Lees-Edwards boundary conditions for translation invariant shear flow: implementation and transport properties

Sebastian Bindgen*

KU Leuven, Soft Matter, Rheology and Technology,
Celestijnenlaan 200f - box 2424, 3001 Leuven, Belgium

Florian Weik and Rudolf Weeber

University of Stuttgart, Institute for Computational Physics,
Allmandring 3, 70569 Stuttgart, Germany

Erin Koos†

KU Leuven, Soft Matter, Rheology and Technology,
Celestijnenlaan 200f - box 2424, 3001 Leuven, Belgium

Pierre de Buyl‡

Royal Meteorological Institute of Belgium,
Avenue Circulaire 3, 1180 Brussels, Belgium and
KU Leuven, Institute for Theoretical Physics,
Celestijnenlaan 200d - box 2415, 3001 Leuven, Belgium

(Dated: April 28, 2021)

Abstract

Molecular dynamics (MD) simulations represent a powerful investigation tool in the field of soft matter. By using shear flows, one can probe the bulk rheology of complex fluids, also beyond the linear response regime, in a way that imitates laboratory experiments. One solution to impose a shear flow in particle-based simulations is the Lees-Edwards technique which ensures that particles experience shear by imposing rules for motion and interactions across the boundary in the direction of the shear plane. Despite their presentation in 1972, a readily available public implementation of Lees-Edwards boundary conditions has been missing from MD simulation codes. In this article, we present our implementation of the Lees-Edwards technique and discuss the relevant technical choices. We used the ESPResSo software package for Molecular Dynamics simulation of soft matter system, which can be used as a reference for other implementers. We illustrate our implementation using bulk dissipative particle dynamics fluids, compare different viscosity measurement techniques, and observe the anomalous diffusion in our samples during continuous and oscillatory shear, in good comparison to theoretical estimates.

[*] ORCID:0000-0002-5861-0990

[†] ORCID:0000-0002-2468-2312

[‡] ORCID:0000-0002-6640-6463

I. INTRODUCTION

The response of many soft matter materials to deformations includes a viscous and an elastic component, thereby giving rise to the category of viscoelastic materials. Many experimental techniques have been developed throughout the years to investigate these unique properties, in both the linear and non-linear regime, including the combination of confocal microscope imaging and other techniques to monitor microstructural changes [1]. Despite these new developments, many experimental challenges remain [2], such as probing shorter time and length scales. Optical [3] and magnetic tweezers [4] enable the monitoring of minimal forces and displacements, but remain restricted to small observation windows.

Computer simulations represent an interesting alternative to experimental observations for soft condensed matter. In Molecular Dynamics (MD) simulations, the movement of atoms is governed by Newton's laws of motion. One can, thus, access the coordinates of the atoms and compute all relevant observables [5, 6]. The size and duration of MD simulations is mainly restricted by the available computational power. Dedicated computational methods have been developed to address larger system sizes and durations with respect to atomistic methods, such as the coarse-graining of atoms into "effective atoms" that represent atomic ensembles (e.g. water molecules or polymeric units) or fluid elements (e.g. DPD and MPCD for instance [7, 8]), or colloidal particles.

Thermostats can also be used to achieve a set temperature, as opposed to constant-energy systems, using for instance the Langevin thermostat whose properties are well known [9]. In practice, the thermostatting is achieved by adding a random force, the noise, and a dissipative force, the friction, whose magnitudes are related by the fluctuation dissipation theorem. Such simulations, using a Langevin thermostat, do not represent the fluid flows and can only mimic liquid-like behavior. More specifically, they do not take collective effects into account and do not conserve momentum. Recent developments related to the fluctuation dissipation theorem could solve some of these restrictions. This allows the linear modulus of more complex systems, including yield stress systems, to be accessible.[10] Despite these improvements, the fact that the Langevin thermostat breaks momentum conservation makes it a poor choice for pseudorheological measurements.

The dissipative particle dynamics (DPD) method was introduced for the simulation of thermostatted particle-based soft-matter systems [11–13]. DPD simulations use only pairwise forces, including for noise and friction, are not restricted to the linear regime, and provide direct access to nonequilibrium situations for which Green-Kubo methods would not apply. Due to momentum conservation, DPD can be used to study hydrodynamic phenomena. Applications of DPD include polymer solutions [14], colloidal suspensions [15], multiphase flow phenomena [16] and biological systems [17]. Recent investigations [18] have shown further ways to improve the accuracy of the DPD method and pointed out its possible use to investigate non-linear material behaviour.

To simulate the non-linear behavior using pseudorheological measurements, several solutions are available. The simplest way to introduce a deformation of the simulated volume element is to abandon the periodicity in the shear plane. Replacing these boundaries with moving walls leads to a shear flow [19]. Applications of such simulations can be found in, e.g., the migration of polymer in small gaps as present in bearings [20]. The drawback of this method is the loss in periodicity and the appearance of boundary effects. Large systems can only be simulated by increasing the size of the primary simulations box, which entails a corresponding increase of computational work load. Two interesting techniques have been developed to avoid using walls for driving the shear flow. The first is the SLLOD technique, which consists of modified equation of motions in which the flow velocity is added to the particles' motion [21–23]. When using the SLLOD equations of motion, thermostatting is done on the peculiar velocity of the particles (the laboratory reference frame minus the assigned flow velocity). In practice, this means that the flow profile is imposed via a bias in the thermostat [24–26]. As the flow velocity is different at the boundaries of the shear plane, the simulation boundaries must be adjusted by deforming the box according to the shear velocity. The SLLOD technique is available in the LAMMPS package (Large-scale Atomic/Molecular Massively Parallel Simulator [27]), for instance. The second method consists in establishing, via an applied periodic external force, a periodic flow profile [26]. The periodic flow method is convenient to implement as it does not require modification to the box geometry or to the boundary conditions. Whereas the results of SLLOD simulation can be used to simulate materials in a simple shear flow, they do not provide any feedback between the material structure and the flow profile, which means they fail with regard to more complex systems such as yield stress fluids [26]. Shear banding is one of the many effects that cannot be observed with such a technique [28]. Furthermore, it is not possible to achieve correct hydrodynamics, since the bulk fluid is only modelled implicitly. Viscous losses cannot be measured and hence the loss modulus of any material is inaccessible. A limitation of periodic flow simulations is that one cannot use them for linear shear profiles.

The most promising technique to combine the advantages of the previous mentioned approaches are Lees-Edwards boundary conditions (LEbc). Introduced in 1972 [29], they are a technique to address non-linear material behaviour during flow, and are distinguished from other non-linear simulation methods as they do not require a biased thermostat [23, 30] or non-periodic walls to initiate a shear flow, but rely on the specific rules at the boundary that lead to a translationally invariant system. LEbc are sometimes referred to as sliding brick boundary conditions [18]. Despite being developed more than four decades ago, there is presently no open-source implementation of the Lees-Edwards boundary condition. There is a clear need for such flow phenomena simulations and simulations using the Lees-Edwards boundary conditions are of broad interest in academic as well as industrial research.

In this paper, we present the principle of the Lees-Edwards method and its implementation in the ESPResSo molecular simulation package in section II. We provide the corresponding code under the same open-source license as ESPResSo. The source code availability, as well as the parameter and analysis files, are discussed in appendix A. We describe the simulation methods, including the details of the dissipative particle dynamics method, in section III. We present our results on the dynamics of fluid and colloidal particles in shear flow in section IV and conclude in section V.

II. LEES-EDWARDS BOUNDARY CONDITIONS

A. Principle of Lees Edwards boundary conditions

Lees-Edwards boundary conditions (LEbc), are a generalisation of the periodic boundary conditions for systems undergoing shear [29]. With periodic boundary conditions, a particle exiting the simulation cell is replaced at its periodic location inside the cell and the computation of distances across the boundaries uses the minimal image convention [6]. When using LEbc, a particle crossing the shear plane is also replaced in the simulation box. The position and velocity of the particle, however, are shifted so that the trajectory of the particle is compatible with its image in the adjacent moving cell. The LEbc thus allows the simulation of infinitely extended systems, as with periodic boundary conditions with a prescribed shear, using a finite simulation cell.

In the stationary regime, a constant shear flow in the shear direction, here the x-direction, is obtained and the simulation has translational invariance in the direction normal to the shear plane, i.e., the gradient (y) and vorticity (z) direction. The change in position x' as a function of time t for a particle that leaves the computational domain in the velocity gradient direction normal to the shear plane is

$$x'(t) = x(t) + x_{\mathrm{LE}} \tag{1}$$

where the Lees-Edwards offset, x_{LE} the displacement of the adjacent simulation cell with respect to the primary cell, is

$$x_{\mathrm{LE}} = v_{\mathrm{LE}} \cdot t \tag{2}$$

for steady shear, with v_{LE} as the Lees-Edwards velocity. The change in velocity is

$$v'_x(t) = v_x(t) + v_{\mathrm{LE}} \tag{3}$$

based on the drift velocity v_{LE} of the periodic images. This can be seen in Figure 1(a) where a particle leaves the primary simulation box and is re-introduced at position p'' instead of p'. The updated position is then wrapped into the primary simulation cell. When the periodic boundary conditions in the other directions remain unaltered these modifications result in a shear flow of the magnitude $\dot{\gamma} = v_{\mathrm{LE}}/h$, where h is the height of the simulation box (in y).

B. Application of Lees-Edwards boundary conditions to the velocity Verlet integrator

To implement the principle of the Lees-Edwards boundary conditions in a molecular dynamics (MD) program, it is necessary to specify practical details: the computation of distances, of relative velocities, and the combination with the velocity Verlet algorithm [31, 32]. We implemented the LEbc method in the ESPResSo package with the goal to provide a reference implementation of LEbc and a user friendly interface for steady shear and for sinusoidal shear, which is useful to determine the dynamic moduli G' and G''.

The update of a particle's coordinates in the velocity Verlet integrator occurs in the following order:

$$v^* = v(t) + \frac{1}{2m} f(t) \tag{VV 1}$$

$$x(t + \Delta t) = x(t) + \frac{1}{2} (v(t) + v^*) \times \Delta t \tag{VV 2}$$

$$\text{update all forces at time } t + \Delta \text{ , using } x(t + \Delta t)$$

$$v(t + \Delta t) = v^* + \frac{1}{2m} f(t + \Delta t) \tag{VV 3}$$

where Δt is the time step, m is the particle mass, and f is the force on the particle.

To translate the LEbc, we must apply the rules exposed above within this framework, which leads to a two step approach for the application of the velocity and position jumps which we illustrate in Figure 1(b). A one step approach for the integration is not suitable as it can lead to numerical instability. In the first step, we determine the Lees-Edwards velocity at the present simulation time t, i.e. $v_{\mathrm{LE}}(t)$ and the Lees-Edwards offset at one half time-step Δt ahead of the simulation time: $x_{\mathrm{LE}}(t + \frac{\Delta t}{2})$. After the position update in step (VV 2), we check if the particle has left the primary computational domain $0 \leq y(t + \Delta t) < h$. In that case, we apply the position jump

$$x(t + \Delta t) \rightarrow x(t + \Delta t) - x_{\mathrm{LE}} \left(t + \frac{\Delta t}{2} \right) \tag{4}$$

and apply half the velocity jump

$$v^* \rightarrow v^* - v_{\mathrm{LE}}(t + \Delta t/2) \ , \tag{5}$$

following the structure of the velocity Verlet scheme. The forces, including the DPD dissipative and random forces, are updated at the middle of the time step of the integration. It is thus necessary to have current velocities to ensure the correct thermalisation of the particles. Therefore, we tag the particle as undergoing the Lees-Edwards transformation, so that we can apply the second part of the jump after step (VV 3), using $v_{\mathrm{LE}}(t + \Delta t)$.

As simulation cells move with respect to each other due to the shear, the distance and the relative velocity between particles across the shear plane must include the Less-Edwards offset. This is shown in Figure 1(a) where the arrows crossing the boundary represent the applied distance function. While particles might experience a positive offset while crossing the box, the distance function in fact must include the negative offset to reflect these changes correctly. Accordingly, we modified the distance function in ESPResSo, so that

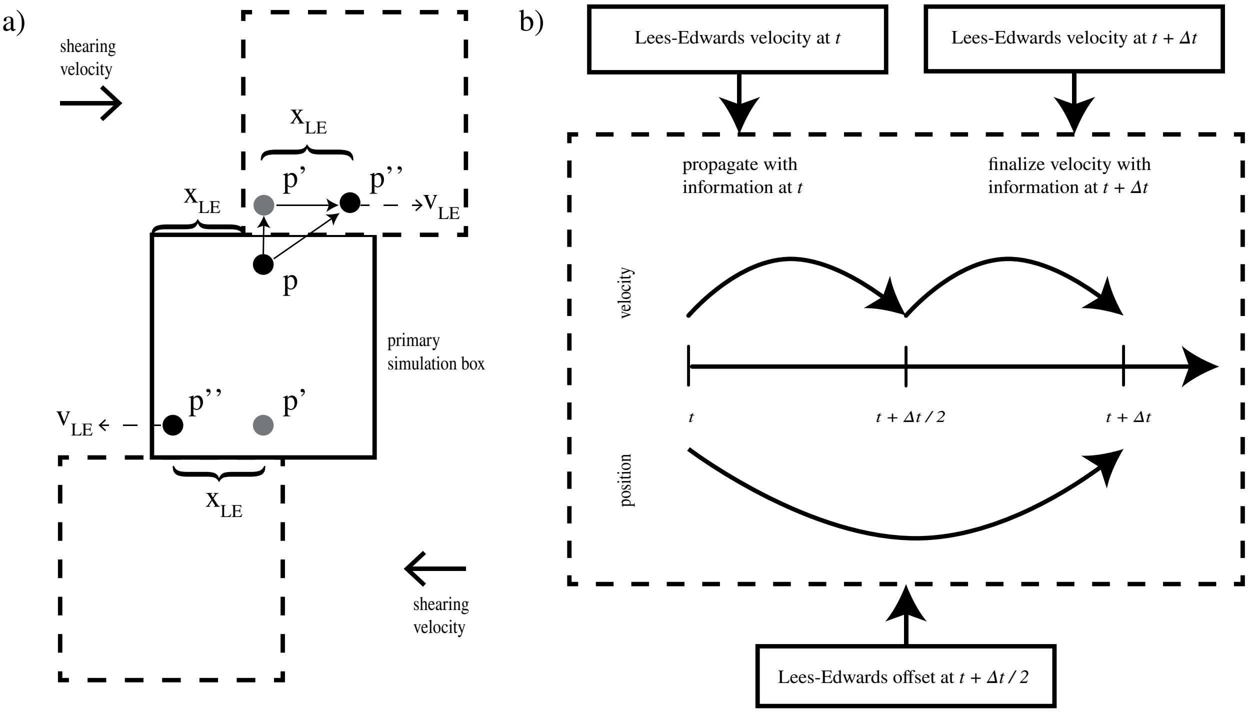
a)
shearing velocity
x_{LE}
p'
p''
v_{LE}
x_{LE}
p
primary simulation box
v_{LE}
p''
p'
x_{LE}
shearing velocity
b)
Lees-Edwards velocity at t
Lees-Edwards velocity at $t + \Delta t$
propagate with information at t
finalize velocity with information at $t + \Delta t$
velocity
t
$t + \Delta t / 2$
$t + \Delta t$
position
Lees-Edwards offset at $t + \Delta t / 2$

FIG. 1. **(a) A simplified representation of the particle movement including the Lees-Edwards boundary conditions. The particle is reintroduced at position p'' instead of p' due to the shift in the image boxes. This has to be captured in the distance function as indicated by the arrows crossing the box boundary. (b) Changes to the commonly used velocity Verlet integrator, which is displayed in the central box introduced with the inclusion of the Lees-Edwards boundary conditions. As indicated by the surrounding boxes, the Lees-Edwards velocity is updated at t and $t + \Delta t$, and the offset is included at $t + \Delta t/2$.**

the appropriate offset distance is used for all actions, such as computing the force, as well as building the neighbour lists. The modification of the relative velocities is necessary for the computation of the velocity-dependent DPD forces. Thus the velocity function is also modified in ESPResSo.

The trajectories in LEbc simulations will display discontinuous jumps whenever a particle crosses the shear plane. Since LEbc simulations include an infinitely extended system, it is possible to recontruct physically meaningful trajectories. For this purpose, we store the accumulated offset in position during the numerical integration x_{LE} of a particle and its movement in a periodic image $i(t)$ according to

$$x_{\text{part, LE}} = \sum_j x_{\text{LE}}(t_{\text{jump}}) + \sum_t \Delta t \cdot v_{\text{LE}}(t) \cdot i(t). \tag{6}$$

where j stands for occurring jumps at the LE boundary. $x_{\text{part, LE}}$ represents the displacement of the particle as it moves outside of the primary simulation cell. This data, which is necessary for the reconstruction of the full trajectory of the particle, must be computed as the simulation proceeds and cannot be obtained later using only recorded positions and velocities.

C. Modification of the cell system

In principle, the number of pairs in a system of N particles is $\mathcal{O}(N^2)$. Such a high computational cost of the force calculation is unpractical. It is, in principle, possible to compute only $\mathcal{O}(N)$ pair forces as long as only short-range forces are used, which can be cut off after a certain distance. This can be accomplished via neighbor lists and it is often practical to sort the particles into cells. This is realised in ESPResSo with the technique of domain decomposition, where the system is partitioned into cubic cells for the purpose of storing the particles' coordinates and for spatially sorting the particles[33]. The sliding nature of the boundary in shear flow simulations breaks the periodic assumption on which the domain decomposition is based and requires an appropriate modification.

To keep the computational advantage of domain decomposition, we introduce a columnar domain decomposition: we treat all cells in the layer adjacent to the boundary of the primary simulation box as neighbours, as shown using the orange-red colors in Figure 2. It does not influence the domain decomposition in the gradient and vorticity directions. A special node grid that consist of [x, y, z] = [m, n, o], i.e. $N_{\text{nodes}} = m \cdot n \cdot o$ nodes, has to be chosen. This grid must be chosen such that it has exactly one node in the shear direction, i.e. the x-direction as shear direction leads to a [1, n, o] node grid. This guarantees that no possible particle interactions are lost or considered twice due to the Lees-Edwards offset. In this way, a "re-wiring" of the cell-neighborship relations during a running simulation can be avoided. Figure 2 shows a representation of this system and illustrates the column as well as the communication directions used.

The cells located near the Lees-Edwards boundary, oriented along a column in the shear

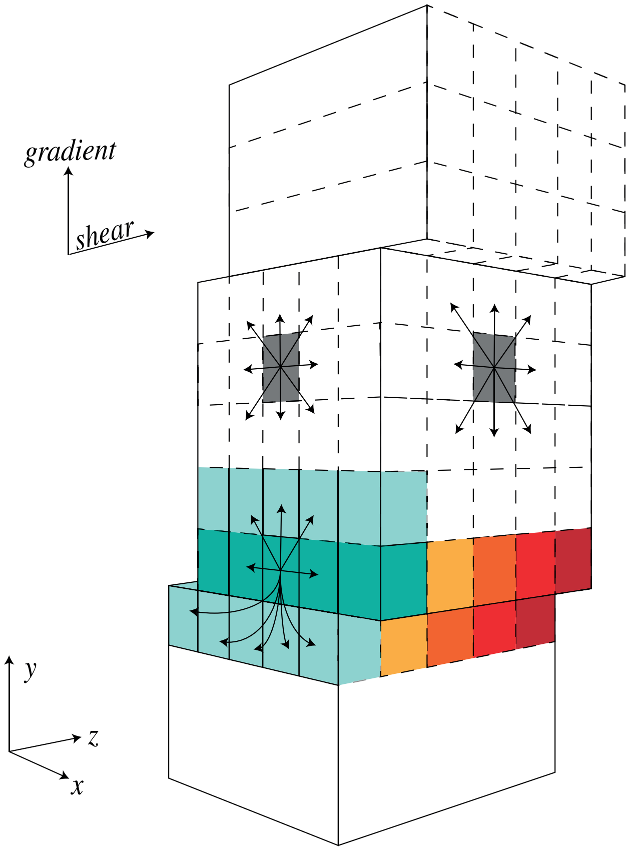

FIG. 2. **Communication pattern for the columnar cell system. The arrows shown on the x-y surface show how the local cells communicate with all other cells of the same z-column and $y \pm 1$. For local cells in the other directions the usual domain decomposition method is used which is shown here.**

direction, communicate with all other cells in the x-column as well as with the cells in the columns directly above and below ($y \pm 1$). Thus, as shown in the x-y plane wit arrows, all teal cells are considered in the neighbor list. These interactions are superimposed onto the usual domain decomposition between local cells, as shown for the grey box on the y-z plane with the limitation that all cells in a column communicate with each other. Communications in the shear directions are carried out via all cells in a column, where as for communications in the other two directions a regular domain decomposition is used. During all distance calculations the modified minimum image vector is used because it accounts for jumps across the box. The strategy chosen here minimizes the changes to the source code of the existing simulation package ESPResSo.

D. Spurious discontinuity of the velocity profile

Some articles about the Lees-Edwards method have reported discontinuous velocity profiles near the shear plane boundary. We consider here the work of Chatterjee [34]: in order to mitigate the discontinuity in the velocity profile, Chatterjee proposed to disable the thermostatting (pairwise friction and noise terms) in the vicinity of the simulation cell boundary that corresponds to the shear plane. Since the LEbc method is invariant under translation, Leimkuhler and Shang argued that the strategy of Chatterjee was only necessary in order to counteract programming errors in the simulation code [18]. In their article, Leimkuhler and Shang verify their hypothesis by introducing voluntarily the suspected bug in their own simulation code. In the presence of this bug, they are able to reproduce the suspicious profile observed by Chatterjee. In our implementation, we have not introduced any local change to the thermostat and find a continuous and linear velocity profile for the DPD particles. We present those results in Section IV B.

III. SIMULATION METHODS

We implemented the Lees-Edwards method, from scratch, within the software package ESPResSo (Extensible Simulation Package for Research on Soft Matter) [35–38]. We provide the details on the specific version in appendix A.

A. Dissipative particle dynamics

We consider a bulk fluid consisting of point particles with a mass m, a position $\mathbf{r}_\mathrm{i}$ and a velocity $\mathbf{v}_\mathrm{i}$. We use Molecular Dynamics (MD), which solves Newton's equation for all particles, subject to interaction forces:

$$m\frac{\mathrm{d}^2\mathbf{v}_\mathrm{i}}{\mathrm{d}t^2} = \mathbf{f}_\mathrm{i} \tag{7}$$

where $\mathbf{f}_\mathrm{i}$ is the total force on the i-th particle. For dissipative particle dynamics (DPD) [39], there are two modifications from this starting point: the particles are coarse-grained to represent effective fluid elements instead of atoms, and the relative velocity of particle pairs is thermostatted to introduce thermal motion and damping. The pairwise thermostatting method in DPD implies that the method conserves linear momentum and can be used for hydrodynamic simulations, which sets it apart from Langevin dynamics, another common choice in coarse-grained MD simulations.

The total DPD force $\mathbf{f}_{\mathrm{i}}^{\mathrm{DPD}}$ on particle i can be decomposed into:

$$\mathbf{f}_{\mathrm{ij}}^{\mathrm{DPD}} = \sum_{i \neq j} \left(\mathbf{f}_{\mathrm{ij}}^{\mathrm{R}} + \mathbf{f}_{\mathrm{ij}}^{\mathrm{D}} + \mathbf{f}_{\mathrm{ij}}^{\mathrm{C}} \right) , \tag{8}$$

where the random force $\mathbf{f}_{\mathrm{ij}}^{\mathrm{R}}$ is

$$\mathbf{f}_{\mathrm{ij}}^{\mathrm{R}} = \sigma \; w^{\mathrm{R}}(r_{\mathrm{ij}}) \; \theta_{\mathrm{ij}} \; \hat{\mathbf{r}}_{\mathrm{ij}} \; , \tag{9}$$

the dissipative force $\mathbf{f}_{\mathrm{ij}}^{\mathrm{D}}$ is

$$\mathbf{f}_{\mathrm{ij}}^{\mathrm{D}} = -\gamma \; w^{\mathrm{D}}(r_{ij}) \; (\hat{\mathbf{r}}_{\mathrm{ij}} \cdot \mathbf{v}_{\mathrm{ij}}) \; \hat{\mathbf{r}}_{\mathrm{ij}} \; , \tag{10}$$

and $\mathbf{f}_{\mathrm{ij}}^{\mathrm{C}}$ is a conservative force, which we define in Eq. (16). σ is the noise amplitude and $\hat{\mathbf{r}}_{\mathrm{ij}}$ is the distance vector. All the components of the DPD interaction are short-ranged, with a cutoff distance r_{cut}. The factors γ and θ_{ij} characterize the strength of the random and dissipative force, subject to the weight functions, w^{R} and w^{D} when $r < r_{\mathrm{cut}}$. The time averaged white noise θ_{ij} must have the following properties:

$$\langle \theta_{\mathrm{ij}} \rangle = 0 \tag{11}$$

$$\langle \theta_{\mathrm{ij}}(t) \theta_{\mathrm{kl}}(t') \rangle = (\delta_{\mathrm{ik}}\delta_{\mathrm{jl}} + \delta_{\mathrm{il}}\delta_{\mathrm{jk}})\delta(t - t') \tag{12}$$

with δ_{ij} as Kronecker's delta. We define the relation between w^{R} and w^{D} as

$$\left[w^{\mathrm{R}}(r) \right]^2 = w^{\mathrm{D}}(r) \tag{13}$$

for $r < r_{\mathrm{cut}}$ [40]. In order to satisfy the fluctuation dissipation theorem, the relation

$$\sigma^2 = 2 k_B T \gamma \; , \tag{14}$$

where k_B is Boltzmann's constant and T is the temperature, must hold [13]. This results in the noise amplitude σ. The weight function w^{R} is defined as:

$$w^{\mathrm{R}}(r) = 1 - \frac{r}{r_{\mathrm{cut}}} \tag{15}$$

In DPD simulations, it is customary to use a soft repulsive interaction with amplitude a_{ij} to account for conservative (C) forces $\mathbf{f}_{\mathrm{ij}}^{\mathrm{C}}$.

$$\mathbf{f}_{\mathrm{ij}}^{\mathrm{C}} = a_{\mathrm{ij}} \left(1 - \frac{r}{r_{\mathrm{cut}}} \right) \tag{16}$$

for $r < r_{\mathrm{cut}}$ and zero otherwise. This soft potential allows the use of larger time steps in the simulation.

B. Green-Kubo techniques

The simplest way to evaluate bulk properties in MD simulations is to use Green-Kubo relations. They allow bulk properties to be connected to macroscopic fluxes caused by thermal fluctuations. In the present study, we focus on the self-diffusion coefficient D and the shear viscosity η. Thus we evaluate the fluxes of the particle velocities and the shear stress. The characteristic equation for the self-diffusion coefficient is

$$D = \frac{1}{3} \int_0^\infty \langle \mathbf{v}_i(0)\mathbf{v}_i(t+\tau)\rangle_{|t} d\tau \tag{17}$$

where $\mathbf{v}$ represents the velocity of an individual particle i in three dimensions. The angular brackets $\langle\ \rangle_{|t}$ represent the ensemble average over all lag times present in the simulation. We use τ as the lag time. For the shear viscosity in the unsheared case, we use

$$\eta = \frac{V}{k_B T} \int_0^\infty \langle \sigma_{xy}(0)\sigma_{xy}(t+\tau)\rangle_{|t} d\tau \tag{18}$$

where $k_B T$ is the thermal energy, V is the box volume, and σ_{xy} is the off diagonal element of the instantaneous virial stress tensor, also known as Irving-Kirkwood stress tensor,

$$\sigma_{k,l} = \frac{\sum_i m_i v_i^{(k)} v_i^{(l)}}{V} + \frac{\sum_{j>i} F_{ij}^{(k)} r_{ij}^{(l)}}{V} \tag{19}$$

where k and $l \in [x, y, z]$ indicate the dimension of the coordinate.

C. Brownian motion

The migration of the fluid particles with time is evaluated using the mean-square displacement (MSD)

$$\mathrm{MSD}(\tau) = \langle (\mathbf{x}(t+\tau) - \mathbf{x}(t))^2 \rangle \tag{20}$$

where $\mathbf{x}$ describes the position of a particle in three dimensions. With the shear flow being absent, the MSD allows the diffusion coefficient to be calculated using the relation $6Dt = \mathrm{MSD}(t)$. The computation of the mean square displacement is also relevant in shearing and non-steady state regimes. Furthermore, it can be evaluated for several directions independently allowing more detailed insight. This is of particular interest for the simulations with shear flow.[41] Several predictions can be made for the diffusion of Brownian particles under shear flow. The MSD of our DPD fluid particles should follow the prediction for Brownian particles. In steady shear flow, the displacement of particles along the shear direction x is given by [42]

$$\langle (x(t) - x(0) - \dot{\gamma} z(0) t)^2 \rangle = 2Dt \left[1 + \frac{1}{3} (\dot{\gamma} t)^2 \right] \tag{21}$$

with a cubic dependence of time, indicating an enhanced diffusion due to the particles migrating through regions with different shear velocities [41]. For oscillatory shear flow, the MSD in the shearing direction x follows the relation $\langle \Delta x(t)^2 \rangle = 2D_{\mathrm{eff}} t$ [43] with

$$D_{\mathrm{eff}} = D \left[1 + \frac{\gamma_0^2}{2} \left(2 \sin^2 \Phi + 1 \right) \right] \tag{22}$$

where Φ represents the phase and γ_0 the deformation amplitude, indicating a squared dependence from the amplitude of the shear flow. No position correction is necesary if particle positions are evaluated at integers of complete periods.

D. Calculation of correlations

We rely on two different procedures to compute formulas of the form

$$\langle X(t_i) X(t_j) \rangle \tag{23}$$

found in Eqs. (17) and (20). A logarithmic correlator is available in ESPResSo for the set of built-in observables. Such a correlator samples the term $X(t_i)X(t_j)$ for fixed time differences $[0, M^m \Delta t, 2M^m \Delta t, \ldots, N \cdot M^m \Delta t]$, for consecutive values of the exponent m, taking the form of blocks having time intervals that increase by a factor M between successive blocks. Storing lag times that are M times larger implies the addition of N samples instead of M times more samples: Storing samples up to a time lag $\tau_{max} = N \cdot M^{m_{max}} \Delta t$ requires $m_{max} N$, which is $\mathcal{O}(\log \tau_{max})$ hence the name logarithmic correlator. This technique, which is useful when the number of samples would otherwise exceed the available memory, is presented in the book by Frenkel and Smit [5].

Another technique is the Fast Correlation Algorithm (FCA) that relies on Fourier transforms [44] to speed up the computation. We use the implementation provided by the Python package tidynamics [45] for autocorrelation and mean-square displacements. We refer to this method as a linear correlator as it requires data samples linearly spaced in time. To use the FCA method, we store the variables of interest to disk (the position for the mean squared displacement or selected components of the stress tensor for the viscosity).

The logarithmic correlator and the FCA only differ in their statistical sampling. The FCA method is equivalent to compute the pairwise correlation for all time intervals available and provides the same results, up to rounding errors, as computing the correlations with a naive $\mathcal{O}(N_{samples}^2)$ loop, where $N_{samples}$ is the total number of sample times.

For the computation of Eq. (22), we do not perform an average over time. The correlation is not of the form (23) and the two techniques presented above do not apply.

IV. RESULTS AND DISCUSSION

We carried out all of our simulations with 10,000 particles and densities of $\rho = [3, 4, 5, 6, 7]$. We used a strength of the repulsive parameter from $a_{ij} = 0$, i.e. no repulsive force, up to $a_{ij} = 175$. These parameters are similar to the ones used by Zohravi *et al.* [46] as this was the most complete study concerning the influence of the density ρ and the strength of the conservative interaction parameter a_{ij} on the shear viscosity, thus providing a good reference point to benchmark our method. All simulations use a time step of $\Delta t = 0.005$. The results are available in full in the analysis notebooks, see appendix A for details.

A. Self-diffusion coefficient and viscosity of DPD fluids

In this section, we use the mean square displacement, Green-Kubo techniques, and Lees Edwards boundary conditions to evaluate the equilibrium and non-equilibrium properties of the DPD fluid. First, we start by measuring the self-diffusion coefficient D via the mean square displacement and then we investigate the shear-viscosity η and its various contributions using the two other mentioned methods. Simulations using Green-Kubo were conducted at quiescent conditions whereas samples under shear used the LEbc method.

1. Self-diffusion coefficient

We show the diffusion coefficient that was obtained from the mean square displacement, equation (20), of 10,000 particles at $k_BT = 1.0$ in Figure 3. Each individual trajectory of the particle was correlated using the logarithmic correlator from ESPResSo and subsequent fitting of MSD $= 6Dt$. This results in 10,000 individual diffusion coefficients per simulation run. The results, shown in Figure 3, are obtained from an average of three individual quiescent runs. Our results show that the diffusion coefficient decreases with an increase

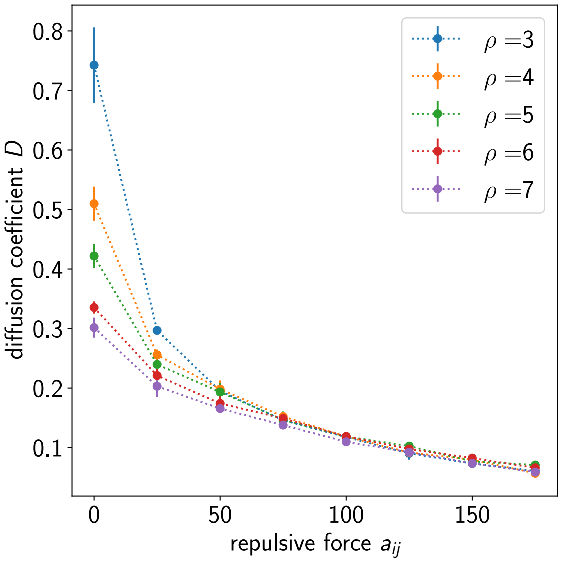

FIG. 3. **The self-diffusion coefficient of a quiescent DPD fluid in dependence of the repulsive force parameter a_{ij} for the densities $\rho = [3, 4, 5, 6, 7]$ and $k_BT = 1.0$. The lines connecting the symbols are to guide the eye.**

in repulsive strength as is expected since particles are hindered in their movement by the other surrounding particles and their repulsive interaction. By increasing either the number density of the particles or their repulsive strength, the overlap between the repulsive spheres increases, hence, the movement is hindered through "caging" particles with other DPD particles due to the higher number of possible interaction partners or the higher interaction strength. This leads to the lower effective diffusion coefficients with increasing density and increasing repulsive strength shown in Figure 3. The diffusion coefficient reduces monotonically with increasing repulsive strength for all number densities. Furthermore, the figure shows that diffusion coefficients for lower densities are higher than those with increased samples. This effect is present over the entire range of repulsive forces.

2. Viscosity

The viscosity of the DPD fluid, on the one hand, is connected to the inter-particle forces that are described in equation (8) and on the other hand to the overall momentum of the particles that results in the kinetic contribution. The contributions can be further devided up into the viscosity based on the random force $\mathbf{f}^{\mathrm{R}}$, the dissipative force $\mathbf{f}^{\mathrm{D}}$, the conservative force $\mathbf{f}^{\mathrm{C}}$. The sum of the kinetic and the conservative viscosity is referred to as the total viscosity $\mathbf{f}^{\mathrm{T}}$. We measure the viscosity using the Green-Kubo method as well as the Lees Edwards method, which measures the instantaneous stress at the "wall". That means that we include the noise in these measurements as it generates a non-negligible contribution to the integral of the autocorrelation. We measured the viscosity by two methods: First, by the Green-Kubo formula (18) and, hence, by measuring the stress fluctuations in quiescent simulations and second, by directly measuring the stress in a fluid sheared with the Lees-Edwards method which will be explained in the next part of the paper. Here, we start by discussing the results of the quiescent simulations. In order to fully describe our methods, we first show how we obtained these results. Green Kubo results are calculated with the logarithmic correlator of Espresso. We also sampled the data in a trajectory file at linear time intervals and used the `acf` method of tidynamics. The online correlator collects data up to $\tau = 100,000$. Following the initial warm up of 1,000,000 integration steps, we run the simulation for a total 500,000 time steps of $\Delta t = 0.005$. We then plot the integral of the autocorrelation function and choose a uniform cut-off for all the simulation in order to avoid any bias between the iterations. The convergence of the autocorrelation function as needed by the Green Kubo method is illustrated in Figure 4.

Depending on the contribution, we show the two possible correlation methods. For the kinetic component, a linear correlator was used (shown in green in the left plot of Figure 4). This linear correlator was necessary because the kinetic component cannot be extracted from the simulation package directly. A distinct plateau beginning at $t = 2.0$ can be found in this data. For the conservative stress, Figure 4 center, we used a linear (green) as well a logarithmic (blue) correlator, to illustrate differences between them only due to the sampling. Once more, a distinct plateau beginning at the cut-off is visible. The dissipative part of the viscosity as shown in the right plot of Figure 4 could only be evaluated using the logarithmic correlator. The dominant part of this viscosity contribution originates in the

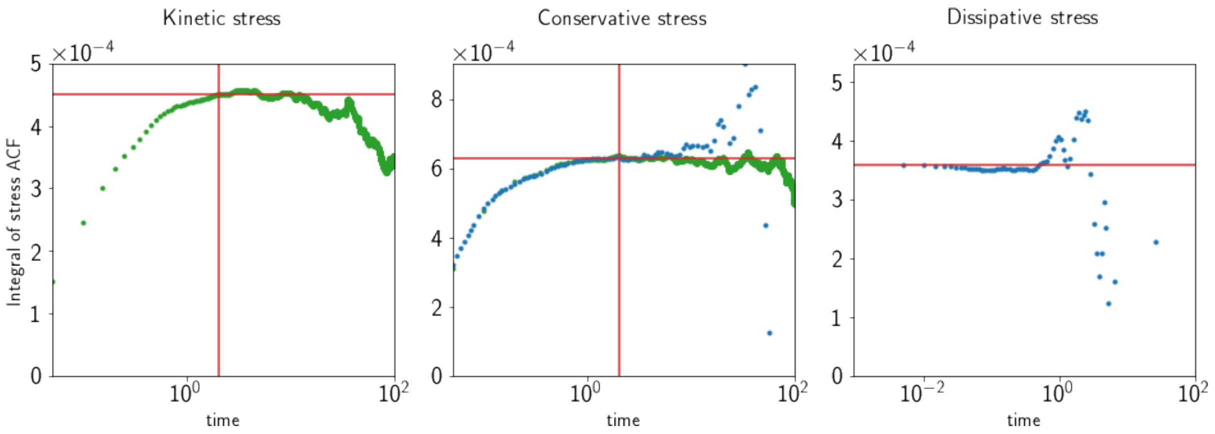

FIG. 4. **Integral of the autocorrelation (ACF) of the DPD fluid ($\rho = 6.0$ and $a_{ij} = 25$) as measured by the Green Kubo method. Data obtained using a linear correlator are shown in green and a logarithmic correlator in blue.**

delta peak around $\tau = 0$ of the autocorrelation due to the sampling of the random noise. The fluctuating contribution to the autocorrelation function are very small. We decided to cut off the integral at this very initial point where the noise starts to dominate.

The results for the Lees Edwards experiments were obtained from one simulation run per data point. The warm up of our fluid consists of 100,000 integration steps. After this warm up, we turn on the shear flow with $\dot{\gamma} = 1.0$ and obtain 200,000 stress values, 100 integration steps apart from each other (20,000,000 integration steps in total). We applied the blocking method [47] to obtain mean and standard deviation of this data and present the results in the right column of Figure 5 using the pyblock Python module [48]. For this analysis, we only used the last $2^{17} = 131,072$ measurement values in order to ensure a steady shear profile is obtained. The usage of more or fewer data points, e.g. the last 2^{16} or 170,000, did not change the results, Therefore, we are comfortable with the assumption that a stationary regime was obtained. We also chose the number of sampling points as a power of 2 because it is most efficient to apply the blocking method on such a data set.

In Figure 5, we show the collected results from all simulations. The left column shows the quiescent results from the Green Kubo method and the right column shows the shear results from the experiments using the Lees Edwards method. We also show a superimposed view of this data in the analysis notebook supplementary information to this paper. The kinetic

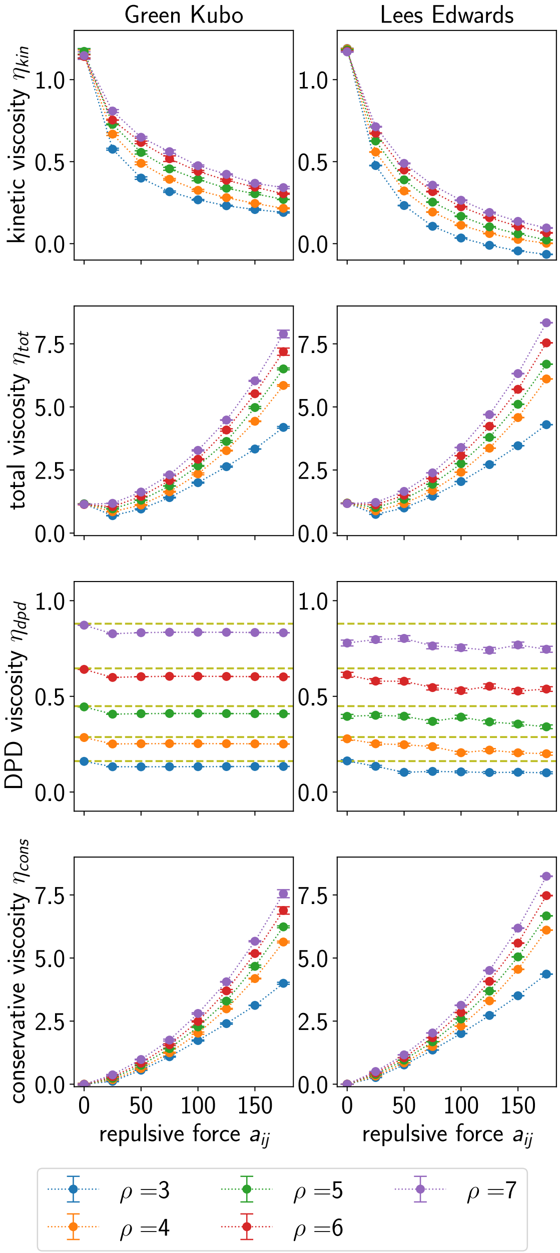

FIG. 5. **The viscosity of a DPD fluid for various repulsive force parameters a_{ij} for the densities $\rho = [3, 4, 5, 6, 7]$, at $k_bT = 1.0$. The left column shows the results of the quiescent Green Kubo analysis and the right column shows the same values as obtained with shearing Lees Edwards boundary conditions. The dashed line shown in the DPD viscosity represent the approximation found by Groot and Warren [39]. The dotted lines are shown to guide the eye.**

viscosity decreases with increasing repulsive strength for both methods. This is in agreement with the reduced diffusion constant of the particles (Figure 3). Samples with a higher density ρ have a lower kinetic viscosity, except for the case of $a_{ij} = 0.0$ (no repulsion). Here, the kinetic viscosity is the highest for all samples with a value of around $\eta_{kin} = 1.2$. The same trends can be observed for viscosity measured via the Lees Edwards technique. However, the reduction in the kinetic viscosity is more pronounced; the quiescent method always showing higher viscosities than for Lees Edwards at high repulsive forces. For example, $\eta_{kin} = 0.19$ using Green-Kubo and $\eta_{kin} = -0.07$ using Lees Edwards at $\rho = 3.0$ and $a_{ij} = 175$. This phenomenon can be explained by the additional impulse particle undergo when exposed to a shear flow. Since the total viscosity consists of the contribution from the kinetic part and from the conservative part that means it takes the contributions of the increasing repulsive strength into account. The results are consistent with the case of no repulsive force present where all samples show the same kinetic and total viscosity. The negative value for the Lees-Edwards result can be explained by the high repulsive force between the spheres that might result in a huge elasticity of the fluid. While there is a difference between the kinetic component of the viscosity between the quiescent and shearing samples, no such difference is observed in the total viscosity. Increasing the repulsive strength between the particles increases the viscosity. The similarity between the Green Kubo and Lees Edwards results is remarkable on average, there is only a maximal difference of $\frac{\Delta\eta}{\eta} = \frac{\eta_{le}-\eta_{gk}}{\eta_{gk}} = 0.07$ between the two methods.

In our experiments, we also directly measure the dissipative part of the viscosity and compare these values to the prediction by Groot and Warren [39]. Figure 5 shows that, for no repulsive force present, the data points from Green Kubo lie exactly on the predicted line whereas all points seem to be slightly shifted downwards but remain on a constant value for higher repulsive forces. For the measurements obtained by the Lees Edwards technique, a bigger separation between the theoretical prediction and the obtained values can be observed. The values for an absent repulsive force also already deviate from the predicted values. A possible reason for this discrepancy could be caused by the linear interpolation of the velocity differences between the particles at the boundary. This might underestimate the real velocity difference and, hence, also the real stress caused by the relative movement. Furthermore, the introduction of additional energy via the shear could change the system in a way that makes is effectively different from the quiescent one.

The conservative viscosity shows a similar trend as the total viscosity. It starts from $\eta_{cons} = 0.0$ in the cases without conservative force and increases monotonically thereafter. The higher the density of a sample, the steeper is this increase. Error bars in this plot are based on the sum of errors from the kinetic viscosity and the total one. Overall, our experiments clearly show an agreement between both the static and the dynamic measuring technique, even though the trends and numerical values for the Lees Edwards measurements are less obvious and show a larger error.

B. Flow profile

We perform simulations for a DPD fluid with $N = 10000$ particles, a number density $\rho = 3$, a friction coefficient $\gamma = 4.5$, a cut-off radius $r_{\text{cut}} = 1.0$ and $F_{\text{max}} = 25.0$. The shear velocities in our simulations were $v = 0.1$, 1.0, and 1.5. We investigated the height dependence of the flow velocity in the gradient direction to check if the flow profile was properly equilibrated and uniform across the box. For this purpose we divided the box in 50 horizontal slabs, oriented along the gradient direction and determined the average velocity of the DPD fluid particles in each slab after a start-up time of 1000 $t = 50,000$. We show the average and standard deviation based on three different, independent snapshots for three different shear velocities in Figure 6. The expected flow linear flow profile is also included as dashed lines for comparison.

The resulting shear gradient is linear and in good agreement with the expected shape. Furthermore, an increase in the number of bins to improve the sampling resolution along the box height did not have an impact on the linearity or slope. We find that our implementation does not show any discontinuity or spurious flows in the shear profile near the simulation boundary. Therefore, the correction proposed by Chatterjee of omitting dissipative forces at the boundary is not required here and such discontinuities are not inherent to Lees-Edwards boundary conditions. We conclude that the corrections suggested by Chatterjee [34] are not necessary and confirm the previous research by Leimkuhler and Shang [18].

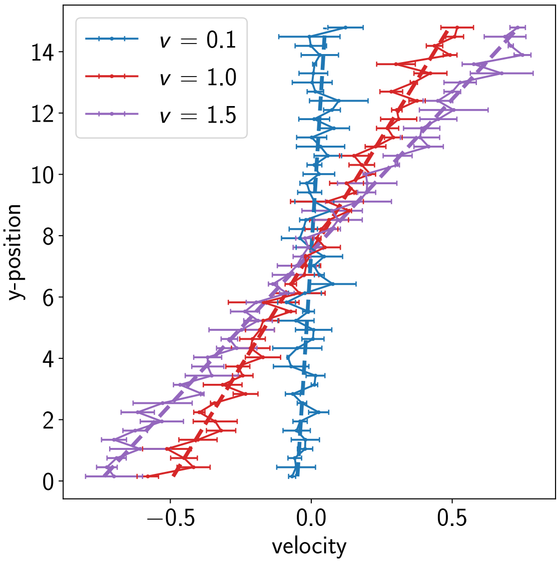

FIG. 6. **Height dependent velocity profile of the DPD fluid for 3 different shear velocities $v = [0.1, 1.0, 1.5]$. The expected linear shear profile is shown as a dashed line.**

C. Brownian motion with shear flow

The analysis of the mean-square displacement (MSD) enables the identification of Brownian motion by the linear dependence of the MSD on time. Whereas the viscosity of the DPD fluid is of Newtonian character, the MSD is influenced by the shear. In sheared systems, there exists a cubic-in-time contribution (Eq. (21)) to the MSD that was observed experimentally for polystyrene spheres by Orihara and Takikawa [42].

In computer simulations using the Lees-Edwards method, the study of diffusion depends on the ability to reconstruct the physical trajectories of the particles even though they experience "jumps" when crossing the boundaries. As the coordinates remain in the primary simulation box, we use the accumulated offset defined in Eq. (6) to obtain physically consistent trajectories. The study of Brownian motion thus serves as an extra verification of the correctness of our implementation. Once more we perform simulations for a DPD fluid with with the parameters mentioned in subsection IV B. A repulsive force of $F_{\max} = 25.0$ for the continuous shear simulations and $F_{\max} = 5.0$ for the oscillatory shear simulations was used.

1. Continuous shear

We chose five different shear velocities between $v = 0.1$ and $v = 1.5$ resulting in shear rates ranging from $\dot{\gamma} \approx 0.003$ and $\dot{\gamma} \approx 0.05$. The MSD of the particles was measured after equilibration of the shear flow.

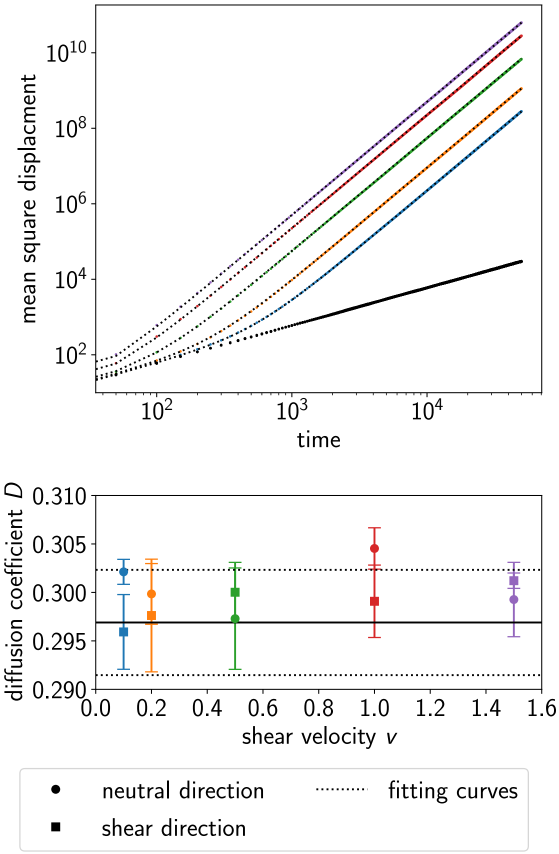

FIG. 7. **Results for the MSD and diffusion coefficient D in continuous shear. The upper part shows the development of the MSD over time for the neutral direction (black) and the shearing direction for 5 different shear rates $\dot{\gamma}$. The lower part shows diffusion coefficient D determined from fits to the neutral and shearing direction, for each shear rate.**

Figure 7 shows the expected scaling behaviour of the MSD for the neutral and the shearing direction for one example per shear rate. Colored curves show the actual measurement data, black dotted lines show fitted curves to this data. Following the relations presented in subsection III C the slope $m = 1.0$ in the neutral direction is unaffected by the shearing. The MSD in the shearing direction shows a gradual transition from $m = 1.0$ to $m = 3.0$ depending on the shear rate $\dot{\gamma}$. The corrected mean squared particle displacement, which accounts for the shear contribution, shows a cubic behaviour as expected from equation (21). The lower part of Figure 7 shows the measured diffusion coefficient D and standard deviation for the neutral and shearing direction as determined by three independent runs. We obtain these values by fitting the theoretical expressions of subsection III C to the measured values. The ratio of the diffusion coefficient in the neutral and vorticity direction has a maximum value of around 2.5%. Hence, we can say that the values are in excellent agreement with each other.

2. *Oscillatory shear*

We measured the diffusion coefficient in the neutral direction during the oscillatory flow to be $D = 0.61 \pm 0.01$. We then plot the expected effective diffusion coefficient D_{eff} following equation (22). The results shown were obtained from fits to 299 periods of oscillatory shear. Fittings of the MSDs were cut off at $\tau = 10^4$ as the MSD at larger times is the result of too few averaging points.

For oscillatory shear, we can obtain a phase dependent diffusion coefficient, as shown in Figure 8 for two different strains γ. Furthermore, we show the strain dependence of D_{eff} at two fixed phases ϕ which follows a squared relationship with the strain. Both results indicate the correct handling of jumps across a boundary and the correct handling of interactions. For large strains, we can observe a deviation of the measured D_{eff} that is significantly higher. This deviation can be explained by the high velocity at the boundaries. Since the shear velocity is higher in this case.

Hence, Lees Edwards boundary conditions are indeed translationally invariant and do not require any special modification at the boundary in order to avoid a spurious discontinuities in the flow. As long as the thermalization and velocity difference is calculated correctly by taking the shear velocity into account one can even model shear flow phenomena with a shear velocity that is changing over time.

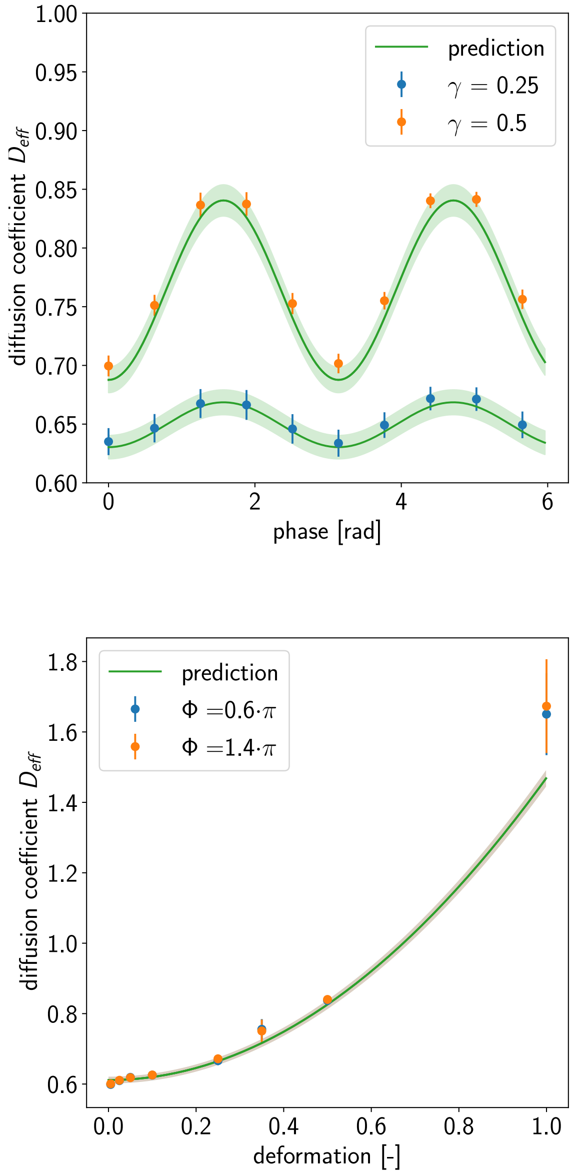

FIG. 8. **Results for the MSD and diffusion coefficient D_{eff} in oscillatory shear. The upper part illustrates the phase dependent diffusion coefficient for two different strains $\gamma = 0.25$ and $\gamma = 0.5$. The lower part shows the strain dependence of the diffusion coefficient D_{eff}.**

V. CONCLUSIONS

We have designed and implemented the method proposed by Lees and Edwards in 1972 for the simulation of linear shear flows in Molecular Dynamics. We provide in section II the information for the practical implementation in the simulation software, specifically on the distance function, the cell system, and the storage of the trajectory offset, that will be useful as a starting point for other scientists. In addition, our code is available publicly under an open-source license.

We demonstrated the Lees-Edwards method with a dissipative particle dynamics (DPD) fluid, a common choice in mesoscopic fluid simulations, to obtain a linear velocity profile. We find a good agreement between the equilibrium and non-equilibrium properties of DPD fluids as evaluated by Green-Kubo, for quiescent experiments, with Lees-Edwards boundary conditions experiments under shear flow. We were able to reconstruct continuous trajectories from the shear simulations, as if the system was infinitely extended, as is typically done for periodic simulation boxes. We observe the diffusion of particles with the mean square displacement and diffusion coefficient in equilibrium as well as in non-equilibrium situations, using then the reconstructed trajectories. We recover the predicted enhanced diffusion of Brownian particles in shear flow, which would be impossible to do without the quantity $x_{\text{part, LE}}$ defined in Eq. (6). These results are of special interest as they allow for a direct comparison to the experiment of Orihara and Takikawa under steady shear [42] and to the one of Takikawa and Orihara under oscillatory shear [43].

Our work opens up new possibilities to conduct numerical experiments involving simulations that require an explicit solvent undergoing shear flow within the convenient simulation package ESPResSo. We confirm the results of Leimkuhler and Shang [18] that the combination of DPD and Lees-Edwards yields a translationally invariant system, which ensures a sound basis for further research with this simulation setup.

One prospective use case is the yielding of gels where periodic boundaries are necessary to capture the macroscopic behavior. Highly localized restructuring can lead to a feedback between the applied shear deformation and network structure that would not be captured by other methods. This work also allows us to capture the shear-induced orientation of, e.g., soft particles or liquid crystals where many neighboring interactions must be considered.

ACKNOWLEDGMENTS

We acknowledge funding of the Research Foundation - Flanders (FWO) Odysseus Program (grant agreement number G0H9518N) and from the International Fine Particle Research Institute (IFPRI). Pierre de Buyl was a postdoctoral fellow of the Research Foundation-Flanders (FWO) while preparing most of this work. The resources and services used in this work were provided by the VSC (Flemish Supercomputer Center), funded by the Research Foundation - Flanders (FWO) and the Flemish Government.

Appendix A: Computational reproducibility

We perform the simulation with the package ESPResSo [35–38]. We based our work on version 4.0 of ESPResSo and took care to minimize the number of locations modified. Our modifications to ESPResSo are available on Zenodo [49]. ESPResSo is a C++ package with Python bindings, so that a simulation consists in a Python program that configures and executes the algorithm from the C++ "core". For dumping trajectories, we used the H5MD [50] writer of ESPResSo. H5MD is a HDF5-based specification for molecular simulation data.

We collected all the parameters, simulation programs, and analysis notebooks (jupyter notebooks `http://jupyter.org/`) in a dedicated repository, also archived on Zenodo, for reproducibility purposes [51]. We use NumPy [52] for basic numerical operations, SciPy [53] for numerical integration and curve fitting, matplotlib [54] for the figures, h5py [55] to read HDF5 files, tidynamics [45] to compute correlations, and pyblock [48] for the block analysis.

[1] D. E. Smith, H. P. Babcock, and S. Chu, Science **283**, 1724 (1999), https://science.sciencemag.org/content/283/5408/1724.full.pdf.

[2] L. G. Wilson and W. C. K. Poon, Phys. Chem. Chem. Phys. **13**, 10617 (2011).

[3] M. Tassieri, Current opinion in colloid & interface science (2019).

[4] J. P. Rich, J. Lammerding, G. H. McKinley, and P. S. Doyle, Soft Matter **7**, 9933 (2011).

[5] D. Frenkel and B. Smit, *Understanding Molecular Simulation (Second Edition)*, second edition ed. (Academic Press, San Diego, 2002) pp. 139 – 163.

[6] M. P. Allen and D. J. Tildesley, *Computer Simulation of Liquids.*, 2nd ed. (Oxford University Press, 2017) p. 640.

[7] A. Malevanets and R. Kapral, The Journal of chemical physics **110**, 8605 (1999).

[8] R. Kapral, Advances in Chemical Physics **140**, 89 (2008).

[9] R. Kubo, Reports on Progress in Physics **29**, 255 (1966).

[10] J. Wittmer, H. Xu, O. Benzerara, and J. Baschnagel, Molecular Physics **113**, 2881 (2015).

[11] P. Hoogerbrugge and J. Koelman, EPL (Europhysics Letters) **19**, 155 (1992).

[12] P. Espanol, Physical Review E **52**, 1734 (1995).

[13] T. Soddemann, B. Dünweg, and K. Kremer, Physical Review E **68**, 046702 (2003).

[14] V. Symeonidis, G. E. Karniadakis, and B. Caswell, Computing in science & engineering **7**, 39 (2005).

[15] W. Pan, B. Caswell, and G. E. Karniadakis, Langmuir **26**, 133 (2009).

[16] D. Pan, N. Phan-Thien, and B. C. Khoo, Journal of Non-Newtonian Fluid Mechanics **212**, 63 (2014).

[17] X. Li, P. M. Vlahovska, and G. E. Karniadakis, Soft Matter **9**, 28 (2013).

[18] B. Leimkuhler and X. Shang, Journal of Computational Physics **324**, 174 (2016).

[19] R. Khare, J. J. de Pablo, and A. Yethiraj, Macromolecules **29**, 7910 (1996).

[20] T. Kreer, M. Müser, K. Binder, and J. Klein, Langmuir **17**, 7804 (2001).

[21] R. Edberg, G. Morriss, and D. J. Evans, The Journal of chemical physics **86**, 4555 (1987).

[22] K. P. Travis, P. J. Daivis, and D. J. Evans, The Journal of chemical physics **103**, 1109 (1995).

[23] D. Evans and G. Morriss, *Non-Equilibrium Statistical Mechanics of Liquids* (Cambridge University Press, Cambridge, 2008).

[24] X. Shang, M. Kröger, and B. Leimkuhler, Soft matter **13**, 8565 (2017).

[25] A. Imperio, L. Reatto, and S. Zapperi, Physical Review E **78**, 021402 (2008).

[26] B. Hess, The Journal of Chemical Physics **116**, 209 (2002).

[27] S. Plimpton, Journal of computational physics **117**, 1 (1995).

[28] J. Cao and A. E. Likhtman, Physical review letters **108**, 028302 (2012).

[29] A. Lees and S. Edwards, Journal Of Physics C: Solid State Physics **5**, 1921 (1972).

[30] D. J. Evans and G. P. Morriss, Physical review letters **56**, 2172 (1986).

[31] L. Verlet, Physical review **159**, 98 (1967).

[32] W. C. Swope, H. C. Andersen, P. H. Berens, and K. R. Wilson, The Journal of Chemical

Physics **76**, 637 (1982).

[33] W. Smith, Computer Physics Communications **62**, 229 (1991).

[34] A. Chatterjee, Molecular Simulation **33**, 1233 (2007).

[35] A. Arnold, K. Breitsprecher, F. Fahrenberger, S. Kesselheim, O. Lenz, and C. Holm, Entropy **15**, 4569 (2013).

[36] A. Arnold, O. Lenz, S. Kesselheim, R. Weeber, F. Fahrenberger, D. Roehm, P. Košovan, and C. Holm, in *Meshfree methods for partial differential equations VI* (Springer, 2013) pp. 1–23.

[37] H.-J. Limbach, A. Arnold, B. A. Mann, and C. Holm, Computer Physics Communications **174**, 704 (2006).

[38] F. Weik, R. Weeber, K. Szuttor, K. Breitsprecher, J. de Graaf, M. Kuron, J. Landsgesell, H. Menke, D. Sean, and C. Holm, The European Physical Journal Special Topics **227**, 1789 (2019).

[39] R. D. Groot and P. B. Warren, The Journal of chemical physics **107**, 4423 (1997).

[40] P. Espanol and P. Warren, EPL (Europhysics Letters) **30**, 191 (1995).

[41] R. T. Foister and T. G. M. V. D. Ven, Journal of Fluid Mechanics **96**, 105–132 (1980).

[42] H. Orihara and Y. Takikawa, Phys. Rev. E **84**, 061120 (2011).

[43] Y. Takikawa and H. Orihara, Journal of the Physical Society of Japan **81**, 124001 (2012).

[44] G. R. Kneller, V. Keiner, M. Kneller, and M. Schiller, Computer Physics Communications **91**, 191 (1995).

[45] P. de Buyl, Journal of Open Source Software **3**, 877 (2018).

[46] E. Zohravi, E. Shirani, and A. Pishevar, Molecular Simulation **44**, 254 (2018).

[47] H. Flyvbjerg and H. G. Petersen, The Journal of Chemical Physics **91**, 461 (1989).

[48] J. Spencer, "pyblock," .

[49] S. Bindgen, P. de Buyl, F. Weik, K. Szuttor, R. Weeber, J. Landsgesell, J.-N. Grad, G. Rempfer, K. Breitsprecher, M. Kuron, T. Dr. Bogdan, H. Menke, O. Lenz, D. Sean, J. de Graaf, P. Kreissl, O. Hickey, S. Kesselheim, C. Junghans, and P. Kosovan, "bindgens1/espresso: Lees Edwards Implementation for Espresso," (2021), DOI: 10.5281/zenodo.4627017.

[50] P. de Buyl, P. H. Colberg, and F. Höfling, Comp. Phys. Commun. **185**, 1546 (2014).

[51] S. Bindgen and P. de Buyl, "bindgens1/lees_edwards_implementation: Companion data for the Lees Edwards Implementation for Espresso," (2021), DOI: 10.5281/zenodo.4719091.

[52] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del R'ıo, M. Wiebe, P. Peterson, P. G'erard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, Nature **585**, 357 (2020).

[53] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, Nature Methods **17**, 261 (2020).

[54] J. D. Hunter, Comput. Sci. Eng. **9**, 90 (2007).

[55] A. Collette, *Python and HDF5* (O'Reilly, Sebastopol, CA, 2013).