Coupling Molecular Dynamics and Lattice Boltzmann to Simulate Brownian Motion with Hydrodynamic Interactions

Burkhard Dünweg

Max Planck Institute for Polymer Research
Ackermannweg 10, 55128 Mainz, Germany
and
Department of Chemical Engineering, Monash University,
Melbourne, VIC 3800, Australia
E-mail: duenweg@mpip-mainz.mpg.de

In soft–matter systems where Brownian constituents are immersed in a solvent, both thermal fluctuations and hydrodynamic interactions are important. The article outlines a general scheme to simulate such systems by coupling Molecular Dynamics for the Brownian particles to a lattice Boltzmann algorithm for the solvent. By the example of a polymer chain immersed in solvent, it is explicitly demonstrated that this approach yields (essentially) the same results as Brownian Dynamics.

1 Introduction

Remark: The present contribution intends to just give a very brief overview over the subject matter. It is an updated version of a similar article[1] that the author has written on occasion of the 2009 NIC winter school. For more detailed information, the reader is referred to a longer review article, Ref. 2. —

Many soft–matter systems are comprised of Brownian particles immersed in a solvent. Prototypical examples are colloidal dispersions and polymer solutions, where the latter, in contrast to the former, are characterized by non–trivial internal degrees of freedom (here: the many possible conformations of the macromolecule). Fundamental for these systems is the separation of length and time scales between "large and slow" Brownian particles, and "small and fast" solvent particles. "Mesoscopic" simulations focus on the range of length and time scales which are, on the one hand, too small to allow a description just in terms of continuum mechanics of the overall system, but, on the other hand, large enough to allow the replacement of the solvent by a hydrodynamic continuum. This latter approximation is much less severe than one would assume at first glance; detailed Molecular Dynamics simulations have shown that hydrodynamics works as soon as the length scale exceeds a few particle diameters, and the time scale a few collision times.

To simulate such systems consistently, one has to take into account that the length and time scales are so small that thermal fluctuations cannot be neglected. The "Boltzmann number" Bo (a term invented by us) is a useful parameter for quantifying how important fluctuations are. Given a certain spatial resolution b (for example, the lattice spacing of a grid which is used to simulate the fluid dynamics), we may ask ourselves how many solvent particles N_p correspond to the scale b. On average, this is given by $N_p = \rho b^3/m_p$, where ρ is the mass density and m_p the mass of a solvent particle (and we assume a three–

dimensional system). The relative importance of fluctuations is then given by

$$Bo = N_p^{-1/2} = \left(\frac{m_p}{\rho b^3}\right)^{1/2}. \tag{1}$$

It should be noted that for an ideal gas, where the occupation statistics is Poissonian, Bo is just the relative statistical inaccuracy of the random variable N_p. In soft–matter systems, b is usually small enough such that Bo is no longer negligible.

Furthermore, *hydrodynamic interactions* must be modeled. In essence, this term refers to dynamic correlations between the Brownian particles, mediated by fast momentum transport through the solvent. The separation of time scales can be quantified in terms of the so–called Schmidt number

$$Sc = \frac{\eta_{kin}}{D}, \tag{2}$$

where $\eta_{kin} = \eta/\rho$ is the kinematic viscosity (ratio of dynamic shear viscosity η and mass density ρ) of the fluid, measuring how quickly momentum propagates diffusively through the solvent, and D is the diffusion constant of the particles. Typically, in a dense fluid $Sc \sim 10^2 \ldots 10^3$ for the solvent particles, while for large Brownian particles Sc is even much larger. Finally, we may also often assume that the solvent dynamics is in the creeping–flow regime, i. e. that the Reynolds number

$$Re = \frac{ul}{\eta_{kin}}, \tag{3}$$

where u denotes the velocity of the flow and l its typical size, is small. This is certainly true as long as the system is not driven strongly out of thermal equilibrium.

These considerations lead to the natural (but, in our opinion, not always correct) conclusion that the method of choice to simulate such systems is Brownian Dynamics (BD)[3]. Here the Brownian particles are displaced under the influence of particle–particle forces, hydrodynamic drag forces (calculated from the particle positions), and stochastic forces representing the thermal noise. However, the technical problems to do this efficiently for a large number N of Brownian particles are substantial. The calculation of the drag forces involves the evaluation of the hydrodynamic Green's function, which depends on the boundary conditions, and has an intrinsically long–range nature (such that all particles interact with each other). Furthermore, these drag terms also determine the correlations in the stochastic displacements, such that the generation of the stochastic terms involves the calculation of the matrix square root of a $3N \times 3N$ matrix. Recently, there has been substantial progress in the development of fast algorithms[4]; however, currently there are only few groups who master these advanced and complicated techniques. Apart from this, the applicability is somewhat limited, since the Green's function must be re–calculated for each new boundary condition, and its validity is questionable if the system is put under strong nonequilibrium conditions like, e. g., a turbulent flow — it should be noted that the Green's function is calculated for low–Re hydrodynamics.

Therefore, many soft–matter researchers have rather chosen the alternative approach, which is to simulate the system including the solvent degrees of freedom, with explicit momentum transport. The advantage of this is a simple algorithm, which scales linearly with the number of Brownian particles, and is easily parallelizable, due to its locality. The disadvantage, however, is that one needs to simulate many more degrees of freedom than

those in which one is genuinely interested — *and* to do this on the short inertial time scales in which one is not interested either. It is clear that such an approach involves essentially Molecular Dynamics (MD) for the Brownian particles.

Many ways are possible how to simulate the solvent degrees of freedom, and how to couple them to the MD part. It is just the universality of hydrodynamics that allows us to invent many models which all will produce the correct physics. The requirements are rather weak — the solvent model has to just be compatible with Navier–Stokes hydrodynamics on the macroscopic scale. Particle methods include Dissipative Particle Dynamics (DPD) and Multi–Particle Collision Dynamics (MPCD)[5], while lattice methods involve the direct solution of the Navier–Stokes equation on a lattice, or lattice Boltzmann (LB). The latter is a method with which we have made quite good experience, both in terms of efficiency and versatility. The efficiency comes from the inherent ease of memory management for a lattice model, combined with ease of parallelization, which comes from the high degree of locality: Essentially an LB algorithm just shifts populations on a lattice, combined with collisions, which however only happen locally on a single lattice site. The coupling to the Brownian particles (simulated via MD) can either be done via boundary conditions, or via an interpolation function that introduces a *dissipative* coupling between particles and fluid. In this article, we will focus on the latter method.

2 Coupling Scheme

As long as we view LB as just a solver for the Navier–Stokes equation, we may write down the equations of motion for the coupled system as follows:

$$\frac{d}{dt}\vec{r}_i = \frac{1}{m_i}\vec{p}_i, \tag{4}$$

$$\frac{d}{dt}\vec{p}_i = \vec{F}_i^c + \vec{F}_i^d + \vec{F}_i^f, \tag{5}$$

$$\partial_t \rho + \partial_\alpha j_\alpha = 0, \tag{6}$$

$$\partial_t j_\alpha + \partial_\beta \pi^E_{\alpha\beta} = \partial_\beta \eta_{\alpha\beta\gamma\delta}\partial_\gamma u_\delta + f^h_\alpha + \partial_\beta \sigma^f_{\alpha\beta}. \tag{7}$$

Here, $\vec{r}_i$, $\vec{p}_i$ and m_i are the positions, momenta, and masses of the Brownian particles, respectively. The forces $\vec{F}_i$ acting on the particles are conservative (c, i. e. coming from the interparticle potential), dissipative (d), and fluctuating (f). The equations of motion for the fluid have been written in tensor notation, where Greek indexes denote Cartesian components, and the Einstein summation convention is used. The first equation describes mass conservation; the mass flux $\rho\vec{u}$, where $\vec{u}$ is the flow velocity, is identical to the momentum density $\vec{j}$. The last equation describes the time evolution of the fluid momentum density. In the absence of particles, the fluid momentum is conserved. This part is described via the stress tensor, which in turn is decomposed into the conservative Euler stress $\pi^E_{\alpha\beta}$, the dissipative stress $\eta_{\alpha\beta\gamma\delta}\partial_\gamma u_\delta$, and the fluctuating stress $\sigma^f_{\alpha\beta}$. The influence of the particles is described via an external force density $\vec{f}^h$.

The coupling to a particle i is introduced via an interpolation procedure where first the flow velocities from the surrounding sites are averaged over to yield the flow velocity right

at the position of i. In the continuum limit, this is written as

$$\vec{u}_i \equiv \vec{u}(\vec{r}_i) = \int d^3\vec{r}\,\Delta(\vec{r},\vec{r}_i)\vec{u}(\vec{r}), \tag{8}$$

where $\Delta(\vec{r},\vec{r}_i)$ is a weight function with compact support, satisfying

$$\int d^3\vec{r}\Delta(\vec{r},\vec{r}_i) = 1. \tag{9}$$

Secondly, each particle is assigned a phenomenological friction coefficient Γ_i, and this allows us to calculate the friction force on particle i:

$$\vec{F}_i^d = -\Gamma_i\left(\frac{\vec{p}_i}{m_i} - \vec{u}_i\right). \tag{10}$$

A Langevin noise term $\vec{F}_i^f$ is added to the particle equation of motion, in order to compensate the dissipative losses that come from $\vec{F}_i^d$. $\vec{F}_i^f$ satisfies the standard fluctuation–dissipation relation

$$\left\langle F_{i\alpha}^f \right\rangle = 0, \tag{11}$$

$$\left\langle F_{i\alpha}^f(t)\, F_{j\beta}^f(t')\right\rangle = 2k_BT\Gamma_i\delta_{ij}\delta_{\alpha\beta}\delta(t-t'), \tag{12}$$

where T is is the absolute temperature and k_B the Boltzmann constant. While the conservative forces $\vec{F}_i^c$ conserve the total momentum of the particle system, as a result of Newton's third law, the dissipative and fluctuating terms ($\vec{F}_i^d$ and $\vec{F}_i^f$) do not. The associated momentum transfer must therefore have come from the fluid. The overall momentum must be conserved, however. This means that the force term entering the Navier–Stokes equation must just balance these forces. One easily sees that the choice

$$\vec{f}^h(\vec{r}) = -\sum_i\left(\vec{F}_i^d + \vec{F}_i^f\right)\Delta(\vec{r},\vec{r}_i) \tag{13}$$

satisfies this criterion. It should be noted that we use the *same* weight function to interpolate the forces back onto the fluid; this is necessary to satisfy the fluctuation–dissipation theorem for the overall system, i. e. to simulate a well–defined constant–temperature ensemble. The detailed proof of the thermodynamic consistency of the procedure can be found in Ref. 2.

We still need to specify the remaining terms in the Navier–Stokes equation. The viscosity tensor $\eta_{\alpha\beta\gamma\delta}$ describes an isotropic Newtonian fluid:

$$\eta_{\alpha\beta\gamma\delta} = \eta\left(\delta_{\alpha\gamma}\delta_{\beta\delta} + \delta_{\alpha\delta}\delta_{\beta\gamma} - \frac{2}{3}\delta_{\alpha\beta}\delta_{\gamma\delta}\right) + \eta_b\delta_{\alpha\beta}\delta_{\gamma\delta}, \tag{14}$$

with shear and bulk viscosities η and η_b. This tensor also appears in the covariance matrix of the fluctuating (Langevin) stress $\sigma_{\alpha\beta}^f$:

$$\left\langle \sigma_{\alpha\beta}^f \right\rangle = 0, \tag{15}$$

$$\left\langle \sigma_{\alpha\beta}^f(\vec{r},t)\,\sigma_{\gamma\delta}^f(\vec{r}',t')\right\rangle = 2k_BT\eta_{\alpha\beta\gamma\delta}\delta(\vec{r}-\vec{r}')\,\delta(t-t'). \tag{16}$$

Finally, the Euler stress

$$\pi^E_{\alpha\beta} = p\delta_{\alpha\beta} + \rho u_\alpha u_\beta \tag{17}$$

describes the equation of state of the fluid (p is the thermodynamic pressure), and convective momentum transport.

3 Low Mach Number Physics

At this point an important simplification can be made. The equation of state only matters for flow velocities u that are comparable with the speed of sound c_s, i. e. for which the Mach number

$$Ma = \frac{u}{c_s} \tag{18}$$

is large. In the low Mach number regime, the flow may be considered as effectively incompressible (although no incompressibility constraint is imposed in the algorithm). The Mach number should not be confused with the Reynolds number Re, which rather measures whether inertial effects are important. Now it turns out that essentially all soft–matter applications "live" in the low–Ma regime. Furthermore, large Ma is anyway inaccessible to the LB algorithm, since it provides only a finite set of lattice velocities — and these essentially determine the value of c_s. In other words, the LB algorithm simply cannot realistically represent flows whose velocity is not small compared to c_s. For this reason, the details of the equation of state do not matter, and therefore one chooses the system that is by far the easiest — the ideal gas. Here the equation of state for a system at temperature T may be written as

$$k_B T = m_p c_s^2. \tag{19}$$

In the D3Q19 model (the most popular standard LB model in three dimensions, using nineteen lattice velocities, see below) it turns out that the speed of sound is given by

$$c_s^2 = \frac{1}{3}\frac{b^2}{h^2}, \tag{20}$$

where b is the lattice spacing and h the time step. Therefore the Boltzmann number can also be written as

$$Bo = \left(\frac{m_p}{\rho b^3}\right)^{1/2} = \left(\frac{3k_B T h^2}{\rho b^5}\right)^{1/2}. \tag{21}$$

4 Lattice Boltzmann 1: Statistical Mechanics

The lattice Boltzmann algorithm starts from a regular grid with sites $\vec{r}$ and lattice spacing b, plus a time step h. We then introduce a small set of velocities $\vec{c}_i$ such that $\vec{c}_i h$ connects two nearby lattice sites on the grid. In the D3Q19 model, the lattice is simple cubic, and the nineteen velocities correspond to the six nearest and twelve next–nearest neighbors, plus a zero velocity. On each lattice site $\vec{r}$ at time t, there are nineteen populations $n_i(\vec{r}, t)$.

Each population is interpreted as the mass density corresponding to velocity $\vec{c}_i$. The total mass and momentum density are therefore given by

$$\rho(\vec{r},t) = \sum_i n_i(\vec{r},t), \tag{22}$$

$$\vec{j}(\vec{r},t) = \sum_i n_i(\vec{r},t)\vec{c}_i, \tag{23}$$

such that the flow velocity is obtained via $\vec{u} = \vec{j}/\rho$. The number of "lattice Boltzmann particles" which correspond to n_i is given by

$$\nu_i = \frac{n_i b^3}{m_p} \equiv \frac{n_i}{\mu}, \tag{24}$$

where m_p is the mass of a lattice Boltzmann particle, and μ the corresponding mass density. It should be noted that μ is a measure of the thermal fluctuations in the system, since, according to Eq. 21, one has $Bo^2 = \mu/\rho$.

If we now assume a "velocity bin" i to be in thermal contact with a large reservoir of particles, the probability density for ν_i is Poissonian. Furthermore, if we assume that the "velocity bins" are statistically independent, but take into account that mass and momentum density are fixed (these variables are conserved quantities during an LB collision step and should therefore be handled like conserved quantities in a microcanonical ensemble), we find

$$P\left(\{\nu_i\}\right) \propto \left(\prod_i \frac{\bar{\nu}_i^{\nu_i}}{\nu_i!} e^{-\bar{\nu}_i}\right) \delta\left(\mu \sum_i \nu_i - \rho\right) \delta\left(\mu \sum_i \nu_i \vec{c}_i - \vec{j}\right). \tag{25}$$

for the probability density of the variables ν_i. This must be viewed as the statistics which describes the local (single–site) equilibrium under the condition of fixed values of the hydrodynamic variables ρ and $\vec{j}$. The parameter $\bar{\nu}_i$ is the mean occupation imposed by the reservoir, and we assume that it is given by

$$\bar{\nu}_i = a^{c_i} \frac{\rho}{\mu}, \tag{26}$$

where $a^{c_i} > 0$ is a weight factor corresponding to the neighbor shell with speed c_i.

From normalization and cubic symmetry we know that the low–order velocity moments of the weights must have the form

$$\sum_i a^{c_i} = 1, \tag{27}$$

$$\sum_i a^{c_i} c_{i\alpha} = 0, \tag{28}$$

$$\sum_i a^{c_i} c_{i\alpha} c_{i\beta} = \sigma_2 \, \delta_{\alpha\beta}, \tag{29}$$

$$\sum_i a^{c_i} c_{i\alpha} c_{i\beta} c_{i\gamma} = 0, \tag{30}$$

$$\sum_i a^{c_i} c_{i\alpha} c_{i\beta} c_{i\gamma} c_{i\delta} = \kappa_4 \, \delta_{\alpha\beta\gamma\delta} + \sigma_4 \left(\delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\gamma}\delta_{\beta\delta} + \delta_{\alpha\delta}\delta_{\beta\gamma}\right), \tag{31}$$

where σ_2, σ_4, κ_4 are yet undetermined constants, while $\delta_{\alpha\beta\gamma\delta}$ is unity if all four indexes are the same and zero otherwise.

Employing Stirling's formula for the factorial, it is straightforward to find the set of populations n_i^{eq} which maximizes P under the constraints of given ρ and $\vec{j}$. Up to second order in u (low Mach number!) the solution is given by

$$n_i^{eq} = \rho a^{c_i} \left(1 + \frac{\vec{u} \cdot \vec{c}_i}{\sigma_2} + \frac{(\vec{u} \cdot \vec{c}_i)^2}{2\sigma_2^2} - \frac{u^2}{2\sigma_2} \right). \tag{32}$$

The low–order moments of the equilibrium populations are then given by

$$\sum_i n_i^{eq} = \rho, \tag{33}$$

$$\sum_i n_i^{eq} c_{i\alpha} = j_\alpha, \tag{34}$$

$$\sum_i n_i^{eq} c_{i\alpha} c_{i\beta} = \rho c_s^2 \delta_{\alpha\beta} + \rho u_\alpha u_\beta. \tag{35}$$

The first two equations are just the imposed constraints, while the last one (meaning that the second moment is just the hydrodynamic Euler stress) follows from imposing two additional conditions, which is to choose the weights a^{c_i} such that they satisfy $\kappa_4 = 0$ and $\sigma_4 = \sigma_2^2 (= c_s^4)$. From the Chapman–Enskog analysis of the LB dynamics (see below) it follows that the asymptotic behavior in the limit of large length and time scales is compatible with the Navier–Stokes equation only if Eq. 35 holds, and this in turn is only possible if the abovementioned isotropy conditions are satisfied. Together with the normalization condition, we thus obtain a set of three equations for the a^{c_i}. Therefore at least three neighbor shells are needed to satisfy these conditions, and this is the reason for choosing a nineteen–velocity model. For D3Q19, one thus obtains $a^{c_i} = 1/3$ for the zero velocity, $1/18$ for the nearest neighbors, and $1/36$ for the next–nearest neighbors. Furthermore, one finds $c_s^2 = \sigma_2 = (1/3) b^2/h^2$.

For the fluctuations around the most probable populations n_i^{eq},

$$n_i^{neq} = n_i - n_i^{eq}, \tag{36}$$

we employ a saddle–point approximation and approximate u by zero. This yields

$$P\left(\{n_i^{neq}\}\right) \propto \exp\left(- \sum_i \frac{(n_i^{neq})^2}{2\mu\rho a^{c_i}} \right) \delta\left(\sum_i n_i^{neq} \right) \delta\left(\sum_i \vec{c}_i \, n_i^{neq} \right). \tag{37}$$

We now introduce normalized fluctuations via

$$\hat{n}_i^{neq} = \frac{n_i^{neq}}{\sqrt{\mu\rho a^{c_i}}} \tag{38}$$

and transform to normalized "modes" (symmetry–adapted linear combinations of the n_i, see Ref. 2) $\hat{m}_k^{neq}$ via an orthonormal transformation $\hat{e}_{ki}$:

$$\hat{m}_k^{neq} = \sum_i \hat{e}_{ki} \hat{n}_i^{neq}, \tag{39}$$

$k = 0, \ldots, 18$, and obtain

$$P\left(\{m_k\}\right) \propto \exp\left(-\frac{1}{2}\sum_{k\geq 4} m_k^2\right). \tag{40}$$

It should be noted that the modes number zero to three have been excluded; they are just the conserved mass and momentum densities.

5 Lattice Boltzmann 2: Stochastic Collisions

A collision step consists of re-arranging the set of n_i on a given lattice site such that both mass and momentum are conserved. Since the algorithm should simulate thermal fluctuations, this should be done in a way that is (i) stochastic and (ii) consistent with the developed statistical–mechanical model. This is straightforwardly imposed by requiring that the collision is nothing but a Monte Carlo procedure, where a Monte Carlo step transforms the pre–collisional set of populations, n_i, to the post–collisional one, $n_i^\star$. Consistency with statistical mechanics can be achieved by requiring that the Monte Carlo update satisfies the condition of detailed balance. Most easily this is done in terms of the normalized modes $\hat{m}_k$, which we update according to the rule ($k \geq 4$)

$$\hat{m}_k^\star = \gamma_k \hat{m}_k + \sqrt{1-\gamma_k^2}\, r_k. \tag{41}$$

Here the γ_k are relaxation parameters with $-1 < \gamma_k < 1$, and the r_k are statistically independent Gaussian random numbers with zero mean and unit variance. Mass and momentum are automatically conserved since the corresponding modes are not updated. Comparison with Eq. 40 shows that the procedure indeed does satisfy detailed balance. The parameters γ_k can in principle be chosen at will; however, they should be compatible with symmetry. For example, mode number four corresponds to the bulk stress, with a relaxation parameter γ_b, while modes number five to nine correspond to the five shear stresses, which form a symmetry multiplett. Therefore one must choose $\gamma_5 = \ldots = \gamma_9 = \gamma_s$. For the remaining kinetic modes one often uses $\gamma_k = 0$ for simplicity, but this is not necessary.

6 Lattice Boltzmann 3: Chapman–Enskog Expansion

The actual LB algorithm now consists of alternating collision and streaming steps, as summarized in the LB equation (LBE):

$$n_i(\vec{r} + \vec{c}_i h, t + h) = n_i^\star(\vec{r}, t) = n_i(\vec{r}, t) + \Delta_i\left\{n_i(\vec{r}, t)\right\}. \tag{42}$$

The populations are first re–arranged on the lattice site; this is described by the so–called "collision operator" Δ_i. The resulting post–collisional populations $n_i^\star$ are then propagated to the neighboring sites, as expressed by the left hand side of the equation. After that, the next collision step is done, etc.. The collision step may include momentum transfer as a result of external forces (for details, see Ref. 2); apart from that, it is just given by the update procedure outlined in the previous section.

A convenient way to find the dynamic behavior of the algorithm on large length and time scales is a multi–time–scale analysis. One introduces a "coarse–grained ruler" by transforming from the original coordinates $\vec{r}$ to new coordinates $\vec{r}_1$ via

$$\vec{r}_1 = \epsilon \vec{r}, \tag{43}$$

where ϵ is a dimensionless parameter with $0 < \epsilon \ll 1$. The rationale behind this is the fact that any "reasonable" value for the scale r_1 will automatically force r to be large. In other words: By considering the limit $\epsilon \to 0$ we automatically focus our attention on large length scales. The same is done for the time; however, here we introduce *two* scales via

$$t_1 = \epsilon t \tag{44}$$

and

$$t_2 = \epsilon^2 t. \tag{45}$$

The reason for this is that one needs to consider both wave–like phenomena, which happen on the t_1 time scale (i. e. the real time is moderately large), and diffusive processes (where the real time is *very* large). We now write the LB variables as a function of $\vec{r}_1, t_1, t_2$ instead of $\vec{r}, t$. Since changing ϵ at fixed $\vec{r}_1$ changes $\vec{r}$ and thus n_i, we must take into account that the LB variables depend on ϵ:

$$n_i = n_i^{(0)} + \epsilon n_i^{(1)} + \epsilon^2 n_i^{(2)} + O(\epsilon^3). \tag{46}$$

The same is true for the collision operator:

$$\Delta_i = \Delta_i^{(0)} + \epsilon \Delta_i^{(1)} + \epsilon^2 \Delta_i^{(2)} + O(\epsilon^3). \tag{47}$$

In terms of the new variables, the LBE is written as

$$n_i(\vec{r}_1 + \epsilon \vec{c}_i h, t_1 + \epsilon h, t_2 + \epsilon^2 h) - n_i(\vec{r}_1, t_1, t_2) = \Delta_i. \tag{48}$$

Now, one systematically Taylor–expands the equation up to order ϵ^2. Sorting by order yields a hierarchy of LBEs of which one takes the zeroth, first, and second velocity moment. Systematic analysis of this set of moment equations (for details, see Ref. 2) shows that the LB procedure, as it has been developed in the previous sections, indeed yields the fluctuating Navier–Stokes equations in the asymptotic $\epsilon \to 0$ limit — however only for low Mach numbers; in the high Mach number regime, where terms of order u^3/c_s^3 can no longer be neglected, the dynamics definitely deviates from Navier–Stokes.

In particular, this analysis shows that the zeroth–order populations must be identified with n_i^{eq}, and that it is *necessary* that this "encodes" the Euler stress via suitably chosen weights a^{c_i}. Furthermore, one finds explicit expressions for the viscosities:

$$\eta = \frac{h\rho c_s^2}{2} \frac{1+\gamma_s}{1-\gamma_s}, \tag{49}$$

$$\eta_b = \frac{h\rho c_s^2}{3} \frac{1+\gamma_b}{1-\gamma_b}. \tag{50}$$

7 A Polymer Chain in Solvent

In Ref. 6 we explicitly aimed at a comparison between BD and coupled LB–MD for the *same* system. We chose a well–studied standard benchmark system, a single bead–spring polymer chain of N monomers in good solvent in thermal equilibrium. The BD algorithm is realized via

$$r_{i\alpha}(t+h) = r_{i\alpha}(t) + (k_B T)^{-1} D_{ij\alpha\beta} F_{j\beta} h + \sqrt{2h} B_{ij\alpha\beta} W_{j\beta}, \qquad i = 1, 2, \ldots, N. \quad (51)$$

Here $\vec{r}_i$ is the coordinate of the ith particle, h is the BD time step, $\overleftrightarrow{D}_{ij}$ is the diffusion tensor coupling particles i and j, and $\vec{F}_j$ denotes the deterministic force on particle j (here spring force and excluded–volume force). We assume summation convention with respect to both Cartesian and particle indexes. The tensor $\overleftrightarrow{B}_{ij}$ is the matrix square root of $\overleftrightarrow{D}_{ij}$,

$$D_{ij\alpha\beta} = B_{ik\alpha\gamma} B_{jk\beta\gamma}, \quad (52)$$

while $\vec{W}_i$ is a discretized Wiener process, $\langle W_{i\alpha} \rangle = 0$ and $\langle W_{i\alpha} W_{j\beta} \rangle = \delta_{ij} \delta_{\alpha\beta}$. For the diffusion tensor we used the Rotne–Prager tensor. The computationally most demanding part is the calculation of the matrix square root. The exact numerical solution of this problem via Cholesky decomposition has a computational complexity $O(N^3)$. We therefore rather used Fixman's trick[7,8] to speed up the calculations. This is based on the observation that the "square root" function, if viewed as a function acting on real numbers, needs to be evaluated only within a finite interval, spanning from the smallest to the largest eigenvalue. Since this interval does not contain the singularity at zero, a truncated (Chebyshev) polynomial expansion approximates the function quite well. The same expansion can then also be used to evaluate the matrix square root. The number of terms needed is empirically found to scale as $O(N^{0.25})$. Furthermore, for each term one needs to do a "matrix times vector" operation, which scales as $O(N^2)$, such that the algorithm in total has a computational complexity $O(N^{2.25})$. We did not employ an FFT–based "superfast" BD algorithm[4]; this would have been quite complicated, and also required to assume a simulation box of size L^3 with periodic bondary conditions, such that an extrapolation $L \to \infty$ would have been necessary.

Such a finite box size, combined with an extrapolation, is however precisely what is needed for LB–MD. We therefore ran these simulations for at least three different values of L in order to allow for meaningful extrapolations (and used the total time for these three systems to estimate our CPU effort). The typical box sizes that are needed are given by the requirement that the polymer chain should fit nicely into the box, without much back–folding. Since in a good solvent the polymer radius R scales as $R \propto N^\nu$, where $\nu \approx 0.59$ is the Flory exponent, we find $L^3 \propto N^{3\nu}$. Furthermore, the computational cost is completely dominated by the operations required to run the solvent, and hence the computational complexity is $O(N^{3\nu}) = O(N^{1.8})$. We see that this is slightly better than BD; however, the prefactor is much smaller for BD. In practice, we find that BD is roughly two orders of magnitude faster than LB–MD, for the typical chain lengths used in simulations, see Fig. 1.

The situation is expected to be quite different when one studies a semidilute solution, where the monomer concentration is still quite low (such that the LB–MD CPU effort is still dominated by the solvent), but the chains are so long that they strongly overlap. For

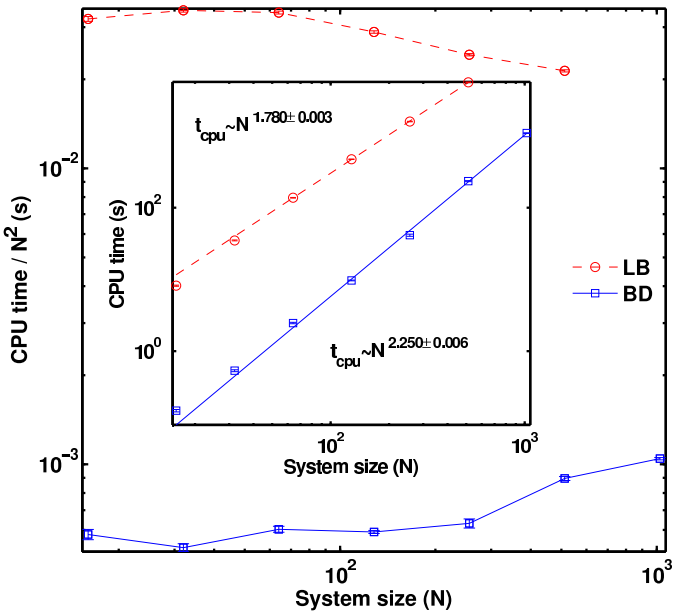

Figure 1. Comparison of the CPU time needed by the LB–MD and BD systems for the equivalent of 1000 LB–MD time steps for various chain lengths N. From Ref. 6.

example, Ref. 9 studied 50 chains of length $N = 1000$, being well in the semidilute regime. While the additional chains for the LB–MD system pose essentially no computational burden at all (rather on the contrary: Flory screening makes the chains shrink, such that one can afford to run the simulation in a somewhat smaller box), the BD effort (for our algorithm) is expected to increase by a factor of $50^{2.25}$, i. e. more than three orders of magnitude — or even more, since one needs a more complicated scheme to evaluate the hydrodynamic interactions for a periodic system. In other words: For such a system, BD can at best be competitive if the "superfast"[4] version is implemented — and to our knowledge, this has not yet been tested.

In order to allow a meaningful comparison, both systems have to be run for the same system and the same parameters. This implies, firstly, identical interaction potentials between the beads, and the same temperature. From this one concludes (and numerically verifies) that the static properties like gyration radius, static structure factor, etc., must all be identical. For the dynamics, it is important that both simulations are run with the same value for the shear viscosity η, which is easy to achieve, plus with the same value for the monomeric friction coefficient. At this point, one has to take into account that the friction coefficient ζ that appears in the BD algorithm (on the diagonal of the diffusion tensor) is a *long–time* friction coefficient, which describes the asymptotic stationary velocity $\vec{v}$ of a particle that is dragged through the fluid with a force $\vec{F}$, $\vec{F} = \zeta\vec{v}$, while the friction coefficient Γ that appears in the LB–MD algorithm via the coupling prescription $\vec{F} = \Gamma(\vec{v} - \vec{u})$ (see Eq. 10) is a corresponding *short–time* coefficient that does not yet take the backflow effects into account. Indeed, for an experiment in which a particle is dragged through the

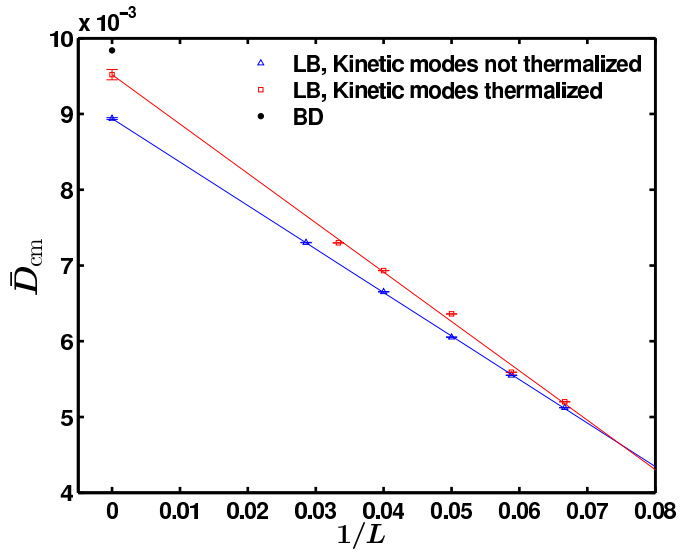

Figure 2. The dimensionless long time diffusion constant for the center of mass at various box lengths L. From Ref. 6

LB fluid, it is clear that the flow velocity $\vec{u}$ will be nonzero, and typically slightly smaller than $\vec{v}$. Hence, $\vec{F} = \zeta\vec{v} = \Gamma(\vec{v} - \vec{u})$, i. e. ζ is smaller than Γ. Since hydrodynamics allows us to estimate $\vec{u}$ up to a numerical prefactor g via a Stokes–like formula, $\vec{F} = g\eta a\vec{u}$, where a is the range of the interpolation scheme, one finds (see also Ref. 2)

$$\zeta^{-1} = \Gamma^{-1} + (g\eta a)^{-1}. \tag{53}$$

For nearest–neighbor linear interpolation, one finds $g \approx 25$ if a is identified with the LB lattice spacing. One hence needs to choose the Γ value in the LB–MD simulations in such a way that it reproduces the BD ζ value.

The diffusion constant of the LB–MD chain depends on the box size, as a result of the hydrodynamic interaction with the periodic images. Since the latter decays like r^{-1}, one concludes an L^{-1} finite size effect, which is nicely borne out by the data of Fig. 2. From these data one sees also that for an accurate description of the dynamics it is necessary to not only thermalize the stress modes in the LB algorithm (only these matter in the strict hydrodynamic limit), but also the kinetic modes, as suggested by the more microscopic theory outlined above. Taking the finite–size effect and the proper thermalization into account, the remaining deviation between BD and LB–MD is only a few percent.

The internal Rouse modes of the chain are defined as ($p = 1, 2, \ldots, N-1$)

$$\vec{X}_p = \frac{1}{N}\sum_{n=1}^{N} \vec{r}_n \cos\left[\frac{p\pi}{N}\left(n - \frac{1}{2}\right)\right]. \tag{54}$$

Figure 3 shows the decay of the normalized mode autocorrelation function up to $p = 5$. Obviously the agreement with BD is quite good, i. e. the finite size effect is quite weak. The reason is the following: The diffusion constant corresponds to the friction of the chain

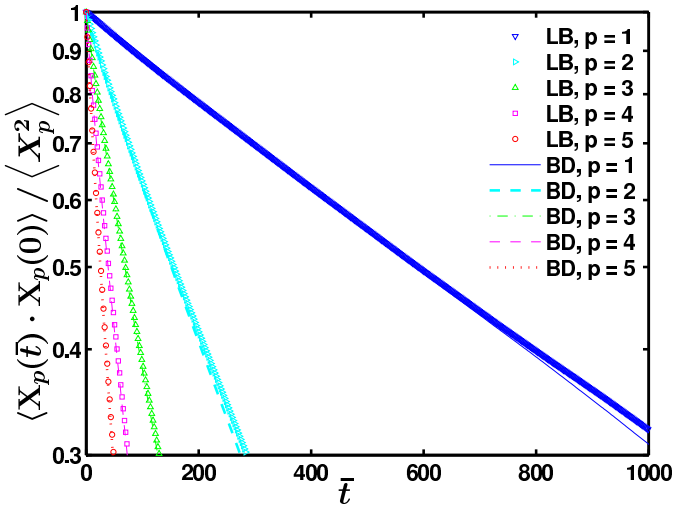

Figure 3. Normalized autocorrelation function of the first 5 Rouse modes $\vec{X}_p$ for LB–MD simulations at fixed $L = 25$ and BD simulations at $L \to \infty$. From Ref. 6.

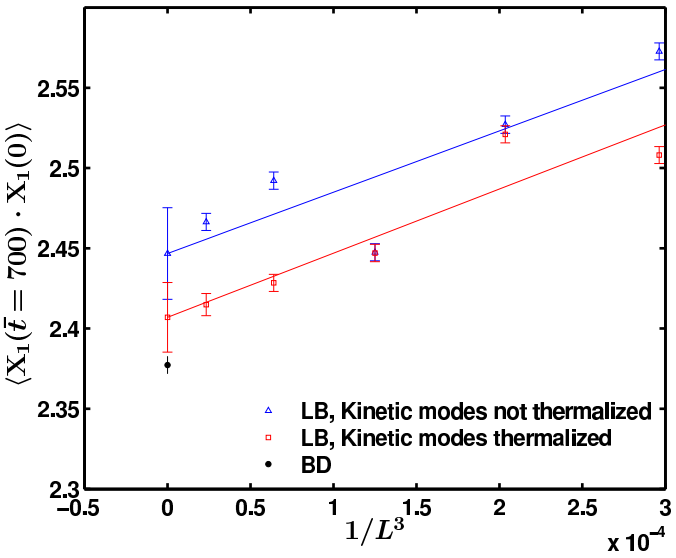

Figure 4. The autocorrelation function for the first Rouse mode $\vec{X}_1$ at a finite time value of $\bar{t} = 700$ for LB–MD simulations at various box lengths L and BD simulations at $L \to \infty$. From Ref. 6.

as a whole, i. e. to an experiment where the chain is being dragged through the fluid with a constant force. This gives rise to a flow field that decays like r^{-1}, and thus an L^{-1} finite size effect. This total force may also be viewed as the monopole moment of a distribution of forces acting on the polymer. The Rouse modes however study the *internal* motion of

the chain, i. e. in the center–of–mass system. Therefore, the monopole contribution of the forces has been subtracted, and only highr–order multipole moments remain. The dipole contribution vanishes for symmetry reasons, i. e. the first higher–order multipole is the quadrupole (this may be vaguely understood by recalling that the mass distribution has a monopole and a quadrupole moment, but not a dipole moment). The quadrupolar flow field decays like r^{-3}, and hence one expects an L^{-3} finite size effect. For a more detailed derivation, see Ref. 10. This finite size effect is indeed observed, see Fig. 4, demonstrating that on the one hand the system is theoretically quite well understood, and that on the other hand such simulations are nowadays so accurate that even rather subtle effects can be analyzed unambiguously.

References

1. B. Dünweg, "Computer simulations of systems with hydrodynamic interactions: The coupled Molecular Dynamics - lattice Boltzmann approach", in: Multiscale Simulation Methods in Molecular Sciences, J. Grotendorst, N. Attig, S. Blügel, and D. Marx, (Eds.). Forschungszentrum Jülich, Jülich, 2009.
2. B. Dünweg and A. J. C. Ladd, *Lattice Boltzmann simulations of soft matter systems*, Advances in Polymer Science, **221**, 89, 2009.
3. G. Nägele, "Brownian dynamics simulations", in: Computational Condensed Matter Physics, S. Blügel, G. Gompper, E. Koch, H. Müller-Krumbhaar, R. Spatschek, and R. G. Winkler, (Eds.). Forschungszentrum Jülich, Jülich, 2006.
4. A. J. Banchio and J. F. Brady, Journal of Chemical Physics, **118**, 10323, 2003.
5. M. Ripoll, "Mesoscale hydrodynamics simulations", in: Computational Condensed Matter Physics, S. Blügel, G. Gompper, E. Koch, H. Müller-Krumbhaar, R. Spatschek, and R. G. Winkler, (Eds.). Forschungszentrum Jülich, Jülich, 2006.
6. Tri T. Pham, Ulf D. Schiller, J. Ravi Prakash, and B. Dünweg, Journal of Chemical Physics, **131**, 164114, 2009.
7. M. Fixman, Macromolecules, **19**, 1204, 1986.
8. R. M. Jendrejack, M. D. Graham, and J. J. De Pablo, Journal of Chemical Physics, **113**, 2894, 2000.
9. P. Ahlrichs, R. Everaers, and B. Dünweg, Physical Review E, **64**, 040501, 2001.
10. P. Ahlrichs and B. Dünweg, Journal of Chemical Physics, **111**, 8225, 1999.