axioms

MDPI

Article

A Physics-Informed Neural Network Based on the Boltzmann Equation with Multiple-Relaxation-Time Collision Operators

Zhixiang Liu [1], Chenkai Zhang [1], Wenhao Zhu [2,*] and Dongmei Huang [3]

[1] College of Information Technology, Shanghai Ocean University, Shanghai 201306, China; zxliu@shou.edu.cn (Z.L.); m220901563@st.shou.edu.cn (C.Z.)
[2] School of Computer Engineering and Science, Shanghai University, Shanghai 200444, China
[3] College of Electronic and Information Engineering, Shanghai University of Electric Power, Shanghai 201306, China; dmhuang@shiep.edu.cn
* Correspondence: whzhu@shu.edu.cn

Abstract: The Boltzmann equation with multiple-relaxation-time (MRT) collision operators has been widely employed in kinetic theory to describe the behavior of gases and liquids at the macro-level. Given the successful development of deep learning and the availability of data analytic tools, it is a feasible idea to try to solve the Boltzmann-MRT equation using a neural network-based method. Based on the canonical polyadic decomposition, a new physics-informed neural network describing the Boltzmann-MRT equation, named the network for MRT collision (NMRT), is proposed in this paper for solving the Boltzmann-MRT equation. The method of tensor decomposition in the Boltzmann-MRT equation is utilized to combine the collision matrix with discrete distribution functions within the moment space. Multiscale modeling is adopted to accelerate the convergence of high frequencies for the equations. The micro–macro decomposition method is applied to improve learning efficiency. The problem-dependent loss function is proposed to balance the weight of the function for different conditions at different velocities. These strategies will greatly improve the accuracy of the network. The numerical experiments are tested, including the advection–diffusion problem and the wave propagation problem. The results of the numerical simulation show that the network-based method can obtain a measure of accuracy at $O(10^{-3})$.

Keywords: deep neural networks; Boltzmann equation; multiple-relaxation-time model; canonical polyadic decomposition

MSC: 68T07

check for updates

Citation: Liu, Z.; Zhang, C.; Zhu, W.; Huang, D. A Physics-Informed Neural Network Based on the Boltzmann Equation with Multiple-Relaxation-Time Collision Operators. *Axioms* **2024**, *13*, 588. https://doi.org/10.3390/axioms13090588

Academic Editors: Branislav Popovic, Marko Janev and Lazar Kopanja

Received: 12 July 2024
Revised: 26 August 2024
Accepted: 27 August 2024
Published: 29 August 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Progressive artificial intelligence techniques provide an alternative route to solving problems that have been challenged by difficulties in the past, such as computer vision [1] and natural language processing [2]. Deep learning has offered an appropriate way for scientists to interact with digital data in the industrial fields and established a foundation for artificial general intelligence [3]. Deep learning is based on neural network algorithms and can process unstructured data. A neural network has layers of nodes containing an input layer, one or more hidden layers, and an output layer. An artificial neuron or node is connected to another one. A neural network relies on training data to learn and improve its predictions by selecting representative features. The effectiveness of deep learning depends on the size of the training set. It is difficult to train an effective neural network using small data sets [4]. In fluid dynamical and astronautical research, conducting experiments in the test laboratory is costly [5]. The establishment of a full-scale database is widely challenged, which influences the application of neural networks for problems in fluid dynamical and astronautical research [6].

Physical laws can be expressed in the form of partial differential equations (PDEs). Physical nature is studied according to partial differential equations, along with initial and boundary conditions. It is difficult to obtain analytic solutions to partial differential equations, and they are usually approximated in various ways to obtain approximate solutions. The physics informed neural network (PINN) is proposed [7]. It can solve partial differential equations that are used to describe kinetic theory, such as the Burgers equation [8] and the wave equation [9]. Physical laws are introduced into the network as regularizers in PINNs. The constraints of physical laws enable PINNs to obtain good results from small training datasets. The parareal physics-informed neural network (PPINN) is introduced [10], which is developed from PINN. It decomposes a high-dimensional problem into many parts. These parts consist of separate small-scale problems conducted by an inexpensive or fast coarse-grained solver. Compared to the original PINN approach, which directly processes the entire large dataset, PPINN can accelerate computations by utilizing a small dataset and also improve efficiency by training in parallel. Theory-guided neural networks (TgNNs) are proposed [11]. In the TgNNs, the neural network is trained using available data guided by the theory of a related problem. It can make more accurate predictions than the DNN because of the prior information provided such as physical laws, engineering controls, and expert experience. Separable physics-informed neural networks (SPINNs) are designed to overcome the limitation of the expensive computational costs and heavy memory overhead [12]. The method operates on a per-axis basis to decrease the number of network propagations in multi-dimensional PDEs instead of point-wise processing in conventional PINNs.

The Boltzmann equation is one of the governing equations for dilute gases, ranging from continuum to free-molecular conditions, and it can describe the behavior of particles of matter in fluid dynamics [13]. This equation involves the general theory of statistics, and it describes the motion of large numbers of particles in a statistical sense. The Boltzmann equation can systematically describe macroscopic transport processes like diffusion, heat flow, and conductivity from the underlying microscopic laws of nature [14]. It underlies theories that describe the behavior of fluids, aerodynamics, and plasma dynamics in a practical and technological way in mesoscopic kinetic theory. This equation builds a solid foundation for the description of systems made up of equilibrium parts and the analysis of transport processes in equilibrium [15]. There is a collision term in the equation, which includes the probability density function of position and momentum. The Bhatnagar–Gross–Krook (BGK) model approximates the collision term of the Boltzmann equation based on a single relaxation process from a nonequilibrium state to an equilibrium state [16]. The multiple-relaxation-time (MRT) model is more stable than the BGK model because of the adjustable ratio between the kinematic and bulk viscosities and different relaxation times that can be individually tuned [17].

There have been many numerical computational methods used to solve the Boltzmann equation. The direct simulation Monte Carlo (DSMC) is proposed [18]. The DSMC method can accurately solve the non-equilibrium gas flow problems and simulate the continuum flows, where it reproduces the results from continuum computational fluid dynamics (CFD). It simulates the collision process between gas molecules and with surfaces by tracking virtual microscopic particles within the computational domain. It is limited by its high computational cost and low efficiency when dealing with a small number of flowing molecules. The discrete velocity method is introduced [19]. It can discretize the distribution function of the Boltzmann equation in different discrete velocity directions. The Fourier spectral method can make an outstanding contribution to accelerating the direct method.

By employing trigonometric functions to approximate the distribution function, the enhanced fast spectral method not only accelerates computations but also alleviates the memory constraints associated with the precomputation phase [20]. To address the need for high scalability, a parallel algorithm is introduced for tackling the regularized lattice Boltzmann method (RLBM) with large eddy simulation (LES) [21]. It features three innovative grid partitioning strategies and incorporates buffering technology to facilitate an efficient

parallel data exchange strategy. A parallel algorithm designed for a CPU–GPU heterogeneous platform aims at resolving the immersed boundary lattice Boltzmann method [22]. The algorithm leverages the combined computational prowess of CPUs and GPUs to yield enhanced results. The introduction of buffering techniques is pivotal in enhancing the precision of solving the Boltzmann equation across grids of varying sizes [23]. By employing buffer grids, the method effectively eliminates the need for temporal interpolation calculations while streamlining spatial interpolation processes.

The key to solving the Boltzmann equation using artificial intelligence is determining how to approximate the collision term effectively. Researchers have explored several methods that combine the advantages of partial differential equations and machine learning to solve the Boltzmann equation. These methods include solving the Boltzmann equation using the Bhatnagar–Gross–Krook collision model (Boltzmann-BGK equation). A neural network based on neural sparse representation for the Boltzmann equation is proposed [24]. A fully connected neural network is utilized to approximate the Boltzmann equation. The integration of Gaussian functions into the neural network is introduced to offer significant advantages in handling the high-dimensional Boltzmann-BGK equation [25]. Three subnetworks are proposed to build the model describing the Boltzmann-BGK equation [26]. The first one is for the equilibrium distribution function. The second one is utilized for the non-equilibrium distribution function. The third one describes the corresponding boundary and initial conditions. The relaxation time used for the equilibrium distribution function at each discrete velocity direction tested in these works is the same, which is unreasonable. The BGK collision model is one special case of the multiple-relaxation-time (MRT) model. The MRT model can be more stable than the BGK model by adjusting some free relaxation parameters [27].

In this work, a new physics-informed neural network describing the Boltzmann equation using multiple-relaxation-time (MRT) collision operators named the network for MRT collision (NMRT) is proposed for the Boltzmann equation. NMRT is an improved ansatz for the MRT collision model and can easily approximate the distribution function based on several parameters. The neural network structure is designed to be consistent with the characteristics of the Boltzmann equation with the MRT collision model. The process of solving the Boltzmann equation is ignored. The input and output are underlined. The setting of the initial conditions and the selection of the loss function determine the approximation efficiency.

To validate the accuracy and effectiveness of the proposed network structure, numerical experiments are conducted. Two one-dimensional problems are described. One is the advection–diffusion problem with continuous initial conditions. Another is the wave problem. To further study the applicability of the methods, the wave propagation problem in the two-dimensional scenarios is tested.

The remainder of this paper is organized as follows. The Boltzmann equation with the MRT collision model and its related properties are introduced in Section 2. The neural network for the MRT collision is discussed in Section 3. The numerical experiments and performances are set up in Section 4. Section 5 draws the conclusions.

2. Boltzmann Equation

The Boltzmann-BGK equation employs a single relaxation model. The relaxation process is described by a single relaxation time. The relaxation rate is the same at every moment of the process, which is not realistic. The multiple-relaxation-time (MRT) collision model can adjust the relaxation rates at different stages of the process and solve the problem. It has the following form [13]:

$$\frac{\partial m(\mathbf{x}, \mathbf{v}, t)}{\partial t} + \mathbf{v} \cdot \bigtriangledown_x m(\mathbf{x}, \mathbf{v}, t) = Q^{MRT}[m], t \in \mathbb{R}^+, x \in \mathbb{R}^3, v \in \mathbb{R}^3 \quad (1)$$

where $m(\mathbf{x}, \mathbf{v}, t)$ is the distribution function, t is the time, $\mathbf{x}$ is the spatial space, and $\mathbf{v}$ is the microscopic velocity of the particles. $Q^{MRT}[m]$ is the MRT collision operator, which has the following form:

$$Q^{MRT}[m] = -M^{-1}S(m - m_{eq}) \tag{2}$$

where m and m_{eq} are the conserved moment function and the equilibrium function, respectively. M and S are the transformation matrix and the diagonal relaxation matrix, respectively. m and m_{eq} can be converted to the distribution function f and the equilibrium distribution function f_{eq}, respectively, via the transformation matrix, which has the following form:

$$m = Mf, \;\; m_{eq} = Mf_{eq} \tag{3}$$

where f_{eq} can be expressed as the Maxwellian equation [28]:

$$f_{eq} = \frac{\rho}{\sqrt{2\pi T}^3} exp\left(-\frac{|v-u|^2}{2T}\right) \tag{4}$$

where ρ is the density, u is the macroscopic velocity, and T is the temperature.

The discrete velocity model D1Q3 [29] is adopted in the one-dimensional case. The corresponding transformation matrix M and the diagonal relaxation matrix S can be given by the following:

$$M = \begin{pmatrix} 1 & 1 & 1 \\ -c & 0 & c \\ c^2 & -2c^2 & c^2 \end{pmatrix}, \tag{5}$$

$$S = diag(s_0, s_1, s_2) \tag{6}$$

where c is the particle velocity. $c = 1$ is adopted in this work.

In the two-dimensional case, the adopted transformation matrix M and the diagonal relaxation matrix S of the discrete velocity model D2Q9 [30] can be given by the following:

$$M = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ -4 & -1 & -1 & -1 & -1 & 2 & 2 & 2 & 2 \\ 4 & -2 & -2 & -2 & -2 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & -1 & 0 & 1 & -1 & -1 & 1 \\ 0 & -2 & 0 & 2 & 0 & 1 & -1 & -1 & 1 \\ 0 & 0 & 1 & 0 & -1 & 1 & 1 & -1 & -1 \\ 0 & 0 & -2 & 0 & 2 & 1 & 1 & -1 & -1 \\ 0 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 \end{pmatrix}, \tag{7}$$

$$S = diag(s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8) \tag{8}$$

where s_i is the relaxation parameter, which is located in the range of $(0, 2)$.

In the D1Q3 MRT, the first two relaxation times $(s0, s1)$ are related to the mass and momentum conservation. The kinematic shear and bulk viscosities are related to the $s2$ relaxation. In the D2Q9 MRT, the first three relaxation times $(s0, s1, s2)$ are related to the mass and momentum conservation. The shear viscosity and the bulk viscosity are related to the relaxation parameters $(s3, s4, s5)$. $(s6, s7, s8)$ remain as free parameters.

In this work, $S = diag(0, 0, 0, s, s, s, 1.9, 1.9, 1.54)$ is adopted in the two-dimensional case, and $S = diag(0, 0, s)$ is adopted in the one-dimensional case.

3. Network for Boltzmann-MRT Equation

In this section, the neural network NMRT is proposed to approximate the Boltzmann equation with the MRT collision. A fully connected neural network is utilized to approximate

the Boltzmann-MRT equation, and the general framework of the neural network is proposed. The loss function to optimize the network parameters is designed. Several strategies of NMRT are also presented to efficiently approximate the equilibrium distribution function and the nonequilibrium function in the network.

3.1. Discrete Velocity Model

Considering that the distribution function depends on space, velocity space, and time, it is very challenging for the neural network to approximate it. The discrete velocity method [19] is widely utilized to discretize the Boltzmann equation in the microscopic velocity space. The method is helpful to solve the Boltzmann-MRT equation using a neural network. The key objective of discretization in the microscopic velocity space is to select a finite number of definite points, which are taken from the velocity space. They can be substituted into the Boltzmann equation to compute the distribution function at these points as well as the equilibrium distribution function. The summation of this series of points satisfies the conservation law.

The partition strategy is designed to divide the points in the microscopic velocity space into different groups and assign the corresponding groups of weights. It can adjust the training process and enhance the approximation efficiency of the network.

We assume three groups of discrete points in the microscopic velocity space as follows:

$$\begin{aligned} P_{IC} &= (v_1, v_2, \cdots, v_i), \\ P_{BC} &= (v_1, v_2, \cdots, v_j), \\ P_{IN} &= (v_1, v_2, \cdots, v_k) \end{aligned} \tag{9}$$

where $v_l (l = 1, 2, ..., max(i, j, k))$ are the points in the velocity space, P_{IC} are the discrete points for the initial condition, P_{BC} are the discrete points for the boundary condition, and P_{IN} are the interior discrete points for the Boltzmann equation.

The microscopic velocity spaces are made up of the three groups of points:

$$V \triangleq [P_{IC}, P_{BC}, P_{IN}]^T. \tag{10}$$

The discrete distribution functions are as follows:

$$\mathbf{m}(\mathbf{x}, \mathbf{v}, t) \triangleq \left[m_1(\mathbf{x}, \mathbf{v}_1 t), m_2(\mathbf{x}, \mathbf{v}_2, t), \cdots, m_{i+j+k}\Big(\mathbf{x}, \mathbf{v}_{i+j+k}, t\Big) \right]. \tag{11}$$

For improving the readability, $m_l(\mathbf{x}, t) = m(\mathbf{x}, \mathbf{v}_l, t)$ $(l = 1, 2, ..., i + j + k)$ is defined. For the MRT collision term, the discrete collision term is labeled as follows:

$$Q_l^{MRT} = Q\Big[m_1, m_2, \cdots, m_{i+j+k}\Big](\mathbf{x}, \mathbf{v}_l, t). \tag{12}$$

The Boltzmann-MRT equation is simplified as follows:

$$\begin{cases} \frac{\partial m_1(\mathbf{x},t)}{\partial t} + v_1 \cdot \bigtriangledown_x m_1(\mathbf{x}, t) = Q_1^{MRT} \\ \vdots \\ \frac{\partial m_{i+j+k}(\mathbf{x},t)}{\partial t} + v_{i+j+k} \cdot \bigtriangledown_x m_{i+j+k}(\mathbf{x}, t) = Q_{i+j+k}^{MRT} \end{cases} \tag{13}$$

Let

$$Q^{MRT}[\mathbf{m}] \triangleq \left[Q_1^{MRT}, Q_2^{MRT}, \cdots, Q_{i+j+k}^{MRT} \right]. \tag{14}$$

The Equation (1) is reduced as follows:

$$\frac{\partial \mathbf{m}(\mathbf{x}, t)}{\partial t} + \mathbf{v} \cdot \bigtriangledown_x \mathbf{m}(\mathbf{x}, t) = Q^{MRT}[\mathbf{m}](\mathbf{x}, t). \tag{15}$$

3.2. Framework of Neural Network

The general framework of the neural network is presented in this section. A fully connected neural network is utilized to approximate the Boltzmann-MRT equation. It consists of linear layers. Each node of the fully connected layer is fully interconnected with the nodes of the previous layer. The nodes can integrate the features extracted from the previous network and map these features to the sample labeling space. They will perform a weighted summation of the output of the features from the previous layer and input the result into the activation function to finalize the classification of the target [31]. The propagation of the network has the following form:

$$a^0 \to z^1 \to a^1 \to \cdots \to z^l \to a^l \tag{16}$$

where the l-th layer is as follows:

$$z^l = a^{l-1}W^l + b^l, \tag{17}$$

$$a^l = h\left(z^l\right). \tag{18}$$

Substituting the specific form of Equation (18) into Equation (17), it will be rewritten as follows:

$$a_j^l = h\left(z_j^l\right) = h\left(\sum_{k=1}^{m} w_{jk}^l a_k^{l-1} + b_j^l\right), j = 1, \ldots, n^l. \tag{19}$$

The superscript l is the data of the l-th layer, and the subscript is the matrix or vector index. n^l is the dimension of the l-th layer, and $h(z)$ is the activation function. W^l is the weight matrix of the l-th layers, and b^l is the network bias of the l-th layer. a^0 is the input and a^l is the output of the network.

The neuron activation function, which is the sine function, is utilized here:

$$h(z) = sin(z) \tag{20}$$

The sine activation function is a periodic function. Due to the existence and stability of the time periodic solution to the Boltzmann equation [32], it can capture the periodic characteristics of the data more easily than softplus or tanh. It is nonlinear and can introduce some nonlinear transformations to exhibit better neural representations [33], which is helpful to fit the data distribution.

To solve the Boltzmann-MRT equation using a neural network, it is important to determine the parameters of the network. $a^0 = (\mathbf{x}, t)$ are the inputs. The moment function m is the output.

3.2.1. Multiscale Modeling

According to the frequency principle [34], deep neural networks can accelerate the convergence of low frequencies and capture low frequencies first in computational problems. Low frequencies converge faster than high frequencies. When solving high-frequency partial differential equations, it is challenging to accelerate the convergence of high frequencies for the equations. Due to the high dimensionality of the Boltzmann equation, multiscale modeling is adopted to integrate multiscale, multiphysics data [35]. It can obtain the learning process of data with low frequencies through the conversion of data with high frequencies, which can improve the convergence speed of networks [36].

A series of constants is utilized to adjust the learning of data by multiplying the inputs of the network. The straightforward approach can produce a multiscale structure to achieve the goal of speeding up the convergence to the solution across a wide range of frequencies with a uniform accuracy. It has the following form:

$$a^0_{scale} = \begin{pmatrix} 1 \\ 2 \\ \vdots \\ n \end{pmatrix} a^0 = \begin{pmatrix} 1 \\ 2 \\ \vdots \\ n \end{pmatrix} (\mathbf{x}, t). \tag{21}$$

The series of constants ranges from 1 to a large number, which is problem-dependent. It is a hyperparameter in the network that can convert the neural network to the network at multiple topological scales. The network is able to achieve a faster convergence when approximating the PDE model. It is not explicitly clear which constant functions well, making it challenging to select them in a proper way.

Based on the discrete wavelet transform in wavelet theory [37], the constants are chosen as $(1, 4, 16)$, which are powers of 2.

3.2.2. Micro–Macro Decomposition

The Chapman–Enskog expansion is an analytical tool in kinetic theory that is also used in the Boltzmann equation. It can derive macroscopic balance laws from collision models in the Boltzmann equation. The micro–macro decomposition method [38] is carried out, corresponding to a first-order Chapman–Enskog expansion. It can split the distribution function into an equilibrium part and a non-equilibrium part [39]. The moment function has the following form:

$$\mathbf{m}(\mathbf{x}, t) = \mathbf{M}^{eq}(\mathbf{x}, t) + \mathbf{C}\mathbf{M}^{neq}(\mathbf{x}, t) \tag{22}$$

where $\mathbf{C}$ is a constant. It is related to the construction of the coupled system.

Substituting Equation (22) into Equation (15), we obtain the following:

$$\begin{aligned} &\frac{\partial \mathbf{M}^{eq}(\mathbf{x}, t)}{\partial t} + \mathbf{v} \cdot \bigtriangledown_x \mathbf{M}^{eq}(\mathbf{x}, t) \\ &+ \mathbf{C}\left(\frac{\partial \mathbf{M}^{neq}(\mathbf{x}, t)}{\partial t} + \mathbf{v} \cdot \bigtriangledown_x \mathbf{M}^{neq}(\mathbf{x}, t) \right) \\ &= Q^{MRT}[\mathbf{M}^{eq}(\mathbf{x}, t) + \mathbf{C}\mathbf{M}^{neq}] \end{aligned} \tag{23}$$

with

$$\begin{aligned} &Q^{MRT}[\mathbf{M}^{eq}(\mathbf{x}, t) + \mathbf{C}\mathbf{M}^{neq}(\mathbf{x}, t)] \\ &= -M^{-1}S(\mathbf{M}^{eq}(\mathbf{x}, t) + \mathbf{C}\mathbf{M}^{neq} - \mathbf{M}^{eq}(\mathbf{x}, t)) \\ &= -\mathbf{C}M^{-1}S\mathbf{M}^{neq}. \end{aligned} \tag{24}$$

The equilibrium part in Equation (22) has the following form:

$$\mathbf{M}^{eq}(\mathbf{x}, t) = M\mathbf{f}^{eq}(\mathbf{x}, t) \tag{25}$$

where

$$\mathbf{f}^{eq}(\mathbf{x}, t) = \frac{\mathbf{æ}(\mathbf{x}, t)}{\sqrt{2\pi\mathbf{T}(\mathbf{x}, t)}^3} exp\left(-\frac{|\mathbf{v} - \mathbf{u}(\mathbf{x}, t)|^2}{2\mathbf{T}(\mathbf{x}, t)} \right). \tag{26}$$

The non-equilibrium function has the following form:

$$\mathbf{M}^{neq}(\mathbf{x}, t) = \theta(\mathbf{x}, t). \tag{27}$$

Two neural networks (NN_{eq} and NN_{neq}) are proposed to approximate the equilibrium $\mathbf{M}^{eq}$ and the non-equilibrium $\mathbf{M}^{neq}$, respectively:

$$\begin{aligned} NN_{eq} &\Rightarrow \rho, \mathbf{u}, \mathbf{T}, \\ NN_{neq} &\Rightarrow \theta. \end{aligned} \tag{28}$$

For NN_{eq}, only $\rho, \mathbf{u}, \mathbf{T}$ are focused on and substituted into Equation (26) to obtain the equilibrium part. They are utilized to generate the distribution function instead of outputting the equilibrium part by training the neural network directly, which can improve the adaptability of the neural network to approximate the Boltzmann equation.

For NN_{neq}, the output is θ.Different from NN_{eq}, the number of the output channels in NN_{neq} is problem-dependent. The output channel is the prediction or decision made by a machine learning model based on input data [40]. The number of output channels directly affects the ability of the model to react to various data and identify the features that are best-suited for the model. Due to the complexity of the non-equilibrium $\mathbf{M}^{neq}$, the rank for different output channels is set. The rank depends on the complexity of the initial condition problem.

3.2.3. Approximation of MRT Collision Term

When using a neural network to approximate the MRT collision term, the key point is to combine the collision matrix with discrete distribution functions within the moment space. The Boltzmann equation with the MRT collision model is a high-dimensional PDE. The high-order tensor obtained after discretization can be represented using a variety of lower-order tensors via tensor decomposition [41]. Tensor decomposition is exploited for the discrete distribution function. The decomposed lower-order tensors can be multiplied by the collision matrix directly. Tensor decomposition can also reduce the number of parameters in the last layer of the network, which decreases the computational cost and increases the approximation efficiency [42].

The singular value decomposition (SVD) of a matrix is a factorization of that matrix into three matrices [43]. For the high-order tensor in the Boltzmann-MRT equation, the canonical polyadic decomposition (CPD) [44] is utilized, which is a straightforward extension of SVD in higher-dimensional space.

Assuming three groups of discrete points are selected, as shown in Equation (10), the moment function in Equation (11) is a third-order tensor with the following form:

$$\mathbf{m}(\mathbf{x}, t) \in \mathbb{R}^{\mathrm{i}\times\mathrm{j}\times\mathrm{k}} \tag{29}$$

where i, j, and k are the same as those in Equation (10).

It can be approximated using CPD as follows:

$$\begin{aligned} \mathbf{m}(\mathbf{x}, t) &\simeq \sum_{r=1}^{R} a_r \circ b_r \circ c_r, \\ \mathbf{m}_{ijk}(\mathbf{x}, t) &\simeq \sum_{r=1}^{R} a_{ir} b_{jr} c_{kr} \end{aligned} \tag{30}$$

where $a \in \mathbb{R}^{\mathrm{i}}, b \in \mathbb{R}^{\mathrm{j}}, c \in \mathbb{R}^{\mathrm{k}}$, and R is the tensor rank.

For the equilibrium $\mathbf{M}^{eq}$, it has the low-rank form below:

$$\mathbf{M}^{eq} = \mathbf{P}_i \mathbf{O}_j \mathbf{R}_k \tag{31}$$

where

$$\begin{aligned} \mathbf{P}_i &= \frac{\rho^{\frac{1}{3}}}{\sqrt{2\pi T}^3} exp\left(-\frac{|v_i - u|^2}{2T}\right), \\ \mathbf{O}_j &= \frac{\rho^{\frac{1}{3}}}{\sqrt{2\pi T}^3} exp\left(-\frac{|v_j - u|^2}{2T}\right), \\ \mathbf{R}_k &= \frac{\rho^{\frac{1}{3}}}{\sqrt{2\pi T}^3} exp\left(-\frac{|v_k - u|^2}{2T}\right). \end{aligned} \tag{32}$$

Define $\mathbf{m} \triangleq \langle (a, b, c) \rangle$, it holds that

$$(\mathbf{m}(t))' \triangleq (\langle (a(t), b(t), c(t)) \rangle)' \triangleq \left\langle \left(\widehat{a}, \widehat{b}, \widehat{c} \right) \right\rangle \tag{33}$$

where $\widehat{a} = [a'(t), a(t), a(t)], \widehat{b} = [b(t), b'(t), b(t)], \widehat{c} = [c(t), c(t), c'(t)]$.

The process of decomposing the third-order tensors will be completed automatically in the learning process of the neural network. The total structure of the neural network for MRT collision (NMRT) is shown in Figure 1. It is designed to approximate the collision term Q^{MRT} effectively. The collision term Q^{MRT} is a part of the Boltzmann equation:

$$Q^{MRT} = \Lambda(m - m_{eq}) \tag{34}$$

where $\Lambda = -M^{-1}S$. m and m_{eq} are given in Equation (22) and Equation (25), respectively.

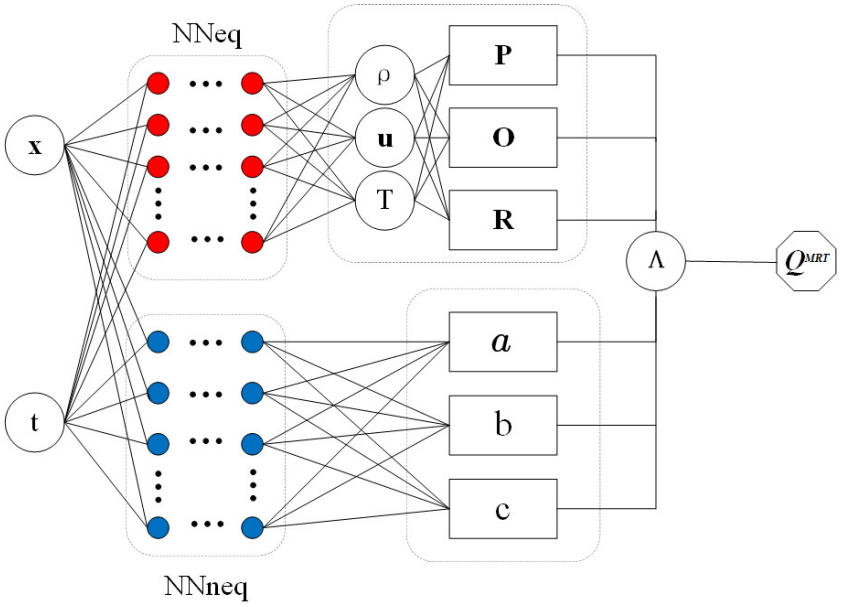


Figure 1. Network architecture. **x** is spatial space and t is time, which all are the inputs of the network. The Monte Carlo method is used to create a dataset. Multiscale modeling and canonical polyadic decomposition are adopted in the neural network.

3.3. Loss Function

The loss function designed to optimize the network parameters is discussed in this section. To inform the neural network of prior knowledge about the equation, the loss function usually consists of four parts [45]: the data matching loss (DM loss), the residual loss for partial differentiable structure (PDE loss), the boundary condition loss (BC loss), and the initial condition loss (IC loss). It has the following form:

$$L = L_{DM} + L_{PDE} + L_{BC} + L_{IC}. \tag{35}$$

For the NMRT, the dataset is generated via Monte Carlo sampling, which consists of uniformly distributed random points. It is not a representation of real-world data. There is no data matching term for the training data matched with the true value. The loss function of the NMRT has the following form:

$$L_{MRT} = L_{PDE} + L_{BC} + L_{IC}. \tag{36}$$

Assuming the residual points for the partial differentiable structure, boundary, and initial conditions are Γ^f, Γ^b, and Γ^i, respectively, the loss functions can be rewritten as follows:

$$\begin{aligned} L_{PDE} &= \frac{1}{|\Gamma^f|}\sum_{\mathbf{x}\in\Gamma^f}\|\mathbf{r}(\mathbf{x},t)\|_2^2, \\ L_{BC} &= \frac{1}{|\Gamma^b|}\sum_{\mathbf{x}\in\Gamma^b}\left\|\mathbf{m}(\mathbf{x},t)-\mathbf{m}^b(\mathbf{x},t)\right\|_2^2, \\ L_{IC} &= \frac{1}{|\Gamma^i|}\sum_{\mathbf{x}\in\Gamma^i}\left\|\mathbf{m}(\mathbf{x},0)-\mathbf{m}^i(\mathbf{x},0)\right\|_2^2 \end{aligned} \tag{37}$$

where $\mathbf{m}^b(\mathbf{x},t)$ and $\mathbf{m}^i(\mathbf{x},0)$ are the boundary and initial conditions for the Boltzmann-MRT equation. The function $\mathbf{r}(\mathbf{x},t)$ is the residual of the network. By moving a term from one side of the equation to the other, it has the following form:

$$\mathbf{r}(\mathbf{x},t) = \frac{\partial \mathbf{m}(\mathbf{x},t)}{\partial t} + \mathbf{v}\cdot\bigtriangledown_x\mathbf{m}(\mathbf{x},t) - Q^{MRT}[\mathbf{m}](\mathbf{x},t). \tag{38}$$

The neural network is designed to approximate the Boltzmann equation. All differential terms in the residual of the network can be considered as parts of the equation. They are obtained automatically during the learning process of the neural network.

Gram–Schmidt orthogonalization is utilized to transform the matrix. L_2-norm is selected as the distance function. However, all the discrete points have the same weight. The discrete distribution function for different points in the microscopic velocity space does not affect the result of the Boltzmann-MRT equation. The revised loss function is chosen and will be introduced below.

Problem-Dependent Weight Loss

When the distance function is the L_2-norm distance, the distribution function with points for different initial and boundary conditions will play different roles for different outcomes. It is important to balance the weight of the distance for functions with different conditions at different velocities.

Three groups of weights corresponding to Equation (9) are set as follows:

$$\begin{aligned} W_{IC} &= (w_1, w_2, \cdots, w_i), \\ W_{BC} &= (w_1, w_2, \cdots, w_j), \\ W_{IN} &= (w_1, w_2, \cdots, w_k). \end{aligned} \tag{39}$$

Define

$$\mathbf{W} \triangleq [W_{IC}, W_{BC}, W_{IN}]^T. \tag{40}$$

A lower bound-constrained uncertainty weighting method [31] is utilized to revise the loss function in Equation (37). It can be rewritten as follows:

$$\begin{aligned} L_{PDE} &= \frac{1}{|\Gamma^f|}\sum_{\mathbf{x}\in\Gamma^f}\sum_{i\in Y^f}\frac{1}{2\left(\varepsilon^2+(w_{IN})_i^2\right)}\mathbf{r}^2(\mathbf{x},t) + \log\left(\varepsilon^2+(w_{IN})_i^2\right), \\ L_{BC} &= \frac{1}{|\Gamma^b|}\sum_{\mathbf{x}\in\Gamma^b}\sum_{i\in Y^b}\frac{1}{2\left(\varepsilon^2+(w_{BC})_i^2\right)}\left(\mathbf{m}(\mathbf{x},t)-\mathbf{m}^b(\mathbf{x},t)\right)^2 + \log\left(\varepsilon^2+(w_{BC})_i^2\right), \\ L_{IC} &= \frac{1}{|\Gamma^i|}\sum_{\mathbf{x}\in\Gamma^i}\sum_{j\in Y^i}\frac{1}{2\left(\varepsilon^2+(w_{IC})_j^2\right)}\left(\mathbf{m}(\mathbf{x},0)-\mathbf{m}^i(\mathbf{x},0)\right)^2 + \log\left(\varepsilon^2+(w_{IC})_j^2\right) \end{aligned} \tag{41}$$

where Y^f, Y^b and Y^i are the weights corresponding to Γ^f, Γ^b and Γ^i. The lower bound of the loss term is constrained by ε^2 when w is decreased to 0, which can prevent division by

zero. The w values are problem-dependent and adaptive in the neural network, which are set as parameters. The compositions of the loss function are shown in Figure 2.

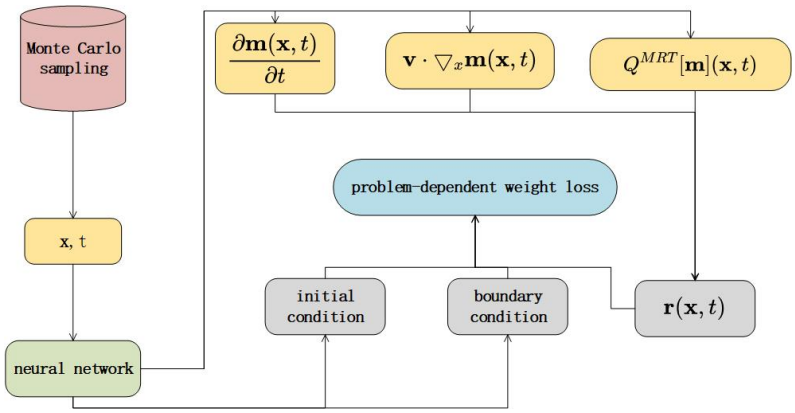


Figure 2. Compositions of the loss function. **x** is spatial space and t is time, which are inputs to the network. The Monte Carlo method is used to create a dataset. The loss function is made up of three parts: IC loss, BC loss, and PDE loss. The problem-dependent weights for the three loss parts are adopted.

4. Numerical Experiment

In this section, several comprehensive experimental validations are presented using classical CFD cases, which are applied to the NMRT method proposed in this work.

In order to ensure applicability, the one-dimensional advection–diffusion problem and the wave problem, as well as the wave problem within the two-dimensional scenario, are tested. The Adam optimizer is an optimization algorithm that is different from the classical stochastic gradient descent procedure. It integrates the concepts of momentum and adaptive learning rates, offering fast convergence and the ability to automatically adjust. It can adaptively and iteratively update network weights. It is widely utilized in solving PDEs using a neural network [26]. Cosine annealing is a type of learning rate schedule that starts with a large learning rate. The basic idea is to start with a higher learning rate and then reduce it according to the cosine function over multiple epochs. The learning rate will be relatively rapidly decreased to a minimum value before being increased rapidly again [46]. It is commonly used in the training of neural networks to fine-tune the learning rate for better performance and convergence. The t-th step of the learning rate has the following form:

$$\alpha_t = \eta \cdot \cos\left(\frac{\pi \cdot t}{T}\right) \tag{42}$$

where α_t is the learning rate of the t-th iteration, η is the initial learning rate, and T denotes the total number of iterations. In the process of cosine annealing, the learning rate will gradually decrease from the initial value to the minimum value, which can make the training process more stable and effective.

All the experiments are conducted using PyTorch 2.1. The computational resources employed in this paper include NVIDIA RTX Titan (sourced from Shanghai Ocean University's GPU server, Shanghai, China) based on the Turing architecture.

4.1. Advection Diffusion

Advection diffusion is a mechanism of material dispersion within a fluid, driven by variations in density and temperature that induce convective movements of the fluid's constituents [47]. This process involves the movement of particles along paths of minimal resistance due to the fluid's motion, leading to diffusion. Advection diffusion encompasses both eddy diffusion and molecular diffusion occurring at the interface between the turbulent flow and the boundary layer. It is a concept of significant relevance in various domains, including fluid dynamics, environmental engineering, and chemical engineering.

Advection–diffusion problems are often difficult to solve analytically, making numerical simulation an important tool for studying such issues. Techniques such as CFD are widely used to simulate convective diffusion phenomena [48].

The initial condition has the following form:

$$\begin{aligned} \rho(x) &= \frac{1}{\sqrt{0.5\pi}} e^{-8x^2}, \\ u(x) &= 0, \\ T(x) &= cos(\pi x) + 1. \end{aligned} \tag{43}$$

The computational domain x is from -0.5 to 0.5. The simulation time is considered as 0.1 s, which is enough to demonstrate the ability of the network.

In the simulation for the advection–diffusion problem using the NMRT, M^{eq} and m are two fully connected networks of five layers. There are 80 neurons in each layer. The numbers for three groups of points in Equation (10) are chosen as N_{IC} = 100, N_{BC} = 100, and N_{IN} = 700. A total of 100 points are sampled in the computational space at the initial moment. A total of 100 points are sampled at the origin for the whole process of the simulation time. A total of 700 points are sampled in $x \times t \in [-0.5, 0.5] \times [0, 0.1]$ for the Boltzmann-MRT equation. The generated mesh for the microscopic velocity is considered as $[-10, 10]^3$. The grid size in the generated mesh in each direction of the microscopic velocity space is 24. The diagonal element s of the relaxation matrix S is the relaxation parameter, considered as 0.01, 0.1, and 1.0. The iteration step epoch is 10,000. The parameters of the NMRT method for the advection–diffusion problem are shown in Table 1.

Table 1. Parameters of the NMRT method for the advection–diffusion problem.

Neural Network	Layer number	5
	Neurons	80
	Steps	10,000
Sampling Points	N_{IC}	100
	N_{BC}	100
	N_{IN}	700
Computational Parameters	Time	$t \in [0, 0.1]$
	Relaxation parameter s	0.01, 0.1, 1.0
	Microscopic velocity space	$[-10, 10]^3$
	Grid numbers	72
Optimizer	Method	Adam
	Max learning rate	0.001
	Min learning rate	0.00005
	Decay algorithm	Cosine annealing

The numerical results of the advection–diffusion problem using NMRT for $s = 0.01, 0.1$, and 1.0 are plotted in Figure 3. Three macroscopic variables, including ρ, u, and T, are studied, and their results at $t = 0$ and $t = 0.1$ are shown. The fast spectral method (FSM) [49] is adopted as the reference to validate the accuracy of the numerical results. There is a small error between the numerical results and the reference result. Compared to the results at $t = 0$, the numerical results show no difference from the reference one at $t = 0.1$ for the three relaxation parameters. The solution gradually becomes smoother over time, with a corresponding decrease in error, which is a departure from traditional numerical methods.

To further validate the effectiveness of the NMRT method, the relative errors for the three macroscopic variables, including ρ, u, and T, between the numerical results and the reference result are defined.They have the following form:

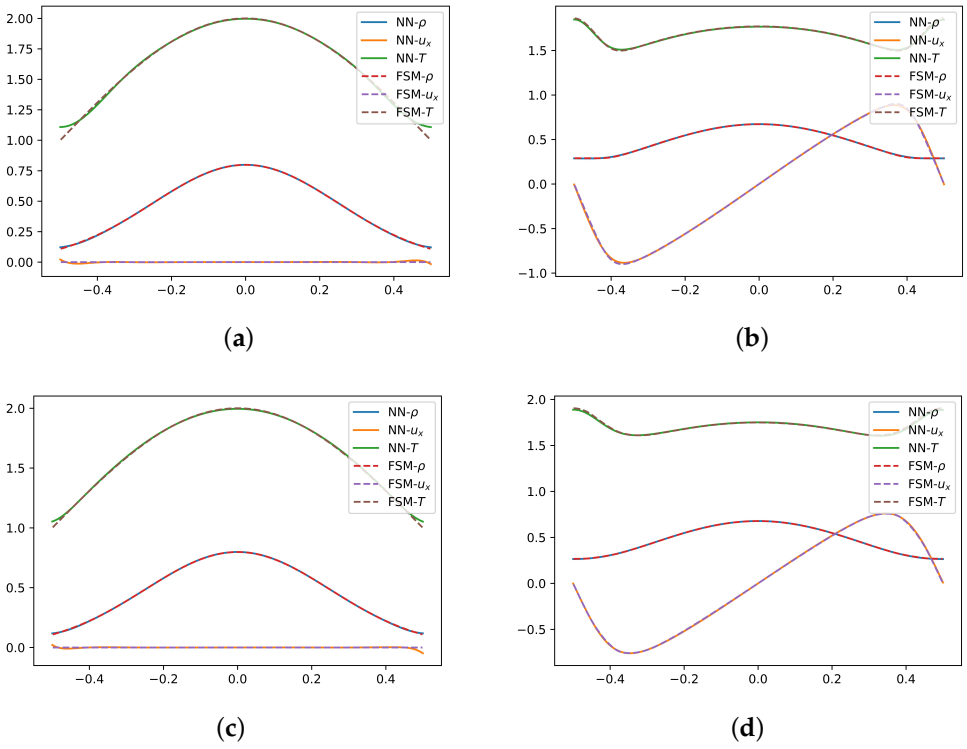
$$\begin{aligned} error_\rho &= \frac{\|\rho_{NMRT} - \rho_{FSM}\|_2}{\|\rho_{NMRT}\|_2}, \\ error_u &= \frac{\|u_{NMRT} - u_{FSM}\|_2}{0.01 + \|u_{NMRT}\|_2}, \\ error_T &= \frac{\|T_{NMRT} - T_{FSM}\|_2}{\|T_{NMRT}\|_2} \end{aligned} \tag{44}$$

where it is necessary to avoid the denominator $u = 0$ to prevent division by zero. The relative errors for ρ, u, and T with different relaxation parameters at $t = 0$ and $t = 0.1$ are shown in Table 2.

Table 2. Relative error of the advection–diffusion problem for ρ, u, and T between NMRT and FSM with $s = 0.01, 0.1$, and 1.0 at $t = 0$ and $t = 0.1$.

	s = 0.01			**s = 0.1**			**s = 1.0**		
t	***ρ***	***u***	***T***	***ρ***	***u***	***T***	***ρ***	***u***	***T***
0.0	1.30×10^{-3}	3.49×10^{-2}	1.07×10^{-2}	4.35×10^{-3}	3.35×10^{-2}	1.47×10^{-2}	3.45×10^{-3}	3.26×10^{-2}	1.20×10^{-2}
0.1	3.10×10^{-3}	2.41×10^{-2}	8.43×10^{-3}	1.85×10^{-3}	1.59×10^{-2}	4.69×10^{-3}	2.54×10^{-3}	4.55×10^{-2}	1.05×10^{-2}

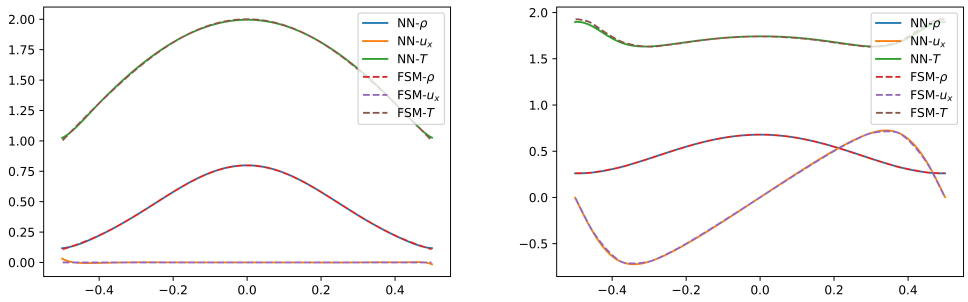
The errors at $t = 0$ for T can all reach the magnitude $O(10^{-2})$. The error does not monotonically increase over time because the function becomes smoother in the neural network [50].



(**a**) (**b**)

(**c**) (**d**)

Figure 3. *Cont.*



(**e**) (**f**)

Figure 3. Numerical solution of advection–diffusion problem using NMRT for $s = 0.01, 0.1, 1.0$ at $t = 0.0$ and 0.1. The numerical solution of NMRT is the solid line, and the reference solution is the dashed line. (**a**) s = 0.01, t = 0; (**b**) s = 0.01, t = 0.1; (**c**) s = 0.1, t = 0; (**d**) s = 0.1, t = 0.1; (**e**) s = 1.0, t = 0; (**f**) s = 1.0, t = 0.1.

4.2. Wave Propagation

Wave analysis plays an important role in many fields such as electromagnetics and acoustics. In fluid dynamics, the fluid field and wave source can determine wave propagation. With changes in fluid attributes, wave propagation is also affected in many different ways. To further understand flow behavior and potential mechanisms like heat transfer and turbulence, it is important to analyze the fluid wave phenomenon. The wave equation is used here to describe the wave propagation.

The initial condition of the wave equation has the following form:

$$\begin{aligned} \rho(x) &= \frac{\sin(2\pi x)}{2} + 1, \\ \mathbf{u}(x) &= 0, \\ T(x) &= \frac{\sin(2\pi x + 0.2)}{2} + 1. \end{aligned} \tag{45}$$

The computational domain x is from -0.5 to 0.5. The simulation time is considered as 0.1 s, which is sufficient to demonstrate the ability of the network. The periodic boundary condition is imposed. The cycle is equal to 1. In the simulation of the wave propagation problem using the NMRT, the numbers for three groups of points in Equation (10) are chosen as $N_{IC} = 100$, $N_{BC} = 100$, and $N_{IN} = 700$. The grid size in the generated mesh in each direction of the microscopic velocity space is 24. The iteration step epoch is 10,000. The parameters of the NMRT method for the wave propagation problem are shown in Table 3.

Table 3. Parameters of the NMRT method for the wave propagation problem.

Neural Network	Layer number	5
	Neurons	80
	Steps	10,000
Sampling Points	N_{IC}	100
	N_{BC}	100
	N_{IN}	700
Computational Parameters	Time	$t \in [0, 0.1]$
	Relaxation parameter s	0.01, 0.1, 1.0
	Microscopic velocity space	$[-10, 10]^3$
	Grid numbers	24

Table 3. *Cont.*

Optimizer	Method	Adam
	Max learning rate	0.001
	Min learning rate	0.00005
	Decay algorithm	Cosine annealing

The numerical results of the wave propagation problem using NMRT for $s = 0.01, 0.1$ and 1.0 are plotted in Figure 4. Three macroscopic variables, including ρ, u, and T, are studied, and their results at $t = 0$ and $t = 0.1$ are shown. The FSM is adopted as the reference to validate the accuracy of the numerical results. There is little discrepancy in the numerical results between the network-based solution and the reference solution. They agree well with each other.

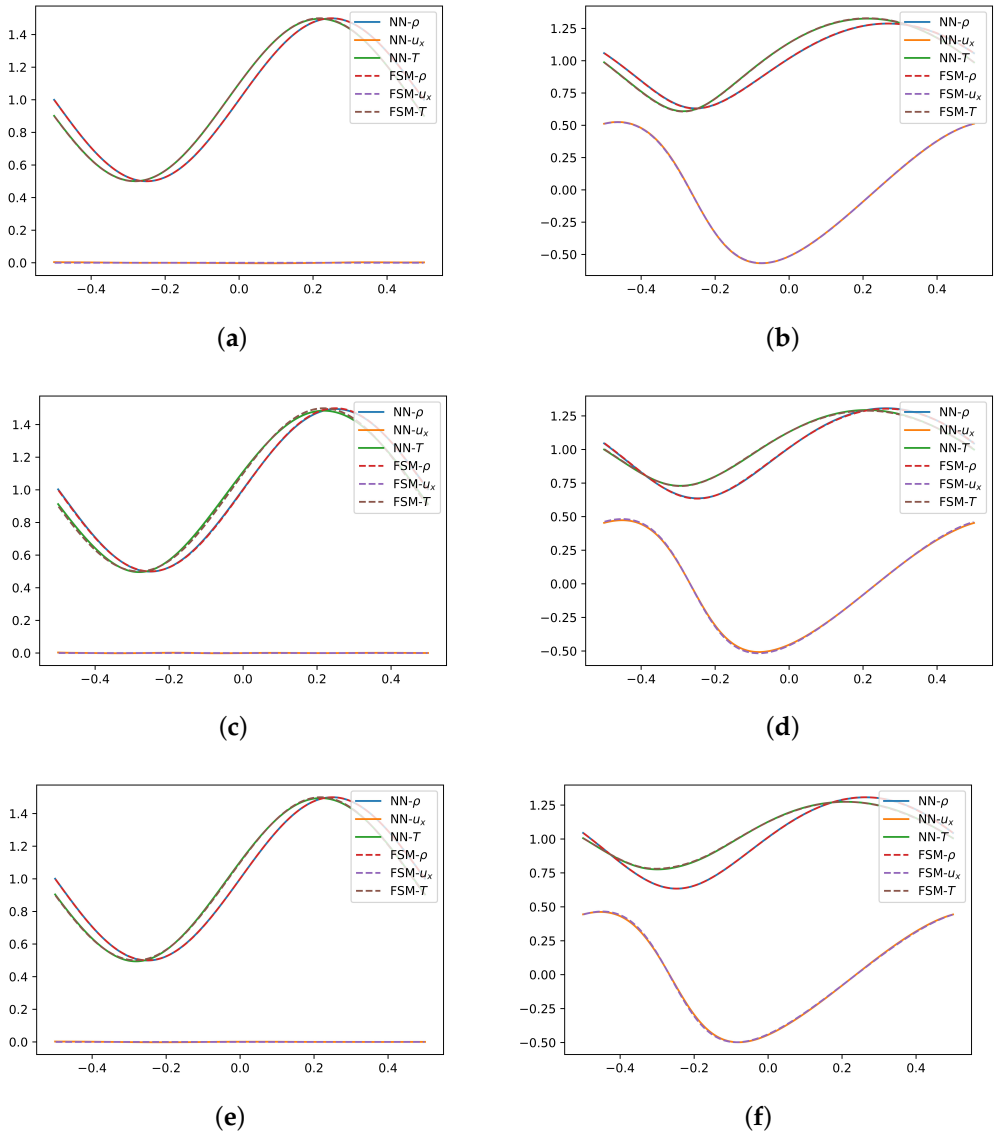


Figure 4. Numerical solution of wave propagation problem using NMRT for $s = 0.01, 0.1, 1.0$ at $t = 0.0$ and 0.1. The numerical solution of NMRT is the solid line, and the reference solution is the dashed line. (**a**) s = 0.01, t = 0; (**b**) s = 0.01, t = 0.1; (**c**) s = 0.1, t = 0; (**d**) s = 0.1, t = 0.1; (**e**) s = 1.0, t = 0; (**f**) s = 1.0, t = 0.1.

Table 4 shows the relative errors for ρ, u, and T with different relaxation parameters at $t = 0$ and $t = 0.1$.

Table 4. Relative error of wave propagation problem for ρ, u, and T between NMRT and FSM with $s = 0.01, 0.1$, and 1.0 at $t = 0$ and $t = 0.1$.

	s = 0.01			**s = 0.1**			**s = 1.0**		
t	ρ	u	T	ρ	u	T	ρ	u	T
0.0	1.68×10^{-3}	5.42×10^{-3}	1.74×10^{-3}	1.88×10^{-3}	3.09×10^{-3}	5.32×10^{-3}	2.22×10^{-3}	5.57×10^{-3}	7.21×10^{-3}
0.1	1.36×10^{-3}	4.21×10^{-3}	1.65×10^{-3}	1.71×10^{-3}	1.01×10^{-2}	1.95×10^{-3}	1.29×10^{-3}	1.64×10^{-2}	4.37×10^{-3}

This validates the high accuracy of the approximating functions of the neural network. The errors at $t = 0$ and $t = 1$ for ρ and T can all reach the magnitude $O(10^{-3})$.

4.3. Wave Propagation in Two-Dimensional Scenarios

The wave propagation problem in the one-dimensional case describes how the wave propagates positively along the x-axis. It needs to take more into account for wave propagation in two-dimensional scenarios.

The initial condition of the two-dimensional problem has the following form:

$$\begin{aligned} \rho(x,y) &= \frac{\sin(2\pi x)\sin(2\pi y)}{2} + 1 \\ \mathbf{u}(x,y) &= 0 \\ T(x,y) &= 1 \end{aligned} \tag{46}$$

The computational space $x \times y$ is $[-0.5, 0.5] \times [-0.5, 0.5]$. The simulation time is considered as 0.1 s, which is sufficient to demonstrate the ability of the network. The adopted boundary condition is periodic, which can make the macroscopic variables evolve periodically.

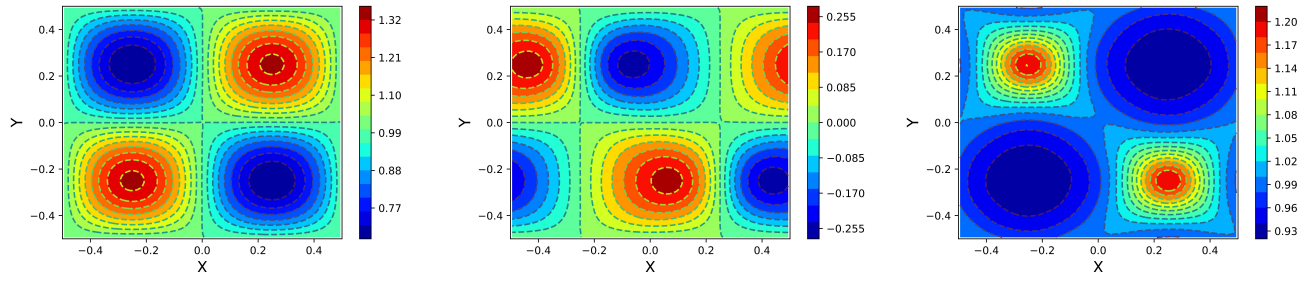
In the simulation for the wave propagation problem in two-dimensional scenarios using the NMRT, the numbers for three groups of points in Equation (10) are chosen as N_{IC} = 500, N_{BC} = 500, and N_{IN} = 2000. Due to its two-dimensionality, the points in $x \times y$ at $t = 0$ selected are all increased. The grid numbers in the generated mesh in each direction of the microscopic velocity space are 24. The iteration step epoch is increased to 12,000. The parameters of the NMRT method for the wave propagation problem in two-dimensional scenarios are shown in Table 5.

Table 5. Parameters of the NMRT method for the two-dimensional wave propagation problem.

Neural Network	Layer number	5
	Neurons	80
	Steps	12,000
Sampling Points	N_{IC}	500
	N_{BC}	500
	N_{IN}	2000
Computational Parameters	Time	$t \in [0, 0.1]$
	Relaxation parameter s	0.01, 0.1, 1.0
	Microscopic velocity space	$[-10, 10]^3$
	Grid numbers	24
Optimizer	Method	Adam
	Max learning rate	0.002
	Min learning rate	0.00005
	Decay algorithm	Cosine annealing

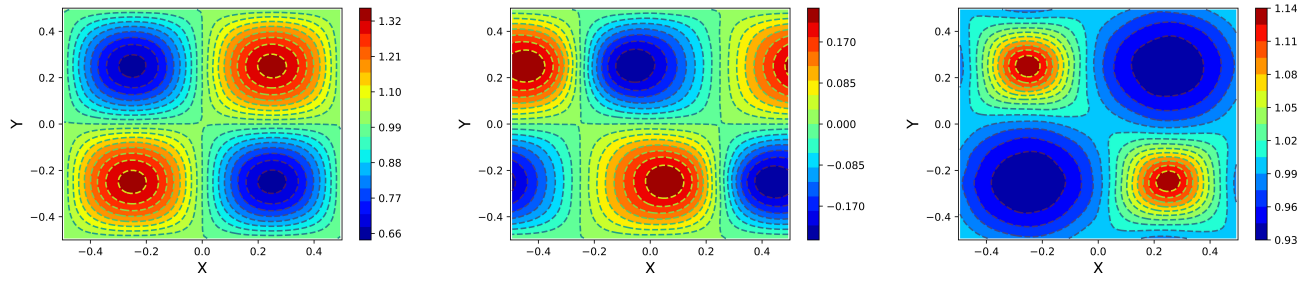
The numerical results of the wave propagation problem using NMRT in two-dimensional scenarios for $s = 0.01, 0.1$, and 1.0 at $t = 0.1$ are plotted. Three macroscopic variables, including ρ, u, and T for $s = 0.01$ at $t = 0.1$ are shown in Figure 5. All of them are consistent with the reference. The FSM with the spatial mesh is adopted as the reference to validate

the accuracy of the numerical results. Three macroscopic variables, including ρ, u, and T for $s = 0.1$ and 1.0 at $t = 0.1$, are shown in Figures 6 and 7. There is little discrepancy in the numerical results between the network-based solution and the reference solution. Whether the reference solution is accurate enough depends on the maximum memory of the parameter setting.



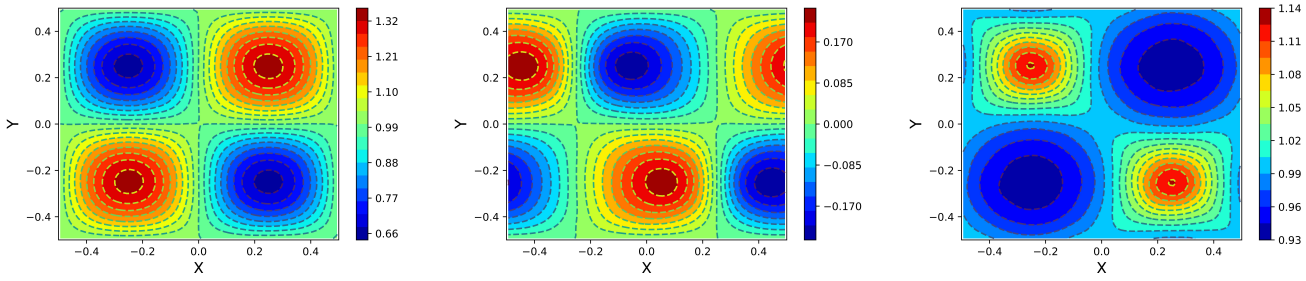
(**a**) (**b**) (**c**)

Figure 5. Numerical solution of the wave propagation problem in two-dimensional scenarios using NMRT for $s = 0.01$ at $t = 0.1$. The first row corresponds to the density ρ, the second column corresponds to the macroscopic velocity u, and the last column corresponds to the temperature T. The numerical solution of NMRT is the solid line, and the result of FSM is the dashed line. (**a**) $s = 0.01$, $t = 0.1$; (**b**) $s = 0.01$, $t = 0.1$; (**c**) $s = 0.01$, $t = 0.1$.



(**a**) (**b**) (**c**)

Figure 6. Numerical solution of wave propagation problem in two-dimensional scenarios using NMRT for $s = 0.1$ at $t = 0.1$. The first row corresponds to the density ρ, the second column corresponds to the macroscopic velocity u, and the last column corresponds to the temperature T. The numerical solution of NMRT is the solid line, and the result of FSM is the dashed line. (**a**) $s = 0.1$, $t = 0.1$; (**b**) $s = 0.1$, $t = 0.1$; (**c**) $s = 0.1$, $t = 0.1$.



(**a**) (**b**) (**c**)

Figure 7. Numerical solution of the wave propagation problem in two-dimensional scenarios using NMRT for $s = 1.0$ at $t = 0.1$. The first row corresponds to the density ρ, the second column corresponds to the macroscopic velocity u, and the last column corresponds to the temperature T. The numerical solution of NMRT is the solid line, and the result of FSM is the dashed line. (**a**) $s = 1.0$, $t = 0.1$; (**b**) $s = 1.0$, $t = 0.1$; (**c**) $s = 1.0$, $t = 0.1$.

The relative errors for ρ, u, and T with different relaxation parameters at $t = 0$ and $t = 0.1$ are shown in Table 6.

Table 6. Relative error of wave propagation problem in two-dimensional scenarios for ρ, u, and T between NMRT and FSM with $s = 0.01, 0.1$, and 1.0 at $t = 0$ and $t = 0.1$.

	s = 0.01			**s = 0.1**			**s = 1.0**		
t	ρ	u	T	ρ	u	T	ρ	u	T
0.0	2.73×10^{-1}	2.72×10^{-4}	9.63×10^{-4}	2.73×10^{-1}	3.90×10^{-4}	7.32×10^{-4}	2.73×10^{-1}	2.86×10^{-4}	1.28×10^{-3}
0.1	3.23×10^{-3}	2.50×10^{-3}	1.70×10^{-3}	3.62×10^{-3}	2.98×10^{-3}	4.32×10^{-3}	3.74×10^{-3}	3.47×10^{-3}	4.33×10^{-3}

This demonstrates that there is little error for the initial data at $t = 0$, mostly reaching the magnitude $O(10^{-4})$. The errors at $t = 1$ for ρ and T are all increased to the same order, reaching the magnitude $O(10^{-3})$.

5. Conclusions

Neural networks provide a new route for developing scientific modeling and simulations. A new physics-informed neural network describing the Boltzmann equation using multiple-relaxation-time (MRT) collision operators, named the network for MRT collision (NMRT), is proposed for the Boltzmann equation. This neural network is an improved ansatz for the MRT collision model and can easily approximate the distribution function using several parameters. Multiscale modeling is adopted to accelerate the convergence of high frequencies for the equations. The method of tensor decomposition in the Boltzmann-MRT equation is proposed to combine the collision matrix with discrete distribution functions within the moment space. The problem-dependent weight loss function is designed to balance the weight of the distance for the function at different conditions at different velocities and improve the efficiency of the NMRT, which is composed of PDE loss, BC loss, and IC loss.

Numerical experiments using classical CFD cases are applied to the NMRT method to validate the accuracy and efficiency of the method. The one-dimensional advection–diffusion problem and wave problem are tested. The results show that the network can approximate the function well. The L_2-norm error can reach the magnitude $O(10^{-3})$. The wave problem in two-dimensional scenarios is tested to validate the applicability of the method. The potential of the NMRT in fluid mechanics can be further developed, and more work will be done in the future.

Author Contributions: Conceptualization, W.Z. and D.H.; formal analysis, Z.L. and C.Z.; methodology, Z.L. and C.Z.; software, C.Z.; supervision, Z.L., W.Z., and D.H.; validation, Z.L. and C.Z.; writing—original draft, Z.L. and C.Z.; writing—review and editing, Z.L., W.Z., and D.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (Grant No. 42376194).

Data Availability Statement: The experimental data related to this paper can be requested from the authors via email: zxliu@shou.edu.cn.

Acknowledgments: The authors would like to express their gratitude for the support of the Fishery Engineering and Equipment Innovation Team of Shanghai High-level Local University.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. O'Mahony, N.; Campbell, S.; Carvalho, A.; Harapanahalli, S.; Hernandez, G.V.; Krpalkova, L.; Riordan, D.; Walsh, J. Deep learning vs. traditional computer vision. In *Advances in Computer Vision, Proceedings of the 2019 Computer Vision Conference (CVC), Las Vegas, NV, USA, 25–26 April 2019*; Arai, K., Kapoor, S., Eds.; Advances in Intelligent Systems and Computing; Springer: Cham, Switzerland, 2020; Volume 943, pp. 128–144.
2. Chowdhary, K.; Chowdhary, K. Natural language processing. In *Fundamentals of Artificial Intelligence*; Springer: New Delhi, India, 2022; pp. 603–649.
3. Sejnowski, T.J. The unreasonable effectiveness of deep learning in artificial intelligence. *Proc. Natl. Acad. Sci. USA* **2020**, *117*, 30033–30038. [CrossRef]

4. Koppe, G.; Meyer-Lindenberg, A.; Durstewitz, D. Deep learning for small and big data in psychiatry. *Neuropsychopharmacology* **2021**, *46*, 176–190. [CrossRef] [PubMed]
5. Oberkampf, W.L.; Trucano, T.G. Verification and validation in computational fluid dynamics. *Prog. Aerosp. Sci.* **2002**, *38*, 209–272. [CrossRef]
6. Ma, H.; Zhang, Y.x.; Haidn, O.J.; Thuerey, N.; Hu, X.y. Supervised learning mixing characteristics of film cooling in a rocket combustor using convolutional neural networks. *Acta Astronaut.* **2020**, *175*, 11–18. [CrossRef]
7. Raissi, M.; Perdikaris, P.; Karniadakis, G.E. Physics informed deep learning (Part I): Data-driven solutions of nonlinear partial differential equations. *arXiv* **2017**, arXiv:1711.10561.
8. Raissi, M.; Perdikaris, P.; Karniadakis, G.E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.* **2019**, *378*, 686–707. [CrossRef]
9. Guo, Y.; Cao, X.; Liu, B.; Gao, M. Solving partial differential equations using deep learning and physical constraints. *Appl. Sci.* **2020**, *10*, 5917. [CrossRef]
10. Meng, X.; Li, Z.; Zhang, D.; Karniadakis, G.E. PPINN: Parareal physics-informed neural network for time-dependent PDEs. *Comput. Methods Appl. Mech. Eng.* **2020**, *370*, 113250. [CrossRef]
11. Wang, N.; Zhang, D.; Chang, H.; Li, H. Deep learning of subsurface flow via theory-guided neural network. *J. Hydrol.* **2020**, *584*, 124700. [CrossRef]
12. Cho, J.; Nam, S.; Yang, H.; Yun, S.B.; Hong, Y.; Park, E. Separable physics-informed neural networks. *Adv. Neural Inf. Process. Syst.* **2024**, *36*, 23761–23788.
13. Succi, S.; Benzi, R.; Higuera, F. The lattice Boltzmann equation: A new tool for computational fluid-dynamics. *Phys. Nonlinear Phenom.* **1991**, *47*, 219–230. [CrossRef]
14. Li, J. *Multiscale and Multiphysics Flow Simulations of Using the Boltzmann Equation*; Applications to Porous Media and MEMS; Springer: Cham, Switzerland, 2020.
15. Simonis, S. Lattice Boltzmann Methods for Partial Differential Equations. Ph.D. Thesis, Karlsruher Institut für Technologie (KIT), Karlsruhe, Germany, 2023.
16. Xu, K. A generalized Bhatnagar–Gross–Krook model for nonequilibrium flows. *Phys. Fluids* **2008**, *20*, 026101. [CrossRef]
17. Shi, Y.; Shan, X. A multiple-relaxation-time collision model for nonequilibrium flows. *Phys. Fluids* **2021**, *33*, 037134. [CrossRef]
18. Bird, G.A. *Molecular Gas Dynamics and the Direct Simulation of Gas Flows*; Oxford University Press: Oxford, UK, 1994.
19. Liu, C.; Xu, K. A unified gas-kinetic scheme for micro flow simulation based on linearized kinetic equation. *Adv. Aerodyn.* **2020**, *2*, 21. [CrossRef]
20. Gamba, I.M.; Haack, J.R.; Hauck, C.D.; Hu, J. A fast spectral method for the Boltzmann collision operator with general collision kernels. *SIAM J. Sci. Comput.* **2017**, *39*, B658–B674. [CrossRef]
21. Liu, Z.; Chen, Y.; Xiao, W.; Song, W.; Li, Y. Large-Scale Cluster Parallel Strategy for Regularized Lattice Boltzmann Method with Sub-Grid Scale Model in Large Eddy Simulation. *Appl. Sci.* **2023**, *13*, 11078. [CrossRef]
22. Liu, Z.; Liu, H.; Huang, D.; Zhou, L. The Immersed Boundary-Lattice boltzmann method parallel model for fluid-structure interaction on heterogeneous platforms. *Math. Probl. Eng.* **2020**, *2020*, 3913968. [CrossRef]
23. Liu, Z.; Li, S.; Ruan, J.; Zhang, W.; Zhou, L.; Huang, D.; Xu, J. A New Multi-Level Grid Multiple-Relaxation-Time Lattice Boltzmann Method with Spatial Interpolation. *Mathematics* **2023**, *11*, 1089. [CrossRef]
24. Li, Z.; Wang, Y.; Liu, H.; Wang, Z.; Dong, B. Solving Boltzmann equation with neural sparse representation. *arXiv* **2023**, arXiv:2302.09233.
25. Oh, J.; Cho, S.Y.; Yun, S.B.; Park, E.; Hong, Y. Separable Physics-informed Neural Networks for Solving the BGK Model of the Boltzmann Equation. *arXiv* **2024**, arXiv:2403.06342.
26. Lou, Q.; Meng, X.; Karniadakis, G.E. Physics-informed neural networks for solving forward and inverse flow problems via the Boltzmann-BGK formulation. *J. Comput. Phys.* **2021**, *447*, 110676. [CrossRef]
27. Lin, Y.; Hong, N.; Shi, B.; Chai, Z. Multiple-relaxation-time lattice Boltzmann model-based four-level finite-difference scheme for one-dimensional diffusion equations. *Phys. Rev. E* **2021**, *104*, 015312. [CrossRef]
28. Maxwell, J. Illustrations of the dynamical theory of gases. *Philos. Mag.* **1867**, *19*, 19–32. [CrossRef]
29. Choe, Y.S.; Kim, Y.J.; Ri, T.N.; Kim, T.K. One-dimensional lattice Boltzmann simulation of parallel plate dielectric barrier discharge plasma in atmospheric argon. *Math. Comput. Simul.* **2023**, *213*, 115–126. [CrossRef]
30. Ba, Y.; Liu, H.; Li, Q.; Kang, Q.; Sun, J. Multiple-relaxation-time color-gradient lattice Boltzmann model for simulating two-phase flows with high density ratio. *Phys. Rev. E* **2016**, *94*, 023310. [CrossRef]
31. Huang, X.; Liu, H.; Shi, B.; Wang, Z.; Yang, K.; Li, Y.; Weng, B.; Wang, M.; Chu, H.; Zhou, J.; et al. Solving partial differential equations with point source based on physics-informed neural networks. *arXiv* **2021**, arXiv:2111.01394.
32. Zhang, K.; Feng, X.F.; Jing, H.F.; Jiang, Y.L. An improved MRT-LBM and investigation to the transition and periodicity of 2D lid-driven cavity flow with high Reynolds numbers. *Chin. J. Phys.* **2023**, *84*, 51–65. [CrossRef]
33. Sitzmann, V.; Martel, J.; Bergman, A.; Lindell, D.; Wetzstein, G. Implicit neural representations with periodic activation functions. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 7462–7473.
34. Xu, Z.Q.J.; Zhang, Y.; Luo, T.; Xiao, Y.; Ma, Z. Frequency principle: Fourier analysis sheds light on deep neural networks. *arXiv* **2019**, arXiv:1901.06523.

35. Alber, M.; Buganza Tepole, A.; Cannon, W.R.; De, S.; Dura-Bernal, S.; Garikipati, K.; Karniadakis, G.; Lytton, W.W.; Perdikaris, P.; Petzold, L.; et al. Integrating machine learning and multiscale modeling—perspectives, challenges, and opportunities in the biological, biomedical, and behavioral sciences. *NPJ Digit. Med.* **2019**, *2*, 115. [CrossRef]
36. Liu, Z.; Cai, W.; Xu, Z.Q.J. Multi-scale deep neural network (MscaleDNN) for solving Poisson-Boltzmann equation in complex domains. *arXiv* **2020**, arXiv:2007.11207.
37. Edwards, T. Discrete Wavelet Transforms: Theory and Implementation. 1991. Available online: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=f7efbe4055f84612ec0851f6ccd11d2d4999141b (accessed on 11 June 2024).
38. Jin, S.; Shi, Y. A micro-macro decomposition-based asymptotic-preserving scheme for the multispecies Boltzmann equation. *SIAM J. Sci. Comput.* **2010**, *31*, 4580–4606. [CrossRef]
39. Gamba, I.M.; Jin, S.; Liu, L. Micro-macro decomposition based asymptotic-preserving numerical schemes and numerical moments conservation for collisional nonlinear kinetic equations. *J. Comput. Phys.* **2019**, *382*, 264–290. [CrossRef]
40. Shlezinger, N.; Eldar, Y.C.; Boyd, S.P. Model-based deep learning: On the intersection of deep learning and optimization. *IEEE Access* **2022**, *10*, 115384–115398. [CrossRef]
41. Boelens, A.M.; Venturi, D.; Tartakovsky, D.M. Parallel tensor methods for high-dimensional linear PDEs. *J. Comput. Phys.* **2018**, *375*, 519–539. [CrossRef]
42. Reynolds, M.J.; Doostan, A.; Beylkin, G. Randomized alternating least squares for canonical tensor decompositions: Application to a PDE with random data. *SIAM J. Sci. Comput.* **2016**, *38*, A2634–A2664. [CrossRef]
43. Lange, K. Singular value decomposition. In *Numerical Analysis for Statisticians*; Statistics and Computing; Springer: New York, NY, USA, 2010; pp. 129–142.
44. Evert, E.; Vandecappelle, M.; De Lathauwer, L. Canonical Polyadic Decomposition via the generalized Schur decomposition. *IEEE Signal Process. Lett.* **2022**, *29*, 937–941. [CrossRef]
45. Peng, W.; Zhou, W.; Zhang, J.; Yao, W. Accelerating physics-informed neural network training with prior dictionaries. *arXiv* **2020**, arXiv:2004.08151.
46. Loshchilov, I.; Hutter, F. Sgdr: Stochastic gradient descent with warm restarts. *arXiv* **2016**, arXiv:1608.03983.
47. Adeyemo, O.D.; Motsepa, T.; Khalique, C.M. A study of the generalized nonlinear advection-diffusion equation arising in engineering sciences. *Alex. Eng. J.* **2022**, *61*, 185–194. [CrossRef]
48. Maragkos, G.; Beji, T. Review of convective heat transfer modelling in cfd simulations of fire-driven flows. *Appl. Sci.* **2021**, *11*, 5240. [CrossRef]
49. Wu, L.; Reese, J.M.; Zhang, Y. Solving the Boltzmann equation deterministically by the fast spectral method: Application to gas microflows. *J. Fluid Mech.* **2014**, *746*, 53–84. [CrossRef]
50. Schäfer, V. Generalization of Physics-Informed Neural Networks for Various Boundary and Initial Conditions. Ph.D. Thesis, Technische Universität Kaiserslautern, Kaiserslautern, Germany, 2022.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.