Particle-Size Effects on the Enhanced Diffusion of Tracer Particles in Microswimmer Suspensions

Thesis for the degree of Master of Science,
Project Duration: 4 months

André Nüßlein

LUNDS
UNIVERSITET

Supervisors: Joakim Stenhammar, Ferdi Aryasetiawan
Department of Physics
Division: Mathematical Physics
May 2020

Abstract

Active suspensions of microswimmers such as bacteria or microalgae are found in oceans or lakes, and within living organisms, such as the human body. These suspensions can exhibit complex flow patterns and enhanced diffusion of passive tracer particles due to the advection from the long-ranged dipolar flow fields of the swimmers. The diffusion of tracers at varying radii is poorly understood but one recent experimental study points to a non-monotonic behaviour with a certain radius that maximizes diffusion. In this thesis, we study the effect of nonlinearities in the flow field on the effective diffusion of spherical tracer particles. To do so, we model the swimmers as force dipoles which create a known fluid field around them. In a single-swimmer-single-tracer simulation this field is used directly to study the advection while a lattice Boltzmann simulation allows for many-particle simulations. For non-interacting swimmers, corresponding to very dilute suspensions, we find that the diffusion coefficient as a function of tracer radius is non-monotonic, although it is convex in the probed range. However, the simple one-swimmer-one-tracer simulation indicates that the many-particle simulation is only valid below a certain tracer radius ($R_0 = 2.5 \times$ the swimmers' length) where the function is slightly decreasing. In this range, for interacting swimmers, the effect of interaction is increased for both of the studied swimmer types, pushers and pullers.

Acknowledgements

I would first like to thank my supervisor Joakim Stenhammar who welcomed me, a physicist, at the Department of Chemistry. His door was always open and, during the COVID-19 isolation, he was helpful from a distance.

I would also like to thank my co-supervisor Ferdi Aryasetiawan who made it possible for me to write a thesis at another department while still getting a Master's degree in physics.

My office mates Henrik and Dóra must also not be left unacknowledged. They reviewed my code, helped me with Linux commands and the LBM code, and introduced me to the lovely Department of Chemistry.

Abbreviations

BGK	Bhatnagar–Gross–Krook
CFD	computational fluid dynamics
E. coli	Escherichia coli
LBM	lattice Boltzmann method
NS	Navier-Stokes

Contents

1. Introduction

Bacteria are one of the earliest forms of life on earth and while they seem simple when compared to the human organism, they are still actively researched in several disciplines. For physicists, one interesting aspect about them is their collective behaviour, in other words, how they swim together in a suspension of many other bacteria. While each and every one of them can swim independently, they swim in a correlated fashion once they overcome a critical density. Just like birds or fishes, they dispose of some advanced means of communication (chemotaxis and quorum sensing) but in 3D the correlations are mainly a result of their individual fluid flows interacting. Large scale simulations have shown exactly this [2] and a result from such a simulation can be seen in figure 1.1.

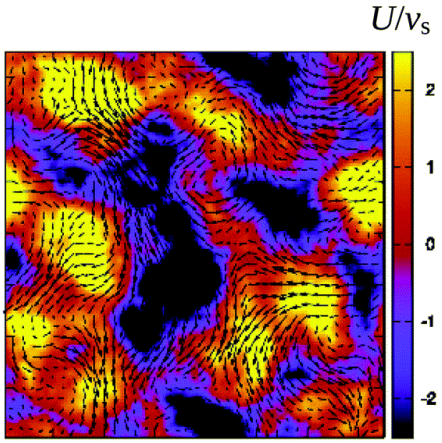

Figure 1.1: The fluid velocity in one plane of a 3D microswimmer simulation where vectors show the velocity field in the plane and the colours indicate its out-of-plane component. Adapted from [1] with permission from The Royal Society of Chemistry.

Many swimming microorganisms can be classified as either pushers or pullers. Pushers have flagella attached to their rear end that rotate and thus propel the swimmer forward, the pullers' flagella achieve this by "breaststrokes". In doing so, they create flow fields whose far field has a dipolar structure [3]. A good model for pusher is a positive force dipole, as indicated in figure 1.2. The most iconic example is the *Escherichia coli* (*E. coli*) bacterium (around 2 µm in size) and one puller example would be the algae *Chlamydomonas* (around 20 µm in size). The puller model is very similar to the pusher one, only that the forces would point inwards. These simple models explain why pusher and puller suspensions differ greatly when their density is high. For example, pushers exhibit active turbulence, pullers do not [1] and this is a result entirely due to hydrodynamic interactions.

The cartoon figure 1.2 also includes a tracer particle that is modeled as a simple sphere (usually bigger than the swimmer). In nature, such particles could be nutrients or just sand that is stirred up in oceans. Without microswimmers, passive particles would still move by thermal diffusion (always damped by fluid viscosity). This is a result from collisions with the fluid molecules, and leads to random trajectories, called Brownian motion. If the fluid additionally contains bacteria (or other self-propelling particles), equilibrium is never reached because the bacteria create flows, inject energy and generate mechanical stresses. The resulting behaviour is then categorised under non-equilibrium thermodynamics. However, the tracers can still be considered to be diffusive, only that now there is an additional active diffusion term.

As such, the natural first factor to study when inserting tracers into swimmer baths is their diffusivity D. It turns out that it can be greatly enhanced compared to regular Brownian (ther-

mal) diffusivity D_0 [4]. In the early days of active matter research, the diffusivity was only ever observed to increase linearly with bacteria concentration n [5]. Using different experimental parameters, this was since then corrected to hold only at sufficiently low bacteria concentration where there is no collective motion [6]. An explanation for this effect is tracer advection in the swimmers' field that adds an active diffusion term D_A to D_0 [7]. This approach is also taken in this thesis although it strictly holds only at low concentrations.

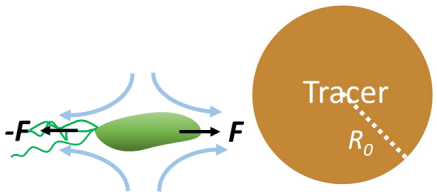

Figure 1.2: A bacteria with rotating flagella is modelled as a positive force dipole. The blue arrows indicate the surrounding mean flow field. Tracers, usually bigger than the swimmers, are modeled as spheres.

It is already known that the diffusivity of point-like tracers is enhanced in pusher suspensions as compared to puller suspensions [2]. But instead of further investigating the well studied dependence on swimmer density n, it is the radius R_0 of the tracers that we will focus on here. The pioneers of tracer diffusion research suggested that, at least for large particles ($R_0 > 4.5\,\mu\text{m}$), diffusion scales as $1/R_0$ [8]. They based this on an argument of friction - just as in passive fluids. More recently, a simplified theory and simulation predicted non-monotonic size behaviour [9] based on the interplay between active and passive diffusion. A separate experimental observation also showed a non-monotonic behaviour [10]. In this thesis we investigate the connection between non-linearities in the flow field and the diffusion as a function of tracer radius.

Understanding the effects of size on particle dynamics is important because tracers are used, for example, to gauge the activity of a fluid. If different particle sizes give rise to different diffusion constants, keeping everything else constant, fluids might be mischaracterized if one is careless about this effect. In nature, bacteria density and tracer size (e.g. nutrients) have surely co-evolved to an optimal condition but if one wants to exploit diffusion for technological purposes (such as drug delivery), one must find alternatives to the ever so slow process of trial and error. Finding a good model that could predict the size behaviour even at high bacterial densities is therefore indispensable. For this purpose, a reliable theory and efficient numerical simulations are needed.

If a theory of advection only takes into account fluid flow, an excellent way to simulate its effects would be to use the numerical lattice Boltzmann method (LBM) because it offers the possibility to simulate a comparatively large number of swimmers ($N > 10^6$). Just such an analytical theory has been developed in 1922 by the Swedish physicist Hilding Faxén [11] and the more recently developed lattice Boltzmann code LBSWIM from Lund [12] could easily be modified to include his correction.

The thesis therefore starts off with the theory of fluid flow, in particular Faxén's law that describes advection of a passive particle in laminar flow. Thereafter, the necessary theory for diffusion is introduced. Then, the microswimmer model is described. The method section reviews the lattice Boltzmann approach and compares it to a semi-analytical method. Results are presented for both non-interacting and interacting systems and their validity is discussed by making use of the results from the semi-analytical method.

2. Background and Theory

2.1 Fluid Flow at Low Reynolds Number

This thesis deals with swimmers and tracers suspended in a fluid. The swimmers self propel, exerting forces on the fluid, and thus stir up their surroundings. The other swimmers and the passive tracers are affected by the motion of the fluid and advect. To know the strength and direction of the velocity field $\boldsymbol{u}$ at a certain point and time, the Navier-Stokes (NS) equations can be solved. This set of differential equations is the equivalent of Newton's second law for fluid motion. In their most general form, they are quite involved and complicated enough so that it has not yet been proven whether they can always be solved in three dimensions [13]. In our domain, however, the so called Reynolds number, Re, is very small which leads to a simplification. By definition, $Re = \rho UL/\mu$, where ρ is the fluid density, μ is the dynamic viscosity, and L and U are the characteristic length and velocity. The Reynolds number is the ratio of inertial to viscous forces in a fluid. Inertial forces are due to the momentum of the fluid which increases with fluid density and velocity. Between regions of different velocities friction arises and turbulent flow may be the result. The fluid viscosity, however, inhibits turbulence such that fluids with low Re exhibit laminar flow. Swimming *E. coli* in water, for example, has $Re \approx 10^{-4}$ [14], low enough for the Navier-Stokes equations to simplify to the Stokes equations,

$$-\nabla P + \mu \Delta \boldsymbol{u} = 0 \tag{2.1a}$$

$$\nabla \cdot \boldsymbol{u} = 0, \tag{2.1b}$$

where $\boldsymbol{u}$ is the velocity of the fluid, P is the pressure, and μ is the dynamic viscosity. Fluid flow described by these equations is called Stokes flow or creeping flow. Apart from its applications to microorganisms, it also describes the flow of lava, which provides a useful macroscopic visualization of the creeping motion.

2.2 Faxén's law

Once the fluid velocity $\boldsymbol{u}$ is found, it is still not obvious how a particle would react to it. The Swedish physicist Hilding Faxén made this the topic of his dissertation published in 1922, considering spherical particles of radius R_0 [11]. Faxén's law thus states that the force $\boldsymbol{F}$ on such a suspended tracer is given by

$$\boldsymbol{F} = 6\pi\mu R_0 \Big(1 + \frac{R_0^2}{6}\Delta\Big)\boldsymbol{u} - 6\pi\mu R_0 \boldsymbol{U}, \tag{2.2}$$

where $\boldsymbol{U}$ is the tracer velocity and the first term is evaluated at the sphere's center. Without an external force, $\boldsymbol{F} = 0$, the equation can simply be solved for $\boldsymbol{U}$. Furthermore, the derivative term is commonly neglected when the velocity field does not change much over distances $\sim R_0$. In this case, one says that the tracer is simply advected, i.e. $\boldsymbol{U} = \boldsymbol{u}$. It is justified when R_0 is small. Experimental evidence confirms that there are indeed regimes in which the enhanced diffusion coefficient is independent of tracer size [15, 16, 17]. Other studies, however, detect a non monotonic behaviour of active diffusion with respect to tracer size [10]. One explanation for this relies simply on collisions between swimmers and tracers. The evidence for this is based on observations of individual collisions [18] and it could not yet be excluded that the extra term in Faxén's law also contributes to the behaviour.

2.3 Diffusion

Already in the previous section, the term diffusion was mentioned several times. Most generally, diffusion is the intermingling of substances by the natural movement of their particles [19]. The net flow of, for example, tracers from a region of high concentration to a region of low concentration results partly from random thermal motion. To begin with, we assume that the tracers are solely such random walkers and we will see later how things change when advection is also taken into account.

2.3.1 Mean Square Displacement and Brownian Motion

In our case, the tracers are equally distributed and there will be no net flow. However, the individual tracers will still move about and it is useful to introduce the mean square displacement (MSD), a measure of how much space each random walker explores in a given time. At time t, it is defined as

$$\mathrm{MSD} \equiv \frac{1}{N} \sum_{i=1}^{N} \left| \boldsymbol{x}^{(i)}(t) - \boldsymbol{x}^{(i)}(0) \right|^2, \tag{2.3}$$

where N is the total number of particles, $\boldsymbol{x}^{(i)}(0)$ is the starting position of the i^{th} particle, and $\boldsymbol{x}^{(i)}(t)$ is the position of the i^{th} particle at time t. In experiments or simulations, the particles can easily be tracked and the MSD computed. Often it is the diffusion constant D that is then plotted as a function of, for example, particle size. The diffusion constant is defined as a factor in the diffusion equation:

$$\frac{\partial \phi(\boldsymbol{r}, t)}{\partial t} = D \nabla^2 \phi(\boldsymbol{r}, t), \tag{2.4}$$

where $\phi(\boldsymbol{r}, t)$ is the density of the diffusing material. To derive the relation between D and MSD in the case of Brownian (random) motion, $\phi(\boldsymbol{r}, t)$ is replaced by the probability density function for a particle (this is the method originally used by Einstein). It is then easy to show

that the probability to find a particle at $\boldsymbol{x}$ is a Gaussian centered around the initial position $\boldsymbol{x}(0)$ [20]. From this it follows (not trivially) that in three dimensions

$$\mathrm{MSD} = 6Dt. \tag{2.5}$$

Einstein also obtained a relation between D and the atomic properties of matter. Named in his honour, the Stokes-Einstein relation reads

$$D = \frac{k_B T}{6\pi\mu R_0}, \tag{2.6}$$

where k_B is Boltzmann's constant and T is temperature.

2.3.2 Langevin Equation and Active Diffusion

Another method that leads to the same results was pioneered by the French physicist Langevin. The Langevin equation describes the motion of a Brownian particle immersed in a liquid at temperature T:

$$\frac{d\boldsymbol{x}}{dt} = -\frac{1}{\gamma}\frac{dV(\boldsymbol{x})}{d\boldsymbol{x}} + \boldsymbol{\eta}^T, \tag{2.7}$$

where $V(\boldsymbol{x})$ is a potential, γ is the particle friction coefficient and $\boldsymbol{\eta}^T$ is thermal noise characterized by $\langle \eta_\alpha^T(t)\eta_\beta^T(t')\rangle = 2D_T\delta_{\alpha\beta}\delta(t-t')$ and $D_T = \mu k_B T$ (α and β representing Cartesian components). This method is particularly useful because it allows us to add another type of noise, an active one, $\boldsymbol{\eta}^A$, that is due to the swimmers' velocity field. If this active noise is described by an Ornstein-Uhlenbeck process (a random walk with a tendency to move back towards its center [21]), its correlation function is given by $\langle \eta_\alpha^A(t)\eta_\beta^A(t')\rangle = D_A\delta_{\alpha\beta}\exp(-|t-t'|/\tau_a)/\tau_a$. Adding it to Eq. 2.7 the MSD can be computed:

$$\mathrm{MSD}(t) = 2D_T t + \frac{L^2}{\tau_a}[t - \tau_a(1 - e^{-t/\tau_a})], \tag{2.8}$$

where L is a characteristic distance along which the swimmers drag (on average) the particle [22]. The characteristic cross-over time τ_a corresponds to when tracers transit from an initially ballistic regime for $t \ll \tau_a$ to a diffusive regime with for $t \gg \tau_a$. This becomes clear when looking at an exemplary log-log plot of this equation as in Fig. 2.1 where $D_T = 0$. In this case, we find $D = D_A = L^2/6\tau_a$ because in the long time limit the MSD needs to approach Eq. 2.5.

2.4 Review of Experimental Results

We mentioned previously that the diffusion coefficient can vary non-monotonously with respect to tracer size [10]. Now that τ_a has been introduced, it is suitable to show the two figures

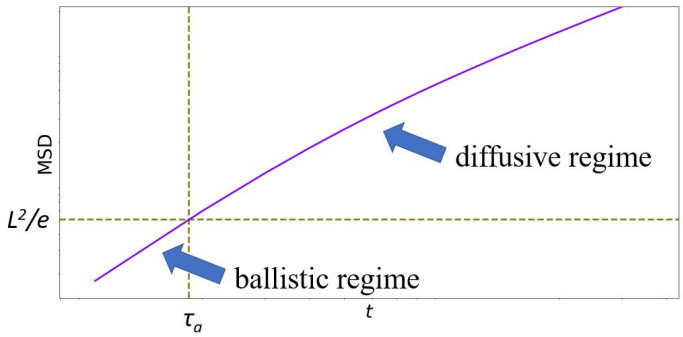

Figure 2.1: A log-log plot of Eq. 2.8 with arbitrary parameters L^2 and τ_a and $D_T = 0$.

from this paper, which our simulation tries to replicate. In experiments, one cannot avoid thermal diffusion, but its diffusion coefficient can be computed from Eq. 2.6 and then simply subtracted from the measured diffusion to obtain the active diffusion coefficient D_A. This is what the authors of ref. [10] have done, and the plot for D_A with respect to tracer diameter d is shown in Fig. 2.2 (a). Next to it, the plot for the cross-over time as a function of d is replicated. As we can see, D_A as a function of particle size is non-monotone for all tested bacterial concentrations c. The cross-over time τ_a (τ in the figure) increases monotonically with tracer diameter.

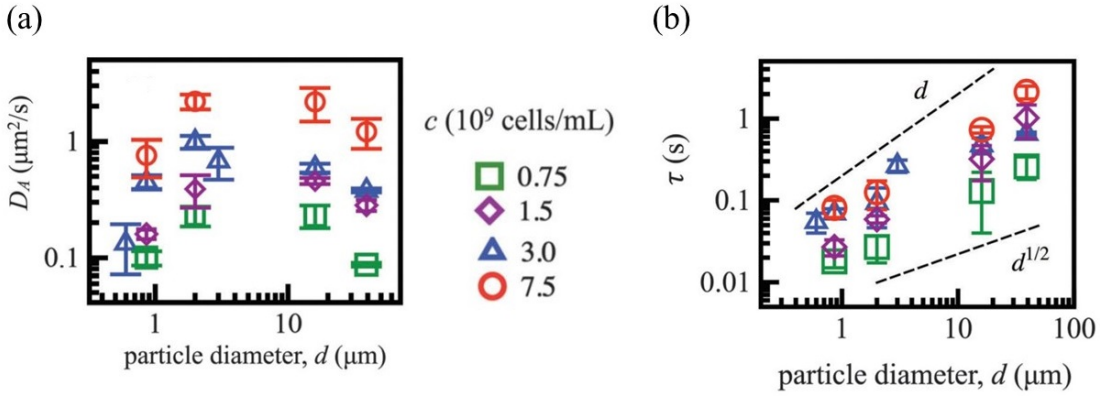

Figure 2.2: (a) Active particle diffusivities D_A versus particle diameter d at varying bacterial density c. (b) The crossover-time τ_a (in figure: τ) increases with d, scaling as approximately d^n, where $1/2 < n < 1$. Adapted from [10] with permission from The Royal Society of Chemistry.

2.5 Force Dipole Model

Many biological microswimmers self-propel by turning rear attached flagella counterclockwise, examples are the bacteria *Escherichia coli* and *Salmonella typhimurium* [14, 23]. The flagellar bundle traces out a helix with a contour length $\sim 10\,\mu\text{m}$ propelling the organism to speeds of up to $v_s = 20 - 60\ \mu\text{m/s}$. If one or more flagella changes rotational direction, the flagellar bundle disintegrates leading to a change in the direction $\boldsymbol{p}$ of the swimmer, resulting in a

run-and-tumble motion. Such swimmers are called pushers and they can be modeled as a force dipole characterized by two equal forces F a distance l apart (see Fig. 2.3). Pullers are similar only that their forces point inwards, resulting in entirely different interactions and collective motion [2]. Mechanically, this is achieved by "breaststrokes"; an example is the algae *Chlamydomonas*. Putting all the relevant parameters together, swimmers in a liquid are characterized by their dipole strength $\kappa = \pm Fl/\mu$, where μ is the dynamic viscosity of the liquid, and $\kappa < 0$ for pullers and $\kappa > 0$ for pushers. Because of how they reorient each other, pushers lead to large scale correlations, and pullers do not (see Fig. 2.3 for two-body interactions).

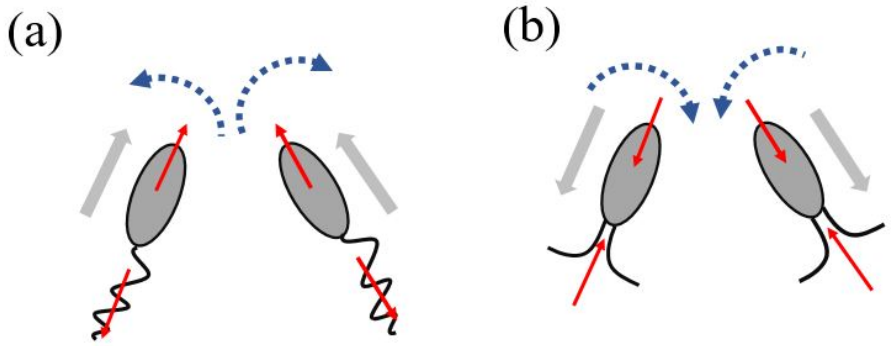

Figure 2.3: a) Bacteria such as *E. coli* are "pusher swimmers", whose flow field is modeled by a positive force-dipole (red arrows). Two converging pushers reorient each other (gray arrows indicate swimming direction, blue arrows reorientation), leading toward a configuration with parallel cells swimming side-by-side. b) The "breaststrokes" by pullers lead to a negative dipole field such that two diverging pullers reorient each other, leading toward a configuration of antiparallel cells, swimming away from each other.

2.6 Point Force Dipole Approximation

The lattice Boltzmann method uses the model of the force dipole to extrapolate the fluid behaviour at certain points. Another approach is based directly on the velocity field of a pusher in the point force dipole approximation:

$$\boldsymbol{u}(r) = \frac{P}{|\boldsymbol{r}|^2}[3(\hat{\boldsymbol{r}} \cdot \boldsymbol{d}') - 1]\hat{\boldsymbol{r}}, \quad \hat{\boldsymbol{r}} = \frac{\boldsymbol{r}}{|\boldsymbol{r}|}, \tag{2.9}$$

where $p = \kappa/8\pi$, $\boldsymbol{d}'$ is the unit vector in the swimming direction, and $\boldsymbol{r}$ is the distance vector relative to the dipole center. This can be used because the field of the point dipole is equivalent to that of an extended dipole at large separations. It was experimentally shown that many bacterial swimmers' flow fields can be well described by this equation, as can be seen in Fig. 2.4. From this, it could be extrapolated that $p = 31.8\,\mu\mathrm{m}^3/\mathrm{s}$ (for *E. coli* swimming at an average speed of $v_s = 22\,\mu\mathrm{m/s}$).

This point dipole approximation is used in a simulation that assumes low swimmer density n. How to find an approximate diffusion coefficient of one tracer in such suspensions will

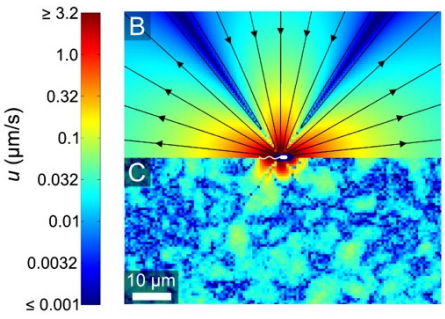

Figure 2.4: Average flow field created by a single freely swimming *E. coli* bacterium. Streamlines indicate the local direction of flow, and the color indicates flow speed magnitudes. (B) Best-fit force dipole flow to the experimental flow field. (C) The difference between the best-fit dipole model and the measured field. Adapted from [24].

take up the remainder of this section. Without loss of generality, swimmers will be moving a distance λ with speed U_s from left to right horizontally. This is the average distance a swimmer would stay on a straight path before tumbling in a random direction. During this time, a tracer is advected in the flow field, which is referred to as scattering. One such event results in an average tracer displacement of Δ that depends on the initial distance to the swimmer. Thus, two relevant parameters are the initial perpendicular distance to the particle, $a > 0$, and the relative distance b between the start of the trajectory and the point of initial closest approach (see Fig. 2.5).

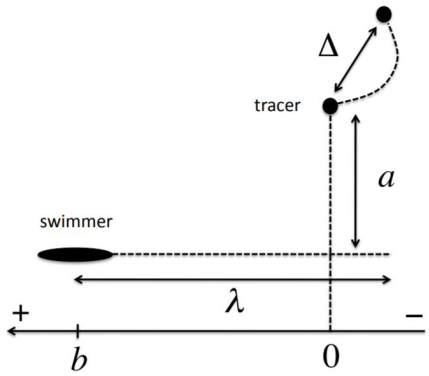

Figure 2.5: Sketch of a scattering event between a swimmer and a tracer: a is the shortest distance between the swimmer's trajectory and the tracer's original position, and b is the distance between the swimmer's initial position and the point of closest approach. The total distance travelled by the swimmer is λ and Δ denotes the tracer's net displacement. Reproduced from [25] with permission from The Royal Society of Chemistry.

Following Morozov and Marenduzzo [25] (who in turn followed Lin et al. [26] and who themselves were inspired by Einstein's theory of Brownian motion [27]), the diffusion coefficient can be found by considering M such events with different a and b that happen during time t. If the tracer starts at the origin, its position after all these encounters is given by

$$\boldsymbol{r}(t) = \sum_{k=1}^{M} \Delta(a_k, b_k)\hat{\boldsymbol{r}}_k \tag{2.10}$$

where $\hat{\boldsymbol{r}}_k$ is a direction vector. The swimmers are assumed identical and non-interacting, and the scatterings are taken to be statistically independent. Their swimming directions are assumed isotropically distributed. Also, a_k and b_k are identically distributed for each encounter. In this case, the mean square displacement is given by

$$\mathrm{MSD} = \langle |\boldsymbol{r}(t)|^2 \rangle = M(t)\langle \Delta^2(a,b)\rangle_{a,b}, \tag{2.11}$$

where $\langle ... \rangle_{a,b}$ is the average over all possible scattering parameters a and b. This average can be written as an integral and the number of encounters can be found easily: during time t, each swimmer changes its directions $U_s t/\lambda$ times, so that $M(t)$ must be proportional to this. In 3D, the number of swimmers in a ring of radius r and thickness dr around the target particle is $4\pi n r^2 dr$, where n is the number density of swimmers. Equation 2.11 can therefore be written as

$$\mathrm{MSD} = n\frac{U_s t}{\lambda}\int_0^\infty da \int_{-\infty}^{\infty} db 2\pi a \Delta^2(a,b) = 6Dt, \tag{2.12}$$

where the last equality comes from Eq. 2.5 and the integral has been changed from spherical coordinates to a and b coordinates. In an approximation, the infinite boundaries can be replaced by values of a and b where the tracer displacement becomes negligible. To aid evaluation of the integral we introduce $\sigma = \sqrt{p/U}$ and substitute $a = \sigma e^{\xi}, b = \lambda\chi$, and $\Delta = \sigma\tilde{\Delta}$. We then conclude that

$$D = AnU\sigma^4, \tag{2.13}$$

where

$$A = \frac{\pi}{3}\int_{-\infty}^{\infty} d\xi \int_{-\infty}^{\infty} d\chi e^{2\xi}\tilde{\Delta}^2(\xi,\chi). \tag{2.14}$$

This integral can be computed by numerically evaluating the integrand for a sufficient set of scattering parameters a and b. The errors at the grid boundaries are small since the integrand is very small there. Note, that even though we included n here, the derivation does not hold for an arbitrary many-body simulation because it assumes very dilute suspensions. This was important to be able to consider the encounters independent of each other (Eq. 2.11).

3. Method

The last part of the theory chapter leads directly to the method chapter where we first describe how the correction to Faxén's law can be simulated in the point force dipole approximation. We then explain how this can be useful to judge the validity of our many body simulation before the latter is described in detail.

3.1 The Point Dipole Method

The point force dipole approximation is easy to implement because we know the velocity field, Eq. 2.9. If the tracer is simply advected, i.e. $\boldsymbol{U}(r) = \boldsymbol{u}(r)$, its total displacement Δ is found by updating its path in the fluid as the swimmer moves along (in discrete time steps) and then subtracting final from initial position. Including the Faxén correction is only slightly more complicated because the derivative is known analytically (tedious by hand but a Mathematica script can help to find it):

$$\Delta\boldsymbol{u}(r) = \frac{6p}{r^4}[(1 - 5(\hat{\boldsymbol{r}} \cdot \boldsymbol{d}')^2)\hat{\boldsymbol{r}} + 2(\hat{\boldsymbol{r}} \cdot \boldsymbol{d}')\boldsymbol{d}'], \tag{3.1}$$

which has a component in the swimmer's direction $\boldsymbol{d}'$. At each time step, the tracer velocity is then found via Eq. 2.2 and its position is updated. Then, the swimmer continues and we start over. The first reason why this is helpful for the many body simulation is because it allows us to easily check if a cut-off radius might be necessary: it might be essential to not update the tracer when it comes to close to the swimmer. If needed, this simple simulation should also help in choosing what this radius should be.

The second benefit is comparing the analytical derivative with the numerical approximation. A discrete Laplacian in 2D is obtained by considering the velocity field at the point of interest and four points around it:

$$\Delta f(x, y) \approx \frac{f(x - h, y) + f(x + h, y) + f(x, y - h) + f(x, y + h) - 4f(x, y)}{h^2}, \tag{3.2}$$

where h is an appropriately small distance. In 3D the approximation is almost the same, the initial pattern just has to be extended to include z and the number 4 has to be replaced with 6. To increase accuracy, it is also possible to include not just immediate lattice neighbours [28]. Comparing the results of the discrete method with the analytical one (using Eq. 3.1), it is possible to choose the appropriate h, decide whether the higher order approximation is called for, or if the numerical approximation is valid at all.

In units where the length is measured in micrometers and time in seconds, we set $p = 32$, a time step of 0.001, a swimmer velocity of 22 and $\lambda = 10$. This is used to simulate *E. coli* [24].

3.2 The Lattice Boltzmann Method

To be efficient, a many body simulation must rely on different methods than the ones discussed previously. An unique way to solve for the NS fluid flow is the lattice Boltzmann method which is described in the remaining part of this chapter.

3.2.1 An intuitive approach to the LBM

The NS equations can be quite difficult to solve analytically but there are many computational alternatives. Traditional computational fluid dynamics (CFD) would start by discretizing the Navier-Stokes equations and then solving them, numerically, for the desired boundary conditions. This approach, however, can be quite tedious to implement and has difficulties dealing with complex boundaries [29].

In the early 1990s, the lattice Boltzmann method was developed starting from an entirely different perspective: consider the fluid to be an ideal gas that has no macroscopic velocity and is in thermal equilibrium at temperature T. In three dimension, its molecules' thermal velocities $\boldsymbol{v}_c$ will be distributed according to the Boltzmann distribution:

$$P(\boldsymbol{v}_c) = \left(\frac{m}{2\pi k_B T}\right)^{3/2} e^{-m|\boldsymbol{v}_c|^2/2k_B T}, \tag{3.3}$$

where m is the molecular mass. Integrating this function over a range of velocities gives the probability of a molecule to have a velocity in that range; the equation is simply the Boltzmann factor $e^{-E/kT}$, with $E = \frac{1}{2}m|\boldsymbol{v}_c|^2$ and a normalization factor. When the function is integrated over the whole range, the normalization factor guarantees that this integral equals unity. In the LBM, we consider collections of molecules that move together in the same direction. As such, it is a mesoscopic method and not a microscopic one (which would treat every molecule individually) nor a continuous one (solving for the continuous vector field directly in traditional CFD). The movement of each collection of molecules can be solved in parallel and allows for great computational efficiency.

The *lattice* Boltzmann method discretizes both space and time, allowing only certain velocities. This thesis uses a D3Q15 algorithm, meaning that space is discretized in three dimensions with 15 allowed displacement vectors $\boldsymbol{\xi}_i$ (zero being one of them, see Fig. 3.1). One can imagine to use even more velocity components for better accuracy but this would entail a computational cost.

At each time step t, each lattice site $\boldsymbol{x}$ is associated to 15 different probability distributions $f_i(\boldsymbol{x}, t)$, the densities that are moving in the direction $\boldsymbol{\xi}_i$. The total density ρ can be computed from these, as well as the macroscopic velocity $\boldsymbol{u}$. Different f_i at the same lattice point should eventually come to equilibrium and an obvious next step would be to set all the 15 densities equal to these equilibrium values. In real ideal gases, collisions among the molecules would

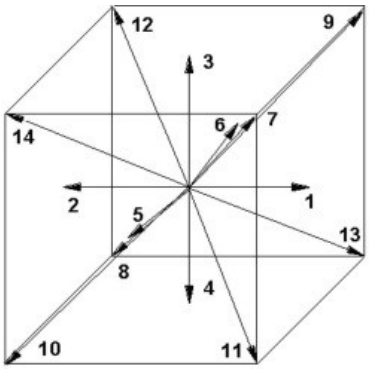

Figure 3.1: The three-dimensional lattice model D3Q15.

bring them closer to this thermal equilibrium. Because the time scale for reaching equilibrium need not be identical to the simulation time step, it is more general to approach equilibrium:

$$f_i^{\text{new}} = f_i^{\text{old}} - \frac{1}{\tau}(f_i^{\text{old}} - f_i^{\text{eq}}), \tag{3.4}$$

where $\tau = 1$ would mean relaxation to equilibrium at each time step. Once this is done for each lattice site, all the moving molecules are put into adjacent or diagonal lattice sites by copying the appropriate values of f_i. This last step is called streaming, while the first steps are referred to as collisions. Alternating them leads to the same large-scale flow behavior as the NS equations would predict (at least if certain conditions are met, such as fluid flowing slower than the speed of sound). It is easy to include boundaries or obstacles by making the fluid that would flow into them in each streaming step reflect in the opposite direction instead.

3.2.2 The Boltzmann Equation

The *lattice* Boltzmann equation governs how the discrete distribution function evolves over time. To motivate it, based on the intuition gained in the previous sub-section, we start from the continuous case, where we have $f(\boldsymbol{x}, \boldsymbol{\xi}, t)$ instead of $f_i(\boldsymbol{x}, t)$ where $\boldsymbol{\xi}$ is velocity, following the notation in [30]. Integration yields relevant macroscopic quantities such as density ρ:

$$\rho = \int f(\boldsymbol{x}, \boldsymbol{\xi}, t) d\xi \tag{3.5}$$

Momentum density $\rho \boldsymbol{u}$, energy density $E\boldsymbol{u}$, and other quantities can be computed from other moments of f (integrals of f, weighted with some function of ξ).

The redistribution of the fluid density (the streaming step) is done after the collision step that equilibriates the different f's. We therefore write

$$\frac{df}{dt} = \Omega(f), \tag{3.6}$$

omitting to explicitly write the dependencies $(\boldsymbol{x}, \boldsymbol{\xi}, t)$. The source term $\Omega(f)$ is called the collision operator. Since all the dependencies are functions of t we can rewrite this as

$$\frac{\partial f}{\partial t} + \xi_\beta \frac{\partial f}{\partial x_\beta} + \frac{F_\beta}{\rho} \frac{\partial f}{\partial \xi_\beta} = \Omega(f), \tag{3.7}$$

where $\frac{dx_\beta}{dt} = \xi_\beta$, and, from Newton's second law, $\frac{d\xi_\beta}{dt} = F_\beta/\rho$. The index notation for vectors follows the Einstein summation convention. The form of $\Omega(f)$ may not be chosen arbitrarily because mass, momentum and translational energy must be conserved. An elegant choice is the Bhatnagar–Gross–Krook (BGK) collision operator:

$$\Omega(f) = -\frac{1}{\tau}(f - f^{eq}), \tag{3.8}$$

where f^{eq} is defined shortly. Equation 3.7 is called the Boltzmann equation and it has a simple interpretation as an advection equation: the first two terms on the left represent the advection of the distribution function f with the velocity $\boldsymbol{\xi}$ of its molecules. The third term accounts for forces affecting this velocity, and the source term on the right hand side represents the local redistribution due to collisions.

The function for f^{eq} in three dimensions can be uniquely determined from the average fluid velocity $\boldsymbol{u}$ and the density ρ:

$$f^{eq}(\rho, \boldsymbol{u}, T, \boldsymbol{\xi}) = \rho\Big(\frac{m}{2\pi k_B T}\Big)^{3/2} e^{-m|\boldsymbol{\xi}-\boldsymbol{u}|^2/2k_B T}. \tag{3.9}$$

Maxwell arrived to this conclusion by demanding that f^{eq} has the same moments of density and energy as f. The fact that it is *unique* was later shown by Boltzmann who used more fundamental statistical mechanics. This is why f^{eq} is called the *Maxwell-Boltzmann distribution*. Equation 3.3, which gave us some intuition, is actually not the starting point for many textbooks but it is here that they first introduce this well known distribution.

3.2.3 The Lattice Boltzmann Equation

We will now simply state the most important results from discretization. To derive these rigorously, some pages of math using Hermite polynomials are required, while their implementation is quite simple. The velocity is no longer continuous so we again write $\boldsymbol{\xi}_i$, i going from 0 to 15 in our case. For convenience we introduce a new particle velocity $\boldsymbol{c}_i = \boldsymbol{\xi}_i/\sqrt{3}$. With this, the discrete equilibrium distribution reads

$$f_i^{eq} = w_i \rho\Big(1 + \frac{\boldsymbol{u}\cdot\boldsymbol{c}_i}{c_s^2} + \frac{(\boldsymbol{u}\cdot\boldsymbol{c}_i)^2}{2c_s^4} - \frac{\boldsymbol{u}\cdot\boldsymbol{u}}{2c_s^2}\Big), \tag{3.10}$$

where c_s is a constant (actually, it can be interpreted as the speed of sound). The weights w_i, in our specific case, are $2/9$ for $i = 0$, $1/9$ for going to the close sides of the square (i from 1

to 6) and $1/72$ for the remaining velocities. Maxwell's approach to find Eq. 3.9 must also hold in the discrete case such that

$$\rho = \sum_i f_i = \sum_i f_i^{eq}, \quad \rho \boldsymbol{u} = \sum_i \boldsymbol{c}_i f_i = \sum_i \boldsymbol{c}_i f_i^{eq}. \tag{3.11}$$

It can be shown that similar relations also hold for the third moment (energy), but not for higher ones.

The discrete-velocity Boltzmann equation is simply

$$\partial_t f_i + c_i \nabla f_i + [\boldsymbol{F} \cdot f]_i = \Omega(f_i) \tag{3.12}$$

and a first-order discretization, ignoring the force term, gives the lattice Boltzmann equation (LBE):

$$f_i(\boldsymbol{x} + \boldsymbol{c}_i \Delta t, t + \Delta t) = f_i(\boldsymbol{x}, t) + \Delta t \Omega_i(\boldsymbol{x}, t). \tag{3.13}$$

Again, the most common collision operator is the BGK operator:

$$\Omega_i(f) = -\frac{f_i - f_i^{eq}}{\tau} \Delta t. \tag{3.14}$$

Putting things together and writing $f_i^\star(\boldsymbol{x}, t)$ for the distribution function after collisions, we obtain

$$f_i^\star(\boldsymbol{x}, t) = f_i(\boldsymbol{x}, t)\Big(1 - \frac{\Delta t}{\tau}\Big) + f_i^{eq}(\boldsymbol{x}, t)\frac{\Delta t}{\tau}. \tag{3.15}$$

The streaming step is then simply

$$f_i(\boldsymbol{x} + \boldsymbol{c}_i \Delta t, t + \Delta t) = f_i^\star(\boldsymbol{x}, t). \tag{3.16}$$

3.2.4 The Lattice Boltzmann Algorithm

Already discussed in subsection 3.2.1, we now summarize the LBM algorithm:

- The densities and velocites are computed from $f_i(\boldsymbol{x}, t)$ via the sums in Eq. 3.11.
- The equilibrium distribution $f_i^{eq}(\boldsymbol{x}, t)$ is obtained from Eq. 3.10.
- The molecules *collide* via Eq. 3.15.
- *Streaming* is performed as shown in Eq. 3.16.
- The time step is increased by setting t to $t + \Delta t$, and all the steps are repeated until convergence or the final time step.

3.2.5 Force Interpolation

So far, we focused on the force free discrete results but they can easily be extended to include a force term [31]. This is important since the swimmers are modeled as extended force dipoles. They move off lattice onto which their forces need to be interpolated. In the continuous case, the swimmers' force density would be represented by

$$\boldsymbol{F}(\boldsymbol{r}) = \int \boldsymbol{f}(\boldsymbol{R})\delta(\boldsymbol{r} - \boldsymbol{R})d\lambda, \tag{3.17}$$

where the force is localized to some manifold with measure $d\lambda$. Clearly, the integral needs to be replaced by a sum in the discrete case, but then also the δ needs to be adapted:

$$\boldsymbol{F}(\boldsymbol{r}) = \sum_a \boldsymbol{f}(\boldsymbol{R}_a)\delta^p(\boldsymbol{r} - \boldsymbol{R}_a). \tag{3.18}$$

Following Nash et al. [31], a regularized Dirac δ function δ^p is used:

$$\delta^p(\boldsymbol{r}) = \frac{1}{h^3} f(\frac{x}{h}) f(\frac{y}{h}) f(\frac{z}{h}), \tag{3.19}$$

where $h = \Delta x = \Delta y = \Delta z$ is the lattice spacing and $f(r)$ is given by

$$f(r) = \begin{cases} \frac{3-2|r|+\sqrt{1+4|r|-4r^2}}{8}, & |r| \leq 1, \\ \frac{5-2|r|-\sqrt{-7+12|r|-4r^2}}{8}, & 1 \leq |r| \leq 2, \\ 0, & |r| \geq 2. \end{cases} \tag{3.20}$$

Each swimmer has two force poles that are independently sampled onto the grid using Eq. 3.18 where the forced LBE is solved. Similarly, velocities from the lattice can be interpolated onto the swimmers and tracers.

To implement the whole of Faxén's law, we also need to compute the Laplacian of the velocities. This is simply done via the three dimensional version of Eq. 3.2 using $h = 0.5$.

3.2.6 Simulation and Units

Implementing such an LBM algorithm and plotting its velocity field results in figures or videos that resemble fluid flows that seem natural, i.e. governed by the Navier-Stokes equation. That this is indeed the case can even be proven using the Chapman-Enskog analysis [30].

The in-house software LBSWIM is such an implementation in FORTRAN. This program allows for two scenarios: in the first one the swimmers just run and tumble without being affected by each others' velocity fields. This is a good approximation at low enough swimmer densities. Above a certain threshold density, swimmer-swimmer interactions should, however, be taken into account. In the second scenario, therefore, each swimmer i moves according to

$$\dot{r}_i^\alpha = v_s p_i^\alpha + U^\alpha(\boldsymbol{r}_i), \quad \dot{p}_i^\alpha = \mathbb{P}_i^{\alpha\beta}\nabla_i^\gamma u^\beta(\boldsymbol{r}_i)p_i^\gamma, \tag{3.21}$$

where Greek indices denote Cartesian coordinates, $\mathbb{P}_i^{\alpha\beta} = \delta_{\alpha\beta}p_i^\alpha p_i^\beta$, $\boldsymbol{u}(\boldsymbol{r}_i)$ is the fluid velocity at the position of swimmer i, and v_s is the swimming speed [2]. In both scenarios the swimmers' orientations are randomized with average tumbling frequency λ_t.

The units used in the LBM are all with respect to the unit cell in the lattice and the time step of the simulation. As such, we have $v_s = 10^{-3}$, $F = 1.57 \cdot 10^{-3}$, $l = 1$, $\lambda_t = 2 \cdot 10^{-4}$ and $\tau = 1$. From the dipole strength κ we define the non-dimensionalized one $\kappa_n = \kappa/(l^2 v_s) = F/(\mu l v_s)$. Plugging in the aforementioned values, $\kappa_n \approx 9.4$. For *E. coli*, $v_s = 22\,\mu\mathrm{m/s}$, $F = 0.42\,\mathrm{pN}$ and $l = 1.9\,\mu\mathrm{m}$ [24] in water we would have $\kappa_n = 11.2$ and the parameters in the LBM were chosen to approximate this value.

It is common to represent all units with respect to the parameters of the simulation. Since the tracer radius is in units of length and $l = 1$, this case is trivial. The SI units of the diffusion constant are $[\mathrm{m}^2\,\mathrm{s}^{-1}]$ such that D will be given as a multiple of $lv_s = 10^{-3}$. Similarly, any parameter that has unit of time (such as τ_a) is given as a multiple of $l/v_s = 1000$.

3.3 Contributions

As mentioned, the LBM simulation LBSWIM only needed minor modifications to include the Faxén correction. However, all the scripts for the data analysis (for example fitting the MSD) were python coded from scratch for this thesis. The simulation for the point dipole approximation described in section 2.6 was also written entirely anew in python and translated to C++ to reduce running time. The python version of the latter is attached in the appendix.

4. Results

4.1 Point Dipole Results

Before implementing the numerical integral described in section 2.6, we simply plot some tracer trajectories from arbitrary parameters: $a = 5$, $b = 20$, $c = 1$ and $\lambda = 40$ (remember, in the point dipole method, all units of length are in μm). The results can be seen in Fig. 4.1. The point like tracer behaves as expected; had we chosen an even bigger λ (and keeping $b = \lambda/2$), the loop would have almost closed on itself. This is a well known result and it is also not surprising that the trajectory of a tracer with a small radius ($R_0 = 2$) does not deviate much from the point tracer's. It is unforeseen, however, that a tracer with $R_0 = 8$ gives such different paths, and that they differ so much depending on whether an analytical or discrete $\Delta \boldsymbol{u}$ is used (in the following, we refer to this as the analytical or the discrete method). Not shown in the figure is that we also implemented a higher order numerical approximation and used various h. We found that the approximation given above (Eq. 3.2) was good enough and that below $h = 0.5$ the trajectory did not change anymore. While the trajectories differ qualitatively, they all end relatively close to each other whence it is not obvious that the diffusion coefficient would be much affected. To learn more about this, we need to actually compute the numerical integral A, see Eqs. 2.13 and 2.14.

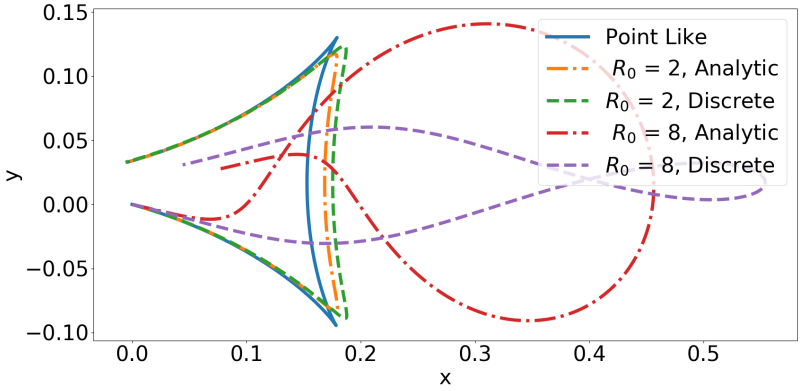

Figure 4.1: The trajectories of different tracers with parameters $a = 5$, $b = 20$, $c = 1$ and $\lambda = 40$.

To do so, the tracer's position is initially only updated whenever its distance to the swimmer is bigger than $c = 1$ because then effects other than the dipole field would also play a role. For the numerical integral to converge, the grid has to extend from -30 to 30 in both directions, with a grid spacing of 0.05. A point like particle then gives $A = 3.72$ and the integrand is focused around the origin with a slight shift to the right as can be seen in Fig. 4.2. The asymmetry arises from the fact that the swimmer does not affect the tracer when too close ($c = 1$) and it looks

qualitatively like the results in [25] (but A is slightly lower compared to 3.75). In this paper, a many-particle simulation was also performed by overlaying many dipole fields. Because the tracer would then not be able to complete its one-dipole path, A is reduced to 3.6. Since its implementation and running time are rather long, this part is not reproduced here.

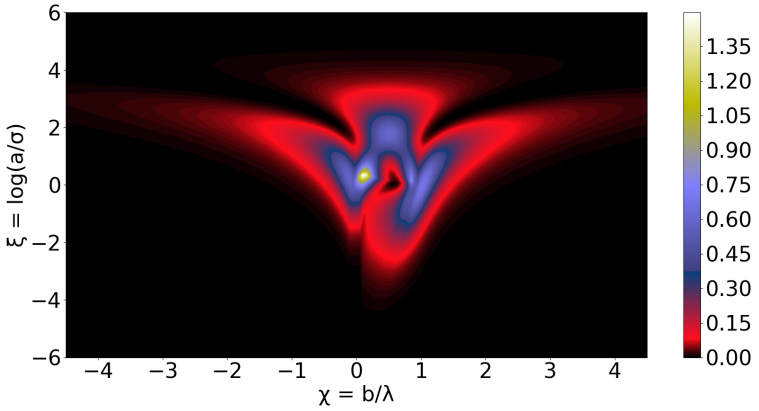

Figure 4.2: The integrand of Eq. 2.14 for a point like tracer (result similar to Fig. 2 in [25]).

To extend from this known result, we now simply include the extra Faxén term into the update of a tracer with $R_0 = 4$. Again, the Laplace operator needed for this is done both discretely and analytically. In the first case we find $A = 152$ and in the second case $A = 151$. The integrands are shown in Figs. 4.3 (a) and 4.3 (b) from which we see that the analytical method gives a much more coarse grained result. This can be explained by considering that the discrete method necessarily takes into account neighbouring points and such softens the edges in the differentiated vector field. It is also evident that a much smaller set of initial conditions a and b now dominates the contribution to A as compared to the point tracer (Fig. 4.2). Here, the integrand reaches values that are a factor of ~ 200 larger than in the previous case. These values, however, are partly so that the swimmer would be found inside the tracer radius showing the need to introduce an R_0-dependent cutoff radius to the swimmer-tracer interactions.

To further investigate the point dipole model, we plotted the paths that gave rise to the biggest discrete and analytic integrand value (starting configuration $a = 2.97, b = 2.00$ and $a = 3.28, b = 3.00$, respectively). They are, together with the $R_0 = 0$ path and the swimmer's path, shown in Figs. 4.3 (c) and 4.3 (d). The trajectory of the point tracer is again as expected, but for $R_0 = 8$ it is remarkable that the two different methods give such similar A. The exact paths differ considerably for the same starting conditions, but apparently this evens out. Also, the jumps in the analytic method manifest themselves in the coarse grained integrand we have seen before. These unphysical jumps are another reason of why a radius dependent cut-off radius should be introduced.

Next, A is computed for different R_0 and varying c. This is interesting because for simplicity we had previously allowed the swimmers to penetrate the tracers, nonphysical but not theoretically inconceivable since there are no excluded volume interactions between the swimmer and the tracer. The result can be seen in Fig. 4.4 where black circumferences mark the cases when the cut-off radius was set to the tracer radius, effectively prohibiting swimmer-tracer overlaps

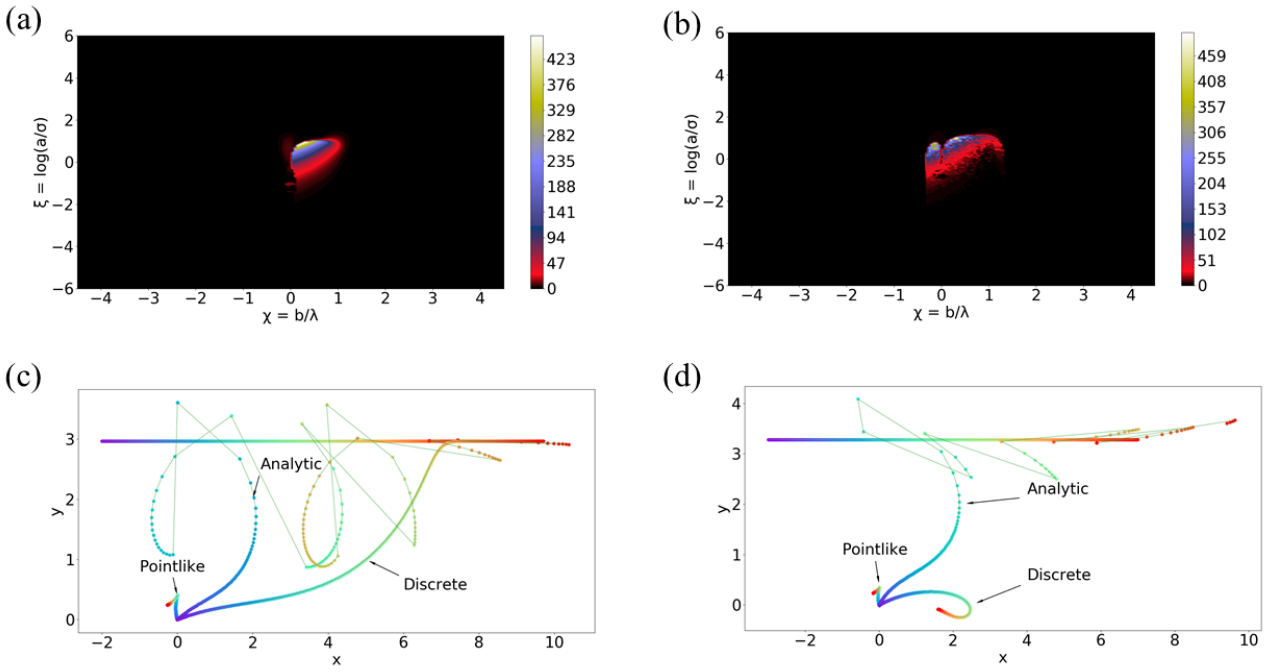

Figure 4.3: (a) The integrand of A when the discrete approximation of the derivative in Faxén's law is included. (b) The same, but with the analytic derivative. (c) The horizontal trajectory of a swimmer and its effect on a point like tracer and a tracer of $R_0 = 8$ with cut-off radius $c = 1$ (the derivative term evaluated analytically and discretely). These parameters ($a = 2.97, b = 2.00$) give rise to the biggest integrand in the discrete case. (d) The same, but with parameters ($a = 3.28, b = 3.00$) that give rise to the biggest integrand in the analytic case.

(only for $R_0 = 0$ this is not possible). Of course, at such distances near field effects would be relevant, and so would be collisions. To conclude something about the LBM, this is useful nonetheless. All but the $R_0 = 0$ graph "converge" the latest when $c = R_0$, and many already at $c = R_0 - 1$ or less (at this point, we note that the butterfly shape of the integrand [see Fig. 4.2] is also recovered). Before that, A, and therefore the diffusion constant, starts off two magnitudes higher than in the point like case with $c = 1$. These high values, however, drop steeply with increasing c until the aforementioned convergence. This shows that a too small cut-off radius gives a large overestimation of A.

If we chose $c = R_0$, we would observe D as a decreasing function of R_0. Choosing c for each R_0 separately, say the first one for which the graph has reasonably plateaued, we could even find a non-monotonic behaviour. This, of course, would be rather artificial and unsatisfactory as an explanation. Much more reasonable is the decreasing behavior from the Faxén correction and that some other effect explains the non-monotone function. Another conclusion from this plot is that the numerical Laplacian can differ substantially from the analytical reference value. Nonetheless, these two approach each other following the general convergence. Choosing a suitable c, therefore, can successfully remedy both an overestimation of D and errors from the numerical approximation. With this in mind, it is now time to turn to the LBM.

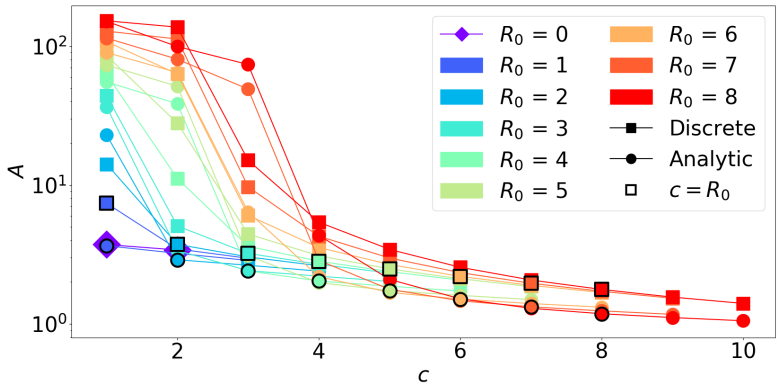

Figure 4.4: A as a function of the cut-off radius c. For all but $R_0 = 0$ a convergence is reached at latest when $c = R_0$ (black circumferences).

4.2 Lattice Boltzmann Results

The swimmers and tracers for the Lattice Boltzmann simulations were placed in a square box of side length 100. 100000 tracers were placed in this box together with a varying number of swimmers: $N = 1000, 2000, 4000, 7000, 10000, 20000, 40000, 70000, 100000$. The tracer radius was then varied from 0 to 5.5. Each parameter combination was run five times with different random seeds.

4.2.1 Non-Interacting Swimmers

Because the relevant experiments were all done at such densities that interactions between swimmers could be ignored, we begin by studying the case of non-interacting microswimmers, where swimmer-swimmer correlations are absent. An exemplary fit of Eq. 2.8 to the MSD in this case can be seen in Fig. 4.5. Most other fits are much better than the one below, but the general trend is that bigger radii lead to worse fits whereas more swimmers lead to better fits. This is also the first indication of what becomes evident later on: the simulation seems to run into problems at higher radius ($R_0 > 3.5$).

From these fits, D, τ_a and L^2 can be extracted. Doing this for all five random seeds, we take the standard deviation of the results as error bars. The result of this can be seen in Fig. 4.6. The diffusion constant with respect to swimmer density n (Fig. 4.6 (a)) is expected to behave linearly in the case of non-correlated swimmers. A rigorous argument for this would follow the lines that led up to Eq. 2.13, where D was shown to be proportional to the swimmer number density n. Here, however, it seemed necessary to fit the curves for the two highest radii with a second order polynomial. This is important to note for it will be used later on.

Maybe the most important figure is the diffusion constant as a function of tracer radius, Fig. 4.6 (b). Up to $R_0 = 2.5$, D declines slowly before increasing rather rapidly. Here, D as a function of R_0 is convex. For larger radii, the diffusion constant also varies more and more between different densities. Again, especially the behaviour for $R_0 > 3.5$ might need a special explanation.

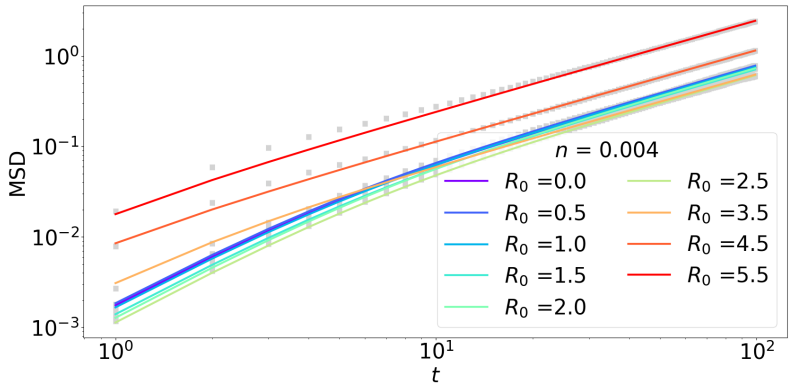

Figure 4.5: Equation 2.8 fitted to the MSD for $n = 0.004$ (the dots are simulation results, and the fits are shown in color).

The cross-over time τ_a as a function of swimmer density stays nearly constant (not explicitly shown here) whereas it clearly differs between radii. To focus on the latter, $\tau_a/\tau_a(0)$ is plotted against swimmer radius in Fig. 4.6 (c). In the first half, it increases with R_0 and the difference between the point tracer and the maximum at $R_0 = 2$ is $\sim 40\,\%$ after which it declines steeply. Already at $R_0 = 3.5$, the function has somewhat converged.

To determine how much L^2 depends on R_0, we plot $L^2(R_0)/L^2(0)$, see Fig. 4.6 (d). In the first half, the function stays nearly constant. In conjunction with Fig. 4.6 (c) (τ_a increases) this means that the crossover point in the MSD experiences a shift to the right (see Fig. 2.1). After this, the function drops sharply but, unlike $\tau_a(R_0)/\tau_a(0)$, it does not plateau.

4.2.2 Interacting Swimmers

When the swimmer-swimmer interactions are turned on via Eqs. 3.21, the difference between pushers and pullers becomes evident. The typical way to represent this is to normalize the resulting diffusion constants with respect to the non-interacting diffusion constants. Explicitly, this is done by dividing each value by the corresponding value of the fit in the non-interacting case. Here, the previously mentioned non-linear fit is important to avoid noisy graphs. An example of how pushers and pullers compare on the same scale is given in Fig. 4.7 (a) for $R_0 = 0$. This is a known result and confirms the functioning of our code [2]. Its interpretation is clear: above a certain density, interacting pusher suspensions enhance the diffusion of point-like tracers while puller suspensions diminish it. The explanation for this is simply the hydrodynamic interactions between swimmers (see section 2.5, specifically Fig. 2.3).

For the remaining radii, the pusher and puller graphs are separated, see Figs. 4.7(b) and 4.7 (c). Below $n = 0.001$, the interactions do not yet change the behaviour as compared to without them. Above that, for any radius, D is enhanced by pushers and diminished by pullers. For both pushers and pullers, however, the effect is non-monotone as a function of R_0.

To ascertain how exactly R_0 affects the diffusivity in these suspensions, we take an exemplary density, $n = 0.07$, and plot $D(R_0)/D_{free}(R_0)$ normalized with $D(0)/D_{free}(0)$, see

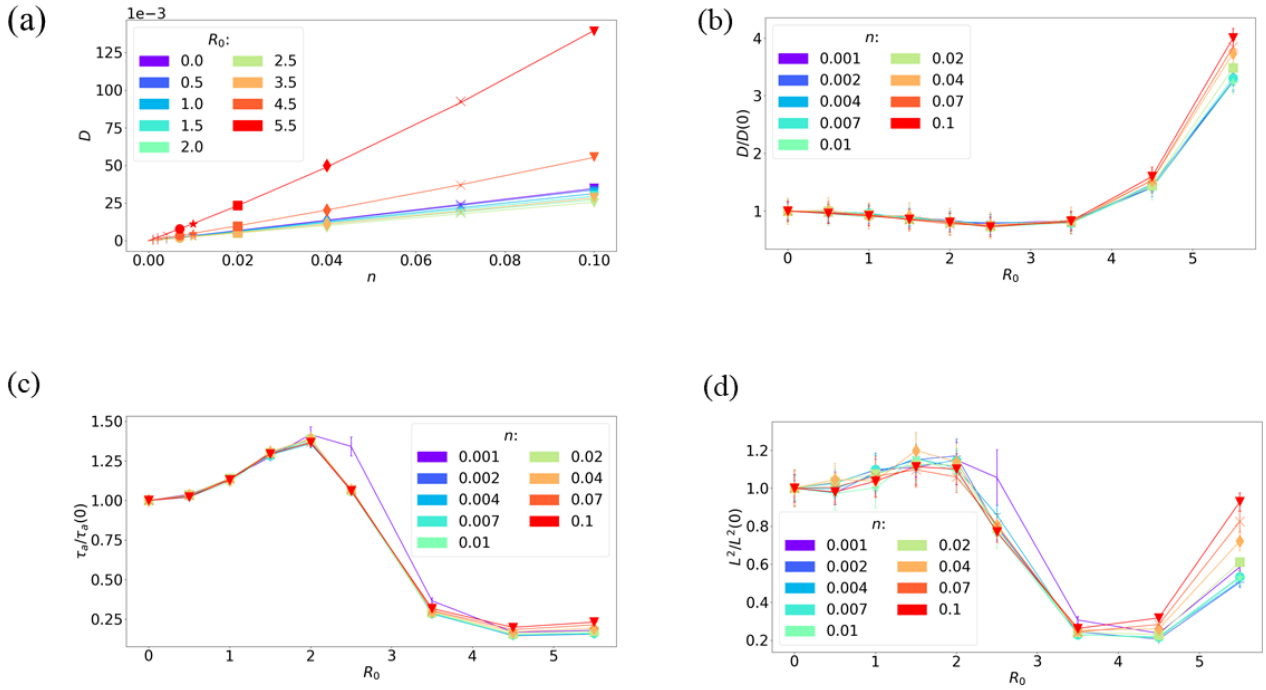

Figure 4.6: The markers indicate the densities throughout. (a) The diffusion coefficient as a function of swimmer density (linear and polynomial fits, see text). (b) The reduced diffusion coefficient as a function of tracer radius. (c) The reduced cross-over time τ_a as a function of tracer radius. (d) The reduced characteristic distance L^2 as a function of tracer radius.

Fig. 4.7(d). Initially, this function increases for pushers, which means that the enhancement effect of the interactions increases with R_0. Above $R_0 > 2.5$, the function drops sharply, but this might again be an artifact of overlapping pushers and swimmers. For pullers, the function is nearly mirrored. This means that the interaction effect, now a diminishing one, is also increased in puller suspensions. Why the higher radii are more difficult to interpret is addressed in the following discussion.

4.3 Discussion

So far, the LBM predicts that the Faxén correction does indeed affect the diffusion of finite size tracers significantly - especially when $R_0 > 2.5$. It does not, however, predict the results seen in some experiments. For example, $D(R_0)$ in the simulation is a convex function and not concave.

At the same time, the point force dipole method was successful in showing that a cut-off radius $\sim R_0$ must eventually be introduced. Otherwise, the calculated diffusion constant is strongly increased from near-field effects due to the diverging flow-field, which are absent in experiments due to excluded volume interactions. We also inferred from it that a cut-off radius on the order of R_0 is needed. In the LBM, no cut-off radius is explicitly implemented, but we know that the flow-field is regularized for distances below 2 lattice units, according to Eq. 3.20.

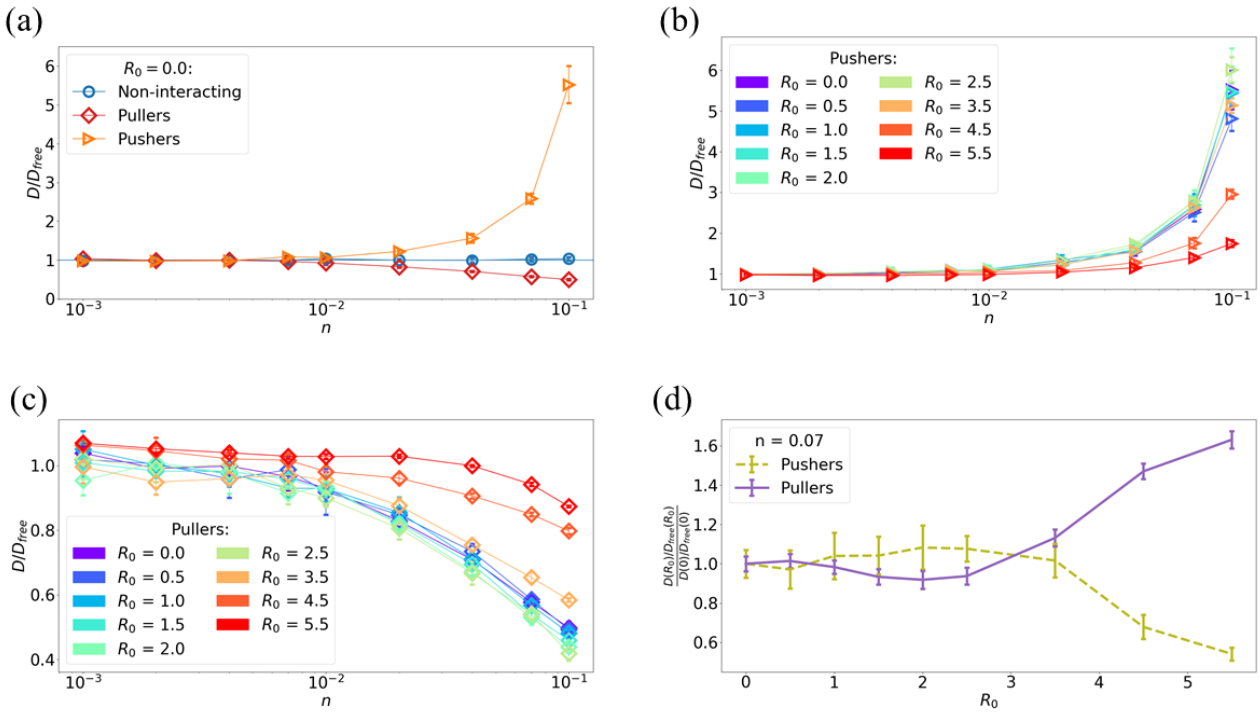

Figure 4.7: The difference between pushers and pullers surfaces in the interacting case. (a) The diffusion constants normalized with the corresponding non-interacting case. (b) The diffusion constants of pushers, normalized with the fit of the corresponding non-interacting case, as a function of swimmer density. (c) The same but for pullers. (d) At fixed n, the diffusion constants are non-monotone as functions of R_0.

Any lattice position further away than 2 lattice units from a swimmer does not "feel" the swimmer's force. Similarly, any swimmer or tracer only feels the fluid in its immediate vicinity. It seems to be the case that the stark deviation for large R_0 has to do with these interpolations. For small radii, this issue is not dominant but eventually it cannot be ignored.

From, for example, Fig. 4.6 (b) we had already concluded that up to $R_0 = 2.5$ the LBM simulation yields reasonable results. These observations are in accordance with the notion that the LBM uses an "implicit" cutoff radius of ~ 2, and explains why the results become seemingly unreliable for $R_0 > 2.5$. We see from Fig. 4.4 of the point dipole method that roughly all the cases up to $R_0 = 5\,\mu\text{m}$ plateau if we choose $c = 4\,\mu\text{m}$. Similarly, the "implicit" cut-off radius in the LBM causes the results for $R_0 < 3.5$ to not be inflicted by strong near field effects.

Again, up to this point, the diffusion constant was a slightly decreasing function of tracer radius. If everything beyond this is to be ignored, the non-monotone behaviour from experiment (Fig. 2.2 (a)) could not be reproduced. Similarly, the cross-over time in the interpretable regime changed much less than observed in experiments (compare Fig. 2.2 with Fig. 4.6 (c)).

It might be the case that the non-monotone behaviour can be explained by other effects, or a superpositions of multiple ones. The three established factors in diffusivity are Brownian motion, the "hydrodynamic" diffusivity discussed in this thesis and possible excluded volume

interactions with bacteria (i.e. collisions). It is not obvious that these three can be analyzed separately and than simply summed up to completely describe the effective diffusivity. At best, they should be studied all together because the velocity field around a tracer is a function of the tracer's position with respect to the bacteria, and this relative position depends both on Brownian motion and the bacterial velocity field. Similarly, collisions happen more often where there are more bacteria, but in these regions the velocity field is different from more dispersed regions. Attempts have been made in this direction [9] but never in the LBM framework nor with any other method that accounts for swimmer-swimmer interactions.

A natural continuation for the experiments would be to increase the swimmer density. It could then be possible to validate our results on interacting swimmer. At such high densities, however, collisions would become more and more important and we would really have to include them in simulations.

5. Conclusion and Outlook

From experiments we know that the diffusivity of tracer particles in microswimmer suspensions of specific bacterial species and size range is not monotone. We tried to predict this behaviour in an LBM simulation by including the derivative term in Faxén's law that is commonly ignored. Doing so for suspensions in which swimmers are independent of each other, we do indeed observe a non-monotone behaviour, although with a minimum instead of a maximum. Up to this minimum, the diffusivity is a slightly decreasing function. Using a simpler simulation that mimics the scattering of a single tracer by a point force dipole we could conclude that a cut-off radius is needed for realistic simulation. If the swimmer and the tracer are allowed to overlap, the Faxén correction becomes dominant and the diffusivity increases dramatically. It also makes physical sense not to allow such a penetration because the boundaries of either organism do not allow for it. Moreover, at such a short distance other effects such as collision should anyway dominate. A conservative conclusion from our LBM simulation is therefore that for small R_0, D decreases slightly with tracer size, whereas the subsequent increase with R_0 can possibly be a result from swimmer-tracer penetration.

It would obviously be useful to include a cut-off parameter in the LBM simulation but this is far less trivial than in the simple point dipole approximation. One could imagine to instead only update the tracer position if it is about to move below a threshold value. This, however, would probably reduce the code's efficiency. Furthermore, it would be nice to include collision effects into the simulation. Others use these to explain the non-monotone response [18] but they have never been simulated or otherwise analyzed in many particle systems. Because such an approach involves more than the flow field, it would not be an easy task to incorporate them into the LBM.

While this research is carried out on the fundamental level, understanding tracer diffusion is key to some important applications. For example, there are ways to transport microparticles for drug delivery, and microswimmers are one of them [32]. Knowing how tracers behave in their vicinity could lead to new delivery methods. A less obvious application might be artificial stirring in zones of oceans that are dead due the a lack of microorganisms [33]. Research is still conducted on how appropriate artificial swimmers might look like, and this also requires understanding of their effects on tracers. It is useful to aid all of this research using LBM simulations because they are so unique in dealing with high swimmer and tracer densities.

Bibliography

[1] Dóra Bárdfalvy, Henrik Nordanger, Cesare Nardini, Alexander Morozov, and Joakim Stenhammar. Particle-resolved lattice boltzmann simulations of 3-dimensional active turbulence. *Soft Matter*, 15:7747–7756, 2019.

[2] Joakim Stenhammar, Cesare Nardini, Rupert W. Nash, Davide Marenduzzo, and Alexander Morozov. Role of correlations in the collective behavior of microswimmer suspensions. *Phys. Rev. Lett.*, 119:028005, Jul 2017.

[3] Takuji Ishikawa. Suspension biomechanics of swimming microbes. *Journal of The Royal Society Interface*, 6:815 – 834, 2009.

[4] Kyriacos C. Leptos, Jeffrey S. Guasto, J. P. Gollub, Adriana I. Pesci, and Raymond E. Goldstein. Dynamics of enhanced tracer diffusion in suspensions of swimming eukaryotic microorganisms. *Phys. Rev. Lett.*, 103:198103, Nov 2009.

[5] Xiao-Lun Wu and Albert Libchaber. Particle diffusion in a quasi-two-dimensional bacterial bath. *Phys. Rev. Lett.*, 84:3017–3020, Mar 2000.

[6] Gastón Miño, Thomas E. Mallouk, Thierry Darnige, Mauricio Hoyos, Jeremi Dauchet, Jocelyn Dunstan, Rodrigo Soto, Yang Wang, Annie Rousselet, and Eric Clement. Enhanced diffusion due to active swimmers at a solid surface. *Phys. Rev. Lett.*, 106:048102, Jan 2011.

[7] Alys Jepson, Vincent A. Martinez, Jana Schwarz-Linek, Alexander Morozov, and Wilson C. K. Poon. Enhanced diffusion of nonswimmers in a three-dimensional bath of motile bacteria. *Phys. Rev. E*, 88:041002, Oct 2013.

[8] Xiao-Lun Wu and Albert Libchaber. Particle diffusion in a quasi-two-dimensional bacterial bath. *Phys. Rev. Lett.*, 84:3017–3020, Mar 2000.

[9] T. V. Kasyap, Donald L. Koch, and Mingming Wu. Hydrodynamic tracer diffusion in suspensions of swimming bacteria. *Physics of Fluids*, 26(8):081901, 2014.

[10] Alison E. Patteson, Arvind Gopinath, Prashant K. Purohit, and Paulo E. Arratia. Particle diffusion in active fluids is non-monotonic in size. *Soft Matter*, 12:2365–2372, 2016.

[11] Hilding Faxén. Der widerstand gegen die bewegung einer starren kugel in einer zähen flüssigkeit, die zwischen zwei parallelen ebenen wänden eingeschlossen ist. *Annalen der Physik*, 373(10):89–119, 1922.

[12] Joost de Graaf and Joakim Stenhammar. Lattice-boltzmann simulations of microswimmer-tracer interactions. *Phys. Rev. E*, 95:023302, Feb 2017.

[13] Egon Krause. *The Millennium-Problem of Fluid Mechanics – The Solution of the Navier-Stokes Equations*, pages 317–341. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[14] Eric Lauga and Thomas R. Powers. The hydrodynamics of swimming microorganisms. *Reports on Progress in Physics*, 72(9):096601, Aug 2009.

[15] Gastón Miño, Thomas E. Mallouk, Thierry Darnige, Mauricio Hoyos, Jeremi Dauchet, Jocelyn Dunstan, Rodrigo Soto, Yang Wang, Annie Rousselet, and Eric Clement. Enhanced diffusion due to active swimmers at a solid surface. *Phys. Rev. Lett.*, 106:048102, Jan 2011.

[16] Gastón Miño, Jocelyn Dunstan, Annie Rousselet, E. Clément, and Rodrigo Soto. Induced diffusion of tracers in a bacterial suspension: Theory and experiments. *Journal of Fluid Mechanics*, 729, 10 2012.

[17] Alys Jepson, Vincent A. Martinez, Jana Schwarz-Linek, Alexander Morozov, and Wilson C. K. Poon. Enhanced diffusion of nonswimmers in a three-dimensional bath of motile bacteria. *Phys. Rev. E*, 88:041002, Oct 2013.

[18] Antoine Lagarde, Noémie Dagès, Takahiro Nemoto, Vincent Démery, Denis Bartolo, and Thomas Gibaud. Colloidal transport in bacteria suspensions: from bacteria scattering to anomalous and enhanced diffusion. *arXiv*, 2020.

[19] diffusion. `https://www.lexico.com/en/definition/diffusion`, 2020. [Online; accessed 06-May-2020].

[20] Paul D. Beale R. K. Pathria. *Statistical Mechanics–3rd ed*. Elsevier Science, 2011.

[21] NG Van Kampen. *Stochastic processes in physics and chemistry*. North Holland, 2007.

[22] Aykut Argun, Ali-Reza Moradi, Erçağ Pinçe, Gokhan Baris Bagci, Alberto Imparato, and Giovanni Volpe. Non-boltzmann stationary distributions and non-equilibrium relations in active baths. In *Optics in the Life Sciences Congress*, page OtW3E.5. Optical Society of America, 2017.

[23] Jens Elgeti, Robert Winkler, and Gerhard Gompper. Physics of microswimmers–single particle motion and collective behavior: a review. *Reports on progress in physics. Physical Society*, 78 5:056601, 2015.

[24] Knut Drescher, Jörn Dunkel, Luis H. Cisneros, Sujoy Ganguly, and Raymond E. Goldstein. Fluid dynamics and noise in bacterial cell–cell and cell–surface scattering. *Proceedings of the National Academy of Sciences*, 108(27):10940–10945, 2011.

[25] Alexander Morozov and Davide Marenduzzo. Enhanced diffusion of tracer particles in dilute bacterial suspensions. *Soft Matter*, 10:2748–2758, 2014.

[26] Zhi Lin, Jean-Luc Thiffeault, and Stephen Childress. Stirring by squirmers. *Journal of Fluid Mechanics*, 669:167–177, 2011.

[27] Albert Einstein. *Investigations on the Theory of the Brownian Movement*. Dover Books on Physics Series. Dover Publications, 1956.

[28] Irene Stegun Milton Abramowitz. Handbook of mathematical functions. `http://people.math.sfu.ca/~cbm/aands/page_885.htm`, 2010. [Online; accessed 01-May-2020].

[29] Daniel V. Schroeder. Lattice-boltzmann fluid dynamics. `https://physics.weber.edu/schroeder/javacourse/LatticeBoltzmann.pdf`, 2012. [Online; accessed 28-April-2020].

[30] Timm Krüger, Halim Kusumaatmaja, Alexander Kuzmin, Orest Shardt, Goncalo Silva, and Erlend Magnus Viggen. *The Lattice Boltzmann Method - Principles and Practice*. Springer, 10 2016.

[31] Rupert W. Nash, Ronojoy Adhikari, and Michael E. Cates. Singular forces and pointlike colloids in lattice boltzmann hydrodynamics. *Phys. Rev. E*, 77:026709, Feb 2008.

[32] Liqiang Ren, Nitesh Nama, Jeffrey M. McNeill, Fernando Soto, Zhifei Yan, Wu Liu, Wei Wang, Joseph Wang, and Thomas E. Mallouk. 3d steerable, acoustically powered microswimmers for single-particle manipulation. *Science Advances*, 5(10), 2019.

[33] Kakani Katija. Morphology Alters Fluid Transport and the Ability of Organisms to Mix Oceanic Waters. *Integrative and Comparative Biology*, 55(4):698–705, 06 2015.

Appendices

A. Python Code

```
# -*- coding: utf-8 -*-
"""
Created on Fri May 29 15:40:12 2020

@author: André Nuesslein
A swimmer creates a flow field and a passive tracer advects.
Partly to reproduce results from a paper (DOI: 10.1039/c3sm52201f).
Extension: include the Faxén Laplcian correction for non point-like tracers.
Laplacian implemented both analytically and discretely.
Also: analyze different cut-off parameters c (see paper).
The code produces results such as figure 2 in the paper.
It also creates files which list the parameter A (see paper) as a function of c.
"""

import numpy as np
from scipy.spatial import distance
import math
import matplotlib.pyplot as plt
plt.rcParams.update({'font.size': 32})

#swimmer parameters:
p = 32 #coefficient in front of the velocity field
orientation = [1, 0] #orientation of the swimmer
DT = 0.001 #time step
V = 22 #velocity of the swimmer
Lambda = 10 #average distance covered by swimmer before tumbling
Steps = int(Lambda/V/DT)
sigma = math.sqrt(p/V)

h = 0.05 #parameter for discrete Laplacian

#various initial distances between tracer and swimmer:
GridSpacing = 0.05
Beg = -30
End = 30
Length = End - Beg
num = int(Length/GridSpacing + 1)
Xi_array = np.linspace(Beg, End, num)
Chi_array = np.linspace(Beg, End, num)

tracerInitial = np.array([0,0]) #convenient to set the tracer in the origin

Radius_List = [8]
C_List = [1, 3, 5, 7, 9, 11] #cut-off radius

def u(A, r, d): #velocity field (at distance r) of a swimmer placed at
```

```
    the
47                  #origin with orientation d (equation 1 in DOI:
    1019079108)
48      r_hat = r / np.linalg.norm(r)
49      d_hat = d / np.linalg.norm(d)
50      return A/(np.linalg.norm(r)**2) * (3*(np.dot(r_hat, d_hat))**2 - 1)
    * r_hat
51
52  def Laplace2D(A, Position, d, h): #Laplace of a velocity field
53      Position_x1 = Position + [h,0]
54      Position_x2 = Position + [-h,0]
55      Position_y1 = Position + [0,h]
56      Position_y2 = Position + [0,-h]
57      V_x1 = u(A, Position_x1, d)
58      V_x2 = u(A, Position_x2, d)
59      V_y1 = u(A, Position_y1, d)
60      V_y2 = u(A, Position_y2, d)
61      Laplace = (V_x1 + V_x2 + V_y1 + V_y2 - 4 * u(A, Position, d))/h**2
62      return Laplace
63
64  def LaplaceAnalytic(A, r, Radius): #assumes horizontal swimmer
    orientation
65        rr = np.linalg.norm(r)
66        LaplaceU = A/(rr**7)*np.array([r[0]*(3*rr**2 - 5*r[0]**2), \
67                                       r[1]*(rr**2 - 5*r[0]**2)])*(Radius
    **2)
68        return LaplaceU
69
70  def NoLaplaceDistanceWrite(radius, c):
71      f = open(f'Integrand_TracerRadius_0_c_{c}.dat','w')
72      for Xi in Xi_array:
73          for Chi in Chi_array:
74              b = Lambda * Chi
75              a = sigma * np.exp(Xi)
76              swimmerInitial = np.array([-b, a])
77              swimmer = swimmerInitial
78              tracer = tracerInitial
79              for t in range(0, Steps):
80                  r = np.subtract(tracer, swimmer) #The position of the
    tracer in the coordinate system centered around the swimmer
81                  if np.linalg.norm(r)>c:
82                      velocity = u(p, r, orientation)
83                      tracer = np.add(tracer, velocity*DT)
84                  swimmer = np.add(swimmer, [V*DT, 0])
85              d = distance.euclidean(tracer, tracerInitial)
86              Integrand = np.exp(2*Xi) * d**2 / sigma**2
87              f.write(f'{Chi} {Xi} {Integrand}\n')
88      f.close()
89
90  def DiscreteLaplaceDistanceWrite(radius, c):
91      f = open(f'Discrete_Laplace_Integrand_TracerRadius_{radius}_c_{c}.
    dat','w')
92      for Xi in Xi_array:
93          for Chi in Chi_array:
```

```
            b = Lambda * Chi
            a = sigma * np.exp(Xi)
            swimmerInitial = np.array([-b, a])
            swimmer = swimmerInitial
            tracer = tracerInitial
            for t in range(0, Steps):
                r = np.subtract(tracer, swimmer) #The position of the tracer in
                            #the coordinate system centered around the swimmer
                if np.linalg.norm(r)>c:
                    velocity = u(p, r, orientation)
                    Laplaced = Laplace2D(p, r, orientation, h)
                    FaxVelocity = velocity + radius**2/6*Laplaced
                    tracer = np.add(tracer, FaxVelocity*DT)
                swimmer = np.add(swimmer, [V*DT, 0])
            d = distance.euclidean(tracer, tracerInitial)
            Integrand = np.exp(2*Xi) * d**2 / sigma**2
            f.write(f'{Chi} {Xi} {Integrand}\n')
    f.close()

def AnalyticLaplaceDistanceWrite(radius, c):
    f = open(f'Analytic_Laplace_Integrand_TracerRadius_{radius}_c_{c}.dat','w')
    for Xi in Xi_array:
        for Chi in Chi_array:
            b = Lambda * Chi
            a = sigma * np.exp(Xi)
            swimmerInitial = np.array([-b, a])
            swimmer = swimmerInitial
            tracer = tracerInitial
            for t in range(0, Steps):
                r = np.subtract(tracer, swimmer) #The position of the tracer in
                            #the coordinate system centered around the swimmer
                if np.linalg.norm(r)>c:
                    velocity = u(p, r, orientation)
                    LaplaceU = LaplaceAnalytic(p, r, radius)
                    FaxVelocity = velocity + LaplaceU
                    tracer = np.add(tracer, FaxVelocity*DT)
                swimmer = np.add(swimmer, [V*DT, 0])
            d = distance.euclidean(tracer, tracerInitial)
            Integrand = np.exp(2*Xi) * d**2 / sigma**2
            f.write(f'{Chi} {Xi} {Integrand}\n')
    f.close()

def ComputeIntegral(FileName):
    Sum = 0
    with open(FileName,'r') as f:
        for line in f:
            values = [float(s) for s in line.split()]
            Sum += values[2]
    Integral = Sum*Length**2/num**2
```

```
    return Integral

X, Y = np.meshgrid(Chi_array, Xi_array)


for radius in Radius_List:
    f = open(f'Discrete_Laplace_Integrand_TracerRadius_{radius}.dat','w')
    f.write('c A \n')
    g = open(f'Analytic_Laplace_Integrand_TracerRadius_{radius}.dat','w')
    g.write('c A \n')
    for c in C_List:
        #discrete verison:
        DiscreteLaplaceDistanceWrite(radius, c)
        Integral = ComputeIntegral(f'Discrete_Laplace_Integrand_TracerRadius_{radius}_c_{c}.dat')
        A = np.pi/3*Integral
        f.write(f'{c} {A}\n')

        Data = np.loadtxt(f'Discrete_Laplace_Integrand_TracerRadius_{radius}_c_{c}.dat', delimiter=' ')
        Z = np.array(np.split(Data[:,2], len(X)))

        fig = plt.figure(figsize=(6,5))
        left, bottom, width, height = 0.1, 0.1, 0.8, 0.8
        ax = fig.add_axes([left, bottom, width, height])

        cp = plt.contourf(X, Y, Z, 500, cmap = 'gist_stern')
        plt.colorbar(cp)

        ax.set_ylim(-6, 6)
        ax.set_xlim(-4.5, 4.5)
        ax.set_title(f'r = {radius}, Discrete, DT = {DT}, num = {num}, from {Beg} to {End}, A = {A}, c = {c}')
        ax.set_xlabel('\u03C7 = b/\u03BB')
        ax.set_ylabel('\u03BE = log(a/\u03C3)')
        plt.show()
        #analytic verison:
        AnalyticLaplaceDistanceWrite(radius, c)
        Integral = ComputeIntegral(f'Analytic_Laplace_Integrand_TracerRadius_{radius}_c_{c}.dat')
        A = np.pi/3*Integral
        g.write(f'{c} {A}\n')

        Data = np.loadtxt(f'Analytic_Laplace_Integrand_TracerRadius_{radius}_c_{c}.dat', delimiter=' ')
        Z = np.array(np.split(Data[:,2], len(X)))

        fig = plt.figure(figsize=(6,5))
        left, bottom, width, height = 0.1, 0.1, 0.8, 0.8
        ax = fig.add_axes([left, bottom, width, height])

        cp = plt.contourf(X, Y, Z, 500, cmap = 'gist_stern')
```

```
        plt.colorbar(cp)

        ax.set_ylim(-6, 6)
        ax.set_xlim(-4.5, 4.5)
        ax.set_title(f'r = {radius}, Analytic, DT = {DT}, num = {num},
    from {Beg} to {End}, A = {A}, c = {c}')
        ax.set_xlabel('\u03C7 = b/\u03BB')
        ax.set_ylabel('\u03BE = log(a/\u03C3)')
        plt.show()
    f.close()
    g.close()
```