

Machine-learning-enhanced collision operator for the lattice Boltzmann method based on invariant networks

Mario Christopher Bedrunka ^{*}

*Chair of Fluid Mechanics, University of Siegen, Paul-Bonatz-Straße 9-11, 57076 Siegen-Weidenau, Germany
and Institute of Technology, Resource and Energy-efficient Engineering (TREE), Bonn-Rhein-Sieg University of Applied Sciences,
Grantham-Allee 20, 53757 Sankt Augustin, Germany*

Tobias Horstmann 

Department of Engine Acoustics, German Aerospace Center, Bismarckstraße 101, 10625 Berlin, Germany

Ben Picard 

*Institute of Technology, Resource and Energy-efficient Engineering (TREE),
Bonn-Rhein-Sieg University of Applied Sciences, Grantham-Allee 20, 53757 Sankt Augustin, Germany*

Dirk Reith 

*Institute of Technology, Resource and Energy-efficient Engineering (TREE),
Bonn-Rhein-Sieg University of Applied Sciences, Grantham-Allee 20, 53757 Sankt Augustin, Germany
and Fraunhofer Institute for Algorithms and Scientific Computing (SCAI), Schloss Birlinghoven, 53754 Sankt Augustin, Germany*

Holger Foysi 

Chair of Fluid Mechanics, University of Siegen, Paul-Bonatz-Straße 9-11, 57076 Siegen-Weidenau, Germany



(Received 10 December 2024; revised 3 September 2025; accepted 18 September 2025; published 12 November 2025)

Integrating machine learning (ML) techniques in established numerical solvers represents a modern approach to enhance computational fluid dynamics simulations. Within the lattice Boltzmann method (LBM), the collision operator serves as an ideal entry point to incorporate machine learning (ML) techniques to enhance its accuracy and stability. In this work, an invariant neural network is constructed, acting on an equivariant collision operator, optimizing the relaxation rates of nonphysical moments. This optimization enhances robustness to symmetry transformations and ensures consistent behavior across geometric operations. The proposed neural collision operator (NCO) is trained using forced isotropic turbulence simulations driven by spectral forcing, ensuring stable turbulence statistics. The desired performance is achieved by minimizing the energy spectrum discrepancy between direct numerical simulations and underresolved simulations over a specified wave number range. The loss function is further extended to tailor numerical dissipation at high wave numbers, ensuring robustness without compromising accuracy at low and intermediate wave numbers. The NCO's performance is demonstrated using three-dimensional Taylor-Green vortex (TGV) flows, where it accurately predicts the dynamics even in highly underresolved simulations. Compared to other LBM models, such as the BGK and KBC operators, the NCO exhibits improved accuracy while maintaining stability. In addition, the operator shows robust performance in alternative configurations, including turbulent three-dimensional cylinder flow. Finally, an alternative training procedure using time-dependent quantities is introduced. It is based on a reduced TGV model along with newly proposed symmetry boundary conditions. The reduction in memory consumption enables training at significantly higher Reynolds numbers, successfully leading to stable yet accurate simulations.

DOI: [10.1103/rbf2-p8tf](https://doi.org/10.1103/rbf2-p8tf)

I. INTRODUCTION

Turbulent flows are ubiquitous and play a key role in various physical phenomena, ranging from atmospheric dynamics and ocean currents to aeroacoustics and industrial applications [1–5]. Hence, understanding and accurately predicting the behavior of turbulent flows is paramount for numerous scientific and engineering fields. However, because of their inherently complex nature, turbulent flows pose significant challenges. Direct numerical simulations (DNS) require immense computational resources and are therefore not

^{*}Contact author: bedrunka.research@gmail.com

suitable for most realistic flows of interest in engineering [6]. Traditionally, scientists bypass complexity by making use of Reynolds-averaged Navier-Stokes modeling (RANS) or large-eddy simulations (LES), among others. An alternative way to solve the problem is to utilize a discrete version of the Boltzmann equation on a regular grid with a simplified collision operator, referred to as Lattice Boltzmann method (LBM) [7]. Contrary to the methods mentioned above, LBM simulates fluid behavior through collision and streaming of particle distribution functions on this discrete grid. Despite its strengths, including advantages regarding parallelization, locality, and simplicity, the utilization of LBM in turbulent flows still confronts the fundamental challenges of substantial computational expense and inherent physical complexity. Consequently, the stable and accurate simulation of under-resolved engineering applications remains an active area of research.

Recent studies have shown promising potential in enhancing numerical methods for simulating turbulent flows by integrating neural networks [8]. These advanced computational tools learn from data, enabling them to predict complex flow dynamics more efficiently than traditional methods. For example, neural networks (NN) can refine simulations by correcting errors in coarse-grid computations [9], offer improved models for unresolved scales in large eddy simulations [10], and even directly construct turbulence dynamics [11]. Using machine learning to identify complex patterns in the data, these approaches can potentially overcome some limitations of traditional methods. Combining machine learning with classical computational fluid dynamics (CFD) can thus provide alternative methods for analyzing turbulent phenomena that are both more efficient and potentially more accurate than traditional approaches.

In recent decades, LBM, as mentioned above, has emerged as a powerful alternative computational tool for studying fluid dynamics [12,13]. The fundamental algorithm of the lattice Boltzmann equation has been further enriched by recent studies. These include, among many others, off-lattice Boltzmann methods to separate the coupling between time and space discretization [14–16]. Other developments were designed to tackle multiscale flows, such as the discrete unified gas kinetic scheme (DUGKS) [17], or to capture interfaces in multiphase flows by combining a Lax-Wendroff propagation scheme with a specific equilibrium distribution function [18]. Several LBM methods were developed to provide stability even at higher Reynolds numbers or in under-resolved situations [19–23]. Most recently, first steps have been taken to explore the integration of neural networks in LBM methods [24–26]. These neural networks can enhance flow simulations by adding correction values to the numerical method to enhance accuracy in simulations of under-resolved flow (e.g., cylinder flow [27]) or by predicting optimal relaxation parameters to ensure predetermined behavior [24]. This is contrary to classical approaches often used for optimization (e.g. [28]). The main goal here is to achieve an optimal balance between stability and accuracy, addressing the limitations of the standard LBM method. Recently, Ortali *et al.* [29] have demonstrated a kinetic data-driven approach, training a neural network to define a novel collision operator for the LBM that acts as a subgrid-scale closure model for turbulence.

A collision operator decisively constitutes the solver's physics of the LBM. It determines the number of degrees of freedom represented by hydrodynamic moments and higher-order moments. The latter is usually propagated in such a way as to enable them to improve accuracy and stability. These models are well known as multiple relaxation time operators [20,30,31]. To examine the relaxation rates required for these models, Simons *et al.* [32] identified optimal sets by brute-force analysis to conclude stability criteria in turbulent flows. However, these sets are chosen as constant values through the simulation and do not adapt to dynamic changes, including laminar-turbulent transition. In this work, we want to take advantage of propagating higher-order moments to define a neural collision operator NCO as an adaptive model to calculate the relaxation rates depending on the local flow state.

Recently, a PyTorch-based LBM framework has been developed to easily incorporate ML algorithms into LBM [24]. The present paper makes use of this framework to combine ML with LBM to develop a novel neural collision operator NCO, allowing accurate and robust three-dimensional simulations of turbulence. Additionally, leveraging the inherent symmetries of the stencils, an invariant neural network is constructed to provide predictions regardless of orientation.

This paper is structured as follows: Sec. II describes the methodology, which comprises the lattice Boltzmann method in its fundamental form, the invariant neural network, and the coupling by introducing an NCO with equivariant properties. Section III shows the training results and the performance of the NCO. Subsequent to the establishment of the parameters and weights, the system's generalization capabilities are evaluated through the application of the NCO to diverse flow configurations, including forced isotropic turbulence, Taylor-Green vortex, and three-dimensional cylinder flow, without the necessity for retraining (refer to Sec. III for further details). Secondly, an alternative training paradigm is explored, one that is based on time-dependent dissipation data from a three-dimensional Taylor-Green vortex. This approach is used to evaluate an alternative training methodology (see Sec. IV).

II. METHODOLOGY

A. Lattice Boltzmann method

The lattice Boltzmann method (LBM) is derived from the Boltzmann equation, which is discretized to describe a particle distribution function $\mathbf{f} = (f_0, f_1, \dots, f_i, \dots, f_{q-1})$ as

$$f_i(\mathbf{x} + \mathbf{c}_i \delta_t, t + \delta_t) = f_i(\mathbf{x}, t) + \Omega_i(f_i(\mathbf{x}, t)) + S_i(f_i(\mathbf{x}, t)), \\ \times i \in \{0, 1, \dots, q-1\}. \quad (1)$$

The symbols $\mathbf{c}_i$ and δ_t define the discrete particle velocities and the time step, respectively [13]. The fundamental method is based on two operations. First, a collision operator $\Omega : \mathbb{R}^q \rightarrow \mathbb{R}^q$ models the particle interaction on each node, which leads to a local redistribution of the particle distribution function, where q is the number of discrete particle velocities, determined by the stencil $DdQq$, including the zero velocity. The symbol d describes the number of dimensions. Second, the streaming step moves the distributions along their trajectories. The most frequently used collision operator is the Bhatnagar-Gross-Krook (BGK) model, which is written for

the components as

$$\Omega_i(\mathbf{f}) = -\frac{f_i - f_i^{\text{eq}}}{\tau}. \quad (2)$$

This operator relaxes the distribution function towards its discrete quadratic equilibrium, as defined by

$$f_i^{\text{eq}}(\mathbf{x}, t) = w_i \rho \left(1 + \frac{\mathbf{u} \cdot \mathbf{c}_i}{c_s^2} + \frac{(\mathbf{u} \cdot \mathbf{c}_i)^2}{2c_s^4} - \frac{\mathbf{u} \cdot \mathbf{u}}{2c_s^2} \right), \quad (3)$$

where w_i and c_s are the lattice weights and the speed of sound, respectively. The approach to equilibrium is modulated by a single relaxation parameter τ , which controls the rate of relaxation. The density ρ and the fluid velocity $\mathbf{u}$ are obtained as

$$\rho(\mathbf{x}, t) = \sum_i f_i(\mathbf{x}, t) \quad \text{and} \quad \rho \mathbf{u}(\mathbf{x}, t) = \sum_i \mathbf{c}_i f_i(\mathbf{x}, t). \quad (4)$$

Incorporating forces into the lattice Boltzmann method can be achieved using the source term S_i in the lattice Boltzmann equation. External effects can be represented, enabling the modeling of energy injection into the system and the generation of accelerating forces that impact the flow dynamics. This study utilizes the proposed force method of Kupershtokh [33] who proposed a scheme that shifts f_i in velocity space such that:

$$S_i = f_i^{\text{eq}}(\rho, \mathbf{u} + \Delta \mathbf{u}) - f_i^{\text{eq}}(\rho, \mathbf{u}), \quad (5)$$

where $\Delta \mathbf{u}$ describes the velocity shift within the equilibrium distribution and is obtained using the force $\mathbf{F}$ as

$$\Delta \mathbf{u} = \mathbf{F} \Delta t / \rho. \quad (6)$$

B. Neural collision operator

In a previous study [24], we have shown that the collision operator stands out as a suitable solution for integrating ML into the LBM, dynamically adjusting the relaxation rates in response to the evolving characteristics of the flow. In this study, we extend the neural collision operator (NCO) $\Omega : \mathbb{R}^q \rightarrow \mathbb{R}^q$ to three dimensions and design an architecture to ensure its adaptability under equivariance constraints. First, we model Ω as follows

$$\Omega(f_i, \theta) = -\mathbf{M}^{-1} S_\theta (\mathbf{M} \mathbf{f} - \mathbf{m}^{\text{eq}}), \quad (7)$$

where $S_\theta = \text{diag}(\mathbf{1}_{\text{dim}+1}, \omega_v, \omega_n^{\text{NN}})$ is a diagonal matrix of relaxation rates. The relaxation rates for conserved moments $(\rho, \mathbf{u})$ are set to unity, the ones for the shear moments $(\omega_v = 1/\tau_v)$ are determined by the kinematic viscosity as $v = c_s^2(\tau_v - \frac{\Delta t}{2})$, and the higher-order ones $(\omega_n^{\text{NN}} = 1/\tau_n^{\text{NN}})$ are obtained by a neural network Φ . For the D3Q27 stencil, these higher-order moments are grouped into four order groups n , each with a determined relaxation rate. The neural network is a function of the transformed moments $\mathbf{m} = \mathbf{M} \mathbf{f}$ with components m_i or $\mathbf{m}^{\text{eq}} = \mathbf{M} \mathbf{f}^{\text{eq}}$ and is trained with respect to local features that depend on the current distribution function component f_i . In this study, the transformation matrix $\mathbf{M}$ is constructed based on Hermite polynomials. Specifically, the n -th order Hermite polynomial, denoted as $\mathcal{H}^{(n)}$, is defined as

$$\mathcal{H}^{(n)}(\xi_i) = \frac{(-1)^n}{\omega(\xi_i)} \nabla_{\xi_i}^n \omega(\xi_i), \quad (8)$$

where $\omega(\xi_i)$ represents the normalized weight function given by

$$\omega(\xi_i) = (2\pi)^{-D/2} \exp\left(-\frac{\xi_i^2}{2}\right). \quad (9)$$

Additionally, the relationship between the abscissas ξ_i and the velocity sets $\mathbf{c}_i$ is given by

$$\mathbf{c}_i = \xi_i / \sqrt{3}. \quad (10)$$

The multivariate nabla operator $\nabla^{(n)}$ provides a compact way to express n -th order spatial derivatives. For clarity, it is defined as

$$\nabla^{(n)} = \nabla_{\alpha_1 \dots \alpha_n}^{(n)} = \frac{\partial}{\partial x_{\alpha_1}} \dots \frac{\partial}{\partial x_{\alpha_n}}. \quad (11)$$

In this study, since we are working in three spatial dimensions ($D = 3$), the indices $\alpha_1 \dots \alpha_n$ take values from $\{x, y, z\}$. In order to construct orthogonal vectors $\mathbf{m}$, the transformation matrix $\mathbf{M}$ is defined as

$$\begin{aligned} \mathbf{M} = & (\mathcal{H}_i^{(0)}, \mathcal{H}_{i,x}^{(1)}, \mathcal{H}_{i,y}^{(1)}, \mathcal{H}_{i,z}^{(1)}, \mathcal{H}_{i,xx}^{(2)}, \mathcal{H}_{i,xy}^{(2)}, \mathcal{H}_{i,xz}^{(2)}, \mathcal{H}_{i,yy}^{(2)}, \\ & \times \mathcal{H}_{i,yz}^{(2)}, \mathcal{H}_{i,zz}^{(2)}, \mathcal{H}_{i,xyy}^{(3)}, \mathcal{H}_{i,xxz}^{(3)}, \mathcal{H}_{i,xyy}^{(3)}, \mathcal{H}_{i,xyy}^{(3)}, \mathcal{H}_{i,xxz}^{(3)}, \\ & \times \mathcal{H}_{i,yyz}^{(3)}, \mathcal{H}_{i,yzz}^{(3)}, \mathcal{H}_{i,xyy}^{(4)}, \mathcal{H}_{i,xyy}^{(4)}, \mathcal{H}_{i,xxz}^{(4)}, \mathcal{H}_{i,xyy}^{(4)}, \\ & \times \mathcal{H}_{i,xyy}^{(4)}, \mathcal{H}_{i,xyy}^{(4)}, \mathcal{H}_{i,xyy}^{(5)}, \mathcal{H}_{i,xyy}^{(5)}, \mathcal{H}_{i,xyy}^{(5)}, \mathcal{H}_{i,xyy}^{(5)}, \\ & \times \mathcal{H}_{i,xyy}^{(6)})^T. \end{aligned} \quad (12)$$

T. S. Cohen and M. Welling [34,35] introduced group equivariant convolutional networks to be more robust to symmetry operations and to reduce the required amount of training. The discrete lattice velocities $(e_i, i \in \{0, 1, \dots, q-1\})$ are invariant with respect to a symmetry group $\mathcal{G}$ described by reflections and discrete rotations of $\pi/2$ radians in all directions. These operations span the discrete dihedral group $\mathcal{G} = D_4$ of order 8 and the full octahedral group $\mathcal{G} = O_h$ of order 48 in 3D. Each group operation $g \in \mathcal{G}$ corresponds to a permutation $\Pi[g]$ of discrete lattice velocities locally and thus to a permutation of distribution functions $f_i = (f_0, \dots, f_{q-1})$. This permutation can be expressed using permutation matrices Π_g corresponding to each group operation and either acting on the distribution function vector $\mathbf{f}$ or the collision operator Ω . Exemplarily, Fig. 1 illustrates the group operations applied to the D2Q9 stencil, revealing rotation and reflection operations as well as their combinations. It would be a similarly beneficial property if the collision operator were equivariant with respect to $\mathcal{G}$,

$$\Omega \circ \Pi[g](f_i) = \Pi[g] \circ \Omega(f_i) \quad \text{for all } g \in \mathcal{G}, \quad (13)$$

meaning that the collision function Ω commutes with the group action. The special form of the MRT model used in this paper was found to be equivariant for both D2Q9 and D3Q27 under the group action. For all possible reflections and rotations, it was checked numerically that equivariance was fulfilled. This required checking that

$$\Pi_g \mathbf{M}^{-1} S_\theta \mathbf{M} = \mathbf{M}^{-1} S_\theta \mathbf{M} \Pi_g. \quad (14)$$

A series of tests was performed in order to provide numerical validation of this constraint. For each symmetry transformation in the group, two parallel simulations were compared: (1)

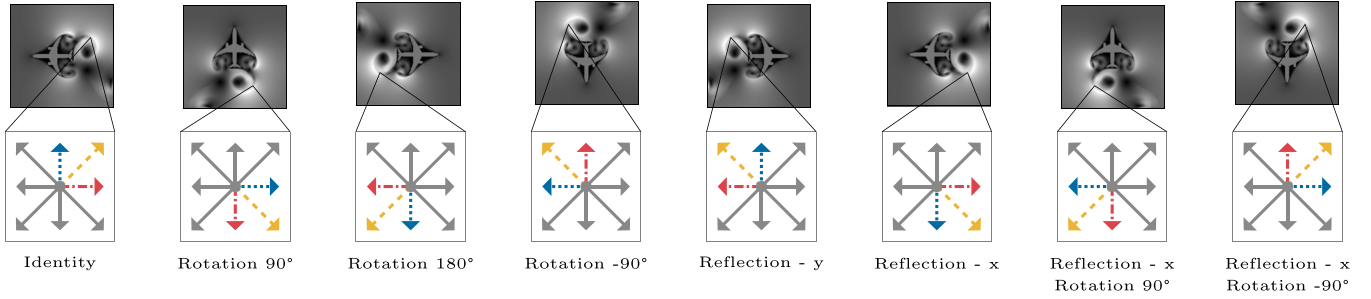


FIG. 1. A visualization of the group actions on the $D2Q9$ stencil demonstrates the results of rotations and reflections. The utilization of color serves to differentiate the moving distribution functions, thereby elucidating the manner in which each function is mapped to a new orientation by the symmetry operations. The configuration and the flow field are intended to provide an illustrative representation of these transformations.

one starting from a transformed initial state (e.g., rotation), and (2) one starting from the original state, with the transformation being applied to the final flow field. It was confirmed that the resulting flow fields from both simulations using the NCO were identical, with element-wise differences on the order of machine precision. The commutation of these matrices could be traced back to ordering the moments according to their order and simultaneously relaxing moments of equal order with the *same* constant relaxation rate. This results in groups of constant elements in the diagonal matrix S_θ . Relaxing this condition, thereby allowing for distinct relaxation rates, immediately violated the equivariance property of Ω . To further guarantee invariance of the neural network with respect to the various group actions discussed above, further steps needed to be taken.

The permutations are used to account for the changing feature inputs when applying the group operations. As stated above, the transformation matrix $\mathbf{M}$ systematically organizes the moments into groups of their corresponding order. Subsequently, the moments within each group are relaxed collectively, maintaining consistency within their respective order n . Each permutation, therefore, leads to a different ordered set of moments $\mathbf{m}^{(g)} = \mathbf{M}\Pi_g \mathbf{f}$ as inputs for a standard neural network Φ , which consists of two layers with 20 neural nodes each and 1064 parameters (see Fig. 2). The network was trained to provide the relaxation rates of the higher-order moments. An invariant neural network N_Φ can thus be defined as an average over the networks using all transformed moments $\mathbf{m}^{(g)}$ as inputs

$$\tilde{\tau}_n : N_\Phi(\mathbf{f}) = \frac{1}{|\mathcal{G}|} \sum_{g \in \mathcal{G}} \Phi(\mathbf{m}^{(g)}), \quad (15)$$

where Π_g describes the transformation operator of the group $\mathcal{G}$ and $|\mathcal{G}|$ denotes the number of elements in the group (48 for the full octahedral group in 3D). This averaging ensures that the neural network exhibits group-invariant properties, making it robust to transformations described by $\mathcal{G}$. This can be seen by simply applying a different transformation Π_i via

$$N_\Phi(\Pi_i \mathbf{f}) = \frac{1}{|\mathcal{G}|} \sum_{g \in \mathcal{G}} \Phi(\mathbf{M}\Pi_g \Pi_i \mathbf{f}). \quad (16)$$

Since the complete octahedral group $\mathcal{G}$ is closed under multiplication of transformations, the composition of

transformations is again an element of the group. Therefore, the neural network is invariant as the sum includes the same elements $\Phi(\mathbf{m}^{(g)})$, just in a different order to lead to:

$$N_\Phi(\Pi_i \mathbf{f}) = N_\Phi(\mathbf{f}). \quad (17)$$

Furthermore, we assert that the learned relaxation parameters are >0.5 to ensure over-relaxation and thus stability, by applying a sigmoid function σ at the output as

$$\tau_n = \frac{1}{2(1 + e^{-\tilde{\tau}_n})} + 0.5. \quad (18)$$

This transformation functions as a crucial safety layer for two key reasons. Firstly, the lower bound of 0.5 guarantees that the NCO's predictions are numerically stable [13,36]. Secondly, the upper bound of 1.0 confines the predictions to the well-behaved “over-relaxation” regime. The limiting case of $\tau = 1.0$ for the higher-order moments is equivalent to the collision step in a regularized LBM, where these moments are relaxed directly to their equilibrium values. Embedding this physical knowledge in the model's architecture stabilizes the training process and makes the resulting operator inherently robust.

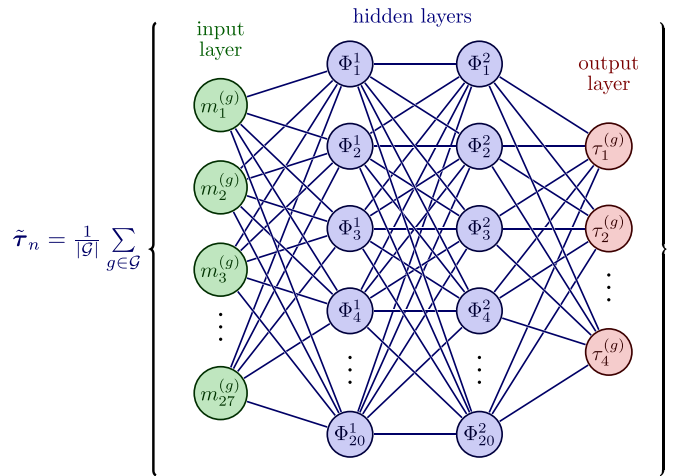


FIG. 2. Illustration of the neural network topology, including the standard network with two hidden layers and the averaging resulting in an invariant neural network, see Eq. (15).

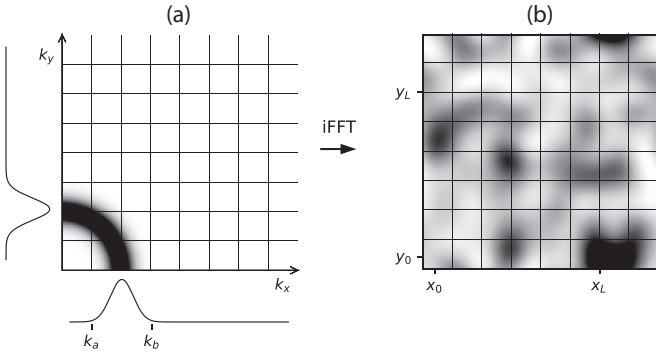


FIG. 3. 2D representation of the spectral force forcing. (a): A Gaussian distribution stimulates specific wave numbers in Fourier space. (b): The force field is then obtained by an inverse Fourier transformation.

C. Training data—forced isotropic turbulence

In this study, we employ simulations of forced isotropic turbulence as training data. This flow offers an infinitely variable range of distribution functions due to randomly generated force fields. At the same time, the deliberate injection of energy enables predictable turbulence statistics in the converged state, which serves as the basis for reference as ground truth in the loss function. Kolmogorov's hypothesis of local isotropy due to independence of the small and large scales of homogeneous turbulence is often invoked. Indeed, local triadic interactions in Fourier space support this hypothesis, leading to a decoupling of the smallest and largest eddies as energy is, in the mean, transported to the dissipative scales. Additionally, a direct coupling between small and large scales within distant triadic interactions is reported in the literature, too, see Brasseur [37], for example. Nevertheless, this additionally motivates us to use forced isotropic turbulence as training data to obtain realistic values for higher-order moments within neural networks to obtain a physically based dissipation to stabilize simulations at higher Reynolds numbers. This approach is furthermore transferable to other flows in the best-case scenario.

The effects of various forcing methods for achieving forced isotropic turbulence within the LBM were analyzed. Among these, spectral forcing was found to have distinct advantages, including the ability to generate statistically steady turbulence levels with minimal spurious correlations compared to other established stimulation methods, such as trigonometric forcing. Furthermore, spectral forcing effectively omits compressibility effects by reducing the divergence of the velocity field, which, in turn, decreases numerical errors within the LBM simulations. Following these observations, we utilize Alvelius' [38] developed spectral models that inject a power input $\mathcal{P}$ over a discrete time interval Δt by stimulating random spherical spheres as shown in Fig. 3. This method involves a force function $\hat{F}_\alpha(\mathbf{k}, \mathbf{t})$ in Fourier space that distributes the force across a range of small wave numbers facilitated by randomly generated phases. The force function reads

$$\hat{F}_\alpha(\mathbf{k}, \mathbf{t}) = A_{\text{ran}}(\mathbf{k}, \mathbf{t})e_{1\alpha}(\mathbf{k}) + B_{\text{ran}}(\mathbf{k}, \mathbf{t})e_{2\alpha}(\mathbf{k}) \quad (19)$$

with unit vectors $e_{i\alpha}$ chosen to be normal to the wave number vector $\mathbf{k}$ and orthogonal to each other. This mathematical

constraint leads to a divergence-free force field $\mathbf{k} \cdot \hat{\mathbf{F}} = 0$ and is satisfied by

$$e_{1x} = \frac{k_x k_y}{\sqrt{(k_x^2 + k_y^2)}}, \quad e_{1y} = -\frac{k_x k_y}{\sqrt{(k_x^2 + k_y^2)}}, \quad e_{1z} = 0, \quad (20)$$

and

$$e_{2x} = \frac{k_x k_z/k}{\sqrt{(k_x^2 + k_y^2)}}, \quad e_{2y} = \frac{k_y k_z/k}{\sqrt{(k_x^2 + k_y^2)}}, \quad e_{2z} = -\frac{\sqrt{(k_x^2 + k_y^2)}}{k}, \quad (21)$$

with $k = |\mathbf{k}|$ [38]. The symbols A_{ran} and B_{ran} describe random complex numbers, defined as

$$A_{\text{ran}} = \left(\frac{F(k)}{2\pi k^2} \right)^{1/2} \exp(i\theta_1) g_A(\phi), \quad (22)$$

$$B_{\text{ran}} = \left(\frac{F(k)}{2\pi k^2} \right)^{1/2} \exp(i\theta_2) g_B(\phi). \quad (23)$$

The force distribution is mainly determined by the force density function

$$F(k) = A \exp\left(-\frac{(k - k_f)^2}{c}\right), \quad (24)$$

shaped in this study by a Gaussian distribution. The symbols k_f and c determine the wave number peak of the forcing spectrum and the width of the Gaussian distribution, respectively. The parameter A describes the amplitude of the forcing depending on the power input $\mathcal{P}$ as

$$A = \frac{\mathcal{P}}{\Delta t} \int_{k_a}^{k_b} \exp\left(-\frac{(k - k_f)^2}{c}\right) dk. \quad (25)$$

In his formulation, Alvelius introduced two stochastic variables, g_A and g_B , that conform to the constraint $g_A^2(\phi) + g_B^2(\phi) = 1$, with ϕ being a uniformly distributed random number within the interval $[0, \pi)$. This random number is generated anew for each wave number and time step. The variables g_A and g_B are determined by the equations

$$g_A = \sin(2\phi) \quad \text{and} \quad g_B = \cos(2\phi), \quad (26)$$

ensuring the normalization of these stochastic factors.

The angles θ_1 and θ_2 are related through $\psi = \theta_2 - \theta_1$, where ψ is also a uniformly distributed random number, within the range $[0, 2\pi)$. The magnitude of θ_1 is then calculated using the tangent function,

$$e \tan \theta_1 = \frac{g_A(\phi)\text{Re}\{\xi_1\} + g_B(\phi)(\sin \psi \text{Im}\{\xi_2\} + \cos \psi \text{Re}\{\xi_2\})}{-g_A(\phi)\text{Im}\{\xi_1\} + g_B(\phi)(\sin \psi \text{Re}\{\xi_2\} - \cos \psi \text{Im}\{\xi_2\})}, \quad (27)$$

where each ξ_i corresponds to the projection of the velocity field along the unit vectors by $\xi_i = \hat{u}_\alpha e_{i\alpha}$, thereby integrating both the direction and magnitude of the velocity field into the force distribution process. In this study, training data is generated within a cubic domain with an edge length of 2π on a grid size of 256^3 . These simulations emulate the dynamics of a fluid characterized by a viscosity of $\nu = 1.61 \times 10^{-3}$

and subjected to an energy input rate of $\mathcal{P} = 0.1$. This specified viscosity directly sets the relaxation time, τ_v , for the physical shear moments. The simulations and training are performed using Lettuce, a PyTorch-based LBM framework co-developed by the authors [24]. Lettuce is designed to seamlessly integrate machine learning models within LBM simulations.

D. Learning task

In accordance with the simulation discussed for forced isotropic turbulence, we generate a series of highly resolved distribution function batches on a grid size of 256^3 ($f_{t=0,\dots,t_N}^{\text{DNS}}$), which are downsampled to a coarser grid to achieve a distribution function on a grid size of 32^3 . The decision to employ a coarse grid of size 32^3 was primarily driven by the available hardware memory, a crucial consideration in the context of the training procedure. The process necessitates the storage of the entire computation graph for backpropagation through hundreds of simulation time steps, a task that fully utilizes the 80GB of VRAM on the NVIDIA A100 GPU utilized for this work. This practical memory constraint is a common challenge when training neural networks and was a primary motivation for developing the alternative, memory-efficient training strategy presented in Sec. IV. The transformation from fine to coarse is achieved through acoustic scaling in the form of

$$f_{i,c} = f_i^{\text{eq}}(\rho(\mathbf{x}_{f \rightarrow c}), \mathbf{u}(\mathbf{x}_{f \rightarrow c})) + \frac{2\tau_c}{\tau_f} f_{i,f}^{\text{neq}}(\mathbf{x}_{f \rightarrow c}), \quad (28)$$

where f and c correspond to the finer and coarser underlying grid, respectively. The non-equilibrium part of the distribution function is defined through the difference between the distribution function and the equilibrium distribution function as $f_i^{\text{neq}} = f_i - f_i^{\text{eq}}$. The dissipation rates of a collision operator correlate with the shape of the resulting energy spectrum. Consequently, the goal is to preserve the true energy spectrum, as obtained from the direct numerical simulation (DNS) data. This is achieved by optimizing the loss function:

$$L(\theta) = \text{MSE}(E(\kappa, \theta), E(\kappa)) + \omega E(\kappa = \kappa_{\max}), \quad 4 \leq \kappa \leq 10. \quad (29)$$

The loss function compares the shape of the energy spectrum for the wave number range $4 \leq k \leq 10$ between the results obtained using the NCO [see Eq. (7)] and the ground truth. The initial modes are subject to fluctuations due to random spectral excitation and were therefore excluded from consideration. Another important extension of the loss function consists of adding the expression $\omega E(\kappa = \kappa_{\max})$. This term, active at the largest wave number, is further minimized to inhibit the accumulation of energy in the higher modes and to tailor the amount of dissipation added for stabilization of under-resolved flows. This minimization is controlled through the weight ω . Consequently, dissipation can be baked into the NCO by construction similar to implicit LES modeling and contrary to stabilizing a simulation by means of artificial diffusivity or direct filtering (Refs. [19,39,40]), although a similar energy spectrum might emerge.

The loss function is optimized with a stochastic gradient descent optimizer (i.e., ADAM) with a learning rate of

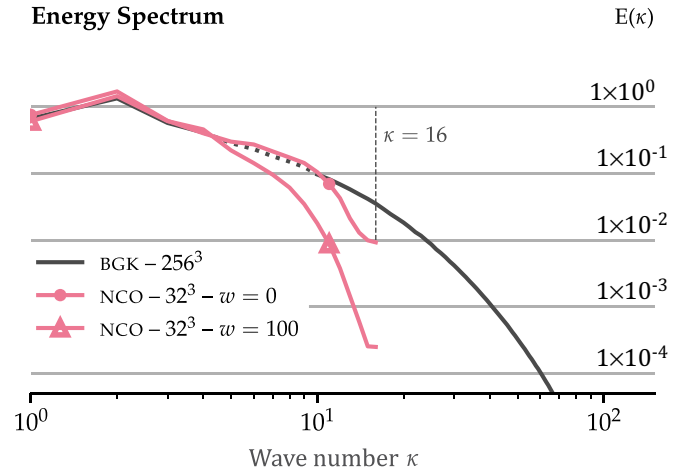


FIG. 4. Impact of the weight on the last wave number in the loss function after training: a smaller weight results in energy accumulation at higher scales, whereas a larger weight causes a sharp decline in energy at these scales. The dashed segment of the graph represents the modes with minimized discrepancy relative to the reference.

1×10^{-4} . It was observed that a higher learning rate led to an unstable training process. The training data set includes 40 batches and is processed in 90 epochs. To incrementally improve stability, the loss function is initially executed after 200 simulation steps on the coarser grid. Subsequently, the number of time steps is increased by 200 time steps every ten epochs. After 90 epochs, the simulation completes one full integral time scale after each application of the loss function.

III. TRAINING BASED ON FORCED ISOTROPIC TURBULENCE

In this chapter, we explore the calibration process to achieve the desired dissipation behavior. This investigation involves adjusting the weighting factor ω within the loss function, a parameter that significantly impacts the energy spectrum at the highest wave number. By fine-tuning this factor, we aim to control the dissipation rate in parallel to still ensure that the energy spectrum aligns with theoretical expectations and empirical observations for lower wave numbers. Then, we evaluate an appropriate parameter set across a range of flow scenarios. Specifically, the three-dimensional Taylor-Green vortex is explored, characterized by deterministic flow phases, including laminar and quasi-turbulent flows. Finally, we explore the flow dynamics around a three-dimensional cylinder, offering insights into its capabilities in simulating complex fluid dynamics. The network remains shallow in order to keep the computational costs feasible. In the first application, it consists of two hidden layers with 20 nodes, each separated by a ReLU function. However, it is straightforward to expand the network in future studies, making use of a more complex or deeper network architecture to enhance performance. Here, the emphasis was first on balancing the trade-off between computational cost and model sophistication.

A. Forced isotropic turbulence

Figure 4 illustrates the energy spectrum for two different training procedures applied to the NCO, which differ

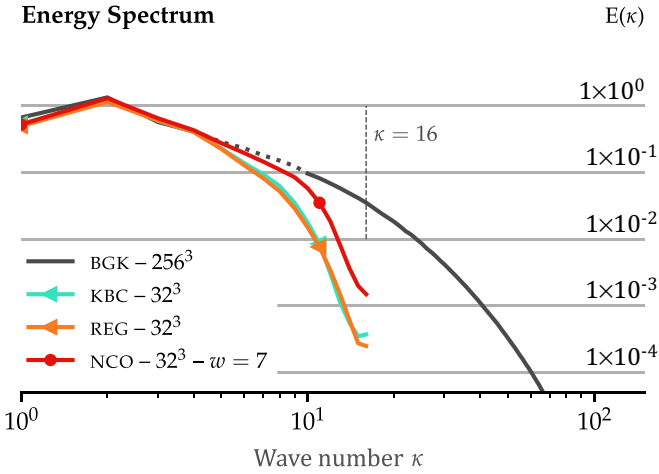


FIG. 5. Energy spectrum comparison: analyzing different collision operators relative to DNS data with a weight of $\omega = 7$ used in the training procedure of the NCO. The dashed segment of the graph represents the modes with minimized discrepancy relative to the reference.

primarily in the weighting of the last wave number in the loss function. A lower weight leads to closer agreement with the DNS energy spectrum in the wave number range. However, the low weight leads only to small amounts of numerical dissipation affecting the highest wave numbers, leading to unstable simulations in strongly under-resolved simulations. Conversely, a higher weight induces excessive energy dissipation, already affecting the spectrum at low wave numbers. Therefore, an intermediate value was chosen to strike the balance between accuracy and robustness. The value of the parameter ω was selected through a pragmatic compromise, based on the findings of a prior parameter study. This value was determined to provide sufficient dissipation to ensure stability in under-resolved cases without excessively damping the energy at resolved scales. Figure 5 shows the energy spectrum for the trained NCO using a weight of $\omega = 7$ to adjust the influence of $E(\kappa = \kappa_{\max})$ within the loss function. To assess this choice of weight on the intrinsic numerical dissipation, the energy spectrum was compared with the results obtained using the Karlin-Bösch-Chikatamarla (KBC) operator [20], the regularized operator (REG) [41] and the standard BGK model that provides the DNS data. It is obvious that, compared to other commonly used collision operators, the NCO maintains the profile of the energy spectrum more accurately before dropping off at higher wave numbers. This choice of weighting parameter leads to a smaller amount of dissipation than other LBM methods, but still enough to guarantee stable behavior, as demonstrated in the upcoming examples. Therefore, the training succeeded in preventing the accumulation of energy at higher wave numbers while at the same time more accurately capturing the energy spectrum of the reference. The late decline of the energy spectrum close to the grid cutoff resembles the situation seen when using approximate deconvolution (ADM) or direct filtering instead of explicit LES modeling [19,28,39,42–44]. The filter transfer function cutoff is shifted to wave numbers close to the grid

cutoff, thereby minimizing the affected wave numbers due to dissipation.

Another known approach is to use the numerical schemes directly to generate dissipation in implicit LES procedures (see, e.g., [45–47]). In a similar fashion, the neural collision operator is constructed to include the dissipation required to stabilize the simulations appropriately and, as such, does not necessarily require an explicit LES model.

As illustrated in Table I, a quantitative comparison of key turbulence statistics is provided, thus verifying that the NCO-driven simulation reproduces the expected behavior of statistically steady, forced isotropic turbulence on a grid size of 256^3 . The findings indicate a consensus among the primary energy balance indicators. The mean energy dissipation rate (ϵ) and the mean energy input rate ($\langle \mathbf{uF} \rangle$) of the NCO simulation are in close agreement with the reference BGK data. This demonstrates that the NCO correctly maintains the global energy equilibrium of the turbulent system, which is a crucial characteristic of statistically steady isotropic turbulence. Furthermore, both simulations maintain a mean velocity divergence ($\text{div}(\mathbf{u})$) near zero, thereby confirming the preservation of the incompressibility constraint. The velocity derivative skewness (S), a sensitive measure of the energy cascade, is found to be approximately -0.4 . This value is in excellent agreement with both the reference simulation and the expected findings for forced isotropic turbulence reported in the literature. These characteristic results confirm that the NCO successfully sustains a physically consistent, chaotic turbulent state.

Following the completion of the training on forced isotropic turbulence, the NCO's parameters are now frozen. For all subsequent applications presented in this section, including the Taylor-Green vortex and cylinder flow, this exact fixed operator is employed to rigorously test its generalization to unseen flow configurations.

B. Convergence

To further validate the precision of the NCO, we initiate a convergence study utilizing the two-dimensional Taylor-Green vortex (TGV2D). The TGV2D, a canonical problem for examining the numerical accuracy and stability of computational fluid dynamics codes, serves as an ideal benchmark due to the availability of an analytical solution. This study meticulously evaluates the convergence behavior of the NCO within the LBM by varying mesh resolutions and assessing its impact on the flow physics, comparing it to the theoretical predictions. Specifically, we quantified the deviation of the simulated velocity and pressure field from their theoretical values after 100 time steps. The analytic solution reads

$$\mathbf{u}(\mathbf{x}) = \begin{pmatrix} U_0 \cos(x) \sin(y) \\ -U_0 \sin(x) \cos(y) \end{pmatrix} \exp^{-2\nu t}, \quad (30)$$

$$p(\mathbf{x}) = \frac{\rho}{4} (\cos(2x) + \cos(2y)) \exp^{-4\nu t}. \quad (31)$$

Given the three-dimensional operational architecture of the NCO, the TGV2D is extended and repeated to the third dimension without introducing a gradient, thereby omitting shear stresses and reproducing the profile of the two-dimensional

TABLE I. Evaluation of spectrally forced, homogeneous isotropic turbulence simulations on a 256^3 grid with a viscosity of $\nu = 1.61 \times 10^{-3}$. The table displays the results of the BGK simulation (training data) and the results from the neural network-enhanced simulation.

Collision	Re_λ	ϵ	$\langle \mathbf{uF} \rangle$	S	$\text{div}(\mathbf{u})$
BGK	115 ± 4	0.098 ± 0.006	0.103 ± 0.009	-0.4297 ± 0.0472	$-7.0 \times 10^{-07} \pm 1.9 \times 10^{-04}$
NCO	118 ± 6	0.101 ± 0.007	0.103 ± 0.009	-0.4205 ± 0.0438	$-8.1 \times 10^{-06} \pm 2.5 \times 10^{-04}$

flow. The findings in Fig. 6 from this convergence study attest the anticipated reduction in error, as quantified by the L2 norm [13].

C. Three-dimensional Taylor-Green vortex

The performance of the trained NCO is evaluated simulating a three-dimensional Taylor-Green vortex, providing both a smooth initialized flow field and turbulence throughout the simulations beginning at approximately $t = 7.5$ [48] as visible in Fig. 7. The initial condition on the computational domain $S = [0, 2\pi]$ are

$$\mathbf{u}(\mathbf{x}, t = 0) = \begin{pmatrix} \sin(x) \cos(y) \cos(z) \\ -\cos(x) \sin(y) \cos(z) \\ 0 \end{pmatrix}, \quad (32)$$

$$p(\mathbf{x}, t = 0) = \frac{1}{16}(\cos(2x) + \cos(2y) \cos(2z + 2)). \quad (33)$$

To accurately capture the evolving turbulent profiles after the initialization phase, it is necessary to investigate the associated dissipation rates. As the Reynolds number increases, capturing the correct rates at low resolution becomes increasingly challenging, affecting both accuracy and stability. Figure 8 shows the dissipation rates that are well approximated by $-dk/dt$, due to homogeneity and lack of forcing, for several resolutions and Reynolds numbers. The reference data was simulated at a resolution of 512^3 grid points using the BGK operator, which is consistent with the findings of Brachet *et al.* [49] at a Reynolds number of 1600. The fundamental BGK operator calculates the predicted profiles with considerable accuracy but lacks stability when the flow field becomes turbulent. In contrast, the KBC operator [20] maintains stability but its predicted dissipation rates deviate from the reference

data with higher Reynolds numbers. One can clearly see a strong increase at $t < 8$ that provides excessive dissipation. It stands out that the neural collision operator is less dissipative, achieving good agreement with the reference dissipation rate. For $t < 8$ the dissipation rate is slightly higher than that of the reference solution but still lower than that of the KBC operator. Thereafter, the peak in the dissipation is slightly underpredicted, without generating instabilities. Therefore, one of the goals of providing more accurate but stable results when becoming strongly under-resolved (visible in the simulation case at a resolution of 32^3), could be achieved. At a resolution of 128^3 and a Reynolds number of 1600, the NCO operator remains stable and accurately captures the dissipation peak, unlike the KBC operator in this regime. The simulations are a first step in demonstrating that using a forced isotropic turbulence simulation as training data for a neural network to determine the higher-order relaxation rates is successful and exhibits transferability to other applications.

D. 3D cylinder flow

The flow around a circular cylinder positioned perpendicularly to the flow direction serves as a classical test case for the fluid-structure interaction. This scenario has been the subject of extensive experimental and simulation studies over decades, yet it remains a popular choice due to its simplicity, relevance, and applicability to more complex scenarios. Therefore, the evaluation of the three-dimensional cylinder flow investigates the practicality of using the NCO for real-world applications. The geometry is depicted in Fig. 9. The streamwise length X of the domain is twice the height Y , the crosswise length is indicated by Z , and the resolution of the lengths is defined relative to the resolution of the diameter D .

With the exception of the inlet and outlet regions, the boundaries of the simulation domain are configured to enforce periodic conditions: populations that leave the computational domain are assigned the values of the corresponding populations entering the domain on the opposite side.

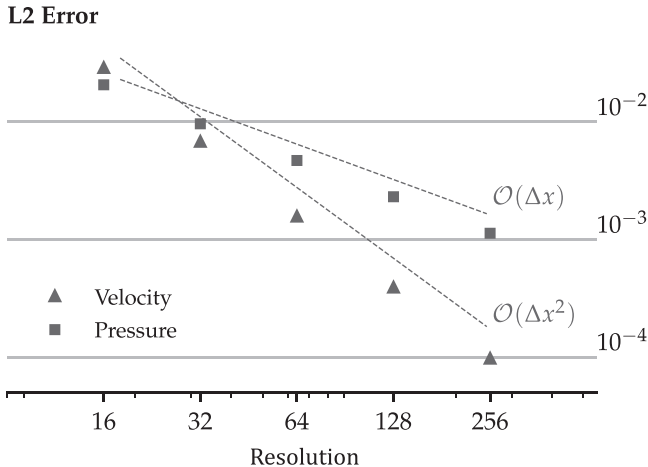


FIG. 6. Convergence behavior of the NCO within the LBM for the two-dimensional Taylor-Green vortex flow.

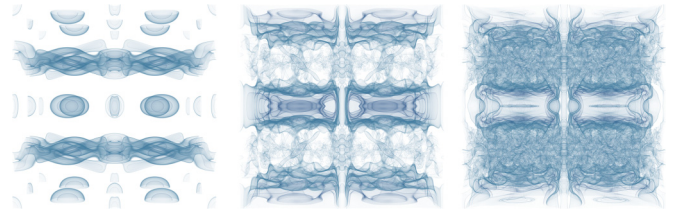


FIG. 7. Volumetric rendering of the vorticity of the 3D Taylor-Green vortex at $t = 2.5, 7.5, 12.5$, illustrating different deterministic flow phases.

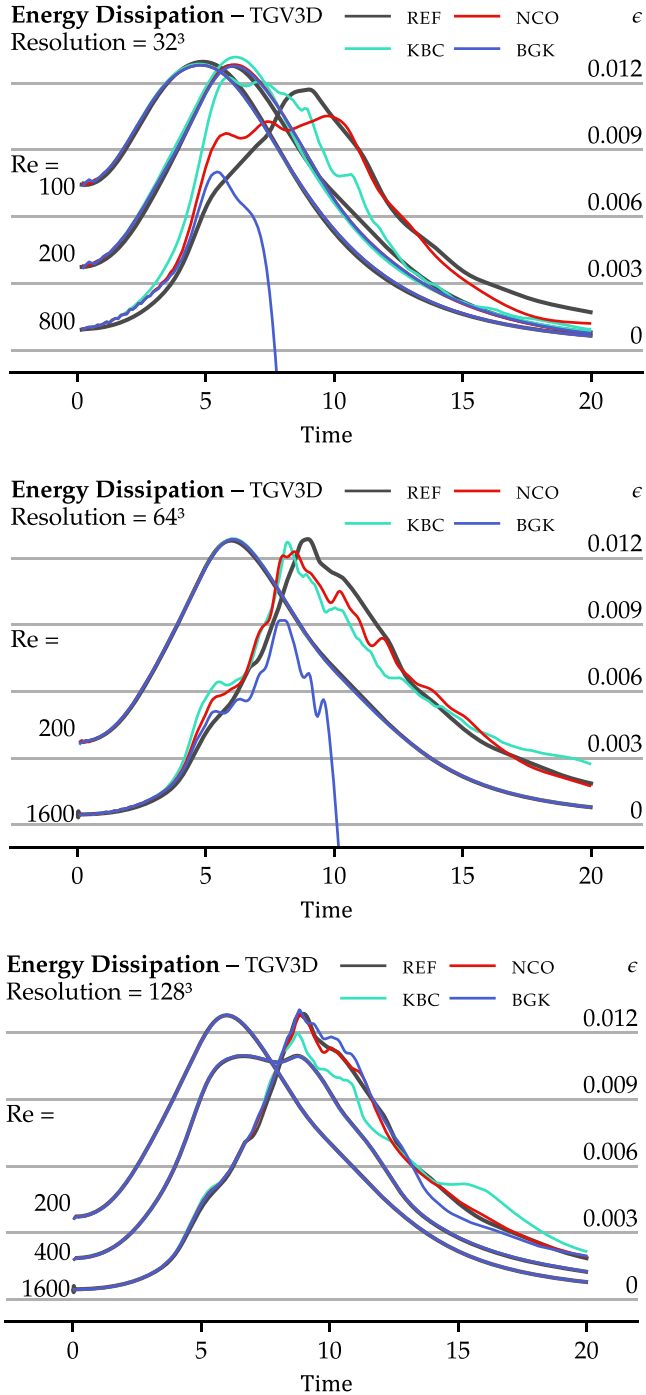


FIG. 8. Energy dissipation $-dk/dt$ of the three-dimensional Taylor-Green vortex for different resolutions (32^3 , 64^3 , 128^3) and Reynolds numbers.

Inlet. The equilibrium boundary condition stands out as the simplest form of a nonperiodic boundary condition, where the particle populations are initialized according to the equilibrium distribution function [Eq. (3)]. Although not fully capturing the complexities of physical interactions within the flow, this method is effectively utilized as an inlet condition. By prescribing the flow's velocity and density at the boundary, it allows for the simulation of inflow without the need for

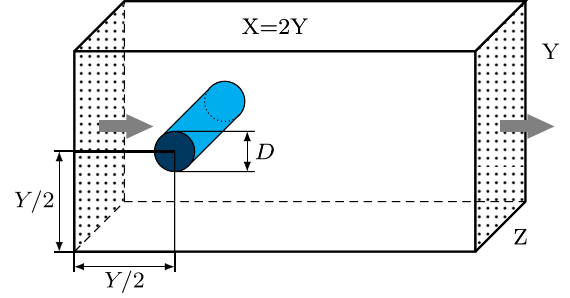


FIG. 9. Domain geometry: The stream-wise domain length X is twice the domain height Y . The cylinder of diameter D is placed at $Y/2$ from the inlet. Boundary conditions: Inlet on the left, outlet on the right, periodic boundaries for all lateral boundaries.

detailed information about the flow's behavior in the bulk of the domain.

Outlet. An advanced method for simulating open boundaries is the anti-bounce-back boundary condition. This method approximates the behavior of an open fluid volume by extending the properties of the flow into the boundary, thereby allowing for a seamless transition of the fluid properties at the interface. The fluid velocity at the boundary is calculated from the interior fluid region according to

$$f_i^*(x_b, t + \Delta t) = -f_i^*(x_b, t) + 2w_i\rho_w \left(1 + \frac{(c_i \cdot u_w)^2}{2c_s^4} - \frac{u_w^2}{2c_s^2} \right), \quad (34)$$

x_b is the fluid node interacting with the boundary, which lies half a lattice spacing away from it. The velocity u_w can be determined as:

$$u_w = u(x_b) + [u(x_b) - u(x_{b+1})]/2, \quad (35)$$

where x_{b+1} is the next fluid node following the inward normal vector of the boundary. With anti-bounce-back boundary conditions, constant pressure outlets can be constructed.

Solid. The interpolated bounce-back boundary method represents an improvement over the simple rudimentary bounce-back method by integrating the precise position of the boundary, denoted as d (i.e., the distance from the fluid node to the solid wall), into the established bounce-back scheme. This approach utilizes interpolation of bounced populations based on the boundary's relative position, employing either two (linear interpolation) or three (quadratic interpolation) adjacent fluid nodes. The expression for linear interpolation is formalized as:

$$f_i^*(x, t + \Delta t) = \begin{cases} 2df_i^*(x_b, t) + (1 - 2d)f_i^*(x_f, t) & d \leq \frac{1}{2} \\ \frac{1}{2d}f_i^*(x_b, t) + \frac{2d-1}{2d}f_i^*(x_b, t) & d > \frac{1}{2} \end{cases}, \quad (36)$$

where x_b and x_f represent the fluid node directly adjacent to the solid boundary and the second neighboring fluid node, respectively.

In the context of flows around objects such as cylinders, the determination of force coefficients (e.g., drag and lift coefficients) serves as key indicators of the flow behavior and interaction with the object. Therefore, these coefficients are

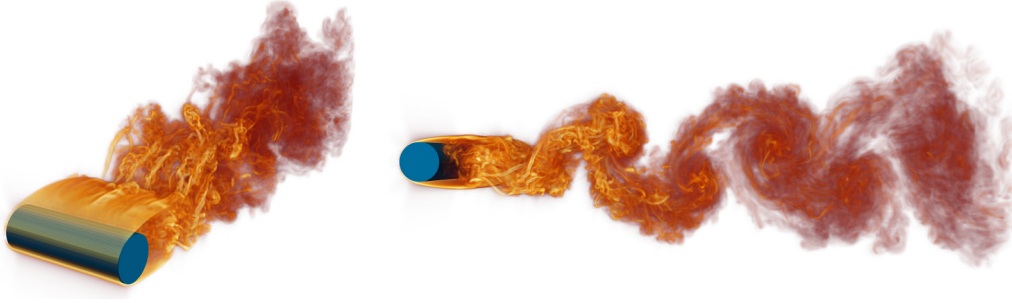


FIG. 10. Volumetric rendering of the vorticity magnitude in three-dimensional cylinder flow, illustrating the Kàrmàn vortex street phenomenon.

crucial for benchmarking simulations of the cylinder flow. The drag coefficient quantifies the resistance offered by the object against the flow direction, whereas the lift coefficient measures the force perpendicular to the flow direction.

These coefficients are derived from the drag force F_x and lift force F_y , respectively, and are calculated by

$$C_D = \frac{2F_x}{\rho U^2 A} \quad \text{and} \quad C_L = \frac{2F_y}{\rho U^2 A} \quad (37)$$

where ρ is the fluid density, U is the free-stream velocity, and A is the reference area perpendicular to the characteristic direction of the force coefficients. Within the LBM, the momentum exchange method is widely recognized for its straightforwardness and effectiveness in computing the force exerted on an object by the flow by

$$\mathbf{F} = \Delta P / \Delta t. \quad (38)$$

The momentum change ΔP is calculated by aggregating the momentum contributions from incoming f_i^{in} and reflected f_i^{out} distributions at the boundary, expressed as

$$\Delta P = \Delta x^3 \sum_{x_i^b} \Delta p(x_i^b) = \Delta x^3 \sum_{x_i^b} (f_i^{\text{in}} c_i - f_i^{\text{out}} c_i). \quad (39)$$

The Strouhal number, derived from the shedding frequency, can be determined by observing the Kàrmàn vortex street. This characteristic wake pattern is illustrated in Fig. 10, which depicts the volumetric vorticity. The simulation performed at a Reynolds number of $\text{Re} = 200$ and a Mach number of $\text{Ma} = 0.1$ involves a domain setup characterized by a grid nodes per diameter (GPD) of 30, resulting in a resolution of $x = 40 \times \text{GPD}$, $y = 20 \times \text{GPD}$, and $z = 2 \times \text{GPD}$. The study's evaluation led to drag and lift coefficients of $C_D = 1.36$ and $C_L = 0.68$, respectively, as shown in Fig. 11. The simulation demonstrates a good agreement with referenced benchmarks [50–55], highlighting the operator's effectiveness.

An increase in the Reynolds number significantly amplifies the magnitudes of turbulence. To facilitate a more detailed analysis, time-averaged velocity profiles are evaluated in the flow's converged state at a Reynolds number of 3900 illustrated in Fig. 10. The dimensions of the setup change to: $\text{GPD} = 46$, $x = 20 \times \text{GPD}$, $y = 10 \times \text{GPD}$, and $z = 3 \times \text{GPD}$.

Figure 12 presents the time-averaged velocity and Reynolds shear stress profiles along the y -axis at three different streamwise positions, comparing them to refer-

ence studies. These references are derived from experiments (i.e., Kravchenko and Moin [56]) or simulations. Common to the simulations is the use of locally refined grid domains, which enhance the accuracy in capturing peak velocity magnitudes. However, even without additional grid refinement, the NCO operator is capable of capturing the essential characteristics and maintains stability throughout the simulation.

IV. TRAINING BASED ON DISSIPATION RATES

In the preceding section, the capacity of the NCO to generalize statistical data from forced isotropic turbulence was established. The present study will shift the focus to an entirely different and independent training paradigm. In this section, an alternative strategy is investigated. This strategy involves the training of a new neural network from scratch, directly on time-dependent, global quantities. More specifically, the strategy involves the training of the neural network on the dissipation rate from a three-dimensional TGV. The purpose of this section is to examine the feasibility and performance of a different class of loss functions and training objectives.

As a building block, the three-dimensional TGV was chosen, making use of its deterministic nature to train neural networks based on key quantities that change over time. In numerical fluid dynamics, dissipation is often used for benchmarking the solver's accuracy for various Reynolds

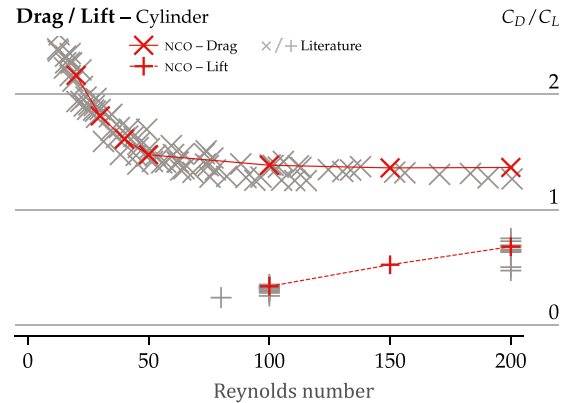


FIG. 11. Benchmark evaluation for cylinder flow. This figure compares the drag (C_D) and lift (C_L) coefficients resulting from our simulation using the trained NCO with established experimental and numerical data.

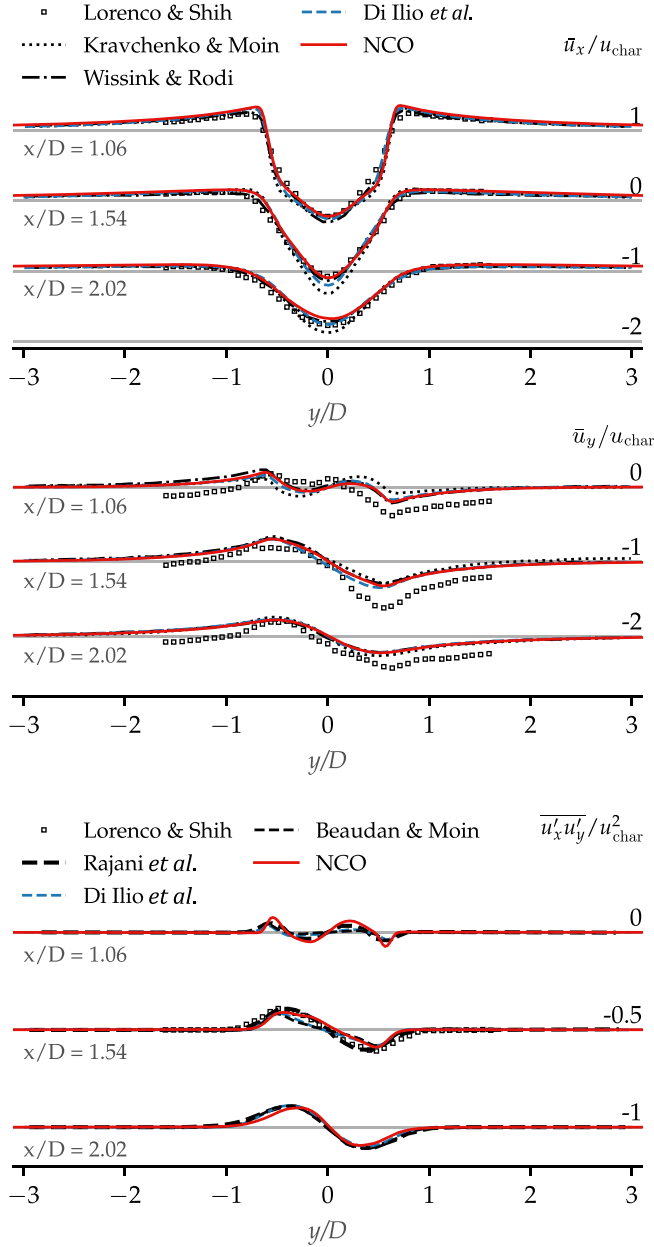


FIG. 12. Cylinder flow: Time-averaged profiles along y -axis for velocities u_x (top), u_y (middle) and Reynolds stress profiles $\overline{u'_x u'_y} / u_{\text{char}}^2$ (bottom), complemented by simulations and measurements depicted at different positions along the x -axis. u_{char} denotes the inflow velocity.

numbers, as presented above. Therefore, one idea is to use its temporal evolution itself to train for the desired collision operator behavior. Since a training step of the neural network spans multiple simulation steps and the dissipation rate is a global quantity based on local information, a high memory requirement is needed to store the gradients. At this point, one can make use of the symmetries inherent to TGV flow. Since the initial symmetry of the flow is preserved over time, this symmetry can be exploited by quartering the domain along an edge in combination with suitable symmetry boundary conditions. In three dimensions, this results in a memory reduction of a factor of 64. Assuming fixed hardware resources,

TABLE II. Training cases: Each combination of Reynolds number, Mach number, and resolution is executed within an epoch.

Reynolds number	Mach number			
	0.05	0.1	0.15	0.2
400	16^3	10^3	10^3	10^3
		16^3	16^3	16^3
		25^3	25^3	25^3
			32^3	32^3
800	16^3	10^3	10^3	10^3
		16^3	16^3	16^3
		25^3	25^3	25^3
			32^3	32^3
1600	16^3	10^3	10^3	10^3
		16^3	16^3	16^3
		25^3	25^3	25^3
			32^3	32^3

this allows for a higher grid resolution and enables the simulation of higher Reynolds numbers. The domain truncation along the symmetry planes requires specific boundary conditions to be applied at the new boundary faces - for details, see Appendix.

A. Incompressible kinetic energy

The configurations described in Table II represent the different combinations of Reynolds numbers, Mach numbers, and resolutions that are executed within each epoch during training. The learning rate used for the training is set at 1×10^{-4} . An additional hidden layer containing 20 nodes is introduced to enhance the capabilities of the neural network. The training spans a total of 100 epochs. The loss function used is the mean squared error of the difference in the dissipation rates, $-dk/dt$, between the reference results and the results simulated using the NCO. The dissipation is compared across 100 discrete points between $t = 0$ and $t = 15$. In contrast to training using the energy spectrum provided by forced isotropic turbulence, no extra effort was used here to tailor the dissipation behavior close to the grid cutoff.

Figure 13 compares the energy dissipation rates obtained using the BGK operator, the KBC operator, the REG operator, and the newly trained NCO using time-dependent dissipation data. The results show that the NCO not only accurately captures the dissipation behavior, closely aligning with the reference data throughout the time interval at a resolution of 128^3 , but also maintains stability when the resolution is decreased. In contrast, the BGK operator fails to capture turbulent behavior effectively at lower resolutions. The KBC and REG operators demonstrate better stability than BGK, particularly in under-resolved conditions. The REG model introduces excessive dissipation at earlier times at a resolution of 32^3 grid points over all Mach numbers. Even at a resolution of 64^3 , the KBC and REG operators required slightly higher dissipation at the beginning to maintain stability compared to the NCO. The KBC and NCO are similar at low Mach numbers, with the KBC model producing higher dissipation rates when the Mach number increases. All three models underes-

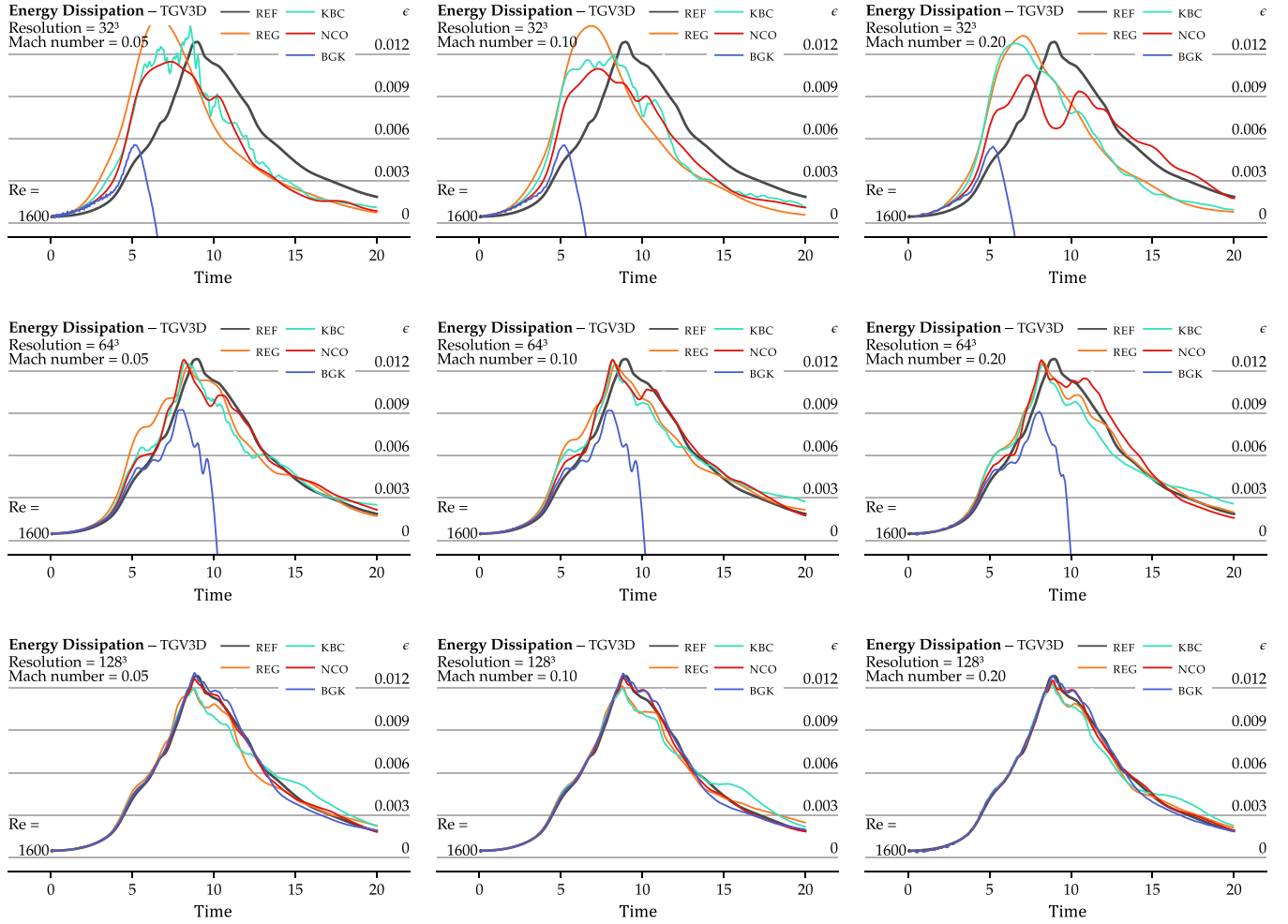


FIG. 13. Energy dissipation rate, $-dk/dt$, of the three-dimensional Taylor-Green vortex across various configurations. Rows represent different spatial resolutions, while columns correspond to varying Mach numbers.

timate the dissipation after reaching the peak value. Only the NCO is able to improve for the highest Mach number case. Increasing the resolution to 64^3 grid points improves the models with the exception of the BGK model. The regularized LBM still produces higher dissipation rates during the transition phase and underestimates it after reaching the peak value for the two lowest Mach number cases. The KBC model is similar to the NCO model for early times but deviates for larger times after reaching the dissipation peak. At the largest Mach number, now the NCO produces slightly higher dissipation levels after $t > 10$. The NCO demonstrates accuracy in capturing the dissipation behavior, aligning closely with the reference data across all Mach numbers tested at a resolution of 128^3 . The overall consistency across all Mach and resolution numbers tested here highlights the NCO's ability to produce accurate and robust results, a performance that is not achieved similarly by the KBC or regularized (REG) operators.

Overall, the section demonstrated that the reduced domain setup preserved the original dynamics of the Taylor-Green vortex initialization while enabling training at higher Reynolds numbers, which is crucial for turbulent flow simulations. The results indicate that the NCO trained on time-dependent dissipation data produce both stable and ac-

curate outcomes. This was achieved even in challenging under-resolved conditions, without attempting to further adjust the numerical dissipation. A significant benefit of employing these global observables is that it facilitates the utilization of reference data sourced directly from established literature, potentially circumventing the necessity to generate high-resolution data.

B. Shear rates

While the procedure described previously results in a stable and accurate model, this section examines an alternative training approach for the NCO, in which dissipation rates are calculated using shear rates instead of relying on kinetic energy. This method leads to a larger discrepancy between the predicted dissipation and the ground truth, particularly in under-resolved simulations. In general, a much higher resolution is required to capture the right dissipation rates by means of the shear rates. By tuning the relaxation rates based on this procedure, the neural collision operator aims to achieve greater accuracy in tackling the dissipation rates directly compared to the other common operators. As shown in Fig. 14, the shear rate-based NCO outperformed even the

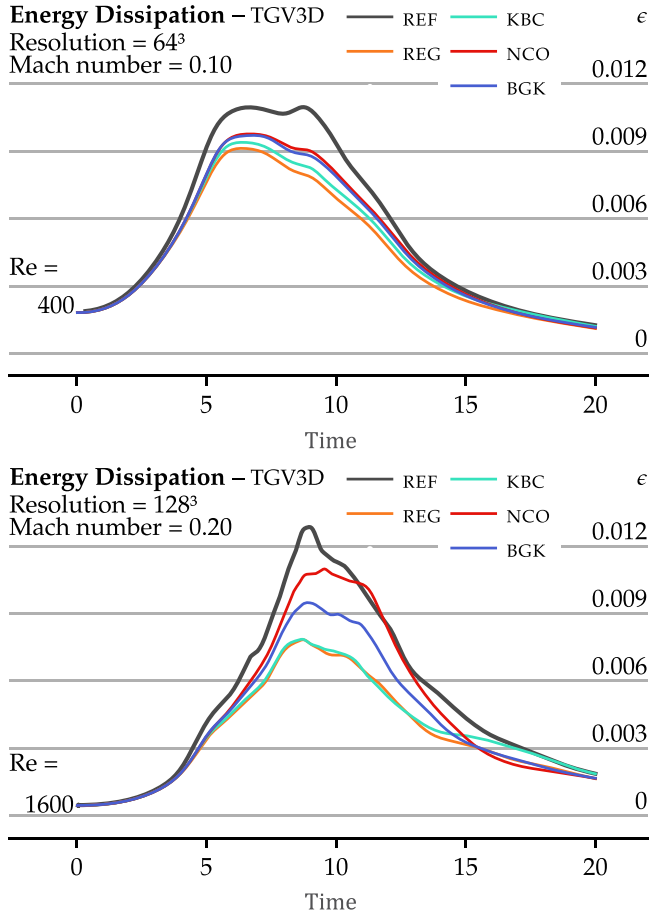


FIG. 14. Comparison of energy dissipation rates for different collision operators based on shear rates.

BGK operator in terms of capturing the correct dissipation rates. Despite its high accuracy, the collision operator trained with this shear rate approach exhibited stability issues when applied to strongly under-resolved scenarios or high Reynolds number flows due to insufficient numerical dissipation. In summary, the shear rate-based training approach resulted in an NCO that closely reproduced the dissipation behavior of the reference in turbulent flow fields. However, the increased accuracy came at the cost of reduced stability, especially in underresolved simulations. Future work could focus on addressing these stability issues, possibly by incorporating additional regularization during training or combining the shear rate approach with an adaptive stability mechanism. Compared to the more stable but less accurate training discussed, this approach demonstrates the potential to adaptively tune relaxation rates for a specific aim.

V. CONCLUSION

This paper describes the development of a neural collision operator NCO in three dimensions, based on an invariant network architecture within an equivariant collision operator. Several different ways to determine the relaxation parameters were discussed. In the first step, a simple training procedure was introduced, allowing the fine-tuning of the numerical dissipation to provide accurate and stable results. The neu-

ral network parameters were optimized using data from a forced isotropic turbulence simulation, which was stimulated by spectral forcing. The training process was designed to reproduce the energy spectrum observed in DNS by tuning the higher-order nonphysical moments over a maximum range of wave numbers. This training was performed for various resolutions, including highly under-resolved simulation data. Furthermore, robustness to guarantee stable simulations was part of the design, by adjusting the dissipation for the wave numbers close to the grid cutoff. Therefore, contrary to simulations using explicit filtering or explicit subgrid models, dissipation is implicitly enforced using the NCO, similarly to implicit LES. A tradeoff was chosen between more dissipative models like KBC and the standard but often unstable BGK collision operator. It was demonstrated that the suggested modified loss function provided the ability to adjust the spectrum shape accordingly, with dissipation acting predominantly only at the highest wave numbers. Therefore, the present NCO is the result of a well-chosen balance between accuracy and robustness.

Upon completion of the training, the NCO was assessed through various test cases. A convergence study confirmed the expected behavior of the operator. Moreover, the simulation of the three-dimensional Taylor-Green vortex not only exhibited stable performance at under-resolved conditions and at high Reynolds numbers, but it also closely aligned with reference data, demonstrating improved accuracy and stability in under-resolved conditions compared to the alternative LBM models tested. In addition, we simulated a three-dimensional cylinder flow as a more complex test case. Here, we accurately computed characteristic physical coefficients, such as drag and lift. Furthermore, the velocity profiles behind the cylinder showed good agreement with reference data, thus confirming the model's capability to capture essential physical properties in under-resolved simulations. Supplementing the procedure above, alternative ways of determining the relaxation rates were presented. These consisted of training the NCO using time-evolving dissipation rates from the Taylor-Green vortex (TGV). For this, the three-dimensional TGV constitutes an ideal flow case due to its deterministic nature and inherent symmetries. These symmetries in association with specific boundary conditions could be used to reduce the required domain size to achieve training at higher Reynolds numbers with manageable memory requirements. Contrary to basing the training on the dissipation determined using the shear rates directly, this approach proved to be very successful in accurately capturing the dissipation rates. This could be achieved using the constructed NCO without sacrificing stability, offering an alternative to train neural networks in computational fluid dynamics.

ACKNOWLEDGMENTS

We would like to express our sincere thanks to Andreas Krämer and Dominik Wilde for their meticulous inputs, which significantly enhanced the quality of our work. The simulations were performed using the Platform for Scientific Computing at Bonn-Rhein-Sieg University of Applied Sciences, which is funded by the German Ministry of Education and Research and the Ministry for Culture and Science North Rhine-Westphalia.

TABLE III. Planes: Displacement of the distribution function that match the direction on the side surfaces due to symmetry in three-dimensional space.

Displacement	Direction c_i	Transf.
$f_i(x, y^+, z) \rightarrow f_j(x, y^-, z)$	$d_{i,y} = 1$	S_y
$f_i(x, y, z^+) \rightarrow f_j(x, y, z^-)$	$d_{i,z} = 1$	S_z
$f_i(x^-, y, z) \rightarrow f_j(x^+, y, z)$	$d_{i,x} = -1$	S_x
$f_i(x^+, y, z) \rightarrow f_j(y, y^-, z)$	$d_{i,x} = 1$	$R_z(3\pi/2)$
$f_i(x, y^+, z) \rightarrow f_j(x^-, y^+, z)$	$d_{i,y} = 1$	$R_z(\pi/2)$
$f_i(x, y, z^-) \rightarrow f_j(y, x, z^+)$	$d_{i,z} = -1$	T_{xy}, S_z

DATA AVAILABILITY

Raw data were generated at the Platform for Scientific Computing at Bonn-Rhein-Sieg University of Applied Science. Derived data supporting the findings of this study are available from the corresponding author on request. The Lettuce simulation framework used in this study is openly available on GitHub at [57].

APPENDIX: REDUCED TAYLOR-GREEN VORTEX

In this Appendix, the mathematical derivation of the symmetry boundary conditions is presented. These are used to reduce the simulation domain of the Taylor-Green vortex (TGV), allowing the simulation of only a fraction of the domain by taking advantage of periodic repetitions in the flow structure. Initially, the trigonometric definition remains consistent with Eq. (32); however, it is now defined in the interval $[0 + \frac{\Delta x}{2}, \frac{\pi}{2} - \frac{\Delta x}{2}]$. The symmetry boundary conditions involve transforming the distribution function at the boundaries in each time step to accurately account for symmetric interactions. By leveraging the inherent periodic computation of the algorithm, the outward-pointing distribution vectors $d_{i,\alpha}$ at the boundaries are shifted so that they act on the correct nodes prescribed by the symmetry conditions. The computational domain is discretized so that the nodes are positioned at a distance of $\Delta x/2$ from the symmetry edges or planes, ensuring no nodes lie directly on the boundaries and allowing for a symmetric discretization of the domain. In this context,

TABLE IV. Edges: Displacement of the distribution functions due to symmetry at the edges in three-dimensional space.

Displacement	Vector direction d_i	Transf.
$f_i(x^-, y^+, z) \rightarrow f_j(x^-, y^+, z)$	$d_{i,x} = -1, d_{i,y} = 1$	I
$f_i(x^+, y^-, z) \rightarrow f_j(x^+, y^-, z)$	$d_{i,x} = 1, d_{i,y} = -1$	I
$f_i(x^-, y^-, z) \rightarrow f_j(x^+, y^+, z)$	$d_{i,x} = -1, d_{i,y} = -1$	$R_z(\pi)$
$f_i(x^+, y^+, z) \rightarrow f_j(x^-, y^-, z)$	$d_{i,x} = 1, d_{i,y} = 1$	$R_z(\pi)$
$f_i(x^-, y, z^+) \rightarrow f_j(x^+, y, z^-)$	$d_{i,x} = -1, d_{i,z} = 1$	$R_y(\pi)$
$f_i(x, y^-, z^+) \rightarrow f_j(x, y^+, z^-)$	$d_{i,y} = -1, d_{i,z} = 1$	$R_x(\pi)$
$f_i(x, y^+, z^-) \rightarrow f_j(x^-, y, z^-)$	$d_{i,y} = 1, d_{i,z} = 1$	$R_z(\pi/2), S_z$
$f_i(x^+, y, z^-) \rightarrow f_j(y, y^-, z^-)$	$d_{i,x} = 1, d_{i,z} = 1$	$R_z(3\pi/2), S_z$
$f_i(x, y^+, z^-) \rightarrow f_j(x, y^-, z^+)$	$d_{i,y} = 1, d_{i,z} = -1$	$R_x(\pi)$
$f_i(x, y^-, z^-) \rightarrow f_j(x^+, y, z^+)$	$d_{i,y} = -1, d_{i,z} = -1$	$R_z(\pi/2), S_z$
$f_i(x^-, y, z^-) \rightarrow f_j(y, y^+, z^+)$	$d_{i,x} = -1, d_{i,z} = -1$	$R_z(3\pi/2), S_z$
$f_i(x^+, y, z^-) \rightarrow f_j(x^-, y, z^+)$	$d_{i,x} = 1, d_{i,z} = -1$	$R_y(\pi)$

TABLE V. Corners: Displacement of the distribution functions at the corners in three-dimensional space.

Displacement	Vector d_i	Vector d_j
$f_i(x^-, y^-, z^-) \rightarrow f_j(x^+, y^+, z^+)$	$(-1, -1, -1)^T$	$(1, 1, 1)^T$
$f_i(x^+, y^+, z^+) \rightarrow f_j(x^-, y^-, z^-)$	$(1, 1, 1)^T$	$(-1, -1, -1)^T$
$f_i(x^-, y^-, z^+) \rightarrow f_j(x^+, y^+, z^-)$	$(-1, -1, 1)^T$	$(1, 1, -1)^T$
$f_i(x^+, y^+, z^-) \rightarrow f_j(x^-, y^-, z^+)$	$(1, 1, -1)^T$	$(-1, -1, 1)^T$
$f_i(x^-, y^+, z^-) \rightarrow f_j(x^+, y^-, z^+)$	$(-1, 1, -1)^T$	$(1, -1, 1)^T$
$f_i(x^+, y^-, z^+) \rightarrow f_j(x^-, y^+, z^-)$	$(1, -1, 1)^T$	$(-1, 1, -1)^T$
$f_i(x^+, y^-, z^-) \rightarrow f_j(x^-, y^+, z^+)$	$(1, -1, -1)^T$	$(-1, 1, 1)^T$
$f_i(x^-, y^+, z^+) \rightarrow f_j(x^-, y^+, z^+)$	$(1, -1, -1)^T$	$(-1, 1, 1)^T$

the boundary positions are characterized by x^- and x^+ , with $x^- = \Delta x/2$ and $x^+ = \pi/2 - \Delta x/2$. This convention also applies to the other spatial directions $y^\pm$ and $z^\pm$.

To implement the symmetry boundary condition, we identify pairs of distribution function vectors that are symmetric counterparts. By applying appropriate transformations, such as reflections and rotations defined by transformation matrices, we map outgoing distribution functions at the boundaries to their symmetric incoming counterparts on the opposite side of the domain. This process ensures that the distribution functions interact correctly with the boundary nodes under the symmetry conditions. To account for symmetry in three dimensions, a set of transformation matrices is employed:

(1) *Reflection matrices S_α* . Reverse the direction of the normal vector to the symmetry plane along the axis α .

$$S_x = \begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad S_y = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

$$S_z = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \quad (\text{A1})$$

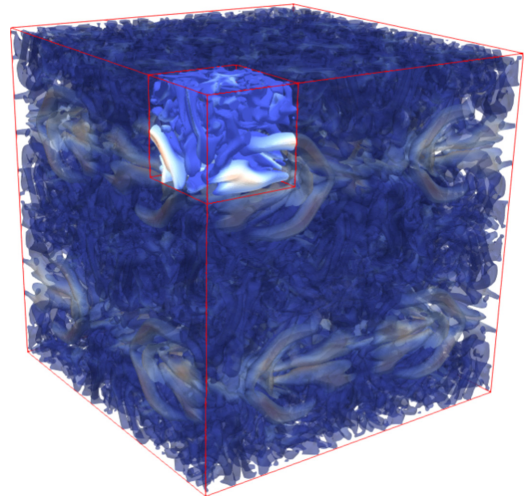


FIG. 15. Q-criterion visualization comparing the full Taylor-Green vortex with the computationally simulated subdomain. This smaller section utilizes symmetry boundary conditions to accurately represent the complete periodic flow domain, significantly reducing computational cost.

(2) *Rotation matrices* $R_b(\beta)$. Rotate vectors around the axis b by angle β .

$$R_x(\alpha) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) \\ 0 & \sin(\alpha) & \cos(\alpha) \end{pmatrix}, \quad (\text{A2})$$

$$R_y(\alpha) = \begin{pmatrix} \cos(\alpha) & 0 & \sin(\alpha) \\ 0 & 1 & 0 \\ -\sin(\alpha) & 0 & \cos(\alpha) \end{pmatrix}, \quad (\text{A3})$$

$$R_z(\alpha) = \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (\text{A4})$$

(3) *Permutation matrix* T_{xy} . Swap the x and y components of a vector

$$T_{x,y} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (\text{A5})$$

The mapping of outgoing distribution function vectors to their symmetric counterparts involves identifying the indices of vectors that point outward from the symmetry planes, edges, or corners based on their direction components d_i, α . Appropriate symmetry transformations, such as reflections and rotations are applied using the transformation matrices as defined in Tables III, IV, and V to find the corresponding inward-pointing vectors on the shifted side of the domain. This process leverages the periodicity of the computational grid to shift the outgoing vectors to the positions of their symmetric counterparts without altering the underlying periodic computation of the algorithm.

In three dimensions, special attention must be given to vectors associated with edges and corners, as they involve combinations of two or all three direction components. Vectors pointing out along edges require transformations that account for the combination of direction components (e.g., $d_{i,x} = -1, d_{i,y} = 1$). Similarly, vectors pointing out from corners involve all three components and may necessitate combined

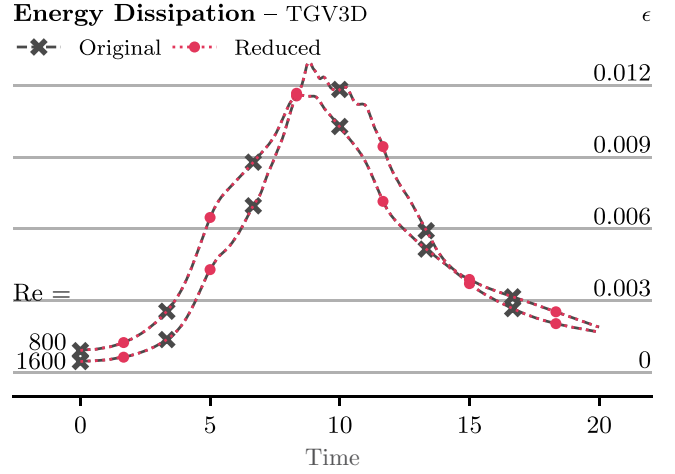


FIG. 16. Comparison of the energy dissipation of the Taylor-Green vortex, demonstrating that equivalent results can be achieved using a reduced domain with symmetry boundary conditions.

transformations using reflections and rotations to accurately map them to their symmetric counterparts. Although some stencils, such as D3Q19, may not include vectors directly pointing out from corners, it is crucial to handle them appropriately if present to ensure accurate symmetric interactions. By systematically rearranging the distribution functions that point outward from the domain in each time step, we ensure that the symmetry boundary conditions are accurately represented. A Q-criterion visualization comparing the full Taylor-Green vortex with the computationally simulated subdomain is provided in Fig. 15. Figure 16 compares the dissipation rates calculated using the BGK operator with the same discretization Δx . It can be seen that the results remain unchanged when simulating only a segment of the original domain, provided that the discussed boundary conditions are properly applied.

- [1] L. S. G. Kovasznay, Turbulence in supersonic flow, *J. Aeronautical Sci.* **20**, 657 (1953).
- [2] V. M. Canuto and J. Christensen-Dalsgaard, Turbulence in astrophysics: Stars¹, *Annu. Rev. Fluid Mech.* **30**, 167 (1998).
- [3] S. B. Pope, *Turbulent Flows* (Cambridge University Press, Cambridge, 2000).
- [4] M. Wang, J. B. Freund, and S. K. Lele, Computational prediction of flow-generated sound, *Annu. Rev. Fluid Mech.* **38**, 483 (2006).
- [5] G. Falcucci, G. Amati, P. Fanelli, V. K. Krastev, G. Polverino, M. Porfiri, and S. Succi, Extreme flow simulations reveal skeletal adaptations of deep-sea sponges, *Nature (London)* **595**, 537 (2021).
- [6] G. Coleman and R. Sandberg, A primer on direct numerical simulation of turbulence—methods, procedures and guidelines,

Technical Report No. A FM-09/01a, University of Southampton, 2010.

- [7] S. Succi, R. Benzi, and F. Higuera, The lattice Boltzmann equation: A new tool for computational fluid-dynamics, *Physica D* **47**, 219 (1991).
- [8] S. L. Brunton, B. R. Noack, and P. Koumoutsakos, Machine learning for fluid mechanics, *Annu. Rev. Fluid Mech.* **52**, 477 (2020).
- [9] J.-X. Wang, J.-L. Wu, and H. Xiao, Physics-informed machine learning approach for reconstructing Reynolds stress modeling discrepancies based on DNS data, *Phys. Rev. Fluids* **2**, 034603 (2017).
- [10] R. Maulik and O. San, Adil Rasheed, and Prakash Vedula, Subgrid for two-dimensional turbulence using neural networks, *J. Fluid Mech.* **858**, 122 (2019).

- [11] B. Kim, V. C. Azevedo, N. Thuerey, T. Kim, M. Gross, and B. Solenthaler, Deep fluids: A generative network for parameterized fluid simulations, in *Computer Graphics Forum* (Wiley Online Library, 2019), Vol. 38, pp. 59–70.
- [12] S. Chen and G. D. Doolen, Lattice boltzmann method for fluid flows, *Annu. Rev. Fluid Mech.* **30**, 329 (1998).
- [13] T. Krüger, H. Kusumaatmaja, A. Kuzmin, O. Shardt, G. Silva, and E. M. Viggen, *The Lattice Boltzmann Method* (Springer, Heidelberg, 2017).
- [14] A. Krämer, K. Küllmer, D. Reith, W. Joppich, and H. Foysi, Semi-Lagrangian off-lattice Boltzmann method for weakly compressible flows, *Phys. Rev. E* **95**, 023305 (2017).
- [15] A. Krämer, D. Wilde, K. Küllmer, D. Reith, H. Foysi, and W. Joppich, Lattice boltzmann simulations on irregular grids: Introduction of the natrium library, *Comput. Math. Appl.* **79**, 34 (2020).
- [16] D. Wilde, A. Krämer, D. Reith, and H. Foysi, High-order semi-Lagrangian kinetic scheme for compressible turbulence, *Phys. Rev. E* **104**, 025301 (2021).
- [17] Z. Guo and K. Xu, Progress of discrete unified gas-kinetic scheme for multiscale flows, *Adv. Aerodyn.* **3**, 6 (2021).
- [18] Q. Lou and Z. Guo, Interface-capturing lattice Boltzmann equation model for two-phase flows, *Phys. Rev. E* **91**, 013302 (2015).
- [19] D. Ricot, S. Marié, P. Sagaut, and C. Bailly, Lattice Boltzmann method with selective viscosity filter, *J. Comput. Phys.* **228**, 4478 (2009).
- [20] I. V. Karlin, F. Bösch, and S. S. Chikatamarla, Gibbs’ principle for the lattice-kinetic theory of fluid dynamics, *Phys. Rev. E* **90**, 031302 (2014).
- [21] M. Geier, M. Schönherr, A. Pasquali, and M. Krafczyk, The cumulant lattice boltzmann equation in three dimensions: Theory and validation, *Comput. Math. Appl.* **70**, 507 (2015).
- [22] A. Krämer, D. Wilde, K. Küllmer, D. Reith, and H. Foysi, Pseudoentropic derivation of the regularized lattice Boltzmann method, *Phys. Rev. E* **100**, 023302 (2019).
- [23] S. A. Hosseini, M. Atif, S. Ansumali, and I. V. Karlin, Entropic lattice Boltzmann methods: A review, *Comput. Fluids* **259**, 105884 (2023).
- [24] M. C. Bedrunka, D. Wilde, M. Kliemank, D. Reith, H. Foysi, and A. Krämer, Lettuce: Pytorch-based lattice Boltzmann framework, In *International Conference on High Performance Computing* (Springer, Heidelberg, 2021), pp. 40–55.
- [25] A. Corbetta, A. Gabbana, V. Gyrya, D. Livescu, J. Prins, and F. Toschi, Toward learning lattice Boltzmann collision operators, *Eur. Phys. J. E* **46**, 10 (2023).
- [26] J. T. Horstmann, M. C. Bedrunka, and H. Foysi, Lattice Boltzmann method with artificial bulk viscosity using a neural collision operator, *Comput. Fluids* **272**, 106191 (2024).
- [27] M. Ataei and H. Salehipour, XLB: A differentiable massively parallel lattice Boltzmann library in Python, *Comput. Phys. Commun.* **300**, 109187 (2024).
- [28] D. Marinc and H. Foysi, Investigation of a continuous adjoint-based optimization procedure for aeroacoustic control of plane jets, *Int. J. Heat Fluid Flow* **38**, 200 (2012).
- [29] G. Ortali, A. Gabbana, N. Demo, G. Rozza, and F. Toschi, Kinetic data-driven approach to turbulence subgrid modeling, *Phys. Rev. Res.* **7**, 013202 (2025).
- [30] P. Lallemand and L.-S. Luo, Theory of the lattice Boltzmann method: Dispersion, dissipation, isotropy, galilean invariance, and stability, *Phys. Rev. E* **61**, 6546 (2000).
- [31] P. J. Dellar, Incompressible limits of lattice Boltzmann equations using multiple relaxation times, *J. Comput. Phys.* **190**, 351 (2003).
- [32] S. Simonis, M. Haussmann, L. Kronberg, W. Dörfler, and M. J. Krause, Linear and brute force stability of orthogonal moment multiple-relaxation-time lattice boltzmann methods applied to homogeneous isotropic turbulence, *Phil. Trans. R. Soc. A* **379**, 20200405 (2021).
- [33] A. L. Kupershtokh, D. A. Medvedev and D. I. Karpov, On equations of state in a lattice Boltzmann method, *Comput. Math. Appl.* **58**, 965 (2009).
- [34] T. Cohen and M. Welling, Group equivariant convolutional networks, in *International Conference on Machine Learning* (PMLR, 2016), pp. 2990–2999.
- [35] T. Cohen, Equivariant convolutional networks, Ph.D. thesis, Taco Cohen, 2021.
- [36] F. Zhao, Optimal relaxation collisions for lattice Boltzmann methods, *Comput. Math. Appl.* **65**, 172 (2013).
- [37] J. Brasseur, Comments on the Kolmogorov hypothesis of local isotropy in the small scales, in *Proceedings of the 29th Aerospace Sciences Meeting, Reno, NV* (AIAA, Reston, VA, 1991), AIAA Paper 1991-230.
- [38] K. Alvelius, Random forcing of three-dimensional homogeneous turbulence, *Phys. Fluids* **11**, 1880 (1999).
- [39] J. Mathew, H. Foysi, and R. Friedrich, A new approach to LES based on explicit filtering, *Int. J. Heat Fluid Flow* **27**, 594 (2006).
- [40] B. Fiorina and S. K. Lele, An artificial nonlinear diffusivity method for supersonic reacting flows with shocks, *J. Comput. Phys.* **222**, 246 (2007).
- [41] J. Latt and B. Chopard, Lattice Boltzmann method with regularized pre-collision distribution functions, *Math. Comput. Simul.* **72**, 165 (2006).
- [42] S. Stolz and N. A. Adams, An approximate deconvolution procedure for large-eddy simulation, *Phys. Fluids* **11**, 1699 (1999).
- [43] J. Mathew, R. Lechner, H. Foysi, J. Sesterhenn, and R. Friedrich, An explicit filtering method for large eddy simulation of compressible flows, *Phys. Fluids* **15**, 2279 (2003).
- [44] P. Sagaut, *Large Eddy Simulation for Incompressible Flows: An Introduction* (Springer, Heidelberg, 2006).
- [45] M. Meinke, W. Schröder, E. Krause, and T. Rister, A comparison of second- and sixth-order methods for large-eddy simulations, *Comput. Fluids* **31**, 695 (2002).
- [46] F. F. Grinstein, L. G. Margolin, and W. J. Rider, *Implicit Large Eddy Simulation: Computing Turbulent Fluid Dynamics* (Cambridge University Press, Cambridge, 2007), Vol. 10.
- [47] N. A. Adams and S. Hickel, Implicit large-eddy simulation: Theory and application, in *Advances in Turbulence XII: Proceedings of the 12th EUROMECH European Turbulence Conference, September 7–10, 2009, Marburg, Germany* (Springer, Berlin, Heidelberg, 2009), pp. 743–750.
- [48] Z. J. Wang, K. Fidkowski, R. Abgrall, F. Bassi, D. Caraeni, A. Cary, H. Deconinck, R. Hartmann, K. Hillewaert, and H. T. Huynh *et al.*, High-order CFD methods: Current status and perspective, *Int. J. Numer. Methods Fluids* **72**, 811 (2013).

- [49] M. E. Brachet, D. I. Meiron, S. A. Orszag, B. G. Nickel, R. H. Morf, and U. Frisch, Small-scale structure of the Taylor–Green vortex, *J. Fluid Mech.* **130**, 411 (1983).
- [50] J.-I. Choi, R. C. Oberoi, J. R. Edwards, and J. A. Rosati, An immersed boundary method for complex incompressible flows, *J. Comput. Phys.* **224**, 757 (2007).
- [51] M. Braza, P. H. H. M Chassaing, and H. H. Minh, Numerical study and physical analysis of the pressure and velocity fields in the near wake of a circular cylinder, *J. Fluid Mech.* **165**, 79 (1986).
- [52] D. Russell and Z. J. Wang, A Cartesian grid method for modeling multiple moving objects in 2d incompressible viscous flow, *J. Comput. Phys.* **191**, 177 (2003).
- [53] C. Shu, N. Liu, and Y.-T. Chew, A novel immersed boundary velocity correction–lattice Boltzmann method and its application to simulate flow past a circular cylinder, *J. Comput. Phys.* **226**, 1607 (2007).
- [54] C. Liu, X. Zheng, and C. H. Sung, Preconditioned multigrid methods for unsteady incompressible flows, *J. Comput. Phys.* **139**, 35 (1998).
- [55] J. A. Wright and R. W. Smith, An edge-based method for the incompressible Navier–Stokes equations on polygonal meshes, *J. Comput. Phys.* **169**, 24 (2001).
- [56] A. G. Kravchenko and P. Moin, Numerical studies of flow over a circular cylinder at $Re_D = 3900$, *Phys. Fluids* **12**, 403 (2000).
- [57] <https://github.com/lettucecfd/lettuce>.