AIAA AVIATION Forum
August 2-6, 2021, VIRTUAL EVENT
AIAA AVIATION 2021 FORUM

10.2514/6.2021-2895

A structure-preserving surrogate model for the closure of the moment system of the Boltzmann equation using convex deep neural networks

Steffen Schotthöfer*, Tianbai Xiao†, Martin Frank‡
Computational Science and Mathematical Methods Division,
Department of Mathematics, KIT Karlsruhe,
Bldg 20.30, Englerstraße 2, 76131 Karlsruhe, Germany

Cory D. Hauck§
Computer Science and Mathematics Division,
Oak Ridge National Laboratory, and Department of Mathematics (Joint Faculty), University of Tennessee,
1 Bethel Valley Road, Bldg. 4100, Oak Ridge, TN 37831 USA

Direct simulation of physical processes on a kinetic level is prohibitively expensive in aerospace applications due to the extremely high dimension of the solution spaces. In this paper, we consider the moment system of the Boltzmann equation, which projects the kinetic physics onto the hydrodynamic scale. The unclosed moment system can be solved in conjunction with the entropy closure strategy. Using an entropy closure provides structural benefits to the physical system of partial differential equations. Usually computing such closure of the system spends the majority of the total computational cost, since one needs to solve an ill-conditioned constrained optimization problem. Therefore, we build a neural network surrogate model to close the moment system, which preserves the structural properties of the system by design, but reduces the computational cost significantly. Numerical experiments are conducted to illustrate the performance of the current method in comparison to the traditional closure.

I. Introduction

In rarefied gas dynamics, the continuum assumption is no longer applicable and one has to rely on a more general description of the physical system at hand, which is given by kinetic equations such as the Boltzmann equation [1]. These equations arise in a variety of other applications as well, such as neutron transport [2], radiative transport [3] and semiconductors [4]. The Boltzmann equation is a high dimensional integro-differential equation, with phase space dependency on time, space and particle velocity. This high dimensionality of the phase space presents a severe computational challenge for massive numerical simulations.

Several methods for phase space reduction have been proposed to solve the Boltzman equation, including the discrete ordinate/velocity methods [2, 5, 6] and moment methods [7–11]. Discrete ordinate methods evaluate the velocity space at specific points, which yields a system of equations only coupled by the integral scattering operator. While computationally efficient, these methods suffers from numerical artifacts, which are called ray effects [5]. Moment methods eliminate the dependency of the phase space on the velocity variable by computing the moment hierarchy of the Boltzmann equation. Due to the structure of the advection term, the resulting moment system is unclosed. One distinguishes moment methods according to the modelling of their closure. The classical P_N closure uses a simple truncation that results in a system of linear hyperbolic equations. The main drawback of this method is its numerical artifacts, specifically large oscillations of the particle density, which may result in negative particle concentrations. This effect is particularly present in the streaming particle regime [12].

A moment closure, which preserves important physical and mathematical properties [13] of the Boltzmann equation, is constructed by solving a convex constrained optimization problem, which is based on the entropy minimization

*Ph.D Candidate, steffen.schotthoefer@kit.edu
†Postdoc, tianbai.xiao@kit.edu
‡Professor, martin.frank@kit.edu
§Professor, hauckc@ornl.gov

principle [8, 11].The method, which is commonly referred to as M_N closure, is accurate in the diffusive limit [14] and unlike the P_N closure, this method is also accurate in the streaming limit [15]. Although the M_N closure is methodically superior to the P_N closure, it is by far more expensive to compute. Garret et al. [9] have demonstrated, that in a high performance implementation, more than 80% of the computational time of the whole solver is required for the solution of the entropy minimization problem. This motivates the development of a neural network surrogate model to accelerate the M_N closure.

Several machine learning inspired methods have been proposed recently. The authors of [16] close the moment system by learning the gradient of the highest order moment. In [17], the authors pursue two strategies. First, they use a encoder-decoder network to generate generalized moments and then learn the moment closure of system with its dynamics in mind. Second, they learn directly the correction term to the Euler equations. In [18], Galilean invariant machine learning methods for partial differential equations are developed using the conservation dissipation formalism. Using convolutional networks, a closure for the one dimensional Euler-Poisson system was constructed in [19]. In [20], a dense neural network was used to model the deviation from the Maxwellian in the collision term of the Boltzmann equation. The authors of [21] use neural networks to reproduce physical properties of known magnetized plasma closures. In [22], fully connected, dense and discrete Fourier transform networks are used to learn the Hammett-Perkins Landau fluid closure. Physics informed neural networks were employed to solve forward and inverse problems via the Boltzmann-BGK formulation to model flows in continuum and rarefied regimes in [23], to solve the radiative transfer equation [24] and the phonon Boltzmann equation in [25]. In [26], the authors propose a data driven surrogate model of the minimal entropy closure using convex splines and empirically convex neural networks. To ensure convexity at the training data points, the authors penalize a non symmetric positive definite Hessian of the network output.

The goal of this publication is to construct a deep neural network surrogate to model the entropy closure of the moment system of the Boltzmann Equation. The neural network and maps a given moment to its minimal mathematical entropy. In contrast to the work proposed in [26], the neural network is input convex by design using the methods of Amos et al. [27]. By this ansatz, the learned closure automatically inherits all structural properties of the entropy closure for the moment system. The derivative of the network with respect to the moments maps to the corresponding optimal Lagrange multipliers of the entropy minimization problem. We train the neural network on the output, the Lagrange multiplier and additionally on the reconstructed moments, whereas in [26], the authors train on the output, the reconstructed moments and the Hessian of the network.

The remainder of this paper is structured as follows. In Section II, we briefly present the Boltzmann equation and its properties. We give a review to the moment method and the minimal entropy closure. In Section III, we present the input convex neural network approach. Then we construct the neural network closure as a surrogate model to the minimal entropy closure. We show that the neural entropy closure has an entropy entropy-flux pair, fulfills a local dissipation law, is hyperbolicity preserving and invariant in range. Furthermore, it fulfills the H-theorem and the moment system conserves mass. We discuss the generation of training data and propose two algorithms for data generation. Lastly we describe the training and inference of the neural network on dimension reduced input data. Section IV describes the kinetic scheme for solving the moment system and its interplay with the neural network surrogate model. Section V contains a selection of $1D$ and $2D$ numerical test cases, where we demonstrate our findings.

II. Kinetic equations and the moment method

Classical kinetic theory is profoundly built upon the Boltzmann equation, which describes the space-time evolution of the one-particle kinetic density function $f(t, \mathbf{x}, \mathbf{v})$ with $t > 0$, $x \in \mathbf{X} \subset \mathbb{R}^3$, $\mathbf{v} \in \mathbf{V} \subset \mathbb{R}^3$ in a many-particle system,

$$\partial_t f + \mathbf{v} \cdot \nabla_{\mathbf{x}} f = Q(f). \tag{1}$$

The left-hand side of the equation describes particle transport, while the right-hand side models collisions. If the particles only collide with a background material one can model this behavior with the linear Boltzmann collision operator

$$Q(f)(\mathbf{v}) = \int_{\mathbf{V}} \mathcal{B}(\mathbf{v}_*, \mathbf{v}) \left[f(\mathbf{v}_*) - f(\mathbf{v}) \right] d\mathbf{v}_*, \tag{2}$$

where the collision kernel $\mathcal{B}(\mathbf{v}_*, \mathbf{v})$ models the strength of collisions at different velocities. If the interactions among particles are considered, the collision operator becomes nonlinear. For example, the two-body collision results in

$$Q(f) = \int_{\mathbf{V}} \int_{\mathcal{S}^2} \mathcal{B}(\cos\beta, |\mathbf{v} - \mathbf{v}_*|) \left[f(\mathbf{v}') f(\mathbf{v}'_*) - f(\mathbf{v}) f(\mathbf{v}_*) \right] d\mathbf{\Omega} d\mathbf{v}_*, \tag{3}$$

where $\{\mathbf{v}, \mathbf{v}_*\}$ are the pre-collision velocities of two colliding particles, and $\{\mathbf{v}', \mathbf{v}_*'\}$ are the corresponding post-collision velocities and $\mathcal{S}^2$ is the unit sphere. The right-hand side is a fivefold integral, where β is the so-called deflection angle.

In the following, we use the notation

$$\langle \cdot \rangle = \int_{\mathbf{v}} \cdot \, \mathrm{d}\mathbf{v} \tag{4}$$

to define integrals over velocity space.

The Boltzmann equation is a first-principles model based on direct modeling. It possesses some key structural properties, which are intricately related to the physical processes and its mathematical existence and uniqueness theory. We briefly review some of these properties, where we follow [7, 13]. First, the time evolution of the solution is invariant in range, i.e. if $f(0, \mathbf{x}, \mathbf{v}) \in B \subset [0, \infty)$, then $f(t, \mathbf{x}, \mathbf{v}) \in B \subset [0, \infty)$ for all $t > 0$. Particularly this implies non-negativity of f. Second, if ϕ is a collision invariant fulfilling

$$\langle \phi Q(g) \rangle = 0, \quad \forall g \in \mathrm{Dom}(Q), \tag{5}$$

the equation

$$\partial_t \langle \phi f \rangle + \nabla_{\mathbf{x}} \cdot \langle \mathbf{v} \phi f \rangle = 0 \tag{6}$$

is a local conservation law. Third, for each fixed direction $\mathbf{v}$, the advection operator, i.e. the left-hand side term of Eq. (1), is hyperbolic in space and time. Forth, let $D \subset \mathbb{R}$. There is a twice continuously differentiable, strictly convex function $\eta : D \to \mathbb{R}$, which is called kinetic entropy density. It has the property

$$\langle \eta'(g) Q(g) \rangle \leq 0, \quad \forall g \in \mathrm{Dom}(Q) \text{ s.t. } \mathrm{Im}(g) \subset D. \tag{7}$$

Applied to Eq. (1), we get the local entropy dissipation law

$$\partial_t \langle \eta(f) \rangle + \nabla_{\mathbf{x}} \cdot \langle \mathbf{v} \eta(f) \rangle \leq 0. \tag{8}$$

Usually we set $D = B$. Lastly, the solution f fulfills the H-theorem.

The Boltzmann equation is an integro-differential equation model defined on a seven-dimensional phase space. With the nonlinear five-fold integral, it is challenging to solve accurately and efficiently. The well known moment methods encode the velocity dependence of the Boltzmann equation in a moment vector $u \in \mathbb{R}^{N+1}$ of order N [7, 13]. The moments are calculated with respect to a vector of basis functions $m(\mathbf{v}) \in \mathbb{R}^{N+1}$ by integrating over the velocity space,

$$u(t, \mathbf{x}) = \langle m f \rangle . \tag{9}$$

Common choices for the basis functions are monomials and spherical harmonics, depending on the application. Typically, they include the collision invariants defined in Eq. (5). The moment vector satisfies the system of transport equations

$$\partial_t u(t, \mathbf{x}) + \nabla_{\mathbf{x}} \cdot \langle \mathbf{v} m(\mathbf{v}) f \rangle = \langle m(\mathbf{v}) Q(f) \rangle , \tag{10}$$

which is called moment system. By construction, the advection operator depends on f and thus, the moment system is unclosed. Moment methods aim to find a meaningful closure for this system. Since the kinetic equation dissipates entropy and fulfill a local entropy dissipation law, one can close the system by choosing the reconstructed kinetic density $\tilde{f}$ out of all possible functions g that fulfill $u(t, \mathbf{x}) = \langle mg \rangle$ as the one with minimal entropy h. The minimal entropy closure can be formulated as a constrained optimization problem for a given set of moments u.

$$\begin{aligned} h(u) = \langle \eta(f_u) \rangle &= \min_{g \in F} \langle \eta(g) \rangle \\ \text{s.t. } u &= \langle mg \rangle \end{aligned} \tag{11}$$

where $F = \{g : \mathrm{Range}(g) \subset D\}$. The set $\mathcal{R}$ of all moment vectors u for which a solution exists is called the set of realizable moments. These are all moments that belong to a kinetic density function f that fulfills the invariant range property of the kinetic equation. This condition is enforced by the constraint of the optimization problem of Eq. (11), which is called realizability constraint. For all moment vectors u for which a solution exists, this solution is unique and of the form

$$f_u = \eta_*'(\alpha(u) \cdot m), \tag{12}$$

where the Lagrange multiplier $\alpha : \mathbb{R}^{N+1} \to \mathbb{R}^{N+1}$ maps u to the solution of the dual problem

$$\alpha_u(u) = \underset{\alpha \in \mathbb{R}^{N+1}}{\operatorname{argmax}} \{\alpha \cdot u - \langle \eta_*(\alpha \cdot m) \rangle\} \tag{13}$$

and η_* is the Legendre dual of η. Inserting f_u into the moment system (10) gives a closed system of equations in space and time. The function

$$h(u) = \alpha_u \cdot u - \langle \eta_*(\alpha_u \cdot m) \rangle \tag{14}$$

is convex in u and a suitable entropy functional for the moment system in Eq. (10), see [7] . Furthermore, the derivative of h recovers the optimal Lagrange multipliers of Eq. (13),

$$\frac{\mathrm{d}}{\mathrm{d}u} h = \alpha_u \tag{15}$$

This minimal entropy closure also conserves the above listed structural properties of the Boltzmann equation . We present the above properties for the moment system for the sake of completeness, where we follow [7, 11]. First, the invariant range property of the solution f translates to the set of realizable moments $\mathcal{R}$, which can be described as the set of moments corresponding to a kinetic density with range in B. One demands that $u(t, \mathbf{x}, \mathbf{v}) \in \mathcal{R}$ for all $t > 0$. Second, if a moment basis function $m_i(\mathbf{v})$ is a collision invariant, then

$$\partial_t u(t, x) + \nabla_{\mathbf{x}} \cdot \langle \mathbf{v} m f_u \rangle = 0, \tag{16}$$

is a local conservation law. Third, one can write Eq. (10) as a symmetric hyperbolic conservation law in α_u. Forth, for $u \in \mathcal{R}$, $h(u)$ and $j(u) = \langle \mathbf{v} \eta(f_u) \rangle$ is a suitable entropy and entropy-flux pair compatible with the advection operator $\langle \mathbf{v} m f_u \rangle$ and yield a semi-discrete version of the entropy dissipation law.

$$\partial_t h(u) + \nabla_{\mathbf{x}} j(u) = h'(u) \cdot \langle m Q(f_u(\alpha_u)) \rangle \leq 0 \tag{17}$$

Note that convexity of $h(u)$ is crucial for the entropy dissipation property. Lastly, the moment system fulfills the H-theorem.

A numerical method to solve the moment system therefore consists of an iterative discretization scheme for the moment system (10) and a Newton optimizer for the dual minimal entropy optimization problem in Eq. (13). The former scheme can be a finite volume or discontinuous Garlerkin scheme, for example. The drawback of the method is the high computational cost associated with the Newton solver. The optimization problem in Eq. (13) needs to be solved in every grid cell at every time step of the PDE solver. The computational effort to solve the minimal entropy optimization problem grows over-proportionate with the dimension N of the moment basis m. Using three basis functions, the optimizer requires 80% of the computation time and 87% when using seven basis functions, as Garrett et al. have demonstrated in a computational study [10]. Furthermore, the optimization problem is ill-conditioned, if the moments u are near the boundary of the realizable set $\mathcal{R}$ [8]. At the boundary $\partial\mathcal{R}$, the Hessian of the objective function becomes singular and the kinetic density f_u is a sum of delta functions [28].

III. Neural entropy closures

In the following, we propose a neural network to solve the minimal entropy closure problem of Eq. (13), which preserves the intricate structure of the moment system and the Boltzmann equation. The advantage of a neural network compared to a Newton optimizer is a significantly lower computational expense. Instead of inverting a possibly near singular Hessian multiple times, a trained neural network only needs a comparatively small amount of fast tensor operations to compute the closure.

A. Design of the neural network closure

A neural network $\mathcal{N}_\theta : \mathbb{R}^n \mapsto \mathbb{R}^m$ is a parameterized mapping from an input x to the network prediction $\mathcal{N}_\theta(x)$. In this work, we focus on dense neural networks. A multi-layer neural network $\mathcal{N}_\theta$ is a concatenation of non-linear (activation) functions f_k applied to weighted sums of the previous layer's output z_{k-1}. An M layer network can be described in tensor formulation as follows.

$$z_k = f_k(W_k z_{k-1} + b_k), \qquad k = 1, \ldots, M \tag{18}$$

$$x = z_0, \tag{19}$$

$$\mathcal{N}_\theta(x) = z_M \tag{20}$$

where W_k is the weight matrix of layer k and b_k the corresponding bias vector. In the following, we denote the set of all trainable parameters of the network, i.e. weights and biases by θ. With the correct trainable parameters, a neural network is capable of approximating any given target function with arbitrary accuracy [29], however it is not trivial how to choose them. Usually, one chooses a set of training data points $T = \left\{(x_j, y_j)\right\}_{j\in J}$ to input in a loss function, for example the mean squared error

$$L(x, y; \theta) = \frac{1}{|J|} \sum_{j\in J} \left\| y_j - \mathcal{N}_\theta(x_j) \right\|_2^2 . \tag{21}$$

Then one can set up the process of finding suitable weights, called training of the network, as an optimization problem

$$\min_\theta L(x, y; \theta) \tag{22}$$

The optimization is often carried out with gradient-based algorithms, such as stochastic gradient descent [30] or related methods as ADAM [31], which we use in this work.

The proposed neural network based closure for the minimal entropy problem, called neural entropy closure, is a network that maps a realizable moment vector $u \in \mathcal{R}$ to the entropy functional $h(u)$ in Eq. (14), which is the objective functional of the entropy optimization problem in Eq. (11). Assuming the neural network is trained, we have the following relations,

$$h_\theta := \mathcal{N}_\theta \approx h(u), \tag{23}$$

$$\alpha_\theta := \frac{\mathrm{d}}{\mathrm{d}u}\mathcal{N}_\theta \approx \frac{\mathrm{d}}{\mathrm{d}u} h = \alpha_u, \tag{24}$$

$$f_\theta := \eta_*'\left(\frac{\mathrm{d}}{\mathrm{d}u}\mathcal{N}_\theta \cdot m\right) \approx \eta_*'(\alpha_u \cdot m) = f_u, \tag{25}$$

$$u_\theta := \left\langle m\eta_*'\left(\frac{\mathrm{d}}{\mathrm{d}u}\mathcal{N}_\theta \cdot m\right)\right\rangle \approx \left\langle m\eta_*'(\alpha_u \cdot m)\right\rangle = u, \tag{26}$$

by using Eq. (12), Eq. (15) and the definition of the moment. These relations give a set of possible loss functions for the network training and a workflow for the neural network accelerated kinetic solver. We choose h, α_u and u as network outputs, since f_u is dependent on the velocity variable $\mathbf{v}$. The loss function of the network then becomes

$$L(u, \alpha_u, h; \theta) = \frac{1}{|J|} \sum_{j\in J} \left\| h(u_j) - \mathcal{N}_\theta(u_j) \right\|_2^2 + \left\| \alpha_u(u_j) - \alpha_\theta(u_j) \right\|_2^2 + \left\| u_j - u_\theta(u_j) \right\|_2^2 . \tag{27}$$

We demand that the neural entropy closure preserves the structure of the moment system of the Boltzmann equation, which is described in Section II. The invariant range property of f_θ depends solely on the range of η_*'. Popular choices for the kinetic entropy density η are Maxwell-Boltzmann, Bose Einstein or Fermi-Dirac. In this work, we focus on the Maxwell-Boltzmann entropy, which has the following definition, Legendre dual and derivative.

$$\eta(z) = z\ln(z) - z, \qquad z \in D = \mathbb{R}_+ \tag{28}$$

$$\eta'(z) = \ln(z), \qquad z \in D = \mathbb{R}_+ \tag{29}$$

$$\eta_*(y) = \exp(y), \qquad y \in \mathbb{R} \tag{30}$$

$$\eta_*'(y) = \exp(y), \qquad y \in \mathbb{R} \tag{31}$$

By construction, the neural entropy closure is of invariant range, since $f_\theta(v) = \exp(\frac{\mathrm{d}}{\mathrm{d}u}\mathcal{N}_\theta \cdot m(v)) > 0$. Interchanging the entropy functional by a neural network does not affect the conservation property of the moment system. Consider the hyperbolicity requirement. In order to define the Legendre dual of h, it must be convex. In the proof of the hyperbolicity property, which is conducted in [13] for α_u and u as the system variable, h'' (respectively h_*'') must be symmetric positive definite. As a consequence, $h(u)$ and therefore the neural network $N_\theta(u)$ must be strictly convex in u. Strict convexity of the entropy functional h is the crucial requirement for the related properties entropy dissipation and the H-theorem [13].

Convex neural networks have been inspected in [27], where the authors propose several deep neural networks that are strictly convex with respect to the input by design. The design is led by the following principles [32]. First, a positive

sum of convex functions is convex. Second, let $f : \mathbb{R}^n \to \mathbb{R}$ be the concatenation of the functions $h : \mathbb{R}^k \to \mathbb{R}$ and $g : \mathbb{R}^n \to \mathbb{R}^k$. Then $f(x) = h(g(x))$ is convex, if h is convex, h is non-decreasing in each argument and all $g_{i=1,\dots,k}$ are convex. Applying these conditions to the definition of a layer of a neural network, Eq. (18), we get that all entries of the weight matrix W_k must be positive in all layers except the first. Furthermore, the activation function of each layer must be strictly convex. However, in practice it turns out that very deep networks with positive weights have difficulties to train. The authors therefore modifies the definition of a hidden layer in Eq. (18) to

$$z_k = f_k(W_k^z z_{k-1} + W_k^x x + b_k^z), \qquad k = 2, \dots, M, \tag{32}$$

$$z_k = f_k(W_k^x x + b_k^z), \qquad k = 1, \tag{33}$$

where W_k^z must be non-negative, and W_k^x may attain arbitrary values. This approach is related to the idea of residual neural networks [33]. We choose the strictly convex softplus function $\ln(1 + e^x)$ as layer activation function for $f_k, k = 1, \dots, M-1$ and a linear activation for the last layer, since we are dealing with a regression task.

B. Training Data

The inputs to the neural entropy closure are realizable moments $u \in \mathcal{R}$ and it is trained on the corresponding Lagrange multipliers $\alpha \in \mathbb{R}^{N+1}$ and the values of the entropy functional $h(u, \alpha)$ defined in Eq. (13). The last output of the network is u itself, where is measured, how well the network can reconstruct the input moment, see Eq. (26). The training datset therefore consists of triplets $(h(u, \alpha_u, \alpha_u, u)$, which have the relations described in Eq. (13) and Eq. (14). Unfortunately, $\mathcal{R}$ is unbounded, since $u_0 = \langle f \rangle \geq 0$, if $f \geq 0$ and thus all non-negative moments of order zero are realizable. A strategy to characterize the realizable set, is to consider normalized moments [34]

$$\overline{u} = \frac{u}{u_0} = [1, \frac{u_1}{u_0}, \dots, \frac{u_N}{u_0}]^T \in \mathbb{R}^{N+1}, \tag{34}$$

$$\overline{u}^r = [\overline{u}_1^r, \dots, \overline{u}_N^r]^T \in \mathbb{R}^N, \tag{35}$$

where we call $\overline{u}$ the normalized moments and $\overline{u}_r$ the reduced normalized moments. Analogously, we define the set of normalized realizable moments $\overline{\mathcal{R}}$ and the reduced normalized realizable moments as $\overline{\mathcal{R}}^r$

$$\overline{\mathcal{R}} = \{u \in \mathcal{R} : u_0 = 1\} \subset \mathbb{R}^{N+1}, \tag{36}$$

$$\overline{\mathcal{R}}^r = \left\{\overline{u}^r \in \mathbb{R}^N [1, \overline{u}^T]^T \in \overline{\mathcal{R}}\right\} \subset \mathbb{R}^N. \tag{37}$$

Both, $\overline{\mathcal{R}}$ and $\overline{\mathcal{R}}^r$ are bounded and convex [34, 35]. In [34], the authors have derived expressions for $\overline{\mathcal{R}}^r$ up to moment order $N = 4$ and one spatial dimension, i.e. $V, X \subset \mathbb{R}^1$ for monomial basis functions.

$$1 \geq \overline{u}_1^r \geq -1 \tag{38a}$$

$$1 \geq \overline{u}_2^r \geq (\overline{u}_1^r)^2 \tag{38b}$$

$$\overline{u}_2^r - \frac{(\overline{u}_1^r - \overline{u}_2^r)^2}{1 - \overline{u}_1^r} \geq \overline{u}_3^r \geq -\overline{u}_2^r + \frac{(\overline{u}_1^r + \overline{u}_2^r)^2}{1 + \overline{u}_1^r} \tag{38c}$$

$$\frac{(\overline{u}_2^r)^3 - (\overline{u}_3^r)^2 + 2\overline{u}_1^r \overline{u}_2^r \overline{u}_3^r}{\overline{u}_2^r - (\overline{u}_1^r)^2} \geq \overline{u}_4^r \geq \overline{u}_2^r - \frac{(\overline{u}_1^r - \overline{u}_3^r)^2}{(1 - \overline{u}_2^r)} \tag{38d}$$

where equality indicates the boundary $\partial\overline{\mathcal{R}}^r$. Here, the basis functions are monomials. The realizable set for higher order moments can be characterized using the more general results in [28]. This characterization enables a sampling strategy for training data, where one first samples uniformly in $\overline{\mathcal{R}}^r$ and afterwards solves the dual entropy minimization problem in Eq. (13) using $\overline{u}$, to compute α_u and $h(u, \alpha_u)$, see Algorithm 1. The algorithm has some drawbacks. First, we do not solve Eq. (13) exactly, but numerically with a tolerance τ. As a result, the neural network accuracy is bounded from below by τ. In practise however, a Newton optimizer is more accurate than the training accuracy of a neural network. Second, the optimization problem Eq. (13) becomes ill-conditioned near the boundary $\partial\overline{\mathcal{R}}^r$ and singular at $\partial\overline{\mathcal{R}}^r$, so we can not generate training data at the boundary using this approach. Lastly, necessary and sufficient conditions for $\partial\overline{\mathcal{R}}^r$ are only derived for $\mathbf{V}, \mathbf{X} \subset \mathbb{R}^1$. Necessary conditions for $\mathbf{V}, \mathbf{X} \subset \mathbb{R}^d$, $d = 2, 3$ are stated in [35] for $N \leq 2$. In 1D M_1, they simplify to

$$\|\overline{u}^r\| \leq 1. \tag{39}$$

Algorithm 1: Data generation with uniform sampling of $\overline{\mathcal{R}}$

Result: $\left\{(\overline{u}_j, \alpha_j, h_j)\right\}_{j\in J}$
Set N, $|J|$;
Set the training accuracy $\tau > 0$;
Set the boundary distance $\delta > 0$;
for $j = 1, \ldots, |J|$ **do**
  Sample $\overline{u}_j$ using Eq. (38) such that $\text{dist}(u_j, \partial\overline{\mathcal{R}}) > \delta$;
  Solve Eq. (13) for $\alpha_{u,j}$ using a Newton optimizer with tolerance τ;
  Compute h_j using Eq. (14);
end

Algorithm 2: Data generation with uniform sampling of Lagrange multipliers

Result: $\left\{(\overline{u}_j, \alpha_j, h_j)\right\}_{j\in J}$
Set N, $|J|$;
Determine $A \subset \mathbb{R}^N$;
for $j = 1, \ldots, |J|$ **do**
  Sample α_j^r such that $\alpha \in A$;
  Compute $\alpha_{0,j}$ using Eq. (41);
  Set $\alpha_j = [\alpha_{0,j}, \alpha_j^{rT}]$;
  Compute $\overline{u_j}$ using Eq. (12);
  Compute h_j using Eq. (14);
end

A different strategy for generating training data is to sample the Lagrange multipliers α_u and then reconstruct u with Eq. (12) and Eq. (9). In order to generate normalized moments, we have to construct the image of $\overline{\mathcal{R}}$ which is a N dimensional hyperplane. For a monomial basis and the Maxwell-Boltzmann entropy, we get the dependent Lagrange multiplier by the definition of the moment of order zero.

$$1 = \left\langle m_0 \eta_*'(\alpha^T m)\right\rangle = \langle \exp(\alpha \cdot m)\rangle = \langle \exp(\alpha^r \cdot m^r)\exp(\alpha_0)\rangle\,, \tag{40}$$

which we can transform to

$$\alpha_0 = -\ln(\langle \exp(\alpha^r \cdot m^r)\rangle), \tag{41}$$

where $\alpha^r = [\alpha_1, \ldots, \alpha_N]^T$ and $m^r = [m_1, \ldots, m_N]^T$. For numerical stability, one needs to restrict the sampled α^r to a bounded set $A \subset \mathbb{R}^N$, since as u approaches $\partial\mathcal{R}$, $\|\alpha\| \to \infty$. Algorithm 2 summarizes this approach. Computational resources for both sampling algorithms can be found in the open source framework KiT-RT, see [36].

C. Inference and training of the neural entropy closure

In Section III.B, we have discussed a normalization and a dimension reduction of the input data, which enables the neural network to train and execute on a bounded and convex set of possible inputs and therefore drastically increases its reliability. Each unseen data point is in the convex hull of seen data points. Normalization of $\mathcal{R}$ allows for a dimension reduction, thus the neural entropy closure takes normalized reduced moments $\overline{u}^r$ as . One can think of the constant zero order moment being encrypted in the bias term of the first layer. The primary output $\mathcal{N}_\theta(\overline{u}^r)$ is trained on $h(\overline{u}, \alpha)$, but the input derivative $\partial_{\overline{u}^r}$ now only reveals α_θ^r. The missing value of α_0 can be computed using Eq. (41). The third network output, $\overline{u}_\theta$ is computed using Eq. (41), Eq. (12) and Eq. (9). In order to train the network on α, respectively α^r, we compute the sensitivity of the input derivative of the neural network, which can be thought of training in Sobolev norm. This concept has been investigated in [37], where the authors find that training in Sobolev norm increases training accuracy and data efficiency.

After training, the neural network has the task to compute the minimal entropy closure for $u \in \mathcal{R}$. Inference of a normalized moment $\overline{u}$ with $\mathcal{N}_\theta$ produces $\alpha_\theta(\overline{u})$. By considering again the definition of moment zero

$$1 = \langle m \exp(\alpha_\theta(\overline{u}))\rangle \tag{42}$$

$$\iff u_0 = \langle m \exp(\ln(u_0)\exp(\alpha_\theta(\overline{u}))\rangle \tag{43}$$

we can deduce for $u_0 > 0$

$$\alpha_\theta(u) = [\alpha_\theta(\overline{u})_0 + \ln(u_0), \alpha_\theta(\overline{u})_1, \ldots, \alpha_\theta(\overline{u})_N]^T. \tag{44}$$

Then $\alpha_\theta(u)$ minimizes the entropy functional for u, if and only if $\alpha_\theta(\overline{u})$ minimizes the entropy functional for $\overline{u}$.
Note, that solving the normalized minimal entropy closure, respectively inference of the network with the normalized

moments can be interpreted as replacing the kinetic entropy density η by the normalized kinetic entropy density

$$\overline{\eta}(f) = \eta\left(\frac{f}{\langle f\rangle}\right) \tag{45}$$

which is still a strictly convex function and therefore fulfills the hyperbolicity, entropy-dissipation and H-theorem properties.

IV. Kinetic Scheme for the Moment Method

In this section, we briefly describe the kinetic scheme that is used to solve the moment method described in Section II. The idea of a kinetic scheme [8–10] is to combine a numerical scheme for hyperbolic conservation laws with an according velocity space discretization.

A Quadrature approximation of the velocity integral is needed in several parts of the kinetic scheme, namely the computation of the numerical flux and the collision operator Q.

$$\langle f(\mathbf{v})\rangle \approx \sum_{q=1}^{n_q} w_q f(\mathbf{v_q}), \tag{46}$$

where w_q are the quadrature weights and v_q the corresponding quadrature points. Depending on the dimension d of $\mathbf{V}$, we use different parametrizations and quadrature rules [38, 39], see Table 1.

Table 1 Parametrization of V and used quadrature rule

d	$\mathbf{V}$	parametrization of $\mathbf{v}$	Quadrature Rule
1	$[-1,1]$	$\mathbf{v} = \mu$	Legendre
2	$[-1,1]\times[0,2\pi)$	$\mathbf{v} = \begin{pmatrix} \sqrt{1-\mu}\cos(\phi) \\ \sqrt{1-\mu}\sin(\phi) \end{pmatrix}$	Tensorized Gauss-Legendre, projected
2	$[-1,1]\times[0,2\pi)$	$\mathbf{v} = \begin{pmatrix} \sqrt{1-\mu}\cos(\phi) \\ \sqrt{1-\mu}\sin(\phi) \\ \mu \end{pmatrix}$	Tensorized Gauss-Legendre

The spatial discretization is implemented with a finite volume method using a quadrilateral grid in the 2 dimensional test cases. The notification is similar for the 1 dimensional test case. We consider the flux function

$$F(f,\mathbf{v}) = \nabla_{\mathbf{x}} \cdot \langle \mathbf{v} m(\mathbf{v}) f_u\rangle \tag{47}$$

We denote the moment vector averaged over control volume i as u^i and average Eq. (1) over the area A_i of the control volume. The flux function $F(f,v)$ is approximated using an out of the box numerical flux $F_n(f_{u,i}, f_{u,j}, \mathbf{v})$ at the face of control volume i and its neighbor j. This yields the kinetic flux $G(i,\mathbf{v})$ at control volume i

$$G(i,\mathbf{v}) = \frac{1}{A_i}\left\langle \sum_{j\in\mathcal{N}(i)} F_{up}(i,j,\mathbf{v}) l_{i,j}\right\rangle, \tag{48}$$

where $\mathcal{N}(i)$ are all neighboring grid cells of control volume i and $l_{i,j}$ is the length of the face of control volume i and its neighbor j. Finally, we approximate the integral with a suitable integration rule and get

$$G(i) = \frac{1}{A_i}\sum_{q=1}^{n_q} w_q \sum_{j\in\mathcal{N}(i)} F_{up}(i,j,\mathbf{v}_q) l_{i,j}. \tag{49}$$

This means, that we have to evaluate the numerical flux for each quadrature point and for each equation in the moment system, which is quite costly but easy to parallelize.

We discretize the moments of the collision operator similarly. Note that the collision integral simplifies, if the chosen moments are collision invariants. The collision operator $Q(f_{u,i})$, respectively its moments $\left\langle m(v)Q(f_{u,i})\right\rangle$ at control volume i can be computed using the above described quadrature rule

$$Q(f_{u,i}(\mathbf{v})) \approx Q_n(f_{u,i}(\mathbf{v})) = \sum_{q=1}^{n_q} w_q \mathcal{B}(\mathbf{v}_{*q}, \mathbf{v}) \left[f_{u,i}(\mathbf{v}_{*q}) - f_{u,i}(\mathbf{v})\right], \tag{50}$$

and

$$\frac{1}{A_i}\left\langle m(\mathbf{v})Q(f_{u,i})\right\rangle \approx D(i) = \frac{1}{A_i}\sum_{q=1}^{n_q} w_q m(\mathbf{v}_q) Q_n(f_{u,i}(\mathbf{v}_q)), \tag{51}$$

which we denote by $D(i)$. If there is a source term in addition to the collision term, it can be discretized analogously. The temporal discretization can be done by any time-stepping scheme for hyperbolic partial differential equation. We consider the spatial discretization of the kinetic equation as a system of ordinary differential equations

$$\partial_t u^i = -G(i;t) + D(i;t), \tag{52}$$

which is then discretized by an explicit Euler method with step size Δt

$$u^{t+\Delta t,i} = u^{t,i} + \Delta t \left(D(i;t) - G(i;t)\right). \tag{53}$$

Recall, that we need to compute f_u using either a numerical Newton solver for the optimization problem in Eq (13) or by evaluating the neural entropy closure N_θ. Either way, we introduce an error and thus the reconstructed kinetic density f_{u^t} does not match exactly the moments u^t. As a consequence, the updated moment of the next time step $u^{t+\Delta t}$ is not necessarily realizable [8, 10, 40]. In [8], the authors propose a time-step restriction to preserve realizability of the finite volume updatem, whereas in [40], the authors change the current moment $u^{t,i}$ to $\langle m f_u\rangle$. This procedure yields an exact pair of moments and Lagrange multipliers, but comes to the cost of losing the conservation property of the scheme. Computational resources for the kinetic solver can be found in the open source kinetic solver Kinetic.jl [41, 42]. The neural networks are implemented in tensorflow 2.0 [43] and the computational resources for the networks in this paper can be found in the repository [44].

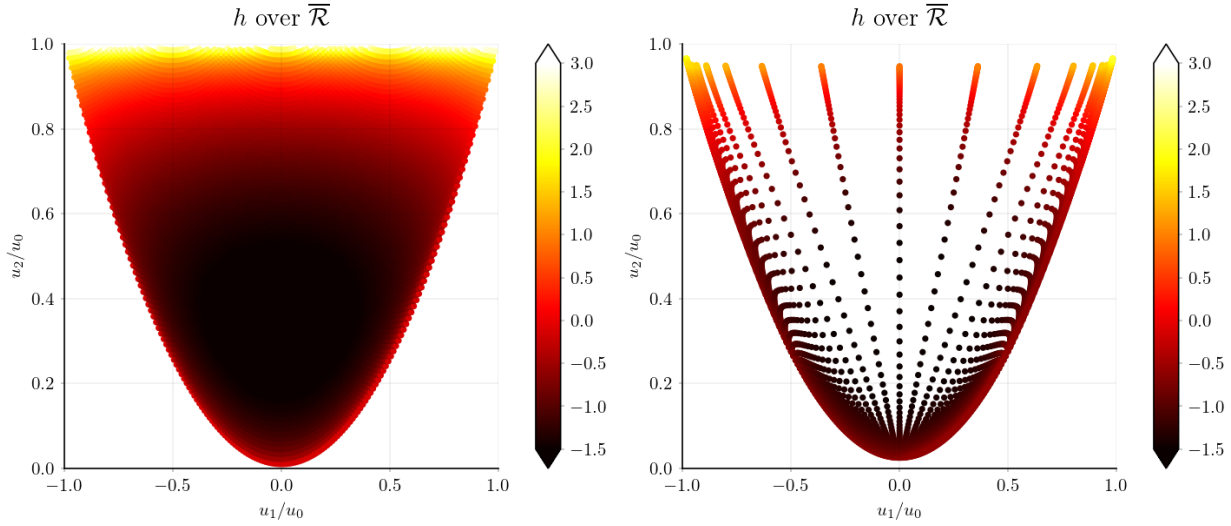
V. Numerical Results

In this section, we present numerical results and investigate the performance of the neural entropy closure. First, we compare the sampling strategies in Algorithm 1 and Algorithm 2. Then we train the networks using the generated data. Finally, we employ the network in the kinetic solver and compare the results with the benchmark solution, where the minimal entropy problem is solved with a Newton solver.

A. Comparison of sampling strategies

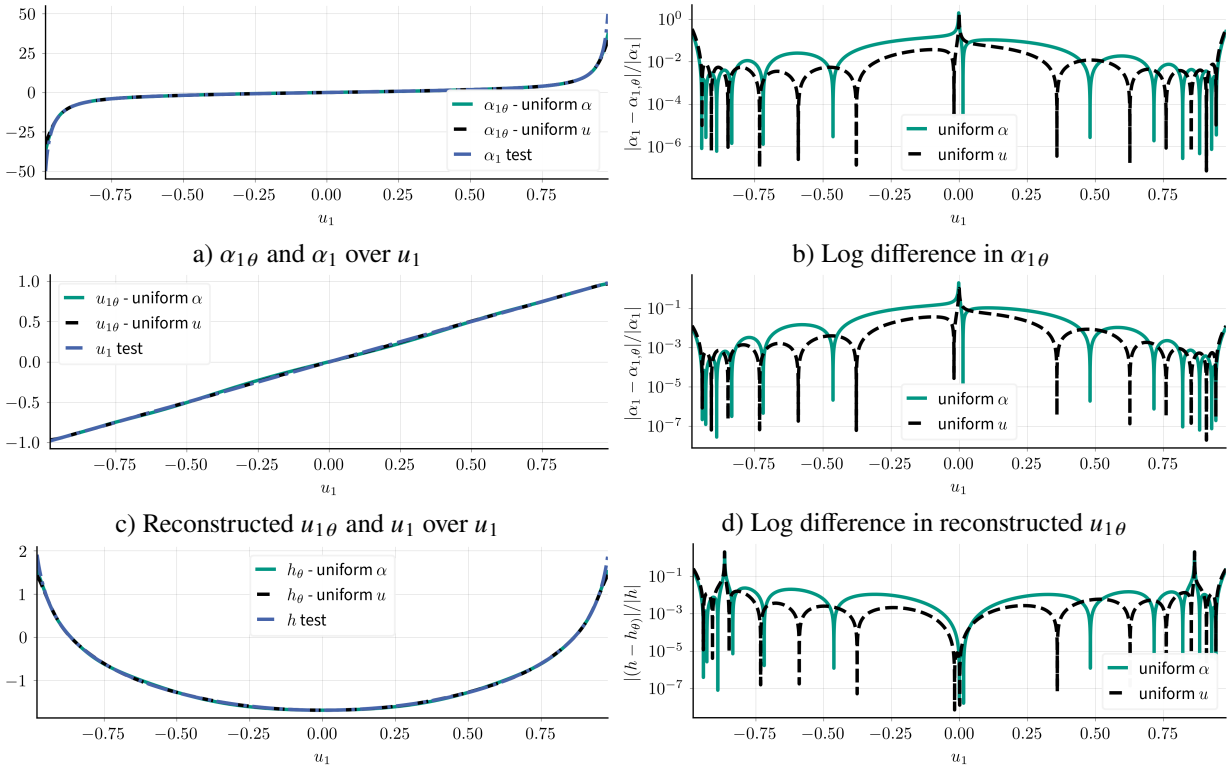
In this section, we compare networks trained on data using Algorithm 1 and Algorithm 2. By construction, the data distribution generated by these strategies are completely different, as it is shown in Fig. 1. Figure 2a) shows, that as we approach the boundary $\partial\overline{\mathcal{R}}^r$, the values of α increase rapidly, while the slope is comparatively flat in the middle of the domain $[-1, 1]$. Thus, uniform sampling in α results in a sparse distribution in the interior of $\overline{\mathcal{R}}^r$ and an accumulation of data-points near $\partial\overline{\mathcal{R}}^r$. In addition, it is not trivial to select the Lagrange multipliers in a way such that the distance of the set of corresponding moments is uniform to $\partial\overline{\mathcal{R}}^r$. This can be seen in the left image Fig. 1, where we sample α from a set $A = [-50, 50] \times [-50, 50]$. The upper boundary of the set of corresponding $\overline{u}^r$ is then concave, however the upper boundary of $\overline{\mathcal{R}}^r$ is a straight line from $(\overline{u}^r_1, \overline{u}^r_2) = (-1, 1)$ to $(1, 1)$.

We perform a synthetic validation test of identical models for the M_1 1D closure once trained on data generated with Algorithm 1 and once trained on data generated with Algorithm 2. Since in this test case, $\mathcal{R}$ is one dimensional and we can sample the exact same interval with both strategies. We train 50 models for each data set using 10000 training samples until convergence and compare the models. Figure 2 shows the evaluations of the model using 100000 unseen test data-points uniformly distributed in $\mathcal{R}$. The left column shows the evaluations u_θ, α_θ, h_θ of both models and the test data u, α and h. The right column shows relative errors of the networks predictions. As expected, Algorithm 1



a) Sampling with Algorithm 1 b) Sampling with Algorithm 2

Fig. 1 Scatter plots of $\overline{R}^r$ for $N = 2$ with data generated from different strategies. Color indicates the value of the minimum point of the entropy functional.



a) $\alpha_{1\theta}$ and α_1 over u_1 b) Log difference in $\alpha_{1\theta}$

c) Reconstructed $u_{1\theta}$ and u_1 over u_1 d) Log difference in reconstructed $u_{1\theta}$

e) h_θ and h over u_1 a) Log difference in h_θ

Fig. 2 Validation tests for M1 1D networks trained on data-sets generated uniformly in α and u. 10^3 training points and 10^4 testing points. Distance to $\partial\mathcal{R}$ is 0.01

performs better in the interior of $\mathcal{R}$ in objective functions h, α and u, since in this region, there is more training data available. However, the advantage of Algorithm 2 near the boundary is not as big as the advantage of Algorithm 1 in the interior, where we experience almost an order of accuracy difference. Note, that the error in h_θ decreases, as $u_1 \to 0$, whereas the error in u_θ and α_θ increases in this region. By Eq. (41) and the definition of u_θ, the error in u_θ is nonlinearly dependent on the error in α_θ. The spike in the relative error is due to the proximity of α (and) u to 0. Similar behavior can be seen at the roots of h. As a conclusion, we use Algorithm 1 to generate training data for the models employed in the following test cases.

Table 2 Validation losses for different closure networks

	Layout	MSE(h, h_θ)	MSE(α, α_θ)	MSE(u, u_θ)	MAE(h, h_θ)	MAE(α, α_θ)	MAE(u, u_θ)
M_1 2D	18×8	1.10e−6	3.39e−5	2.93e−6	1.02e−3	3.36e−3	0.96e−3
M_1 1D	10×7	7.87e−7	7.52e−4	1.47e−6	7.57e−4	1.26e−2	9.59e−4
M_2 1D	15×7	1.33e−5	2.81e−4	2.81e−4	3.11e−3	1.23e−2	1.23e−2

All networks are build using the input convex architecture discussed in Section III. Each network has a dense input layer followed by a block of convex bridge layers defined in Eq. (32). This convex block is described by the layout column in Table 2, with format width × depth. After the convex block, a convex bridge layer with half the width of the block followed by a convex output layer is added. The output layer has the same design as a convex bridge layer, but lacks the softplus activation function, since we deal with a regression task. The validation loss of the neural networks after training can be seen in Table 2. The networks are trained on an Nvidia RTX 3090 GPU in single-precision floating-point accuracy. Notice, that the mean absolute error in α is significantly higher than the error in h or u for all networks. The reason for this is the high range of values, that α can attain, which is inconvenient for the neural network.

B. Computational Efficiency

In the following, we compare the computational efficiency of the neural network surrogate model and the Newton optimizer in an isolated, synthetic test case. We consider the M_2 closure in $1D$ and use only normalized moments. In contrast to the neural network, the performance of the Newton solver is dependent on the proximity of the moments u to the boundary $\partial\overline{\mathcal{R}}$, we consider three test cases. First, the moments are uniformly sampled using Algorithm 1, second we only draw moments near the center of $\overline{\mathcal{R}}$ and lastly, we use only moments in proximity to $\partial\overline{\mathcal{R}}$. The Newton solver is implemented in the KiT-RT [36] framework. In the kinetic scheme, there is one optimization problem for each grid cell. Furthermore, optimization problems of different grid cells are independent of each other. A meaningful and straight-forward way to parallelize the minimal entropy closure is to employ one instance of the Newton optimizer per available CPU-core that handles a batch of cells. For comparability, we set the accuracy tolerance of the Newton solver to single-precision floating point accuracy. On the other hand, we interpret the number of grid cells as the batch size in of the neural entropy closure. Parallelization is carried out by the tensorflow backend.

We execute the neural network once on the CPU and once on the GPU. The used CPU is a 24 thread AMD Ryzen9 3900x with 32GB RAM and the GPU is a RTX3090 with 20GB RAM. The experiments are reiterated 100 times to reduce time measurement fluctuations. Table 3 displays the mean timing for each configuration and corresponding standard deviation. Considering Table 3, we see that the timing of a neural network is independent on the condition of

Table 3 Computational cost for one iteration of the 1D solver in seconds s

	Newton	neural closure CPU	neural closure GPU
uniform, 10^3 samples	0.00648 ± 0.00117 s	0.00788 ± 0.00051 s	0.00988 ± 0.00476 s
uniform, 10^7 samples	5.01239 ± 0.01491 s	0.63321 ± 0.00891 s	0.03909 ± 0.00382 s
boundary, 10^3 samples	38.35292 ± 0.07901 s	0.00802 ± 0.00064 s	0.00974 ± 0.00475 s
boundary, 10^7 samples	27179.51012 ± 133.393 s	0.63299 ± 0.00853 s	0.03881 ± 0.00352 s
interior, 10^3 samples	0.00514 ± 0.00121 s	0.00875 ± 0.00875 s	0.00956 ± 0.00486 s
interior, 10^7 samples	4.24611 ± 0.03862 s	0.63409 ± 0.00867 s	0.03846 ± 0.00357 s

the optimization problem, whereas the Newton solver is 6300 times slower on a on a moment u with $\left\|u - \partial\overline{\mathcal{R}}\right\|_2 = 0.01$ compared to a moment in the interior. The average time to compute the Lagrange multiplier of a uniformly sampled moment u is 27% higher than a moment of the interior. However, we need to take into account that the neural entropy closure is less accurate near $\partial\overline{\mathcal{R}}$ as shown in Fig. 2. Furthermore, we see that the acceleration gained by usage of the neural network surrogate model is higher in cases with more sampling data. This is apparent in the uniform and interior sampling test cases, where the computational time increases by a factor of ≈ 73, when the data size increases by a factor of 10^4. The time consumption of the Newton solver increases by a factor of ≈ 840 in the interior sampling case, respectively ≈ 782 in the uniform sampling case. Note, that in this experiment, all data points fit into the memory of the GPU, so it can more efficiently perform SIMD parallelization.

Table 4 Computational setup of the test cases

	1D M1 Linesource	1D M2 Linesource	2D M1 Periodic
T	$(0, 0.7]$	$(0, 0.7]$	$(0, 1.84]$
N_t	1399	1399	1226
X	$[0, 1]$	$[0, 1]$	$[-1.5, 1.5] \times [-1.5, 1.5]$
N_X	100	100	100^2
Quadrature	Legendre	Legendre	Tensorized Gauss Legendre
$\mathbf{V}$	$[-1, 1]$	$[-1, 1]$	$[-1, 1] \times [0, 2\pi)$
$N_{\mathbf{V}}$	28	28	200
Basis	Monomial	Monomial	Monomial
CFL	0.05	0.05	0.1
σ	1.0	1.0	0.0

C. Inflow into purely scattering homogeneous medium in 1D

Let us first study the particle transports in an isotropic scattering medium. We consider the one-dimensional geometry, where the linear Boltzmann equation reduces to

$$\partial_t f + \mu \partial_{\mathbf{x}} f = Q(f) = \sigma \left(\frac{1}{2} \langle f \rangle - f \right), \tag{54}$$

and the corresponding moment model becomes

$$\partial_t u + \partial_{\mathbf{x}} \langle \mu m f_u \rangle = \langle m Q(f_u) \rangle \tag{55}$$

$$f_u = \eta_*(\partial_u \mathcal{N}_\theta(u) \cdot m) \tag{56}$$

The initial domain is set as vacuum with $f(0, \mathbf{x}, \mu) = 0$, and an inflow condition is imposed at the left boundary of domain with $f(t > 0, 0, \mu > 0) = 0.5$. The detailed computational setup can be found in Table 4. The solution profiles at final time t_f of the neural entropy closed moment system and the benchmark solver are presented in Fig. 3 for the M_1 and M_2 system. In Fig. 4, we see the corresponding errors of the M_1 and M_2 solution for each grid cell. The plots show the error of the moment vector u in l_1 norm as well as the l_1 errors of its components. We notice a slight increase of the error in the M_2 case, overall it displays a good approximation as well.

D. 2D Test cases

We consider a rectangular domain in two spatial dimensions. The phase space of the Boltzmann equation is thus five dimensional, where $\mathbf{X} = [-1.5, 1.5]^2$, $\mathbf{V} = \left\{ v \in \mathbb{R}^2 : \|v\|_2 < 1 \right\}$ and $t > 0$. We consider the M_1 closure with a monomial basis $m(v) = [1, v_x, v_y]^T$, where v_x and v_y are defined in Table 1. The Boltzmann equation reduces to

$$\begin{aligned} \partial_t f + v_x \partial_x f + v_y \partial_y f =& Q(f)(v) \\ =& \sigma \left(\left\langle \frac{1}{2\pi} f \right\rangle - f \right), \end{aligned} \tag{57}$$

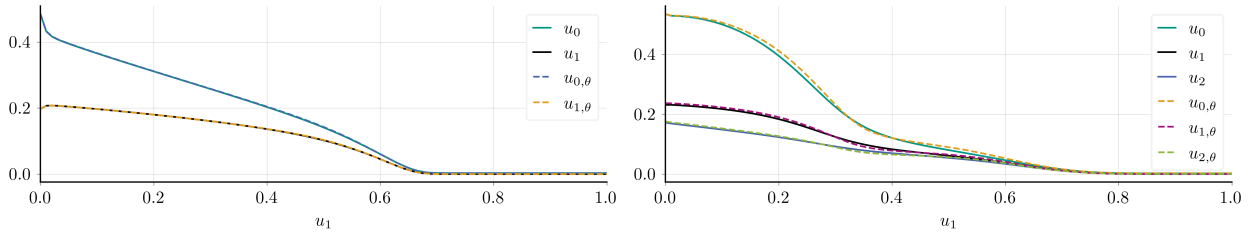


Fig. 3 Comparison of neural closed and benchmark solution at $t = 0.7$. The left figure displays the M_1 and the right figure displays the M_2 test case.

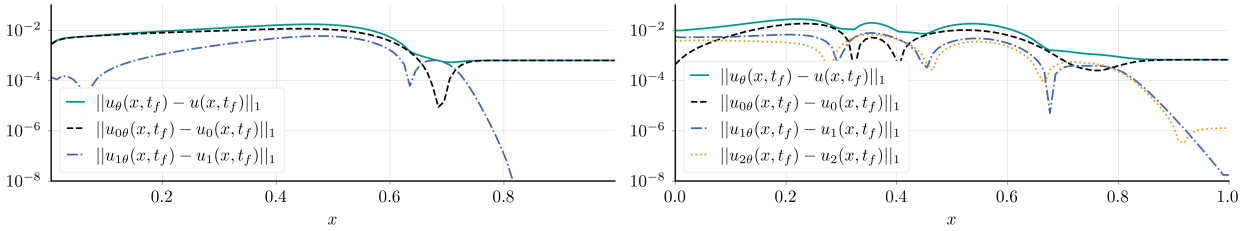


Fig. 4 Comparison of the systems entropy over time, M_1 case on the left and M_2 case on the right.

where σ is the scattering coefficient. The corresponding moment system with neural entropy closure reads as

$$\begin{aligned} \partial_t u + \partial_x \langle v_x m f_u \rangle + + \partial_y \left\langle v_y m f_u \right\rangle &= \langle m Q(f) \rangle \\ f_u &= \eta_*(\partial_u \mathcal{N}_\theta(u) \cdot m) \end{aligned} \tag{58}$$

We inspect the M_1 closure choose the systems initial condition as

$$u_0 = 1.5 + \cos(2\pi x)\cos(2\pi y), \qquad (x, y) \in \mathbf{X} \tag{59}$$

$$u_1 = 0.3 u_0 \tag{60}$$

$$u_2 = 0.3 u_0 \tag{61}$$

The moment system is discretized using the kinetic scheme described in Section IV. We compare the kinetic scheme once closed using a Newton solver and once closed using the neural entropy closure. We use the solver configuration in Table 4 and an entropy consistent numerical scheme. We run the simulation until a final time $t_f = 3.5$, which translates to 1630 time-steps. Figure 5 shows the neural entropy closed solution at time iteration $i = 60$ on the left image and the relative error of the entropy closed solution compared to the Newton optimizer closed solution on the right image. The solution snapshot shows no signs of numerical artifacts and indeed, the relative error per grid cell is mostly in the order of 10^{-2} or lower. In the second experiment, we run both solvers up to final time t_f and compare the propagated error of the neural entropy closure to the Newton solution, that is configured to solve the entropy optimization problem Eq. (13) up to machine precision. The results can be seen in Fig. 6. Note, that the mean relative error does not increase over a threshold of 2.3‰.

Lastly, we look at the entropy of the system at each time step. Due to the periodic boundaries, the physical system is closed. We have chosen the upwind scheme for the numerical flux of the moment system, which is an entropy dissipating scheme. Figure 6 shows that the Newton and neural entropy closed system both dissipate mathematical entropy over time and furthermore, the dissipation rate of both schemes is similar. This shows, that the entropy dissipation property is conserved by the neural entropy closure.

VI. Summary and Conclusion

In this paper we addressed the moment system of the linear Boltzmann equation, its minimal entropy closure, and the challenges of classical numerical approaches. We introduced a novel convex neural network based approach to close the moment hierarchy of the linear Boltzmann equation. The nature of the entropy minimization problem allows clear definition of the convex set of all possible input data for the neural network. On the other hand, the problem is ill

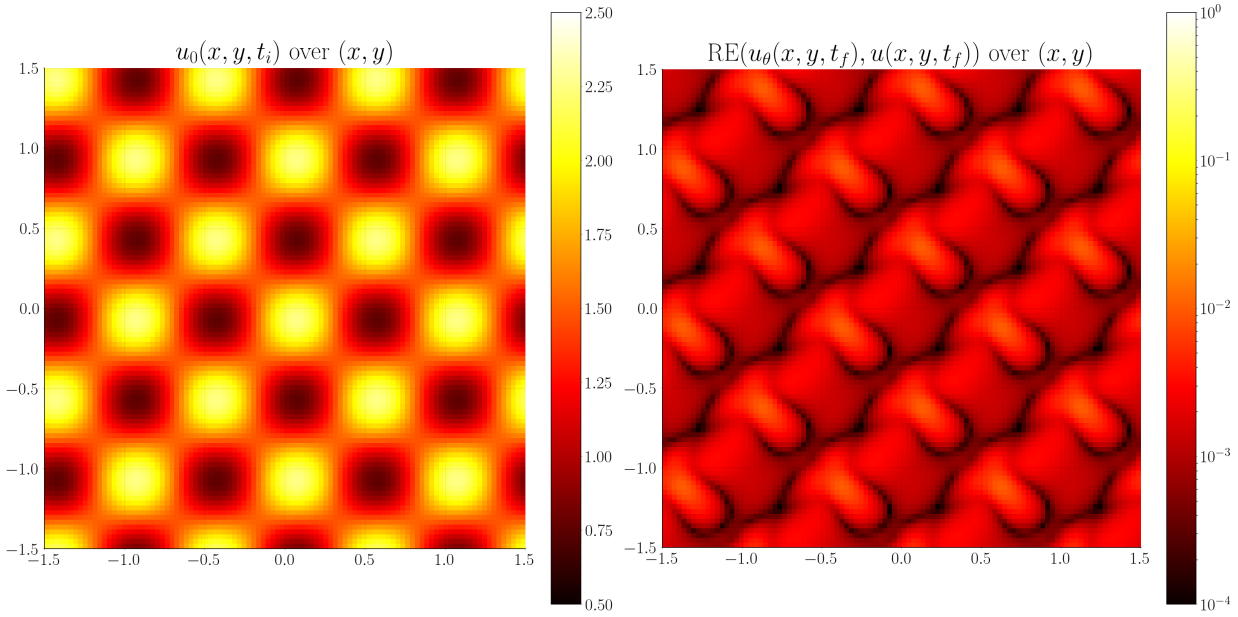


Fig. 5 Solution and relative errors in the periodic $2D$ test case at iteration $i = 60$.

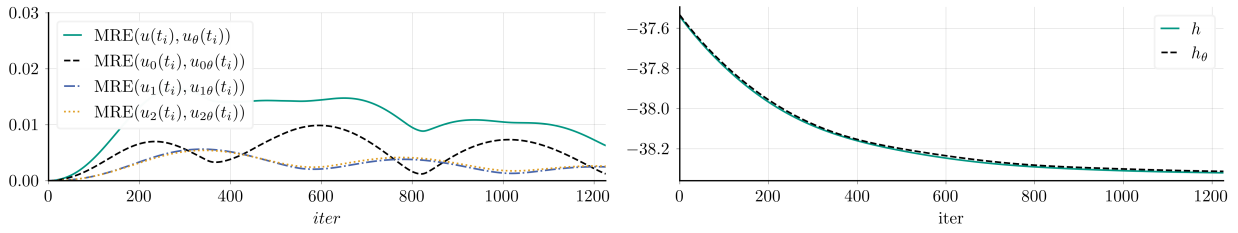


Fig. 6 Mean relative error of neural entropy closed and Newton closed solution (left) system entropy (right) at each time step.

conditioned on the boundary of the realizable set and thus poses significant challenges for generating training data on and near the boundary of it. We propose two algorithms to tackle this challenge and compared them in terms of training and validation performance of the corresponding networks. We found that uniform sampling of $\mathcal{R}$ is advantageous to uniform sampling of the Lagrange multipliers α.

Next, we have constructed an input convex neural network, that mimics the entropy functional of the moment system of the Boltzmann equation. We successfully employed this neural entropy closure in several 1D and 2D test cases and systems of different moment order. We found a good agreement between the neural entropy closed system and the solutions computed with a conventional Newton solver within the boundaries of the training performance of the neural networks. As expected, the neural network based closure is significantly more efficient in terms of computational time compared to a Newton solver.

Several challenges remain to be studied. In higher spatial dimensions and higher order moment systems, it is not clear how to characterize $\overline{\mathcal{R}}$ directly, thus one needs to apply Algorithm 2 and sample in the space of Lagrange multipliers. Then, one needs to explore other sampling distributions due to their shown disadvantages compared to uniform sampling in $\overline{\mathcal{R}}$. Furthermore, it is preferable to characterize the distance to $\partial\overline{\mathcal{R}}$ in terms of α in order to construct an efficient sampling algorithm. Additionally, a rigorous error analysis for the neural entropy closure needs to be conducted. Further research will be dedicated to a sophisticated sampling algorithm for higher dimensions, and consider an error analysis of the neural network reconstruction.

Acknowledgements

The authors acknowledge support by the state of Baden-Württemberg through bwHPC. Furthermore, the authors would like to thank Dr. Jonas Kusch for fruitful discussions about realizability and the minimal entropy closures as well as Jannick Wolters for support in scientific computing matters. The research is funded by the Alexander von Humboldt Foundation (Ref3.5-CHN-1210132-HFST-P).

The work of Cory Hauck is sponsored by the Office of Advanced Scientific Computing Research, U.S. Department of Energy, and performed at the Oak Ridge National Laboratory, which is managed by UT-Battelle, LLC under Contract No. De-AC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

References

[1] Cercignani, C., *The Boltzmann Equation and Its Applications*, Springer, New York, NY, 1988. doi:https://doi.org/10.1007/978-1-4612-1039-9.

[2] Lewis, E., and Miller, W., *Computational methods of neutron transport*, John Wiley and Sons, Inc, 1984. URL `http://inis.iaea.org/search/search.aspx?orig_q=RN:17089238`.

[3] Chahine, M. T., "Foundations of Radiation Hydrodynamics (Dimitri Mihalas and Barbara Weibel Mihalas)," *Siam Review*, Vol. 29, 1987, pp. 648–650.

[4] Markowich, P., Ringhofer, C., and Schmeiser, C., "Semiconductor Equations," 1990.

[5] Camminady, T., Frank, M., Küpper, K., and Kusch, J., "Ray effect mitigation for the discrete ordinates method through quadrature rotation," *J. Comput. Phys.*, Vol. 382, 2019, pp. 105–123.

[6] Xiao, T., Liu, C., Xu, K., and Cai, Q., "A velocity-space adaptive unified gas kinetic scheme for continuum and rarefied flows," *Journal of Computational Physics*, Vol. 415, 2020, p. 109535.

[7] Alldredge, G. W., Frank, M., and Hauck, C. D., "A Regularized Entropy-Based Moment Method for Kinetic Equations," *SIAM Journal on Applied Mathematics*, Vol. 79, No. 5, 2019, pp. 1627–1653. doi:10.1137/18M1181201, URL `https://doi.org/10.1137/18M1181201`.

[8] Alldredge, G. W., Hauck, C. D., and Tits, A. L., "High-Order Entropy-Based Closures for Linear Transport in Slab Geometry II: A Computational Study of the Optimization Problem," *SIAM Journal on Scientific Computing*, Vol. 34, No. 4, 2012, pp. B361–B391. doi:10.1137/11084772X, URL `https://doi.org/10.1137/11084772X`.

[9] Garrett, C. K., and Hauck, C. D., "A Comparison of Moment Closures for Linear Kinetic Transport Equations: The Line Source Benchmark," *Transport Theory and Statistical Physics*, Vol. 42, No. 6-7, 2013, pp. 203–235. doi:10.1080/00411450.2014.910226, URL https://doi.org/10.1080/00411450.2014.910226.

[10] Kristopher Garrett, C., Hauck, C., and Hill, J., "Optimization and large scale computation of an entropy-based moment closure," *Journal of Computational Physics*, Vol. 302, 2015, pp. 573 – 590. doi:https://doi.org/10.1016/j.jcp.2015.09.008, URL http://www.sciencedirect.com/science/article/pii/S0021999115005896.

[11] Levermore, C. D., "Entropy-based moment closures for kinetic equations," *Transport Theory and Statistical Physics*, Vol. 26, No. 4-5, 1997, pp. 591–606. doi:10.1080/00411459708017931, URL https://doi.org/10.1080/00411459708017931.

[12] Brunner, T., "Forms of Approximate Radiation Transport," 2002.

[13] Levermore, C., "Moment closure hierarchies for kinetic theories," *Journal of Statistical Physics*, Vol. 83, 1996, pp. 1021–1065.

[14] Goudon, T., and Lin, C., "Analysis of the M1 model: Well-posedness and diffusion asymptotics," *Journal of Mathematical Analysis and Applications*, Vol. 402, No. 2, 2013, pp. 579–593. doi:https://doi.org/10.1016/j.jmaa.2013.01.042, URL https://www.sciencedirect.com/science/article/pii/S0022247X13000619.

[15] Dubroca, B., and Feugeas, J.-L., "Etude théorique et numérique d'une hiérarchie de modèles aux moments pour le transfert radiatif," *Comptes Rendus de l'Académie des Sciences - Series I - Mathematics*, Vol. 329, No. 10, 1999, pp. 915–920. doi:https://doi.org/10.1016/S0764-4442(00)87499-6, URL https://www.sciencedirect.com/science/article/pii/S0764444200874996.

[16] Huang, J., Cheng, Y., Christlieb, A. J., and Roberts, L. F., "Machine learning moment closure models for the radiative transfer equation I: directly learning a gradient based closure," , 2021.

[17] Han, J., Ma, C., Ma, Z., and E, W., "Uniformly accurate machine learning-based hydrodynamic models for kinetic equations," *Proceedings of the National Academy of Sciences*, Vol. 116, No. 44, 2019, pp. 21983–21991. doi:10.1073/pnas.1909854116, URL https://www.pnas.org/content/116/44/21983.

[18] Huang, J., Ma, Z., Zhou, Y., and Yong, W.-A., "Learning Thermodynamically Stable and Galilean Invariant Partial Differential Equations for Non-equilibrium Flows," , 2020.

[19] Bois, L., Franck, E., Navoret, L., and Vigon, V., "A neural network closure for the Euler-Poisson system based on kinetic simulations," , 2020.

[20] Xiao, T., and Frank, M., "Using neural networks to accelerate the solution of the Boltzmann equation," *arXiv: Computational Physics*, 2020.

[21] Maulik, R., Garland, N. A., Burby, J. W., Tang, X.-Z., and Balaprakash, P., "Neural network representability of fully ionized plasma fluid model closures," *Physics of Plasmas*, Vol. 27, No. 7, 2020, p. 072106. doi:10.1063/5.0006457, URL https://doi.org/10.1063/5.0006457.

[22] Ma, C., Zhu, B., Xu, X.-Q., and Wang, W., "Machine learning surrogate models for Landau fluid closure," *Physics of Plasmas*, Vol. 27, No. 4, 2020, p. 042502. doi:10.1063/1.5129158, URL https://doi.org/10.1063/1.5129158.

[23] Lou, Q., Meng, X., and Karniadakis, G. E., "Physics-informed neural networks for solving forward and inverse flow problems via the Boltzmann-BGK formulation," , 2020.

[24] Mishra, S., and Molinaro, R., "Physics informed neural networks for simulating radiative transfer," *Journal of Quantitative Spectroscopy and Radiative Transfer*, Vol. 270, 2021, p. 107705. doi:https://doi.org/10.1016/j.jqsrt.2021.107705, URL https://www.sciencedirect.com/science/article/pii/S0022407321001989.

[25] Li, R., Lee, E., and Luo, T., "Physics-informed neural networks for solving multiscale mode-resolved phonon Boltzmann transport equation," *Materials Today Physics*, Vol. 19, 2021, p. 100429. doi:https://doi.org/10.1016/j.mtphys.2021.100429, URL https://www.sciencedirect.com/science/article/pii/S2542529321000900.

[26] Porteous, W. A., Laiu, M. P., and Hauck, C. D., "Data-driven, structure-preserving approximations to entropy-based moment closures for kinetic equations," , 2021.

[27] Amos, B., Xu, L., and Kolter, J. Z., "Input Convex Neural Networks," *Proceedings of the 34th International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 70, edited by D. Precup and Y. W. Teh, PMLR, 2017, pp. 146–155. URL http://proceedings.mlr.press/v70/amos17b.html.

[28] Curto, R. E., and Fialkow, L. A., "Recursiveness, positivity, and truncated moment problems," *Houston J. Math*, ????, pp. 603–635.

[29] Cybenko, G., "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals, and Systems (MCSS)*, Vol. 2, No. 4, 1989, pp. 303–314. doi:10.1007/BF02551274, URL http://dx.doi.org/10.1007/BF02551274.

[30] Robbins, H., and Monro, S., "A Stochastic Approximation Method," *Ann. Math. Statist.*, Vol. 22, No. 3, 1951, pp. 400–407. doi:10.1214/aoms/1177729586, URL https://doi.org/10.1214/aoms/1177729586.

[31] Kingma, D. P., and Ba, J., "Adam: A Method for Stochastic Optimization," , 2017.

[32] Boyd, S., and Vandenberghe, L., *Convex Optimization*, Cambridge University Press, 2004. doi:10.1017/CBO9780511804441.

[33] He, K., Zhang, X., Ren, S., and Sun, J., "Deep Residual Learning for Image Recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[34] Kershaw, D., "Flux limiting nature's own way – A new method for numerical solution of the transport equation," 1976.

[35] Monreal, P., "Moment realizability and Kershaw closures in radiative transfer," Ph.D. thesis, Aachen, 2012. URL https://publications.rwth-aachen.de/record/210538, prüfungsjahr: 2012. - Publikationsjahr: 2013; Aachen, Techn. Hochsch., Diss., 2012.

[36] Schotthöfer, S., Wolters, J., Kusch, J., Xiao, T., and Stammer, P., "KiT-RT," https://github.com/CSMMLab/KiT-RT, 2021.

[37] Czarnecki, W. M., Osindero, S., Jaderberg, M., Swirszcz, G., and Pascanu, R., "Sobolev Training for Neural Networks," *CoRR*, Vol. abs/1706.04859, 2017. URL http://arxiv.org/abs/1706.04859.

[38] Atkinson, K., "Numerical integration on the sphere," *The ANZIAM Journal*, Vol. 23/3, pp. 332-347, 1982.

[39] Atkinson, K., and Han, W., *Spherical harmonics and approximations on the unit sphere: an introduction*, Vol. 2044, Springer Science & Business Media, 2012.

[40] Frank, M., Kusch, J., and Wolters, J., "Entropy–Based Methods for Uncertainty Quantification of Hyperbolic Conservation Laws," *Recent Advances in Numerical Methods for Hyperbolic PDE Systems*, edited by M. L. Muñoz-Ruiz, C. Parés, and G. Russo, Springer International Publishing, Cham, 2021, pp. 29–56.

[41] Xiao, T., "Kinetic.jl: A portable finite volume toolbox for scientific and neural computing," *Journal of Open Source Software*, Vol. 6, No. 62, 2021, p. 3060.

[42] Xiao, T., and Schotthöfer, S., "KitML," https://github.com/vavrines/KitML.jl, 2021.

[43] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X., "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems," , 2015. URL https://www.tensorflow.org/, software available from tensorflow.org.

[44] Schotthöfer, S., "neuralEntropyClosures," https://github.com/CSMMLab/neuralEntropyClosures, 2021.