HYPERBOLIC MACHINE LEARNING MOMENT CLOSURES FOR THE BGK EQUATIONS*

ANDREW J. CHRISTLIEB†§, MINGCHANG DING†, JUNTAO HUANG‡, AND
NICHOLAS A. KRUPANSKY§

Abstract. We introduce a hyperbolic closure for the Grad moment expansion of the Bhatnagar–Gross–Krook (BGK) kinetic model using a neural network (NN) trained on BGK's moment data. This closure is motivated by the exact closure for the free streaming limit that we derived in our paper on closures in transport [J. Huang et al., *J. Comput. Phys.*, 453 (2022), 110941]. The exact closure relates the gradient of the highest moment to the gradient of four lower moments. As with our past work, the model presented here learns the gradient of the highest moment in terms of the coefficients of gradients for all lower ones. By necessity, this means that the resulting hyperbolic system is not conservative in the highest moment. For stability, the output layers of the NN are designed to enforce hyperbolicity and Galilean invariance. This ensures the model can be run outside of the training window of the NN. Unlike our previous work on radiation transport that dealt with linear models, the BGK model's nonlinearity demanded advanced training tools. These comprised an optimal learning rate discovery, one-cycle training, batch normalization in each neural layer, and the use of the `AdamW` optimizer. To address the nonconservative structure of the hyperbolic model, we adopt the FORCE numerical method to achieve robust solutions. This results in a comprehensive computing model combining learned closures with methods for solving hyperbolic models. The proposed model generalizes beyond the training time window for low to moderate Knudsen numbers. Our paper details the multiscale model construction and is run on a range of test problems.

Key words. BGK equation, moment closure, machine learning, neural network, hyperbolicity, moment approximation

MSC codes. 82C32, 82C40, 82C70

DOI. 10.1137/24M1629377

1. Introduction. Kinetic models are important in many of the physical sciences and engineering applications when a physical system is far away from a local equilibrium; such is the case in microelectromechanical system (MEMS) [34], atmospheric reentry [46], and controlled nuclear fusion [42], to name a few. When describing systems of interest, these models are plagued by the so-called "curse of dimensionality": in one spatial dimension, one velocity dimension, and time evolution, we have a problem in $\mathbb{R}^3$, and in three spatial dimensions, this associated model is in $\mathbb{R}^7$ phase space for each particle. This leads to the use of distributions, which are further complicated by their multiscale nature, where the behavior of solutions is dependent on the characteristic length of collisions involved. Resolving features in detail on realistic applications can quickly reach resource limits in computation with respect to discretization.

†Department of Computational Mathematics, Science, and Engineering, Michigan State University, East Lansing, MI 48824 USA (christli@msu.edu, dingmin2@msu.edu).

‡Department of Mathematics and Statistics, Texas Tech University, Lubbock, TX 79409 USA (juntao.huang@ttu.edu).

§Department of Mathematics, Michigan State University, East Lansing, MI 48824 USA (krupans4@msu.edu).

The method of moments is an approach to reduce the dimensionality in kinetic models. The first such technique was by Chapman–Enskog and Hilbert expansion, which suffered from singular moments after the first few. The "modern" example is due to Grad [13], where the distribution function in kinetic equations is expanded by a finite number of moments by integrating over velocity. Expanding to a fixed number of moments, say, the kth moment, we require information of the next highest moment, the $(k+1)$th moment, which results in an unclosed system. This leads to the so-called moment closure problem, which addresses how to handle moments beyond the fixed number desired. However, Grad's moment system and closure suffer from a loss of hyperbolicity. This deficiency drew renewed attention by Levermore [28] and has driven contributions to moment closures in kinetic problems too numerous to properly exposition here. We highlight some recent work focused on closures that preserve properties of the moment system, such as hyperbolicity. One such approach is the hyperbolic moment equations (HMEs) [4, 5], where Cai et al. utilize independence between moment expansion coefficients and the Jacobian's characteristic polynomial from Grad's moment system, leading to a regularization that gives global hyperbolicity. Another is the quadrature-based moment equations [23, 24, 26], where hyperbolicity is preserved through the use of quadrature-based projection methods.

Recent efforts in solving the closure problem employ machine learning (ML) techniques that preserve desirable qualities of the kinetic and transport systems. The pioneering work by Han et al. [14] demonstrated the utility of trained neural networks on high-fidelity kinetic data to learn closures over a wide range of Knudsen numbers, with many subsequent works that preserve symmetries and invariances. Li et al. [29] also employed neural networks, specifically the U-net architecture, to train closure models that preserve Galilean, reflecting, and scaling invariance. Charalampopoulos et al. applied recurrent neural networks to correct the quadrature-based moment methods to improve accuracy in the unclosed moment [7].

ML methods have also been applied to kinetic models in other ways than the moment closure problem. Miller et al. applied neural networks to approximate the collision operator of the Boltzmann equation [33]. Neural networks have also been used to represent the Boltzmann equation in a sparse manner by Li et al. [30].

Hyperbolicity-preserving neural networks have been developed and applied to the closure problem. In the radiative transfer equation (RTE), the work of Huang et al. [15, 17, 16], combined the structure of the closure system with a neural network, learning coefficients on the gradients of the moments. A key feature of RTE work is that the model is a local closure, while previous works were global. This is useful in large-scale, distributed computing. Schotthöfer et al. [37, 38, 36] use neural network learned closures that embed entropy convexity while preserving minimum entropy and hyperbolicity for the Boltzmann equation.

By using neural networks trained on kinetic model data, there is a potential to achieve closures that are more accurate by incorporating kinetic effects instead of relying on the strong assumption of local equilibrium, where one can use periodic, microscopic simulations to derive local equations of state (as in the case of the SESAME database [20, 32]). Our work builds on the RTE neural network closure efforts of Huang et al. [15, 17, 16] and applies it to the Bhatnagar–Gross–Krook (BGK) model of the Boltzmann kinetic equation. An analytical closure was developed for the free streaming limit in the first effort [15]. This closure relates the gradient of the highest moment to the next four lower moments. The follow-up RTE efforts [17, 16] develop provably hyperbolic closures relating many gradients of lower moments to the gradient of the highest moment. Our current work follows in that progression, and by necessity, while the system is hyperbolic, it is not conservative in the highest moment.

Hence, for nonlinear problems such as the BGK model, we develop path-conservative numerical methods for solving the fluid ML closure [6]. In the numerical examples, we test two sets of ML moment closures. The first set is to recover the HME model using neural networks. To be more specific, we train the neural network using the data generated from the HME model. The purpose is to validate the correctness of the algorithm and code. The numerical results show good agreement between HMEs and our trained closures for smooth and discontinuous initial conditions across Knudsen number regimes. The second set is to train the neural network using the data from the kinetic model. The goal is to capture the kinetic effects in the ML closure. We have promising results for a kinetic-trained closure on smooth data up to transitional and free streaming (Knudsen numbers between 0.1 and 10). Our contribution is demonstrating hyperbolicity-preserving, local closures from neural networks that have predictive power beyond the training time window across a range of Knudsen numbers.

Although the current work is limited to one spatial dimension and one velocity dimension (1D1V), our proposed approach to moment closure applies to current problems in inertial confinement fusion, which use Vlasov–Fokker–Planck equation in 1D2V [45, 44]. Surrogate models are needed for these Vlasov–Fokker–Planck models in order to perform sensitivity studies ahead high-fidelity computations. Future work could focus on extending our methods beyond 1D1V models.

The rest of the paper is organized as follows. In section 2, we outline the moment expansion and closure system for the BGK equations. Then, in section 3, we outline the neural network structure, training data, and training method. In section 4, we discuss path-conservative solvers used in our calculations with the closures. Finally, in section 5, we present results using our trained closures.

2. Moment closure for the BGK equation. In this section, we review the moment expansion and closure problem for the BGK model and develop the necessary structure for a neural network closure.

The BGK model [2] is a simplification of the Boltzmann kinetic equation, which deals specifically with the collision term. The BGK model assumes that the distribution, f, tends to a local Maxwellian distribution and has several collision invariants that recover conservation laws [43]. The BGK equation is given by

$$(2.1) \qquad \frac{\partial f}{\partial t} + v \frac{\partial f}{\partial x} = \frac{1}{\tau} \left(f_M - f \right),$$

where the unknown function $f = f(x, v, t)$ is the distribution (also called the phase density), v is the microscopic velocity, $\tau > 0$ is a relaxation time related to the Knudsen number, and f_M is the Maxwellian distribution given by

$$(2.2) \qquad f_M = \frac{\rho}{\sqrt{2\pi\theta}} \exp\left(-\frac{(v-u)^2}{2\theta} \right).$$

Physical density, temperature, and macroscopic velocity are given by ρ, θ, and u, respectively [4].

The BGK model has several collision invariant quantities. Let $\boldsymbol{\Phi}(v) = \left(1, v, \frac{1}{2}v^2\right)$; we have the following relationships [4, 43]:

$$(2.3) \qquad \int f \, \boldsymbol{\Phi}(v) \, dv = \begin{pmatrix} \rho \\ \rho u \\ \frac{1}{2}\rho\theta + \frac{1}{2}\rho u^2 \end{pmatrix}.$$

Applying $\boldsymbol{\Phi}$ to both sides of (2.1) and integrating over velocity, we recover the conservation laws:

$$(2.4) \qquad \frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x}(\rho u) = 0,$$

$$(2.5) \qquad \frac{\partial}{\partial t}(\rho u) + \frac{\partial}{\partial x}(\rho\theta + \rho u^2) = 0,$$

$$(2.6) \qquad \frac{\partial}{\partial t}\left(\frac{1}{2}\rho\theta + \frac{1}{2}\rho u^2\right) + \frac{\partial}{\partial x}\left(q + \frac{3}{2}\rho u\theta + \frac{1}{2}\rho u^3\right) = 0,$$

where q is the heat flux.

2.1. Moment method. A common method to solve for the distribution of a kinetic model is the so-called moment method, where the distribution is expanded by a finite number of moments. Due to the finite expansion and relationship of the moments, the system of equations developed by the moment expansion is not closed, and the system requires a closure in the highest-order moment.

In the seminal paper [13], Grad proposed a moment expansion using Hermite polynomials in the following manner:

$$(2.7) \qquad f(x, v, t) = \sum_{k=0}^{\infty} \frac{1}{\sqrt{2\pi}} \theta^{-\frac{k+1}{2}} \operatorname{He}_k(z) \exp\left(-\frac{z^2}{2}\right) f_k(x, t),$$

where $z := \frac{v-u}{\sqrt{\theta}}$, He_k is the kth-order Hermite polynomial, and $f_k = f_k(x, t)$ is called the kth-order moment. It is easy to show that the first four moments in the expansion (2.7) satisfy $f_0 = \rho$, $f_1 = f_2 = 0$, and $3f_3 = q$. Continuing for $k > 3$, the evolution of the moment can be derived using the properties of Hermite polynomials [13]:

$$(2.8) \quad \begin{aligned} &\frac{\partial f_k}{\partial t} - \frac{1}{\rho} f_{k-2} \frac{\partial q}{\partial x} - f_{k-1} \frac{\theta}{\rho} \frac{\partial \rho}{\partial x} + \frac{\partial \theta}{\partial x}\left(\frac{k-1}{2} f_{k-1} + \frac{\theta}{2} f_{k-3}\right) + (k+1) f_k \frac{\partial u}{\partial x} \\ &\quad + u \frac{\partial f_k}{\partial x} + \theta \frac{\partial f_{k-1}}{\partial x} + (k+1)\frac{\partial f_{k+1}}{\partial x} = -\frac{1}{\tau} f_k. \end{aligned}$$

This relation reveals the closure problem since the time evolution of f_k has a dependence on the next-order moment f_{k+1}.

Truncating the moment hierarchy up to order M and setting $f_{M+1} = 0$ yields the so-called Grad's moment closure system. Formulating the moment system from (2.4)–(2.6) and (2.8) in primitive variables $\mathbf{w} = (\rho, u, \theta, f_3, f_4, \ldots, f_M)^T$, we arrive at the Grad's moment system in nonconservative form [13, 4]:

$$(2.9) \qquad \frac{\partial \mathbf{w}}{\partial t} + A^{\mathrm{Grad}} \frac{\partial \mathbf{w}}{\partial x} = \mathbf{Q},$$

where A^{Grad} is a matrix of the form

(2.10)

$$\begin{pmatrix} u & \rho & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ \frac{\theta}{\rho} & u & 1 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 2\theta & u & \frac{6}{\rho} & 0 & 0 & 0 & \cdots & 0 \\ 0 & 4f_3 & \frac{1}{2}\rho\theta & u & 4 & 0 & 0 & \cdots & 0 \\ -\frac{\theta}{\rho}f_3 & 5f_4 & \frac{3}{2}f_3 & \theta & u & 5 & 0 & \cdots & 0 \\ -\frac{\theta}{\rho}f_4 & 6f_5 & 2f_4 & -\frac{3}{\rho}f_3 & \theta & u & 6 & \cdots & 0 \\ -\frac{\theta}{\rho}f_5 & 7f_6 & \frac{\theta}{2}f_3 + \frac{5}{2}f_5 & -\frac{3}{\rho}f_4 & 0 & \theta & u & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & 0 \\ -\frac{\theta}{\rho}f_{M-2} & Mf_{M-1} & \frac{\theta}{2}f_{M-4} + \frac{M-2}{2}f_{M-2} & -\frac{3}{\rho}f_{M-3} & 0 & \cdots & 0 & \theta & u & M \\ -\frac{\theta}{\rho}f_{M-1} & (M+1)f_M & \frac{\theta}{2}f_{M-3} + \frac{M-1}{2}f_{M-1} & -\frac{3}{\rho}f_{M-2} & 0 & \cdots & & 0 & \theta & u \end{pmatrix}$$

and **Q** is the collision term

$$\mathbf{Q} = \left(0, 0, 0, -\frac{1}{\tau}f_3, -\frac{1}{\tau}f_4, \ldots, -\frac{1}{\tau}f_M\right)^T.$$

However, the Grad's moment system (2.9) is not globally hyperbolic; i.e., A^{Grad} is not generally real diagonalizable. To fix this problem, Cai et al. utilize independence between moment expansion coefficients and the Jacobian's characteristic polynomial from Grad's moment system, leading to a regularization that gives global hyperbolicity [4]. This is the so-called HMEs. The change in HMEs occurs in the last row of matrix (2.10) for the second and third entries, which now become

$$(2.11) \qquad \left(-\frac{\theta}{\rho}f_{M-1} \quad 0 \quad -f_{M-1} + \frac{\theta}{2}f_{M-3} \quad -\frac{3}{\rho}f_{M-2} \quad 0 \quad \cdots \quad 0 \quad \theta \quad u\right).$$

In this paper, our goal is to replace the last row of this matrix with a relationship trained by deep learning methods in a way that captures the kinetic effects of the BGK equation but with a relatively small number of moments.

2.2. Deep learning moment closure. We approach the closure problem following our recent work on the RTE [15]. This work differs from the approaches in [14, 29] by learning the gradients of the moments instead of the moments directly and in follow-up efforts [17, 16] preserving the hyperbolicity of the system.

2.2.1. Learning the gradient of moments and hyperbolicity. As shown in the RTE work [15], in the free streaming limit, where the scattering and absorption cross sections vanish, with isotropic initial data, an exact moment closure can be derived. Motivated by this, Huang et al. proposed to directly learn the gradient of the unclosed moment [15]. Generalizing the same idea to the BGK model, we seek a closure relation

$$(2.12) \qquad \frac{\partial f_{M+1}}{\partial x} = \sum_{i=0}^{M} \mathcal{N}_i(\rho, u, \theta, f_3, \ldots, f_M) \frac{\partial f_i}{\partial x},$$

where $\mathcal{N}_i$ is the ith output from the neural network and is a nonlinear function depending on the lower-order moments, $(\rho, u, \theta, f_3, \ldots, f_M)$. Substituting (2.12) into (2.8), we have

$$\frac{\partial f_M}{\partial t} - \frac{1}{\rho}f_{M-2}\frac{\partial q}{\partial x} - f_{M-1}\frac{\theta}{\rho}\frac{\partial \rho}{\partial x} + \frac{\partial \theta}{\partial x}\left(\frac{M-1}{2}f_{M-1} + \frac{\theta}{2}f_{M-3}\right) + (M+1)f_M\frac{\partial u}{\partial x}$$

$$+ u\frac{\partial f_M}{\partial x} + \theta\frac{\partial f_{M-1}}{\partial x} + (M+1)\sum_{i=0}^{M}\mathcal{N}_i(\rho, u, \theta, f_3, \ldots, f_M)\frac{\partial f_i}{\partial x} = 0.$$

Thus, the moment system (2.9) is now changed with the last row of matrix (2.10) given by

$$(2.13) \qquad \begin{pmatrix} a_0 & a_1 & a_2 & a_3 & \cdots & a_{M-1} & a_M \end{pmatrix},$$

where

$$
(2.14) \quad a_i = \begin{cases}
-\dfrac{\theta}{\rho} f_{M-1} + (M+1)\mathcal{N}_0, & i = 0, \\[2mm]
(M+1)f_M + (M+1)\mathcal{N}_1, & i = 1, \\[2mm]
\dfrac{\theta}{\rho} f_{M-3} + \dfrac{M-1}{2} f_{M-1} + (M+1)\mathcal{N}_2, & i = 2, \\[2mm]
-\dfrac{3}{\rho} f_{M-2} + (M+1)\mathcal{N}_3, & i = 3, \\[2mm]
\theta + (M+1)\mathcal{N}_{M-1}, & i = M-1, \\[2mm]
u + (M+1)\mathcal{N}_M, & i = M, \\[2mm]
(M+1)\mathcal{N}_i, & \text{otherwise.}
\end{cases}
$$

We denote this ML moment closure system by

$$
(2.15) \quad \frac{\partial \mathbf{w}}{\partial t} + A^{\mathrm{ML}} \frac{\partial \mathbf{w}}{\partial x} = \mathbf{Q},
$$

where A^{ML} is a matrix which only differs from A^{Grad} in the last row. Next, we will derive some constraints on the neural networks $\mathcal{N}_i$ for $0 \le i \le M$ in (2.12) such that the system (2.15) satisfies Galilean invariance and hyperbolicity.

2.2.2. Galilean invariance. Our moment closure should preserve Galilean invariance. We prove a condition for Galilean invariance.

THEOREM 2.1. *Consider the following moment closure system without collision terms:*

$$
(2.16) \quad \frac{\partial \mathbf{w}}{\partial t} + A(\mathbf{w}) \frac{\partial \mathbf{w}}{\partial x} = 0,
$$

where $\mathbf{w} = (\rho, u, \theta, f_3, f_4, \ldots, f_M)^T \in \mathbb{R}^{M+1}$. The system is Galilean invariant if and only if $(A(\mathbf{w}) - uI)$ is independent of u, where I is the identity matrix.

Proof. Consider two inertial (nonaccelerating) frames of reference given by (x, t) and (x^*, t^*). The frames are equivalent at an initial time, and the second frame is moving at a constant velocity, c, with respect to the other frame. That is, we have $t^* = t$ and $x^* = x - ct$.

The Galilean invariance means that, if $w(x, t)$ satisfies (2.16), then $\mathbf{w}^*(x^*, t^*) = (\rho^*, u^*, \theta^*, f_3^*, \ldots, f_M^*)^T$ with $\rho = \rho^*$, $u^* = u - c$, $\theta^* = \theta$, and $f_k^* = f_k$ for all k such that $3 \le k \le M$, also satisfies (2.16).

Applying the chain rule, we have

$$
\frac{\partial}{\partial t} = \frac{\partial t^*}{\partial t} \frac{\partial}{\partial t^*} + \frac{\partial x^*}{\partial t} \frac{\partial}{\partial x^*} = \frac{\partial}{\partial t^*} - c \frac{\partial}{\partial x^*}
$$

and

$$
\frac{\partial}{\partial x} = \frac{\partial t^*}{\partial x} \frac{\partial}{\partial t^*} + \frac{\partial x^*}{\partial x} \frac{\partial}{\partial x^*} = \frac{\partial}{\partial x^*}.
$$

Proving ($\impliedby$): Suppose that $\mathbf{w} = \mathbf{w}(x, t)$ solves (2.16). Then, $\mathbf{w} = \mathbf{w}(x^*, t^*)$ solves

$$
\frac{\partial \mathbf{w}}{\partial t^*} - c \frac{\partial \mathbf{w}}{\partial x^*} + A(\mathbf{w}) \frac{\partial \mathbf{w}}{\partial x^*} = 0,
$$

rearranging

$$\frac{\partial \mathbf{w}}{\partial t^*} + (A(\mathbf{w}) - cI)\frac{\partial \mathbf{w}}{\partial x^*} = 0.$$

Since $A(\mathbf{w}) - uI$ is independent of u,

$$A(\mathbf{w}) - uI = A(\mathbf{w}^*) - u^* I = A(\mathbf{w}^*) - (u - c)I$$

or

$$A(\mathbf{w}^*) = A(\mathbf{w}) - cI.$$

Thus, $\mathbf{w}^*(x^*, t^*)$ satisfies $\frac{\partial \mathbf{w}^*}{\partial t^*} + A(\mathbf{w}^*)\frac{\partial \mathbf{w}^*}{\partial x^*} = 0$.

Proving ($\Longrightarrow$): Comparing equations involving $\mathbf{w}$ and $\mathbf{w}^*$, we have

$$A(\mathbf{w}^*) = A(\mathbf{w}) - cI.$$

Differentiating with respect to c,

$$-\frac{\partial A(\mathbf{w}^*)}{\partial c} = -I,$$

and thus,

$$\frac{\partial}{\partial u}(A(\mathbf{w}^*) - uI) = 0. \qquad \qquad \square$$

From Theorem 2.1, it is easy to show the corollary 2.1.

COROLLARY 2.1. *Consider the ML moment closure system* (2.15). *Denote the eigenvalues of A^{ML} by r_k for $0 \leq k \leq M$. The system is Galilean invariant if and only if either of the two conditions is satisfied:*

  1. *a_k for $0 \leq k \leq M-1$ and $(a_m - u)$ are independent of u.*
  2. *$(r_k - u)$ for $0 \leq k \leq M$ are independent of u.*

From corollary 2.1, to guarantee the Galilean invariance, the eigenvalues of A^{ML} in (2.15) must have the form $r_k = u + \tilde{r}_k$, where $\tilde{r}_k$ is independent of u.

2.2.3. Provable hyperbolic structure. By taking advantage of the moment system structure, we will develop the framework for the neural network to maintain hyperbolicity by relating the coefficients of gradients to the eigenvalues of the moment system. First, we recall some definitions and theorems in [16].

DEFINITION 2.1 (lower Hessenberg matrix [16]). *The matrix $H = (h_{ij})_{n \times n}$ is called a lower Hessenberg matrix if $h_{ij}=0$ for $j>i+1$. If $h_{i,i+1} \neq 0$ for $i=1, 2, \ldots, n-1$, it is called an unreduced lower Hessenberg matrix.*

DEFINITION 2.2 (associated polynomial sequence [10, 16]). *Let $H = (h_{ij})_{n \times n}$ be an unreduced lower Hessenberg matrix. The associated polynomial sequence $\{q_i\}_{0 \leq i \leq n}$ with H is defined as $q_0 = 1$ and*

$$(2.17) \qquad q_i(x) = \frac{1}{h_{i,i+1}}\left(x q_{i-1}(x) - \sum_{j=1}^{i} h_{ij} q_{j-1}(x)\right), \quad 1 \leq i \leq n$$

with $h_{n,n+1} = 1$.

THEOREM 2.3 ([10, 16]). *Let $H = (h_{ij})_{n \times n}$ be an unreduced lower Hessenberg matrix and $\{q_i\}_{0 \le i \le n}$ be the associated polynomial sequence with H. The following are true:*

1. *If λ is a root of q_n, then λ is an eigenvalue of H, and a corresponding eigenvector is $(q_0(\lambda), q_1(\lambda), \ldots, q_{n-1}(\lambda))^T$.*
2. *If all the roots of q_n are simple, then the characteristic polynomial of H is μq_n, where $\mu = \prod_{i=1}^{n-1} h_{i,i+1}$; that is,*

$$(2.18) \qquad \det(xI - H) = \mu q_n(x),$$

where I is the identity matrix.

THEOREM 2.4 ([16]). *Let $H = (h_{ij})_{n \times n}$ be an unreduced lower Hessenberg matrix and $\{q_i\}_{0 \le i \le n}$ be the associated polynomial sequence with H. The following are equivalent:*

1. *H is real diagonalizable,*
2. *all eigenvalues of H are distinct and real, and*
3. *all roots of q_n are simple and real.*

Theorem 2.5 provide a method to ensure hyperbolicity of the moment system when training a neural network–based closure. We derive the associated polynomial sequence for the matrix of the ML moment system (2.15).

THEOREM 2.5. *The associated polynomial sequence $\{q_k(x)\}$, where $0 \le k \le M+1$ for A^{ML} in (2.15), satisfies the following relationship:*

$$q_0(x) = \mathrm{He}_0, \quad q_1(x) = \frac{\theta^{\frac{1}{2}}}{\rho}\mathrm{He}_1, \quad q_2(x) = \frac{\theta}{\rho}\mathrm{He}_2, \quad q_3(x) = \frac{\theta^{\frac{3}{2}}}{6}\mathrm{He}_3,$$

$$q_k(x) = \frac{\theta^{\frac{k}{2}}}{k!}\mathrm{He}_k - \frac{\theta f_{k-2}}{2\rho}\mathrm{He}_2 - \frac{\theta^{\frac{1}{2}}f_{k-1}}{\rho}\mathrm{He}_1, \quad 4 \le k \le M,$$

$$\begin{aligned}
q_{M+1}(x) = {} & \frac{\theta^{\frac{M+1}{2}}}{M!}\mathrm{He}_{M+1} - (a_M - u)\frac{\theta^{\frac{M}{2}}}{M!}\mathrm{He}_M \\
& + \left(\frac{\theta^{\frac{M+1}{2}}}{(M-1)!} - a_{M-1}\frac{\theta^{\frac{M-1}{2}}}{(M-1)!}\right)\mathrm{He}_{M-1} \\
& - \sum_{k=4}^{M-2} a_k \frac{\theta^{\frac{k}{2}}}{k!}\mathrm{He}_k + \left(-\frac{\theta^{\frac{3}{2}}f_{M-2}}{2\rho} - a_3\frac{\theta^{\frac{3}{2}}}{6}\right)\mathrm{He}_3 \\
& + \left(-\frac{\theta f_{M-1}}{\rho} + (a_M - u)\frac{\theta f_{M-2}}{2\rho} + \sum_{k=6}^{M} a_{k-1}\frac{\theta f_{k-3}}{2\rho} - a_2\frac{\theta}{\rho}\right)\mathrm{He}_2 \\
& + \left(-\frac{\theta^{\frac{3}{2}}f_{M-2}}{\rho} + (a_M - u)\frac{\theta^{\frac{1}{2}}f_{M-1}}{\rho} + \sum_{k=5}^{M} a_{k-1}\frac{\theta^{\frac{1}{2}}f_{k-2}}{\rho} - a_1\frac{\theta^{\frac{1}{2}}}{\rho}\right)\mathrm{He}_1 \\
& + \left(-\frac{\theta f_{M-1}}{\rho} - a_0\right)\mathrm{He}_0,
\end{aligned}$$

where the argument $\xi = \frac{x-u}{\sqrt{\theta}}$ in the Hermite polynomials is omitted. If further assuming that all the roots of q_{M+1} are simple, the eigenpolynomial of A^{ML} satisfies

$$(2.19) \qquad p_{M+1}(x) = M! q_{M+1}(x).$$

Proof. Using Definition 2.2, $q_i(x)$ for $0 \leq i \leq 3$ can be directly calculated, and the details are omitted here.

Next, we assume that the relationship holds for $k = n - 1$ with $4 \leq n \leq M$, and we would like to prove that it also holds for $k = n$. By definition (2.17), we have

$$
\begin{aligned}
nq_n(x) = {}& (x - u)q_{n-1}(x) + \frac{\theta f_{n-2}}{\rho} q_0(x) - nf_{n-1}q_1(x) \\
& - \frac{1}{2}((n-2)f_{n-2} + \theta f_{n-4})q_2(x) + \frac{3f_{n-3}}{\rho}q_3(x) - \theta q_{n-2} \\
= {}& (x-u)\left(\frac{\theta^{\frac{n-1}{2}}}{(n-1)!} \operatorname{He}_{n-1}(\xi) - \frac{\theta f_{n-3}}{2\rho} \operatorname{He}_2(\xi) - \frac{\theta^{\frac{1}{2}} f_{k-2}}{\rho} \operatorname{He}_1(\xi) \right) \\
& + \frac{\theta f_{n-2}}{\rho} \operatorname{He}_0(\xi) - nf_{n-1}\frac{\theta^{\frac{1}{2}}}{\rho} \operatorname{He}_1(\xi) - \frac{1}{2}((n-2)f_{n-2} + \theta f_{n-4})\frac{\theta}{\rho}\operatorname{He}_2(\xi) \\
& + \frac{3f_{n-3}}{\rho}\frac{\theta^{\frac{3}{2}}}{6} \operatorname{He}_3(\xi) \\
& - \theta\left(\frac{\theta^{\frac{n-2}{2}}}{(n-2)!} \operatorname{He}_{n-2}(\xi) - \frac{\theta f_{n-4}}{2\rho} \operatorname{He}_2(\xi) - \frac{\theta^{\frac{1}{2}} f_{k-3}}{\rho} \operatorname{He}_1(\xi) \right) \\
= {}& \theta^{\frac{1}{2}}\xi\left(\frac{\theta^{\frac{n-1}{2}}}{(n-1)!} \operatorname{He}_{n-1}(\xi) - \frac{\theta f_{n-3}}{2\rho} \operatorname{He}_2(\xi) - \frac{\theta^{\frac{1}{2}} f_{k-2}}{\rho} \operatorname{He}_1(\xi) \right) \\
& + \frac{\theta f_{n-2}}{\rho} \operatorname{He}_0(\xi) - nf_{n-1}\frac{\theta^{\frac{1}{2}}}{\rho} \operatorname{He}_1(\xi) - \frac{1}{2}((n-2)f_{n-2} + \theta f_{n-4})\frac{\theta}{\rho}\operatorname{He}_2(\xi) \\
& + \frac{3f_{n-3}}{\rho}\frac{\theta^{\frac{3}{2}}}{6} \operatorname{He}_3(\xi) \\
& - \theta\left(\frac{\theta^{\frac{n-2}{2}}}{(n-2)!} \operatorname{He}_{n-2}(\xi) - \frac{\theta f_{n-4}}{2\rho} \operatorname{He}_2(\xi) - \frac{\theta^{\frac{1}{2}} f_{n-3}}{\rho} \operatorname{He}_1(\xi) \right) \\
= {}& \frac{\theta^{\frac{n}{2}}}{(n-1)!} \operatorname{He}_n(\xi) - n\frac{\theta f_{n-2}}{2\rho} \operatorname{He}_2(\xi) - n\frac{\theta^{\frac{1}{2}} f_{n-1}}{\rho} \operatorname{He}_1(\xi),
\end{aligned}
$$

where we use $x - u = \theta^{\frac{1}{2}}\xi$ and the recurrence relation $\xi \operatorname{He}_k(\xi) = \operatorname{He}_{k+1}(\xi) + k \operatorname{He}_{k-1}(\xi)$. Thus, we have

$$
q_n(x) = \frac{\theta^{\frac{n}{2}}}{n!} \operatorname{He}_n(\xi) - \frac{\theta f_{n-2}}{2\rho} \operatorname{He}_2(\xi) - \frac{\theta^{\frac{1}{2}} f_{n-1}}{\rho} \operatorname{He}_1(\xi).
$$

For $k = M + 1$, we have

$$
\begin{aligned}
q_{M+1}(x) = {}& (x - a_M)\left(\frac{\theta^{\frac{M}{2}}}{M!} \operatorname{He}_M - \frac{\theta f_{M-2}}{2\rho} \operatorname{He}_2 - \frac{\theta^{\frac{1}{2}} f_{M-1}}{\rho} \operatorname{He}_1 \right) \\
& - a_{M-1}\left(\frac{\theta^{\frac{M-1}{2}}}{(M-1)!} \operatorname{He}_{M-1} - \frac{\theta f_{M-3}}{2\rho} \operatorname{He}_2 - \frac{\theta^{\frac{1}{2}} f_{M-2}}{\rho} \operatorname{He}_1 \right) \\
& \cdots \\
& - a_5\left(\frac{\theta^{\frac{5}{2}}}{5!} \operatorname{He}_5 - \frac{\theta f_3}{2\rho} \operatorname{He}_2 - \frac{\theta^{\frac{1}{2}} f_4}{\rho} \operatorname{He}_1 \right)
\end{aligned}
$$

$$- a_4 \left(\frac{\theta^{\frac{4}{2}}}{4!} \operatorname{He}_4 - \frac{\theta^{\frac{1}{2}} f_3}{\rho} \operatorname{He}_1 \right)$$

$$- a_3 \frac{\theta^{\frac{3}{2}}}{6} \operatorname{He}_3 - a_2 \frac{\theta}{\rho} \operatorname{He}_2 - a_1 \frac{\theta^{\frac{1}{2}}}{\rho} \operatorname{He}_1 - a_0 \operatorname{He}_0,$$

where we omit the argument in the Hermite polynomials. The first term can be simplified using the recurrence relation:

$$(x - a_M) \left(\frac{\theta^{\frac{M}{2}}}{M!} \operatorname{He}_M - \frac{\theta f_{M-2}}{2\rho} \operatorname{He}_2 - \frac{\theta^{\frac{1}{2}} f_{M-1}}{\rho} \operatorname{He}_1 \right)$$

$$= (x - u) \left(\frac{\theta^{\frac{M}{2}}}{M!} \operatorname{He}_M - \frac{\theta f_{M-2}}{2\rho} \operatorname{He}_2 - \frac{\theta^{\frac{1}{2}} f_{M-1}}{\rho} \operatorname{He}_1 \right)$$

$$\quad - (a_M - u) \left(\frac{\theta^{\frac{M}{2}}}{M!} \operatorname{He}_M - \frac{\theta f_{M-2}}{2\rho} \operatorname{He}_2 - \frac{\theta^{\frac{1}{2}} f_{M-1}}{\rho} \operatorname{He}_1 \right) n$$

$$= \frac{\theta^{\frac{M}{2}}}{M!} (\theta^{\frac{1}{2}} \operatorname{He}_{M+1} + M\theta^{\frac{1}{2}} \operatorname{He}_{M-1}) - \frac{\theta f_{M-2}}{2\rho} (\theta^{\frac{1}{2}} \operatorname{He}_3 + 2\theta^{\frac{1}{2}} \operatorname{He}_1)$$

$$\quad - \frac{\theta^{\frac{1}{2}} f_{M-1}}{\rho} (\theta^{\frac{1}{2}} \operatorname{He}_2 + \theta^{\frac{1}{2}} \operatorname{He}_0)$$

$$\quad - (a_M - u) \left(\frac{\theta^{\frac{M}{2}}}{M!} \operatorname{He}_M - \frac{\theta f_{M-2}}{2\rho} \operatorname{He}_2 - \frac{\theta^{\frac{1}{2}} f_{M-1}}{\rho} \operatorname{He}_1 \right).$$

Then,

$$q_{M+1}(x) = \frac{\theta^{\frac{M}{2}}}{M!} (\theta^{\frac{1}{2}} \operatorname{He}_{M+1} + M\theta^{\frac{1}{2}} \operatorname{He}_{M-1}) - \frac{\theta f_{M-2}}{2\rho} (\theta^{\frac{1}{2}} \operatorname{He}_3 + 2\theta^{\frac{1}{2}} \operatorname{He}_1)$$

$$\quad - \frac{\theta^{\frac{1}{2}} f_{M-1}}{\rho} (\theta^{\frac{1}{2}} \operatorname{He}_2 + \theta^{\frac{1}{2}} \operatorname{He}_0)$$

$$\quad - (a_M - u) \left(\frac{\theta^{\frac{M}{2}}}{M!} \operatorname{He}_M - \frac{\theta f_{M-2}}{2\rho} \operatorname{He}_2 - \frac{\theta^{\frac{1}{2}} f_{M-1}}{\rho} \operatorname{He}_1 \right)$$

$$\quad - a_{M-1} \left(\frac{\theta^{\frac{M-1}{2}}}{(M-1)!} \operatorname{He}_{M-1} - \frac{\theta f_{M-3}}{2\rho} \operatorname{He}_2 - \frac{\theta^{\frac{1}{2}} f_{M-2}}{\rho} \operatorname{He}_1 \right)$$

$$\cdots$$

$$\quad - a_5 \left(\frac{\theta^{\frac{5}{2}}}{5!} \operatorname{He}_5 - \frac{\theta f_3}{2\rho} \operatorname{He}_2 - \frac{\theta^{\frac{1}{2}} f_4}{\rho} \operatorname{He}_1 \right)$$

$$\quad - a_4 \left(\frac{\theta^{\frac{4}{2}}}{4!} \operatorname{He}_4 - \frac{\theta^{\frac{1}{2}} f_3}{\rho} \operatorname{He}_1 \right)$$

$$\quad - a_3 \frac{\theta^{\frac{3}{2}}}{6} \operatorname{He}_3 - a_2 \frac{\theta}{\rho} \operatorname{He}_2 - a_1 \frac{\theta^{\frac{1}{2}}}{\rho} \operatorname{He}_1 - a_0 \operatorname{He}_0$$

$$= \frac{\theta^{\frac{M+1}{2}}}{M!} \operatorname{He}_{M+1} - (a_M - u) \frac{\theta^{\frac{M}{2}}}{M!} \operatorname{He}_M$$

$$\quad + \left(\frac{\theta^{\frac{M+1}{2}}}{(M-1)!} - a_{M-1} \frac{\theta^{\frac{M-1}{2}}}{(M-1)!} \right) \operatorname{He}_{M-1}$$

$$- \sum_{k=4}^{M-2} a_k \frac{\theta^{\frac{k}{2}}}{k!} \operatorname{He}_k + \left(-\frac{\theta^{\frac{3}{2}} f_{M-2}}{2\rho} - a_3 \frac{\theta^{\frac{3}{2}}}{6} \right) \operatorname{He}_3$$

$$+ \left(-\frac{\theta f_{M-1}}{\rho} + (a_M - u)\frac{\theta f_{M-2}}{2\rho} + \sum_{k=6}^{M} a_{k-1} \frac{\theta f_{k-3}}{2\rho} - a_2 \frac{\theta}{\rho} \right) \operatorname{He}_2$$

$$+ \left(-\frac{\theta^{\frac{3}{2}} f_{M-2}}{\rho} + (a_M - u)\frac{\theta^{\frac{1}{2}} f_{M-1}}{\rho} + \sum_{k=5}^{M} a_{k-1} \frac{\theta^{\frac{1}{2}} f_{k-2}}{\rho} - a_1 \frac{\theta^{\frac{1}{2}}}{\rho} \right) \operatorname{He}_1$$

$$+ \left(-\frac{\theta f_{M-1}}{\rho} - a_0 \right) \operatorname{He}_0 .$$

From Theorem 2.3, we see that the characteristic polynomial is given by $p_{M+1}(x) = M! q_{M+1}(x)$. □

2.2.4. Translating eigenvalues to coefficients of gradients. Proceeding as in the RTE effort [16], we can derive a relationship between the eigenvalues and the weights of the gradient-based closure using Vieta's formula as well as a transformation between monomial coefficients and the Hermite polynomials.

The relation between monomials and the Hermite polynomials is [9]

$$(2.20) \qquad x^m = \sum_{k=0}^{\lfloor \frac{m}{2} \rfloor} F(m,k) \operatorname{He}_{m-2k}(x),$$

where $F(m,k) = \frac{m!}{2^k k!(m-2k)!}$. We can rewrite this into an equivalent form

$$(2.21) \qquad x^m = \sum_{k=0}^{m} b_{mk} \operatorname{He}_k(x), \quad m \geq 0$$

with

$$(2.22) \qquad b_{mk} = \begin{cases} F(m, \frac{1}{2}(m-k)) & \text{if } m \equiv k \pmod 2, \\ 0, & \text{otherwise.} \end{cases}$$

From this formula, we can expand any polynomials in terms of Hermite polynomials.

$$\sum_{i=0}^{n} c_i x^i = \sum_{i=0}^{n} c_i \left(\sum_{k=0}^{i} b_{ik} \operatorname{He}_k(x) \right) = \sum_{i=0}^{n} \left(\sum_{i=k}^{n} c_i b_{ik} \right) \operatorname{He}_k(x) = \sum_{k=0}^{n} \alpha_k \operatorname{He}_k(x),$$

where c_i are polynomials coefficients and the coefficients of Hermite polynomials are determined by

$$(2.23) \qquad \alpha_k = \sum_{i=k}^{n} c_i b_{ik}.$$

Now, we proceed to derive a relationship between the eigenvalues of A^{ML} in (2.15) and the last row of learned coefficients. Let $r_k = u + \tilde{r}_k$ for $k = 0, 1, \ldots, M$, and let us write the eigenpolynomial as

$$(2.24) \qquad p_{M+1} = c_0 + c_1(x - u) + \cdots + c_M (x - u)^M + (x - u)^{M+1};$$

then, we have that

$$(2.25) \quad (x - u - \tilde{r}_0)(x - u - \tilde{r}_1)\cdots(x - u - \tilde{r}_M)$$
$$= c_0 + c_1(x - u) + \cdots + c_M(x - u)^M + (x - u)^{M+1}.$$

Applying Vieta's formula, we can relate the coefficients c_k to sums and products of $\tilde{r}_k$ in the following manner:

$$\tilde{r}_0 + \tilde{r}_1 + \cdots \tilde{r}_{M-1} + \tilde{r}_M = -c_M$$
$$(\tilde{r}_0\tilde{r}_1 + \tilde{r}_0\tilde{r}_2 + \cdots \tilde{r}_0\tilde{r}_M) + (\tilde{r}_1\tilde{r}_2 + \tilde{r}_1\tilde{r}_3 + \cdots + \tilde{r}_1\tilde{r}_M) + \cdots + \tilde{r}_{M-1}\tilde{r}_M = c_{M-1}$$
$$\vdots$$
$$\tilde{r}_0\tilde{r}_1 \cdots \tilde{r}_{M-1}\tilde{r}_M = (-1)^{M+1}c_0,$$

or equivalently,

$$(2.26) \quad \sum_{0 \le i_1 < i_2 < \cdots < i_k \le M} \left(\prod_{j=1}^{k} \tilde{r}_{i_j} = (-1)^k c_{M+1-k} \right),$$

where the indices i_k are sorted strictly increasing to ensure each product of k roots is used once.

We then transform the basis $(x - u)^k$ into the Hermite basis $\mathrm{He}_k(\frac{x-u}{\theta^{\frac{1}{2}}})$,

$$c_0 + c_1(x - u) + \cdots + c_M(x - u)^M + (x - u)^{M+1} = \beta_0 \mathrm{He}_0\left(\frac{x - u}{\theta^{\frac{1}{2}}}\right)$$
$$+ \beta_1\theta^{\frac{1}{2}} \mathrm{He}_1\left(\frac{x - u}{\theta^{\frac{1}{2}}}\right) + \cdots + \beta_M\theta^{\frac{M}{2}} \mathrm{He}_M\left(\frac{x - u}{\theta^{\frac{1}{2}}}\right) + \theta^{\frac{M+1}{2}} \mathrm{He}_{M+1}\left(\frac{x - u}{\theta^{\frac{1}{2}}}\right).$$

Let $s = \frac{x-u}{\theta^{\frac{1}{2}}}$; then, we have

$$c_0 + c_1\theta^{\frac{1}{2}}s + \cdots + c_M\theta^{\frac{M}{2}}s^M + \theta^{\frac{M+1}{2}}s^{M+1}$$
$$= \beta_0 \mathrm{He}_0(s) + \beta_1\theta^{\frac{1}{2}} \mathrm{He}_1(s) + \cdots + \beta_M\theta^{\frac{M}{2}} s\, \mathrm{He}_M(s) + \theta^{\frac{M+1}{2}} \mathrm{He}_{M+1}(s),$$

and thus, by (2.23),

$$(2.27) \quad \beta_k\theta^{\frac{k}{2}} = \sum_{i=k}^{M+1} c_i\theta^{\frac{i}{2}}b_{ik}, \quad k = 0, \ldots, M+1$$

with $c_{M+1} = 1$. These β_k coefficients can then be transformed to entries a_k of (2.13) using the equations of the eigenpolynomials for A^{ML} (2.19).

3. Training setup and neural network architecture. Our effort seeks to apply the hyperbolicity-preserving deep learning closure methods developed for RTE [16] and apply them to the BGK model. The goal is to capture kinetic effects in the learned closure by generating high-fidelity training data solving the BGK equation via discrete velocity methods (DVMs) with smooth and discontinuous initial data [14] over a range of Knudsen numbers. During the course of the investigation, several modifications were needed to the training methods and neural network architectures originally used in RTE [16] to improve training errors for the BGK model. This is likely due to the strong nonlinearity of the BGK model compared to RTE. We

discuss the machine learning aspects of our effort. All deep learning methods were implemented using PyTorch [35].

In the numerical examples, we test two sets of ML moment closures. The first set is to recover the HME model using neural networks. To be more specific, we train the neural network using the data generated from the HME model. The purpose is to validate the correctness of the algorithm and code. The second is to train the neural network using the data from the kinetic model. The goal is to capture the kinetic effects in the ML closure.

3.1. Training data preparation. We generate training data following [14]. When generating data from the HME model, we use the high-order path-conservative numerical scheme in section 4.2. When generating the data from the kinetic model, we use the fifth-order finite difference weighted essentially non-oscillatory (WENO) scheme in physical space [19], DVM in velocity space [3], and third-order implicit-explicit method in time [1]. In Table 1, we list the various parameters for the HME and kinetic model training data.

Following [14], we have two sets of training data. The first set is generated from smooth data as the initial condition, which is also called wave data. First, we generate the following macroscopic variables:

$$(3.1) \qquad U = \begin{cases} \rho(x,0) = a_\rho \sin\left(\frac{2k_\rho \pi x}{L} + \psi_\rho\right) + b_\rho, \\ u(x,0) = 0, \\ \theta(x,0) = a_\theta \sin\left(\frac{2k_\theta \pi x}{L} + \psi_\theta\right) + b_\theta, \end{cases}$$

where the random variables are uniformly sampled in their associated ranges $a_z \in [0.2, 0.3]$, $\psi_z \in [0, 2\pi]$, $b_z \in [0.5, 0.7]$, k_z is a random integer sampled from the set $\{1, 2, 3, 4\}$, and z can be either ρ or θ. Then, two local Maxwellian distributions are generated from the above macroscopic variables U_1 and U_2 and blended together to form the wave data

$$(3.2) \qquad f_{wave} = \frac{\alpha_1 f_M(v; U_1) + \alpha_2 f_M(v; U_2)}{\alpha_1 + \alpha_2 + \epsilon},$$

where f_M is the Maxwellian defined in (2.2) and α_1 and α_2 are uniformly random variables in $[0, 1]$ with $\epsilon = 10^{-6}$ preventing division zero.

The second set of training data is called the mixed initial conditions, which are created by taking wave data f_{wave} and superpositioning with shock data f_{shock}. The shock data are a Riemann problem where the interval $[x_1, x_2]$ has discontinuities at its endpoints, which are uniform random variables with $x_1 \in [0.2, 0.4]$ and $x_2 \in [0.6, 0.8]$. The primitive variables are given on the left side (ρ_L, u_L, θ_L) and the right side $\rho_R, u_L, \theta_R)$ with $u_R = u_L = 0$, ρ_L and θ_L being two independent random

TABLE 1
Parameters used to generate the training data for the HME model and kinetic model.

Parameters	HMEs	Kinetic
Spatial domain x	$[-0.5, -0.5]$	$[0, 1]$
Spatial mesh N_x	256	256
Time domain t	$[0, 10]$	$[0, 1]$
Time steps N_t	320	1000
Knudsen number	log-uniform $[10^{-3}, 10]$	log-uniform $[10^{-3}, 10]$

variables uniformly sampled from $[1, 2]$, and ρ_R and θ_R being two independent random variables uniformly sampled from $[0.55, 0.9]$. This shock condition is blended with smooth data as follows:

$$(3.3) \qquad f_{mix} = \alpha f_{wave} + (1 - \alpha) f_{shock},$$

where α is a random variable uniformly sampled from $[0.2, 0.4]$.

Following [14], we generate the training data to produce distribution functions with both smooth and discontinuous initial conditions and trajectories. These are a reasonable approximation to distributions of interest for the BGK equation with no external force. This method still limits the distributions seen in training by the frequency limits on the parameter used to generate the sinusoidal data in (3.1). Any model with external forces or associated fields, such as magneto-hydrodynamics, would need a different method to generate training data.

3.2. Neural network architecture. The networks we utilize have the following structure, which is depicted in Figure 1. Moments from training data are input into a fully connected, neural network. The network has a repeating structure of block layers: linear layer, activation layer, and batch normalization layer. The output of the network is the eigenvalues of matrix (2.10) with the last row as (2.13). This output is then passed to an implementation of Vieta's formula, which is passed to a linear transform that matches the structure required in the last row of the matrix (2.14). It is this final output that will be used in the nonconservative solver to close the system.

3.3. Training considerations and techniques. For the training data recovering the HME closure, the training techniques and neural network structure followed that of previous RTE work [15, 17, 16], with the modifications in section 2 for the BGK model. Table 2 summarizes hyperparameters for the HME closure training.

For the kinetic training data, both the neural network architecture and training methods were modified to overcome difficulties minimizing training and testing error:
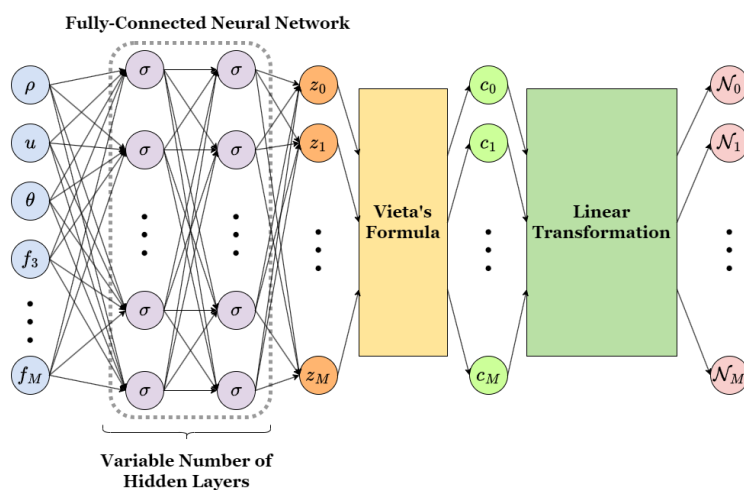

FIG. 1. *Neural network architecture from the left; the light blue f_k are moments from the training data set. The boxed, gray circles with σ represent the fully connected neural network; each column should be thought of as a block of layers (linear, activation, batch normalization). The yellow block applies Vieta's formula, and the final green appropriately transforms the output to coefficients for the moment gradients.*

TABLE 2
Summary of neural network training parameters for both HMEs and kinetic closures.

Parameters	HMEs	Kinetic
Training data	100 Initial Conditions (ICs)	800 ICs
Split train/test %	80/20	80/20
Activation function	Rectified linear unit (ReLU)	ReLU
Learning rate	schedule: `StepLR` rate: 10^{-7} decay factor: 0.5 step schedule: 100 epoch	schedule: `OneCycleLR` varies with count of moment input over epochs
Optimizer	`Adam`	`AdamW`
Loss term $\partial_x m_{N+1}^{\mathrm{Train}}(x_j, t_n)$	$-f_N \partial_x u - \frac{f_{N-1}}{2}\partial_x\theta$	$\partial_x m_{N+1}^{\mathrm{DVM}}(x_j, t_n)$

TABLE 3
Loss and errors, both in training and testing, for neural network of 9 layers, 128 neurons wide, and ReLU activation at the end of training with 500 epoch on kinetic smooth data.

Moments in input	Loss	Relative L^2 error (train)	Relative L^2 error (test)
5	1.49E-06	1.72E-01	1.76E-01
6	4.38E-08	1.02E-01	1.05E-01
7	1.22E-09	6.42E-02	6.55E-02
8	1.56E-10	9.46E-02	9.52E-02

specifically, in the optimizer used to minimize the loss function, cycling the learning rate to avoid local minimums and finally a change to the architecture with the inclusion of batch normalization layers to avoid vanishing and exploding gradients. We discuss each of these modifications individually. Table 2 summarizes hyperparameters for the kinetic closure training. Table 3 shows the associated training and testing error in the kinetic closure.

The loss function, $\mathcal{L}$, for training the neural network follows previous RTE work [15, 16] and has the form

$$(3.4) \qquad \mathcal{L} = \frac{1}{N_{data}} \sum_{j,n} |\partial_x f_{N+1}^{\mathrm{Train}}(x_j, t_n) - \partial_x f_{N+1}^{\mathrm{NN}}(x_j, t_n)|^2,$$

where term $\partial_x f_{N+1}^{\mathrm{Train}}(x_j, t_n)$ is moment gradients from either the HME or kinetic training data and $\partial_x f_{N+1}^{\mathrm{NN}} = \sum_{i=0}^{N} \mathcal{N}_i(\rho, u, \theta, f_3, \ldots, f_N)\partial_x f_i$ is the neural network approximation on the associated training set, as in (2.12). In the HME model case, $\partial_x f_{N+1}^{\mathrm{Train}}(x_j, t_n) = \partial_x f_{N+1}^{\mathrm{HME}} = -f_N\partial_x u - \frac{f_{N-1}}{2}\partial_x\theta$ and is computed from the generated training data $(u, \theta, f_{N-1}, f_N)$. For the kinetic BGK case, $\partial_x f_{N+1}^{\mathrm{Train}}(x_j, t_n) = \partial_x f_{N+1}^{\mathrm{DVM}}$.

Achieving good training errors for HME data was straightforward. However, several modifications were needed to achieve results with the kinetic data. These modifications are discussed in subsection 3.3.1. We outline the general training process in Algorithm 3.1 for both HME and kinetic training data.

3.3.1. `Adam` vs. `AdamW`. The RTE work [15, 17, 16] used the `Adam` [21] optimizer in PyTorch. This optimizer uses adaptive estimates for lower-order moments (mean, variance) of the gradient via exponential averaging and applying bias corrections.

The `AdamW` [31] optimizer, which has been also been used in recent moment closure work [29], modifies the regularization that was implemented in `Adam`. In Loshchilov

---

Algorithm 3.1. Train closure model.

Data: The $M + 1$ moments in primitive variables $(\rho, u, \theta, f_3, f_4, \ldots, f_M, f_{M+1})$ over the training time interval for every initial condition.

Input: A set of M moments in primitive variables $(\rho, u, \theta, f_3, f_4, \ldots, f_M)$.

Output: Coefficients on the gradient of the M moments, $\mathcal{N}_0, \mathcal{N}_1, \ldots, \mathcal{N}_M$

1 **begin** Prepare Training Set
2   **for** *all time steps within training time range, in all training data* **do**
3   Load: $M + 1$ moments, f_{M+1} from training data
4   Compute: $M + 1$ moment gradients, $\partial_x f_{M+1}$ from training data
5   **return** *Prepared training set, $W^{Training}$*

6 **begin** Training
7   **while** *under number of training epoch* **do**
8   Sample M moments, M gradients and $\partial_x f_{M+1}$ from $W^{Training}$
9   // *Neural network defined in Subsection* 3.2
10   Evaluate Neural Network: $\mathcal{NN}(\rho, u, \theta, f_3, f_4, \ldots, f_M) = (\mathcal{N}_0, \mathcal{N}_1, \ldots, \mathcal{N}_M)$
11   Compute: $\partial_x f_{M+1}^{\mathrm{NN}} = \sum_{i=0}^{M} \mathcal{N}_i(\rho, u, \theta, f_3, \cdots, f_M) \partial_x f_i$
12   Compute Loss: $\mathcal{L}(\partial_x f_{M+1}^{\mathrm{NN}}, \partial_x f_{M+1})$
13   Back-propagation to optimize

TABLE 4

Comparison of relative training error using the Adam and AdamW optimizer in PyTorch for a 9-layer, 128-neuron–wide, ReLU activated neural network with batch normalization. This is for training a neural network closure on kinetic data and shows the improvement of switching optimizers.

Moments in input	`Adam` relative L^2 error (train)	`AdamW` relative L^2 error (train)
6	1.23E-01	1.02E-01
7	1.17E-01	6.42E-02
8	1.06E-01	9.46E-02

and Hutter [31], it was argued that weight decay and L^2 regularization are not equivalent for adaptive gradient descent methods but are equivalent for stochastic gradient descent. `AdamW` separates moment estimates from the decay of the parameters, decoupling the optimization of the loss function from the weight decay. Empirically, `AdamW` has shown improved performance over wider array of hyperparameter selection.

By using `AdamW` over `Adam` for training our neural network closure, we were able to improve overall training error significantly, as shown in Table 4. This was needed for the kinetic training data.

3.3.2. Adding batch normalization. To further improve training errors in the neural network closure, a batch normalization layer was added in the hidden layers between the linear and activation layers. Batch normalization [18] is a commonly used method in deep learning. This process normalizes the batch input to the layer to have mean-zero, variance-one by tracking running mean and variance over the course of training. In Figure 1, the normalization layers can be thought of as being contained in the layers with σ. Adding the batch normalization layer made optimal learning rate ranges more uniform across the number of moments input, as seen in Figure 2(a), as well as created a large region of decreasing loss that stays relatively consistent over
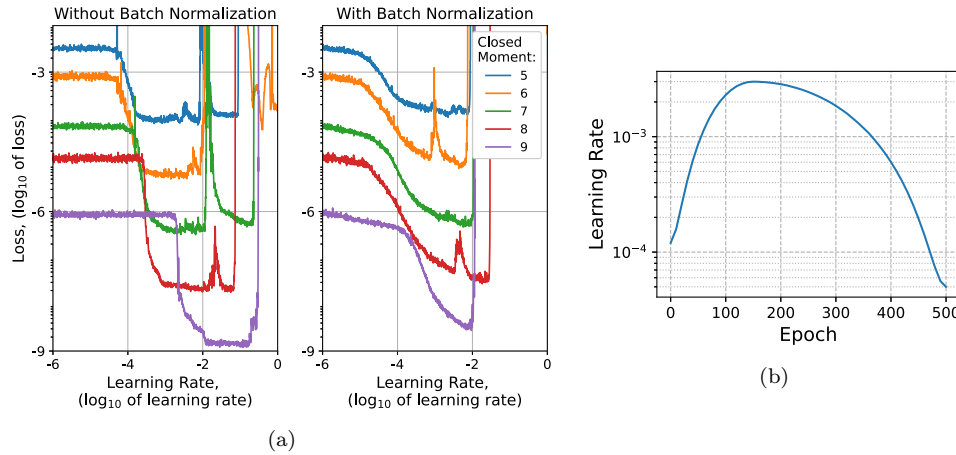
FIG. 2. *Examples of learning rate range tests with the effects of batch normalization and a sample one-cycle learning rate schedule.* (*a*) *Representative learning rate range for a* 9-*layer,* 256-*neuron–wide network that is trained on kinetic smooth/wave initial conditions. The horizontal axes are the learning rate, and the vertical axes are the loss. Each color represents the moment that is being closed by the network. On the left, the networks have no batch normalization layer, and on the right, the networks do. We can see that batch normalization (right) gave overlapping regions of loss decrease that were spread over a wider range of learning rates compared to networks without.* (*b*) *An example of the one cycle learning rate variation. The horizontal axis is the epoch of training, and the vertical axis is the learning rate.*

changing the number of moments that are input to the network. Also, this batch normalization layer also improves training errors during training.

3.3.3. Learning rate cycling. During training, there was concern that the neural network closure was settling in a local minimum rather than finding a global minimum. The initial implementation followed the RTE work [16] with a halving schedule for the learning rate hyperparameter over a fixed number of epochs.

For the BGK neural network closure, we implemented a one-cycle learning rate schedule, where the learning rate is raised and then lowered in a cycle following a cosine function [41]. The learning rate changes occurs over each iteration of the optimizer, that is, after each batch of training data rather than over epoch, which would be once through the entire training data set. A sample one-cycle learning rate schedule is show in Figure 2(b).

This method requires finding optimal maximum learning rates that do not cause the training to diverge in an unrecoverable way from the minimum. This is done using the so-called learning rate range test (LRRT) [40, 11]. This process is useful for finding reasonable learning rate values for training in general. The LRRT varies the learning rate every training iteration over a range (orders of magnitude) of learning rates and records the loss. Usually revealed is a plateau where the learning rate is too small to approach the minimum and loss is nearly constant, followed by a region of decreasing loss that ends with a steep jump in loss as the learning rate becomes too large to resolve the minimum. By choosing a maximum learning rate that occurs at the end of the decreasing loss region and a minimum learning rate that occurs before the loss region, training will remain within this loss region while using a one-cycle learning rate schedule.

There are drawbacks to using this method. The LRRT must be done prior to training a given neural network with a given hyperparameter configuration. Also, if

the maximum learning rate is chosen too far away from the end of the decrease, we have observed a nonrecoverable divergence away from a global minimum. However, this method does give a relatively good indication of where optimal learning rates are located.

4. Numerical method. In this section, we briefly review the first-order and high-order path-conservative numerical methods [6] for solving the nonconservative hyperbolic moment system. We will show the formulation of the path-conservative method. It is important to note that we make use of a partially conservative transformation. This change of variables places the first Mth moments in conservative form and the last moment in a nonconservative form, as demonstrated in [25, 22]. The transformation to partially conserved variables greatly improves the performance of the path-conservative method.

For simplicity, we adopt a uniform spatial discretization of domain $[x_a, x_b]$,

$$x_a = x_{\frac{1}{2}} < x_{\frac{3}{2}} \cdots < x_{N_x - \frac{1}{2}} < x_{N_x + \frac{1}{2}} = x_b,$$

where $x_{j-\frac{1}{2}} = x_a + j\Delta x, j = 1, 2, \ldots, N_x + 1$ with mesh size $\Delta x = \frac{x_b - x_a}{N_x}$. The cells and the midpoints are denoted by

$$I_j = [x_{j-\frac{1}{2}}, x_{j+\frac{1}{2}}], \quad x_j = \frac{1}{2}(x_{j-\frac{1}{2}} + x_{j+\frac{1}{2}}), \quad j = 1, 2, \ldots N_x.$$

We denote $\overline{\mathbf{w}}_j^n = \frac{1}{\Delta x} \int_{x_{j-\frac{1}{2}}}^{x_{j+\frac{1}{2}}} \mathbf{w}(x, t^n) \, dx$ as the cell average value of $\mathbf{w}(x, t)$ at time $t^n = n\Delta t$ with Δt being the time stepping size.

4.1. First-order path-conservative scheme. In the case of systems of hyperbolic conservation laws, when $A = \frac{\partial F}{\partial \mathbf{w}}$ is a Jacobian matrix of a flux function $F(\mathbf{w})$, (2.9) can be written as

$$\frac{\partial \mathbf{w}}{\partial t} + \frac{\partial F(\mathbf{w})}{\partial x} = 0. \tag{4.1}$$

Then, a first-order conservative wave propagation method [27] can be applied to (4.1),

$$\overline{\mathbf{w}}_j^{n+1} = \overline{\mathbf{w}}_j^n - \frac{\Delta t}{\Delta x} \left(\mathcal{A}_{j-\frac{1}{2}}^+ \Delta \mathbf{w}_{j-\frac{1}{2}}^n + \mathcal{A}_{j+\frac{1}{2}}^- \Delta \mathbf{w}_{j+\frac{1}{2}}^n \right) \tag{4.2}$$

with jump $\Delta \mathbf{w}_{j\pm\frac{1}{2}}^n = \mathbf{w}_{j\pm\frac{1}{2}}^{n,+} - \mathbf{w}_{j\pm\frac{1}{2}}^{n,-}$. Here, $\mathbf{w}_{j\pm\frac{1}{2}}^{n,\pm}$ are the left and right limits of solution $\mathbf{w}^n$ at the cell boundaries $x_{j\pm\frac{1}{2}}$. In particular, $\mathbf{w}_{j-\frac{1}{2}}^{n,+}$ can be approximated using the cell average sequence from the neighboring cells $\{\overline{\mathbf{w}}_{j-q}^n, \ldots, \overline{\mathbf{w}}_{j-p}^n\}$ for $q, p \geq 0$. In (4.2), it is sufficient for us to achieve first-order accuracy taking $\mathbf{w}_{j-\frac{1}{2}}^{n,+} = \overline{\mathbf{w}}_j$ and $\mathbf{w}_{j-\frac{1}{2}}^{n,-} = \overline{\mathbf{w}}_{j-1}$; thus, $\Delta \mathbf{w}_{j-\frac{1}{2}}^n = \overline{\mathbf{w}}_j^n - \overline{\mathbf{w}}_{j-1}^n$ and $\Delta \mathbf{w}_{j+\frac{1}{2}}^n = \overline{\mathbf{w}}_{j+1}^n - \overline{\mathbf{w}}_j^n$ in (4.2).

When $F(\mathbf{w}) = A\mathbf{w}$ is a linear function, $\mathcal{A} = A$, we have $A_{j\pm\frac{1}{2}}^\pm$ in (4.2) defined by

$$A_{j\pm\frac{1}{2}}^+ = R_{j\pm\frac{1}{2}} \Lambda_{j\pm\frac{1}{2}}^+ R_{j\pm\frac{1}{2}}^{-1}, \quad A_{j\pm\frac{1}{2}}^- = R_{j\pm\frac{1}{2}} \Lambda_{j\pm\frac{1}{2}}^- R_{j\pm\frac{1}{2}}^{-1}, \tag{4.3}$$

where $R_{j\pm\frac{1}{2}}$ is a $(M+1) \times (M+1)$ matrix storing the right eigenvectors of $A_{j\pm\frac{1}{2}}$, $\Lambda_{j\pm\frac{1}{2}}^\pm = \text{diag}((\lambda_{j+\frac{1}{2}}^1)^\pm, (\lambda_{j+\frac{1}{2}}^2)^\pm, \ldots (\lambda_{j+\frac{1}{2}}^{M+1})^\pm)$ with positive or negative eigenvalues of $A_{j\pm\frac{1}{2}}$, respectively. When $F(\mathbf{w})$ is nonlinear, the system shock speed at a discontinuity is given by Rankine–Hugoniot conditions.

Because A in (2.10) and (2.13) cannot be expressed as a Jacobian matrix, we follow the definition of the nonconservative product $A(\mathbf{w})\mathbf{w}_x$ in [8]. For the left and right limits $\mathbf{w}^-, \mathbf{w}^+$ of $\mathbf{w} \in \Omega$ at the discontinuity, consider a family of Lipschitz continuous path $\phi(s; \mathbf{w}^-, \mathbf{w}^+) : [0,1] \times \Omega \times \Omega \to \Omega$ such that

$$\phi(0; \mathbf{w}^-, \mathbf{w}^+) = \mathbf{w}^-, \quad \phi(1; \mathbf{w}^-, \mathbf{w}^+) = \mathbf{w}^+, \quad \phi(s; \mathbf{w}, \mathbf{w}) = \mathbf{w},$$

where $\phi(s; \mathbf{w}^-, \mathbf{w}^+)$ can be seen as a parameterization of the integral curve connecting $\mathbf{w}^-, \mathbf{w}^+$. In [25], there exist many different choices of paths ϕ connecting W_L and W_R, such as a simple linear path

$$(4.4) \qquad \phi(s; \mathbf{w}^-, \mathbf{w}^+) = \mathbf{w}^- + s \cdot (\mathbf{w}^+ - \mathbf{w}^-), \quad s \in [0,1]$$

and polynomial paths ϕ_-^N and ϕ_+^N:

$$(4.5) \qquad \begin{aligned} \phi_-^N(s; \mathbf{w}^-, \mathbf{w}^+) &= \mathbf{w}^- + s^N \cdot (\mathbf{w}^+ - \mathbf{w}^-), \\ \phi_+^N(s; \mathbf{w}^-, \mathbf{w}^+) &= \mathbf{w}^+ + (s-1)^N \cdot (\mathbf{w}^- - \mathbf{w}^+). \end{aligned}$$

In [6], given a family of paths ϕ, a numerical scheme is said to be ϕ-conservative if it can be written under the form

$$(4.6) \qquad \overline{\mathbf{w}}_j^{n+1} = \overline{\mathbf{w}}_j^n - \frac{\Delta t}{\Delta x} \left(D_{j-\frac{1}{2}}^+ + D_{j+\frac{1}{2}}^- \right),$$

where $D_{j\pm\frac{1}{2}}^\pm = D^\pm(\mathbf{w}_{j\pm\frac{1}{2}}^{n,-}, \mathbf{w}_{j\pm\frac{1}{2}}^{n,+})$ with D^- and D^+ being two continuous functions satisfying

$$(4.7) \qquad D^\pm(\mathbf{w}, \mathbf{w}) = 0 \quad \text{for all } \mathbf{w} \in \Omega$$

and

$$(4.8) \qquad \begin{aligned} &D^- \left(\mathbf{w}_{j\pm\frac{1}{2}}^{n,-}, \mathbf{w}_{j\pm\frac{1}{2}}^{n,+} \right) + D^+ \left(\mathbf{w}_{j\pm\frac{1}{2}}^{n,-}, \mathbf{w}_{j\pm\frac{1}{2}}^{n,+} \right) \\ &\qquad = \int_0^1 A \left(\phi \left(s; \mathbf{w}_{j\pm\frac{1}{2}}^{n,-}, \mathbf{w}_{j\pm\frac{1}{2}}^{n,+} \right) \right) \frac{\partial \phi}{\partial s} \left(s; \mathbf{w}_{j\pm\frac{1}{2}}^{n,-}, \mathbf{w}_{j\pm\frac{1}{2}}^{n,+} \right) \, ds. \end{aligned}$$

Consider a Roe linearization $\mathcal{A}_\phi(\mathbf{w}_{j\pm\frac{1}{2}}^-, \mathbf{w}_{j\pm\frac{1}{2}}^+)$ satisfying

$$(4.9)$$
$$\mathcal{A}_\phi \left(\mathbf{w}_{j\pm\frac{1}{2}}^-, \mathbf{w}_{j\pm\frac{1}{2}}^+ \right) \Delta \mathbf{w}_{j\pm\frac{1}{2}} = \int_0^1 A \left(\phi \left(s; \mathbf{w}_{j\pm\frac{1}{2}}^-, \mathbf{w}_{j\pm\frac{1}{2}}^+ \right) \right) \frac{\partial \phi}{\partial s} \left(s; \mathbf{w}_{j\pm\frac{1}{2}}^-, \mathbf{w}_{j\pm\frac{1}{2}}^+ \right) \, ds,$$

and let $D^\pm(\mathbf{w}_{j\pm\frac{1}{2}}^{n,-}, \mathbf{w}_{j\pm\frac{1}{2}}^{n,+}) = \mathcal{A}_\phi^\pm \Delta \mathbf{w}_{j\pm\frac{1}{2}}$, the first-order path-conservative scheme for (2.9), be given as

$$(4.10) \qquad \overline{\mathbf{w}}_j^{n+1} = \overline{\mathbf{w}}_j^n - \frac{\Delta t}{\Delta x} \left(\mathcal{A}_\phi^+ \Delta \mathbf{w}_{j-\frac{1}{2}} + \mathcal{A}_\phi^- \Delta \mathbf{w}_{j+\frac{1}{2}} \right).$$

When using (4.4), (4.9) reduces to

$$(4.11) \qquad \mathcal{A}_\phi \left(\mathbf{w}_{j\pm\frac{1}{2}}^-, \mathbf{w}_{j\pm\frac{1}{2}}^+ \right) = \int_0^1 A \left(\phi \left(s; \mathbf{w}_{j\pm\frac{1}{2}}^-, \mathbf{w}_{j\pm\frac{1}{2}}^+ \right) \right) \, ds,$$

which could be computed via

$$(4.12) \qquad \mathcal{A}_\phi\left(\mathbf{w}^-_{j\pm\frac{1}{2}}, \mathbf{w}^+_{j\pm\frac{1}{2}}\right) \approx \sum_{i=1}^{p} \omega_i A\left(\phi\left(s_i, \mathbf{w}^-_{j\pm\frac{1}{2}}, \mathbf{w}^+_{j\pm\frac{1}{2}}\right)\right)$$

using the high-order quadrature rule with points $\{s_i\}$ and the corresponding weights $\{\omega_i\}$. When using (4.5), $\mathcal{A}_\phi$ can be computed as

$$(4.13) \qquad \mathcal{A}_\phi\left(\mathbf{w}^-_{j\pm\frac{1}{2}}, \mathbf{w}^+_{j\pm\frac{1}{2}}\right) = \int_0^1 A\left(\phi\left(s; \mathbf{w}^-_{j\pm\frac{1}{2}}, \mathbf{w}^+_{j\pm\frac{1}{2}}\right)\right) \cdot n \cdot s^{n-1}\ ds$$

$$(4.14) \qquad \approx \sum_{i=1}^{p} \omega_i A\left(\phi\left(s_i; \mathbf{w}^-_{j\pm\frac{1}{2}}, \mathbf{w}^+_{j\pm\frac{1}{2}}\right)\right) \cdot n \cdot s_i^{n-1}$$

via high-order quadrature rules.

4.2. High-order path-conservative scheme. We again start from (4.1); a conservative semidiscrete method is given as

$$(4.15) \qquad \mathbf{w}'_j(t) = -\frac{1}{\Delta x}\left(G_{j+\frac{1}{2}}\left(\mathbf{w}^-_{j+\frac{1}{2}}(t), \mathbf{w}^+_{j+\frac{1}{2}}(t)\right) - G_{j-\frac{1}{2}}\left(\mathbf{w}^-_{j-\frac{1}{2}}(t), \mathbf{w}^+_{j-\frac{1}{2}}(t)\right)\right)$$

with the numerical flux

$$(4.16a) \qquad G_{j\pm\frac{1}{2}}\left(\mathbf{w}^-_{j\pm\frac{1}{2}}(t), \mathbf{w}^+_{j\pm\frac{1}{2}}(t)\right) = F\left(\mathbf{w}^+_{j\pm\frac{1}{2}}(t)\right) - \mathcal{A}^+_{j\pm\frac{1}{2}}\Delta\mathbf{w}_{j\pm\frac{1}{2}}(t)$$

$$(4.16b) \qquad = F\left(\mathbf{w}^-_{j\pm\frac{1}{2}}(t)\right) + \mathcal{A}^-_{j\pm\frac{1}{2}}\Delta\mathbf{w}_{j\pm\frac{1}{2}}(t).$$

Note that (4.16a) and (4.16b) are obtained from

$$F\left(\mathbf{w}^+_{j\pm\frac{1}{2}}(t)\right) - F\left(\mathbf{w}^-_{j\pm\frac{1}{2}}(t)\right) = \mathcal{A}_{j\pm\frac{1}{2}}\Delta\mathbf{w}_{j\pm\frac{1}{2}}(t).$$

Plug (4.16) into (4.15); we have

(4.17a)

$$\mathbf{w}'_j(t) = -\frac{1}{\Delta x}\left[\mathcal{A}^+_{j-\frac{1}{2}}\Delta\mathbf{w}_{j-\frac{1}{2}}(t) + \mathcal{A}^-_{j+\frac{1}{2}}\Delta\mathbf{w}_{j+\frac{1}{2}}(t) + F(\mathbf{w}^-_{j-\frac{1}{2}}(t)) - F(\mathbf{w}^+_{j-\frac{1}{2}}(t))\right]$$

(4.17b)

$$= -\frac{1}{\Delta x}\left[\mathcal{A}^+_{j-\frac{1}{2}}\Delta\mathbf{w}_{j-\frac{1}{2}}(t) + \mathcal{A}^-_{j+\frac{1}{2}}\Delta\mathbf{w}_{j+\frac{1}{2}}(t) + \int_{x_{j-\frac{1}{2}}}^{x_{j+\frac{1}{2}}} A(P_j(x))\frac{d}{dx}P_j(x)\ dx\right]$$

with smooth function $P_j(x)$ on cell I_j satisfying

$$(4.18) \qquad \lim_{x\to x^+_{j-\frac{1}{2}}} P_j(x) = \mathbf{w}^+_{j-\frac{1}{2}}(t) \qquad \lim_{x\to x^-_{j+\frac{1}{2}}} P_j(x) = \mathbf{w}^-_{j+\frac{1}{2}}(t).$$

Equation (4.17) leads to a ϕ-conservative scheme for (2.9):

(4.19)

$$\mathbf{w}'_j(t) = -\frac{1}{\Delta x}\left[\mathcal{A}^+_\phi\Delta\mathbf{w}_{j-\frac{1}{2}}(t) + \mathcal{A}^-_\phi\Delta\mathbf{w}_{j+\frac{1}{2}}(t) + \int_{x_{j-\frac{1}{2}}}^{x_{j+\frac{1}{2}}} A(P_j(x))\frac{d}{dx}P_j(x)\ dx\right],$$

which can be written as

$$\mathbf{w}'_j(t) = -\frac{1}{\Delta x}\left(D^+_{j-\frac{1}{2}} + D^-_{j+\frac{1}{2}}\right)$$

with

$$(4.20) \qquad D^+_{j-\frac{1}{2}} = \mathcal{A}^+_\phi \Delta \mathbf{w}_{j-\frac{1}{2}}(t) + \int_{x_{j-\frac{1}{2}}}^{x_j} A(P_j(x))\frac{d}{dx}P_j(x)\ dx,$$

$$(4.21) \qquad D^-_{j+\frac{1}{2}} = \mathcal{A}^-_\phi \Delta \mathbf{w}_{j+\frac{1}{2}}(t) + \int_{x_j}^{x_{j+\frac{1}{2}}} A(P_j(x))\frac{d}{dx}P_j(x)\ dx,$$

and

$$\begin{aligned}
D^-_{j+\frac{1}{2}} + D^+_{j+\frac{1}{2}} &= \int_{x_j}^{x_{j+\frac{1}{2}}} A(P_j(x))\frac{d}{dx}P_j(x)\ dx \\
&\quad + \mathcal{A}_\phi \Delta \mathbf{w}_{j+\frac{1}{2}}(t) \\
&\quad + \int_{x_{j+\frac{1}{2}}}^{x_{j+1}} A(P_{j+1}(x))\frac{d}{dx}P_{j+1}(x)\ dx \\
&= \int_{x_j}^{x_{j+\frac{1}{2}}} A(P_j(x))\frac{d}{dx}P_j(x)\ dx \\
&\quad + \int_0^1 A(\phi(s;\mathbf{w}^-_{j+\frac{1}{2}}(t),\mathbf{w}^+_{j+\frac{1}{2}}(t)))\frac{\partial \phi}{\partial s}(s;\mathbf{w}^-_{j+\frac{1}{2}}(t),\mathbf{w}^+_{j+\frac{1}{2}}(t))\ ds \\
&\quad + \int_{x_{j+\frac{1}{2}}}^{x_{j+1}} A(P_{j+1}(x))\frac{d}{dx}P_{j+1}(x)\ dx.
\end{aligned}$$

In our implementation, the integral term $\int_{x_{j-\frac{1}{2}}}^{x_{j+\frac{1}{2}}} A(P_j(x))\frac{d}{dx}P_j(x)\ dx$ in (4.19) is approximated via a sixth-order Gauss–Lobatto quadrature rule

$$(4.22) \qquad \int_{x_{j-\frac{1}{2}}}^{x_{j+\frac{1}{2}}} A(P_j(x))\frac{d}{dx}P_j(x)\ dx \approx \sum_{i=1}^4 w_i A(P_j(s_i \Delta x + x_j))\frac{d}{dx}P_j(s_i \Delta x + x_j)$$

with quadrature nodes $\{s_i\}_{i=1}^4 \in [-\frac{1}{2},\frac{1}{2}]$ and their associated weights $\{w_i\}_{i=1}^4$. More specifically, the approximated point values $\{\mathbf{w}^+_{j-\frac{1}{2}} = P_j(x_{j-\frac{1}{2}}) = P_j(s_1\Delta x + x_j),$ $P_j(s_2\Delta x + x_j), P_j(s_3\Delta x + x_j), \mathbf{w}^-_{j+\frac{1}{2}} = P_j(x^-_{j+\frac{1}{2}}) = P_j(s_4\Delta x + x_j)\}$ over I_j can be reconstructed using a fifth-order–accurate WENO method [39] from the cell averages $\{\overline{\mathbf{w}}_i, i = j-2, j, j+2\}$. $\{\frac{d}{dx}P_j(x)|x = s_i\Delta x + x_j, i = 1,\ldots,4\}$ can then be approximated by the derivative of polynomial interpolating through $\{P_j(x)|x = s_i\Delta x + x_j, i = 1,\ldots,4\}$ over I_j.

Note that (4.19) becomes a system of ODEs after the spatial discretization to the right-hand side (rhs). For the time integration of (4.19) on the left-hand side, we use a standard third-order strong stability preserving Runge–Kutta (RK3) method [12].

5. Numerical results. In this section, we present numerical calculations using the neural network closures for both HME-trained and kinetic-trained models for the moment system and associated errors.

For the kinetic-trained closure, we are making an assumption on the form of the closure, which is an inherent model error. For HME-trained closure, the closure model is exact, so the only error present is related to the ML errors of training and stochastic

Algorithm 5.1. Computing with the trained closure model. Solve the nonconservative moment system $\frac{\partial \mathbf{w}}{\partial t} + A^{\mathrm{ML}} \frac{\partial \mathbf{w}}{\partial x} = \mathbf{Q}$ using trained neural network closure to complete the last row of A^{ML} and using RK3 and the nonconservative schemes described in section 4. The variables $\mathfrak{a}, \mathfrak{b}, \mathfrak{c}$ are the coefficients for the Runge–Kutta scheme.

Input:
  Moments in primitive variables: $\mathbf{w}^0 = \left(\rho^0, u^0, \theta^0, f_3^0, f_4^0, \ldots, f_M^0 \right)$ at $t = 0$.
  Trained Neural Network Closure: $\mathcal{NN}$
Output: Moments in primitive variables: $\mathbf{w}^{T_{final}} = (\rho, u, \theta, f_3, f_4, \ldots, f_M)$ at $t = T_{final}$.

1 Form A^{ML} with last row $\leftarrow \mathcal{NN}(\mathbf{w}^0) = \begin{pmatrix} a_0 & a_1 & a_2 & a_3 & \cdots & a_{M-1} & a_M \end{pmatrix}$ Equation (2.13)

2 $\mathbf{w}^* \leftarrow \mathbf{w}^0$

3 **while** $t < T_{final}$ **do**

4 | // First Stage

5 | Compute: A^{ML} using $\mathcal{NN}(\mathbf{w}^*)$ for last row.

6 | Compute: maximum wave speeds and CFL from A^{ML}

7 | Compute: $\mathbf{rhs}^1$, the right-hand side of $\frac{\partial \mathbf{w}}{\partial t} = -A^{\mathrm{ML}} \frac{\partial \mathbf{w}}{\partial x} + \mathbf{Q}$ for $\mathbf{w}^*$ using nonconservative methods in Section 4.

8 | $\mathbf{w}^1 \leftarrow \mathbf{w}^* - \Delta t \cdot \mathfrak{a}_{1,0} \cdot \mathbf{rhs}^1$

9 | $t \leftarrow t + \mathfrak{c}_1 \cdot \Delta t$

10 | // Second stage

11 | Compute: A^{ML} using $\mathcal{NN}(\mathbf{w}^1)$ for last row.

12 | Compute: maximum wave speeds and CFL from A^{ML}

13 | Compute: $\mathbf{rhs}^2$, the right-hand side of $\frac{\partial \mathbf{w}}{\partial t} = -A^{\mathrm{ML}} \frac{\partial \mathbf{w}}{\partial x} + \mathbf{Q}$ for $\mathbf{w}^1$ using nonconservative methods in Section 4.

14 | $\mathbf{w}^2 \leftarrow \mathbf{w}^* - \Delta t \cdot \left(\mathfrak{a}_{2,0} \cdot \mathbf{rhs}^1 + \mathfrak{a}_{2,1} \cdot \mathbf{rhs}^2 \right)$

15 | $t \leftarrow t + \mathfrak{c}_2 \cdot \Delta t$

16 | // Third stage

17 | Compute: A^{ML} using $\mathcal{NN}(\mathbf{w}^2)$ for last row.

18 | Compute: maximum wave speeds and CFL from A^{ML}

19 | Compute: $\mathbf{rhs}^3$, the right-hand side of $\frac{\partial \mathbf{w}}{\partial t} = -A^{\mathrm{ML}} \frac{\partial \mathbf{w}}{\partial x} + \mathbf{Q}$ for $\mathbf{w}^2$ using nonconservative methods in Section 4.

20 | $\mathbf{w}^3 \leftarrow \mathbf{w}^* - \Delta t \cdot \left(\mathfrak{b}_0 \cdot \mathbf{rhs}^1 + \mathfrak{b}_1 \cdot \mathbf{rhs}^2 + \mathfrak{b}_2 \cdot \mathbf{rhs}^3 \right)$

21 | $t \leftarrow t + \Delta t$

22 | $\mathbf{w}^* \leftarrow \mathbf{w}^3$

---

randomness associated with it. The purpose of the presenting the HME results is to demonstrate that the ML approach to the closure problem is viable.

An outline of process to compute solutions using the trained neural network closure is given in Algorithm 5.1.

5.1. Realization of HME model using neural network. The neural network recovered the HME model as shown in Figure 3 for wave training data and in Figure 4(b) for mix training data, with the data generated as described in subsection 3.3. For wave-trained HME closure, there was very good agreement during numerical prediction with wave initial conditions, as seen in Figure 3. When mix initial conditions were used for prediction with a closure trained on wave data, the closure had
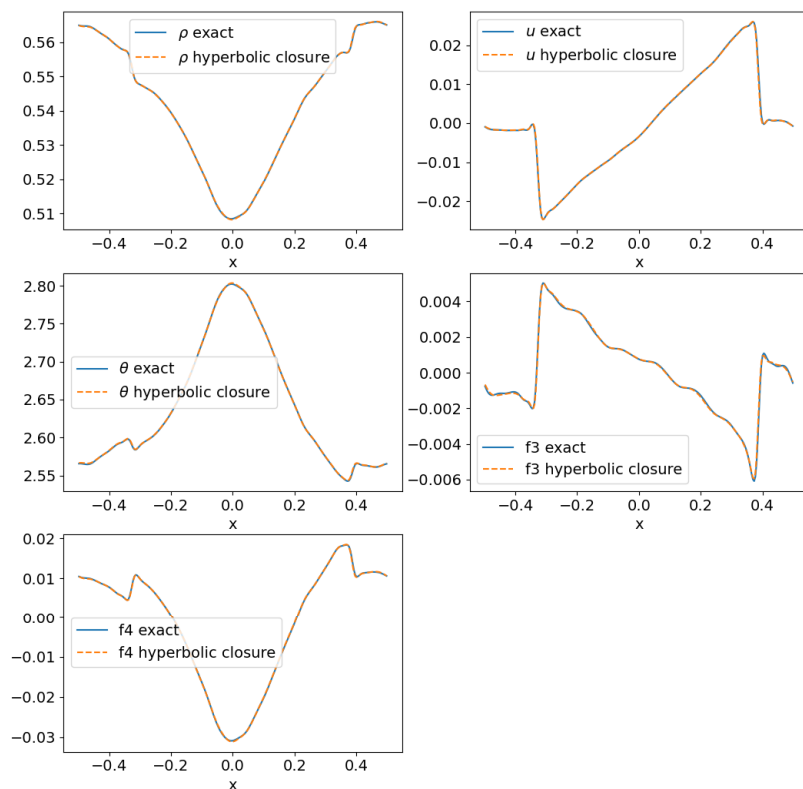
FIG. 3. *HME-trained closure: Sample of* 5 *moments* $(\rho, u, \theta, f_3, f_4)$ *for Knudsen number* $\tau = 1$ *at* $t = 10$ *beyond the training data time range, trained on smooth, wave data, then used to predict moments on smooth initial conditions.*

difficulty recovering moments as seen in Figure 4(a). However, when using mix initial conditions with a closure trained on 50% wave, 50% mix data, the HME model was recovered well, as seen in Figure 4(b). The robust performance of the NN closure trained on HME data, as demonstrated in Figure 3 and Figure 4(b), is universal and representative of its effectiveness across various Knudsen number regimes.

Recovery of the HME model by the trained network occurred over the range of Knudsen numbers tested for both smooth/wave and mix data as seen in Figure 5(a) and (b).

5.2. Kinetic-trained neural network closure results. After training neural networks on moment data generated as described in section 3, we then use this closure in the solver, using the FORCE method [22, 6], to predict moments using initial conditions that were not part of the training data set but generated in the same manner. The results are at $t = 0.3$, which is beyond the training set time interval of $t = 0.05$ to $t = 0.15$. Figure 6 shows individual samples of the first three moments corresponding to the primitive variables of density, velocity, and temperature across the range of Knudsen numbers at the various times. These compare the computed moments using the neural network closure and the moment data at matching times using the DVM in section 3. In addition, to ensure the reproducibility of numerical experiments, in Table 5, we present the parameters for smooth, wave kinetic initial conditions that are used in the numerical predictions shown in Figure 6. Figure 7
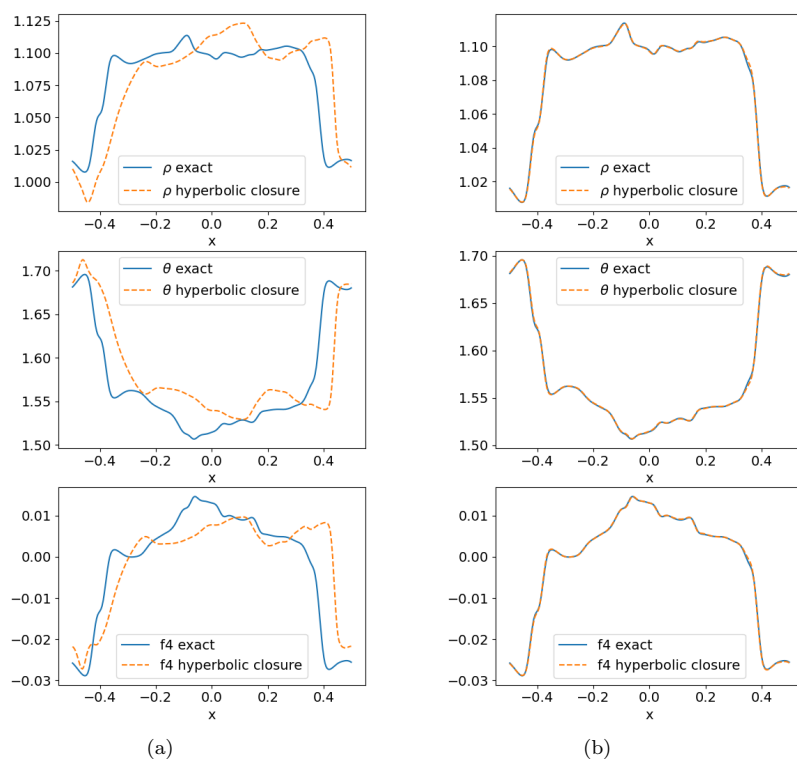
FIG. 4. *HME-trained closure: Sample from* 5*-moment model* (ρ, θ, f_4) *for Knudsen number* $\tau = 1$ *at time* $t = 10$ *beyond the training data time range.* (a) *HME closure trained on smooth, wave data, then used to predict moments on mixed initial conditions.* (b) *HME closure trained on a training set consisting of* 50% *smooth data and* 50% *mix data.*
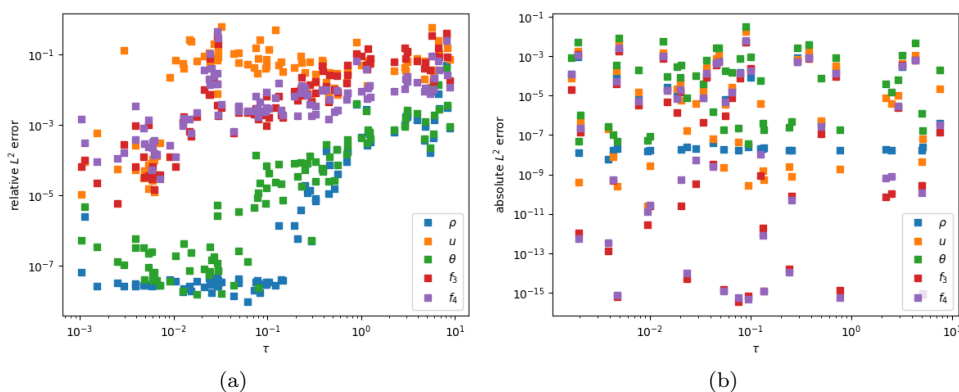

FIG. 5. *HME-trained closure: Errors for Knudsen numbers at* $t = 10$, *beyond the training data time range.* (a) *relative* L^2 *error in* $\rho, u, \theta, f_3, f_4$ *at* $t = 10$, *beyond the training data time range for HME-trained NN closure on smooth/wave data.* (b) *absolute* L^2 *error in* $\rho, u, \theta, f_3, f_4$ *at* $t = 10$, *beyond the training data time range for HME-trained NN closure on mix data.*

shows the relative L^2 error in the primitive variables at the computed times for all initial conditions in this test set.

Generally, we see that, within the training set time interval, we have good agreement between between neural network closure moments and this test set for the highly
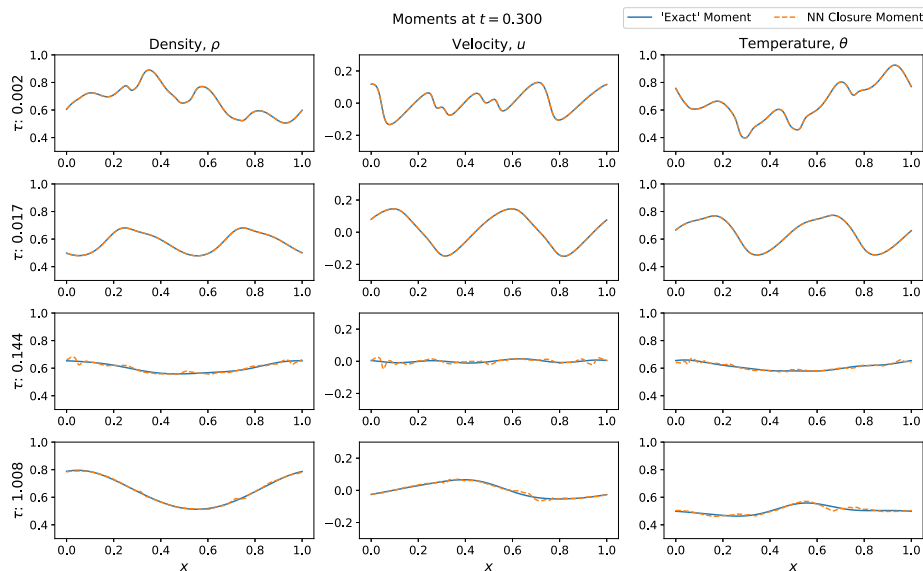
FIG. 6. *Kinetic-trained closure: First three moments ρ, u, θ for various Knudsen numbers at $t = 0.3$ beyond the training data time range from an NN of 9 layers, 128 neurons wide, using ReLU with batch normalization for smooth initial conditions. Each column corresponds to a moment going left to right: ρ, u, θ. Each row is a different Knudsen number, going top to bottom: 0.002, 0.017, 0.144, 1.008.*

collisional range of Knudsen numbers, $[10^{-3}, 10^{-1})$. When computing with trained closures within the training time window, we have very good agreement across the range of Knudsen numbers, with relative L^2 error not exceeding 10%. As we move beyond the training window, results in the transition and free streaming range of Knudsen numbers $[10^{-1}, 10^1]$ vary greatly, as seen in Figure 7. There also appears to be larger error in the macroscopic velocity, u, as well as clustering of moments of different order. This might be due to the parity of the moments, where ρ and θ are even powered and u is odd powered. This could be an avenue of future investigation.

To demonstrate the advantage of our hyperbolic neural network (NN) closure, we train a nonhyperbolic, kinetic NN closure that learns the gradient coefficients as in our previous work [15]. The prediction results with the Knudsen number 10^{-3} and $t = 0.3$ are shown in Figure 8. We observe that the profiles of the lower-order moments ρ, u, and θ for the nonhyperbolic closure agree well with the kinetic data. At the same time, the oscillations appear in the higher-order moments f_3 and f_4. This is due to the loss of hyperbolicity in the ML model. As a comparison, the predictions of our hyperbolic NN closure align well with the kinetic data for all the moments. Additionally, it is worth noting that the oscillations increase over time and worsen with increasing Knudsen numbers, although these results are omitted here due to the page limit.

We also numerically test the Galilean invariance of our ML model. We train the NN using the data set with the macroscopic velocity of $u = 0$. Then, we use the ML model to predict the solution profiles with an initial condition with $u = 0.1$. The profiles of density, velocity, and temperature at $t = 0.3$ are presented in Figure 9. The results of the ML model agree well with those from the kinetic equation. This shows that our NN generates closures, which preserves Galilean invariance.

TABLE 5

Parameters for smooth, wave kinetic initial conditions, described in subsection 3.1, *that are used in the numerical predictions shown in Figure* 6.

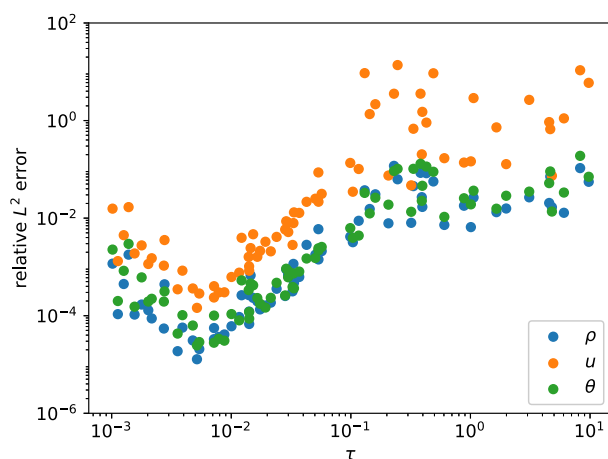| Knudsen	Density				Temperature				Mixing
number	a_ρ	b_ρ	k_ρ	ϕ_ρ	a_θ	b_θ	k_θ	ϕ_θ	α
0.001554	0.261888	0.270198	4	0.347195	0.228545	0.665582	1	2.708624	0.438211
	0.270198	0.627907	1	5.5624	0.223145	0.622083	4	1.573187	0.596517
0.016515	0.222878	0.655877	1	0.484743	0.213417	0.574442	3	3.280153	0.066066
	0.229181	0.585005	2	4.482850	0.253143	0.639633	2	5.905965	0.981557
0.143742	0.246574	0.574064	3	3.953562	0.254433	0.253392	1	4.545444	0.589359
	0.250002	0.686808	4	4.205230	0.253392	0.525255	3	4.694920	0.384366
1.008160	0.224907	0.614791	2	5.343272	0.251982	0.569712	2	0.829738	0.34663
	0.219768	0.68036	1	1.102718	0.250576	0.514169	1	4.491626	0.639318

FIG. 7. *Kinetic-trained closure: Relative L^2 error in ρ, u, θ at $t = 0.3$, beyond the training data time range from an NN of* 9 *layers,* 128 *neurons wide, using ReLU with batch normalization for smooth initial conditions. Of the* 100 *initial conditions,* 26 *cases failed in the solver.*

Predicting beyond the training set time interval with the kinetic-trained closure, we see the highly collisional range of Knudsen numbers continue to have good agreement with the test set moments for smooth initial conditons. However, in the transition and free streaming Knudsen regimes, we see wide variation in the quality of the solutions, with a max relative L^2 error of approximately 5.0. But there are also solutions that are very well behaved and with relative L^2 error of approximately 0.02. This can be seen in Figure 7. Preliminary investigation of the nonuniform performance at intermediate Knudsen number (i.e., $[0.1, 1.0]$) indicates evidence of overfitting within the NN. Adding dropout and increasing L^2-regularization with builtin PyTorch modules and optimizer parameters show some improvement in error and eliminated solver failures on some initial conditions. This will be explored in future work.

Investigating the computed kinetic-trained moment closures in the Knudsen numbers above 10^{-1}, we generally observe high-frequency oscillations develop beyond the training set time interval. This was not observed in the HME NN closures nor in the earlier RTE work, which both had good agreement over all collision regimes. These oscillations did not occur for all initial conditions. During our investigation, we studied the maximum distance between eigenvalues computed from the NN closure, which
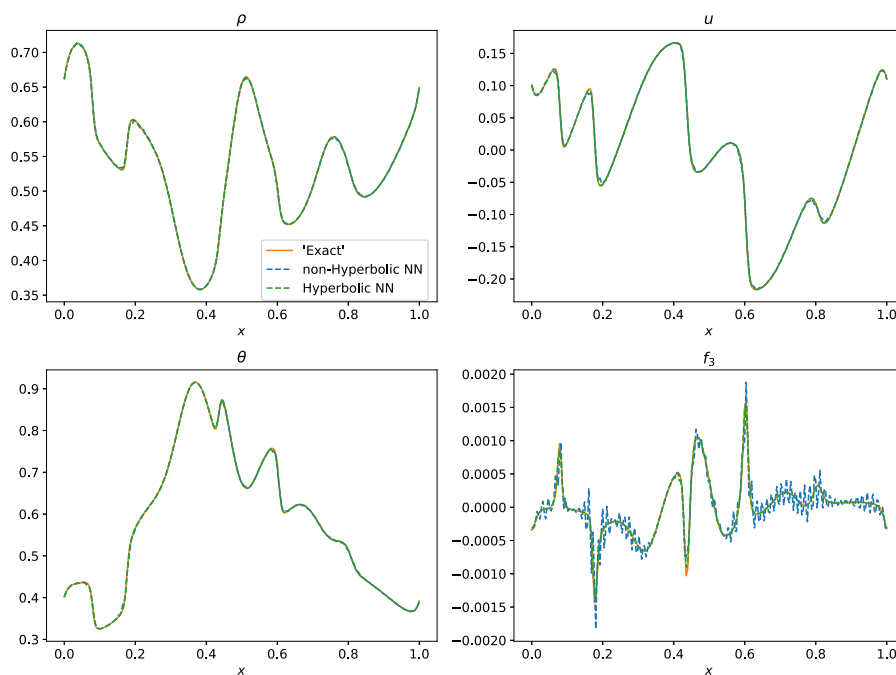
FIG. 8. *Moments calculated using a kinetic-trained NN without hyperbolic method and with the hyperbolic method. This example has a Knudsen number of* 0.001. *This prediction is at* $t = 0.3$.
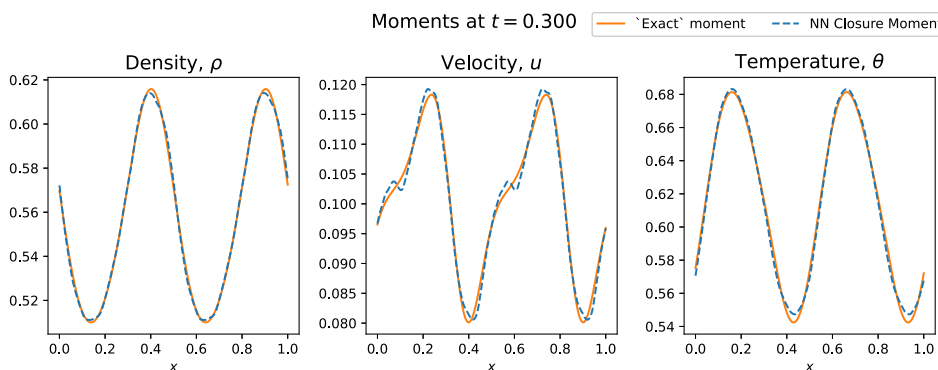

FIG. 9. *Galilean invariance using a kinetic-trained closure for the first three moments,* ρ, u, θ, *for Knudsen number of* 0.063. *The NN closure was trained with a macroscopic velocity of* $u = 0$ *and was used to predict for an initial condition with* $u = 0.1$. *This prediction is at* $t = 0.3$.

hints at the potential source of the oscillations. Comparing cases with similar Knudsen numbers, we observe that well-behaved predictions had well-separated eigenvalues through time, while predictions that saw oscillations develop saw minimum separation between eigenvalues shrink significantly before causing instability.

We compare both the time required for calculation and the relative errors in the primitive variables for a kinetic-trained NN closure with 8 total moments and the rational HME models with varying number of moments for a Knudsen number of 0.0466 in Table 6. The initial condition is sampled from smooth generated data. As a base for comparison, we use a DVM calculation as the reference for time and as

TABLE 6

Comparison of runtime and relative errors for HME models (no NN closure) with 6, 8, 10, and 12 moments and the kinetic-trained NN closure with 8 total moments. Runtimes and errors are computed against the DVM model. The HME model and the kinetic NN closure models were run in python, while the DVM model was run in MATLAB. Comparison made at $t = 0.3$ for a case with Knudsen number of 0.0466 for smooth data.

Model	DVM	HME-6	HME-8	HME-10	HME-12	NN-8
Time (s)	198	86	107	174	237	149
Relative time	1.00	0.43	0.54	0.88	1.20	0.75
Relative L^2 error: ρ	-	1.13%	0.51%	0.28%	0.15%	0.19%
Relative L^2 error: u	-	7.46%	3.18%	2.49%	1.94%	1.51%
Relative L^2 error: θ	-	1.43%	0.60%	0.42%	0.26%	0.35%

the comparison for errors. The kinetic-trained NN closure is competitive with HMEs in computational time while providing improved accuracy in the primitive variables. The accuracy of the solution in the primitive variables fell between the 10- and 12-moment HME models and had a time to solution comparable to 8- and 10-moment HME models. This shows modest improvements for Knudsen value approaching the transition regime. Our future work will focus on improving accuracy in the transition and free streaming limit Knudsen regimes. This only accounts for the computation time provided for a trained NN closure; the time to create training sets and train the associated NN are much longer.

As a point of interest, we formulated and tested classic Riemann problems and compared with kinetic NN closures trained on both smooth/wave and 50-50 smooth and mixed data. For small Knudsen numbers, below 0.01, the ML model accurately captures the appropriate dynamics, including how the initial discontinuity spreads. Interestingly, the added mixed training data provided superior performance in resolving the higher moments. For the transition regime, where the Knudsen number approaches 0.1, the model captures bulk dynamics but exhibits spurious oscillations. Generalizations that accurately capture the dynamics in the transition regime and beyond are subjects of our ongoing and future work.

6. Conclusion. We have developed and applied hyperbolicity-preserving NNs to the moment closure of the Grad expansion of the BGK model. We have demonstrated that, when a local NN moment closure is trained on data from the HME closure of the Grad expansion, the model robustly recovers HMEs both inside and outside of the training time window. Furthermore, when the model is trained on moments of the full kinetic model using a DVM, the local NN moment closure captures kinetic effects over a wider range of Knudsen numbers within the training window and beyond. We also demonstrate the utility of introducing nonsmooth data into the training processes for prediction beyond the training window.

For kinetic BGK training data, we have demonstrated NN closures for smooth initial conditions over a range of Knudsen numbers. In the fluid regime (i.e., Knudsen numbers between 10^{-3} and 10^{-1}), we have good agreement within and beyond the training time window. For transitional and free streaming regimes (i.e., Knudsen numbers between 10^{-1} and 10), we demonstrate good agreement within the training time interval. Outside of the training time window, while relative error is small, we see nonuniform performance in predictions. This may be due to our model not preserving the asymptotic limit of long time dynamics, which was preserved in the symmetrizer approach in earlier RTE work [17].

Future work would be to improve and expand the kinetic BGK closures' ability to generalize beyond training data set, especially in the in and above the transition regime. Possible approaches could include better NN regularization, greatly increasing the training set size with respect to Knudsen numbers, using transfer learning from HME NN closures to the kinetic closure, and implementing NN architectures that have regularization through low-rank tensor decomposition. Another possibility is the development of a symmetrizer for the system as in the earlier RTE work [17], but this would be nontrivial for the BGK model compared to RTE.

REFERENCES

[1] U. M. ASCHER, S. J. RUUTH, AND R. J. SPITERI, *Implicit-explicit Runge–Kutta methods for time-dependent partial differential equations*, Appl. Numer. Math., 25 (1997), pp. 151–167, https://doi.org/10.1016/S0168-9274(97)00056-1.

[2] P. L. BHATNAGAR, E. P. GROSS, AND M. KROOK, *A model for collision processes in gases.* I. *Small amplitude processes in charged and neutral one-component systems*, Phys. Rev., 94 (1954), pp. 511–525, https://doi.org/10.1103/PhysRev.94.511.

[3] J. E. BROADWELL, *Study of rarefied shear flow by the discrete velocity method*, J. Fluid Mech., 19 (1964), pp. 401–414, https://doi.org/10.1017/S0022112064000817.

[4] Z. CAI, Y. FAN, AND R. LI, *Globally hyperbolic regularization of Grad's moment system in one-dimensional space*, Commun. Math. Sci., 11 (2013), pp. 547–571, https://doi.org/10.4310/CMS.2013.v11.n2.a12.

[5] Z. CAI, Y. FAN, AND R. LI, *A framework on moment model reduction for kinetic equation*, SIAM J. Appl. Math., 75 (2015), pp. 2001–2023, https://doi.org/10.1137/14100110X.

[6] M. J. CASTRO, T. M. DE LUNA, AND C. PARÉS, *Well-balanced schemes and path-conservative numerical methods*, in Handb. Numer. Anal. 18, Elsevier, 2017, pp. 131–175.

[7] A. CHARALAMPOPOULOS, S. H. BRYNGELSON, T. COLONIUS, AND T. P. SAPSIS, *Hybrid quadrature moment method for accurate and stable representation of non-Gaussian processes applied to bubble dynamics*, Philos. Trans. Roy. Soc. A, 380 (2022), 20210209, https://doi.org/10.1098/rsta.2021.0209.

[8] G. DAL MASO, P. G. LEFLOCH, AND F. MURAT, *Definition and weak stability of nonconservative products*, J. Math. Pures Appl., 74 (1995), pp. 483–548.

[9] F. W. J. OLVER, A. B. OLDE DAALHUIS, D. W. LOZIER, B. I. SCHNEIDER, R. F. BOISVERT, C. W. CLARK, B. R. MILLER, B. V. SAUNDERS, H. S. COHL, AND M. A. MCCLAIN, eds., *NIST Digital Library of Mathematical Functions*, Release 1.1.12 of 2023-12-15, https://dlmf.nist.gov/.

[10] M. ELOUAFI AND A. D. A. HADJ, *A recursion formula for the characteristic polynomial of Hessenberg matrices*, Appl. Math. Comput., 208 (2009), pp. 177–179, https://doi.org/10.1016/j.amc.2008.11.027.

[11] I. GOODFELLOW, Y. BENGIO, AND A. COURVILLE, *Deep Learning*, MIT Press, Cambridge, MA, 2016, http://www.deeplearningbook.org.

[12] S. GOTTLIEB, C.-W. SHU, AND E. TADMOR, *Strong stability-preserving high-order time discretization methods*, SIAM Rev., 43 (2001), pp. 89–112, https://doi.org/10.1137/S003614450036757X.

[13] H. GRAD, *On the kinetic theory of rarefied gases*, Comm. Pure Appl. Math., 2 (1949), pp. 331–407, https://doi.org/10.1002/cpa.3160020403.

[14] J. HAN, C. MA, Z. MA, AND W. E, *Uniformly accurate machine learning-based hydrodynamic models for kinetic equations*, Proc. Natl. Acad. Sci., 116 (2019), pp. 21983–21991, https://doi.org/10.1073/pnas.1909854116.

[15] J. HUANG, Y. CHENG, A. J. CHRISTLIEB, AND L. F. ROBERTS, *Machine learning moment closure models for the radiative transfer equation* I: *Directly learning a gradient based closure*, J. Comput. Phys., 453 (2022), 110941, https://doi.org/10.1016/j.jcp.2022.110941.

[16] J. Huang, Y. Cheng, A. J. Christlieb, and L. F. Roberts, *Machine learning moment closure models for the radiative transfer equation* III: *Enforcing hyperbolicity and physical characteristic speeds*, J. Sci. Comput., 94 (2022), 7, http://dx.doi.org/10.1007/s10915-022-02056-7.

[17] J. Huang, Y. Cheng, A. J. Christlieb, L. F. Roberts, and W.-A. Yong, *Machine learning moment closure models for the radiative transfer equation* II: *Enforcing global hyperbolicity in gradient-based closures*, Multiscale Model. Simul., 21 (2023), pp. 489–512, https://doi.org/10.1137/21M1423956.

[18] S. Ioffe and C. Szegedy, *Batch normalization: Accelerating deep network training by reducing internal covariate shift*, in Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37, ICML'15, 2015, pp. 448–456, https://dl.acm.org/doi/10.5555/3045118.3045167.

[19] G.-S. Jiang and C.-W. Shu, *Efficient implementation of weighted ENO schemes*, J. Comput. Phys., 126 (1996), pp. 202–228, https://doi.org/10.1006/jcph.1996.0130.

[20] J. Johnson and S. Lyon, *SESAME: The Los Alamos National Laboratory Equations of State Database*, Technical report LA-UR-92-3407, Los Alamos National Laboratory, 1992.

[21] D. P. Kingma and J. Ba, *Adam: A Method for Stochastic Optimization*, preprint, arXiv:1412.6980, 2014.

[22] J. Koellermeier and M. Castro, *High-order non-conservative simulation of hyperbolic moment models in partially-conservative form*, East Asian J. Appl. Math., 11 (2021), pp. 435–467, https://doi.org/10.4208/eajam.090920.130121.

[23] J. Koellermeier, R. P. Schaerer, and M. Torrilhon, *A framework for hyperbolic approximation of kinetic equations using quadrature-based projection methods*, Kinet. Relat. Mod., 7 (2014), pp. 531–549, https://doi.org/10.3934/krm.2014.7.531.

[24] J. Koellermeier and M. Torrilhon, *Hyperbolic moment equations using quadrature-based projection methods*, in AIP Conference Proceedings, AIP Publishing, 2014, pp. 626–633, https://doi.org/10.1063/1.4902651.

[25] J. Koellermeier and M. Torrilhon, *Numerical study of partially conservative moment equations in kinetic theory*, Commun. Comput. Phys., 21 (2017), pp. 981–1011, https://doi.org/10.4208/cicp.OA-2016-0053.

[26] J. Koellermeier and M. Torrilhon, *Two-dimensional simulation of rarefied gas flows using quadrature-based moment equations*, Multiscale Model. Simul., 16 (2018), pp. 1059–1084, https://doi.org/10.1137/17M1147548.

[27] R. J. LeVeque, *Wave propagation algorithms for multidimensional hyperbolic systems*, J. Comput. Phys., 131 (1997), pp. 327–353, https://doi.org/10.1006/jcph.1996.5603.

[28] C. D. Levermore, *Moment closure hierarchies for kinetic theories*, J. Stat. Phys., 83 (1996), pp. 1021–1065, https://doi.org/10.1007/BF02179552.

[29] Z. Li, B. Dong, and Y. Wang, *Learning invariance preserving moment closure model for Boltzmann–BGK equation*, Commun. Math. Stat., 11 (2023), 59–101, https://doi.org/10.1007/s40304-022-00331-5.

[30] Z. Li, Y. Wang, H. Liu, Z. Wang, and B. Dong, *Solving Boltzmann Equation With Neural Sparse Representation*, preprint, arXiv:2302.09233, 2023.

[31] I. Loshchilov and F. Hutter, *Decoupled Weight Decay Regularization*, preprint, arXiv:1711.05101, 2017.

[32] J. D. McHardy, *An Introduction to the Theory and Use of SESAME Equations of State*, Techical report LA-14503, Office of Scientific and Technical Information (OSTI), 2018, https://doi.org/10.2172/1487368.

[33] S. T. Miller, N. V. Roberts, S. D. Bond, and E. C. Cyr, *Neural-network based collision operators for the Boltzmann equation*, J. Comput. Phys., 470 (2022), 111541, https://doi.org/10.1016/j.jcp.2022.111541.

[34] S. Naris, D. Valougeorgis, F. Sharipov, and D. Kalempa, *Discrete velocity modelling of gaseous mixture flows in MEMS*, Superlattice. Microst., 35 (2004), pp. 629–643, https://doi.org/10.1016/j.spmi.2004.02.025.

[35] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, *Pytorch: An imperative style, high-performance deep learning library*, in Advances in Neural Information Processing Systems 32, Curran Associates, Red Hook, NY, 2019, pp. 8024–8035, http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[36] W. A. PORTEOUS, M. P. LAIU, AND C. D. HAUCK, *Data-Driven, Structure-Preserving Approximations to Entropy-Based Moment Closures for Kinetic Equations*, preprint, arXiv:2106.08973, 2021.

[37] S. SCHOTTHÖFER, T. XIAO, M. FRANK, AND C. HAUCK, *Structure preserving neural networks: A case study in the entropy closure of the Boltzmann equation*, in Proceedings of the 39th International Conference on Machine Learning, Proceedings of Machine Learning Research 162, PMLR, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, eds., 2022, pp. 19406–19433, https://proceedings.mlr.press/v162/schotthofer22a.html.

[38] S. SCHOTTHÖFER, T. XIAO, M. FRANK, AND C. D. HAUCK, *Neural Network-Based, Structure-Preserving Entropy Closures for the Boltzmann Moment System*, preprint, arXiv:2201.10364, 2022.

[39] C.-W. SHU AND S. OSHER, *Efficient implementation of essentially non-oscillatory shock-capturing schemes*, J. Comput. Phys., 77 (1988), pp. 439–471, https://doi.org/10.1016/0021-9991(88)90177-5.

[40] L. N. SMITH, *Cyclical Learning Rates for Training Neural Networks*, preprint, arXiv:1506.01186, 2015.

[41] L. N. SMITH AND N. TOPIN, *Super-Convergence: Very Fast Training of Neural Networks Using Large Learning Rates*, preprint, arXiv:1708.07120, 2017.

[42] P. STANGEBY AND G. MCCRACKEN, *Plasma boundary phenomena in tokamaks*, Nucl. Fusion, 30 (1990), pp. 1225–1379, https://doi.org/10.1088/0029-5515/30/7/005.

[43] H. STRUCHTRUP, *Macroscopic Transport Equations for Rarefied Gas Flows: Approximation Methods in Kinetic Theory*, Springer, Berlin, 2005, https://doi.org/10.1007/3-540-32386-4.

[44] W. TAITANO, B. KEENAN, L. CHACÓN, S. ANDERSON, H. HAMMER, AND A. SIMAKOV, *An Eulerian Vlasov–Fokker–Planck algorithm for spherical implosion simulations of inertial confinement fusion capsules*, Comput. Phys. Commun., 263 (2021), 107861, https://doi.org/10.1016/j.cpc.2021.107861.

[45] W. T. TAITANO, A. N. SIMAKOV, L. CHACÓN, AND B. KEENAN, *Yield degradation in inertial-confinement-fusion implosions due to shock-driven kinetic fuel-species stratification and viscous heating*, Phys. Plasmas, 25 (2018), 056310, https://doi.org/10.1063/1.5024402.

[46] K.-C. TSENG, J.-S. WU, AND I. D. BOYD, *Simulations of re-entry vehicles by using DSMC with chemical-reaction module*, in 14th AIAA/AHI Space Planes and Hypersonic Systems and Technologies Conference, American Institute of Aeronautics and Astronautics, 2006, https://doi.org/10.2514/6.2006-8084.