Computer Methods in Applied Mechanics and Engineering 424 (2024) 116906

Contents lists available at ScienceDirect

Comput. Methods Appl. Mech. Engrg.

journal homepage: www.elsevier.com/locate/cma

Gradient-annihilated PINNs for solving Riemann problems: Application to relativistic hydrodynamics

Antonio Ferrer-Sánchez [a,b,*], José D. Martín-Guerrero [a,b], Roberto Ruiz de Austri-Bazan [c], Alejandro Torres-Forné [d,e], José A. Font [d,e]

[a] IDAL, Electronic Engineering Department, ETSE-UV, Avgda. Universitat s/n, Burjassot, 46100, Valencia, Spain
[b] Valencian Graduate School and Research Network of Artificial Intelligence (ValgrAI), Spain
[c] Instituto de Física Corpuscular CSIC-UV, c/Catedrático José Beltrán 2, Paterna, 46980, Valencia, Spain
[d] Departamento de Astronomía y Astrofísica, Universitat de València, Dr. Moliner 50, Burjassot, 46100, Valencia, Spain
[e] Observatori Astronòmic, Universitat de València, Catedrático José Beltrán 2, Paterna, 46980, Valencia, Spain

ARTICLE INFO

Keywords:
Riemann problem
Euler equations
Machine learning
Neural networks
Relativistic hydrodynamics

ABSTRACT

We present a novel methodology based on Physics-Informed Neural Networks (PINNs) for solving systems of partial differential equations admitting discontinuous solutions. Our method, called Gradient-Annihilated PINNs (GA-PINNs), introduces a modified loss function that forces the model to partially ignore high-gradients in the physical variables, achieved by introducing a suitable weighting function. The method relies on a set of hyperparameters that control how gradients are treated in the physical loss. The performance of our methodology is demonstrated by solving Riemann problems in special relativistic hydrodynamics, extending earlier studies with PINNs in the context of the classical Euler equations. The solutions obtained with the GA-PINN model correctly describe the propagation speeds of discontinuities and sharply capture the associated jumps. We use the relative l^2 error to compare our results with the exact solution of special relativistic Riemann problems, used as the reference "ground truth", and with the corresponding error obtained with a second-order, central, shock-capturing scheme. In all problems investigated, the accuracy reached by the GA-PINN model is comparable to that obtained with a shock-capturing scheme, achieving a performance superior to that of the baseline PINN algorithm in general. An additional benefit worth stressing is that our PINN-based approach sidesteps the costly recovery of the primitive variables from the state vector of conserved variables, a well-known drawback of grid-based solutions of the relativistic hydrodynamics equations. Due to its inherent generality and its ability to handle steep gradients, the GA-PINN methodology discussed in this paper could be a valuable tool to model relativistic flows in astrophysics and particle physics, characterized by the prevalence of discontinuous solutions.

1. Introduction

In computer science, Deep Learning (DL) stands as a pivotal tool, widely applied across various domains like image processing [1,2], natural language processing [3], machine translation [4], and time series prediction [5]. Key technologies involve recurrent neural networks (RNNs) [6] like long short-term memory (LSTM) architectures [7], attention mechanisms [8], and encoder–decoder structures [9,10]. Moreover, DL-based techniques demonstrate promise in evolving physical systems by solving the governing

* Corresponding author at: IDAL, Electronic Engineering Department, ETSE-UV, Avgda. Universitat s/n, Burjassot, 46100, Valencia, Spain.
E-mail addresses: Antonio.Ferrer-Sanchez@uv.es, anfesan6@uv.es (A. Ferrer-Sánchez).

https://doi.org/10.1016/j.cma.2024.116906
Received 13 September 2023; Received in revised form 6 February 2024; Accepted 5 March 2024
Available online 12 March 2024

partial differential equations (PDEs), fundamental in describing system behavior and evolution. Physics-Informed Neural Networks (PINNs) [11] have recently emerged as versatile algorithms for PDE solving, offering an alternative to traditional numerical methods in capturing physical scenarios. Since their introduction [11–13], PINNs have established themselves as advanced and widely adopted PDE solvers using machine learning [14,15]. They represent a competitive approach, integrating physical information into neural network architectures and training procedures, enabling the computation of PDE solutions by learning from data and problem-specific physics. The adaptability of PINNs in comprehending and solving physical systems has propelled their popularity in machine learning research and their direct application across scientific disciplines [16,17].

The universal approximation theorem validates Neural Networks (NNs) [18–20] for solving continuous PDEs. In general, NNs can integrate equation information into their loss function to solve equations along with initial and boundary conditions. However, their capability to handle discontinuous PDEs lacks theoretical backing, necessitating modifications in PINNs algorithms to ensure accurate representation. Recent research focuses on enhancing them to manage discontinuities like the compressible Euler equations [21]. These equations, describing fluid/gas flow, produce discontinuous solutions even from smooth initial data due to their inherent nonlinear hyperbolic nature. Riemann problems, e.g., the widely known *Sod Shock Tube* problem [22], introduce typical discontinuities. Studies [16,17] propose PINN adaptations to effectively handle such jumps. Further modifications [23] extend the baseline methodology for diverse discontinuous initial conditions, using strategies such as expanding training data domains and adjusting loss function weights. Additionally, PINNs have been explored for other systems prone to discontinuities, such as the Burgers equation and the convection–diffusion equation [24].

This paper investigates PINN efficiency in solving relativistic hydrodynamics equations (RH), a generalized form of classical Euler fluid equations and a nonlinear hyperbolic conservation system [25–28]. RH applies to scenarios where fluid velocity nears light speed in special relativity, crucial in extreme conditions found in particle physics and astrophysics. Proposing a Gradient-Annihilated PINN (GA-PINN) method, this approach detects discontinuities in physical spacetime using controlled factors tied to fluid variable gradients. The GA-PINN demonstrates competitive accuracy solving Riemann problems with analytical solutions and a second-order central high-resolution shock capturing scheme [24]. It effectively captures propagation speeds and fluid variable discontinuities in relativistic hydrodynamics equations.

The rest of the paper is organized as follows: Section 2 describes the background of our research with a literature review of PINNs, relativistic hydrodynamics, and the application of PINNs to classical hydrodynamics. Section 3 outlines the methodology of our GA-PINN algorithm. The numerical experiments used to assess our method and the corresponding results are presented in Section 4. Finally, in Section 5 we discuss the main conclusions of this work and mention possible suggestions for future work. Code for training examples has been provided by the authors.[1]

2. Background

2.1. Physics-informed neural networks

The PINNs method [11] uses inductive biases to solve PDE systems by encoding physics-related information. This approach aids model convergence and performance improvement by incorporating such information in network architecture, loss functions, initial conditions, and physical properties. Leveraging automatic differentiation available in modern frameworks [29], the algorithm computes derivatives for evaluation, enabling parameter optimization and architecture training. This method typically applies to PDE systems for variables $\boldsymbol{\mathcal{U}} := \boldsymbol{\mathcal{U}}(t, \boldsymbol{x})$ parameterized by a set of parameters $\boldsymbol{\Phi}$, over domain Ω and time interval $[0, T]$, considering initial conditions and boundary restrictions, all defined in Eq. (2).

$$\mathcal{F}\left(t, \boldsymbol{x}; \boldsymbol{\mathcal{U}}; \frac{\partial \boldsymbol{\mathcal{U}}}{\partial t}, \frac{\partial^2 \boldsymbol{\mathcal{U}}}{\partial t^2}, \dots; \frac{\partial \boldsymbol{\mathcal{U}}}{\partial x_1}, \dots, \frac{\partial \boldsymbol{\mathcal{U}}}{\partial x_D}; \frac{\partial^2 \boldsymbol{\mathcal{U}}}{\partial x_1 \partial x_1}, \dots, \frac{\partial^2 \boldsymbol{\mathcal{U}}}{\partial x_1 x_D}; \dots; \boldsymbol{\Phi}\right) = 0, \tag{1}$$

$$\begin{aligned} &\boldsymbol{x} = (x_1, \dots, x_D) \in \Omega, \quad t \in [0, T], \\ &\mathcal{B}(t, \boldsymbol{x}) = 0 \quad \text{with} \quad (t, \boldsymbol{x}) \in (0, T] \times \partial\Omega, \\ &\mathcal{IC}(t, \boldsymbol{x}) = 0 \quad \text{with} \quad (t, \boldsymbol{x}) \in \{0\} \times \Omega. \end{aligned} \tag{2}$$

The operator $\mathcal{F}$ in Eq. (1) encapsulates the k physical constraints, $\mathcal{R}_k(t, \boldsymbol{x})$, while $\mathcal{B}$ and $\mathcal{IC}$ denote predefined sets of domain points crucial for training the neural network. The architecture relies on the parameter set Θ, generating $\boldsymbol{\mathcal{U}}(t, \boldsymbol{x}; \Theta) := \boldsymbol{\mathcal{U}}_\Theta(t, \boldsymbol{x}) \approx \boldsymbol{\mathcal{U}}(t, \boldsymbol{x})$. Training aims to approximate a vector of physical variables to the actual ones. Solution derivation involves a loss function as the sum of distinct terms, each defined within a unique domain.

$$\mathcal{L} = \sum_i \omega_i \mathcal{L}_i := \omega_{IC}\mathcal{L}_{IC} + \omega_B \mathcal{L}_B + \omega_R \mathcal{L}_R. \tag{3}$$

Each one of the addends in Eq. (3) can be defined using some suitable metric such as *Mean Squared Error* (MSE). A weight vector $\boldsymbol{\omega} = (\omega_{IC}, \omega_B, \omega_R)$ controls term mixing in Eq. (3). These weights can be fixed, variable, or linked to physical parameters. Additional factors, like experimental loss considerations, may also be included. The method of simultaneous learning of initial and boundary

[1] GA-PINNsGitHubrepository.

conditions alongside differential equations is denoted as *soft enforcement*, whose general loss terms are written explicitly in Eq. (4). Conversely, strictly imposing these conditions initially is termed *hard enforcement* [30].

$$\begin{aligned}\mathcal{L}_{IC} &:= \frac{1}{N_{IC}} \sum_{\{0\}\times\Omega} |\boldsymbol{\mathcal{U}}_\Theta(0,\boldsymbol{x}) - \boldsymbol{\mathcal{U}}(0,\boldsymbol{x})|^2,\\ \mathcal{L}_{\mathcal{B}} &:= \frac{1}{N_{\mathcal{B}}} \sum_{(0,T]\times\partial\Omega} |\boldsymbol{\mathcal{U}}_\Theta(t,\boldsymbol{x}\in\partial\Omega) - \boldsymbol{\mathcal{U}}(t,\boldsymbol{x}\in\partial\Omega)|^2,\\ \mathcal{L}_{\mathcal{R}} &:= \frac{1}{N_{\mathcal{R}}} \sum_{(0,T]\times\Omega} \underbrace{\sum_k |\mathcal{R}_k(t,\boldsymbol{x};\Theta)|^2}_{|\mathcal{F}|^2}.\end{aligned} \tag{4}$$

2.2. Classical and relativistic hydrodynamics

The Navier–Stokes equations describe fluid behavior with viscosity. When dissipative components are negligible compared to convective ones, the Euler equations, representing conservation of mass, momentum, and energy in an inviscid fluid, emerge in a general conservative form (Eq. (5)). These equations highlight conservation in systems like the Euler equations [31], where $\boldsymbol{\mathcal{U}} = \boldsymbol{\mathcal{U}}(t,\boldsymbol{x})$ represents the physical variables, $f(\boldsymbol{\mathcal{U}})$ is the flux vector which depends on the conserved variables $\boldsymbol{\mathcal{U}}$, and $S(\boldsymbol{\mathcal{U}})$ is a vector accounting for possible source terms. For the state vector $\boldsymbol{\mathcal{U}}^T = (\rho, j, e)$, the classical Euler equations are derived, represented in one spatial dimension using Cartesian coordinates as Eq. (6).

$$\frac{\partial \boldsymbol{\mathcal{U}}}{\partial t} + \nabla \cdot f(\boldsymbol{\mathcal{U}}) = S(\boldsymbol{\mathcal{U}}). \tag{5}$$

$$\frac{\partial}{\partial t}\begin{pmatrix}\rho\\ j\\ e\end{pmatrix} + \frac{\partial}{\partial x}\begin{pmatrix} j\\ p + \frac{j^2}{\rho}\\ \frac{j}{\rho}(e+p)\end{pmatrix} = 0. \tag{6}$$

Fluid properties, such as density (ρ), momentum density ($j = \rho u$ in the x direction), velocity (u), and total energy density per unit volume (e), can be described in Cartesian coordinates. To complete the system along with Eq. (6), an equation of state (EOS) linking fluid pressure to density and energy, $p = p(\rho, e)$, is required. The ideal gas EOS (Eq. (7), Γ being the adiabatic index) is a common choice in modeling system evolution using numerical methods.

$$p = (\Gamma - 1)\left(e - \frac{1}{2}\frac{j^2}{\rho}\right). \tag{7}$$

The classical Euler equations (Eq. (6)) must be adapted for high-energy fluid dynamics, requiring a relativistic treatment when working with high velocities. Extreme astrophysical systems with compact objects (neutron stars, black holes, etc.) and high-energy collisions of heavy ions demand this approach (see e.g. [28,32] and references therein). Relativistic hydrodynamics governs fluid behavior in situations involving either high-speed flows nearing the speed of light or strong gravitational fields causing spacetime curvature effects on fluid motion.

Understanding these systems depends heavily on numerical simulations. Accurate numerical methods exploiting conservative forms of relativistic hydrodynamics equations, like high-resolution shock-capturing schemes (HRSC) [33], are crucial for modeling. General relativistic hydrodynamics equations (GRHD) mirror conservation laws for matter current density, momentum, and energy, derived from components of the energy–momentum tensor ($\mathbf{T}$) and the four-vector of rest mass current ($\boldsymbol{J}$), both defined in Eq. (8).

$$J^\mu = \rho u^\mu, \qquad T^{\mu\nu} = \rho h u^\mu u^\nu + p g^{\mu\nu}, \qquad h := 1 + e + \frac{p}{\rho}. \tag{8}$$

The fluid dynamics equations in relativistic contexts involve fundamental quantities like rest-mass density (ρ), pressure (p), specific enthalpy (h), specific internal energy (e), fluid four-velocity (u^μ), and spacetime metric ($g^{\mu\nu}$) within a Riemannian spacetime manifold, $\mathscr{M}$. Natural units ($c = 1$) are used. An ideal gas EOS is often employed for modeling (Eq. (9)). Moreover, expressing the covariant equations shown in Eq. (8) in a specific coordinate system is crucial for numerical applications. The $\{3+1\}$ formulation in general relativity allows the derivation of GRHD equations in a conservative form. This paper focuses on GRHD equations within Minkowski spacetime, denoted as $\mathscr{M}^4$, of special relativity, following details in [25]. For a comprehensive theoretical approach to obtain GRHD equations in conservation form, refer to [26]. The relativistic Euler equations in Minkowski spacetime, one spatial dimension, Cartesian coordinates (without source terms) are written in Eq. (10) below.

$$p = (\Gamma - 1)\rho e. \tag{9}$$

$$\frac{\partial}{\partial t}\begin{pmatrix}D\\ S\\ \tau\end{pmatrix} + \frac{\partial}{\partial x}\begin{pmatrix} Du\\ Su + p\\ S - Du\end{pmatrix} = 0. \tag{10}$$

In this formulation, the vector of the conserved quantities is $\mathcal{U} := (D, S, \tau)$. Its components are the relativistic densities of mass, momentum (along direction j) and energy, respectively, defined in Eq. (11).

$$\begin{aligned} D &= \rho W, \\ S_j &= \rho h W^2 u_j, \\ \tau &= \rho h W^2 - p - D. \end{aligned} \tag{11}$$

W stands for the Lorentz factor, determined by the fluid velocity across spatial dimensions (represented by the j index) as defined in Eq. (12). It is responsible for a greater coupling of the relativistic Euler equations with respect to their classical counterpart.

$$W^2 := \frac{1}{1 - \sum_{j=1}^{3} u_j^2}. \tag{12}$$

In addition, the specific enthalpy, h, can be written as $h = 1 + \Gamma e$ for the specific case of an ideal gas by using its definition in Eqs. (8) and (9).

Accurately solving relativistic fluid dynamics where strong shock waves are present is a vital task in high-energy physics and astrophysics (see [27,28] and references therein). Shock waves persist in relativistic hydrodynamics (Eq. (10)), distinct from the classical case (Eq. (6)), due to the intricate coupling induced by the Lorentz factor. Modern HRSC schemes are effective in handling ultra-relativistic flows ($W \sim 10$ and higher) [28]. The absence of an analytical expression defining primitive variables, (ρ, u, p), in terms of the conserved ones, (D, S, τ), poses an additional challenge in numerical simulations, especially with finite difference-based codes like HRSC schemes. However, our approach using PINN-based methodology overcomes this limitation by allowing neural networks to handle both variable sets concurrently. This facilitates direct comparison with HRSC-based results.

2.3. Related work

The PINNs algorithm, while valuable for solving numerous physical systems using DL, can encounter limitations, especially in scenarios involving shock waves or discontinuities in initial and/or boundary conditions. Depending on neural architecture and hyperparameter selection (e.g., hidden layers, neurons, activation functions, learning rate, optimizer), performance can vary significantly.

Theoretical exploration on gradient behavior ($\nabla_\Theta \mathcal{L}$) concerning the loss function used reveals the tendency for these gradients related to boundary and initial conditions to diminish within the neural network [34]. This necessitates balancing each term in the final function. Adaptive weights (ω) in Eq. (3)—depended upon these gradients—were proposed, yielding competitive results [35]. These studies typically operate within a framework where initial and boundary conditions are softly enforced.

In optimizing PINNs, [36] focused on refining the sampling method within the $[0, T] \times \Omega$ domain, proposing diverse sampling techniques and oversampling high residue regions. Alternatively, [37] suggested directly adjusting activation function parameters (*ReLU*, *tanh*, *sigmoid*), significantly improving convergence and performance. Addressing discontinuity challenges, some studies advocated the method of characteristics for Euler equations, altering conservation equations (Eq. (5)) to utilize eigenvalues and eigenvectors of the Jacobian matrix (see [21], Section 4.1.6). Recent works like [38] adapted the loss function by introducing a velocity gradient-dependent factor in Euler equations, notably enhancing discontinuity detection in the algorithm.

In this work, we present Gradient-Annihilated PINNs (GA-PINNs), an alternative approach to the methods available in the literature for resolving physical systems with discontinuities throughout their evolution. Our investigation is focused on the physical system given by the RH (Euler) equations.

3. Methodology

For our methodology, we will consider a fully connected dense neural network with a total number of layers K, including the input and output layers as well as the internal (hidden) ones. Each layer k will consist of N_k neurons in general. Furthermore, each layer will take a nonlinear activation function, σ_k, which will transform the output of said layer before being transported to the subsequent layer. Mathematically, one can write the output of the kth layer as

$$\mathcal{U}_{\Theta,k} := \sigma_k \left(\boldsymbol{W}_k \mathcal{U}_{\Theta,k-1} + \boldsymbol{b}_k \right), \tag{13}$$

where $\boldsymbol{W}_k \in \mathbb{R}^{N_k \times N_k}$ and $\boldsymbol{b}_k \in \mathbb{R}^{N_k}$ correspond to the network weights and biases of the kth layer.

Hence, when additional parameters are not taken into account, the collection of modifiable quantities in a typical neural network can be represented as the combination of weights and biases, i.e. $\Theta := \{\boldsymbol{W}_k, \boldsymbol{b}_k\}_{1 \le k \le K}$. With respect to activation functions (σ), it is typical for them to vary across layers and even across individual neurons. Depending on the particular problem at hand, distinct functions may be employed, such as the hyperbolic tangent or the *sigmoid* function, among others.

When constructing a neural network, there exists a substantial degree of flexibility in determining its dimensions, including the number of neurons per layer and the selection of the activation function employed across the various layers. These activations can contain, in turn, some hyperparameters that can be chosen a priori, i.e., before training. One could discuss the slope denoted as a, which works as a multiplicative coefficient for the independent variable. Nonetheless, the selection of such hyperparameters is typically achieved through an empirical approach. Hence, by taking into account the output of the layer k in Eq. (13) and utilizing

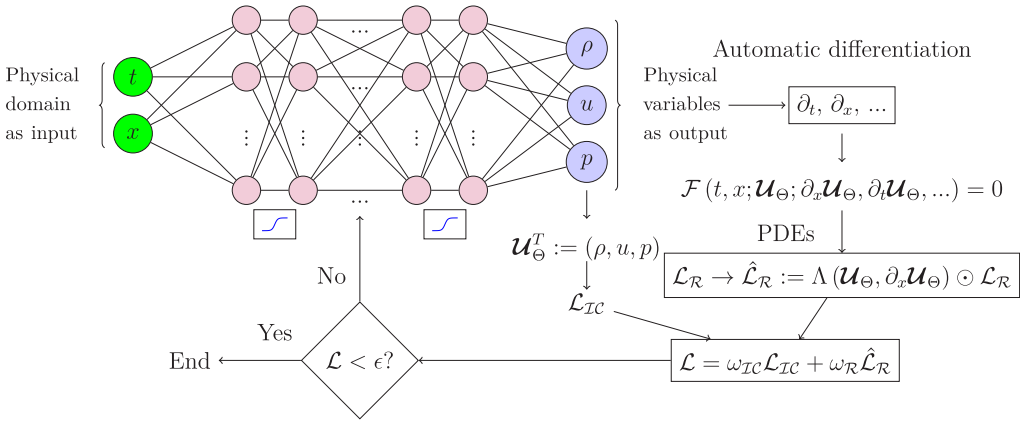

Fig. 1. The methodology involves a neural network processing temporal and spatial physical variables (t, x). Its architecture, including layers, activation functions, and neurons, is flexible. The network outputs primitive variables (ρ, u, p), differentiated automatically to compute physical constraints and residuals of the initial conditions. These results combine using weights (ω_{IC}, ω_R), evaluating the loss function $\mathcal{L}$ to update parameters Θ.

the input $(t, x) \in [0, T] \times \Omega$, where $\Omega \subset \mathbb{R}$, as the input to the network, it is feasible to predict the physical parameters as the resultant output:

$$\boldsymbol{\mathcal{U}}(t, x) \approx \boldsymbol{\mathcal{U}}_\Theta(t, x) = \sigma_K \left(\boldsymbol{\mathcal{U}}_{\Theta,K} \circ \sigma_{K-1} \circ \boldsymbol{\mathcal{U}}_{\Theta,K-1} \circ ... \circ \sigma_1 \circ \boldsymbol{\mathcal{U}}_{\Theta,1} \right)(t, x), \tag{14}$$

where $\circ$ represents the composition operator.

In Eq. (14), each layer k of the neural network is associated with an activation function σ_k. This function acts on each element of the corresponding output tensor, being it possible for the activation functions to vary across layers. Notably, the activation function σ_K of the final output layer may be subject to additional constraints, such as ensuring non-negative or bounded values, in order to align with physical requirements. Once the activation functions have been determined, the problem of training the network can be cast as one of minimizing the loss function, which involves jointly optimizing the weights and biases.

3.1. Modified loss function and optimization

The main novelty of the GA-PINNs will rely on a basic alteration to the employed loss function. As indicated in Eq. (3), when considering the *soft enforcement* of the initial and/or boundary conditions, there is a loss term that represents the fit of the physical constraints on the domain $(0, T] \times \Omega$, i.e. $\mathcal{L}_\mathcal{R}$. Following the restriction in Eq. (1), this part of the loss can be expressed as a MSE,

$$\mathcal{L}_\mathcal{R} := \frac{1}{N_\mathcal{R}} \sum_{(0,T]\times\Omega} |\mathcal{F}|^2, \tag{15}$$

where $N_\mathcal{R}$ represents the number of points in the sampled internal physical domain, also called *collocation points*. For its part, $\mathcal{F}$ written in Eq. (1) represents our system of PDEs to be solved and in general it will depend on the physical domain t and x as well as on the physical fields $(\boldsymbol{\mathcal{U}})$ and their derivatives with respect to space and time. In Riemann problems and in the presence of shock waves, any neural architecture may have difficulty resolving such discontinuities due to its tendency to solve problems smoothly. In this way, and with the aim of improving the resolution of discontinuities with PINNs, we propose the following modification of the loss of placement function (Eq . (15)) as follows

$$\mathcal{L}_\mathcal{R} \to \hat{\mathcal{L}}_\mathcal{R} := \Lambda(\boldsymbol{\mathcal{U}}_\Theta, \partial_x \boldsymbol{\mathcal{U}}_\Theta) \odot \mathcal{L}_\mathcal{R}, \tag{16}$$

where $\odot$ represents the element-wise product. The variable Λ introduced in Eq. (16) will be considered as a function dependent on physical variables, denoted by $\boldsymbol{\mathcal{U}}_\Theta$, and their first spatial derivatives, written as $\partial_x \boldsymbol{\mathcal{U}}_\Theta$. In one-dimensional problems, the spatial dimension reads as x. The primary aim of Λ is to serve as a vector of weights, providing a mixture of residuals that is distinct from what would be obtained by calculating a simple arithmetic mean. The functional form of Λ can be arbitrary, but we shall define it in accordance with the following assumption:

Hypothesis. The PINN increases its capacity to resolve discontinuities in PDEs when it is forced to be deactivated in said areas.

Which, in other words, corresponds to the fact that the PINN improves its performance (the error of its predicted solution with respect to a base resolution is lower) when the network does not pay as much attention to the discontinuous areas since it does not try to solve them smoothly. From a mathematical point of view, we can identify discontinuous areas as those that present a large gradient. Therefore, it is possible to define functions that take this into account, such as

$$\Lambda_1 \left(\boldsymbol{\mathcal{U}}_\Theta, \partial_x \boldsymbol{\mathcal{U}}_\Theta \right) := \frac{1}{L} \sum_{i=1}^{L} \frac{1}{1 + \alpha_i |\partial_x \boldsymbol{\mathcal{U}}_{\Theta,i}|^{\beta_i}}, \tag{17}$$

where L corresponds to the size of the $\mathcal{U}$ vector (number of variables).

In Eq. (17), $\mathcal{U}_i$ represents each of the predicted physical variables, i.e. $\mathcal{U}^T = (\rho, u, p)$. For their part, one can define $\boldsymbol{\alpha} := \{\alpha_i\}_{i=1}^{L}$ and $\boldsymbol{\beta} := \{\beta_i\}_{i=1}^{L}$ as two additional set of hyperparameters that will measure to what extent the respective spatial gradients are considered in the loss function. In particular, for the resolution of the relativistic Euler equations, this mathematical expression can be particularized to

$$\Lambda_1\left(\mathcal{U}_\Theta, \partial_x \mathcal{U}_\Theta\right) := \Lambda_1(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{3}\left(\frac{1}{1+\alpha_\rho|\partial_x\rho|^{\beta_\rho}} + \frac{1}{1+\alpha_u|\partial_x u|^{\beta_u}} + \frac{1}{1+\alpha_p|\partial_x p|^{\beta_p}}\right). \tag{18}$$

From Eq. (18) we can see that in areas where the gradient of the variables shoots up, i.e. in areas with discontinuities, we will have $|\partial_x\rho|, |\partial_x u|, |\partial_x p| \gg 1$, so that

$$\lim_{|\partial_x\rho|,|\partial_x u|,|\partial_x p|\to\infty} \Lambda_1(\boldsymbol{\alpha}, \boldsymbol{\beta}) = 0. \tag{19}$$

In this limiting case the part of the loss function that adjusts the physical constraints, $\mathcal{L}_\mathcal{R}$, would tend to be deactivated at those points in space with discontinuities. In contrast, in the opposite limit, when $|\partial_x\rho|, |\partial_x u|, |\partial_x p| \ll 1$, one has

$$\lim_{|\partial_x\rho|,|\partial_x u|,|\partial_x p|\to 0} \Lambda_1(\boldsymbol{\alpha}, \boldsymbol{\beta}) = 1, \tag{20}$$

thus recovering the PINNs base algorithm. Special care should be taken in choosing the values of $(\boldsymbol{\alpha}, \boldsymbol{\beta})$, but according to our hypothesis there should be at least one set of both parameters for which the error made by our algorithm with respect to a solution considered as true (e.g. analytical solution) should be smaller than or at least equal to the error obtained with an ordinary PINNs-based model without any further modification. In this way, with this simple modification it should be possible to optimize the algorithm, taking its resolving capacity to the maximum and outperforming the PINN base algorithm. Apart from the function Λ presented in Eq. (17) other expressions can also be considered as long as they present similar behaviors at the limits under consideration, such as the one written in Eq. (21) below. Again, particularizing for the relativistic Euler equations, this expression can be reformulated as Eq. (22).

$$\Lambda_2(\mathcal{U}_\Theta, \partial_x \mathcal{U}) := \frac{1}{1+\sum_{i=1}^{L}\alpha_i|\partial_x \mathcal{U}_{\Theta,i}|^{\beta_i}}. \tag{21}$$

$$\Lambda_2\left(\mathcal{U}_\Theta, \partial_x \mathcal{U}_\Theta\right) := \Lambda_2(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{1+\alpha_\rho|\partial_x\rho|^{\beta_\rho}+\alpha_u|\partial_x u|^{\beta_u}+\alpha_p|\partial_x p|^{\beta_p}}. \tag{22}$$

The same limits for Λ_1 can be derived directly for Λ_2, obtaining

$$\lim_{|\partial_x\rho|,|\partial_x u|,|\partial_x p|\to\infty} \Lambda_2(\boldsymbol{\alpha}, \boldsymbol{\beta}) = 0, \qquad \lim_{|\partial_x\rho|,|\partial_x u|,|\partial_x p|\to 0} \Lambda_2(\boldsymbol{\alpha}, \boldsymbol{\beta}) = 1. \tag{23}$$

The example functions outlined in Eqs. (18) and (22) exhibit a desirable characteristic whereby they gradually deactivate the loss function within the internal physical domain. Our aim is to direct the focus of the network toward regions with lower gradients, thereby allocating greater computational effort to resolving smoother areas, which are conventionally deemed more tractable, as opposed to grappling with discontinuities. This strategic emphasis enables the PINN to adeptly address points immediately preceding and following the location of a discontinuity within the physical domain. Consequently, the discontinuity itself is effectively characterized as the network excludes this specific area from its accessible domain and resolves the surrounding regions.

As a visual example, Fig. 2 shows the function $\frac{1}{1+\alpha\, x^\beta}$ for different values of the hyperparameters mentioned. Observations suggest that as the parameters α and β increase, the function exhibits a decreasing trend. However, it is noteworthy that a substantially greater decrease is typically observed for the latter, with differences often spanning several orders of magnitude. The underlying cause for this phenomenon is attributed to the nature of the β parameter, which serves as an exponent to the gradient values. This nonlinear transformation accentuates the disparities between the maximum and minimum values. Hence, upon scrutinizing the shape of the function Λ as expressed in Eq. (18), it is evident that the neural network stands to gain from the utilization of these weights. This is because the discontinuities that manifest uniformly across all variables possess Λ values that converge toward zero. For those that are not common for all, the network might have more difficulty describing them, but as we will see, the set of hyperparameters (α, β) plays an important role in these cases. All the components converge to a competitive resolution, as it will be shown in the Results Section 4.

While the set $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ indicates how gradients are accounted for in the loss function, the parameters $(\omega_{IC}, \omega_\mathcal{R})$ represent a mixture vector that indicates in what proportion the respective parts of the total loss function (Eq. (3)) are to be weighed. Depending on the relative value of these hyperparameters, the network will focus on solving one of the individual losses first, thus adjusting one domain before others. In general, when working with PINNs through *soft enforcement*, networks benefit from using large weights for the initial condition loss versus smaller weights for the part of the loss that tries to fit the physical constraints, i.e. $\omega_{IC} \gg \omega_\mathcal{R}$ [23].

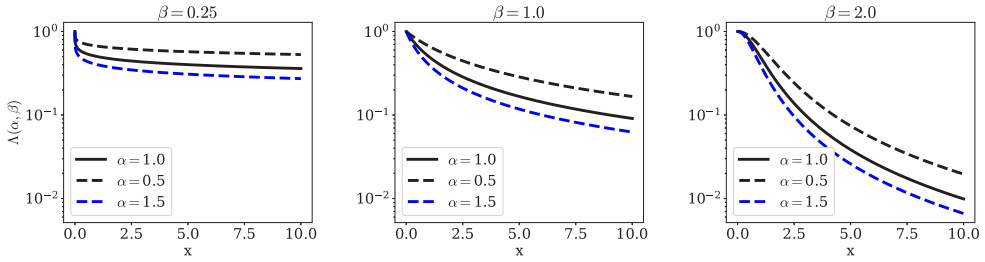

Fig. 2. Example function $\frac{1}{1+\alpha\, x^{\beta}}$ against the independent variable x for different values of α and β, all in the same scale. As both increase, the output tends to decrease in a general way.

It is based on the a priori knowledge that the initial conditions determine the entire evolution of the system in time $t > 0$. So by first adjusting these conditions, the network succeeds in converging faster and better to its correct temporal evolution.

Algorithm 1: Gradient-Annihilated PINN algorithm for the relativistic Euler problem (Eq. (10)).

Input: Physical domain: $t \in [0, T]$ and $x \in \Omega \subset \mathbb{R}$ in the one-dimensional case.

Output: Set of primitive variables predicted by the NN, i.e. $\boldsymbol{\mathcal{U}}_{\theta}^{T} := (\rho, u, p)$, in the domain.

1 **Specify the training set:** Generate the domain (t, x) using some sampling method (e.g. *Uniformly random sampling* or *Sobol sequence* [39]).

2 **Construct the neural network** as a dense fully-connected NN with K layers in total and a specific number of neurons. Initialize its set of parameters, $\Theta := \{\boldsymbol{W}_k, \boldsymbol{b}_k\}_{1 \le k \le K}$, in a random way.

3 Specify the activation functions for the different layers and output variables.

4 **Compute the MSE for the predicted set of variables at** $(t, x) \in \{0\} \times \Omega$:

$$\mathcal{L}_{IC} = \frac{1}{N_{IC}} \sum_{\{0\} \times \Omega} |\boldsymbol{\mathcal{U}}_{\Theta}(0, x) - \boldsymbol{\mathcal{U}}(0, x)|^2.$$

5 **Calculate the spatial and temporal derivatives from the physical output variables**, $\boldsymbol{\mathcal{U}}_{\Theta}$, and determine the residuals of the differential equation, $|\mathcal{F}|^2$, for each point $(t, x) \in (0, T] \times \Omega$:

$$\mathcal{L}_{\mathcal{R}} := \frac{1}{N_{\mathcal{R}}} \sum_{(0,T] \times \Omega} |\mathcal{F}|^2.$$

6 **Calculate the function** $\Lambda\left(\boldsymbol{\mathcal{U}}_{\Theta}, \partial_x \boldsymbol{\mathcal{U}}_{\Theta}\right)$ and modify $\mathcal{L}_{\mathcal{R}}$ performing the product element by element:

$$\mathcal{L}_{\mathcal{R}} \to \hat{\mathcal{L}}_{\mathcal{R}} := \Lambda \odot \mathcal{L}_{\mathcal{R}}.$$

7 **Specify the final loss metric** using the mix vector $(\omega_{IC}, \omega_{\mathcal{R}})$:

$$\mathcal{L} = \omega_{IC} \mathcal{L}_{IC} + \omega_{\mathcal{R}} \hat{\mathcal{L}}_{\mathcal{R}}. \tag{24}$$

8 **Update the set of network variables** by backpropagation, $\{\boldsymbol{W}_k, \boldsymbol{b}_k\}_{1 \le k \le K}$, that minimizes the total loss using a certain defined optimizer:

$$\Theta^{*} := \arg\, \min\left(\mathcal{L}(\Theta)\right).$$

By considering all the relevant factors, it is feasible to construct the algorithm of our GA-PINN methodology, which assists us both visually, as shown in Fig. 1, and in writing, through the step-by-step Algorithm 1. In the initial stages, it is critical to create the training data. In contrast to standard machine learning models, where the complete dataset can be divided into training and testing data, both of which contain certain features that the model perceives and a label (target) that it endeavors to forecast, in PINNs, the training is constituted by the points in the physical domain $(t, x) \in [0, T] \times \Omega$ itself. Prior to any further development, this step is vital.

In general, domain sampling for PINNs does not consider the creation of a test data set unless the goal is to make a prediction at a specific later point in time, for example (e.g. using LSTMs). Within our framework, it is viable to utilize the L^2 norm as a means of comparing an existing solution, referred to as the "ground truth" (such as an analytical solution), with the ultimate prediction produced by the neural architecture, denoted as $\boldsymbol{\mathcal{U}}_{\Theta}(t, x)$.

Consequently, it can be hypothesized beforehand that the point sampling technique may have a significant influence on the effectiveness of our architecture. Nevertheless, as the size of the physical domain increases to a considerable extent, this factor may no longer be decisive. This is a crucial consideration because we aim to minimize any dependence of our findings on factors like the seed utilized in random sampling. Therefore, scenarios where the number of points tends toward infinity are preferable. In Section 3.2 we give a slightly longer description of some of the different sampling methods that can be used, but throughout our work we have focused on the Sobol sequence method [39], which fills the domain in a very uniform way. Once the domain (t, x) is created, it takes on the role of input to the neural model, passes through all internal layers and comes out of said space as a prediction for the physical variables, i.e. $\boldsymbol{\mathcal{U}}_{\Theta}(t, x)$. At this point, automatic differentiation methods are used to compute the temporal and spatial derivatives of the output variables. Afterwards, the respective parts of the loss function are calculated as an

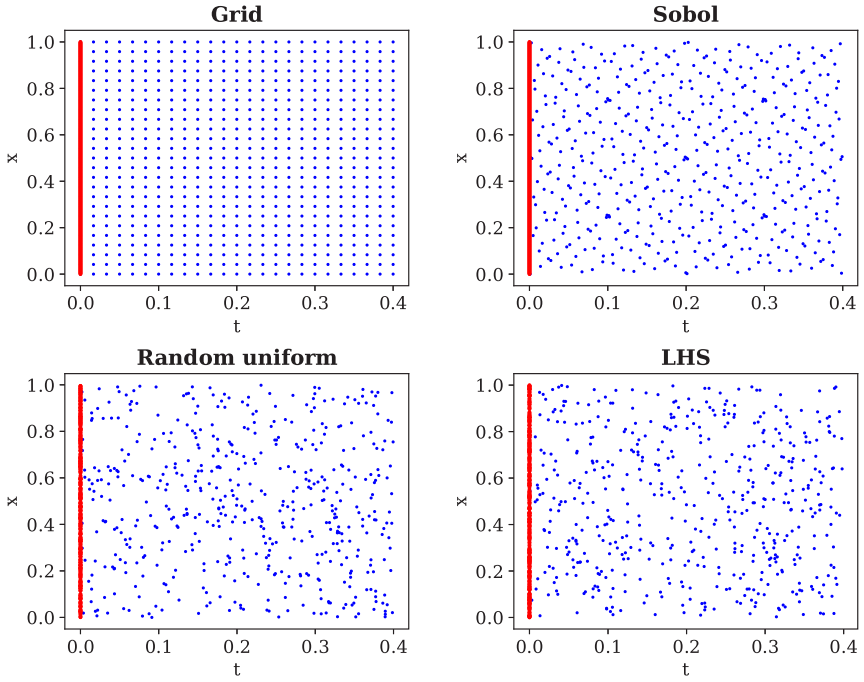

Fig. 3. Examples of 500 points generated in the space $[0, 0.4] \times [0, 1]$ for different sampling methods. The initial domain points are shown in red, i.e. points for $t = 0$, and the collocation or internal points of the domain on which the residual of the PDEs system will be calculated are shown in blue. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

MSE by carefully applying the weights computed using Λ to be propagated back through all the layers of the neural architecture and update the parameter set $\{\boldsymbol{W}_k, \boldsymbol{b}_k\}_{1 \leq k \leq K}$ per gradient descent. This process is repeated for each epoch until the model converges.

3.2. Construction of physical domain: sampling methods

PINN algorithms need to have the physical space as input for each of the problems that arise. As we have commented in Section 2.1, in general, this will be made up of several subdomains that will mainly attend to each of the terms of the final loss function (Eq. (3)). In its entirety, it can be expressed mathematically as $(t, x) \in [0, T] \times \Omega$, where $\Omega \subset \mathbb{R}$ in one-dimensional problems as the ones analyzed in this paper, and where the temporal dimension runs from the initial time to a some arbitrary final instant T that will depend on the problem presented. In our case, this physical space is set from start to finish for each one of the results, with the random seed set from the beginning so that all training sessions are fully repeatable and reproducible. It should be noted that despite the fact that there is some literature on the use of adaptive methods to sample the domain such as the residual-based adaptive refinement (RAR) method [36] so that physical space is altered during training, we will not consider an adaptive method, but will focus exclusively on the effect of our presented methodology and the extent to which the loss function has resolving power.

Regarding the sampling method used, this is a somewhat arbitrary decision that is usually taken from the initial point of the work and which, for consistency, remains fixed for all the results that can be obtained. Sampling methods such as the fixed grid of uniformly distributed points or uniform sampling of random points have been widely used in studies dealing with PINNs and their variations, regardless of the nature of the physical problem. As a visual aid, we show examples of different methods in Fig. 3, with the uniform grid and random sampling forming the left-hand column. In the right column we present the Sobol sampling and the Latin hypercube sampling (LHS) [40,41] which is a statistical method used to generate quasi-random samples using Monte Carlo by intervals taking as a base the same probability and Normal distribution for each one. The Sobol sequence method has been selected for implementation in our algorithm, owing to its documented high yields in fixed sampling within the literature [36]. This sampling method generates points in powers of two in a quasi-random manner, providing domains with relatively low discrepancy and, therefore, achieving a physical space with less overlapping of points and fewer areas without representation, but without neglecting the intrinsic randomness of the method.

4. Numerical experiments and results

In this section we present the results for the solution of the special RH equations in one spatial dimension, given by Eq. (10), considering different initial conditions. As mentioned above, solving those equations is of great interest in some areas of physics, such as high-energy physics (e.g. to model heavy-ion collisions) and astrophysics (e.g. to study gamma-ray bursts or the propagation

Table 1
L^2 norm relative errors of our results against the exact solution for the experiments considered. The errors are presented in sets of three boxes, where each column in each set represents a fluid variable (density, velocity, and pressure). Both the GA-PINN methodology and the base PINN model are considered (first two boxes) alongside HRSC numerical schemes [33] (third box).

Set of ICs	GA-PINN			Base-PINN			HRSC models		
	l^2_ρ	l^2_u	l^2_p	l^2_ρ	l^2_u	l^2_p	l^2_ρ	l^2_u	l^2_p
Problem 1	0.016	0.006	0.008	0.028	0.041	0.029	0.016	0.017	0.011
Problem 2	0.065	0.026	0.022	0.231	0.251	0.119	0.069	0.031	0.017
Problem 3	0.070	0.054	0.058	0.230	0.060	0.062	0.042	0.030	0.029
Problem 4	0.028	0.047	0.040	0.097	0.197	0.146	0.031	0.070	0.044

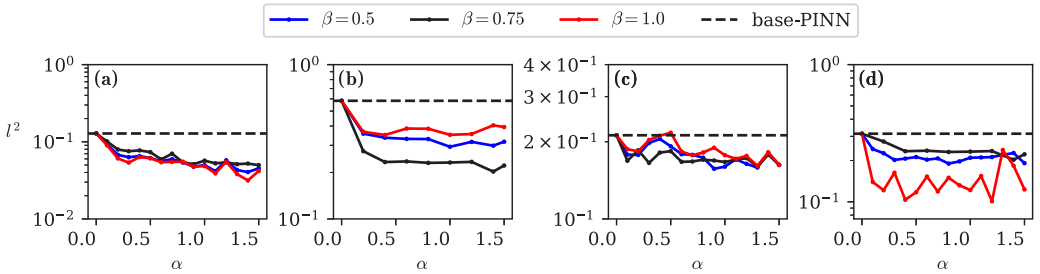

Fig. 4. Relationship between l^2 relative error variation in the GA-PINN model (solid lines) and changes in hyper-parameters α and β is studied for a uniform set of problems. The baseline PINN model is shown by the dashed black line. Error generally decreases with an increased gradient weight (α). Each plot data marker corresponds to the outcome of an independent 100,000-epoch training for Problems 1 to 4, corresponding to plots (**a**) to (**d**), respectively. The overall architecture and training specifications introduced in Section 4 remain consistent throughout.

of jets in active galactic nuclei). These equations are solved given some initial conditions, which are known a priori as data. Helping us from [42,43] the numerical experiments we consider are Riemann problems (i.e. initial value problems with discontinuous data) for which there is no need to specify spatial boundary conditions.

For all the numerical applications considered in this study the training and architecture specifications will remain consistent, encompassing both the GA-PINN methodology and the baseline models, with the exception, of course, of those parameters specific to the proposed methodology. Concerning the sampling of the (t, x) domain, we will employ $N_{\mathcal{R}} = 2^8 \times 2^8$ points, i.e., 2^8 for both t and x, in base 2, as Sobol sampling is utilized. Regarding the neural architecture and optimizer parameters, a fixed architecture of 6 internal layers, each comprising 100 neurons, will be employed. The optimization will be carried out using the *Adam* optimizer with a fixed learning rate of 10^{-5}, chosen to ensure smooth convergence. The weighting factors assigned to different loss functions will be set as $\omega_{IC} = 10^4$ and $\omega_{\mathcal{R}} = 1$ for initial conditions and PDEs, respectively. Activation functions will also be uniform across all problems, utilizing *tanh* for the internal layers of the PINN and fluid velocity at the output (to ensure velocity does not exceed the speed of light). For density and pressure, *softplus* will be utilized to ensure positivity of these variables. Henceforth, we shall designate this arrangement as the "general configuration" applicable to all scenarios presented through the paper. Any possible deviation from this standard will be explicitly highlighted when employed.

To assess the accuracy of our predictions, the L^2 norm or error will be used, which computes the squared disparities between the solution given by the constructed model and a "ground truth" solution. In our case, we consider the analytical solution of the Riemann problems as reference, as it is readily obtainable [43]. Furthermore, since we have more than one variable to predict and in general they can present quite different ranges of values, it is more consistent to use the relative L^2 norm and thus be able to make direct comparisons between them. The relative L^2 norm of an output variable $\boldsymbol{\mathcal{V}}_{\Theta,k}$ at a certain time t is then defined as

$$l^2_{\boldsymbol{\mathcal{V}}_k}(t) := \sqrt{\frac{\sum_n^N ||\boldsymbol{\mathcal{V}}_{\Theta,k}(t, x_n) - \boldsymbol{\mathcal{V}}_k(t, x_n)||_2}{\sum_n^N ||\boldsymbol{\mathcal{V}}_{\Theta,k}(t, x_n)||_2}}, \tag{25}$$

where subscript Θ corresponds to the solution obtained by the model, and the summations are done along the spatial dimension.

The following subsections present the specific results for each problem. The performance evaluation of the models is done using the l^2 relative error metric defined in Eq. (25). An overview of the results is reported in Table 1 which summarizes the values of the above metric for each of the scenarios. The errors reported correspond to the predicted density, velocity, and pressure variables. Notice that each box in the table shows the results obtained with a different approach, GA-PINN, Base-PINN, and HRSC schemes, allowing for a direct comparison of the methods. The relative errors of our GA-PINN (left column) are smaller than those of the baseline PINN method in general (reported in the middle column). Moreover, our method gives competitive results when compared with those from HRSC schemes (right column).

The method will be analyzed from different perspectives. We will propose different scenarios where the PINN model is going to take advantage of the gradient deactivation. Fig. 4 shows the variation of the l^2 error with the hyperparameter α, which is

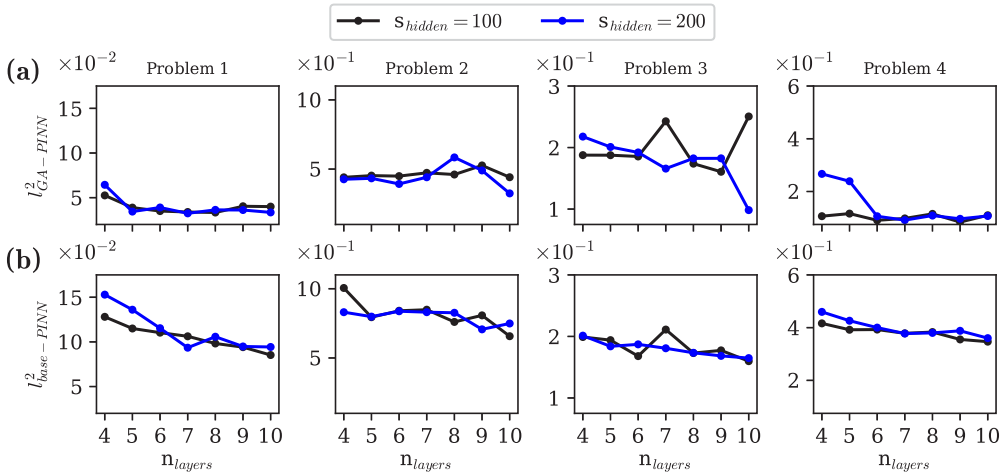

Fig. 5. Depiction of the l^2 given by the GA-PINN in (**a**) and by the baseline model in (**b**) for different number of hidden layers and size (number of neurons) per layer. This size remains constant through the internal architecture. Each column corresponds to the solution for one set of initial conditions.

the physical gradient weight, and with the exponent β[2]. This figure allows to evaluate the hypothesis proposed in Section 3.1 and motivating our study, i.e. that neural networks based on the PINN algorithm take advantage of gradient annihilation in most physical domains dominated by discontinuities to improve their resolution capabilities in such regions. The reduction in the error shown in Fig. 4 is evident when the physical gradient is weighted more heavily, with the error of a baseline PINN model ($\alpha = \beta = 0$) shown as a dashed line.

It is apparent that the resolution capabilities of the model improve in all scenarios. The potential of the gradient deactivation approach would depend on the number of real discontinuities that the proposed situation may present. Several tests have been carried out where no discontinuities were present, being the differences between the GA-PINN and a vanilla model not significant. Nevertheless, the problems considered through this manuscript present more than one in order to explicitly compare the performance of both models. Increasing β typically has a direct impact on the stability of the error as the importance of the gradient increases. Since the hyperparameter β corresponds to the exponent of the gradients, slight variations in their values can result in substantially different outcomes. Therefore, it is essential to exercise caution when selecting this parameter, with $\beta \in [0.0, 1.25]$ constituting a consistent choice.

Regarding the robustness of the method, questions may arise about how its performance would behave under a modification of the internal neural architecture. These considerations may include the amount of internal hidden layers as well as their size regarding the number of neurons. As introduced in the Methodology part (Section 3), the neural architecture behind the GA-PINN will be a simple fully-connected neural network. Therefore, it is possible to carry out a study about the performance that the model achieves when the depth and width of the network are modified. In Fig. 5 the measured l^2 is presented for the GA-PINN (**a**) and the baseline model (**b**), both for the different proposed problems. Each point in the figures corresponds to the final l^2 for the solution given by the method after a 75,000-epoch training, where the number of neurons (denoted as s_{hidden}) has been fixed for all the internal layers. The other hyperparameters such as the weights for the initial conditions are the ones specified as "general configuration", and α and β parameters have been fixed to 1.0 for all problems. The same scale for the l^2 has been considered for both methodologies for each scenario.

In general, both models demonstrate robustness and stability when modifications are considered to the number of layers and neurons in their respective architectures. It is noteworthy, however, that the baseline (vanilla) model tends to exhibit a modest declining trend in certain scenarios, such as Problems 1, 2, or 4, as the depth increases. Nevertheless, in these cases, the GA-PINN consistently achieves a lower error while employing a substantially smaller network. For instance, for the Problem 1, taking for granted that the error trend persists for the baseline model in these specific problems as the number of layers increases, we can see that approximately 15 hidden layers would be required to replicate the error level achieved by the GA-PINN with 5 or 6 layers. In certain cases, the GA-PINN may face challenges in achieving a proper resolution. For instance, Problem 3 reveals erratic behavior in the GA-PINN with increasing network depth, in contrast to the smoother conduct of the baseline model. However, despite this, the GA-PINN yields lower minimum l^2 value. To address this, specific adjustments may be necessary, such as increasing the number of training epochs, particularly in comparison to other scenarios, or considering a reduced learning rate as a potential solution.

[2] The parameter sets $\boldsymbol{\alpha} = \{\alpha_i\}_{i=1}^{L}$ and $\boldsymbol{\beta} = \{\beta_i\}_{i=1}^{L}$ have been condensed in the figure into two unique hyperparameters for the sake of simplicity The same will be used for all physical variables.

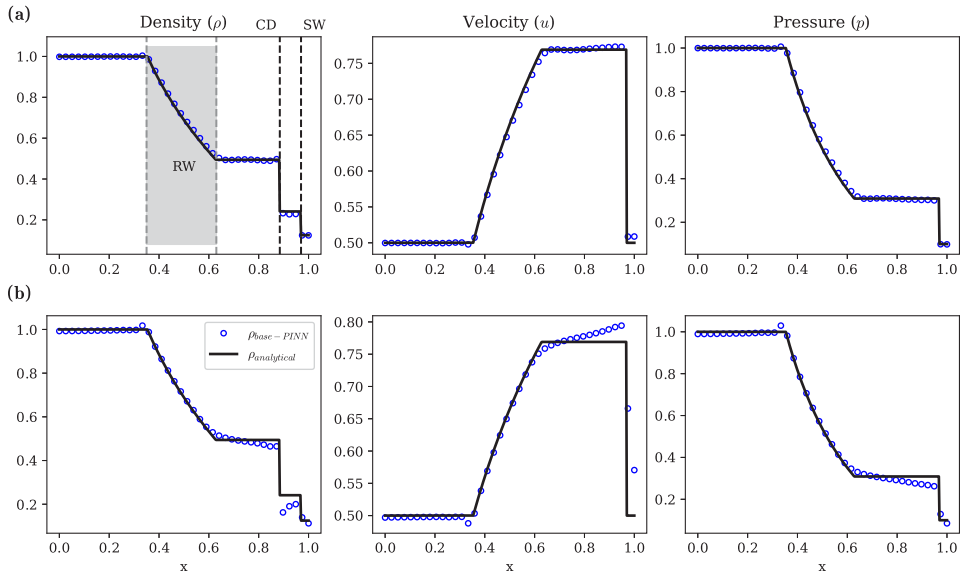

Fig. 6. Physical variables for *Sod Shock Tube* initial conditions (Eq. (26)) solved at $t = 0.5$ with $\Gamma = 5/3$. Analytical solution (solid black curve) serves as reference. GA-PINNs method in top row (**a**), base PINN model in bottom row (**b**). Predictions shown as blue circles. "SW", "CD", and "RW", indicate shock waves, contact discontinuities, and rarefactions, respectively, in the density plot in (**a**) for visual simplicity.

4.1. Problem 1: Sod Shock Tube

The first case we consider is a modification of a problem that has been extensively discussed in the literature on solving PDEs with PINNs: the *Sod Shock Tube* problem [22]. The initial data in the original problem involves a discontinuous reduction in density and pressure along the spatial dimension, while maintaining a constant zero velocity in the entire domain. However, in addressing our issue, we will take into consideration a non-negligible and non-zero velocity in comparison to c, justifying the utilization of the relativistic framework (Eqs. (10)). Our analysis employs an adiabatic index of $\Gamma = 5/3$. Considering the physical space $(t, x) \in [0, 0.5] \times [0, 1]$ and taking $x = 0.5$ as the point where the discontinuities are located at $t = 0$, the initial conditions for this problem read as Eq. (26).

$$\boldsymbol{\mathcal{U}}(0, x) := \boldsymbol{\mathcal{U}}_0(x) = \begin{cases} \rho_0(x) = 1.0 & \text{if } x \leq 0.5, \quad \rho_0(x) = 0.125 \quad \text{otherwise} \\ u_0(x) = 0.5 & \forall x \\ p_0(x) = 1.0 & \text{if } x \leq 0.5, \quad p_0(x) = 0.1 \quad \text{otherwise} \end{cases} \tag{26}$$

The evolution of this Riemann problem leads to the appearance of three elementary waves: a rarefaction wave (RW) on the left, a shock wave (SW) on the right and a contact discontinuity (CD) between the two through which the density and pressure remain constant. The solution at $t = 0.5$ is plotted in Fig. 6, which shows the results for the set of primitive variables (ρ, u, p). The shock wave, visible in all three variables, is located at $x = 0.97$, while the contact discontinuity, only visible in the density, can be found at $x = 0.88$. The three plots at the top row of the figure display the predictions of our method, while those at the bottom show the solution obtained with a baseline PINN model without additional modifications. For all three variables, the results for both methods are shown with blue open circles using as basis for the comparison the analytical solution of this Riemann problem, depicted with a solid black line in each plot. Furthermore, both training processes are identical in all common parameters, including the number of training epochs. As it can be seen, the solution given by the GA-PINN properly reproduces the correct evolution of the system and, in particular, sharply captures both the shock wave and the contact discontinuity. This is in contrast to the base PINN model, which does not yield optimal results. The base model may face challenges in accurately capturing the contact discontinuity in density (ρ) and the shock wave in the velocity (u) due to the extended spatial gradient and the tendency to produce smooth solutions. Furthermore, although the base model correctly localizes the position of the shock wave, it yields incorrect results for the constant state between the tail of the rarefaction wave and the shock visible both in the velocity and the pressure. As the top row of Fig. 6 shows, the GA-PINN mitigates these issues.

The function Λ used in this problem is that given by Eq. (22), setting the values of α and β to $\alpha_\rho = \alpha_u = \alpha_p = 1$ and $\beta_\rho = \beta_u = \beta_p = 1.25$, respectively. This selection of parameters yields results that are commensurate with those obtained via HRSC approaches. The choice of these hyperparameters is initially arbitrary, although it is possible to carry out a sweep and compute the variation in the performance of the models based on them. However, our choice is quite standard and not too aggressive, assuming a simple and direct variation of the base PINN model. Regarding the physical function employed as the loss ($\mathcal{L}$) in this scenario, it comprises the weight function Λ_2 (Eq. (22)) designed to suppress gradients and enhance the resolution of discontinuities. The

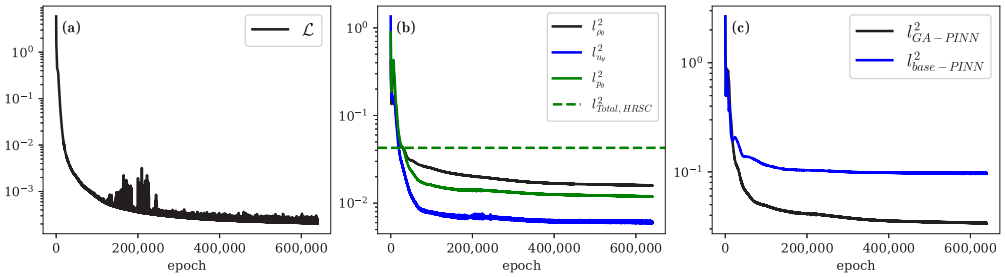

Fig. 7. Graphic representation of loss metrics during training for the *Sod Shock Tube* problem (Problem 1). (**a**) Total physical loss $\mathcal{L}$ as defined in Algorithm 1. (**b**) l^2 prediction errors in GA-PINN compared to the analytical solution, presented by independent variable, with the dashed green line indicating the error of the HRSC numerical method for reference. (**c**) Comparative illustration of the total l^2 error between the GA-PINN model and a baseline model. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

representation of this loss is illustrated in Fig. 7(**a**), while Fig. 7(**b**) displays the l^2 error used as a metric for comparison with the analytical solution, broken down by individual variables (ρ, u, and p). In total, it exhibits a value of $l^2_{\text{GA-PINN}} = 0.034$, comparable to an error of $l^2_{\text{HRSC}} = 0.043$ for the HRSC models [42] (depicted by the dashed green line in the graph). Finally, the aforementioned error obtained through our methodology and a base PINN model is presented in Fig. 7(**c**), demonstrating the variation of both throughout the training process and highlighting an earlier saturation for the base model. This comparison highlights that our proposed methodology precisely captures wave location and speed, comparable to HRSC methods.

4.2. Problem 2

As a second example we study a problem in which the density of the fluid remains initially constant throughout the spatial domain while the pressure has a discontinuity of an order of magnitude. In addition, half of the fluid has a certain constant speed at the initial instant while the other half is at rest. The spacetime domain considered for our solution is $(t, x) \in [0, 0.4] \times [0, 1]$ and the adiabatic index of the fluid is $\Gamma = 5/3$. The particular initial conditions are given in Eq. (27).

$$\mathcal{U}_0(x) = \begin{cases} \rho_0(x) = 1.0 & \forall x \\ u_0(x) = 0.5 & \text{if } x \leq 0.5, \quad u_0(x) = 0.0 \quad \text{otherwise} \\ p_0(x) = 0.1 & \text{if } x \leq 0.5, \quad p_0(x) = 1.0 \quad \text{otherwise} \end{cases} \tag{27}$$

The solution for this problem at $t = 0.4$ is displayed in Fig. 8. The evolution of the Riemann problem leads to a shock wave on the left and a rarefaction wave on the right, with a contact discontinuity in between, visible in the density. The shock wave is located at $x = 0.35$ and the contact discontinuity at 0.48. The solution obtained with our method is shown in the top row of the figure while the prediction of a baseline PINN model is depicted in the bottom row. Again, both models have the same training properties for the common parameters reported at the beginning of Section 4, except for those strictly defined in the proposed approach. Similar to the *Sod Shock Tube* problem, the function Λ employed to appropriately weigh the gradients of variables is computed following Eq. (22). Similarly, the set of hyperparameters has been chosen as $\alpha_\rho = \alpha_u = \alpha_p = 1.0$ and $\beta_\rho = \beta_u = \beta_p = 1.25$.

Fig. 8 suggests that the GA-PINN method exhibits improvements compared to the baseline model. However, certain features suggest that the solution for the GA-PINN method exhibits room for improvement, notably in relation to the density value along the constant state between the shock wave and the contact discontinuity. To increase the performance of the neural model, the activation function *softplus*[3] has been introduced for the density and pressure to better describe the contact discontinuity by achieving a less smooth behavior. The challenges faced by the network are evident in the errors reported in Table 1, where the overall relative error of our methodology is $l^2_{\text{GA-PINN}} = 0.113$, compared to an error of $l^2_{\text{HRSC}} = 0.117$ for the HRSC model therefore both methodologies being at a similar level of performance. Moreover, the improvement of our methodology with respect to the PINN base model is substantial, the latter attaining an associated error of $l^2_{\text{base-PINN}} = 0.601$. This difference in effectiveness is illustrated in Fig. 8, particularly concerning the density variable. The incorporation of a weight that mitigates network responses in regions of discontinuity yields discernible improvements in reconstructing the shock wave and the contact discontinuity.

In Fig. 9(**a**), the evolution of the physical loss throughout the training process is observed, predominantly exhibiting a smooth trend, which is desirable. In Fig. 9(**b**), the l^2 error is presented, separated by variable, in comparison to the error obtained by the HRSC method. It is evident that the density variable exhibits higher values compared to the other two variables in Fig. 9(**c**), a direct consequence of the challenge that the network faces in resolving it due to its more extreme nature. Finally, both quadratic errors are depicted for the GA-PINN and a basic network. A noticeable improvement is observed for the latter, with an anomalous noisy behavior noted during the final stages of training. This phenomenon may result from a sudden divergence of the network from the minimum it was in, suggesting that it was neither global nor stable.

[3] The *softplus* is a smooth approximation to the well-known *ReLU* function which helps the gradient to propagate backwards better because its derivative is a *sigmoid* function and is therefore close to 1 as the input increases. This is a mathematically appropriate property that helps ensure that the gradient in the successive derivatives through the neural structure does not vanish and may be desirable in some cases.

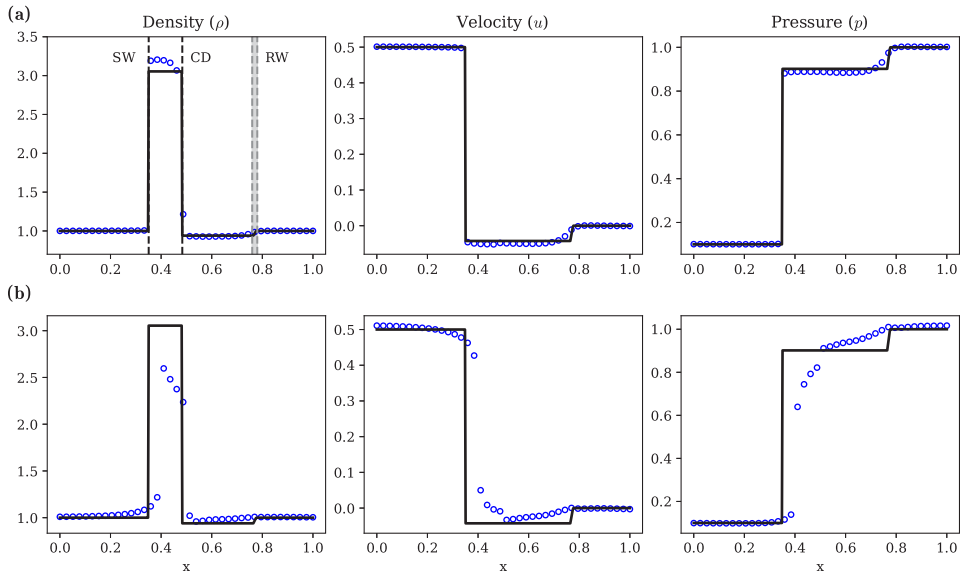

Fig. 8. Physical variables extracted for the initial conditions system defined in Eq. (27) at $t = 0.4$ with $\Gamma = 5/3$. The analytical solution is represented by a solid black line, and predictions are shown as blue circles. The GA-PINN solution is displayed in (**a**), while the baseline PINN model solution is in (**b**). Shock waves, contact discontinuities and rarefraction waves are highlighted with "SW", "CD" and "RW", respectively.

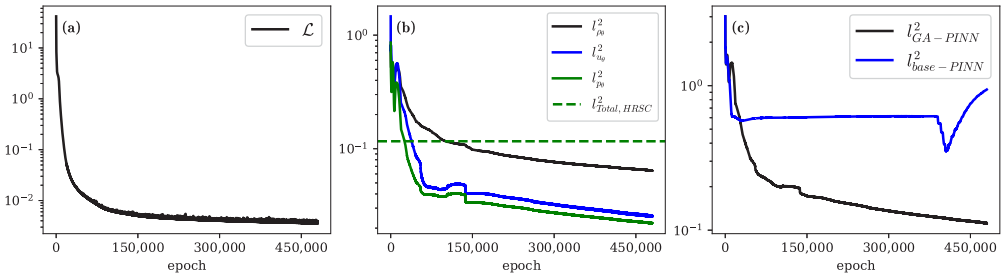

Fig. 9. Visual representation of the loss functions during training for the initial conditions described in Eq. (27). (**a**) Total physical loss, $\mathcal{L}$, Eq. (24). (**b**) l^2 errors of GA-PINN predictions with respect to the analytical solution and numerically obtained HRSC model (dashed green curve). (**c**) Comparison between the learning processes for GA-PINN and a baseline model using l^2. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

4.3. Problem 3

In the third analyzed scenario, the density and velocity exhibit a jump at $x = 0.5$ while the pressure remains constant. This leads to a complete deceleration of the fluid at the critical point $x = 0.5$, $t = 0$. From left to right at the initial time the density is reduced by half, while the pressure does not change. The physical domain is defined as $(t, x) \in [0, 0.4] \times [0, 1]$, and the adiabatic index of the fluid is $\Gamma = 4/3$. The initial conditions for this Riemann problem are summarized in Eq. (28).

$$\boldsymbol{\mathcal{U}}_0(x) = \begin{cases} \rho_0(x) = 1.0 & \text{if } x \leq 0.5, \quad \rho_0(x) = 0.5 \quad \text{otherwise} \\ u_0(x) = 0.5 & \text{if } x \leq 0.5, \quad u_0(x) = 0.0 \quad \text{otherwise} \\ p_0(x) = 1.0 & \text{if } x \leq 0.5, \quad p_0(x) = 1.0 \quad \text{otherwise} \end{cases} \tag{28}$$

Fig. 10 shows the solution at the final time $t = 0.4$. The temporal evolution of the proposed scenario gives rise to two different shock waves, discernible at $x = 0.43$ and $x = 0.61$, with the evident discontinuity common to all variables. Between them, once again, there is an exclusive density contact discontinuity at $x = 0.76$. In alignment with the preceding cases, the function described in Eq. (22) is employed to nullify the gradients of the variables within the loss metric. The hyperparameters are fixed to $\alpha_\rho = \alpha_u = \alpha_p = 1.0$ and also $\beta_\rho = \beta_u = \beta_p = 1.0$, reflecting a non-aggressive modification of the vanilla methodology. Other parameters, including the relative weight of initial conditions and specifications of the neural architecture, are held constant using the general configuration outlined in this section.

Fig. 11 illustrates the behavior of the physical losses ($\mathcal{L}$) and the relative l^2 errors during the training process in a similar way as it has been done in the previous cases. The physical losses in (**a**) exhibit a monotonically decreasing trend; however, a degree

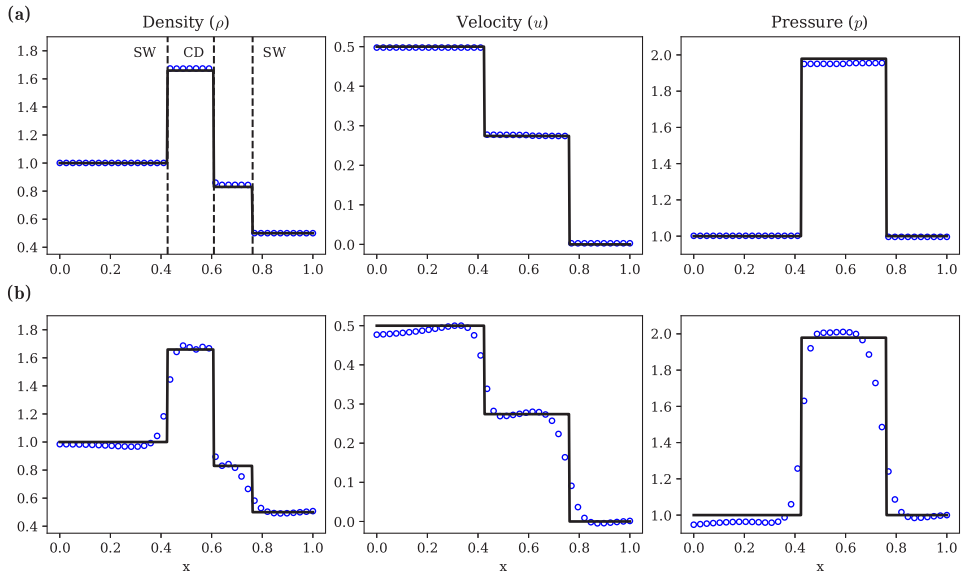

Fig. 10. Solution for the initial conditions presented in Eq. (28) at $t = 0.4$ with $\Gamma = 4/3$. The analytical solution of reference (black solid line) and the models' predictions (blue circles) are depicted for the GA-PINN (**a**) and baseline (**b**) methodologies. Shock waves and contact discontinuity are highlighted using "SW" and "CD" respectively.

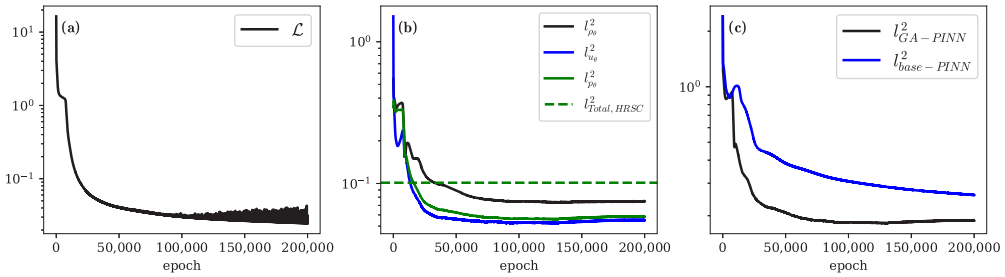

Fig. 11. Evolution of the different metrics during training. (**a**) Physical loss used to train the GA-PINN, $\mathcal{L}$. (**b**) Evolution of the l^2 measurements for the different variables. (**c**) Measured l^2 for the considered methodologies.

of noisy behavior is observed in contrast to previous scenarios. This phenomenon does not translate into adverse behavior in the l^2 norm, as illustrated in (**b**). The system seems to converge around 250,000 epochs, with marginal performance enhancements thereafter, primarily concentrated on fine-tuning the plateau regions of the density and pressure variables. In this scenario, the GA-PINN consistently yields better results compared to the vanilla methodology, with respective errors of $l^2_{\text{GA-PINN}} = 0.182$ and $l^2_{\text{base-PINN}} = 0.353$. Conversely, the HRSC methodology achieves a lower error of $l^2_{\text{HRSC}} = 0.101$, indicating a generally higher resolution. Nevertheless, upon inspecting the final variable profiles in Fig. 10, it becomes evident that the primary disparities between the GA-PINN resolution and the ground truth reference arise from little differences in the plateau zones between discontinuities in density and pressure, particularly in the latter. Notably, the discontinuities are properly resolved, aligning with the primary objective. The observed differences in flat zones may potentially be rectified by extending the training duration of the PINN. Additionally, Fig. 11(**c**) suggests that the l^2 norm of the base-PINN model may outperform the GA-PINN model at certain points due to a visible downward trend. However, the improved performance of the vanilla model is primarily attributed to better resolution in flat areas across different epochs and not in discontinuities. Despite these minor disparities, it is noteworthy that the solution provided by the GA-PINN tends to outperform the one predicted by the vanilla approach.

4.4. Problem 4: Reflection test

The last problem we propose concerns the reflection of a relativistic fluid. For this test the initial density and pressure fields are constant over the entire spatial domain, while the velocity field undergoes a sign change with no change in magnitude. At $t = 0$ the opposite-moving fluids hit against each other at $x = 0.5$. This collision produces two shock waves moving toward $x < 0.5$ and $x > 0.5$, respectively, leaving behind a constant state of shocked, heated fluid at rest. The physical domain for this problem is

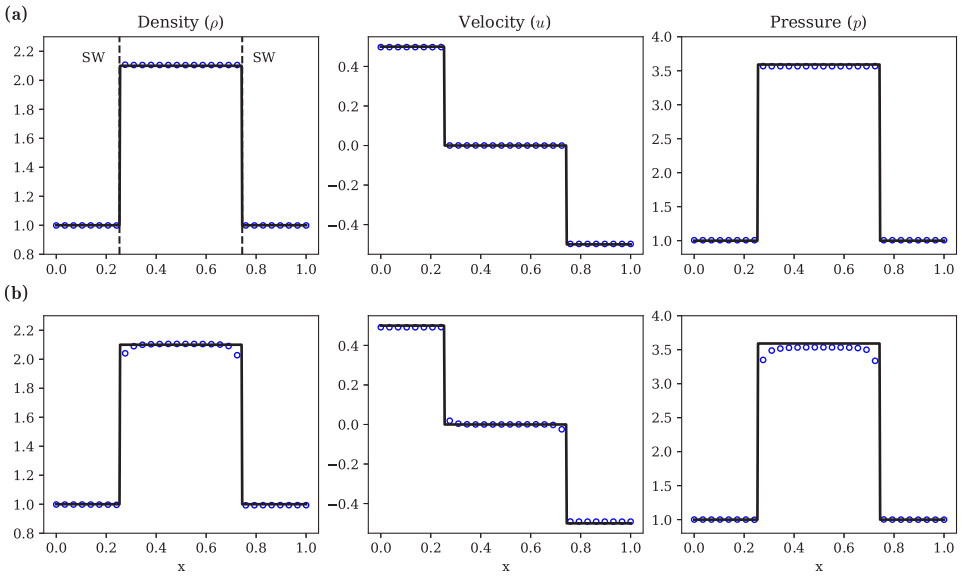

Fig. 12. Solution for the system of initial conditions in Eq. (29) at $t = 0.4$ with an adiabatic index of $\Gamma = 5/3$. The analytical solution for reference is shown in black while the respective predictions for the GA-PINN (**a**) and a vanilla model (**b**) are depicted using blue circles. "SW" in the first density plot stands for "shock wave", being these common for all variables.

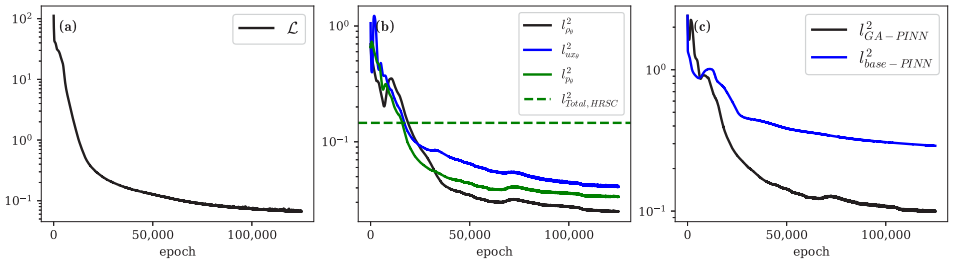

Fig. 13. Depiction of the metrics measured during the learning process for the set of initial conditions in Eq. (29). (**a**) Total physical loss. (**b**) l^2 errors for the different variables in comparison with the HRSC performance. (**c**) Comparison of the different l^2 measurements achieved by the two proposed DL-based models.

$(t, x) \in [0, 0.4] \times [0, 1]$ and the adiabatic exponent of the equation of state will be $\Gamma = 5/3$. The initial conditions are presented in Eq. (29).

$$\mathcal{U}_0(x) = \begin{cases} \rho_0(x) = 1.0 \quad \forall x \\ u_0(x) = 0.5 \quad \text{if } x \leq 0.5, \quad u_0(x) = -0.5 \quad \text{otherwise} \\ p_0(x) = 1.0 \quad \forall x\,. \end{cases} \tag{29}$$

The final variables are depicted in Fig. 12. The two shock waves are located at $x = 0.25$ and $x = 0.75$, and are common for all variables. To solve this problem the global configuration introduced at the beginning of the numerical results section has been employed. The Λ described in Eq. (22) has been used with all α and β hyperparameters set equal to 1, representing a modification that is not overly aggressive. Both neural models capture properly the propagation of the shock waves and the associated jumps. However, a notable difference between our method and a base model is the limited ability of the latter to accurately resolve the corners around discontinuities. This deficiency can lead to an accumulation of errors in those regions, which can degrade the overall performance score, even if the discontinuities themselves are adequately captured. Indeed, the computed overall relative errors are $l^2_{\text{GA-PINN}} = 0.146$ and $l^2_{\text{base-PINN}} = 0.440$, which shows that the proposed methodology tends to achieve better performance than the standard model. Moreover, the GA-PINN model also demonstrates competitiveness with high-resolution numerical models in the literature ($l^2_{\text{HRSC}} = 0.145$), establishing itself as a viable contender in this context.

In a manner similar to preceding scenarios, Fig. 13 illustrates various losses derived from the training process. The physical loss employed for training is depicted in (**a**). It is noteworthy that this metric exhibits a smooth decrease, indicating a gradual and continuous learning process, achieving rapid convergence at approximately 125,000 epochs. This contrasts with instances where prolonged training periods have been necessary. In (**b**), a comparative analysis of the relative mean squared error associated with each variable is presented. It is observed that all three variables exhibit a smooth convergence and remain significantly below the

numerical error obtained by HRSC models. Finally, in (**c**), a simple comparison of the l^2 error between the GA-PINN and the vanilla comparison model is presented. At approximately 20,000 epochs, the proposed model tends to demonstrate improved resolution performance, signifying a substantial improvement in the fidelity of discontinuity resolution. This phenomenon is evident in the proposed final solutions, as depicted in Fig. 12.

5. Discussion and conclusions

In this paper a methodology based on Physics-Informed Neural Networks (PINNs), denoted as Gradient-Annihilated PINNs (GA-PINNs), has been introduced. The primary motivation driving this approach lies in achieving accurate resolution of discontinuities in the temporal evolution of physical scenarios employing a model based on deep learning. The performance of the GA-PINN model has been tested by solving Riemann problems in special relativistic hydrodynamics, a particularly challenging physical system. Simulating these scenarios is demanding, requiring the use of advanced numerical resolution methods such as High-Resolution Central Schemes (HRSC) [33], which leverage the conservative form of the equations. The introduction of methodologies like PINNs in the literature opens new avenues for numerical problem-solving. However, vanilla methodologies may fall short in handling discontinuities in system evolution, encouraging researchers to devise modifications or novel ideas. The primary motivation behind this approach has been the need to address discontinuities in Riemann problems within the field of relativistic hydrodynamics.

In practice, it is feasible to have a general understanding of fluid behavior under specific initial conditions, allowing us to anticipate phenomena like shock waves. However, accurately predicting the exact locations of discontinuities at a later time is challenging and often impossible. While existing methodologies show success in classical fluid dynamics scenarios, such as those based on the Euler equations [21], applying them here is hindered by the inability to pre-determine discontinuity locations. Consequently, manual sampling around these discontinuities for additional data is not feasible. In addition, several studies, including the RAR approach [36], focus on optimizing the sampling of the physical domain for improved model performance. However, determining factors such as the frequency and quantity of added collocation points, the choice of a probability distribution, among other considerations, become crucial. Additionally, assuming that areas with higher physical loss correspond to discontinuities in each learning step may not universally hold true, particularly in situations with extreme conditions in the relativistic regime, where access to an analytical exact solution is often unavailable.

The proposed method establishes a link between the challenge neural networks face when dealing with regions of steep gradients and the improbability of directly extending the universal approximation theorem [18–20] without modifying the neural model to improve its performance at discontinuities. Although PINNs have produced remarkable results in solving a range of physical problems, researchers agree that this approach often over-smooths physical variables, resulting in a limited ability to account for sudden changes in the variables. It is nevertheless possible to adapt them to our needs.

In particular, the GA-PINN methodology introduced in this paper focuses on a key aspect. The discontinuities in physical variables represent abrupt changes within a finite discrete spatiotemporal area. Sudden changes in variables, often irreversible due to the increase of the fluid entropy, pose a challenge for neural network methodologies. Extending training time indefinitely to perfectly solve these discontinuities may lead to an overly smoothed solution. Ideally, the methodology should robustly address the discontinuity by providing effective solutions for points immediately before and after the critical spatiotemporal location, effectively resolving the jump by treating it as if it were not present.

From a technical point of view, this aspect involves introducing a weight function which primarily depends on the gradients of the physical variables of interest and will be inversely proportional to them. This weight function enables the network to reduce its attention to regions with significant gradients, especially those with discontinuities. Our hypothesis is that the network enhances its performance in resolving these regions by selectively limiting learning in these areas. The numerical results have been presented as relative l^2 errors in Fig. 4 of Section 4.

Overall, the obtained results have shown that the GA-PINN model correctly describes the propagation speeds of the discontinuities that appear when solving the special relativistic hydrodynamics equations and sharply captures the associated jumps. In all Riemann problems investigated, the accuracy reached by our GA-PINN model has proven comparable to that obtained with the central, second-order, shock-capturing scheme of [42] and it is consistently higher than the accuracy achieved by a baseline, out-of-the-box PINN algorithm. Moreover, our approach entirely sidesteps one of the main drawbacks of grid-based solutions of the relativistic hydrodynamics equations, the so-called “primitive variable recovery”. In the case of relativistic hydrodynamics, either special or general, the determination of the primitive variables from the conserved variables (i.e. the state vector of the PDE system) is usually not in closed form and must be performed through a root-finding procedure [25,28]. The success of such an approach is not always guaranteed, particularly for highly relativistic flows or when dealing with microphysical (tabulated) equations of state describing neutron star matter. The possibility of avoiding altogether the necessity of having to implement the recovery of the primitive variables highlights an important benefit of our GA-PINN model. The fundamental reason behind this capability lies in the ability of neural networks to concurrently handle both sets of variables in the internal computations they execute.

One situation that remains to be fully exploited with the GA-PINN approach involves ultra-relativistic flows, i.e. fluids with $W \sim 10$ and higher. HRSC schemes written in conservation form have long proven to be well suited in dealing with such ultra-relativistic flows (see e.g. [27,28] and references therein). Such an extreme situation, however, leads to greater complexity in learning through a neural model. Minor deficiencies encountered when learning these ultra-high velocities could lead to significantly greater problems. To solve this subset of problems in relativistic hydrodynamics, it might be advisable to modify the neural architecture by considering a specialized neural subnetwork for each physical variable, rather than a single one whose weights are used for solving all variables. Therefore, by having a separate set of trainable variables for each physical output, the network

could better learn these extreme velocities and lead to a better solution. Another option to consider would be to guide the network in gradually learning the solution for each time step, aligning with the principle of "respecting causality" proposed by [44]. This enables the network to learn each time step effectively, minimizing deviations in the initial conditions. Despite potential numerical challenges in ultra-relativistic regimes, combining this approach with other physical biases may be worthwhile.

The performance of neural networks when handling discontinuities seems to benefit from various forms of suppression of learning, as our experiments have shown. However, the experimental nature of our results and their variability depending on the initial conditions and choice of hyperparameters, clearly indicate that further investigations concerning the proposed methodology are still possible. Nevertheless, the method reported in this paper already provides a solid foundation for future work in this area and constitutes a valuable tool to model relativistic flows in fields of physics characterized by the prevalence of discontinuous solutions, such as astrophysics and particle physics.

CRediT authorship contribution statement

Antonio Ferrer-Sánchez: Conceptualization, Data curation, Investigation, Methodology, Software, Visualization, Writing – original draft, Writing – review & editing. **José D. Martín-Guerrero:** Funding acquisition, Project administration, Resources, Supervision. **Roberto Ruiz de Austri-Bazan:** Conceptualization, Investigation, Methodology, Project administration, Supervision, Resources. **Alejandro Torres-Forné:** Conceptualization, Funding acquisition, Project administration, Resources. **José A. Font:** Conceptualization, Formal analysis, Investigation, Project administration, Supervision, Validation, Visualization.

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.

Data availability

Necessary data and code in order to replicate possible experiments is given as a GitHub repository with a configuration file and a guide.

Acknowledgments

We thank José María Martí for kindly providing the code to compute the exact solution of the Riemann problem for special relativistic hydrodynamics. The authors gratefully acknowledge the computer resources at Artemisa, funded by the European Union ERDF and Comunitat Valenciana as well as the technical support provided by the Instituto de Fisica Corpuscular, IFIC (CSIC-UV). R. RdA is supported by PID2020-113644GB-I00 from the Spanish Ministerio de Ciencia e Innovación. ATF and JAF are supported by the Spanish Agencia Estatal de Investigación (grant PID2021-125485NB-C21 funded by MCIN/AEI/10.13039/501100011033 and ERDF A way of making Europe). AFS and JDMG are partially supported by the agreement funded by the European Union, between the Valencian Ministry of Innovation, Universities, Science and Digital Society, and the network of research centers in Artificial Intelligence (Valencian Foundation valgrAI). It has also been funded by the Valencian Government grant with reference number CIAICO/2021/184; the Spanish Ministry of Economic Affairs and Digital Transformation through the QUANTUM ENIA project call – Quantum Spain project, and the European Union through the Recovery, Transformation and Resilience Plan – NextGenerationEU within the framework of the Digital Spain 2025 Agenda.

References

[1] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, M. Chen, Hierarchical text-conditional image generation with CLIP latents, 2022, http://dx.doi.org/10.48550/ARXIV.2204.06125.

[2] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, B. Ommer, High-resolution image synthesis with latent diffusion models, 2021, http://dx.doi.org/10.48550/ARXIV.2112.10752.

[3] T. Brown, et al., Language models are few-shot learners, in: H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, H. Lin (Eds.), Advances in Neural Information Processing Systems, Vol. 33, Curran Associates, Inc., 2020, pp. 1877–1901.

[4] H. Wang, H. Wu, Z. He, L. Huang, K.W. Church, Progress in machine translation, Engineering 18 (2022) 143–153, http://dx.doi.org/10.1016/j.eng.2021.03.023.

[5] J.G. De Gooijer, R.J. Hyndman, 25 Years of time series forecasting, Int. J. Forecast. 22 (3) (2006) 443–473, http://dx.doi.org/10.1016/j.ijforecast.2006.01.001, Twenty five years of forecasting.

[6] R.M. Schmidt, Recurrent neural networks (RNNs): A gentle introduction and overview, 2019, http://dx.doi.org/10.48550/ARXIV.1912.05911.

[7] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780, http://dx.doi.org/10.1162/neco.1997.9.8.1735.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems, Vol. 30, Curran Associates, Inc., 2017.

[9] K. Aitken, V.V. Ramasesh, Y. Cao, N. Maheswaranathan, Understanding how encoder-decoder architectures attend, 2021, http://dx.doi.org/10.48550/ARXIV.2110.15253.

[10] R. Jozefowicz, O. Vinyals, M. Schuster, N. Shazeer, Y. Wu, Exploring the limits of language modeling, 2016, http://dx.doi.org/10.48550/ARXIV.1602.02410.

[11] M. Raissi, P. Perdikaris, G. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. 378 (2019) 686–707, http://dx.doi.org/10.1016/j.jcp.2018.10.045, URL https://www.sciencedirect.com/science/article/pii/S0021999118307125.
[12] M. Raissi, P. Perdikaris, G.E. Karniadakis, Physics informed deep learning (part I): Data-driven solutions of nonlinear partial differential equations, 2017, http://dx.doi.org/10.48550/ARXIV.1711.10561.
[13] M. Raissi, P. Perdikaris, G.E. Karniadakis, Physics informed deep learning (part II): Data-driven discovery of nonlinear partial differential equations, 2017, http://dx.doi.org/10.48550/ARXIV.1711.10566.
[14] Z.K. Lawal, H. Yassin, D.T.C. Lai, A. Che Idris, Physics-informed neural network (PINN) evolution and beyond: A systematic literature review and bibliometric analysis, Big Data Cogn. Comput. 6 (4) (2022) http://dx.doi.org/10.3390/bdcc6040140.
[15] S. Cuomo, V.S. Di Cola, F. Giampaolo, G. Rozza, M. Raissi, F. Piccialli, Scientific machine learning through physics–Informed neural networks: Where we are and what's next, J. Sci. Comput. 92 (88) (2022) http://dx.doi.org/10.1007/s10915-022-01939-z.
[16] C. Michoski, M. Milosavljević, T. Oliver, D.R. Hatch, Solving differential equations using deep neural networks, Neurocomputing 399 (2020) 193–212, http://dx.doi.org/10.1016/j.neucom.2020.02.015.
[17] R.G. Patel, I. Manickam, N.A. Trask, M.A. Wood, M. Lee, I. Tomas, E.C. Cyr, Thermodynamically consistent physics-informed neural networks for hyperbolic systems, J. Comput. Phys. 449 (2022) 110754, http://dx.doi.org/10.1016/j.jcp.2021.110754.
[18] A. Pinkus, Approximation theory of the MLP model in neural networks, Acta Numer. 8 (1999) 143–195, http://dx.doi.org/10.1017/S0962492900002919.
[19] G.V. Cybenko, Approximation by superpositions of a sigmoidal function, Math. Control Signals Systems 2 (1989) 303–314.
[20] A. Heinecke, J. Ho, W.-L. Hwang, Refinement and universal approximation via sparsely connected reLU convolution nets, IEEE Signal Process. Lett. 27 (2020) 1175–1179, http://dx.doi.org/10.1109/LSP.2020.3005051.
[21] Z. Mao, A.D. Jagtap, G.E. Karniadakis, Physics-informed neural networks for high-speed flows, Comput. Methods Appl. Mech. Engrg. 360 (2020) 112789, http://dx.doi.org/10.1016/j.cma.2019.112789.
[22] G.A. Sod, A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws, J. Comput. Phys. 27 (1) (1978) 1–31, http://dx.doi.org/10.1016/0021-9991(78)90023-2.
[23] A. Papados, Solving hydrodynamic shock-tube problems using weighted physics-informed neural networks with domain extension, 2021, http://dx.doi.org/10.13140/RG.2.2.29724.00642/1.
[24] R. Mojgani, M. Balajewicz, P. Hassanzadeh, Kolmogorov n–width and Lagrangian physics-informed neural networks: A causality-conforming manifold for convection-dominated PDEs, Comput. Methods Appl. Mech. Engrg. 404 (2023) 115810, http://dx.doi.org/10.1016/j.cma.2022.115810.
[25] J.A. Font, J.M. Ibanez, A. Marquina, J.M. Marti, Multidimensional relativistic hydrodynamics: Characteristic fields and modern high-resolution shock-capturing schemes, Astron. Astrophys. 282 (1) (1994) 304–314.
[26] F. Banyuls, J.A. Font, J.M. Ibáñez, J.M. Martí, J.A. Miralles, Numerical {3 + 1} general relativistic hydrodynamics: A local characteristic approach, Astrophys. J. 476 (1) (1997) 221, http://dx.doi.org/10.1086/303604.
[27] J.M. Martí, E. Müller, Numerical hydrodynamics in special relativity, Living Rev. Relativ. 6 (7) (2003) http://dx.doi.org/10.12942/lrr-2003-7.
[28] J.A. Font, Numerical hydrodynamics and magnetohydrodynamics in general relativity, Living Rev. Relativ. 11 (1) (2008) 7, http://dx.doi.org/10.12942/lrr-2008-7.
[29] A.G. Baydin, B.A. Pearlmutter, A.A. Radul, J.M. Siskind, Automatic differentiation in machine learning: a survey, 2015, http://dx.doi.org/10.48550/ARXIV.1502.05767.
[30] L. Lu, R. Pestourie, W. Yao, Z. Wang, F. Verdugo, S.G. Johnson, Physics-informed neural networks with hard constraints for inverse design, SIAM J. Sci. Comput. 43 (6) (2021) B1105–B1132, http://dx.doi.org/10.1137/21M1397908.
[31] R.D. Zucker, O. Biblarz, Fundamentals of Gas Dynamics, John Wiley & Sons, 2002.
[32] J.A. Font, An introduction to relativistic hydrodynamics, J. Phys. Conf. Ser. 91 (1) (2007) 012002, http://dx.doi.org/10.1088/1742-6596/91/1/012002.
[33] E.F. Toro, Riemann Solvers and Numerical Methods for Fluid Dynamics, Springer Verlag, 1997.
[34] S. Wang, Y. Teng, P. Perdikaris, Understanding and mitigating gradient flow pathologies in physics-informed neural networks, SIAM J. Sci. Comput. 43 (5) (2021) A3055–A3081, http://dx.doi.org/10.1137/20M1318043.
[35] Z. Xiang, W. Peng, X. Liu, W. Yao, Self-adaptive loss balanced physics-informed neural networks, Neurocomputing 496 (2022) 11–34, http://dx.doi.org/10.1016/j.neucom.2022.05.015.
[36] C. Wu, M. Zhu, Q. Tan, Y. Kartha, L. Lu, A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks, Comput. Methods Appl. Mech. Engrg. 403 (2023) 115671, http://dx.doi.org/10.1016/j.cma.2022.115671.
[37] A.D. Jagtap, K. Kawaguchi, G.E. Karniadakis, Adaptive activation functions accelerate convergence in deep and physics-informed neural networks, J. Comput. Phys. 404 (2020) 109136, http://dx.doi.org/10.1016/j.jcp.2019.109136.
[38] L. Liu, S. Liu, H. Xie, F. Xiong, T. Yu, M. Xiao, L. Liu, H. Yong1, Discontinuity computing with physics-informed neural network, 2022, http://dx.doi.org/10.36227/techrxiv.19391279.
[39] I. Sobol', On the distribution of points in a cube and the approximate evaluation of integrals, USSR Comput. Math. Math. Phys. 7 (4) (1967) 86–112, http://dx.doi.org/10.1016/0041-5553(67)90144-9.
[40] W.-L. Loh, Ann. Statist. 24 (5) (1996) 2058–2080.
[41] D. McKay Michael, J. Beckman Richard, J. Conover William, Technometrics 42 (1) (2000) 55–61.
[42] A. Lucas-Serrano, J.A. Font, J.M. Ibáñez, J.M. Martí, Assessment of a high-resolution central scheme for the solution of the relativistic hydrodynamics equations, Astron. Astrophys. 428 (2004) 703–715, http://dx.doi.org/10.1051/0004-6361:20035731, arXiv:astro-ph/0407541.
[43] J.M. Martí, E. Müller, The analytical solution of the Riemann problem in relativistic hydrodynamics, J. Fluid Mech. 258 (1994) 317–333, http://dx.doi.org/10.1017/S0022112094003344.
[44] S. Wang, S. Sankaran, P. Perdikaris, Respecting causality is all you need for training physics-informed neural networks, 2022, arXiv:2203.07404.