun, Y. Bengio, et al., Convolutional networks for images, speech, and time series, in: The Handbook of Brain Theory and Neural Networks 3361 (10) (1995) 1995.
[31] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444.
[32] Y. LeCun, B.E. Boser, J.S. Denker, D. Henderson, R.E. Howard, W.E. Hubbard, L.D. Jackel, Handwritten digit recognition with a back-propagation network, in: Advances in Neural Information Processing Systems, 1990, pp. 396–404.
[33] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, et al., Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.
[34] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, Z. Tu, Deeply-supervised nets, in: Artificial Intelligence and Statistics, 2015, pp. 562–570.
[35] R.J. LeVeque, Nonlinear conservation laws and finite volume methods, in: Computational Methods for Astrophysical Fluid Flow, Springer, 1998, pp. 1–159.
[36] R.J. LeVeque, et al., Finite Volume Methods for Hyperbolic Problems, vol. 31, Cambridge University Press, 2002.
[37] S.-P. Liou, A. Singh, S. Mehlig, D. Edwards, R. Davis, An image analysis based approach to shock identification in cfd, in: 33rd Aerospace Sciences Meeting and Exhibit, 1995, p. 117.
[38] Y. Liu, M.-M. Cheng, X. Hu, K. Wang, X. Bai, Richer convolutional features for edge detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 3000–3009.
[39] Y. Liu, Y. Lu, Y. Wang, D. Sun, L. Deng, F. Wang, Y. Lei, A cnn-based shock detection method in flow visualization, Comput. Fluids 184 (2019) 1–9.
[40] D. Lovely, R. Haimes, Shock detection from computational fluid dynamics results, in: 14th Computational Fluid Dynamics Conference, 1999, p. 3285.
[41] Z. Lu, H. Pu, F. Wang, Z. Hu, L. Wang, The expressive power of neural networks: a view from the width, in: I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems 30, Curran Associates, Inc., 2017, pp. 6231–6239.
[42] Y. Lv, Y.C. See, M. Ihme, An entropy-residual shock detector for solving conservation laws using high-order discontinuous Galerkin methods, J. Comput. Phys. 322 (2016) 448–472.
[43] A.L. Maas, A.Y. Hannun, A.Y. Ng, Rectifier nonlinearities improve neural network acoustic models, in: Proc. ICML, vol. 30, 2013, p. 3.
[44] M. Monfort, T. Luciani, J. Komperda, B. Ziebart, F. Mashayek, G.E. Marai, A deep learning approach to identifying shock locations in turbulent combustion tensor fields, in: T. Schultz, E. Özarslan, I. Hotz (Eds.), Modeling, Analysis, and Visualization of Anisotropy, Cham, Springer International Publishing, 2017, pp. 375–392.
[45] N.R. Morgan, S. Tokareva, X. Liu, A. Morgan, A machine learning approach for detecting shocks with high-order hydrodynamic methods, in: AIAA Scitech 2020 Forum, 2020, p. 2024.
[46] R. Paciorri, A. Bonfiglioli, A shock-fitting technique for 2d unstructured grids, Comput. Fluids 38 (3) (2009) 715–726.
[47] H.-G. Pagendarm, B. Seitz, An algorithm for detection and visualization of discontinuities in scientific data fields applied to flow data with shock waves, in: Scientific Visualization: Advanced Software Techniques, 1993, pp. 161–177.
[48] P.-O. Persson, J. Peraire, Sub-cell shock capturing for discontinuous Galerkin methods, in: 44th AIAA Aerospace Sciences Meeting and Exhibit, 2006, p. 112.
[49] S. Pirozzoli, Numerical methods for high-speed flows, Annu. Rev. Fluid Mech. 43 (2011) 163–194.
[50] S. Premasuthan, C. Liang, A. Jameson, Computation of flows with shocks using the spectral difference method with artificial viscosity, I: basic formulation and application, Comput. Fluids 98 (2014) 111–121.
[51] J. Qiu, C.-W. Shu, A comparison of troubled-cell indicators for Runge–Kutta discontinuous Galerkin methods using weighted essentially nonoscillatory limiters, SIAM J. Sci. Comput. 27 (3) (2005) 995–1013.
[52] J. Qiu, C.-W. Shu, Hermite WENO schemes and their application as limiters for Runge–Kutta discontinuous Galerkin method II: two dimensional case, Comput. Fluids 34 (6) (2005) 642–663.
[53] J. Rabault, M. Kuchta, A. Jensen, U. Réglade, N. Cerardi, Artificial neural networks trained through deep reinforcement learning discover control strategies for active flow control, J. Fluid Mech. 865 (2019) 281–302.
[54] D. Ray, J.S. Hesthaven, An artificial neural network as a troubled-cell indicator, J. Comput. Phys. 367 (2018) 166–191.
[55] D. Ray, J.S. Hesthaven, Detecting troubled-cells on two-dimensional unstructured grids using a neural network, J. Comput. Phys. 397 (2019) 108845.
[56] B. Riemann, et al., Über die Fortpflanzung ebener Luftwellen von endlicher Schwingungsweite, Verlag der Dieterichschen Buchhandlung, 1860.
[57] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning representations by back-propagating errors, Nature 323 (6088) (1986) 533.
[58] V. Rusanov, Processing and analysis of computation results for multidimensional problems of aerohydrodynamics, in: Proceedings of the Third International Conference on Numerical Methods in Fluid Mechanics, Springer, 1973, pp. 154–162.
[59] J. Schmidhuber, Deep learning in neural networks: an overview, Neural Netw. 61 (2015) 85–117.
[60] C.W. Schulz-Rinne, J.P. Collins, H.M. Glaz, Numerical solution of the Riemann problem for two-dimensional gas dynamics, SIAM J. Sci. Comput. 14 (6) (1993) 1394–1414.
[61] A. Sheshadri, A. Jameson, Shock detection and capturing methods for high order discontinuous-Galerkin finite element methods, in: 32nd AIAA Applied Aerodynamics Conference, 2014, p. 2688.
[62] C.-W. Shu, High-order finite difference and finite volume WENO schemes and discontinuous Galerkin methods for cfd, Int. J. Comput. Fluid Dyn. 17 (2) (2003) 107–118.
[63] C.-W. Shu, S. Osher, Efficient implementation of essentially non-oscillatory shock-capturing schemes, J. Comput. Phys. 77 (2) (1988) 439–471.
[64] M. Sonntag, Shape derivatives and shock capturing for the Navier-Stokes equations in discontinuous Galerkin methods, Dissertation, University of Stuttgart, 2017.
[65] M. Sonntag, C.-D. Munz, Efficient parallelization of a shock capturing for discontinuous Galerkin methods using finite volume sub-cells, J. Sci. Comput. 70 (3) (2017) 1262–1289.
[66] F. Van Veen, S. Leijnen, The neural network zoo, http://www.asimovinstitute.org/neural-network-zoo.
[67] E. Vorozhtsov, On shock localization by digital image processing techniques, Comput. Fluids 15 (1) (1987) 13–45.
[68] Z.J. Wang, K. Fidkowski, R. Abgrall, F. Bassi, D. Caraeni, A. Cary, H. Deconinck, R. Hartmann, K. Hillewaert, H.T. Huynh, et al., High-order cfd methods: current status and perspective, Int. J. Numer. Methods Fluids 72 (8) (2013) 811–845.
[69] P.J. Werbos, Backpropagation through time: what it does and how to do it, Proc. IEEE 78 (10) (Oct 1990) 1550–1560.
[70] P. Woodward, P. Colella, The numerical simulation of two-dimensional fluid flow with strong shocks, J. Comput. Phys. 54 (1) (1984) 115–173.
[71] Z. Wu, Y. Xu, W. Wang, R. Hu, Review of shock wave detection method in CFD post-processing, Chin. J. Aeronaut. 26 (3) (2013) 501–513.
[72] S. Xie, Z. Tu, Holistically-nested edge detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1395–1403.
[73] M. Yang, Z.-J. Wang, A parameter-free generalized moment limiter for high-order methods on unstructured grids, in: 47th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition, 2009, p. 605.
[74] J. Yu, J.S. Hesthaven, C. Yan, A data-driven shock capturing approach for discontinuous Galerkin methods, Tech. Rep., 2018.
[75] X. Zhong, C.-W. Shu, A simple weighted essentially nonoscillatory limiter for Runge–Kutta discontinuous Galerkin methods, J. Comput. Phys. 232 (1) (2013) 397–415.