Journal of Computational Physics 421 (2020) 109744

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Tensor methods for the Boltzmann-BGK equation

Arnout M.P. Boelens [a], Daniele Venturi [b], Daniel M. Tartakovsky [a,*]

[a] Department of Energy Resources Engineering, Stanford University, Stanford, CA 94305, United States of America
[b] Department of Applied Mathematics, UC Santa Cruz, Santa Cruz, CA 95064, United States of America

ARTICLE INFO

Article history:
Received 12 November 2019
Received in revised form 25 July 2020
Accepted 27 July 2020
Available online 5 August 2020

Keywords:
High-dimensional PDE
Tensor method
Non-equilibrium

ABSTRACT

We present a tensor-decomposition method to solve the Boltzmann transport equation (BTE) in the Bhatnagar-Gross-Krook approximation. The method represents the six-dimensional BTE as a set of six one-dimensional problems, which are solved with the alternating least-squares algorithm and the discrete Fourier transform at N collocation points. We use this method to predict the equilibrium distribution (steady-state simulation) and a non-equilibrium distribution returning to the equilibrium (transient simulation). Our numerical experiments demonstrate $N \log N$ scaling. Unlike many BTE-specific numerical techniques, the numerical tensor-decomposition method we propose is a general technique that can be applied to other high-dimensional systems.

1. Introduction

Whenever the mean free path of molecules becomes larger than the characteristic length scale of a system, the continuity assumption breaks down and so does the validity of the Navier-Stokes equations. This phenomenon occurs in a number of settings, including splashing droplets [1], moving contact lines [2], super- and hyper-sonic flows [3], and flow of electrons in metals [4] and silicon [5]. The physics in this flow regime is often described by the six-dimensional (plus time) Boltzmann transport equation (BTE) [6].

Like many other high-dimensional partial differential equations (PDEs), the BTE suffers from the curse of dimensionality: the computational cost of conventional numerical schemes, such as those based on tensor product representations, grows exponentially with an increasing number of degrees of freedom. One way to mitigate such computational complexity is to use particle-based methods [7], e.g., direct simulation Monte Carlo (DSMC) [8] or the Nambu-Babovsky method [9]. These methods preserve the main physical properties of the system, even far from equilibrium, and are computationally efficient away from near-fluid regimes. In particular, they have low memory requirements and their cost scales linearly with the number of particles. However, their accuracy, efficiency and convergence rate tend to be poor for non-stationary flows, or flows close to continuum regimes [10–12]. This is due to the non-negligible statistical fluctuations associated with finite particle numbers, which are difficult and expensive to filter out in such flow regimes [10,13]. While several general purpose algorithms have been proposed, the most efficient techniques are problem specific [7,14–16]. These methods exploit the BTE's mathematical properties to arrive at an efficient algorithm, but are not generally applicable to other high-dimensional PDEs.

We present a new algorithm based on tensor decompositions to solve the BTE in the Bhatnagar-Gross-Krook (BGK) approximation [17]. The algorithm relies on canonical tensor expansions [18], combined with either alternating direction

* Corresponding author.
E-mail address: tartakovsky@stanford.edu (D.M. Tartakovsky).

https://doi.org/10.1016/j.jcp.2020.109744
0021-9991/

least squares methods [19–22] or alternating direction Galerkin methods [23,24] or any other version of the method of mean weighted residuals (MWR) [25]. Unlike the BTE-specific numerical techniques, tensor-decomposition methods are general-purpose, i.e., they can be applied to other high-dimensional nonlinear PDEs [26,27], including but not limited to the Hamilton-Jacobi-Bellman equation [28], the Fokker-Planck equation [29], and the Vlasov equation [30–32]. This opens the possibility to use tensor methods in many research fields including chemical reaction networks in turbulent flows [33], neuroscience [34], and approximation of functional differential equations [24]. Recently, we developed the tensor-decomposition method [18] to solve a linearized BGK equation. In this paper, we extend it to the full BTE in the BGK approximation, i.e., to obviate the need for the assumption of small fluctuations and to allow for variable density, velocity, temperature, and collision frequency fields.

This paper is organized as follows. Section 2 contains a brief overview of the Boltzmann-BGK equation. In section 3, we propose an efficient algorithm to compute its solution based canonical tensor expansions. Numerical experiments reported in section 4 are used to demonstrate the algorithm's ability to accurately predict the equilibrium distribution of the system (steady-state), and the exponential relaxation to equilibrium (transient simulation) of non-equilibrium initial states. The algorithm exhibits $\mathcal{O}(N\log(N))$ scaling, where N is the number of degrees of freedom in each of the phase variables. Main conclusions drawn from the numerical experimentation and future directions of research are summarized in section 5.

2. Boltzmann equation

In the classical kinetic theory of rarefied gas dynamics, flow of gases is described in terms of a probability density function (PDF) $f(\mathbf{x},\boldsymbol{\xi},t)$, which estimates the number of gas particles with velocity $\boldsymbol{\xi}\in\mathbb{R}^3$ at position $\mathbf{x}\in\mathbb{R}^3$ at time $t\in\mathbb{R}^+$, such that $\mathrm{d}N = f\mathrm{d}\mathbf{x}\mathrm{d}\boldsymbol{\xi}$ with N denoting the number of particles (in moles). In the absence of external forces, the PDF f satisfies the Boltzmann equation [35],

$$\frac{\partial f}{\partial t} + \boldsymbol{\xi}\cdot\nabla_{\mathbf{x}} f = Q(f,f), \tag{1}$$

where $Q(f,f)$ is the collision integral describing the effects of internal forces due to particle interactions. From the mathematical viewpoint, the collision integral is a functional of the PDF f, whose form depends on the microscopic dynamics. For example, in classical rarefied gas flows [36,37],

$$Q(f,f)(\mathbf{x},\boldsymbol{\xi},t) = \int_{\mathbb{R}^3}\int_{\mathbb{S}^2} B(\boldsymbol{\xi},\boldsymbol{\xi}_1,\boldsymbol{\omega})\left|f(\mathbf{x},\boldsymbol{\xi}',t)f(\mathbf{x},\boldsymbol{\xi}_1',t) - f(\mathbf{x},\boldsymbol{\xi},t)f(\mathbf{x},\boldsymbol{\xi}_1,t)\right|\mathrm{d}\boldsymbol{\omega}\mathrm{d}\boldsymbol{\xi}_1. \tag{2}$$

Here, $\boldsymbol{\xi}$ and $\boldsymbol{\xi}_1$ are the velocities of two particles before the collision; $\boldsymbol{\xi}' = (\boldsymbol{\xi}+\boldsymbol{\xi}_1 + \|\boldsymbol{\xi}-\boldsymbol{\xi}_1\|_2\boldsymbol{\omega})/2$ and $\boldsymbol{\xi}_1' = (\boldsymbol{\xi}+\boldsymbol{\xi}_1 - \|\boldsymbol{\xi}-\boldsymbol{\xi}_1\|_2\boldsymbol{\omega})/2$ are these velocities after the collision; and $\boldsymbol{\omega}$ is the unit vector to the three-dimensional unit sphere $\mathbb{S}^2$. The collision kernel $B(\boldsymbol{\xi},\boldsymbol{\xi}_1,\boldsymbol{\omega})$ is a non-negative function of the Euclidean 2-norm $\|\boldsymbol{\xi}-\boldsymbol{\xi}_1\|_2$ and the scattering angle θ between the relative velocities before and after the collision,

$$\cos\theta = \frac{(\boldsymbol{\xi}-\boldsymbol{\xi}_1)\cdot\boldsymbol{\omega}}{\|\boldsymbol{\xi}-\boldsymbol{\xi}_1\|_2}. \tag{3}$$

Specifically,

$$B(\boldsymbol{\xi},\boldsymbol{\xi}_1,\boldsymbol{\omega}) = \|\boldsymbol{\xi}-\boldsymbol{\xi}_1\|_2\,\sigma(\|\boldsymbol{\xi}-\boldsymbol{\xi}_1\|_2\cos\theta), \tag{4}$$

where σ is the cross-section scattering function [37]. The collision operator (2) satisfies a system of three conservation laws [7],

$$\int_{\mathbb{R}^3} Q(f,f)(\mathbf{x},\boldsymbol{\xi},t)\psi(\boldsymbol{\xi})\mathrm{d}\boldsymbol{\xi} = 0, \qquad \psi(\boldsymbol{\xi}) = 1 \text{ or } \boldsymbol{\xi} \text{ or } \|\boldsymbol{\xi}\|_2^2, \tag{5}$$

for mass, momentum, and energy, respectively. It also satisfies the Boltzmann H-theorem,

$$\int_{\mathbb{R}^3} Q(f,f)(\mathbf{x},\boldsymbol{\xi},t)\log\left(f(\mathbf{x},\boldsymbol{\xi},t)\right)\mathrm{d}\boldsymbol{\xi} \le 0, \tag{6}$$

that implies that any equilibrium PDF, i.e., any PDF f for which $Q(f,f)=0$, is locally Maxwellian:

$$f_{\mathrm{eq}}(\mathbf{x},\boldsymbol{\xi},t) = \frac{n(\mathbf{x},t)}{(2\pi k_{\mathrm{B}}T(\mathbf{x},t)/m)^{3/2}}\exp\left(-\frac{m\|\mathbf{U}(\mathbf{x},t)-\boldsymbol{\xi}\|_2^2}{2k_{\mathrm{B}}T(\mathbf{x},t)}\right). \tag{7}$$

Here k_{B} is the Boltzmann constant; m is the particle mass; and the number density n, mean velocity $\mathbf{U}$, and temperature T of a gas are defined as

$$n(\mathbf{x},t) = \int_{\mathbb{R}^3} f(\mathbf{x},\boldsymbol{\xi},t)\mathrm{d}\boldsymbol{\xi}, \tag{8}$$

$$\mathbf{U}(\mathbf{x},t) = \frac{1}{n(\mathbf{x},t)} \int_{\mathbb{R}^3} \boldsymbol{\xi} f(\mathbf{x},\boldsymbol{\xi},t)\mathrm{d}\boldsymbol{\xi}, \tag{9}$$

$$T(\mathbf{x},t) = \frac{m}{3k_{\mathrm{B}} n(\mathbf{x},t)} \int_{\mathbb{R}^3} \|\mathbf{U}(\mathbf{x},t) - \boldsymbol{\xi}\|_2^2 \, f(\mathbf{x},\boldsymbol{\xi},t)\mathrm{d}\boldsymbol{\xi}. \tag{10}$$

The Boltzmann equation (1) is a nonlinear integro-differential equation in six dimensions plus time. By taking suitable averages over small volumes in position space, one can show that the Boltzmann equation is consistent with both the compressible Euler equations [38,39] and the Navier-Stokes equations [40,41].

2.1. BGK approximation of the collision operator

The simplest collision operator satisfying the conservation laws (5) and the Boltzmann H-theorem (6) is the linear relaxation operator,

$$Q(f,f) = \nu(\mathbf{x},t)\left[f_{\mathrm{eq}}(\mathbf{x},\boldsymbol{\xi},t) - f(\mathbf{x},\boldsymbol{\xi},t)\right], \qquad \nu(\mathbf{x},t) > 0. \tag{11}$$

It is known as the Bhatnagar-Gross-Krook (BGK) model [17]. The collision frequency $\nu(\mathbf{x},t)$ is usually set to be proportional to the gas number-density and temperature [42],

$$\nu(\mathbf{x},t) = nKT^{1-\mu}. \tag{12}$$

The exponent μ of the viscosity law depends on the molecular interaction potential and on the type of the gas; and $K = k_{\mathrm{B}}T_{\mathrm{ref}}/(m\mu_{\mathrm{ref}}) > 0$, with μ_{ref} denoting the gas viscosity at the reference temperature T_{ref}.

The combination of (1) and (11) yields the Boltzmann-BGK equation,

$$\frac{\partial f(\mathbf{x},\boldsymbol{\xi},t)}{\partial t} + \boldsymbol{\xi}\cdot\nabla_{\mathbf{x}} f(\mathbf{x},\boldsymbol{\xi},t) = \nu(\mathbf{x},t)\left[f_{\mathrm{eq}}(\mathbf{x},\boldsymbol{\xi},t) - f(\mathbf{x},\boldsymbol{\xi},t)\right]. \tag{13}$$

By virtue of (8)–(10), both $f_{\mathrm{eq}}(\mathbf{x},\boldsymbol{\xi},t)$ in (7) and $\nu(\mathbf{x},t)$ in (12) are nonlinear functionals of the PDF $f(\mathbf{x},\boldsymbol{\xi},t)$. Therefore, (13) is a nonlinear integro-differential PDE in six dimension plus time. It converges to the Euler equations of incompressible fluid dynamics with the scaling $\mathbf{x}' = \epsilon\mathbf{x}$ and $t' = \epsilon t$, in the limit $\epsilon \to 0$. However, it does not converge to the Navier-Stokes equations in this limit. Specifically, it predicts an unphysical Prandtl number [43], which is larger than the one obtained with the full collision operator (2). The Navier-Stokes equations can be recovered as ϵ-limits of more sophisticated BGK models, e.g., the Gaussian-BGK model [44].

In [18], we introduced additional simplifications, i.e., assumed ν to be constant and the equilibrium density, temperature and velocity to be spatially homogeneous. If one additionally assumes $\mathbf{U} \equiv \mathbf{0}$, the resulting model yields an equilibrium PDF f_{eq} in (7). The last assumption effectively decouples the BGK collision operator from the PDF $f(\mathbf{x},\boldsymbol{\xi},t)$. This, in turn, turns the BGK equation into a linear six-dimensional PDE. In this paper, we develop a numerical method to solve the fully nonlinear Boltzmann-BGK equation.

2.2. Scaling

Let us define the Boltzmann-BGK equation (13) on a six-dimensional hypercube, such that $f : \Omega_{\mathbf{x}} \times \Omega_{\boldsymbol{\xi}} \times \mathbb{R}^+ \to \mathbb{R}^+$ with $\Omega_{\mathbf{x}} = [-b_x, b_x]^3$ and $\Omega_{\boldsymbol{\xi}} = [-b_\xi, b_\xi]^3$ representing the spatial domain and the velocity domain, respectively. Furthermore, we impose periodic boundary conditions on all the surfaces of this hypercube. We transform the hypercube $\Omega_{\mathbf{x}} \times \Omega_{\boldsymbol{\xi}}$ into the "standard" hypercube $\Omega_\pi = [-\pi,\pi]^6$, perform simulations in Ω_π, and then map the numerical results back onto $\Omega_{\mathbf{x}} \times \Omega_{\boldsymbol{\xi}}$. This is accomplished by introducing dimensionless independent and dependent variables

$$\tilde{\boldsymbol{\xi}} = \frac{\boldsymbol{\xi}\pi}{b_\xi}, \quad \tilde{\mathbf{x}} = \frac{\mathbf{x}\pi}{b_x}, \quad \tilde{t} = \frac{t b_\xi}{b_x}, \quad \tilde{n} = n b_x^3, \quad \tilde{\mathbf{U}} = \frac{\mathbf{U}\pi}{b_\xi}, \quad \tilde{T} = \frac{T}{T_c}, \quad \tilde{\nu} = \frac{\nu\lambda}{b_\xi},$$

where λ is the mean free path of a gas molecule, and T_c is a characteristic temperature. Furthermore, we define the dimensionless Knudsen (Kn) and Boltzmann (Bo) numbers as

$$\mathrm{Kn} = \frac{\lambda}{b_x} \quad \text{and} \quad \mathrm{Bo} = \frac{m b_\xi^2}{\pi^2 k_{\mathrm{B}} T_c}. \tag{14}$$

Then, the rescaled PDF $\tilde{f}(\tilde{\mathbf{x}},\tilde{\boldsymbol{\xi}},\tilde{t}) = f(\mathbf{x},\boldsymbol{\xi},t) b_x^3 b_\xi^3/\pi^3$ satisfies a dimensionless form of the Boltzmann-BGK equation (13),

$$\frac{\partial \tilde{f}}{\partial \tilde{t}} = L(\tilde{\boldsymbol{\xi}})\tilde{f} + C(\tilde{\mathbf{x}}, \tilde{\boldsymbol{\xi}}, \tilde{t}), \qquad L(\tilde{\boldsymbol{\xi}}) \equiv -\tilde{\boldsymbol{\xi}} \cdot \nabla_{\tilde{\mathbf{x}}}, \quad C(\tilde{\mathbf{x}}, \tilde{\boldsymbol{\xi}}, \tilde{t}) \equiv \frac{\tilde{\nu}}{\mathrm{Kn}}(\tilde{f}_{\mathrm{eq}} - \tilde{f}), \tag{15}$$

where

$$\tilde{f}_{\mathrm{eq}}(\tilde{\mathbf{x}}, \tilde{\boldsymbol{\xi}}, \tilde{t}) = \frac{\tilde{n}}{(2\pi \tilde{T}/\mathrm{Bo})^{3/2}} \exp\left(-\mathrm{Bo}\frac{\|\tilde{\boldsymbol{\xi}} - \tilde{\mathbf{U}}\|^2}{2\tilde{T}}\right), \tag{16}$$

with

$$\tilde{n}(\tilde{\mathbf{x}}, \tilde{t}) = \int\limits_{[-\pi,\pi]^3} \tilde{f}(\tilde{\mathbf{x}}, \tilde{\boldsymbol{\xi}}, \tilde{t}) \mathrm{d}\tilde{\boldsymbol{\xi}}, \tag{17}$$

$$\tilde{\mathbf{U}}(\tilde{\mathbf{x}}, \tilde{t}) = \frac{1}{\tilde{n}(\tilde{\mathbf{x}}, \tilde{t})} \int\limits_{[-\pi,\pi]^3} \tilde{\boldsymbol{\xi}} \tilde{f}(\tilde{\mathbf{x}}, \tilde{\boldsymbol{\xi}}, \tilde{t}) \mathrm{d}\tilde{\boldsymbol{\xi}}, \tag{18}$$

$$\tilde{T}(\tilde{\mathbf{x}}, \tilde{t}) = \frac{\mathrm{Bo}}{3\tilde{n}(\tilde{\mathbf{x}}, \tilde{t})} \int\limits_{[-\pi,\pi]^3} \|\tilde{\boldsymbol{\xi}} - \tilde{\mathbf{U}}(\tilde{\mathbf{x}}, \tilde{t})\|_2^2 \tilde{f}(\tilde{\mathbf{x}}, \tilde{\boldsymbol{\xi}}, \tilde{t}) \mathrm{d}\tilde{\boldsymbol{\xi}}. \tag{19}$$

For notational convenience, we drop the tilde below, while continuing to use the dimensionless quantities.

3. A tensor method to solve the Boltzmann-BGK equation

Temporal discretization of the Boltzmann-BGK equation (15) is complicated by the presence of the collision term $C(\mathbf{x}, \boldsymbol{\xi}, t)$, whose evaluation is computationally expensive. The combination of the Crank-Nicolson time integration scheme with alternating-direction least squares, implemented in [18], would require multiple evaluations of $C(\mathbf{x}, \boldsymbol{\xi}, t)$ per time step, undermining the efficiency of the resulting algorithm. To ameliorate this problem, we replace the Crank-Nicolson method with the Crank-Nicolson Leap Frog (CNLF) scheme [45–48],

$$\frac{f(\cdot, t_{n+1}) - f(\cdot, t_{n-1})}{2\Delta t} = \frac{L(\boldsymbol{\xi}) f(\cdot, t_{n+1}) + L(\boldsymbol{\xi}) f(\cdot, t_{n-1})}{2} + C(\cdot, t_n) + \tau_{n+1}, \tag{20}$$

where τ_{n+1} is the local truncation error at time t_{n+1}.

The CNLF scheme has several advantages over other time-integration methods when applied to tensor discretization of the Boltzmann-BGK equation. First, being an implicit scheme, CNLF allows one to march forward in time by solving systems of linear equations on tensor manifolds with constant rank.[1] Since such manifolds are *smooth* [50,51], one can compute these solutions using, e.g., Riemannian quasi-Newton optimization [50,52,53] or alternating least squares [19,54,55]. Second, CNLF facilitates the explicit calculation of the collision term $C(\boldsymbol{x}, \boldsymbol{\xi}, t)$, and only *once* per time step. To demonstrate this, we rewrite (20) as

$$\underbrace{[I - \Delta t\, L(\boldsymbol{\xi})]}_{A(\boldsymbol{\xi})} f(\cdot, t_{n+1}) = \underbrace{[I + \Delta t\, L(\boldsymbol{\xi})] f(\cdot, t_{n-1}) + 2\Delta t\, C(\cdot, t_n)}_{h(\mathbf{x}, \boldsymbol{\xi}, t_n, t_{n-1})} + 2\Delta t\, \tau_{n+1}, \tag{21a}$$

where I is the identity operator; or

$$A(\boldsymbol{\xi}) f(\cdot, t_{n+1}) = h(\mathbf{x}, \boldsymbol{\xi}, t_n, t_{n-1}) + 2\Delta t\, \tau_{n+1}. \tag{21b}$$

Given $f(\mathbf{x}, \boldsymbol{\xi}, t_n)$ and $f(\mathbf{x}, \boldsymbol{\xi}, t_{n-1})$, this equation allows us to compute $f(\mathbf{x}, \boldsymbol{\xi}, t_{n+1})$ by solving a linear system. In the numerical tensor setting described below, this involves only iterations in $f(\mathbf{x}, \boldsymbol{\xi}, t_{n+1})$, which makes it possible to pre-calculate the computationally expensive collision term $C(\mathbf{x}, \boldsymbol{\xi}, t)$ once per time step.

The choice of the time step Δt in (21) requires some care, since CNLF is conditionally stable [56]. We transform this scheme into an unconditionally stable one by using, e.g., the Robert-Asselin-Williams (RAW) filter [57–59].

3.1. Canonical tensor decomposition and alternating least squares (ALS)

We expand the PDF $f(\mathbf{x}, \boldsymbol{\xi}, t_n)$ in a truncated canonical tensor series [18,22],

$$f(\mathbf{x}, \boldsymbol{\xi}, t_n) \simeq \sum_{l=1}^{r_l} f_1^l(x_1, t_n) f_2^l(x_2, t_n) f_3^l(x_3, t_n) f_4^l(\xi_1, t_n) f_5^l(\xi_2, t_n) f_6^l(\xi_3, t_n). \tag{22}$$

[1] Explicit time-integration algorithms require rank reduction [49] as application of linear operators to tensors, tensor addition, and other tensor operations results in increased tensor ranks.

The separation rank r_l is chosen adaptively to keep the norm of the residual below a pre-selected threshold at each time t_n. To simplify the notation, we introduce the combined position-velocity vector $\boldsymbol{\zeta} = (\mathbf{x}, \boldsymbol{\xi})$ and rewrite (22) as

$$f(\boldsymbol{\zeta}, t_n) \simeq \sum_{l=1}^{r_l} \prod_{k=1}^{6} f_k^l(\zeta_k, t_n), \qquad \boldsymbol{\zeta} \in \Omega_\pi = [-\pi, \pi]^6. \tag{23}$$

Next, we expand each function $f_k^l(\zeta_k, t_n)$ in a finite-dimensional Fourier basis $\phi_s(\zeta_k)$ [60] on $[-\pi, \pi]$,

$$f_k^l(\zeta_k, t_n) = \sum_{s=1}^{Q} \beta_{k,s}^l(t_n)\phi_s(\zeta_k), \tag{24}$$

where Q is the number of modes of the Fourier-series expansion, $\beta_{k,s}^l(t_n)$ are the (unknown) Fourier coefficients, and $\phi_s(\zeta_k)$ are the orthogonal trigonometric functions. Substituting (23) into (21) yields the residual

$$R(\boldsymbol{\zeta}, t_{n+1}, t_n, t_{n-1}) = \sum_{l=1}^{r_l} A(\boldsymbol{\xi})\, f_1^l(\zeta_1, t_{n+1}) \dots f_6^l(\zeta_6, t_{n+1}) - h(\boldsymbol{\zeta}, t_n, t_{n-1}). \tag{25}$$

The coefficients $\beta_{k,s}^l(t_n)$ are obtained by minimizing the L^2 norm of this residual with respect to

$$\boldsymbol{\beta}(t_{n+1}) = [\boldsymbol{\beta}_1(t_{n+1}), \dots, \boldsymbol{\beta}_6(t_{n+1})]. \tag{26}$$

The kth vector $\boldsymbol{\beta}_k(t_{n+1}) = [(\beta_{k,1}^1, \dots \beta_{k,Q}^1), \dots (\beta_{k,1}^r, \dots \beta_{k,Q}^r)]^\top$, for $k = 1, \dots, 6$, collects the degrees of freedom representing the PDF f in (23) corresponding to the phase variable ζ_k at time t_{n+1}, in accordance with (24).

We employ the alternating least squares (ALS) algorithm [55,61] to solve the minimization problem

$$\min_{\boldsymbol{\beta}_k} \| R(\boldsymbol{\zeta}, t_{n+1}, t_n, t_{n-1}) \|_{L^2(\Omega_\pi)}^2 \tag{27}$$

sequentially and iteratively for $k = 1, \dots, 6$. The ALS algorithm is locally equivalent to the linear block Gauss–Seidel iteration method applied to the Hessian of the residual R. As a consequence, it converges linearly with the iteration number [54], provided that the Hessian is positive definite (except on a trivial null space associated with the scaling non-uniqueness of the canonical tensor decomposition). Each minimization in (27) yields an Euler-Lagrange equation,

$$\mathbf{M}_k(t_{n+1})\boldsymbol{\beta}_k(t_{n+1}) = \boldsymbol{\gamma}_k(t_{n+1}), \qquad k = 1, \dots, 6. \tag{28}$$

Its expanded form reads

$$\underbrace{\begin{bmatrix} \begin{bmatrix} (M_k)_{1,1}^{1,1} \cdots (M_k)_{Q,1}^{1,1} \\ \vdots \quad \ddots \quad \vdots \\ (M_k)_{1,Q}^{1,1} \cdots (M_k)_{Q,Q}^{1,1} \end{bmatrix} & \begin{bmatrix} (M_k)_{1,1}^{2,1} \cdots (M_k)_{Q,1}^{2,1} \\ \vdots \quad \ddots \quad \vdots \\ (M_k)_{1,Q}^{2,1} \cdots (M_k)_{Q,Q}^{2,1} \end{bmatrix} \cdots & \begin{bmatrix} (M_k)_{1,1}^{r,1} \cdots (M_k)_{Q,1}^{r,1} \\ \vdots \quad \ddots \quad \vdots \\ (M_k)_{1,Q}^{r,1} \cdots (M_k)_{Q,Q}^{r,1} \end{bmatrix} \\ \begin{bmatrix} (M_k)_{1,1}^{1,2} \cdots (M_k)_{Q,1}^{1,2} \\ \vdots \quad \ddots \quad \vdots \\ (M_k)_{1,Q}^{1,2} \cdots (M_k)_{Q,Q}^{1,2} \end{bmatrix} & \begin{bmatrix} (M_k)_{1,1}^{2,2} \cdots (M_k)_{Q,1}^{2,2} \\ \vdots \quad \ddots \quad \vdots \\ (M_k)_{1,Q}^{2,2} \cdots (M_k)_{Q,Q}^{2,2} \end{bmatrix} \cdots & \begin{bmatrix} (M_k)_{1,1}^{r,2} \cdots (M_k)_{Q,1}^{r,2} \\ \vdots \quad \ddots \quad \vdots \\ (M_k)_{1,Q}^{r,2} \cdots (M_k)_{Q,Q}^{r,2} \end{bmatrix} \\ \vdots & \ddots & \vdots \\ \begin{bmatrix} (M_k)_{1,1}^{1,r} \cdots (M_k)_{Q,1}^{1,r} \\ \vdots \quad \ddots \quad \vdots \\ (M_k)_{1,Q}^{1,r} \cdots (M_k)_{Q,Q}^{1,r} \end{bmatrix} & \begin{bmatrix} (M_k)_{1,1}^{2,r} \cdots (M_k)_{Q,1}^{2,r} \\ \vdots \quad \ddots \quad \vdots \\ (M_k)_{1,Q}^{2,r} \cdots (M_k)_{Q,Q}^{2,r} \end{bmatrix} \cdots & \begin{bmatrix} (M_k)_{1,1}^{r,r} \cdots (M_k)_{Q,1}^{r,r} \\ \vdots \quad \ddots \quad \vdots \\ (M_k)_{1,Q}^{r,r} \cdots (M_k)_{Q,Q}^{r,r} \end{bmatrix} \end{bmatrix}}_{\mathbf{M}_k} \underbrace{\begin{bmatrix} \begin{bmatrix} \beta_{k,1}^1 \\ \vdots \\ \beta_{k,Q}^1 \end{bmatrix} \\ \begin{bmatrix} \beta_{k,1}^2 \\ \vdots \\ \beta_{k,Q}^2 \end{bmatrix} \\ \vdots \\ \begin{bmatrix} \beta_{k,1}^r \\ \vdots \\ \beta_{k,Q}^r \end{bmatrix} \end{bmatrix}}_{\boldsymbol{\beta}_k} = \underbrace{\begin{bmatrix} \begin{bmatrix} \gamma_{k,1}^1 \\ \vdots \\ \gamma_{k,Q}^1 \end{bmatrix} \\ \begin{bmatrix} \gamma_{k,1}^2 \\ \vdots \\ \gamma_{k,Q}^2 \end{bmatrix} \\ \vdots \\ \begin{bmatrix} \gamma_{k,1}^r \\ \vdots \\ \gamma_{k,Q}^r \end{bmatrix} \end{bmatrix}}_{\boldsymbol{\gamma}_k}$$

where

$$(M_k)_{s,q}^{l,z}(t_{n+1}) = \int_{\Omega_\pi} \left[A(\boldsymbol{\xi})\phi_s(\zeta_k) \prod_{\substack{j=1 \\ j\neq k}}^{6} f_j^l(\zeta_j, t_{n+1}) \right] \left[A(\boldsymbol{\xi})\phi_q(\zeta_k) \prod_{\substack{j=1 \\ j\neq k}}^{6} f_j^z(\zeta_j, t_{n+1}) \right] \mathrm{d}\boldsymbol{\zeta}. \tag{29}$$

Since the linear operator, first defined in (21), is fully separable (with rank 4), the 6D integral in (29) turns into the sum of the product of 1D integrals. The right hand side of (29) is

$$\begin{aligned}\gamma^l_{k,q} &= \int_{\Omega_\pi} h(\boldsymbol{\zeta}, t_n, t_{n-1}) A(\boldsymbol{\xi}) \phi_q(\zeta_k) \prod_{\substack{j=1 \\ j\neq k}}^{6} f^l_j(\zeta_j, t_{n+1}) \mathrm{d}\boldsymbol{\zeta} \\ &= \sum_{m=1}^{r_m} \int_{\Omega_\pi} \left\{ [I + \Delta t L(\boldsymbol{\xi})] \prod_{j=1}^{6} f^m_j(\zeta_j, t_{n-1}) \right\} \left\{ A(\boldsymbol{\xi}) \phi_q(\zeta_k) \prod_{\substack{j=1 \\ j\neq k}}^{6} f^l_j(\zeta_j, t_{n+1}) \right\} \mathrm{d}\boldsymbol{\zeta} \\ &\quad + 2\Delta t \int_{\Omega_\pi} C(\boldsymbol{\zeta}, t_n) A(\boldsymbol{\xi}) \phi_q(\zeta_k) \prod_{\substack{j=1 \\ j\neq k}}^{6} f^l_j(\zeta_j, t_{n+1}) \mathrm{d}\boldsymbol{\zeta}.\end{aligned} \tag{30}$$

The 6D integrals under the sum are, as before, a sum of the products of 1D integrals, because L is separable with rank 3.

3.2. Evaluation of the BGK collision term

The BGK collision term $C(\boldsymbol{\zeta}, t_n)$ in (30),

$$C(\boldsymbol{\zeta}, t_n) = \frac{\nu(\mathbf{x}, t_n)}{\mathrm{Kn}} \left[f_{\mathrm{eq}}(\mathbf{x}, \boldsymbol{\xi}, t_n) - f(\mathbf{x}, \boldsymbol{\xi}, t_n) \right], \tag{31}$$

is evaluated by using a canonical tensor decomposition. The equilibrium distribution f_{eq}, defined in (16), is the product of one 3D function and three 4D functions,

$$f_{\mathrm{eq}}(\mathbf{x}, \boldsymbol{\xi}, t_n) = \frac{n(\boldsymbol{x}, t_n)}{[2\pi T(\mathbf{x}, t_n)/\mathrm{Bo}]^{3/2}} \prod_{m=1}^{3} \exp\left(-\mathrm{Bo} \frac{[\xi_m - U_m(\mathbf{x}, t_n)]^2}{2T(\mathbf{x}, t_n)} \right). \tag{32}$$

Each of these terms are expanded in a canonical tensor series once the number density, velocity and temperature are computed using (17)–(19). The integrals in these expressions are reduced to the products of 1D integrals, once the canonical tensor expansion (23) is available. To calculate the normalization by $n(\mathbf{x}, t_n)$ in (17)–(19) and the normalization by $T(\mathbf{x}, t_n)$ in (32), we employ a Fourier collocation method with N points in each variable. That turns all the integrals into Riemannian sums, and the functions $f^l_k(\zeta_k, t_n)$ into

$$f^l_k(\zeta_{k,j}, t_n) = \frac{1}{4\pi} \left[\sum_{s=-N/2+1}^{N/2} \beta^l_{k,s}(t_n)\, \mathrm{e}^{is\zeta_{k,j}} + \sum_{s=-N/2}^{N/2-1} \beta^l_{k,s}(t_n) \mathrm{e}^{is\zeta_{k,j}} \right] \tag{33}$$

with inverse

$$\beta^l_{k,s}(t_n) = h \sum_{j=1}^{N} f^l_k(\zeta_{k,j}, t_n) \mathrm{e}^{-is\zeta_{k,j}}. \tag{34}$$

Here, $h = 2\pi/N$ and $\zeta_{k,j} = -\pi + jh$ with $j = 1, \ldots, N$. The form of (33) with two different summation intervals is chosen to ensure that the highest wave-number is treated symmetrically [62]. The forward and backward transform is performed efficiently with the Fast Fourier Transform (FFT) and its inverse. The collision frequency $\nu(\mathbf{x}, t) = Kn(\mathbf{x}, t) T(\mathbf{x}, t)^{1-\mu}$ is also represented by a canonical tensor series, once $n(\mathbf{x}, t)$ and $T(\mathbf{x}, t)$ are available. To speed up the tensor decomposition algorithm, the result from the previous time step is used as the initial guess for the new decomposition.

3.3. Parallel ALS algorithm

Our implementation of the parallel ALS algorithm for solving the Boltzmann transport equation is encapsulated in Algorithm 1.[2] The subroutine Initialization initializes all the variables needed to run the code. This includes, initialization of the time-step size Δt, the number of time steps $n_{\max}$, the number of collocation points, and the initialization of the coefficients β_{New}, β_{Now}, and β_{Old}. These coefficients store, respectively, the values of $f(\boldsymbol{\zeta}, t_{n+1})$, $f(\boldsymbol{\zeta}, t_n)$, and $f(\boldsymbol{\zeta}, t_{n-1})$ in Fourier space. In addition, the operators A^+, A^-, and Exp_{A^+} are initialized. The operators A^+ and A^- represent $I \pm \Delta t\, L(\boldsymbol{\xi})$, respectively. The rank separated operator Exp_{A^+} represents operator A^+ acting on the Fourier basis function ϕ_s. The representation of A^+ as a single operator reduces the loss of accuracy that comes with multiplying different operators.

[2] Details of the algorithmic implementation of our method and description of the various subroutines mentioned herein are provided in the Supplemental Material.

Algorithm 1 Parallel ALS algorithm.

procedure MAIN
  INITIALIZATION ▷ Load variables and allocate matrices

  for $n \leftarrow 1 : n_{\max}$ **do**
    $C \leftarrow$ COMPUTEARRAYC(β_{Now}) ▷ Compute collision operator

    $\beta_{\text{New}} \leftarrow$ RANDBETA(β_{New}) ▷ Add some random noise
    $\epsilon_{|\beta|} \leftarrow \epsilon_{\text{Tol}} + 10^6$ ▷ Reset stop criterion
    while $\epsilon_{|\beta|} > \epsilon_{\text{Tol}}$ **do**
      $\beta_{\text{Int}} \leftarrow \beta_{\text{New}}$ ▷ Set intermediate value of beta

      $N \leftarrow$ COMPUTEARRAYN($A^+, A^-, \text{Exp}_{A^+}, \beta_{\text{Old}}, \beta_{\text{New}}$)
      $O \leftarrow$ COMPUTEARRAYO($C, A^+, \text{Exp}_{A^+}, \beta_{\text{New}}$)
      $\gamma \leftarrow N + 2\,\Delta t\, O$

      parfor $d \leftarrow 1 : 6$ **do** ▷ Iterate over dimensions
        $M(d) \leftarrow$ COMPUTEARRAYM($A^+, \text{Exp}_{A^+}(d), \beta_{\text{New}}, d$)

      parfor $d \leftarrow 1 : 6$ **do**
        $\beta_{\text{New}}(d) \leftarrow$ COMPUTEBETANEW($\beta_{\text{New}}(d), M(d), \gamma(d)$)

      $\beta_{\text{New}} \leftarrow$ KREAL(β_{New}) ▷ Keep only real part of solution
      $\epsilon_{|\beta|} \leftarrow$ COMPUTENORMBETA($\beta_{\text{Int}}, \beta_{\text{New}}$) ▷ Check for convergence

      if $\epsilon_{|\beta|} > \epsilon_{\text{Tol}}$ **then**
        $\beta_{\text{New}} \leftarrow$ COMPUTEBETANEWDELTA($\beta_{\text{New}}, \beta_{\text{Int}}$) ▷ Update
      else
        $\beta_{\text{New}}, \beta_{\text{Now}} \leftarrow$ COMPUTERAW($\beta_{\text{New}}, \beta_{\text{Now}}, \beta_{\text{Old}}$) ▷ Apply RAW filter

    $\beta_{\text{Old}} = \beta_{\text{Now}}$ ▷ Update for next time step
    $\beta_{\text{Now}} = \beta_{\text{New}}$
end procedure

For every time step, COMPUTEARRAYC evaluates the BGK collision operator $C(\boldsymbol{\zeta}, t_n)$ in (31), and RANDBETA adds a small amount of noise to β_{New} to prevent the algorithm from getting stuck in a local minimum. The WHILE loop contains the parallel ALS algorithm. The functions COMPUTEARRAYN and COMPUTEARRAYO inside the loop compute the two different contributions to the vector γ in (30). The function COMPUTEARRAYM computes the array M inside a parallel FOR loop, which completes the set of equations (28).

The least-squares routine inside the function COMPUTEBETANEW updates β_{New} every iteration inside a parallel FOR loop. Because of the large number of degrees of freedom in the above system, there are multiple (local minimum) solutions, which are not necessarily real and mass conserving. We ameliorate this problem by adding a constraint that, for every iteration, only the real part of the solution for β_{New} is kept. This calculation is performed by the function KREAL. The function COMPUTENORMBETA then computes the convergence criterion, $\epsilon_{|\beta|}$, by comparing the current value of β_{New} to its value at the end of the previous iteration, β_{Int}. If convergence has not been reached, COMPUTEBETANEWDELTA updates β_{New} for the next iteration. If convergence has been reached, COMPUTERAW applies the RAW filter [57–59] to make the CNLF algorithm unconditionally stable, and the algorithm moves on to the next time step.

4. Numerical results

In this section we study the accuracy and computational efficiency of the proposed ALS-CNLF tensor method to solve the Boltzman-BGK equation. (Unless specified otherwise, values of the simulation parameters used in our numerical experiments are collated in Table 1.) We start by validating our solver on a prototype problem involving the Boltzmann-BGK equation in one spatial dimension. Then we consider simulations of the full Boltzmann-BGK equation in three spatial dimensions. The 3D numerical results are split up into steady-state and transient problems. The steady-state problem is used to validate the code against an analytical equilibrium solution to the Boltzmann transport equation. It also enables us to investigate the error convergence as function of various parameters, and the code performance as function of the number of collocation points and the number of processor cores. The transient problem starts with an initial distribution away from equilibrium, including a non-zero average velocity in the x_1 direction, and looks at temporal evolution of the PDF f. We compare our code with a spectral code in 2D and present the results for the full 6D Boltzmann-BGK equation.

4.1. Transient dynamics in one spatial dimension

To validate the proposed Boltzmann-BGK tensor solver, we first compute the numerical solution of (15) in one spatial dimension, and compare it with an accurate benchmark solution obtained with the high-order Fourier pseudo-spectral method [60]. Specifically, we study the initial-value problem

Table 1
Values of parameters used in the simulations of the Boltzmann-BGK equation in three spatial dimensions.

Variable	Value	Description
Δt	0.025	Dimensionless time step
K	1.0	Collision frequency pre-factor
μ	0.5	Collision frequency temperature exponent
Kn	1	Knudsen number
Bo	3.65	Boltzmann number
ϵ_{Tol}	$5.0 \cdot 10^{-5}$	Tolerance on ALS iterations

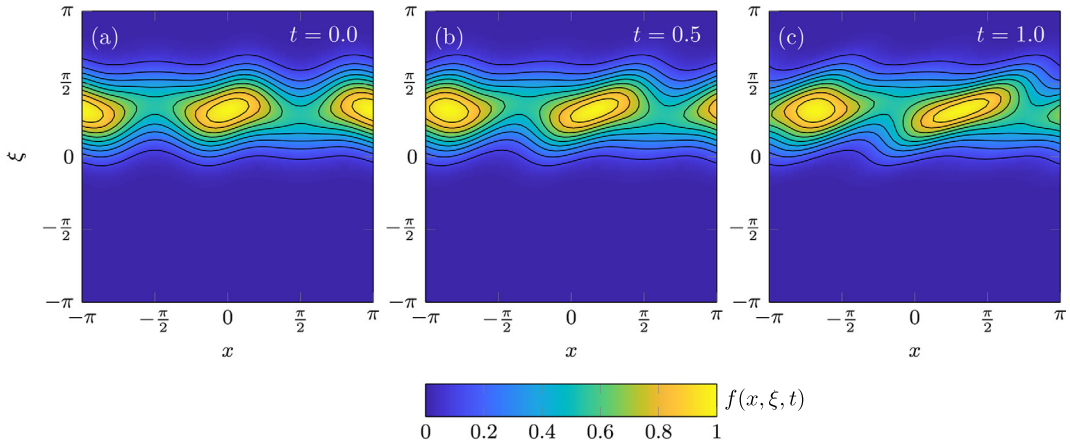

Fig. 1. Contour plots of the reference solution to the 1D Boltzmann-BGK initial-value problem (35)–(37), the PDF $f_{\text{spc}}(x, \xi, t)$, computed with the high-order Fourier pseudo-spectral method and explicit two-steps Adams-Bashforth temporal integrator. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

$$\frac{\partial f(x,\xi,t)}{\partial t} + \xi \frac{\partial f(x,\xi,t)}{\partial x} = \frac{K T(x,t)^{1-\mu}}{\text{Kn}} \left[f_{\text{eq}}(x,\xi,t) - f(x,\xi,t) \right], \tag{35}$$

$$f(x,\xi,0) = \frac{n_0}{\sqrt{(2\pi T_0/\text{Bo})}} \exp\left(-\frac{\text{Bo}}{2T_0} (U_0 - \xi)^2 \right), \tag{36}$$

with

$$n_0 = 1.0 + 0.3\cos(2x), \qquad U_0 = 1.0 + 0.1\sin(3x), \qquad T_0 = 1.0. \tag{37}$$

The benchmark numerical solution is constructed by solving (35)–(37) in the periodic box $(x, \xi) \in [-\pi, \pi]^2$ with a Fourier pseudo-spectral method [60] (odd expansion on a 61×61 grid) and an explicit two-steps Adams-Bashforth temporal integrator with $\Delta t = 0.0005$. The remaining parameters are set to the values from Table 1, except $\text{Kn} = 10$. Fig. 1 exhibits contour plots of the benchmark PDF solution at three times.

We report the accuracy of our ALS-CNLF tensor algorithm[3] by comparing its prediction, f_{ALS}, with the reference solution, f_{spc}, in terms of two metrics. The first is the L^2 norm

$$\left\| f_{\text{ALS}}(x,\xi,t) - f_{\text{spc}}(x,\xi,t) \right\| = \left(\int_{[-\pi,\pi]^2} \left[f_{\text{ALS}}(x,\xi,t) - f_{\text{spc}}(x,\xi,t) \right]^2 \mathrm{d}x\mathrm{d}\xi \right)^{1/2}. \tag{38}$$

The second is the Kullback-Leibler divergence

$$D_{\text{KL}}(F_{\text{ALS}} || F_{\text{spc}}) = \int_{[-\pi,\pi]^2} f_{\text{ALS}}(x,\xi,t) \log\left[\frac{f_{\text{ALS}}(x,\xi,t)}{f_{\text{spc}}(x,\xi,t)} \right] \mathrm{d}x\mathrm{d}\xi. \tag{39}$$

In Fig. 2, we plot these two metrics as function of time t and the tensor rank r, which is kept constant throughout the simulation. Both the L^2 error and the Kullback-Leibler divergence decrease with the tensor rank r.

[3] The tolerance for the ALS iterations is set to $\epsilon_{\text{Tol}} = 10^{-12}$, while the other simulation parameters, such as Δt and the number of grid points are the same as in the Fourier pseudo-spectral method.

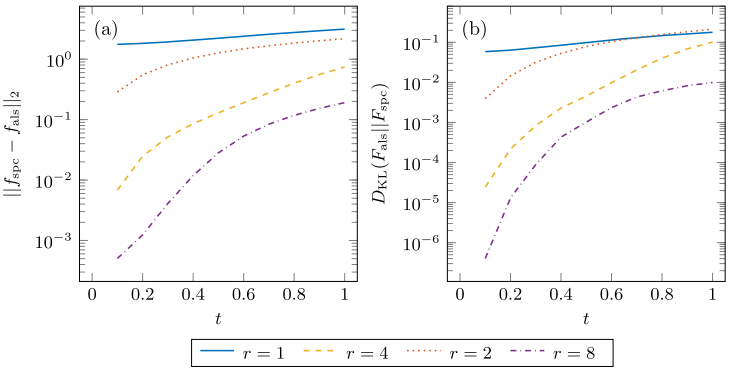

Fig. 2. Boltzmann-BGK problem (35)–(37) in one spatial dimension: L^2 error and Kullback-Leibler divergence of the ALS-CNLF tensor solution, f_{ALS}, relative to the reference solution, f_{spc}, versus time. Both the L^2 error and the Kullback-Leibler divergence decrease with the tensor rank r.

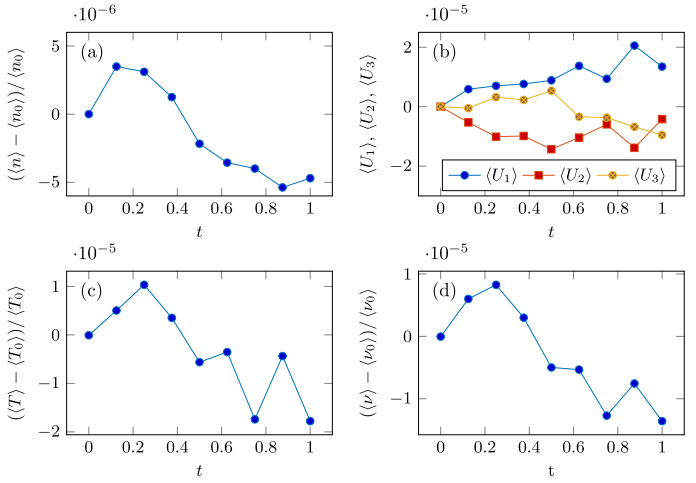

Fig. 3. Temporal variability of relative errors in the spatial averages of density, $\langle n \rangle$, velocity components, $\langle U_1 \rangle$, $\langle U_2 \rangle$, and $\langle U_3 \rangle$, and collision frequency, $\langle \nu \rangle$, as well as of the spatially averaged temperature, $\langle T \rangle$. The number of collocation points per dimension is $N = 64$. The results show that the code is mass conserving, i.e., the moments of the PDF f at equilibrium are constant in time up to a small (10^{-5}) error.

4.2. Steady-state simulations in three spatial dimensions

Since convergence of the ALS algorithm is not guaranteed, e.g., [54,63], and since the equilibrium distribution is one of the few analytical solutions to the Boltzmann transport equation in six dimensions, we report the code's behavior at equilibrium. The initial condition in this experiment is the Maxwell-Boltzmann equilibrium PDF, whose moments, as defined in (17)–(19), are set to $n(\mathbf{x}, 0) = 1$, $U_1(\mathbf{x}, 0) = U_2(\mathbf{x}, 0) = U_3(\mathbf{x}, 0) = 0$, and $T(\mathbf{x}, 0) = 1$. The simulation was ran till $t = 1$. Fig. 3 shows temporal variability of the relative errors of the spatial averages, $\langle n \rangle$ and $\langle T \rangle$, of the moments n and T and that of the collision frequency, $\langle \nu \rangle$. Since the initial values of the velocities $\langle U_1 \rangle$, $\langle U_2 \rangle$ and $\langle U_3 \rangle$ are zero, Fig. 3(b) shows only their average values as function of time. For any quantity $a(\mathbf{x}, t)$, the spatial average is defined as

$$\langle a(t) \rangle = \frac{1}{(2\pi)^3} \int_{[-\pi,\pi]^3} a(\mathbf{x}, t) \mathrm{d}\mathbf{x}. \tag{40}$$

As expected, all the spatially averaged moments at equilibrium remain approximately constant with time, with small (on the order of 10^{-5}) deviations representing the numerical error.

Another metric of the accuracy of our steady-state equilibrium solution, the root-mean-square error (RMSE) of the solution, is shown in Fig. 4 as function of time. The RMSE is defined as

$$\text{RMS}(f - f_0) = \sqrt{\frac{1}{N^6} \sum_{i=1}^{N^6} (f_i - f_{0,i})^2}, \tag{41}$$

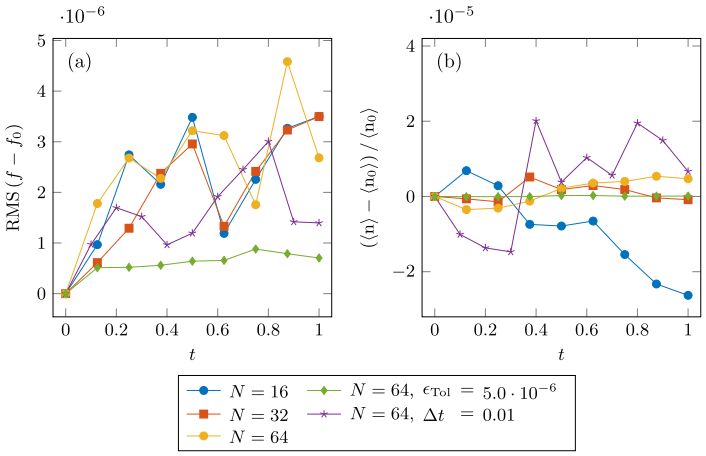

Fig. 4. (a) Temporal variability of the RMSE between the computed PDF f and the initial PDF f_0. Unless otherwise mentioned in the legend, $\Delta t = 0.025$ and $\epsilon_{\text{Tol}} = 5.0 \cdot 10^{-5}$. The solution error is relatively independent from the number of collocation points per dimension, N. However, reducing the time step Δt and, to a larger extent, the convergence criterion ϵ_{Tol} significantly reduces the solution error. (b) Mass conservation as function of time t. The results show a clear improvement in mass conservation going from $N = 16$ to $N = 32$, but not from $N = 32$ to $N = 64$. Decreasing the time step to $\Delta t = 0.01$ does not reduce mass loss, while reducing the convergence criterion to $\epsilon_{\text{Tol}} = 5.0 \cdot 10^{-6}$ significantly improves mass conservation.

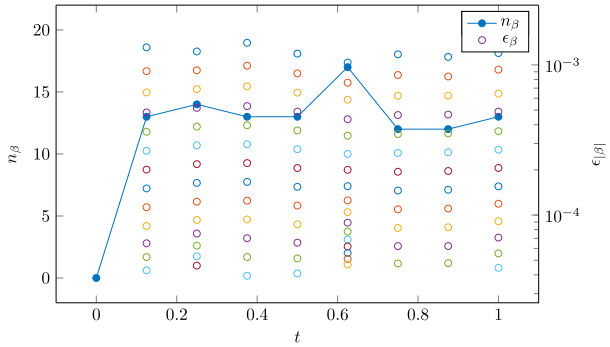

Fig. 5. Temporal variability of the number of iterations, n_β, and convergence, $\epsilon_{|\beta|}$. The number of collocation points per dimension is $N = 64$. Since the same amplitude of random noise in β is injected at every time step, the number of iterations is nearly constant in time. The right axis has logarithmic scale, and the distance between each iteration decreases as convergence is reached. This suggests that convergence follows an exponential decay.

where f_0 is the initial condition. The simulation was performed from $N = 16$ to $N = 64$ collocation points per dimension. However, the number of collocation points does not have a significant effect on the accuracy of the algorithm (Fig. 4(a)). Decrease in the time step, from $\Delta t = 0.025$ to $\Delta t = 0.01$, does not have much effect either. On the other hand, reducing the convergence criterion for β from $\epsilon_{\text{Tol}} = 5.0 \cdot 10^{-5}$ to $\epsilon_{\text{Tol}} = 5.0 \cdot 10^{-6}$ significantly lowers the error. This suggests the presence of a bottleneck in increasing the accuracy of the simulation caused by the convergence criterion. We show in section 4.3 that the bottleneck for higher accuracy can depend on the physical system.

Since the continuity equation is used as an additional constraint, we explore the behavior of the RMSE of the spatially averaged density $\langle n \rangle$. Fig. 4(b) shows that our algorithm satisfies mass conservation even at the lowest number of collocation points, $N = 16$. As the number of collocation points increases from $N = 16$ to $N = 32$, the error in the mass conservation is significantly reduced. However, the further increase in the number of collocation points, from $N = 32$ to $N = 64$, hardly has an effect. Also, decreasing the time step from $\Delta t = 0.025$ to $\Delta t = 0.01$ does not reduce the error. However, as was also observed in Fig. 4(a), reducing the convergence criterion for β from $\epsilon_{\text{Tol}} = 5.0 \cdot 10^{-5}$ to $\epsilon_{\text{Tol}} = 5.0 \cdot 10^{-6}$ significantly reduces the amount of mass loss. This again confirms that the convergence criterion serves as a bottleneck in increasing the simulation accuracy. This finding highlights the importance of picking a convergence-criterion value that satisfies the desired balance between available computational resources and accuracy. A smaller convergence criterion reduces the error, but results in more iterations to reach convergence and, potentially, in a higher rank of the solution, resulting in higher memory usage.

The number of iterations, n_β, needed to reach convergence throughout the simulation is reported in Fig. 5 (the left vertical axis) as function of time t. Since the same amount of random noise is added to β before starting the ALS algorithm at every time step, the number of iterations is nearly constant. Adding a smaller amount of random noise might reduce the number of iterations, but our numerical experiments revealed that doing so causes the ALS algorithm to get stuck in a local minimum and precludes the residual of the continuity equation, $\epsilon_{|\beta|}$, from being properly minimized. Fig. 5 (the right

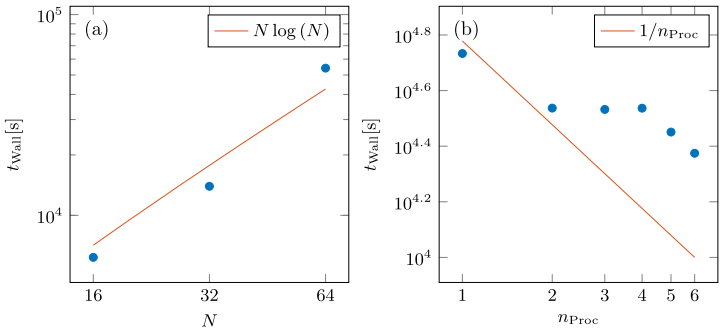

Fig. 6. Scaling of the wall time t_{Wall} with (a) the number of collocation points per dimension, N, and (b) the number of processors, n_{Proc}. The calculations to determine the scaling of the wall time as function of the number of collocation points were performed on a single core.

vertical axis) shows that, as the convergence is reached, the difference in the residual between the successive iterations gets smaller. This suggests an exponential decay towards the minimum residual that can be reached before the rank of the solution needs to be increased.

The computational efficiency of our code is reported in Fig. 6. The left frame shows the scaling of the wall time, t_{Wall}, with the number of collocation points per dimension, N, on one CPU core. The performance of the code is close to $N \log(N)$ in the range of the explored collocation points. The most time is spent in the LSQR [64] subroutine, which is used to implicitly solve for β for each dimension during every iteration of the ALS procedure. This suggests that the code could be further optimized by replacing the existing LSQR algorithm with its more efficient implementation, e.g., [65]. The right frame of Fig. 6 exhibits the scaling of the wall time, t_{Wall}, with the number of processors, n_{Proc}. The curve $1/n_{\text{Proc}}$ represents the ideal scaling without any communication overhead. Even though the different dimensions can all be solved for independently, the scaling of our code with the number of processors is quite poor. This suggests that the code can be optimized further by minimizing the communication between different CPU cores. One way to approach this would be to rewrite the code around a specialized parallel processing MPI library and have finer control over both data communication between processor cores and protocol selection.

4.3. Relaxation to statistical equilibrium in three spatial dimensions

In this section, we study relaxation to statistical equilibrium predicted by the dimensionless Boltzmann-BGK model (15)–(19), subject to the initial condition

$$f(\mathbf{x}, \boldsymbol{\xi}, 0) = W f_1(x_1, \xi_1, 0) f_2(x_2, \xi_2, 0) f_3(x_3, \xi_3, 0) \tag{42a}$$

with

$$f_i(x_i, \xi_i, 0) = \frac{\sqrt[3]{n_0}}{\sqrt{2\pi T_0/\text{Bo}}} \exp\left[-\frac{\text{Bo}}{2T_0}(U_{i,0} - \xi_i)^4\right], \qquad i = 1, 2, 3 \tag{42b}$$

and $n_0 = \prod_{i=1}^{3}(0.5\cos x_i + 1)$, $T_0 = 0.0025\cos x_1$, $U_{1,0} = 1 + 0.025\sin(x_2 - 1)$, $U_{2,0} = 0$, and $U_{3,0} = 0.025\sin(x_1 - 2)$. The integral over the PDF is normalized to 1 by computing the value of W. The difference between the initial PDF and the local equilibrium PDF causes the Boltzmann equation to evolve while the fluctuations in the initial fields are introduced to show the code's ability to operate away from global equilibrium. In this experiment, the Knudsen number is set to $\text{Kn} = 10$. Fig. 7 shows the PDF $f(\mathbf{x}, \boldsymbol{\xi}, t)$ in the $(x_1\text{-}x_2)$ and $(\xi_1\text{-}\xi_2)$ hyper-planes. The PDF f evolves from its initial state far from equilibrium to the equilibrium Maxwell-Boltzmann PDF f_{eq}. Fig. 8 further elucidates this dynamics by exhibiting temporal snapshots of the PDF $f(\mathbf{x}, \boldsymbol{\xi}, t)$ in hyper-planes $(\mathbf{x}; \boldsymbol{\xi}) = (x_1, 0, 0; \mathbf{0})$, $(0, x_2, 0; \mathbf{0})$, $(0, 0, x_3; \mathbf{0})$, $(\mathbf{0}; \xi_1, 0, 0)$, $(\mathbf{0}; 0, \xi_2, 0)$, and $(\mathbf{0}; 0, 0, \xi_3)$.

Finally, Fig. 9 exhibits temporal evolution of the number density $n(\mathbf{x}, t)$, velocity $\mathbf{U}(\mathbf{x}, t)$, temperature $T(\mathbf{x}, t)$, and the collision frequency $\nu(\mathbf{x}, t)$, all evaluated at the hyper-plane $\mathbf{x} = (x_1, 0, 0)$. As expected, the magnitude of the velocity components U_2 and U_3 is about 100 times smaller than that of the U_1 component (frames (a)–(c)). The effect of the flow in the x_1 direction is seen in frame (d), where the density profile shifts to the right as time progresses; in addition, the height of the density profile decreases as the mass is redistributed by diffusion. While not discernible from these figures, the code does suffer from a small amount of mass loss of about 2% per unit time. This is most likely due to a combination of small truncation errors, convergence tolerances, and time step size. Frame (e) shows evolution of the temperature $T(x_1, \cdot, t)$ from its initial nearly uniform state. Increasing velocity fluctuations drive the increase in temperature fluctuations. The collision frequency (frame (f)) is computed directly from the density and temperature fields according to (12). Overall, the PDF and its moments show the expected physical behavior and evolve towards the equilibrium Maxwell-Boltzmann distribution.

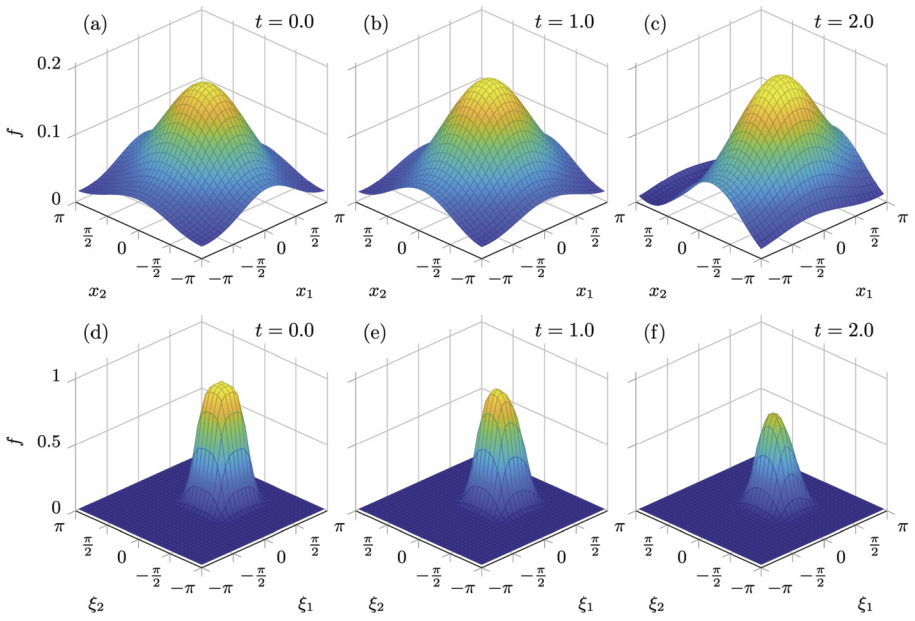

Fig. 7. Temporal evolution of the PDF $f(\mathbf{x}, \boldsymbol{\xi}, t)$ in hyper-planes $(\mathbf{x}; \boldsymbol{\xi}) = (x_1, x_2, 0; \mathbf{0})$ (top row) and $(\mathbf{0}; \xi_1, \xi_2, 0)$ (bottom row). The number of collocation points per dimension is $N = 32$. The color-coding is consistent among the different frames in a row. The PDF f evolves from its initial, far-from-equilibrium state (42) towards its equilibrium state (16).

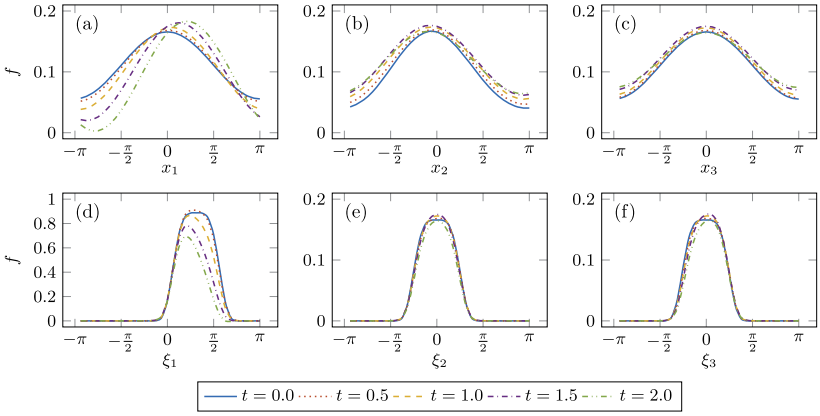

Fig. 8. Temporal evolution of the PDF $f(\mathbf{x}, \boldsymbol{\xi}, t)$ in hyper-planes (a) $(\mathbf{x}; \boldsymbol{\xi}) = (x_1, 0, 0; \mathbf{0})$, (b) $(\mathbf{x}; \boldsymbol{\xi}) = (0, x_2, 0; \mathbf{0})$, (c) $(\mathbf{x}; \boldsymbol{\xi}) = (0, 0, x_3; \mathbf{0})$, (d) $(\mathbf{x}; \boldsymbol{\xi}) = (\mathbf{0}; \xi_1, 0, 0)$, (e) $(\mathbf{x}; \boldsymbol{\xi}) = (\mathbf{0}; 0, \xi_2, 0)$, and (f) $(\mathbf{x}; \boldsymbol{\xi}) = (\mathbf{0}; 0, 0, \xi_3)$. The number of collocation points per dimension is $N = 32$. The PDF f evolves from its initial, far-from-equilibrium state (42) towards its equilibrium state (16).

5. Discussion and conclusions

We demonstrated the ability of canonical polyadic tensor decomposition to solve the BGK approximation of the six-dimensional Boltzmann transport equation with variable density, velocity, temperature, and collision frequency fields. This extends the usage of this method from only fully separable differential operators, to partially separable operators. The dependent variables in the Boltzmann equation are computed via the pseudo-spectral method with collocation points. The different dimensions are solved using a parallel ALS algorithm. Our numerical experiments show that the code is capable of simulating both a system's steady state and its temporal evolution starting from a state far away from equilibrium. The highest simulation accuracy is achieved by identifying a correct bottleneck: in the steady-state simulations the limiting factor is the convergence tolerance. The code's performance scales as $N \log(N)$ with the number of collocation points per dimension, N.

In future work, we will further optimize the code, implement different boundary conditions, and explore the feasibility of using the code with other collision operators. Significant speedup can be achieved by rank reduction of the collision operator. The currently used reconstruction of the collision operator via multiplication of its different components can result in a high-rank operator, which can become degenerate. Reducing the rank of this tensor, while maintaining accuracy,

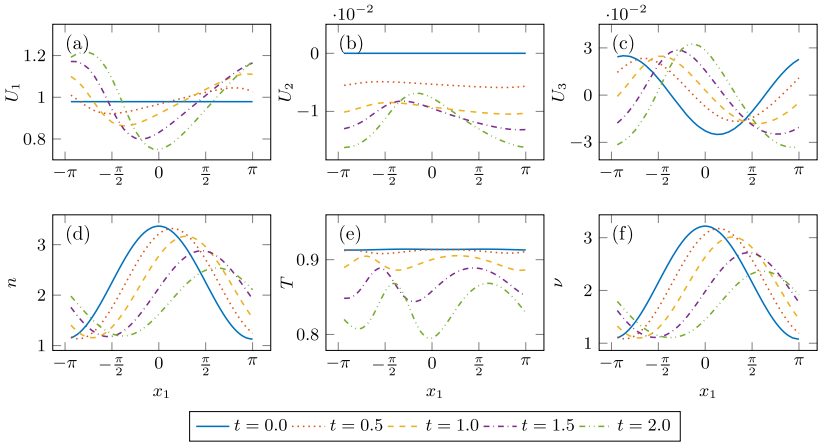

Fig. 9. Temporal evolution of (a)–(c) the velocity components $U_i\mathbf{x}, t)$, (d) number density $n(\mathbf{x}, t)$, (e) temperature $T(\mathbf{x}, t)$, and (f) the collision frequency $\nu(\mathbf{x}, t)$, all evaluated at the hyper-plane $\mathbf{x} = (x_1, 0, 0)$. The number of collocation points per dimension is $N = 32$.

is a topic that needs further study. Another aspect where the code can be improved is parallelization, especially when using the CNLF-ALS method to solve problems of dimensionality higher than that of the Boltzmann-BGK equation. One way to approach this would be to rewrite the code around a specialized parallel processing MPI library and have finer control over data communication between processor cores and protocol selection.

Generalizations to other types of boundary conditions would require an in-depth investigation of the trial function behavior. The choice of a right trial function for a certain system is typically determined by one of the two approaches. First, one selects a trial function that obeys the desired boundary conditions and then sums over the trial functions to reconstruct the solution to a PDE. Second, one selects a trial function that captures the PDE solution and then adds trial functions to reconstruct the boundary condition. Fig. 8 shows that the six-dimensional Boltzmann equation subject to periodic boundary conditions can be efficiently solved by using the discrete Fourier series. The presence of, e.g., a wall would introduce the Maxwell boundary condition [35,66]. The latter consists of two parts: one represents the reflection of particles from the wall and the other represents absorption and emission of particles on the wall. Since there is no known trial function to either solve the Boltzmann equation in the bulk or the Maxwell boundary condition on the wall, another method of the trial-function selection is needed. Such methods include the tau method [67], the penalty approach [68], and the mixed method [69–71]. They have been used to implement boundary condition for the weighted residuals and spectral methods [72,68], but have not yet been applied in the tensor decomposition setting.

While the boundary conditions are very important for engineering applications, some fundamental questions remain to be resolved as well. The empirical evidence shows convergence of our tensor algorithms, but its theoretical proof is lacking. That is largely due to two factors. First, it is generally assumed that increasing the rank of the solution improves its accuracy and in, some cases, tensor decomposition was shown to be exponentially more efficient than one would expect a priori [22]. However, there is no mathematical proof which indicates how high the rank needs to be to reach a certain accuracy and how fast the solution converges as a function of its rank. This question is very important for the speed of tensor decomposition, because the higher the rank of the operators and the solution of the Boltzmann equation, the slower the computation becomes. Second, from one time step to the next convergence is not guaranteed. The ALS method implemented in our code is a variation on the Gauss-Seidel method whose convergence can be proven for some specific cases [54]. However, only the Jacobi method can decouple the different dimensions and parallelize the algorithm. How the adoption of this method would change convergence remains to be investigated.

CRediT authorship contribution statement

Arnout M.P. Boelens: Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing - original draft. **Daniele Venturi:** Conceptualization, Funding acquisition, Software, Writing - review & editing. **Daniel M. Tartakovsky:** Conceptualization, Funding acquisition, Project administration, Supervision, Writing - review & editing.

Declaration of competing interest

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

Acknowledgements

This research was supported in part by the U.S. Army Research Office (ARO) grant W911NF1810309 awarded to Daniele Venturi, and by the Air Force Office of Scientific Research (AFOSR) grant FA9550-18-1-0474 awarded to Daniel Tartakovsky.

Appendix A. Supplementary material

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.jcp.2020.109744.

References

[1] A.M.P. Boelens, A. Latka, J.J. de Pablo, Observation of the pressure effect in simulations of droplets splashing on a dry surface, Phys. Rev. Fluids 3 (2018) 063602.
[2] J.E. Sprittles, Kinetic effects in dynamic wetting, Phys. Rev. Lett. 118 (11) (2017) 114502.
[3] I.D. Boyd, G. Chen, G.V. Candler, Predicting failure of the continuum fluid equations in transitional hypersonic flows, Phys. Fluids 7 (1) (1995) 210–219.
[4] E.H. Sondheimer, The mean free path of electrons in metals, Adv. Phys. 1 (1) (1952) 1–42.
[5] S. Jin, T.-W. Tang, M.V. Fischetti, Simulation of silicon nanowire transistors using Boltzmann transport equation under relaxation time approximation, IEEE Trans. Electron Devices 55 (3) (2008) 727–736.
[6] L. Boltzmann, Weitere Studien über das Wärmegleichgewicht unter Gasmolekülen, vorgelegt in: der Sitzung am 10. October 1872, 1872, k. und k. Hof- und Staatsdr.
[7] G. Dimarco, L. Pareschi, Numerical methods for kinetic equations, Acta Numer. 23 (2014) 369–520.
[8] G.A. Bird, Approach to translational equilibrium in a rigid sphere gas, Phys. Fluids 6 (10) (1963) 1518–1519.
[9] H. Babovsky, On a simulation scheme for the Boltzmann equation, Math. Methods Appl. Sci. 8 (1986) 223–233.
[10] L. Pareschi, G. Russo, An Introduction to Monte Carlo Method for the Boltzmann Equation, ESAIM: Proceedings, vol. 10, EDP Sciences, 2001, pp. 35–75.
[11] J.-P.M. Peraud, C.D. Landon, N.G. Hadjiconstantinou, Monte Carlo methods for solving the Boltzmann transport equation, Annu. Rev. Heat Transf. 2 (2) (2014) 205–265.
[12] F. Rogier, J. Schneider, A direct method for solving the Boltzmann equation, Transp. Theory Stat. Phys. 23 (1–3) (1994) 313–338.
[13] S. Rjasanow, W. Wagner, Stochastic Numerics for the Boltzmann Equation, Springer, Berlin, 2005.
[14] H. Cho, D. Venturi, G.E. Karniadakis, Numerical methods for high-dimensional probability density function equations, J. Comput. Phys. 305 (2016) 817–837.
[15] Z. Zhang, G.E. Karniadakis, Numerical Methods for Stochastic Partial Differential Equations with White Noise, Springer, Berlin, 2017.
[16] W. E, J. Han, A. Jentzen, Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations, Commun. Math. Stat. 5 (4) (2017) 349–380.
[17] P.L. Bhatnagar, E.P. Gross, M. Krook, A model for collision processes in gases. I. Small amplitude processes in charged and neutral one-component systems, Phys. Rev. 94 (3) (1954) 511.
[18] A.M.P. Boelens, D. Venturi, D.M. Tartakovsky, Parallel tensor methods for high-dimensional linear PDEs, J. Comput. Phys. 375 (2018) 519–539.
[19] M.J. Reynolds, A. Doostan, G. Beylkin, Randomized alternating least squares for canonical tensor decompositions: application to a PDE with random data, SIAM J. Sci. Comput. 38 (5) (2016) A2634–A2664.
[20] C. Battaglino, G. Ballard, T.G. Kolda, A practical randomized CP tensor decomposition, SIAM J. Matrix Anal. Appl. 39 (2) (2018) 876–901.
[21] E. Acar, D.M. Dunlavy, T.G. Kolda, A scalable optimization approach for fitting canonical tensor decompositions, J. Chemom. 25 (2) (2011) 67–86.
[22] G. Beylkin, J. Garcke, M.J. Mohlenkamp, Multivariate regression and machine learning with sums of separable functions, SIAM J. Sci. Comput. 31 (2009) 1840–1857.
[23] J. Douglas Jr, T. Dupont, Alternating-direction Galerkin methods on rectangles, in: Numerical Solution of Partial Differential Equations–II, Elsevier, 1971, pp. 133–214.
[24] D. Venturi, The numerical approximation of nonlinear functionals and functional differential equations, Phys. Rep. 732 (2018) 1–102.
[25] B.A. Finlayson, The Method of Weighted Residuals and Variational Principles, SIAM, Philadelphia, 2013.
[26] A. Dektor, D. Venturi, Dynamically orthogonal tensor methods for high-dimensional nonlinear PDEs, J. Comput. Phys. 404 (2020) 109125.
[27] A. Rodgers, D. Venturi, Stability analysis of hierarchical tensor methods for time-dependent PDEs, J. Comput. Phys. 409 (2020) 109341.
[28] M. Akian, J. Blechschmidt, N.D. Botkin, M. Jensen, A. Kröner, A. Picarelli, I. Smears, K. Urban, M.D. Chekroun, R. Herzog, et al., Hamilton-Jacobi-Bellman Equations: Numerical Methods and Applications in Optimal Control, Vol. 21, Walter de Gruyter & Co, 2018.
[29] H. Risken, The Fokker-Planck Equation: Methods of Solution and Applications, Springer, Berlin, 1996.
[30] D.R. Hatch, D. del Castillo-Negrete, P.W. Terry, Analysis and compression of six-dimensional gyrokinetic datasets using higher order singular value decomposition, J. Comput. Phys. 22 (2012) 4234–4256.
[31] S.V. Dolgov, A.P. Smirnov, E.E. Tyrtyshnikov, Low-rank approximation in the numerical modeling of the Farley-Buneman instability in ionospheric plasma, J. Comput. Phys. 263 (2014) 268–282.
[32] K. Kormann, A semi-Lagrangian Vlasov solver in tensor train format, SIAM J. Sci. Comput. 37 (4) (2015) B613–B632.
[33] A.S. Monin, A.M. Yaglom, Statistical Fluid Mechanics, Volume II: Mechanics of Turbulence, Dover, New York, 2007.
[34] M. Breakspear, Dynamic models of large-scale brain activity, Nat. Neurosci. 20 (3) (2017) 340.
[35] C. Cercignani, The Boltzmann Equation and Its Applications, Springer, Berlin, 1988.
[36] C. Cercignani, U.I. Gerasimenko, D.Y. Petrina, Many Particle Dynamics and Kinetic Equations, Kluwer Academic Publishers, New York, 1997.
[37] C. Cercignani, R. Illner, M. Pulvirenti, The Mathematical Theory of Dilute Gasses, Springer, Berlin, 1994.
[38] T. Nishida, Fluid dynamical limit of the nonlinear Boltzmann equation to the level of the compressible Euler equation, Commun. Math. Phys. 61 (2) (1978) 119–148.
[39] R.E. Caflisch, The fluid dynamic limit of the nonlinear Boltzmann equation, Commun. Pure Appl. Math. 33 (5) (1980) 651–666.
[40] H. Struchtrup, Macroscopic Transport Equations for Rarefied Gas Flows, Springer, Berlin, 2005.
[41] C.D. Levermore, Moment closure hierarchies for kinetic theories, J. Stat. Phys. 83 (5–6) (1996) 1021–1065.
[42] L. Mieussens, Discrete-velocity models and numerical schemes for the Boltzmann-BGK equation in plane and axisymmetric geometries, J. Comput. Phys. 162 (2) (2000) 429–466.
[43] J. Nassios, J.E. Sader, High frequency oscillatory flows in a slightly rarefied gas according to the Boltzmann–BGK equation, J. Fluid Mech. 729 (2013) 1–46.
[44] P. Andries, P.L. Tallec, J.-P. Perlat, B. Perthame, The Gaussian-BGK model of Boltzmann equation with small Prandtl number, Eur. J. Mech. B 19 (6) (2000) 813–830.

[45] O. Johansson, H.-O. Kreiss, Über das Verfahren der zentralen Differenzen zur Lösung des Cauchyproblems für partielle Differentialgleichungen, BIT Numer. Math. 3 (2) (1963) 97–107.
[46] W. Layton, C. Trenchea, Stability of two IMEX methods, CNLF and BDF2-AB2, for uncoupling systems of evolution equations, Appl. Numer. Math. 62 (2) (2012) 112–120.
[47] M. Kubacki, Uncoupling evolutionary groundwater-surface water flows using the Crank–Nicolson Leapfrog method, Numer. Methods Partial Differ. Equ. 29 (4) (2013) 1192–1216.
[48] N. Jiang, M. Kubacki, W. Layton, M. Moraiti, H. Tran, A Crank–Nicolson Leapfrog stabilization: unconditional stability and two applications, J. Comput. Appl. Math. 281 (2015) 263–276.
[49] L. Grasedyck, Hierarchical singular value decomposition of tensors, SIAM J. Matrix Anal. Appl. 31 (4) (2010) 2029–2054.
[50] P. Breiding, N. Vannieuwenhoven, A Riemannian trust region method for the canonical tensor rank approximation problem, SIAM J. Optim. 28 (3) (2018) 2435–2465.
[51] A. Uschmajew, B. Vandereycken, The geometry of algorithms using hierarchical tensors, Linear Algebra Appl. 439 (1) (2013) 133–166.
[52] S.T. Smith, Optimization techniques on Riemannian manifolds, Fields Inst. Commun. 3 (3) (1994) 113–135.
[53] Y. Yang, Globally convergent optimization algorithms on Riemannian manifolds: uniform framework for unconstrained and constrained optimization, J. Optim. Theory Appl. 132 (2) (2007) 245–265.
[54] A. Uschmajew, Local convergence of the alternating least squares algorithm for canonical tensor approximation, SIAM J. Matrix Anal. Appl. 33 (2) (2012) 639–652.
[55] J.C. Bezdek, R.J. Hathaway, Convergence of alternating optimization, Neural Parallel Sci. Comput. 11 (2003) 351–368.
[56] N. Hurl, W. Layton, Y. Li, C. Trenchea, Stability analysis of the Crank–Nicolson-Leapfrog method with the Robert–Asselin–Williams time filter, BIT Numer. Math. 54 (4) (2014) 1009–1021.
[57] M. Kwizak, A.J. Robert, A semi-implicit scheme for grid point atmospheric models of the primitive equations, Mon. Weather Rev. 99 (1) (1971) 32–36.
[58] P.D. Williams, A proposed modification to the Robert–Asselin time filter, Mon. Weather Rev. 137 (8) (2009) 2538–2546.
[59] P.D. Williams, The RAW filter: an improvement to the Robert–Asselin filter in semi-implicit integrations, Mon. Weather Rev. 139 (6) (2011) 1996–2007.
[60] J.S. Hesthaven, S. Gottlieb, D. Gottlieb, Spectral Methods for Time-Dependent Problems, Vol. 21, Cambridge University Press, London, 2007.
[61] J.M. Ortega, W.C. Rheinboldt, Iterative Solution of Nonlinear Equations in Several Variables, Academic Press, New York, 1970.
[62] L.N. Trefethen, Spectral Methods in MATLAB, Vol. 10, SIAM, Philadelphia, 2000.
[63] P. Comon, X. Luciani, A.L.F. de Almeida, Tensor decompositions, alternating least squares and other tales, J. Chemom. 23 (2009) 393–405.
[64] C.C. Paige, M.A. Saunders, LSQR: an algorithm for sparse linear equations and sparse least squares, ACM Trans. Math. Softw. 8 (1) (1982) 43–71.
[65] H. Huang, J.M. Dennis, L. Wang, P. Chen, A scalable parallel LSQR algorithm for solving large-scale linear system for tomographic problems: a case study in seismic tomography, Proc. Comput. Sci. 18 (2013) 581–590.
[66] J.C. Maxwell, On stresses in rarefied gases arising from inequalities of temperature, Proc. R. Soc. Lond. 27 (185-189) (1878) 304–308.
[67] C. Lanczos, Applied Analysis, Prentice-Hall, 1956.
[68] J.S. Hesthaven, Spectral penalty methods, Appl. Numer. Math. 33 (1–4) (2000) 23–41.
[69] P. Shuleshko, A new method of solving boundary-value problems of mathematical physics, Aust. J. Appl. Sci. 10 (1959) 1–7.
[70] L.J. Snyder, T.W. Spriggs, W.E. Stewart, Solution of the equations of change by Galerkin's method, AIChE J. 10 (4) (1964) 535–540.
[71] B. Zinn, E. Powell, Application of the Galerkin method in the solution of combustion-instability problems, in: XIXth International Astronautical Congress, vol. 3, 1968, pp. 59–73.
[72] D. Gottlieb, S.A. Orszag, Numerical Analysis of Spectral Methods: Theory and Applications, Vol. 26, SIAM, Philadelphia, 1977.