université de BORDEAUX

université de BORDEAUX

THÈSE PRÉSENTÉE

POUR OBTENIR LE GRADE DE

DOCTEUR

DE L'UNIVERSITÉ DE BORDEAUX

ECOLE DOCTORALE SCIENCES PHYSIQUES ET DE L'INGÉNIEUR

LASERS, MATIÈRE, NANOSCIENCES

Par **Maxime Lavaud**

Confined Brownian Motion

Sous la direction de : **Thomas Salez**
Co-directrion : **Yacine Amarouchene**

Soutenue le 25 décembre 2019

Membres du jury :

Mme. Aude ALPHA	Directrice de Recherche	Université	Rapporteur
M. Bernard BETA	Directeur de Recherche	Université	Rapporteur
M. Georges GAMMA	Directeur de Recherche	Université	Président
Mme. Dominique DELTA	Chargée de Recherche	Université	Examinatrice
M. Eric EPSILON	Ingénieur de Recherche	Université	Examinateur
Mme. Jane DOE	Directrice de Recherche	Université	Directrice
Mme. Simone UNTEL	Ingénieure de Recherche	Université	Invitée

Abstract

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Table of Contents

List of Figures

List of Abbreviations

fps	Frames per second
MRSE	Mean Relative Squared Error
MSD	Mean Squared Displacement
PDF	Probability Density Function
RICM	Reflection Interference Contrast Microscopy
ROI	Region Of Interest
SDE	Stochastic Differential Equations

1 Brownian Motion

1.1 The Brownian motion discovery

in 1827 the Scottish botanist Robert Brown published an article [25] on his observation on the pollen of *Clarkia pulchella* with a lot of details on his reflection processes. His experiments were made to understand the flower reproduction, but, as he was looking through the microscope he observed some minute particles ejected from the pollen grains. At first, he thought the goal of this agitation was to test the presence of a male organ. To test this theory, he extended his observations to Mosses and *Equiseta*, which were drying for a hundred years. However, the fact that this peculiar motion was still observable made him invalidate his theory. Interestingly, each time that he encountered a material that he could reduce to a fine enough powder to be suspended in water, he observed the same type of motion, although, he never understood its particle's movement.

The difficulty at this time to observe and capture such a movement made the study of what we call contemporarily Brownian motion difficult and the first theoretical work was done by Louis Bachelier in his PhD thesis "The theory of speculation," where he described a stochastic analysis of the stock and option market. Nowadays, the mathematical description of random movement is still used in the modern financial industry.

It is finally in 1905 that Albert Einstein theoretically state that "bodies of microscopically visible size suspended in a liquid will perform movements of such a magnitude that they can be easily observed in a microscope" [4]. A remark to make here is that in 1948 Einstein wrote a letter to one of his friends where he stated having deduced the Brownian motion "from mechanics, without knowing that anyone had already observed anything of the kind" [26].

It is in 1908 that Jean Perrin published his experimental work on Brownian motion. That way he could measure the Avogadro number and prove the kinetic theory that Einstein developed. I would also cite Chaudesaigues and Dabrowski, who helped Perrin to track the particles manually, half-minutes by half-minutes, for more than 3000 displacements (25 hours) and several particles. This impressive and daunting work is highly detailed in *"Mouvement brownien et molécules"* [27]. This is partly due to the results this work that Perrin received the Nobel award in 1926.

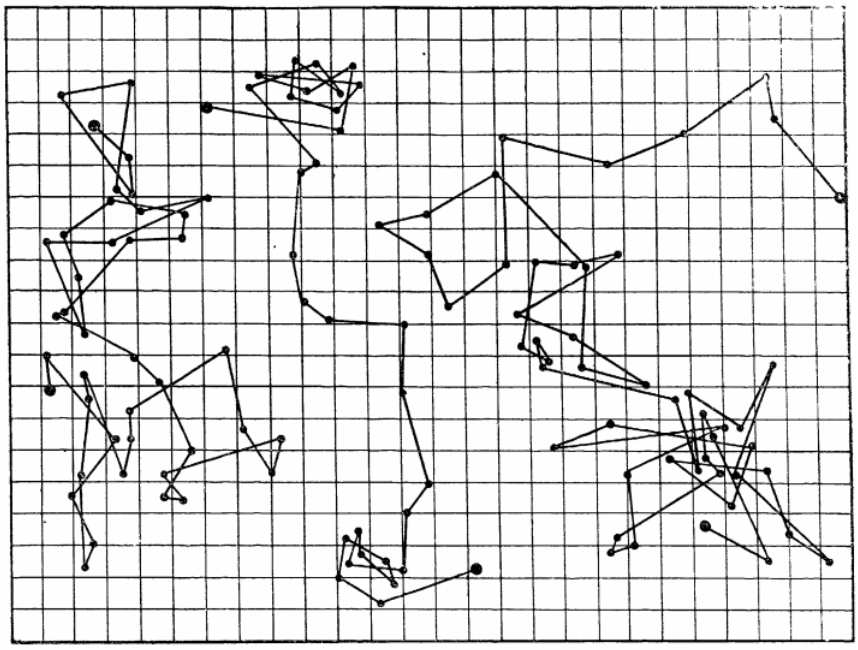


Figure 1: Brownian motion of 1 μm particles in water tracked manually by Jean Perrin and his colleagues. The points are spaced in time by 30 seconds, and 16 divisions represent 50 μm.

1.2 Einstein's Brownian theory

In this section we will derive the main characteristics of bulk Brownian motion in the manner of Einstein in 1905 by summarizing the section 4 of [4]. We will then examine the random motion of particles suspended in a liquid and its relation to diffusion, caused by thermal molecular motion. We assume that each particle motion is independent of other particles; also the motions of one particle at different times are assumed to be independent of one another provided that the time interval is not too small. Furthermore, we now introduce a time interval τ which is small compared to the observation time but large enough so that the displacements in two consecutive time intervals τ may be taken as independent events.

For simplicity, we will here look only at the Brownian motion of n particles in 1D along the x axis. In a time interval τ the position of each particle will increase by a displacement Δ, positive or negative. The number of particles $\mathrm{d}n$ experiencing a displacement lying between Δ and $\Delta + \mathrm{d}\Delta$ in a time interval τ is written as:

$$\mathrm{d}n = n\varphi(\Delta)\mathrm{d}\Delta \ , \tag{1.2.1}$$

where

$$\int_{-\infty}^{\infty} \varphi(\Delta) \mathrm{d}\Delta = 1 \ , \tag{1.2.2}$$

and φ is the probability distribution of displacement. We assume for now, that φ is a Gaussian distribution, with a variance scaling linearly with τ. Additionally, since such a distribution is even, it satisfies: $\varphi(\Delta) = \varphi(-\Delta)$.

Let $f(x,t)$ be the number of particles per unit volume. From the definition of the function $\varphi(\Delta)$ we can obtain the distribution of particles found at time $t+\tau$ from their distribution at a time t, through:

$$f(x, t+\tau) = \int_{-\infty}^{+\infty} f(x-\Delta, t)\varphi(\Delta)\mathrm{d}\Delta \ . \tag{1.2.3}$$

Since we suppose that τ is very small with respect to t, we have at first order in time:

$$f(x, t+\tau) \simeq f(x,t) + \tau \frac{\partial f}{\partial t} \ . \tag{1.2.4}$$

Besides, we can Taylor expend $f(x+\Delta, t)$ in powers of Δ since only small values of Δ contribute. We obtain:

$$f(x-\Delta, t) = f(x,t) - \Delta \frac{\partial f(x,t)}{\partial x} + \frac{\Delta^2}{2!} \frac{\partial^2 f(x,t)}{\partial x^2} \ . \tag{1.2.5}$$

Combining Eqs.1.2.4, 1.2.5 and 1.2.3 we obtain:

$$f + \frac{\partial f}{\partial t}\tau = f \int_{-\infty}^{+\infty} \varphi(\Delta)\mathrm{d}\Delta + \frac{\partial f}{\partial x} \int_{-\infty}^{+\infty} \Delta\varphi(\Delta)\mathrm{d}\Delta + \frac{\partial^2 f}{\partial x^2} \int_{-\infty}^{+\infty} \frac{\Delta^2}{2}\varphi(\Delta)\mathrm{d}\Delta \ . \tag{1.2.6}$$

On the right-hand side, since $\varphi(x)$ is an even function, the second term vanishes. Considering Eq. (1.2.2) and invoking the definition:

$$\frac{1}{\tau} \int_{-\infty}^{+\infty} \frac{\Delta^2}{2} \varphi(\Delta)\mathrm{d}\Delta = D \ , \tag{1.2.7}$$

Eq. (1.2.6) finally becomes:

$$\frac{\partial f}{\partial t} = D\frac{\partial^2 f}{\partial x^2} \ . \tag{1.2.8}$$

We can here recognize a partial equation of diffusion with D the diffusion coefficient. We will now initiate the same position $x = 0$ for all the particles at $t = 0$ as in Fig.2. $f(x,t)\mathrm{d}x$ denotes the number of particles whose positions have increased between the times 0 and t by a quantity lying between x and $x + dx$ such that we must have:

$$f(x \neq 0, t = 0) = 0 \text{ and } \int_{-\infty}^{+\infty} f(x,t)dx = n \ . \tag{1.2.9}$$

The solution Eq. (1.2.8) is then the Green's function of the heat equation in the bulk:

$$f(x,t) = \frac{1}{\sqrt{4\pi D}} \frac{\exp\left(\frac{-x^2}{4Dt}\right)}{\sqrt{t}} \ . \tag{1.2.10}$$

From this solution we can see that the mean value of the displacement along the x axis is equal to 0 and the square root of the arithmetic mean of the squares of displacements (that we commonly call the Root Mean Square Displacement (RMSD)) is given by:

$$\lambda_x = \sqrt{\langle \Delta^2 \rangle} = \sqrt{2Dt}. \tag{1.2.11}$$

The mean displacement is thus proportional to the square root of time. This result is generally the first behavior that we check when we study Brownian motion. In 3D, the square root of the MSD is given by $\lambda_x\sqrt{3}$.

Previously in his article [4], Einstein had found by writing the thermodynamic equilibrium of a suspension of particles that the diffusion coefficient of a particle should read:

$$D = \frac{RT}{N_\mathrm{A}} \frac{1}{6\pi\eta a} = \frac{k_\mathrm{B}T}{6\pi\eta a} \ , \tag{1.2.12}$$

)

with R the gas constant, T the temperature, N_A the Avogadro number, η the fluid

viscosity and k_B the Boltzmann constant. Thus, an experimental measurement of D could lead to a measurement of the Avogadro number since:

$$N_A = \frac{t}{\lambda_x^2} \frac{RT}{3\pi\eta a} \ . \tag{1.2.13}$$

Furthermore, measuring N_A also gives us the mass of atoms and molecules since the mass of a mole is known; as an example the mass of an oxygen atom will be given by $\frac{16}{N_A}$ and the mass a water molecule by $\frac{18}{N_A}$. Finally, Einstein ends up is article [4] by writing, *"Let us hope that a researcher will soon succeed in solving the problem posed here, which is of such importance in the theory of heat!"* I would like here to emphasize the importance of solving this problem at the very beginning of the 20th century. At this time two hypotheses about the fundamental matter components existed, one involving energy and a continuum description in terms of field, and the other one, discrete atoms, especially supported by Boltzmann and his kinetic theory of gases, used by Einstein. Due to a lot of conceptual misunderstandings and experimental error scientist such as Svedberg or Henri thought that Einstein's theory was false [28] by even suggesting that the statistical properties of Brownian motion were changing with the pH of the solution. It is finally in 1908 that Chaudesaigues and Perrin published all the evidence to prove Einstein's theory mainly by their ability to create particle emulsions of well-controlled radii.

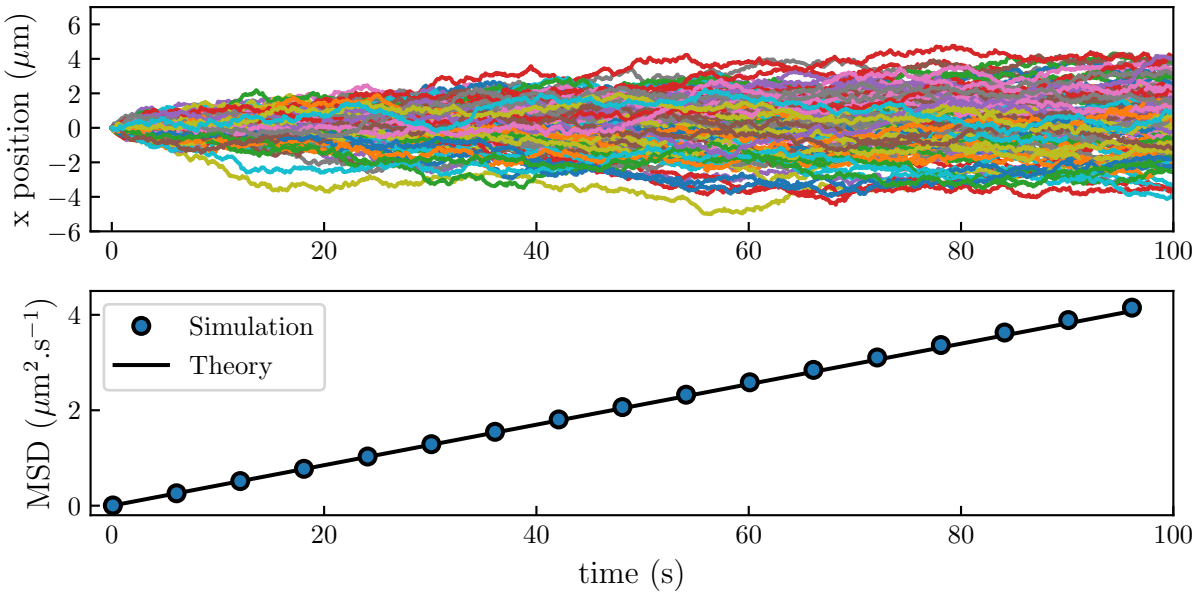


Figure 2: Simulation of over-damped Brownian motion in the bulk (see Eq. (1.4.9)) of 1 μm particles in water. On the top each line represents the trajectory of a Brownian particle over 100 seconds. A total of 100 trajectories are shown. On the bottom, bullets represent the Mean Square Displacement (MSD) computed from the simulated trajectories. The plain black line represents Einstein's theory, which is computed from the square of Eq. (1.2.11).

1.3 The Langevin Equation

in physics we generally describe Brownian motion through a particular Stochastic Differential Equations (SDE). This model was introduced in 1908 by Langevin [29], this model is now used by the major part of physicists working on random processes. The Langevin equation for a free colloid reads:

$$m\mathrm{d}V_t = -\gamma V_t \mathrm{d}t + \alpha \mathrm{d}B_t \ , \tag{1.3.1}$$

with m the mass and V_t the velocity of the particle. This SDE is the Newton's second law, relating the particle momentum change on the left-hand side of the equation to forces on the right-hand side. We see that the total force applied on the particle is given by two terms: a friction term, with a Stokes-like fluid friction coefficient γ, a random force with α that we will detail for a spherical particle, $\mathrm{d}Bt$ a random noise which has a Gaussian distribution of zero mean thus:

$$\langle \mathrm{d}B_t \rangle = 0 \ , \tag{1.3.2}$$

and variance equal to:

$$\langle \mathrm{d}B_t^2 \rangle = \mathrm{d}t \ . \tag{1.3.3}$$

For a spherical particle, the friction term is given by the Stoke's formula: $\gamma = 6\pi\eta a$ with η the fluid viscosity and α the particle radius. Thus, we can derive the mean value of the particle velocity as:

$$\langle \frac{\mathrm{d}V_t}{\mathrm{d}t} \rangle = -\frac{\gamma}{m}\langle V_t \rangle \mathrm{d}t + \frac{\alpha}{m}\langle \mathrm{d}B_t \rangle \ , \tag{1.3.4}$$

with the properties of dB_t given by Eq. (1.3.2), it becomes:

$$\langle \mathrm{d}V_t \rangle = -\frac{\gamma}{m}\langle V_t \rangle \mathrm{d}t \ . \tag{1.3.5}$$

Moreover, without a loss of generality, the average of a variable x, $\langle x \rangle$, is done over a set

of N observations $\{x_i\}$ such as:

$$\langle x \rangle = \frac{1}{N} \sum_{i=1}^{N} x_i \ , \tag{1.3.6}$$

one can then show that:

$$\frac{\mathrm{d}}{\mathrm{d}t}\langle x \rangle = \frac{\mathrm{d}}{\mathrm{d}t}\left[\frac{1}{N}\sum_{i=1}^{N} x_i\right] = \frac{1}{N}\sum_{i=1}^{N}\frac{\mathrm{d}}{\mathrm{d}t}x_i = \langle \frac{\mathrm{d}}{\mathrm{d}t}x \rangle \ . \tag{1.3.7}$$

The latter thus shows that it is possible to invert average value $\langle \cdot \rangle$ and a derivative. Therefore, Eq. (1.3.5) becomes:

$$\frac{\mathrm{d}}{\mathrm{d}t}\langle V_t \rangle = -\frac{\gamma}{m}\langle V_t \rangle \ , \tag{1.3.8}$$

which has a familiar solution:

$$\langle V_t(t) \rangle = V_0 \mathrm{e}^{-\frac{\gamma}{m}t} \ , \tag{1.3.9}$$

with V_0 an initial velocity. This result shows that the average of the velocity should decay to zero with a characteristic time $\tau_{\mathrm{B}} = \frac{m}{\gamma}$. For instance, the polystyrene particles used during my experiments which are micrometric we have $\tau_{\mathrm{B}} \approx 10^{-7}$ s. This signifies that if we measure the displacements of a particle with a time interval $\tau >> \tau_{\mathrm{B}}$ the displacement can be taken as independent events as it was stated by Einstein. In physical terms, this means that we are in the over-damped regime, in this case the Langevin equation reads:

$$-\gamma V_t \mathrm{d}t + \alpha \mathrm{d}B_t = 0 \ . \tag{1.3.10}$$

The experiments done during my thesis used a video camera that can reach a maximum of hundreds frames per second (fps) reaching time steps of $\approx 10^{-2}$ s. Therefore, all my work falls into the over-damped regime. Before focusing definitely on Eq. (1.3.10), we can use Eq. (1.3.4) to characterize further the unknown coefficient α. To do so we compute the mean square value of Eq. (1.3.4), starting by taking the second order Taylor expansion:

$$\begin{aligned} \mathrm{d}\left(V_t^2\right) &\simeq \frac{\partial V_t^2}{\partial V_t}\mathrm{d}V_t + \frac{1}{2}\frac{\partial^2 V_t^2}{\partial V_t^2}\left(\mathrm{d}V_t\right)^2 \\ &= 2V_t\mathrm{d}V_t + \left(\mathrm{d}V_t\right)^2 \end{aligned} \tag{1.3.11}$$

combining Eqs. (1.3.1) and (1.3.11), we obtain by only keeping the terms of order dt:

$$\mathrm{d}\left(V_t^2\right) = 2V_t\left(-\frac{\gamma}{m}V_t\mathrm{d}t + \frac{\alpha}{m}\mathrm{d}B_t\right) + \frac{\alpha^2}{m^2}\mathrm{d}B_t^2 \ . \tag{1.3.12}$$

Thus, the average value of $\mathrm{d}(V_t^2)$ reads:

$$\langle \mathrm{d}(V_t^2)\rangle = -2\frac{\gamma}{m}\langle V_t^2\rangle\mathrm{d}t + 2\frac{\alpha}{m}\langle V_t\mathrm{d}B_t\rangle + \frac{\alpha^2}{m^2}\langle \mathrm{d}B_t^2\rangle \ . \tag{1.3.13}$$

Moreover, since $\mathrm{d}B_t$ is chosen independently of the velocity V_t, one can write $\langle V_t\mathrm{d}B_t\rangle = \langle V_t\rangle\langle \mathrm{d}B_t\rangle = 0$. Taking the latter remark into account and the fact that $\langle \mathrm{d}B_t^2\rangle = \mathrm{d}t$, Eq. (1.3.13) becomes:

$$\langle d(V_t^2)\rangle = \left[-2\frac{\gamma}{m}\langle V_t^2\rangle + \frac{\alpha^2}{m^2}\right]\mathrm{d}t \ . \tag{1.3.14}$$

Since equilibrium averages in thermodynamics must become time independent, we have $\langle d(V_t^2)\rangle = 0$, thus:

$$\langle V_t^2\rangle = \frac{\alpha^2}{2\gamma m} \ . \tag{1.3.15}$$

Besides, from the equipartition of energy we also know that:

$$\langle \frac{1}{2}mV_t^2\rangle = \frac{1}{2}k_{\mathrm{B}}T \ . \tag{1.3.16}$$

The latter equation permits a direct determination of the amplitude of the noise α:

$$\alpha = \sqrt{2k_{\mathrm{B}}T\gamma} \ . \tag{1.3.17}$$

The latter result permits computing the amplitude of the random force in the Langevin equation. Taking the over-damped Langevin equation, it reads:

$$V_t \mathrm{d}t = \sqrt{2\frac{k_\mathrm{B}T}{\gamma}}\mathrm{d}B_t \tag{1.3.18}$$

Furthermore, one can write the position of the particle X_t at a time t, such as:

$$X_t = \int_0^t V_{t'} dt' \ , \tag{1.3.19}$$

where we can suppose at the initial time $t = 0$ that $X_0 = 0$. Computing $\langle X_t^2 \rangle$ using Eqs. (1.3.18), (1.3.19) and (1.3.3) thus gives:

$$\langle X_t^2 \rangle = 2\frac{k_\mathrm{B}T}{\gamma}t = 2Dt \tag{1.3.20}$$

By relating $\langle X_t^2 \rangle$ to the Mean Square Displacement (MSD) to the initial position such as:

$$\mathrm{MSD} = \langle (X_0 - X_t)^2 \rangle = \langle X_t^2 \rangle \ , \tag{1.3.21}$$

we obtain that the MSD should be linear with the time. This result confirms that using the over-damped Langevin equation, leads to the Einstein's result Eq. (1.2.11). Where one can identify the diffusion coefficient of the particle to be $D = k_\mathrm{B}T/\gamma$. Additionally, the latter identity is called the Stokes-Einstein relation.

Additionally, the Langevin equation is great to compute correlator such as the velocity correlator $\langle V_{t'} V_{t''} \rangle$ which the simplest to compute and the one that we will detail below. Indeed, if we use the full-Langevin equation, $\langle X_t^2 \rangle$ can't be that easily computed since $m\mathrm{d}V_t$ does not vanish. We would thus need to rewrite Eq. (1.3.20) using the velocity correletor such as:

$$\langle X_t^2 \rangle = \int_0^t \int_0^t \langle V_{t'} V_{t''} \rangle dt' dt'' \ . \tag{1.3.22}$$

Let us now study how the two-point correlator function $\langle V_{t'} V_{t''} \rangle$, using the full-Langevin

equation multiplied by V_0 and following the same steps as for Eq. (1.3.9), one has:

$$\langle V_t V_0 \rangle = \langle V_0^2 \rangle \mathrm{e}^{-t/\tau_\mathrm{B}} \ . \tag{1.3.23}$$

As the equilibrium state is invariant under temporal translation and assuming that V_0 has an equilibrium steady-state distribution with $\langle V_0^2 \rangle = k_\mathrm{B}T/m$ we have:

$$\langle V_t V_t' \rangle = \frac{k_\mathrm{B}T}{m} \mathrm{e}^{-|t-t'|/\tau_\mathrm{B}} \ . \tag{1.3.24}$$

One can solve Eq. (1.3.22) by splitting the integral in two parts, where $t' > t''$ and $t' < t''$:

$$\begin{aligned} \langle X_t^2 \rangle &= \frac{k_\mathrm{B}T}{m} \int_0^t \mathrm{d}t' \int_0^{t'} \mathrm{d}t'' \mathrm{e}^{-|t'-t''|/\tau_\mathrm{B}} = 2\frac{k_\mathrm{B}T}{\gamma} \left(\int_0^t \mathrm{d}t' \left[1 - \mathrm{e}^{-t'/\tau_\mathrm{B}} \right] \right) \\ &= 2\frac{k_\mathrm{B}T}{\gamma} \left(t - \tau_\mathrm{B} \left[1 - \mathrm{e}^{-t/\tau_\mathrm{B}} \right] \right) \ . \end{aligned} \tag{1.3.25}$$

We can extract two results from that equation. At a short time $t << \tau_\mathrm{B}$, one has:

$$\begin{aligned} \langle X_t^2 \rangle &\simeq 2\frac{k_\mathrm{B}T}{\gamma} \left(t - \tau_\mathrm{B} \left[1 - 1 + \frac{t}{\tau_\mathrm{B}} - \frac{t^2}{2\tau_\mathrm{B}^2} \right] \right) \\ &= \frac{k_\mathrm{B}T}{m} t^2 \ . \end{aligned} \tag{1.3.26}$$

This is the ballistic regime. If one can experimentally explore times shorter than τ_B one will then measure the real velocity of the particle. At longer times, $t >> t_\mathrm{B}$, the MSD is given by:

$$\langle X_t^2 \rangle \simeq 2\frac{k_\mathrm{B}T}{\gamma} t = 2Dt \ . \tag{1.3.27}$$

This is the diffusive regime where the MSD, as found earlier, Eq. (1.3.20) with the over-damped Langevin equation. To study this different result, it can be interesting to simulate the Brownian motion.

1.4 Numerical simulation of bulk Brownian motion

1.4.1 The numerical Langevin Equation

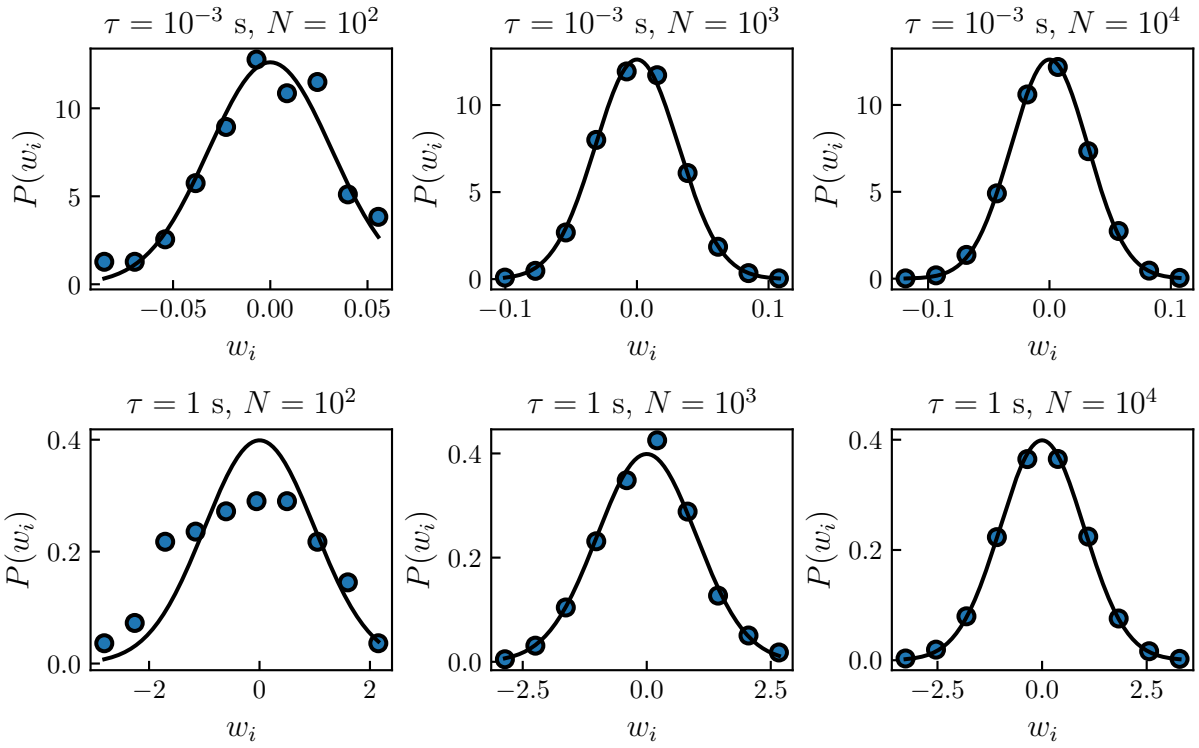


Figure 3: Bullets represents the probability density function of w_i, a Gaussian-distributed number with a mean value $\langle w_i \rangle$ and a variance $\langle w_i^2 \rangle = \tau$. The plain black line is a Gaussian distribution of zero mean and a τ variance (see Eq. (1.4.3)). On the first line, the simulation is done with $\tau = 10^{-3}$ s and $\tau = 1$ s on the second one. Each column corresponds to a number N of draws. From the left to the right: $N = 10^2,\ 10^3$ and 10^4.

The Langevin equation is an ordinary differential equation that can easily be numerically simulated in the bulk case. One approximates the continuous position X_t of a particle at a time t by a discrete-time sequence x_i which is the solution of the equation at a time $t_i = i\tau$, τ being the time step of the simulation. One can then use the Euler method to numerically write V_t as:

$$V_t \simeq \frac{x_i - x_{i-1}}{\tau} \ , \tag{1.4.1}$$

and $\mathrm{d}V_t/\mathrm{d}t$ as

$$\frac{\mathrm{d}V_t}{\mathrm{d}t} \simeq \frac{\frac{x_i - x_{i-1}}{\tau} - \frac{x_{i-1} - x_{i-2}}{\tau}}{\tau} = \frac{x_i - 2x_{i-1} + x_{i-2}}{\tau^2} \,. \tag{1.4.2}$$

The only term remaining to be computed numerically is the random term $\mathrm{d}B_t$. One can thus replace $\mathrm{d}B_t/\mathrm{d}t$ by w_i/τ [1] a Gaussian distributed random number generated with a mean $\langle w_i \rangle = 0$ and a variance $\langle w_i^2 \rangle = \tau$. The Probability Density function (PDF) of the Gaussian distribution is thus given by:

$$P(w_i) = \frac{1}{\sqrt{2\pi\tau}} \mathrm{e}^{-\frac{w_i^2}{2\tau}}. \tag{1.4.3}$$

The random number w_i can be generated with the following Python snippet.

```
import numpy as np

tau = 0.5  # time step in seconds
wi = np.random.normal(0, np.sqrt(tau))
```

In the latter, `random.normal()` is a built-in Numpy module that permits the generation of Gaussian-distributed random numbers. Finally, by combining Eqs. (1.4.1) and (1.4.2), the full-Langevin equation becomes:

$$m\frac{x_i - 2x_{i-1} + x_{i-2}}{\tau^2} = -\gamma\frac{x_i - x_{i-1}}{\tau} + \sqrt{2k_\mathrm{B}T\gamma}\frac{w_i}{\tau} \,. \tag{1.4.4}$$

From the latter, one can write x_i as:

$$x_i = \frac{2 + \tau/\tau_\mathrm{B}}{1 + \tau/\tau_\mathrm{B}} x_{i-1} - \frac{1}{1 + \tau/\tau_\mathrm{B}} x_{i-2} + \frac{\sqrt{2k_\mathrm{B}T\gamma}}{m(1 + \tau/\tau_\mathrm{B})}\tau w_i \,, \tag{1.4.5}$$

where we can observe that two initial conditions are needed, the first two positions of the particle. Numerically, these positions could be randomly generated or set to 0. If enough statics are generated, it will not affect the results.

[1] The notation w was choosen since in mathematical terms, a real-valued continuous-time stochastic process such as $\mathrm{d}B_t$ is called a Wiener process in honor of Norbert Wiener [30].

1.4.2 Simulating Brownian Motion Using Python

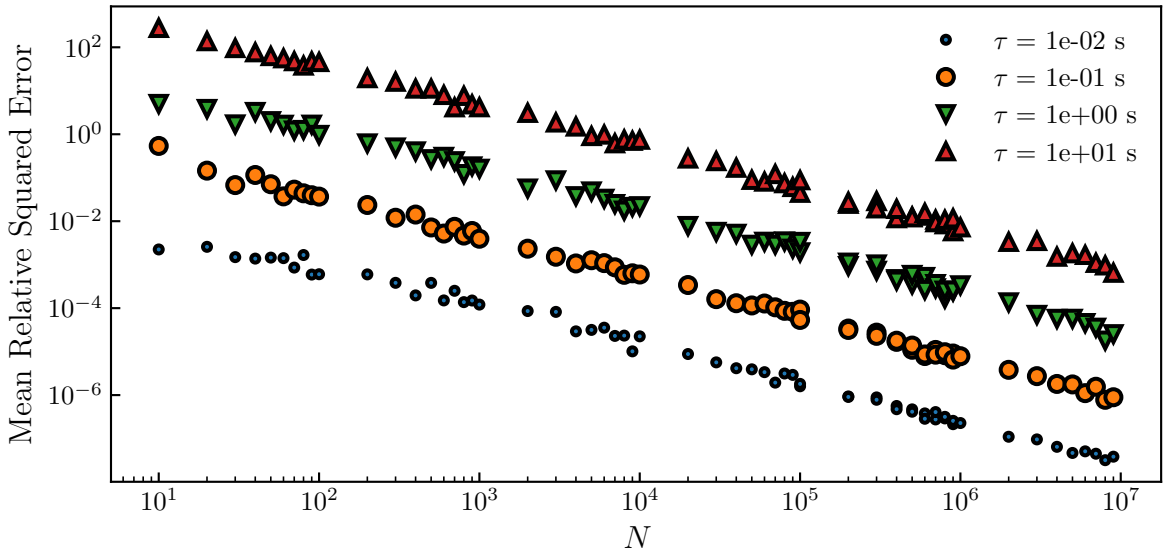


Figure 4: Mean Relative Squared Error (MRSE) of the Probability Density Function PDF measured from a generation of N Gaussian random numbers w_i and the actual Gaussian (see Eq. (1.4.3)) from which the generation is done. The generation is done over a Gaussian which has a mean value $\langle w_i \rangle = 0$ and variance $\langle w_i^2 \rangle = \tau$. We explore parameter ranges from $N = 10$ to 10^7 and $\tau = 10^{-2}$ to 10 s.

Before, diving into the simulation, it could be interesting to ask ourselves how long the simulation should be. Indeed, at equilibrium, the different observables' mean values to remain constant we should wait a sufficient amount of time. It is possible to follow a qualitative approach by generating N numbers w_i, measuring the resulting PDF $P_{\mathrm{c}}(w_i)$ and looking for how much we need to increase N to have $P_{\mathrm{c}}(w_i) \approx P(w_i)$, under some given small-error criterion. As we can see in Fig.3, for simulations made with $\tau = 10^{-3}$ s and $\tau = 1$ s, we observe that as we increase N, the measured PDF, gets closer to the real one given by Eq. (1.4.3).

To have a more quantitative approach, one can compute the Mean Relative Squared Error (MRSE)) between the measured PDF $P_{\mathrm{c}}(w_i)$ and the nominal function $P(w_i)$ as a function of the number N of generated numbers, as:

$$\mathrm{MRSE} = \left\langle \frac{\left(P_{\mathrm{c}}(w_i) - P(w_i)\right)^2}{P(w_i)^2} \right\rangle \Bigg|_N \tag{1.4.6}$$

where the notation $|_N$ denotes the average over N realizations. Additionally, since we measure $P_c(w_i)$ by doing a histogram, the question of how many bins are used should

be answered. It is possible to use the optimal Freedan-Diaconis rule [31] to compute the width of the bins to be used in a histogram. This rule reads:

$$\text{Bin width} = 2\frac{\text{IQR}(\{w_i\})}{\sqrt[3]{N}}, \tag{1.4.7}$$

where IQR is the interquartile range, and $\{w_i\}$ a sample of N random numbers w_i. Moreover, one should at least take 2 bins as a minimum. The optimal number of bins can be computed using the following Python snippet.

```
1
2         import numpy as np
3
4         def _iqr(wi):
5         """Function to compute interquartile range."""
6         return np.subtract(*np.percentile(x, [75, 25]))
7
8         def optimal_bins(wi):
9         """
10        Function to compute the optinal number of bins using Freedan-Diaconis rule.
11        Input: list of random numbers | Output: optimal bins number
12        """
13
14        n = int(diff(wi) / (2 * _iqr(wi) * np.power(len(wi), -1 / 3)))
15
16        if n <= 2:
17        return 2
18        else:
19        return n
```

As we can see in Fig.4, for τ varying between 10^{-2} and 10 seconds, and N between 10 and 10^6, the MRSE decreases as N increases. Moreover, it is interesting to observe that the MRSE is greater as τ increases for a fixed N value. As an example, we would need to only generate $N = 10^{-3}$ numbers to obtain an MRSE of 10^{-4} for $\tau = 0.1$ s, while we would need to $N = 10^6$ for $\tau = 1$ s.

Now that the Langevin equation has been numerically implemented, one could use it to simulate Brownian trajectories. A simple way to do the simulation using Python is provided in appendix.A.1. A set of trajectories simulated for a fictive particle of radius $a = 1$ μm and mass $m = 10$ μg in water is shown in Fig.5-a). For such a particle, the diffusive characteristic time is $\tau_B = 0.53$ s. Moreover, as one can see in Fig.5-b), the

MSD is correctly modeled by Eq. (1.3.26) for $\tau << \tau_{\mathrm{B}}$, and by Eq. (1.3.27) for $\tau >> \tau_{\mathrm{B}}$. Note that for non-continuous data such as the simulation data presented here or sampled experimental trajectories, and for a given time increment Δt, the MSD is generally defined by:

$$\left\langle \Delta x^2 \right\rangle_t = \left\langle (x_i(t+\Delta t) - x_i(t))^2 \right\rangle_t \ , \tag{1.4.8}$$

where the average $\langle\rangle|_t$ is performed over time t.The following Python function can be used to numerically compute the MSD Eq. (1.4.8).

```
1        def msd(x, Dt):
2        """Function that returns the MSD for a list of time indices Dt for a trajectory x"""
3        _msd = lambda x, t: np.mean((x[:-Dt] - x[Dt:]) ** 2)
4        return [_msd(x, i) for i in t]
```

Additionally, as we have seen earlier, the Langevin Equation can be simplified to it is over-damped version of Eq. (1.3.10). In this case, the time step τ of the simulation should be greater than the characteristic time τ_{B}. Thus, if one is interested in the long-time statistical properties of Brownian motion one can use the over-damped Langevin equation. In this case, by putting $m = 0$ into Eq. (1.4.5), one can write x_i as:

$$x_i = x_{i-1} + \sqrt{2D} w_i \ . \tag{1.4.9}$$

The statistical properties at a long time could be retrieved by simulating Brownian motion using the full-Langevin equation. But, since the integration scheme used for Eq. (1.4.5) requires $\tau << \tau_{\mathrm{B}}$, long simulation runs are necessary to retrieve the over-damped properties. A simulation of the over-damped Brownian motion trajectories using Eq. (1.4.9) is shown in Fig.2 and is realized using the following Python Snippet.

```
1        import numpy as np
2
3        N = 1000  # trajectory length
4        D = 1  # diffusion coefficient
5        tau = 0.5  # time step
6        trajectory = np.cumsum(np.sqrt(2 * D) * np.random.normal(0, np.sqrt(tau), N))
```

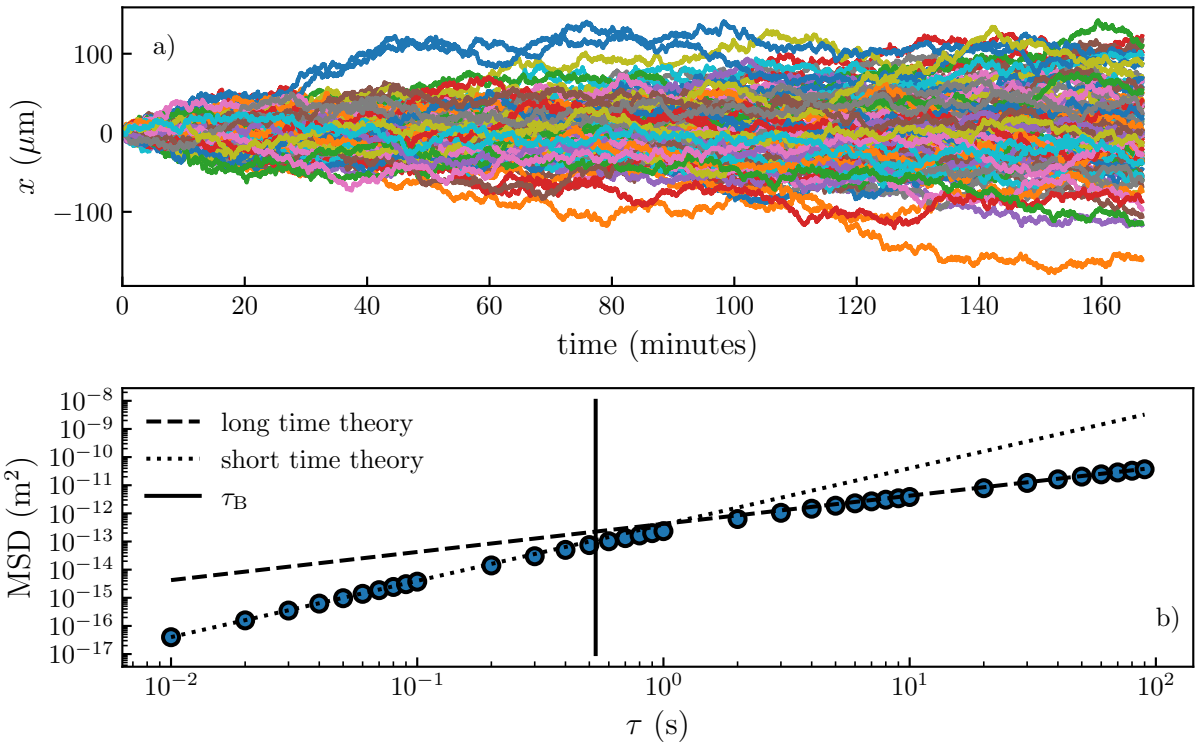


Figure 5: a) Set of 100 trajectories simulated using the full-Langevin equation (see Eq. (1.4.5)) for particles of a radius $a = 1$ μm and mass $m = 10$ μg in water, with viscosity $\eta = 0.001$ Pa.s. The simulations are done with a time step $\tau = 0.01$ s. b) Bullets represent the measured Mean Squared Displacements (MSD) of the simulated trajectories. The plain black line represents the characteristic inertial timescale, $\tau_B = m/\gamma = 0.53$ s. The dotted line represents the MSD in the ballistic regime (see Eq. (1.3.26)), when $t << \tau_B$. The dashed line represents the MSD in the diffusive regime (see Eq. (1.3.27)), when $t >> \tau_B$, MSD $= 2D\tau$. A detailed explanation of the simulation process can be found in appendix.A.1.

1.4.3 Speedup using Cython

I would like to point out that the optimization of a simple simulation of a Brownian trajectory can be interesting. Indeed, using a pure Python code as presented in the first part of appendix.A.1, the simulation of one trajectory of 10^6 steps, needs 6 s to be computed. Thus, more than 10 minutes are required to compute the 100 trajectories shown in Fig.5. This is long and due to how Python systematically verifies what we do is allowed. Indeed, it verifies at each step of the `for` loop the object type of each variable and if the mathematical operation are possible. This is generally the main drawback [32] of interpreted language, and the only solution is to use a compiled language (*e.g* C or Fortran).

The difference between an interpreted (*e.g.* Python) and compiled language (*e.g.* C) lies in the result of the process of interpreting or compiling. A compiler (*e.g.* gcc for

the C language) translate the source code into the computer native language, and create an executable file. The execution of compiled language does not require any more translations, and hence run significantly faster. Contrariwise, an interpreted language is not translated in advance, but is done at the execution, line by line, and each time the program is executed. This process is done by the interpreter, such as Python, Matlab or Perl for their eponymous language. At each execution, the time taken by the interpreter to read and execute each line slow the process, causing execution to take more time.

To overcome this problem with Python, the `cython` package has been developed to translate in C and compile the part of the code that is long to execute, especially the `for` loops. Thus, one can transform is Python source code into a hybrid Python-C code. As presented in appendix.A.1, compiling the `for` loop using `cython` in the full-Langevin simulations reduces the time to generate a 10^6-step trajectory from 6 s to 30 ms thus achieving a speedup factor of 200. Moreover, in the hybrid version, the execution time is limited by the random number generation. Indeed, it takes $\approx$ 24.0ms using `numpy` to generate 10^6 w_i numbers and $\approx$ 6 ms for the trajectory computation. Additionally, as shown at the end of appendix.A.1, even a pure C implementation of the random generation can be slower than the `numpy` one, thanks to `numpy`'s memory optimization. Thus, by using the above mixed-language strategy, the simulation is optimal with the current tools and language at hand.

1.5 Conclusion

In this chapter, we have covered the history of Brownian motion, from the first observation by Robert Brown in the middle of the 19th century to its mathematical and experimental proofs in the early 20th century. We have then described mathematically the bulk Brownian motion and its important statistical properties. Finally, we have used the latter description to simulate Brownian motion using both the full-Langevin equation, and its over-damped version.

2 Particle Characterization and Tracking Using Optical Interferences

2.1 Introduction

Properties of coherent light to produce interference has been widely deployed in metrology for a long time as illustrated by, for example, the famous Fabry-Pérot [33, 34] and Michelson interferometers [35]. The latter was initially employed to evaluate the rotation of the Earth and is still deployed today for the recent measurement of gravitational waves [36]. Since the beginning of the century, interest on tracking and characterizing colloidal particles risen thanks to the democratization of micro fluidics and lab-on-a-chip technologies. In the following, I will provide some insights on the three most used tracking methods:

- Reflection Interference Contrast Microscopy (RICM)
- Lorenz-Mie theory
- Rayleigh-Sommerfeld back propagation

The first one, RICM, uses the principle of optical-path difference in a Michelson interferometer. The other two, use the interferences between the light scattered by the colloid and the incident light. Generally, both sources are collinear, so that we speak of in-line holography.

2.2 Reflection Interference Contrast Microscopy

RICM was first introduced in cell biology by Curtis to study embryonic chick heart fibroblasts [37] in 1964. RICM gained in popularity 40 years after both in biology and physics [38–43]. It was also used recently in soft matter physics to study the elastohydrodynamic lift at a soft wall [44].

When we illuminate a colloid with a plane wave from the bottom, a part of the light is reflected at the surface of the glass substrate and another part, at the colloid surface. The difference of optical paths between two reflections creates an interference pattern. Let us focus on the mathematical description of this phenomenon. In the far field, we can describe two one-dimensional electric field vectors with same angular frequency ω [45] as propagative waves, through:

$$\vec{E}_1(\vec{r},t) = \vec{E}_{0_1}\cos(\vec{k}_1\cdot\vec{r}-\omega t+\epsilon_1)\ , \tag{2.2.1}$$

and:

$$\vec{E}_2(\vec{r},t) = \vec{E}_{0_2}\cos(\vec{k}_2\cdot\vec{r}-\omega t+\epsilon_2)\ , \tag{2.2.2}$$

where the k_i are the wave vector satisfying $|\vec{k_i}| = k = 2\pi n_\mathrm{m}/\lambda$, the angular wavenumber, λ being the illumination wavelength, n_m the optical index of the medium, $\epsilon_{1,2}$ the initial phases of each wave and $\vec{r}$ the position. Here, the origin ($\vec{r}=\vec{0}$) is taken at the position of the first reflection (*i.e.* the glass slide). Thus, on the particle, $\vec{r}$ is given by the particle's height such that $|r| = z$.

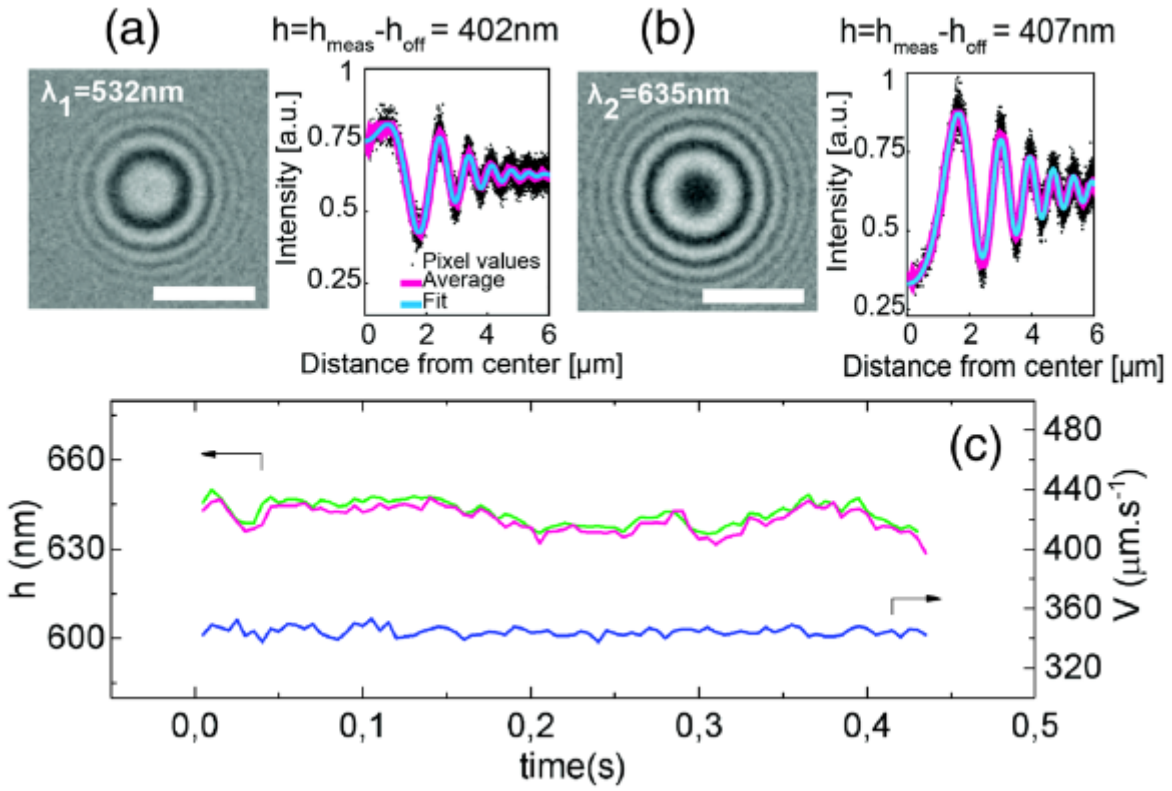


Figure 6: Figure from [44] representing RICM with two wavelengths. (a) Left: interference patterns created with a wavelength $\lambda_1 = 532$ nm (scale bar 5 μm). Right: radial intensity profile (black dots) extracted from the image, azimuthally averaged (magenta line) and fitted with Eq. (2.2.8) to measure the height of the particle (here noted h). (b) Same as (a) with a wavelength $\lambda_2 = 635$ nm. (c) Time series of the height h of a particle (green: λ_1, purple: λ_2) and the particle velocity measured along the flow (in blue).

Experimentally, one measures the intensity of the interference patterns. They can be computed from the time-averaged squared total electric field $\vec{E} = \vec{E}_1 + \vec{E}_2$. The measured intensity is thus given by:

$$I = \langle \vec{E}^2 \rangle = \langle \vec{E}_1^2 + \vec{E}_2^2 + 2\vec{E}_1 \cdot \vec{E}_2 \rangle = \langle \vec{E}_1^2 \rangle + \langle \vec{E}_2^2 \rangle + 2\langle \vec{E}_1 \cdot \vec{E}_2 \rangle \ , \tag{2.2.3}$$

where $\langle \vec{E}_1^2 \rangle$ and $\langle \vec{E}_2^2 \rangle$ are respectively given by I_1 and I_2, the incident intensities. Using trigonometry, we have:

$$\left\langle \vec{E}_1 \cdot \vec{E}_2 \right\rangle = \langle \frac{1}{2} \vec{E}_{0_1} \vec{E}_{0_2} \left[\cos\left(\vec{k}_1 \cdot \vec{r} - \vec{k}_1 \cdot \vec{r} + \phi \right) + \cos\left(2\omega t + \phi'\right) \right] \right\rangle_t \ . \tag{2.2.4}$$

As we average over time, the second cosine vanishes. Thus one has:

$$\langle \vec{E}_1 \cdot \vec{E}_2 \rangle = \frac{1}{2} \langle \vec{E}_{0_1} \vec{E}_{0_2} \rangle \cos\left(\vec{k}_1 \cdot \vec{r} - \vec{k}_2 \cdot \vec{r} + \phi \right) \ , \tag{2.2.5}$$

with ϕ the phase difference between the two fields, which is usually equal to π due to the reflection properties on a higher optical index. Indeed, a colloid has generally a greater optical index than the dilution medium. Finally, the total intensity can be read as:

$$I = I_1 + I_2 + 2\sqrt{I_1 I_2} \cos\left(\vec{k}_1 \cdot \vec{r} - \vec{k}_2 \cdot \vec{r} + \phi \right) \ . \tag{2.2.6}$$

By taking $k_1 = -k_2$ due to the reflection properties, we have:

$$I = I_1 + I_2 + 2\sqrt{I_1 I_2} \cos\left(\frac{4\pi n_{\mathrm{m}}}{\lambda} z + \phi \right) \ . \tag{2.2.7}$$

So far we supposed that the reflection occurs at a unique point; however, we would likely be using spherical colloids. Therefore, illuminating from the bottom, the reflection happens on half of the sphere surface. Moreover, thanks to the spherical geometry the holograms exhibit a radial symmetry, we thus write one can write the radial interference intensity $I(x)$, with x the distance from the pattern center, through [43]:

$$I(x) = A_0 + A_1 \mathrm{e}^{-b_1 x^2} + A_2^{-b_2 x^2} \cos\left[\frac{4\pi n_m}{\lambda} \left(g(x) + z \right) + \phi \right] \ , \tag{2.2.8}$$

Where A_1 and b_1 are parameters [43] that fit the slightly bent background that arises from diffuse reflection on the upper part of the sphere, A_2 and b_2 the decaying contrast

of the higher order maxima, A_0 background intensity, and

$$g(x) = a - \sqrt{a^2 - x^2}\ , \tag{2.2.9}$$

is the sphere profile, to consider the increase sphere-wall as x increases. Finally, this method benefits from equations that are computationally light and enable a quick tracking of particles. However, as we can see in Eq. (2.2.8), because of the periodicity of the cosine, the interference pattern is the same for all heights z separated by a distance $\lambda/(2n_{\mathrm{m}}) \approx 200$ nm for $\lambda = 532$ nm and $n_{\mathrm{m}} = 1.33$.

It is possible to extend this separation to $\simeq 1.2$ μm as used in [44] length by using two different wavelengths. Despite the spatial resolution of this method which can attain 10 nm, the measurement ambiguity is not compatible with the study of Brownian motion due to the periodicity above. Hence RICM it is not usable in our context. As a matter of fact, we experimentally reach height spans of a few microns.

2.3 Lorenz-Mie theory

When a colloid is illuminated with a plane wave, a part of the light is scattered. In consequence, the incident field $\vec{E}_0$ and scattered field $\vec{E}_{\mathrm{s}}$ interferes. The interference patterns thus obtained are called holograms. If the particle is not smaller than the illumination wavelength, it is not possible to use Rayleigh's approximations [46] to describe the scattering. Instead, one needs to use the Lorenz-Mie theory for dielectric spheres. This theory was developed by Lorenz and independently by Mie in 1880 and 1908, respectively [47, 48].

It is in the early 2000s that the Lorenz-Mie theory was first used in order to track and characterize particles [49, 50]. Since then, a lot of studies have been realized with this technique [51]. In the following, I will describe the Lorenz-Mie method. In this part, the height z of the particle is the distance between the particle's center and the focal plane of the objective lens.

Let the incident field be a plane wave uniformly polarized along an axis $\hat{e}$, with an amplitude E_0 and propagating along the $\hat{z}$ direction :

$$\vec{E}_0(\vec{r}, z) = E_0(\vec{r})\mathrm{e}^{ikz}\hat{e} \tag{2.3.1}$$

Let us consider a particle of radius a at a position $\vec{r}_{\mathrm{p}}$. In such case, the scattered field can be written using the Lorenz-Mie theory [45], as:

$$\vec{E}_{\mathrm{s}}(\vec{r},z) = \vec{f}_{\mathrm{s}}(k(\vec{r}-\vec{r}_p))E_0(\vec{r})\exp(-ikz) \ , \tag{2.3.2}$$

with $\vec{f}_{\mathrm{s}}$, the Lorenz-Mie scattering function. The intensity I that we measure at $\vec{r}$ is given by the intensity of superimposition of the incident and scattered amplitudes. Since the measurements are done at the focal plane, i.e. $z=0$, I is given by:

$$\begin{aligned} I(\vec{r}) &= |\vec{E}_{\mathrm{s}}(\vec{r},0) + \vec{E}_0(\vec{r},0)|^2 \\ &= E_0^2(\vec{r}) + 2E_0^2\Re\left(\vec{f}_{\mathrm{s}}(k(\vec{r}-\vec{r}_p))\hat{e}\right) + |\vec{f}_{\mathrm{s}}(k(\vec{r}-\vec{r}_p))|^2 \ . \end{aligned} \tag{2.3.3}$$

Most of the experimental defects on the images are due to spacial illumination variations caused by dust particles. They can be corrected by normalizing the image by the background. In another word, we normalize $I(\vec{r})$ by the intensity of the incident field $I_0 = E_0(\vec{r})^2$ which corresponds the experimental background.

Experimentally, the background can be measured by different methods. One is to have an empty field of view and the other one, which is more convenient, is to compute the median of a stack of images. Additionally, for the latter to work, the video should be long enough for the particle to diffuse sufficiently. If this condition is not satisfied, a ghost of the particle will appear on the background. Moreover, this process permits getting rid of any immobile particles that could generate any additional noise. Examples of hologram before and after the normalization are shown in Figs.7 a-c).

Finally, we write the normalized intensity as:

$$\frac{I(\vec{r})}{I_0(\vec{r})} = 1 + 2\Re\left(\vec{f}_{\mathrm{s}}(k(\vec{r}-\vec{r}_p))\hat{e}\right) + |\vec{f}_{\mathrm{s}}(k(\vec{r}-\vec{r}_p))|^2 \tag{2.3.4}$$

Now that we have the analytical form of the holograms' intensity, it is possible to fit an experimental one to Eq. (2.3.4) as shown in Figs.7 d-e). For the sake of completeness, I will detail the Lorenz-Mie scattering function $\vec{f}_{\mathrm{s}}(k\vec{r})$, which is given by the series:

$$\vec{f}_{\mathrm{s}}(k\vec{r}) = \sum_{n=1}^{n_c} \frac{i^n(2n+1)}{n(n+1)}\left(ia_n\vec{N}^{(3)}_{\mathrm{eln}}(k\vec{r}) - b_n\vec{M}^{(3)}_{\mathrm{oln}}(k\vec{r})\right) \tag{2.3.5}$$

where $\vec{N}_{eln}^{(3)}(k\vec{r})$ and $\vec{M}_{oln}^{(3)}(k\vec{r})$ are the vectorial spherical harmonics, and a_n and b_n are coefficients depending on the particle and illumination properties. For a spherical and isotropic particle of radius a and refractive index n_p, which is illuminated by a linearly polarized plane wave, the a_n and b_n coefficients are expressed in terms of spherical Bessel functions j_n and Hankel functions h_n, as [45]:

$$a_n = \frac{\zeta^2 j_n(\zeta ka) kaj_n'(ka) - j_n(ka)[\zeta kaj_n(\zeta ka)]'}{\zeta^2 j_n(\zeta ka) kah_n^{(1)'}(ka) - h_n^{(1)}(ka)\zeta kaj_n'(\zeta ka)} \ , \tag{2.3.6}$$

and:

$$b_n = \frac{j_n(\zeta ka) kaj_n'(ka) - j_n(ka)\zeta kaj_n'(mka)}{j_n(\zeta ka) kah_n^{(1)'}(ka) - h_n^{(1)}(ka)\zeta kaj_n'(mka)} \ , \tag{2.3.7}$$

where $\zeta = n_\mathrm{p}/n_m$, and where the prime notation denotes differentiation with respect to the argument.

Finally, a hologram is mainly given by the Lorenz-Mie scattering function of Eq. (2.3.5). Moreover, as we can observe in Eqs.2.3.6 and 2.3.7, a hologram depends on a lot of parameters and variables (λ, n_m, n_p, a and $\vec{r}_\mathrm{p}$) . The parameters can be fitted by comparison to experimental data. In general, the illumination wavelength λ and medium index n_m are known and do not need to be fitted. From only one hologram, one can measure the position $\vec{r}_\mathrm{p}$ precisely of the particle and simultaneously characterize the radius and optical index of the colloid. As a side note, it is even possible to characterize a particle without aprioristic knowledge of its characteristics using a Bayesian approach [52, 53].

Computing Eq. (2.3.5) numerically brings another interesting question, as it is analytically written as a sum over n. One could ask after which number n_c of terms the series converges. It has actually been found that the series converges after a number of terms given by [54]:

$$n_\mathrm{c} = ka + 4.05(ka)^{1/3} + 2 \ . \tag{2.3.8}$$

Consequently, the holograms of bigger particles require more terms to converge and, hence, are longer to fit. As an example, the largest particles used during my thesis have a radius $a = 2.5$ μm leading to $n_c = 55$ in water and for an illumination wavelength $\lambda = 532$ nm. For the smallest ones, where $a = 0.5$ μm we find $n_c = 18$ which makes a huge difference in practice. Indeed, if each of the terms of the sum takes the same time to be computed; a 2.5 μm particle's hologram is $55/18 \simeq 3$ times longer to be fitted compared to the

hologram of a 0.5 μm particle.

If a reader wants to evaluate a hologram given by the Lorenz-Mie theory for a peculiar particle and position, it can be done in a few lines with the `holopy` module utilizing the following Python snippet which was employed to make Fig.13 and 14:

```
import holopy as hp
from holopy.scattering import calc_holo, Sphere

sphere = Sphere(n=1.59, r=1.5, center=(4/0.1, 4/0.1, 10))
# n is the optical index of the particle, r its radius in microns
# center is its center position in microns.

medium_index = 1.33
illum_wavelen = 0.532
illum_polarization = (1, 0)
detector = hp.detector_grid(shape=100, spacing=0.1)
# shape is the size in pixels of the camera and the spacing is the pixel's size in microns.

holo = calc_holo(
detector, sphere, medium_index, illum_wavelen, illum_polarization, theory="auto"
)
#the hologram can directly be plotted using:
hp.show(holo)
```

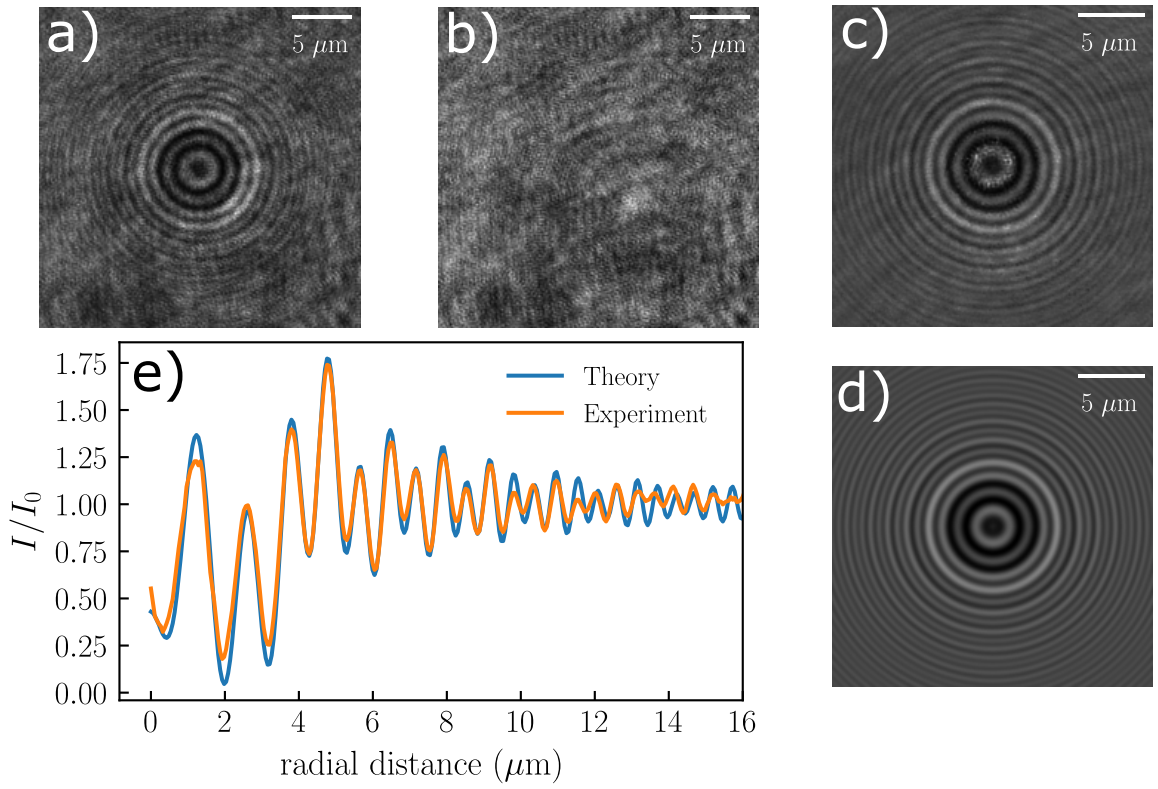


Figure 7: a) Raw hologram of a 2.5 μm polystyrene particle measured experimentally with the setup detailed in the section 2.5. b) Background obtained by taking the median value of an image time series. c) Normalized hologram given by dividing a) by b). d) Result of the fit of c) using Eq. (2.3.4), from which the particle is found to be at a height $z = 14.77\ \mu$m. e) Comparison of the normalized radial intensities, obtained experimentally from c) and theoretically from d).

2.3.1 Hologram dependence on the particle's characteristics

As we can see with the Eq. (2.3.5), the in-line holograms vary with the position, radius and optical index of the particle. For in-line holograms, as both incident and scattered field are collinear, the x and y positions of the particle are given by the center of the hologram. Thus, it is possible to track the motion of a colloid only in two dimensions by using algorithms such as the Hough transforms to find the center. As a side note, in that case, it would be optimal to place the particle just above the focal plane to have an Airy disk-like hologram, as shown in Fig.14 for $a = 2.5\ \mu$m and $z = 5\ \mu$m.

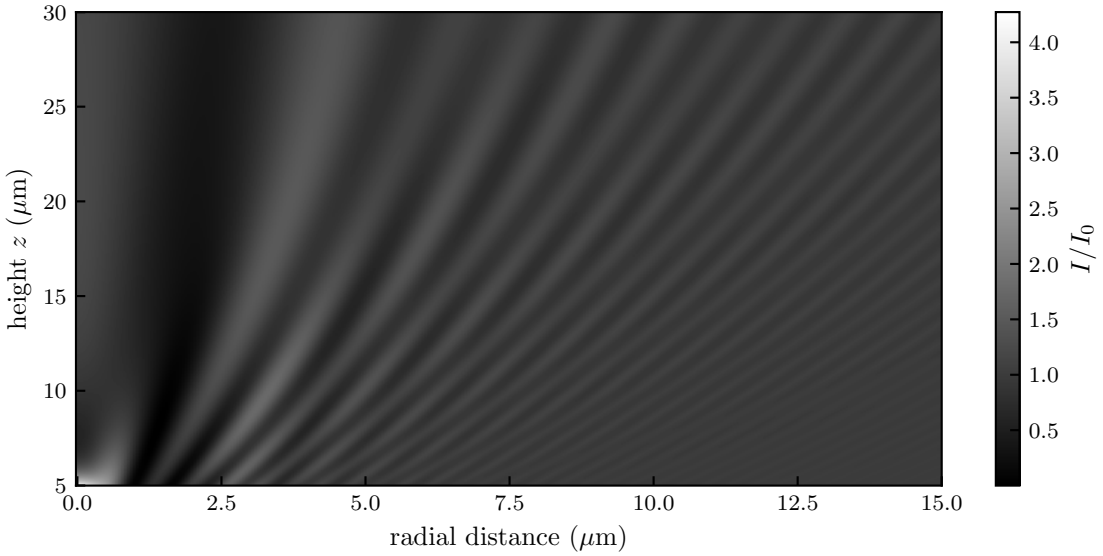


Figure 8: Hologram intensity map in the (r, z)-plan, calculated (see. Eq. (2.3.4)) for a particle of radius $a = 1.5$ μm and optical index $n = 1.59$.

In order to gain some insights on how the holograms vary with the various parameters, one can compute theoretical (see. Eq. (2.3.4)) holograms for particles of different sizes and heights. We start by considering a particle of radius $a = 1.5$ μm, and optical index $n_{\mathrm{p}} = 1.59$ as shown in Fig.8. In this case, one can observe that as the distance z between the particle and the focal plane increases, the hologram's rings get wider.

Additionally, this thickening of the rings can also be observed in the Fig.12, where hologram intensity profiles are plotted as a function of the height z both theoretically and experimentally for a polystyrene colloidal particle of radius $a = 1.5$ μm, and for different couples of parameters in Fig.14.

Also, we note that if z is not large enough compared to the radius of the particle, the center of a hologram can be so bright that the rings could not be seen if the camera does not have a large enough dynamic range. Thus, for having an optimal condition for the fits, one should take care of defocusing enough the objective lens to have $z >> a$.

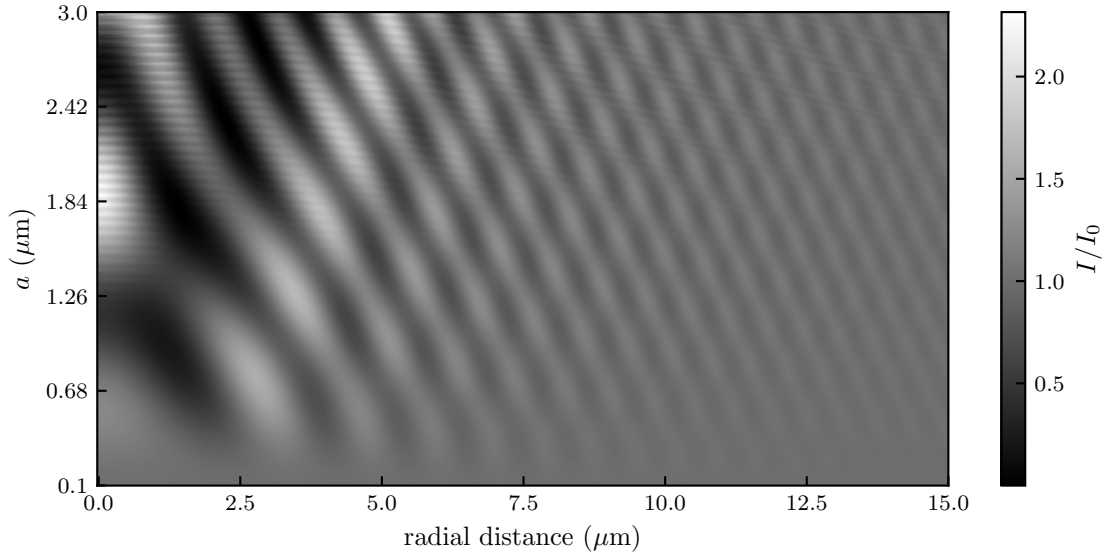


Figure 9: Hologram intensity map in the (r, a)-plan, calculated (see. Eq. (2.3.4)) for a particle of optical index $n = 1.59$, and a distance $z = 15$ μm between the particle center and the focal plane of the objective lens.

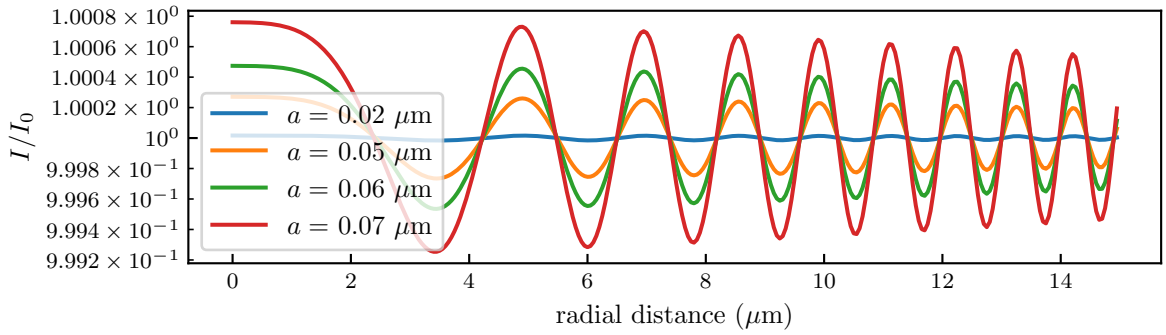


Figure 10: Radial intensity profile for particle radius $a << \lambda$, and an optical index $n_{\mathrm{p}} = 1.59$ with a distance $z = 15$ μm between the particle center and the focal plane of the objective lens, and for a wavelength $\lambda = 532$ nm.

We can now look at the holograms variation with respect to the radius of the particle as shown in Fig.9 for a particle of optical index $n = 1.59$ and at a distance $z = 15$ μm. One can observe that for small particles compared to the wavelength, *i.e.* $a << \lambda$, we do not observe the rings. This is due to the fact that for the small particles, the scattering can be approximated using the Rayleigh theory in which the scattering is isotropic. Thus, the variation of intensity around I_0 will be smaller for smaller particles.

Also, in this small-particle regime, the particle size does not affect the general shape of the hologram but just its intensity as shown in Fig.10, for particles of radii between $a = 0.02$ μm and $a = 0.07$ μm, and for a wavelength $\lambda = 532$ nm.

Additionally, since the signal-to-noise ratio is lower than for larger particles, it is less precise to characterize small colloids compared to the wavelength.

As the particle gets bigger, the scattering becomes anisotropic and is mostly oriented towards the incident plane-wave direction. This effect leads to an increase of the amplitude I/I_0 of the rings, as one can see in Fig.9. Thus, the signal-to-noise ratio is high enough to easily discern the hologram on top of the noise as one can see on the experimental picture of Fig.7-a). One who wants to employ this method should thus use large enough particles for the hologram intensity to be greater than the camera noise level.

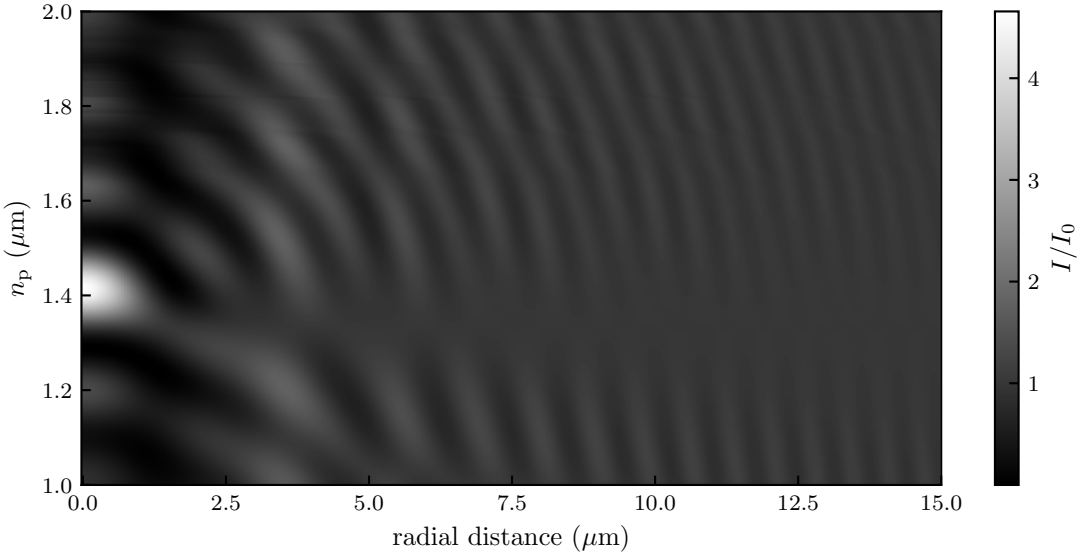


Figure 11: Hologram intensity map in the (r, a)-plan, calculated (see. Eq. (2.3.4)) for a particle of optical index $n = 1.59$, and a distance $z = 15$ μm between the particle center and the focal plane of the objective lens.

Finally, one can check how the holograms are varying with the optical index of a particle. In this case, it is not the particle's optical index n_p which matters the most but the ratio $\zeta = n_\mathrm{p}/n_\mathrm{m}$ which can be found in the a_n and b_n formulas, in Eqs.2.3.6 and 2.3.7. Indeed, for the scattering to happen, the optical index n_p of the colloid needs to be different from the optical index n_m of the surrounding medium. Additionally, the numerical solution of the Lorenz-Mie framework is not stable for $n_\mathrm{p} \simeq n_\mathrm{m}$. In Fig.11, we can observe holograms of a particle of radius $a = 1.5$ μm at fixed distance $z = 15$ μm between the particle and the focal plane of the objective lens with a varying colloid's optical index, in water where $n_\mathrm{m} = 1.33$. In Fig.11, one can thus observe that for $n_\mathrm{p} \simeq n_\mathrm{m}$ we do not see any holograms. Additionally, one can observe that the signal-to-noise ratio gradually increases as n_p becomes different from n_m. One who wants to use this technique should thus have n_m different enough from n_p for the hologram intensity to be greater than the camera

noise level.

2.3.2 Summary on the Lorenz-Mie method

A given set of height, optical index and radius of a colloid thus gives unique holograms. Conversely, this uniqueness of the holograms permits precise extraction of the position, optical index and radius of a colloid. Holograms for different sets of parameters are shown in Figs.13 and 14. Additionally, the interested reader can use the Jupyter Notebook on my Github repository in order to plot any hologram .

Finally, the Lorenz-Mie framework provides the most versatile in-line holographic method. Indeed, it permits tracking and characterizing unique particles even without a priori knowledge on its characteristics. Besides, it is possible to write the Lorenz-Mie function $\vec{f}_{\mathrm{s}}$ for particular cases such as anisotropic particles [55, 56] or particle clusters [55, 57] to name a few. Such possibilities pave the way to a lot of experimental studies. Additionally, the method allows reaching a really high precision as the tenth of nanometers on the position and radius as well as 10^{-3} on the optical index [50].

Unfortunately, the Lorenz-Mie framework suffers from a major drawback which is the time needed to fit one image. For example, a 200 by 200 pixels image, of a 2.5 μm particle's hologram, can take up to two minutes to be fitted using a pure and straightforward Python algorithm. A lot of work has been done to permit faster tracking, such as random-subset fitting [58], GPU (graphical processing units) acceleration, machine learning [59, 60] and deep neural networks [61].

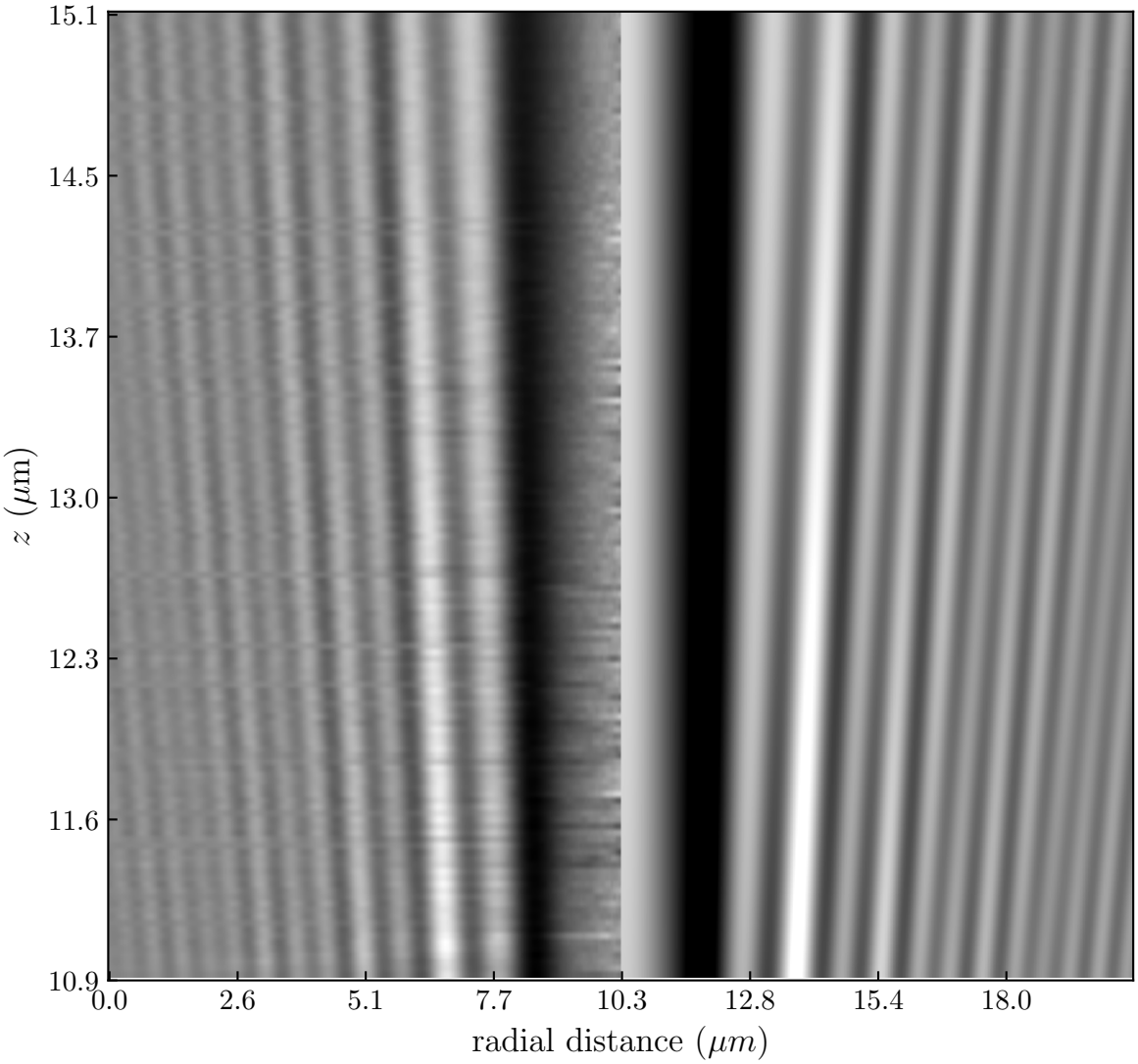


Figure 12: Hologram intensity map in the (r, z)-plan, calculated (see. Eq. (2.3.4)) for a particle of optical index $n = 1.59$, and radius $a = 1.51$ μm using the experimental setup presented in the section 2.5. On the right, the corresponding theoretical intensity using the result of each individual hologram's fit to Eq. (2.3.5).

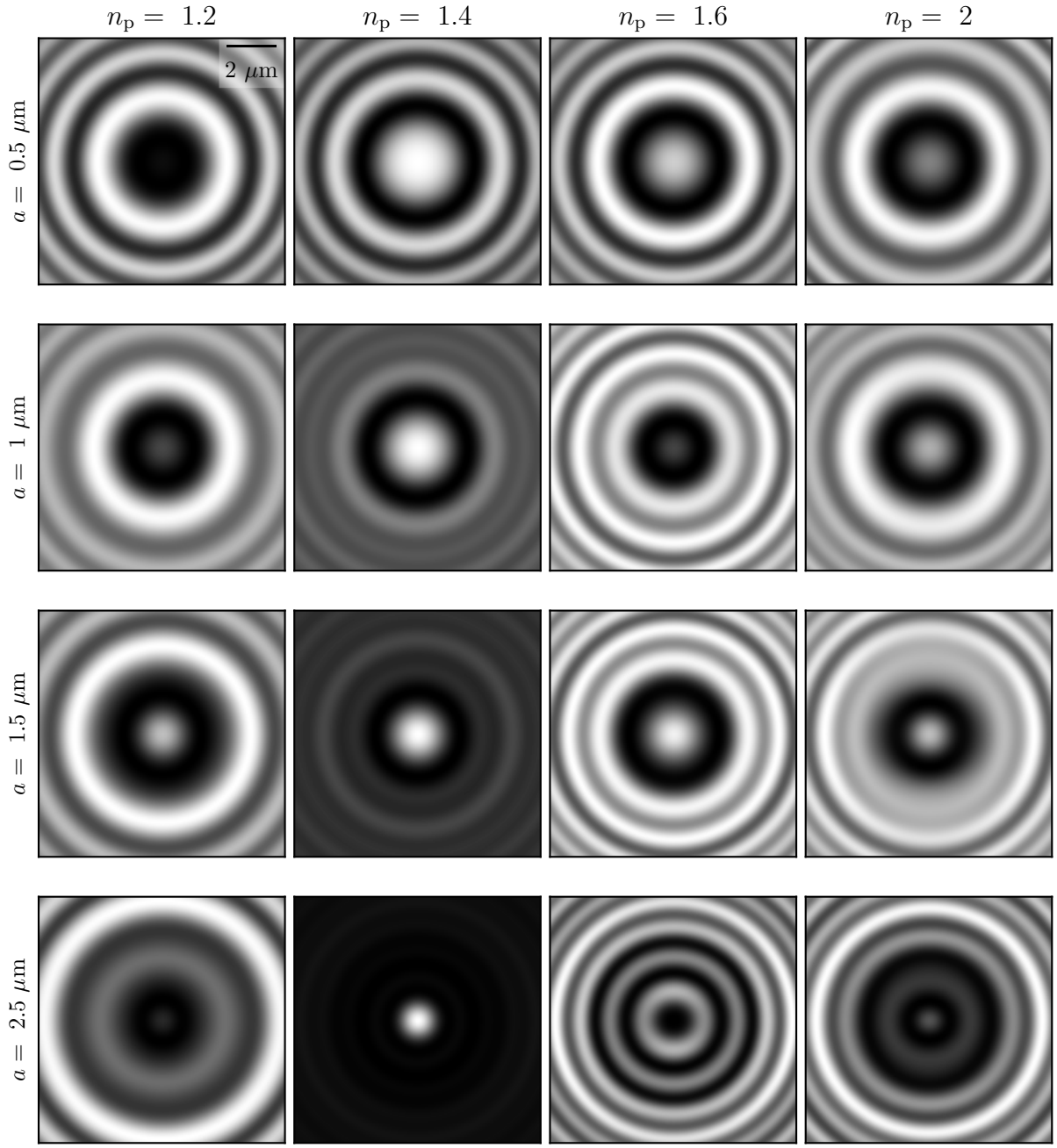


Figure 13: Holograms calculated (see. Eq. (2.3.4)) for different set of parameters (a, n_p), and for a distance $z = 15$ μm between the particle center and the focal plane of the objective lens.

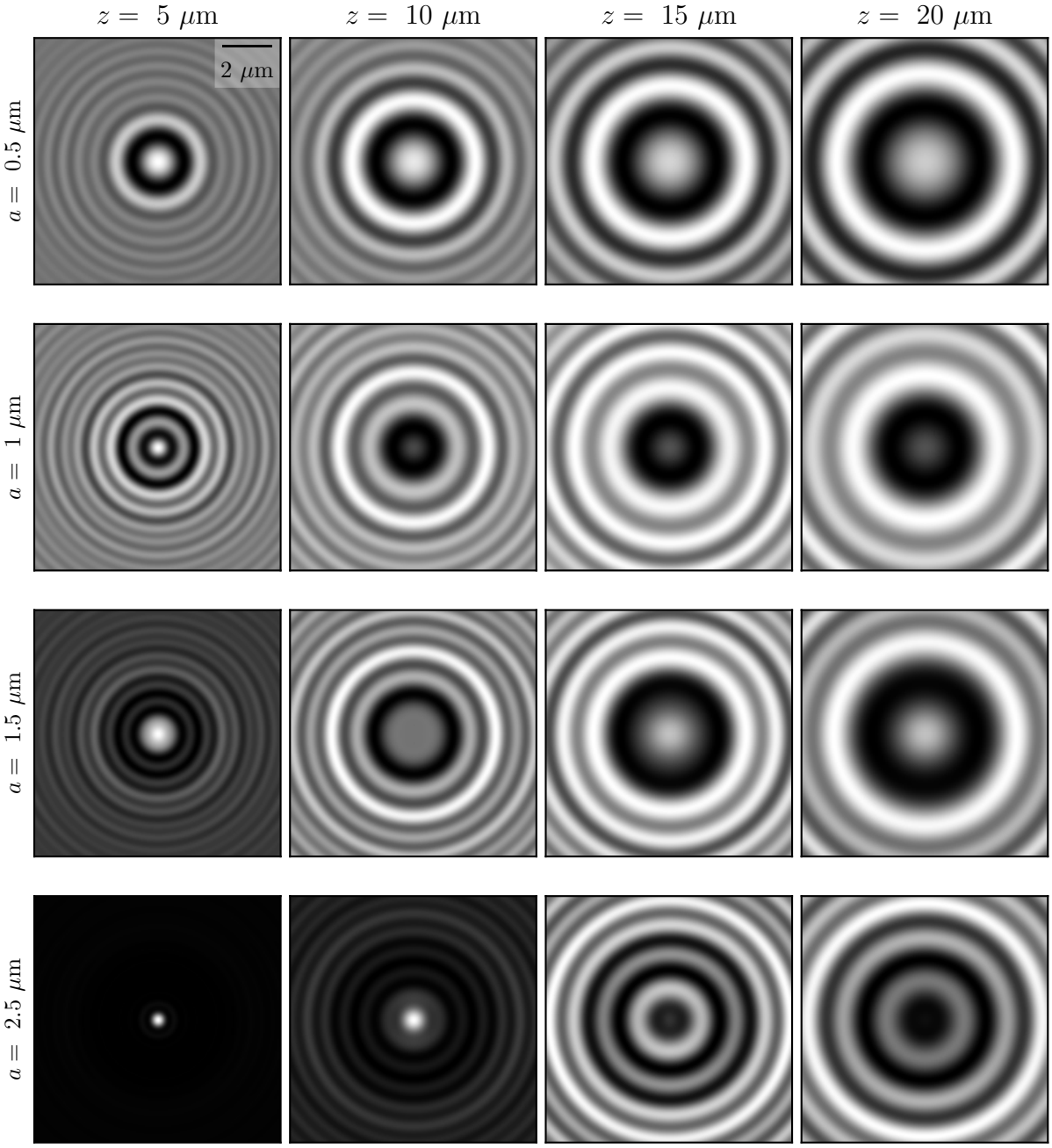


Figure 14: Holograms calculated (see. Eq. (2.3.4)) for different set of parameters (a, z), and for an optical index $n_{\mathrm{p}} = 1.59$

2.4 Rayleigh-Sommerfeld back propagation

Rayleigh-Sommerfeld back propagation [62] works on the same principle as the Lorenz-Mie scattering but assumes small scatterers, and, a low difference of optical indices, such that:

$$|\zeta - 1| << 1 \text{ and } ka|\zeta - 1| << 1 \ . \tag{2.4.1}$$

In this case, at the focal plane, the intensity of the scattered field is smaller than the intensity of the incident field, hence, the term $|\vec{E}_\mathrm{s}|^2$ can be ignored. Thus, the Eq. (2.3.4) can be rewritten as:

$$\frac{I(\vec{r})}{I_0(\vec{r})} = 1 + 2\Re\left(\frac{E_\mathrm{s}(\vec{r},0)}{E_0(\vec{r})}\right) \ . \tag{2.4.2}$$

If one can retrieve the scattered field completely from an image, it is possible to reconstruct it above the focal plane by convolution using the Rayleigh-Sommerfeld propagator [63]:

$$h_{-z}(\vec{r}) = \frac{1}{2\pi}\frac{\partial}{\partial z}\frac{\mathrm{e}^{ikR}}{R} \ , \tag{2.4.3}$$

where $R^2 = r^2 + z^2$ and the sign convention on the propagator indicates that the particle is above the focal plane. Using this propagator we have:

$$E_\mathrm{s}(\vec{r},z) = E_z(\vec{r},0) \otimes h_{-z}(\vec{r}) \ . \tag{2.4.4}$$

By using the convolution theorem [63–66] and supposing a uniform illumination, one can approximately reconstruct the scattered field at height z as:

$$E_\mathrm{s}(\vec{r},z) \approx frac\mathrm{e}^{ikz}4\pi^2\int_{-\infty}^{\infty} B(\vec{q})H(\vec{q},-z)\mathrm{e}^{i\vec{q}\cdot\vec{r}}\mathrm{d}^2q \ , \tag{2.4.5}$$

where $B(\vec{q})$ is the Fourier transform of I/I_0 and where:

$$H(\vec{q},-z) = \mathrm{e}^{iz\sqrt{k^2-q^2}} \ . \tag{2.4.6}$$

Finally, using Eq. (2.4.5) one can reconstruct the scattered field and intensity since $I(\vec{r}) = |E_s(\vec{r})|^2$, as shown in Fig.15. Moreover, by finding the position where we have an inversion of the center from bright to dark in Fig.15, we measure the position of the particle. These equations are way less computationally expensive than Eq. (2.3.5). Thus tracking can be faster.

Additionally, as Eq. (2.4.5) takes only into account the intensity of the image, this method does not require any information on the particle and number of particles. As a matter of fact, to write Eq. (2.4.5), one just needs to assume that we have spherical colloids. Thus, this method is powerful to reconstruct the 3D position of a lot of particles or clusters. However, the Rayleigh-Sommerfeld back propagation suffers from being less precise of the presented methods and we cannot use it to characterize the particles generating the holograms.

2.4.1 Numerical Rayleigh-Sommerfeld back propagation

The `holopy` Python module provides a set of methods that permit implementing the Rayleigh-Sommerfeld back propagation. Given the `hologram` variable containing all the needed metadata about the hologram such as the pixel size, medium index n_n, illumination wavelength λ. Then, one can use the `propagate` method to back propagate a hologram over a set `zstack` of height using the following Python snippet.

```
1  import holopy as hp
2  import numpy as np
3
4  zstack = np.linspace(0, 20, 11)
5  rec_vol = hp.propagate(holo, zstack)
```

Note that using the `propagate` function, each propagation is done by performing a convolution of the reference hologram over the distance to be propagated. However, better reconstruction can be obtained iteratively by propagating holograms over several short distances. The latter method is called Cascaded Free Space Propagation, and is particularly useful when the reconstructions have fine features or when propagating over large distances [67]. It can be done by specifying the argument `cfsp` to the `propagate` method. For example, to change the source of the propagation every three steps, one can use `hp.propagate(holo, zstack, cfsp=3)`.

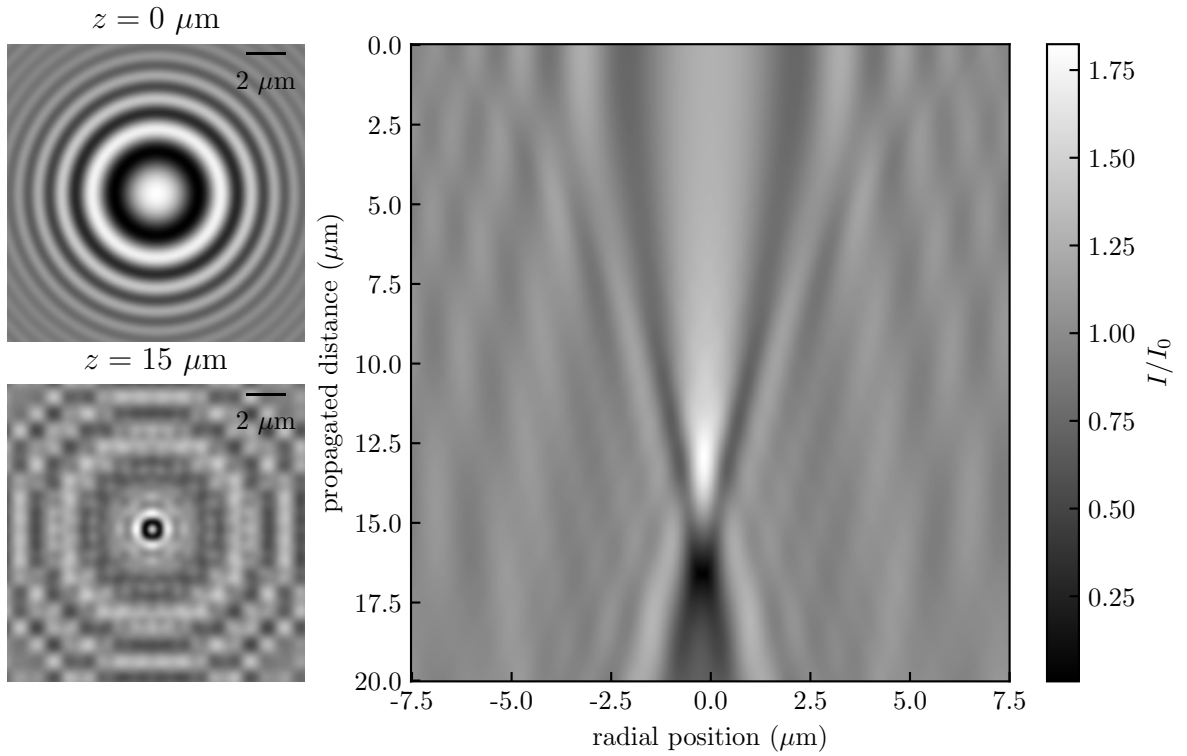


Figure 15: On the left: the original hologram on the top and propagated along 15 μm on the bottom. On the right: reconstruction using Eq. (2.4.5) of the scattered intensity by a single colloidal sphere of radius $a = 0.1$ μm, with optical index $n_{\mathrm{p}} = 1.59$ in water whose index is $n_{\mathrm{m}} = 1.59$, and for a height of 15 μm.

2.5 The experimental setup

The experimental setup I developed during my PhD can be employed to implement both the Lorenz-Mie and Rayleigh-Sommerfeld back propagation methods. In order to observe the holograms, we use a homemade inverted microscope as shown in Fig.16 and schematized in Fig.17. This microscope is built using a ThorLabs cage system. Using the microscope, we observe the holograms resulting from the interactions between a laser source and the beads present in a sample.

A sample consists of a parallelepipedic chamber (1.5 cm $\times$ 1.5 cm $\times$ 150 μm), made from two glass covers, a parafilm spacer, and sealed with vacuum grease, containing a dilute suspension of spherical polystyrene beads. Sealing the sample with vacuum grease permits to drastically decrease evaporation, which reduces the possible evaporation driven-flow in the sample.

We used 3 different colloidal sizes, of nominal radii 0.56 μm, 1.5 μm and 2.5 μm, at room temperature T, in distilled water (type 1, MilliQ device) of dynamic shear viscosity $\eta = 1$ mPa.s. The particles are made of polystyrene of density $\rho = 1050$ kg.m^{-3} and

optical index $n_p = 1.598$ at a wavelength of 532 nm.

The sample is illuminated by a collimated laser beam with a 532 nm wavelength. The laser used delivers a power of 4 W and has a centimetric waist. Since the laser is collimated, it has a near-zero exentricity so that it can be seen as a plane wave. As presented in the section 2.3, the light scattered, by one colloidal particle at a given time t, interferes with the incident beam.

An oil-immersion objective lens (x60 magnification, 1.30 numerical aperture) collects the resulting instantaneous interference pattern, and relays it to a camera (Basler acA1920-155um) with a 51.6 nm/pixel resolution (see Fig.7-a)). The exposure time of the camera is set to $\tau_{expo} = 3$ ms to avoid motion-induced blurring of the image. As a general rule, the particle should not diffuse more than the pixel size during that time, such that $\sqrt{2D\tau_{expo}} < 51.6$ nm.

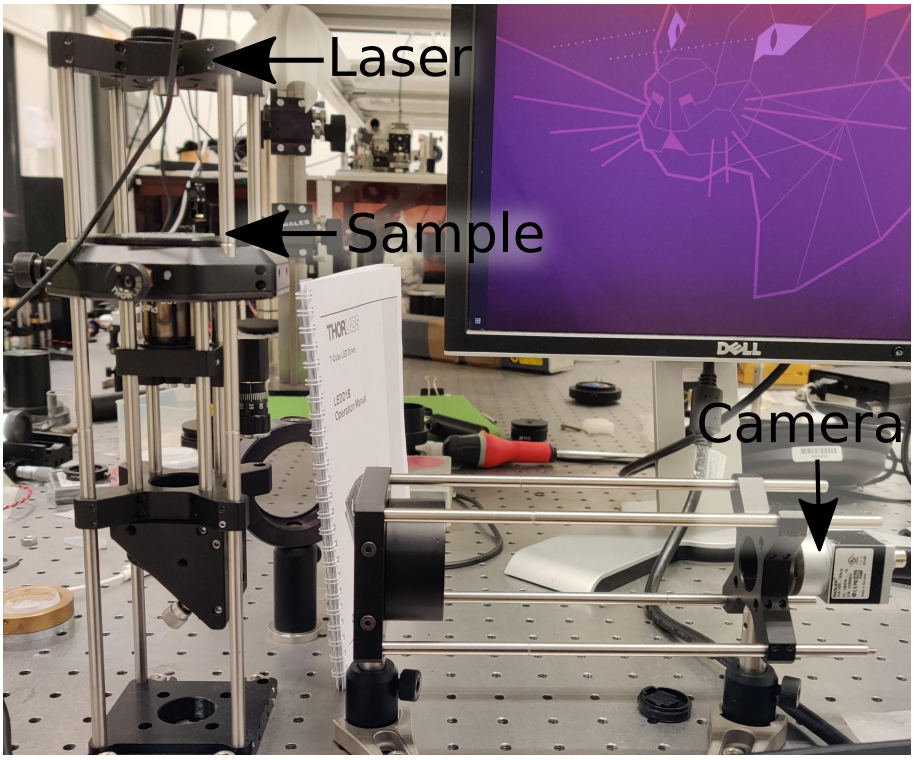


Figure 16: Photo of the custom-built microscope developed in my thesis. It is composed of a Thorlabs cage system. The camera used is a Basler acA1920-155um. We use a x60 magnification and 1.30 numerical aperture oil-immersion objective lens. The light source is a collimated 521 nm wavelength laser.

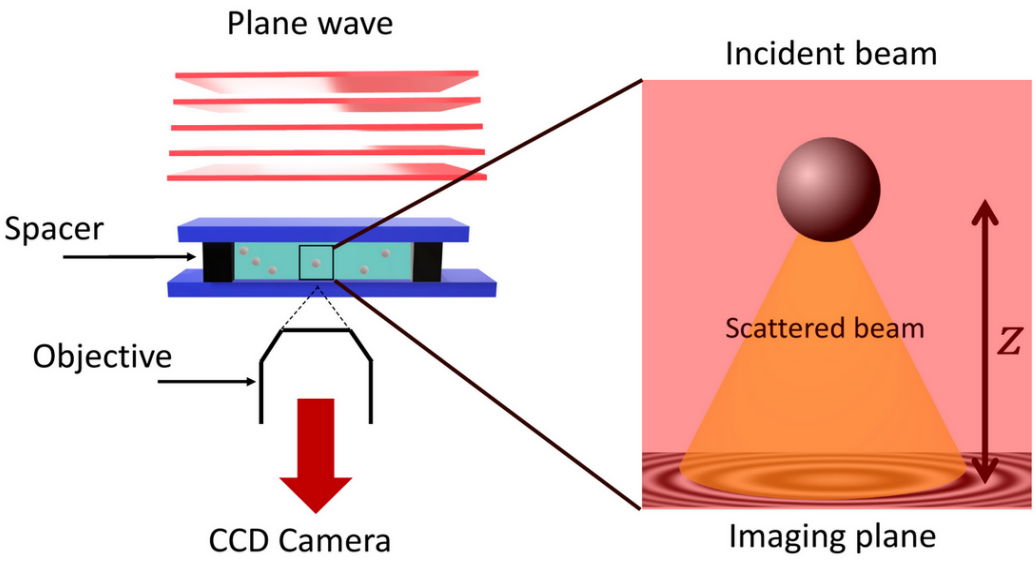


Figure 17: Schematic of the experimental setup. A laser plane wave of intensity I_0 illuminates the chamber containing a dilute suspension of microspheres in water. The light scattered by a particle interferes with the incident beam onto the focal plane of an objective lens, which magnifies the interference patten and relays it to a camera.

2.6 Optical forces

As we illuminate particles with a laser, it is important to know if the optical forces that arise from the interactions between the light and the particle needs to be taken into account. When a plane wave is incident on a sphere, it scatters and absorbs light. This process depends both on the light wavelength λ and on the sphere properties, its radius a and refractive index $n_\mathrm{p} = n_r - jn_i$. For polystyrene $n_i \ll 1$, as shown in Fig.18 such that we neglect it in the Lorenz-Mie framework. However, computing the optical forces, and hence, the light quantity which is absorbed requires n_i, so we consider it this section. Additionally, in the Mie theory, the particle is characterized by the size parameter $\tilde{x} = 2\pi a/\lambda$. The optical force F_opt is given by [45]

$$F_\mathrm{opt} = \frac{I_\mathrm{r} n_\mathrm{m} \pi a^2}{c}(Q_\mathrm{ext} - gQ_\mathrm{sca}) \ , \tag{2.6.1}$$

where I_r is the irradiance in W.m^{-2} on the sphere, c is the speed of light in vacuum, $g = \pi r^2$ the sphere cross section and Q_ext and Q_sca being respectively the extinction and scattering efficiency given by:

$$Q_{\text{ext}} = \frac{2}{k^2 a^2} = \sum_{n=1}^{\infty} (2n+1)(|a_n|^2 + |b_n|^2) \ , \tag{2.6.2}$$

and,

$$Q_{\text{ext}} = \frac{2}{k^2 a^2} = \sum_{n=1}^{\infty} (2n+1)\Re(a_n + b_n) \ , \tag{2.6.3}$$

where the a_n and b_n coefficient are given by the Eqs.2.3.6 and 2.3.7 respectively.

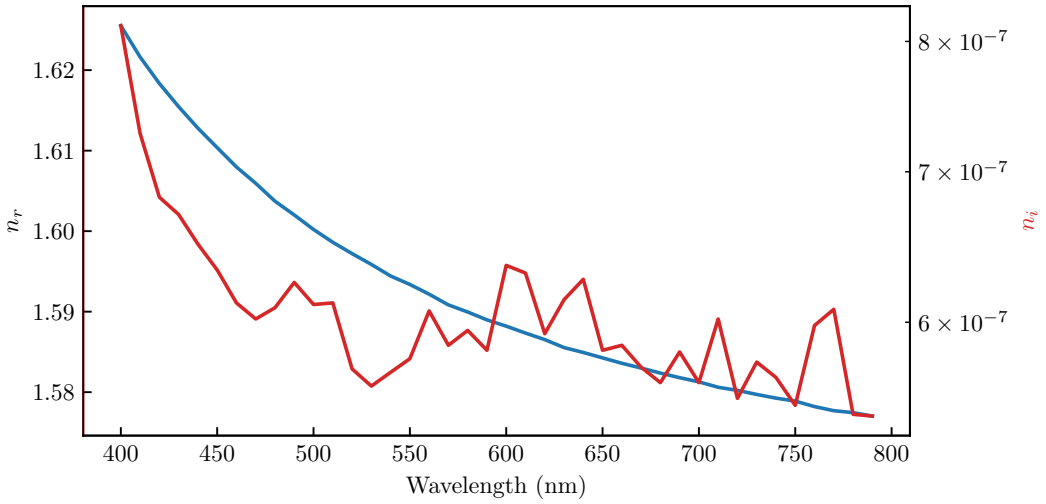


Figure 18: Real (left axis) and imaginary (right axis) part of the refractive index of polystyrene as a function of the incident wavelength. Data obtained from [68]

To compute the optical force I personally employ the `miepython` python's module and retrieving the optical index data from the refractiveindex.info website, using the following Python snippet.

```
import miepython as mp
import numpy as np

# Download the data on the refractiveindex.info website
poly = np.genfromtxt(
r"https://refractiveindex.info/tmp/data/organic/(C8H8)n%20-%20polystyren/Zhang.txt",
delimiter="\t",
)
N = len(poly) // 2
```

```
        poly_lam = poly[1:N, 0]  # wavelength
        poly_nre = poly[1:N, 1]  # real part
        poly_nim = poly[N + 1 :, 1]  # imaginary part

        x = 2 * np.pi * a / poly_lam
        n = poly_nre - 1.0j * poly_nim
        qext, qsca, qback, g = mp.mie(n, x)  # compute the efficiencies
        E0 = 4.5e-3 / (np.pi * 1.75e-3 ** 2)  # compute the irradience
        c = 299792458 / 1.33  # light velocity in the  medium

        F = E0 * np.pi * r0 ** 2 * (qext - g * qsca) / c  # compute the optical force

```

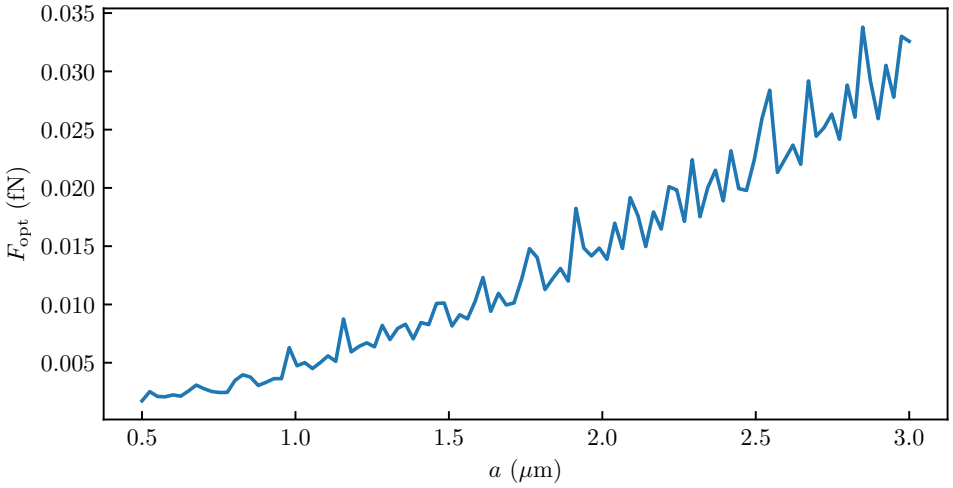


Figure 19: Optical force F_{opt} (see Eq. (2.6.1)) exerted on a spherical particle of radius a by a plane wave of wavelength $\lambda = 532$ nm, and of irradiance $I_{\mathrm{r}} = 467.7$ W.m^{-2}. The force is calculated employing the `miepython` python's module and the refractive index of polystyrene [68].

Using Python, one can thus compute the optical force F_{opt} as a function of the particle radius. As shown on Fig.19 the optical exerted on the colloidal particle from the microscope's illumination source is of the order of magnitude $F_{\mathrm{opt}} \simeq 10^{-2}$ fN. By comparing this result with the buoyant of a colloidal particle F_{g}:

$$F_{\mathrm{g}} = \frac{4}{3}\pi(\rho_{\mathrm{m}} - \rho_{\mathrm{p}})g \ , \tag{2.6.4}$$

we found that for a particle of radius $a = 0.5$ μm, $F_{\mathrm{g}} = -0.3$ fN and for $a = 1.5$ μm, $F_{\mathrm{g}} = -7$ fN. Thus, we can conclude that $F_{\mathrm{opt}} \ll F_{\mathrm{g}}$, hence, in the following of the manuscript we neglect the optical force as it is lower than the other external forces acting

on the colloids. However, experiments with a controlled irradiance (up to 10^6 W.cm^{-2}) and they were able to measure the optical force from the motion statistics [19].

2.7 The experimental procedure

We implement here the Lorenz-Mie fitting method, since it permits the characterization of single particles. Indeed, since we are interested in fine effects near surfaces, we require to know perfectly the radius of the tracked particle. This feature also makes our whole procedure calibration-free, as we do not need to assume any physical properties. An example of the procedure that permits tracking a single-particle trajectory is provided in appendix A.2. In the following, the different steps of the procedure are described.

2.7.1 Recording the holograms

Since we use a Basler camera, we use the provided computer program (Pylon) by the manufacturer in order to record the holograms. The software permits adjusting the parameters of the camera, such as the region of interest (ROI),number of frames per second (fps) or the opturation time to name a few. Also, video can be recorded as a time series of images, in AVI or MP4 formats. AVI files or times series are a great way to save the video since it is lossless. However, in general, we use a ROI of 1000×1000 pixels to record the particle during a long-enough time.

Additionally, since the recording is done using 8 bits per pixel (or 256 gray levels), an image of 1000×1000 pixels needs a disk space of 1 MB [2]. One can see that image sequences and AVI files are not suitable for our case because i) at 100 fps one would need 108 GB to store a 30-minute film, which would lead to several TB of data per experiment which is not manageable; ii) it would require a sequential writing speed of 60 MB/s, which is just below the limit of the better Hard Drive Disks. iii) AVI files are bound to 2 GB maximum thus dramatically reducing the length of the experiments.

To conclude, for all of these reasons, we chose to use the MP4 file format (MPEG-4 encoding) for the video recording. Using the lowest compression, we did not observe any impact on the fitting process due to quality loss. Finally, a video of 30 minutes has an approximate size of 3 GB.

[2] An uppercase B denotes Byte which is equivalent to 8 bits denoted by b, *i.e.* $1\ B = 8\ b$. For storage indications, Bytes are generally used, since historically a set of 8 bits encodes a single text character, and are for this reason the smallest addressable memory units in most computer architectures. As an example, in binary "LOMA" would be encoded by "01001100 01001111 01001101 01000001."

2.7.2 Fitting the holograms

Once the holograms are recorded, we fit all of the images to retrieve the trajectory of the particle. To do so, we chose to use the `pylorenzmie` module developed by the Grier's lab at New York University. Although this module presents a lot of capabilities, it is not adapted to MP4 input. Thus, I developed a wrapper[3] around `pylorenzmie` that I called Wraplorenzmie which can be found on my Github repository .

This wrapper permits to directly load the MP4 files, compute the background and choose if what parameters should be fitted. Also, it manages the process fitting a time series of images by using results of previous image as initial fit parameters.

However, as presented in the section 2.3, the main drawback is the time to fit an image. Indeed, using a Python algorithm, one needs 30 seconds to fit images of 100×100 pixels and a few minutes for a 500×500 pixels hologram. We can directly see a bottleneck, if one wants to track one trajectory made of 100 000 images .In such case, one would need to typically 70 days for a series of images that needs only a few minutes to be recorded experimentally.

When I started my PhD, two groups, the Grier's lab and the Manoharan's lab, had already created Python packages, respectively Pylorenzmie and Holopy, in order to inverse holograms. They had introduced ways to only fit a set of randomly chosen pixels, and demonstrated that taking only 1 % of the image pixels, could lead to similar precision thus improving considerably the fit's execution time [58].

Unfortunately, even if fitting a random subset of pixels is faster, it leads to a few images per second, and is still too long for the amount of data we want to have. This part of my thesis is certainly the one where I spent the longest time, and I learned a lot about code optimization and computer cluster usage.

In the middle of my thesis, `pylorenzmie` got a new commit[4] on the authors' Github repository which was saying that the authors succeeded in using GPU acceleration with CUDA[5]. This was not an easy task since they needed to reconstruct the Bessel functions

[3] A wrapper is a code that encapsulates or "wraps" another code to make it easier to use. For example, it is particularly useful to adapt a program to a particular type of input data. Creating wrapping function is a commonly done by developers add some abstraction to the source codes and readability.

[4] A commit is an update of the files on a Git repository.

[5] CUDA is the acronym of Compute Unified Device Architecture. It is a parallel computing platform, and programming model made to permit an easier use of the GPU for general purposes. CUDA is developed by NVIDIA since 2012, thus all recent NVIDIA's GPUs are CUDA-enable. It is possible to use CUDA with every language as long as a library has been developed, such as `cupy`for Python .

in an understandable way for the GPU. Fortunately, it is possible to do so by using continued fractions [54]. This appreciable update permits fitting whole images with a speed improvement of 20 fps. At this speed, we precisely extract the tridimensional position of the particle, as well as the radius and optical index.

As a remark, the fits are done by solving a least-squares problem using the Levenberg-Marquardt algorithm [69]. This algorithm is largely used in curve-fitting applications due to its capabilities to find a minimum even by starting far from it. As mentioned, it is also possible to use various models of Machine Learning or Deep Learning to do the fits [61]. However, since we can write analytically the holograms, the Deep Learning's models cannot be more accurate than a standard least-square-fitting process. Deep Learning, however, could be a great option is one wanted to prioritize the computation time over the fit's precision.

Finally, to have a more reliable and fast-tracking, we begin by fitting the first 10 000 images with $\vec{r}_{\mathrm{p}}$, a and n_{p} as free parameters. Using the results of this fit, we can characterize the physical properties of the tracked colloid with high precision. Then, using these results we can then fit image with only the position $\vec{r}_{\mathrm{p}}$ as a free parameter.

2.7.3 Radius and optical index characterization

Using 10 000 measurements of a and n_{p} one can do a 2D histogram, as presented in Fig.20 here smoothed using a Gaussian Kernel Density Estimator. Doing so, we determine that the radius of the observed particle is $a = 1.514 \pm 0.003$ μm and its optical index is $n_{\mathrm{p}} = 1.585 \pm 0.002$.

Finally, as explain above, using this measurement of the radius and optical index, we then fit the whole video by removing them from the free parameters. Doing so, we measure the 3D trajectory of the particle as shown in Fig.21 in tridimension for the particle previously characterized.

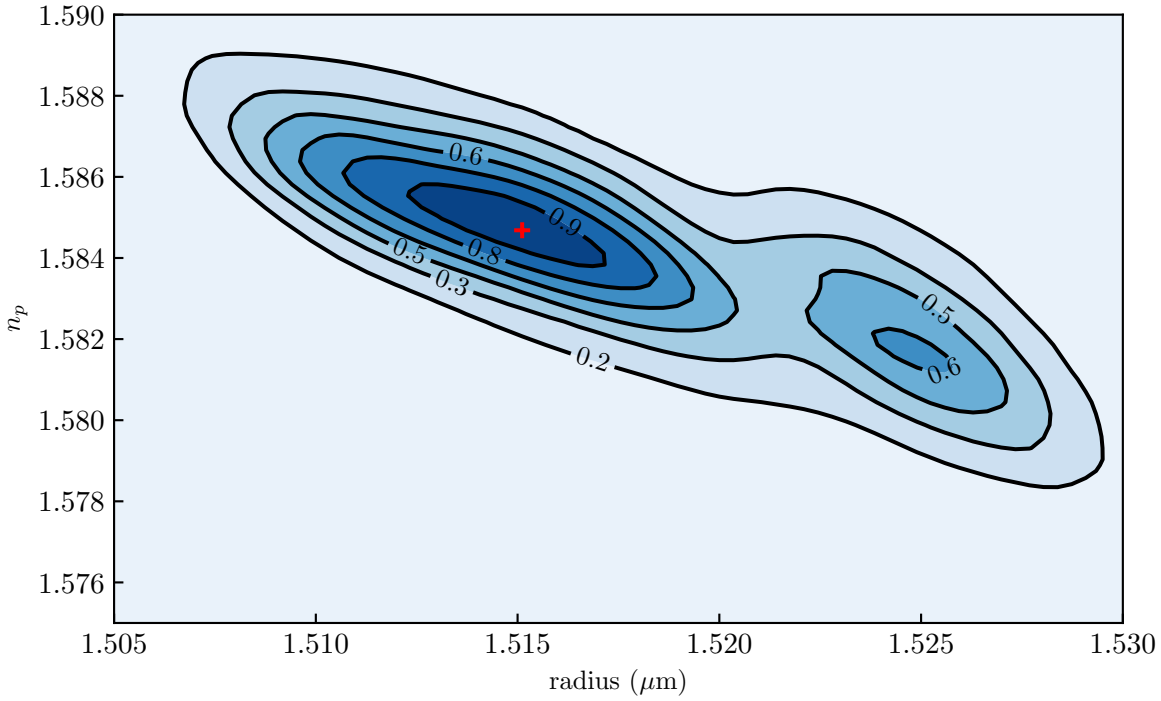


Figure 20: 2D Probability density function of the measurements of the optical index n_p and radius a. Black lines indicate iso-probability. Taking the 10% top probability, we measure $n_\mathrm{p} = 1.585 \pm 0.002$ and $a = 1.514 \pm 0.003$ μm.

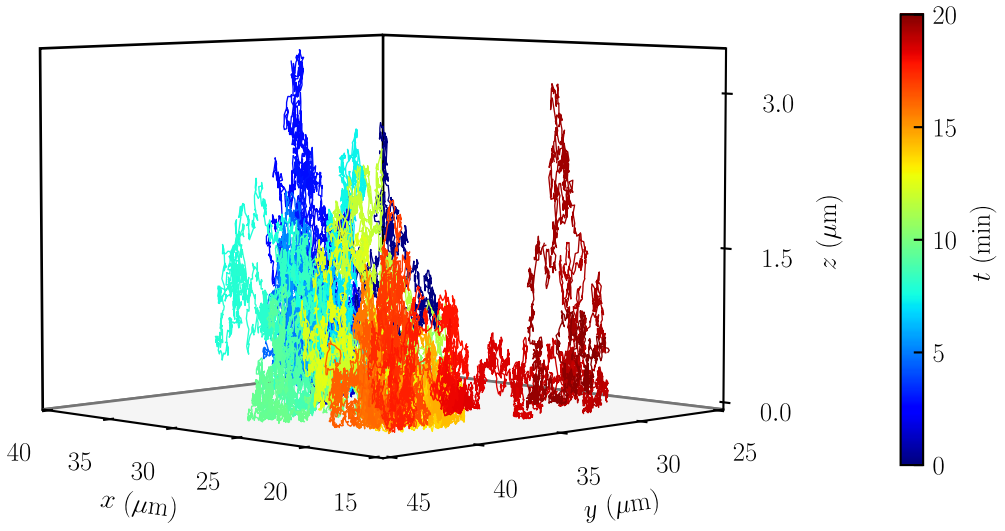


Figure 21: 3D plot of an experimental trajectory measured in water for a particle of optical index $n_\mathrm{p} = 1.585$ and radius $a = 1.514$ μm.

2.7.4 Conclusion

In this chapter, we have covered different techniques that enable the tracking of individual microparticles. Each method has pros and cons. We decided to employ the Lorenz-Mie framework since it requires no calibration. Then, we have shown how we implement it in practice, from the experimental setup to the numerical treatment. An example of the Jupyter notebooks employed for the tracking can be found in appendix A.2. We have discussed how to have fast and accurate fits to retrieve the particle trajectory. To do so, we first characterize the particle fully, namely, its radius and optical index, analyzing solely the trajectory over before tracking a whole video.

Now that we have an understanding on the tracking of single colloids, we can use the measured trajectories in order to understand how the Brownian motion is affected in various configurations.

3 Stochastic Inference of Surface-Induced Effects Using Brownian Motion

3.1 Confined Brownian motion theory

By observing the experimental trajectory along the z-axis of a particle of 1.5 μm as shown in Fig. 22, one can notice that the particle height does not get higher than $\simeq$ 4 μm. Indeed, due to the gravity, a colloid is confined near the surface. Brownian motion in confinement and at interfaces is a canonical situation, encountered from fundamental biophysics to nanoscale engineering. This confinement induces near-wall effects, such as hindered mobility and electrostatic interactions.

In the first part[6] of this chapter, I will detail the theory background of the confined Brownian motion and how to numerically simulate it. In a second part, I will present how to analyze experimental data. In particular, I will detail a multi-fitting procedure that allows a thermal-noise-limited inference of diffusion coefficients spatially resolved at the nanoscale, equilibrium potential, and forces at the femtomewton resolution.

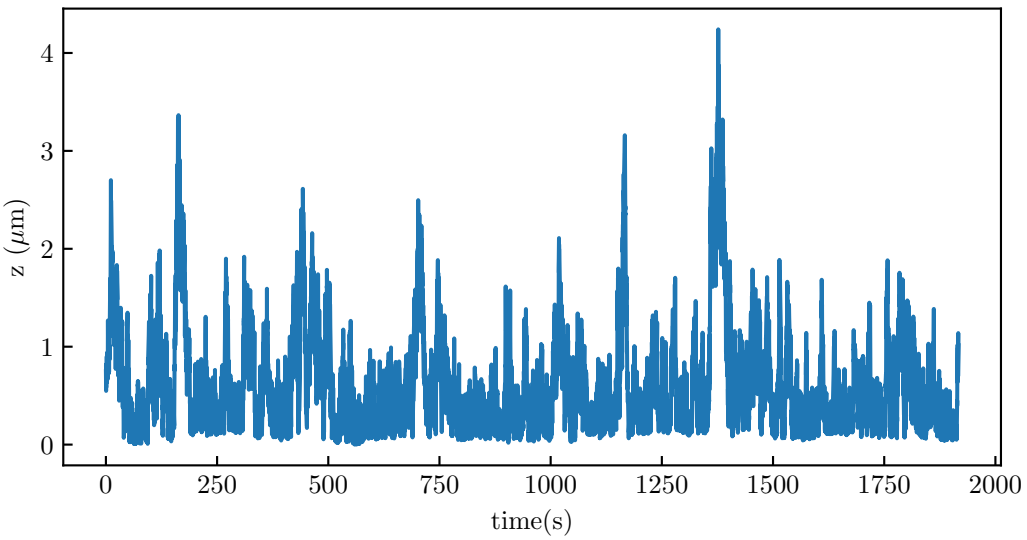


Figure 22: Experimental trajectory of a polystyrene particle of radius $a = 1.5$ μm near a wall ($z = 0$) along the z-axis — perpendicular to the wall.

[6] To write this first part, I took some inspiration from the Matse's master thesis [82].

3.1.1 Gravitational interactions

The density ρ_{p} of an observed colloid is different from the medium density ρ_{m} — in our experiment we used water which density is $\rho_{\mathrm{m}} = 1000\ \mathrm{kg.m^{-3}}$. Thus, particles are subject to gravity, and lies into a gravitational potential given by:

$$U_{\mathrm{g}}(z) = \Delta m g z = \frac{4}{3}\pi a^3 g \Delta\rho z \ , \tag{3.1.1}$$

where Δm is the mass difference of the particle and a fluid sphere of the same size, $\Delta\rho = \rho_{\mathrm{m}} - \rho_{\mathrm{p}}$ the corresponding density difference, and g the gravitational acceleration. By invoking the definition of the Boltzmann length:

$$\ell_{\mathrm{B}} = \frac{k_{\mathrm{B}}T}{4/3\pi a^3 \Delta\rho g} \ , \tag{3.1.2}$$

one can rewrite the gravitational potential Eq. (3.1.1) as:

$$U_{\mathrm{g}}(z) = \frac{k_{\mathrm{B}}T}{\ell_{\mathrm{B}}} z \ . \tag{3.1.3}$$

The Boltzmann length ℓ_{B} is the typical gravitational decay length, and represents the balance between the gravitational potential and thermal energy. This distance was first measured by Perrin [5]. To do so, using a microscope he counted the number of particles as a function of the sample's height. Then, he reconstructed the concentration of the colloidal suspension that exponentially decays as $e^{-z/\ell_{\mathrm{B}}}$. As an example, for a particle of radius $a = 1.5\ \mu$m polystyrene, in water such as $\rho_{\mathrm{p}} = 1050\ \mathrm{kg.m^{-3}}$, one has $\ell_{\mathrm{B}} = 0.58\ \mu$m.

For systems which $\ell_{\mathrm{B}} >> a$, one can consider that the particle does not feel the gravity. This is particularly the case when the density of the colloids and fluid matches, in this particular case $\ell_{\mathrm{B}} = 0$. Thus, density matching can be a way to do gravitation free experiments. In the case of our experiment, we want to measure confinement induced effects. Therefore, we need this gravitational interaction for particles to be near the surface. As particles get larger or denser, ℓ_{B} decreases and particles is, on average, closer to the surface.

3.1.2 Sphere-wall interactions

As we have seen, external forces such as the gravity acts on the particles. However, it is not the only one. As Brownian particles are close to a wall, we can expect some interactions between the surfaces. In our case, we suppose that the Brownian particles do not interact between each other, which is the case for dilute solutions. Indeed, the studied particles are at a minimum 50 μm apart which correspond to 10 times their size for the larger beads.

To describe the interaction between the Brownian particle and the wall, we use the DLVO[7] theory. This theory was first developed to describe the interactions between colloids, and explains the stability of colloidal suspensions. This theory describes the interactions using two forces components; the Van der Waals force which arises from the interactions between surface's molecules and electrostatic interactions due to a double layer formed with the ions present in the solution.

3.1.2.1 Double layer interactions

When a surface is immersed in water, it usually charges [70] due to a high water dielectric constant $\epsilon \simeq 80$ that permits building up charges for low energetic price. Commonly, surface charging is done through the ionization of dissociated surface groups[8], or from the binding of ions from the solution — for example, adsorption of $-OH^-$ onto the water-air interface that charge it negatively. In the bulk, a fluid should be electrically neutral; thus the fluid contains as many ions of opposite charge. However, when a surface is charged negatively, negative ions are repelled from the surface, while positive ions are attracted towards the surface. Therefore, a double-layer charge distribution is formed near the surface, as shown in Fig. 30-a). Experimentally, we use glass slides and polystyrene beads that are both negatively charged in water leading to repulsive double layer. This repulsive force prevents the colloids from sticking together, or to the substrate's surface.

If the solution contains an electrolyte (ions of positive and negative charges), for example a salty solution containing Na^+ and Cl^- ions. The DLVO theory tells that the electrostatic field $\Psi(\vec{r})$ generated by the double layer, satisfies the mean field Poisson's equation [70]:

[7] The DLVO theory is named after Derjaguin, Landau, Verwey, and Overbeek [70].

[8] For example, the dissociation of protons from surface carboxylic groups [70] ($-COOH \rightarrow$ -COO^- + H^+) which charge negatively the surface.

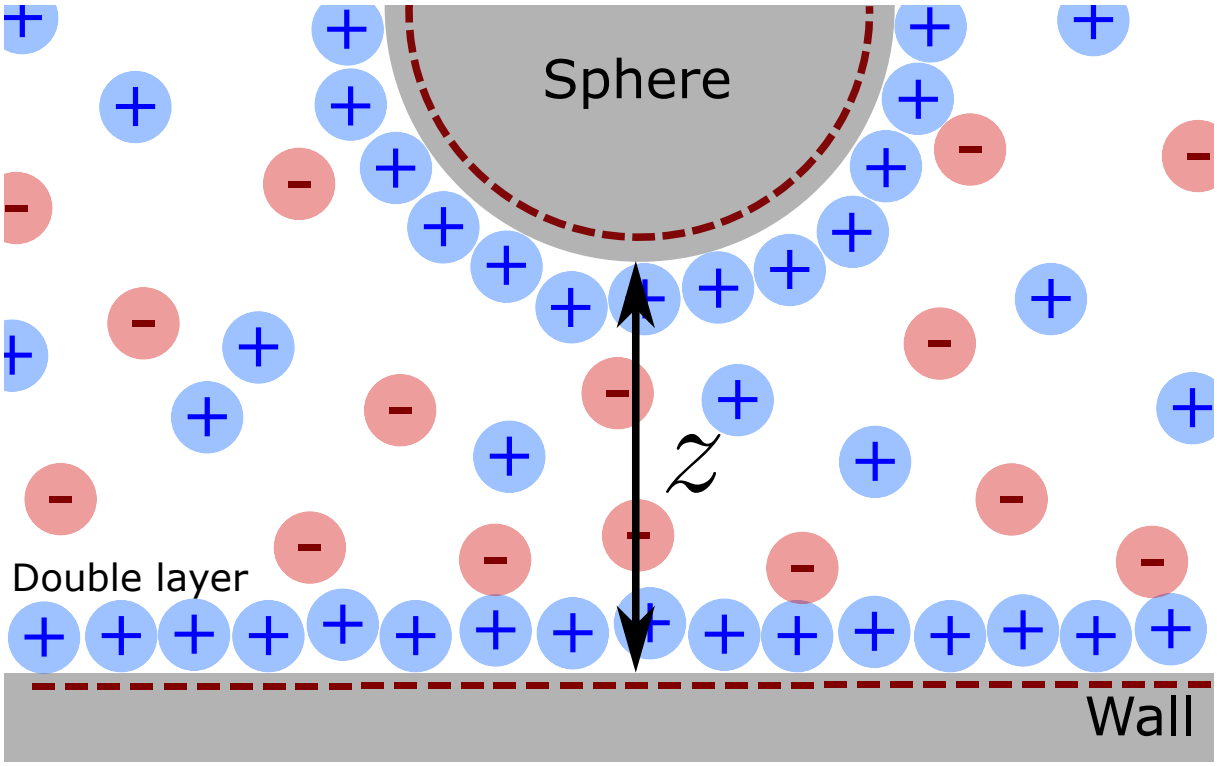


Figure 23: A Brownian colloid diffusing near a wall. Both wall and colloid surface charge negatively. In consequence, a layer of positively charge ions are towards the surfaces, forming a double-layer charge distribution.

$$\nabla^2 \Psi(\vec{r}) = -\frac{1}{\epsilon_r \epsilon_0} \rho_e(\vec{r}) \ , \tag{3.1.4}$$

where ϵ_0 is the vacuum permittivity, ϵ_r the relative permittivity of the fluid, and:

$$\rho_e(\vec{r}) = e \sum_i z_i c_i(\vec{r}) \ , \tag{3.1.5}$$

the local charge density, where e is the elementary charge, i denotes an ionic species of valence z_i and local ionic concentration c_i (number density). If the solution is at the thermodynamic equilibrium, the local ion density is given by a Gibbs-Boltzmann distribution, as:

$$c_i(\vec{r}) = c_i^0 \exp\left(\frac{z_i e \Psi(\vec{r})}{k_\mathrm{B} T}\right) \ , \tag{3.1.6}$$

where c_i^0 is the bulk concentration (number density) of the ionic species i. By combining Eqs. (3.1.4), (3.1.5) and (3.1.6), one can obtain the Poisson-Boltzmann equation:

$$\nabla^2\Psi(\vec{r}) = \sum_i \frac{z_i e c_i^0}{\epsilon_0\epsilon_r} \exp\left(-\frac{z_i e\Psi(\vec{r})}{k_\mathrm{B}T}\right) \ . \tag{3.1.7}$$

Since the Poisson-Boltzmann is nonlinear, it is most likely to be solved numerically. However, for simple geometry such as uniformly charged plane or spheres it can be solved analytically. To simplify, let us consider that we have a monovalent electrolyte, meaning that the electrolyte is composed of two ions of valence equal to 1 — Na^+ Cl^- for example — and c_i^0 is equal to the electrolyte solution concentration c_s^0. In such a case Eq. (3.1.7) simplifies to:

$$\begin{aligned}\nabla^2\Psi(\vec{r}) &= \frac{ec_s^0}{\epsilon_0\epsilon_r}\left[\exp\left(\frac{-e\Psi(\vec{r})}{k_\mathrm{B}T}\right) - \exp\left(\frac{+e\Psi(\vec{r})}{k_\mathrm{B}T}\right)\right] \\ &= 2\frac{ec_s^0}{\epsilon_0\epsilon_r}\sinh\left(\frac{e\Psi(\vec{r})}{k_\mathrm{B}T}\right) \ .\end{aligned} \tag{3.1.8}$$

In the case where Ψ is small enough everywhere such as $e\Psi \ll k_\mathrm{B}T$, which is generally the case when using salty solutions, it is possible, using a Taylor approximation at the second order to write:

$$\exp\left(-\frac{z_i e\Psi(\vec{r})}{k_\mathrm{B}T}\right) \simeq 1 + \frac{z_i e\Psi(\vec{r})}{k_\mathrm{B}T} \ . \tag{3.1.9}$$

Thus, the Poisson-Boltzmann equation Eq. (3.1.7) becomes:

$$\nabla^2\Psi(\vec{r}) = \sum_i \frac{z_i e c_i^0}{\epsilon_0\epsilon_r}\left(1 + \frac{z_i e\Psi(\vec{r})}{k_\mathrm{B}T}\right) \ . \tag{3.1.10}$$

In the bulk, a fluid is electrically neutral. Therefore, the first term vanishes since $\sum_i z_i c_i^0 = 0$. Thus, one can linearize Eq. (3.1.7) to write the Debye-Hünkel equation:

$$\nabla^2\Psi(\vec{r}) = \left[\sum_i \frac{z_i^2 e^2 c_i^0}{\epsilon_0\epsilon_r k_\mathrm{B}T}\right]\Psi(\vec{r}) \ . \tag{3.1.11}$$

From this approximation, one can identify the term between brackets as the inverse of a distance squared. We thus define:

$$\ell_{\mathrm{D}} = \sqrt{\sum_i \frac{\epsilon_0 \epsilon_r k_{\mathrm{B}} T}{z_i^2 e^2 c_i^0}} \ , \tag{3.1.12}$$

the Debye length which is the characteristic length of the double layer, and hence, the electrostatic interactions. For a monovalent electrolyte, at 25 °C (298 K), the Debye length of an aqueous solution is:

$$\begin{aligned} \ell_{\mathrm{D}} &= \sqrt{\frac{\epsilon_0 \epsilon_r k_{\mathrm{B}} T}{2 c_s^0 e^2}} = \sqrt{\frac{8.854 \times 10^{-12} \times 78.4 \times 1.381 \times 10^{-23} \times 298}{2 \times (1.602 \times 10^{-19})^2 \times 6.022 \times 10^{26} M}} \\ &= 0.304 \times 10^{-9} / \sqrt{M} \text{ m} \ , \end{aligned} \tag{3.1.13}$$

with M the molar concentration (1 M = 1 mol.L^{-1} corresponding to a number density of $c_s^0 = 6.022 \times 10^{26}$ m^{-3}). For a salty concentration we have:

$$\ell_{\mathrm{D}} = 0.304 / \sqrt{[\mathrm{NaCl}]} \text{ nm} \tag{3.1.14}$$

For example, $\ell_{\mathrm{D}} = 100$ nm for a concentration [NaCl] = 9.2 μM and $\ell_{\mathrm{D}} = 10$ nm for a concentration [NaCl] = 9.2 mM.

Finally, combining Eqs. (3.1.11) and (3.1.12), the Debye-Hünkel approximation writes:

$$\nabla^2 \Psi(\vec{r}) = \kappa^2 \Psi(\vec{r}) \ , \tag{3.1.15}$$

with $\kappa = 1/\ell_{\mathrm{D}}$. Using the latter approximation one can compute the electrostatic potential around a sphere. Let us consider a perfect sphere of radius a and charge Qe of charge density $\sigma = Qe/4\pi a^2$,with Q being the number of charge on the surface. Since the system has a spherical symmetry, one has $\Psi(\vec{r}) = \Psi(r)$ with $r = |\vec{r}|$. Using the Laplacian operator ∇^2 in the spherical coordinates, Eq. (3.1.15) writes:

$$\frac{1}{r^2} \left[\frac{\partial}{\partial r} \left(r^2 \frac{\partial \Psi(r)}{\partial r} \right) \right] = \kappa^2 \Psi(r) \ , \tag{3.1.16}$$

which has a general solution:

$$\Psi(r) = C_1 \frac{\exp(\kappa r)}{r} + C_2 \frac{-\exp(\kappa r)}{r} \tag{3.1.17}$$

For one sphere, the electrostatic field vanishes at infinity such as $C_1 = 0$, such it has the form of a Yukawa potential:

$$\Psi(r) = C_2 \frac{-\exp(\kappa r)}{r} \ . \tag{3.1.18}$$

Additionally, at the surface of a charged sphere, the electrostatic potential satisfies:

$$\left.\frac{\partial \Psi(r)}{\partial r}\right|_{r=a} = \frac{Qe}{4\pi\epsilon_0\epsilon_r a^2} = \frac{\sigma}{\epsilon_0\epsilon_r} \ . \tag{3.1.19}$$

By applying the latter boundary condition to Eq. (3.1.18) we find:

$$\Psi(r) = \frac{\sigma a^2}{\epsilon_0\epsilon_r} \frac{\exp(\kappa a)}{1+\kappa a} \frac{\exp(-\kappa r)}{r} \ . \tag{3.1.20}$$

This solution can be used to determine the electrostatic field between two spheres. Supposing that the presence of a second sphere does not interfere with the distribution of ions in the double layer of the other, one can use the superposition approximation to write the potential $\Psi_2(z)$ between two spheres. For two spheres of charge σ_1 and σ_2, and radius a_1 and a_2, the potential writes [71]:

$$\Psi_2(z) = \frac{4\pi}{\epsilon_0\epsilon_r} \left(\frac{\sigma_1 a_1^2}{1+\kappa a_1}\right) \left(\frac{\sigma_2 a_2^2}{1+\kappa a_2}\right) \frac{\exp(-\kappa z)}{a_1 + a_2 + z} \ , \tag{3.1.21}$$

with z the distance between the 2 colloids. From the latter equation, it is possible to write the electrostatic field between a wall and a spherical colloid, by setting one of the radii to infinity. Doing so and multiplying by e, one can have the electrostatic potential $U_{\text{elec}}(z)$ between a Brownian particle and the wall:

$$\frac{U_{elec}(z)}{k_{\text{B}}T} = B\mathrm{e}^{-\frac{z}{\ell_{\text{D}}}} \ , \tag{3.1.22}$$

where B is the constant that represents the surface charges, for a sphere of radius a and charge σ_1 and a wall of charge σ_2, one has:

$$B = \frac{4\pi}{k_{\mathrm{B}}T\epsilon_0\epsilon_r}\left(\frac{\sigma_1 a^2}{1+\kappa a}\right)\frac{\sigma_2}{\kappa} \ . \tag{3.1.23}$$

Additionally, B is often written as a function of the surfaces potential as [72]:

$$B = 16\frac{\epsilon_r\epsilon_0}{k_{\mathrm{B}}T}\left(\frac{k_{\mathrm{B}}T}{e}\right)\tanh\left(\frac{e\phi_1}{4k_{\mathrm{B}}T}\right)\tanh\left(\frac{e\phi_2}{4k_{\mathrm{B}}T}\right) \ , \tag{3.1.24}$$

where ϕ_1 and ϕ_2 are the Stern potential of the sphere and the wall surfaces. Typical values for B ranges from 1 to 50. In our study we will keep B to describe the electrostatic energy potential as it complicated to decouple σ_1 and σ_2 when the colloid and wall's surface materials are different [72]. However, this is not impossible and is the idea of future work.

3.1.2.2 Van der Waals interactions

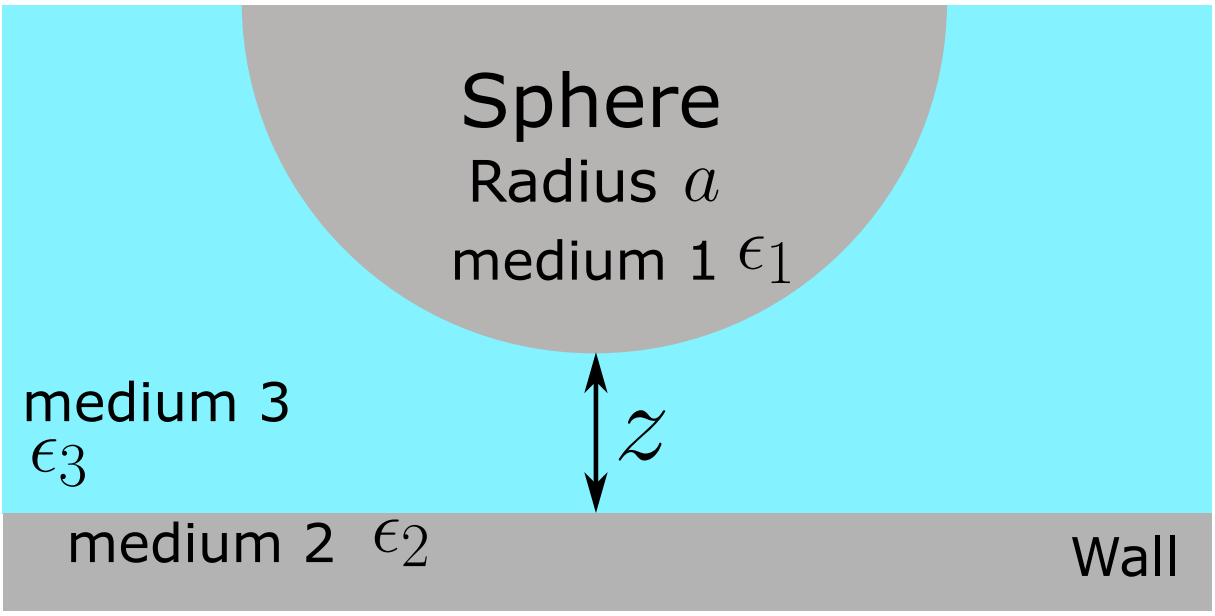


Figure 24: A colloid of radius a at a distance z from the wall. The colloid material has a static dielectric constant ϵ_1, the wall ϵ_2 and the solution ϵ_3.

In the DLVO theory, the Van der Waals forces describes the interaction between colloids at very short range. This force is generally attractive, and in our case having Van der Waals interactions would lead to the particles stick to the wall. The Van der Waals potential energy for a spherical colloid of radius a, at a height z, is given as [70]:

$$U_{VdW} = -\frac{Aa}{6z} \tag{3.1.25}$$

where A is the retarded Hamaker constant. For our system, where the particle, medium and wall are different media as schematize in Fig. 24, the Hamaker constant is given by [70]:

$$A = \frac{3}{4}k_{\mathrm{B}}T\left(\frac{\epsilon_1-\epsilon_3}{\epsilon_1+\epsilon_3}\right)\left(\frac{\epsilon_2-\epsilon_3}{\epsilon_2+\epsilon_3}\right) + \frac{3h}{4\pi}\int_{\nu_1}^{\infty}\left(\frac{\epsilon_1(i\nu)-\epsilon_3(i\nu)}{\epsilon_1(i\nu)+\epsilon_3(i\nu)}\right)\left(\frac{\epsilon_2(i\nu)-\epsilon_3(i\nu)}{\epsilon_2(i\nu)+\epsilon_3(i\nu)}\right)\mathrm{d}\nu \ , \tag{3.1.26}$$

where ϵ_1, ϵ_2 and ϵ_3 are the static dielectric constants of the three media, $\epsilon(i\nu)$ are the imaginary dielectric constant at a frequency ν and $\nu_n = (2\pi k_{\mathrm{B}}Tn/h = 4\times10^{13}ns^{-1}$ at 300 K, and h the Plank's constant. The first term gives the zero-frequency energy of the Van der Waals interaction and the second term the dispersion energy. However, since the static dielectric of water is large compared to the other terms [70], one can approximate the Hamaker constant of our system to only the zero-frequency term:

$$A = \frac{3}{4}1.381\times10^{-23}\times298\left(\frac{2.5-80}{2.5+80}\right)\left(\frac{5-80}{5+80}\right) \simeq 2.48\times10^{-21}\mathrm{J} = 0.62\ k_{\mathrm{B}}T\ . \tag{3.1.27}$$

Since A is positive, we correctly have an attractive Van der Waals force. More over, the zero-frequency Van der Waals force is mainly due to electrostatic interaction and is screened by electrolytes. Thus one needs to add a factor $\approx e^{-z/\ell_D}$ to the Van der Waals forces. Also, it has been proposed [73] that to compute the force at higher distances, one should add a second factor, to account for the retarded Van der Waals force of $\simeq (1+11z/100\mathrm{nm})^{-1}$.

All the effects added together we can calculate that the Van der Waals forces will play a role only a few nanometers from the surface ($z < 10$ nm) as it is commonly observed [19]. In our experiments, the Debye length ℓ_{D} (> 20 nm) is large enough for the particle to avoid this region, in the following of this work the Van der Waals interactions are neglected. To study the Van der Waals interactions with Brownian motion, it is possible, if one adds enough salt to have $\ell_{\mathrm{D}} \simeq 1$ nm. However, with such a short Debye length, all the colloids would stick to the surface and between each other. Interestingly, we have experimentally observed some dynamics on stuck particles, further work on this movement could lead to

interesting determination of the near-wall potential.

3.1.2.3 Total potential and equilibrium distribution

If we combine the gravitational and electrostatic interactions the particles lies into a total energy potential $U(z)$:

$$U(z) = U_{\mathrm{g}} + U_{\mathrm{elec}} \tag{3.1.28}$$

By combining Eqs. (3.1.3), (3.1.22) and (3.1.28), and adding the condition that the particle cannot go inside the wall $U(z)$ finally writes:

$$\frac{U(z)}{k_{\mathrm{B}}T} = \begin{cases} B\,\mathrm{e}^{-\frac{z}{\ell_{\mathrm{D}}}} + \dfrac{z}{\ell_{\mathrm{B}}}\,, & \text{for } z > 0 \\ +\infty\,, & \text{for } z \leq 0 \end{cases}\,, \tag{3.1.29}$$

From this total potential energy, one can then construct the Gibbs-Boltzmann distribution to write the equilibrium PDF of position $P_{\mathrm{eq}}(z)$:

$$P_{\mathrm{eq}}(z) = A \exp\left(-\frac{U(z)}{k_{\mathrm{B}}T}\right)\,, \tag{3.1.30}$$

where A is a normalization constant such that $\int_0^\infty P_{\mathrm{eq}}(z)\mathrm{d}z = 1$. Given an array of heights z_i, one can compute P_{eq} using the following Python snippet, where the A is computed using the `np.trapz` function. An example of a theoretical energy potential and PDF of position can be seen in Fig. 25 for $\ell_{\mathrm{B}} = 500$ nm, $B = 4$ and $\ell_{\mathrm{D}} = 50$ nm.

```
1         import numpy as np
2
3         def _Peq(z):
4         if z <= 0:
5         return 0
6         else:
7         return np.exp(-(B * np.exp(-z / ld) + z / lb))
8
9
10        def Peq(z):
```

```
        P = np.array([_Peq(zi) for zi in z])
        return P / np.trapz(P,z)
```

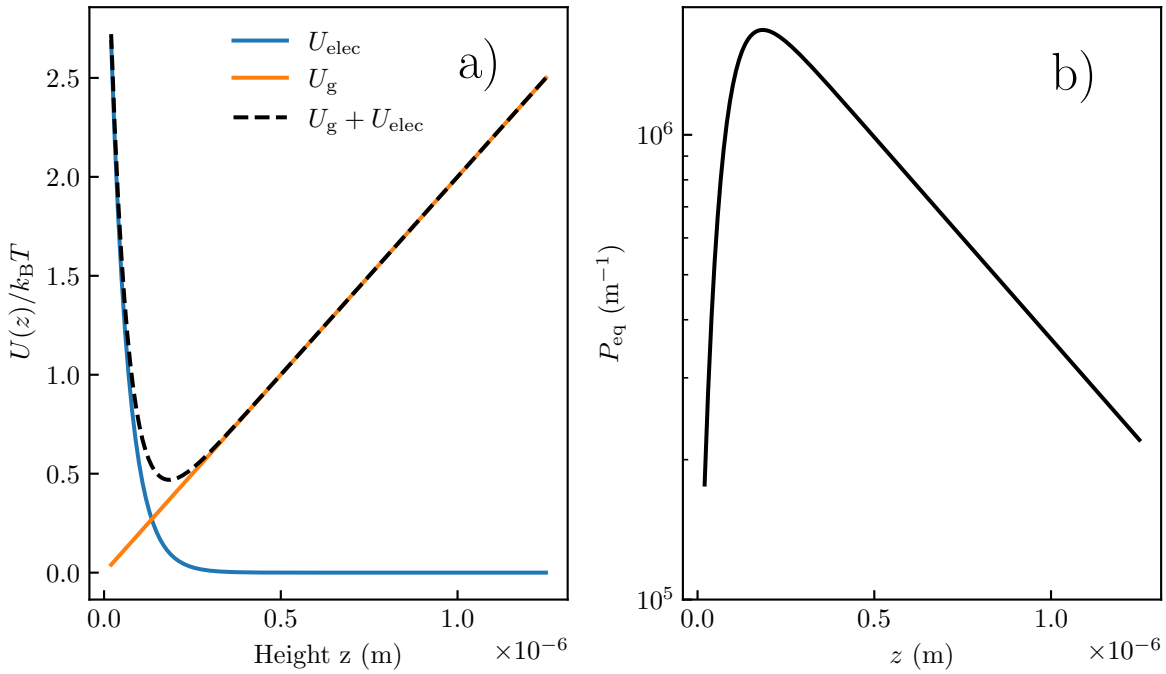


Figure 25: a) In orange gravity potential energy E_{g} (see Eq. (3.1.3)) of a colloid with a Boltzmann length $\ell_{\text{B}} = 500$ nm. In blue, the electrostatic potential U_{elec} (see Eq. (3.1.22)) is characterized by a surface charge constant $B = 4$ and a Debye length $\ell_{\text{D}} = 50$ nm. The dashed line is the total potential, see Eq. (3.1.28) b) Corresponding Gibbs-Boltzmann equilibrium distribution of position calculated using Eq. (3.1.29).

3.1.3 Local diffusion coefficient

As we have seen in Chapter 1 , for a freely diffusing colloid in bulk, the diffusion coefficient is given by Eq. (1.2.12) and is a constant. However, when a particle is confined, the diffusion is hindered, this means that the diffusion coefficient varies with the height of the particle and becomes anisotropic. One of the first measurements that has been done by Faucheux and Libchaber [13]. As we can see in Fig. 26, using a microscope, they tracked Brownian colloids in two dimensions, and measured the average coefficient diffusion for different confinement constant $\gamma = (\langle z \rangle_t - a)/a$. Finally, in Fig. 26, one can observe that the diffusion coefficient parallel to the surface decreases as the particle get closer to the wall, and seems to saturate around $0.3D_0$ for high confinement.

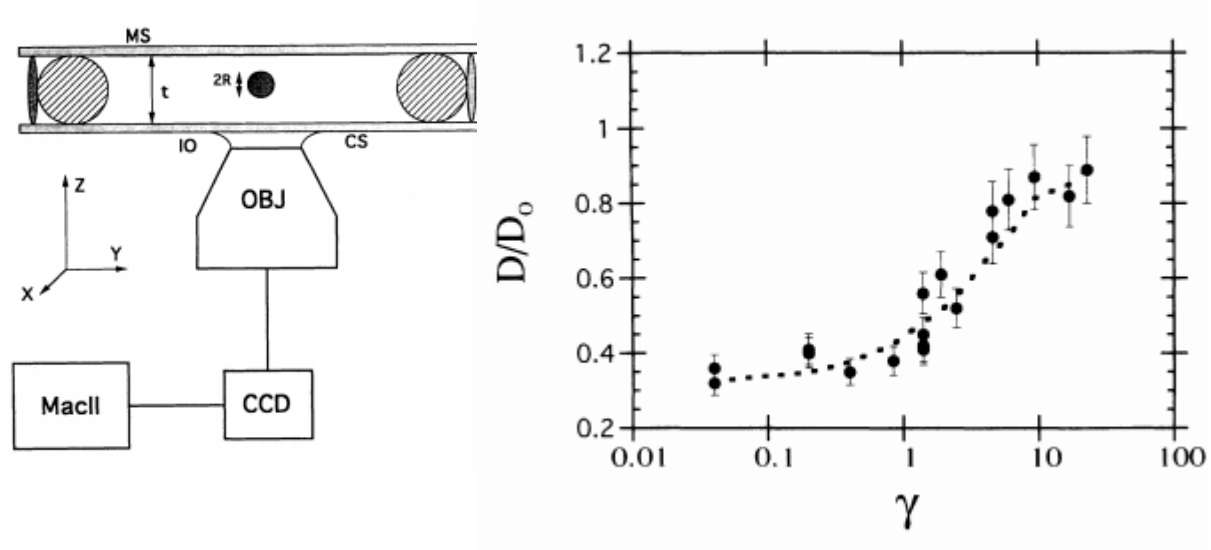


Figure 26: Figure extracted from [13], on the left is the experimental setup used. It is an inverted microscope used in order to track particles of size $2R$ inside a cell of thickness t. On the right is their final result, where they measure the diffusion parallel coefficient $D_\perp$ given by Eq. (3.1.42), normalized by D_0 the bulk diffusion coefficient as a function of a confinement constant $\gamma = (\langle z \rangle_t - a)/a$.

To understand the reason for this hindered diffusion coefficient, let us start by writing the diffusion coefficient D using the fluctuation dissipation theorem:

$$D = \mu k_{\mathrm{B}} T \ , \tag{3.1.31}$$

with:

$$\mu = \frac{v_{\mathrm{sphere}}}{F_{\mathrm{drag}}} \ , \tag{3.1.32}$$

where v_{sphere} is the terminal velocity to an applied force F_{drag}. For a spherical colloid of radius R moving at a velocity v_{sphere}, the drag force in bulk $F^{\mathrm{B}}_{\mathrm{drag}}$ is given by the Stokes' law:

$$F^{\mathrm{B}}_{\mathrm{drag}} = c\pi\eta a v_{\mathrm{sphere}} \ , \tag{3.1.33}$$

where c is a constant that depends on the boundary conditions imposed at the surface of

the colloid, typically $c = 6$ for no-slip boundary conditions and $c = 4$ (such as air bubbles for example) for slip boundary conditions. Combining Eqs. (3.1.31), (3.1.32) and (3.1.33) for a freely diffusing hard sphere in bulk we retrieve Eq. (1.2.12):

$$D_0 = \frac{k_\mathrm{B}T}{6\pi\eta a} \ . \tag{3.1.34}$$

The Stocke's drag force can be computed by solving the Navier-Stokes equation:

$$\rho\left[\frac{\partial\vec{v}}{\partial t} + (\vec{v}\cdot\nabla\vec{v})\right] + \nabla p = \eta\nabla^2\vec{v} \ , \tag{3.1.35}$$

and the continuity equation for incompressible fluids:

$$\nabla\cdot\vec{v} = 0 \tag{3.1.36}$$

where $\vec{v}$ and p are the velocity, and pressure fields, and ρ is the fluid's density. When the Reynolds number $Re = \rho a v_\mathrm{sphere}/\eta \ll 1$, the first two terms are inertial and are negligibly small compared to the viscous term $\eta\nabla^2\vec{v}$. In that case, the Navier-Stokes Eq. (3.1.35), is simplified to the Stockes flow formula:

$$\nabla p = \eta\nabla^2\vec{v} \tag{3.1.37}$$

The diffusion coefficient for a freely diffusing spherical colloid in bulk can thus be found by solving Eqs. (3.1.37) and (3.1.36) by having a no-slip boundary condition on the particle surface and the velocity vanishing at infinity. However, in the case of a confined particle, there is an additional no-slip condition at the wall surface. Indeed, as shown in Fig. 27, the fluid velocity drops to zero at the wall surface, with a shear rate:

$$\dot{\gamma} = \frac{\partial v}{\partial z} \tag{3.1.38}$$

This shear rate, due to the presence of the wall introduce a shear force $F(z) =\simeq \eta\dot{\gamma}(z)$ acting on the particle. Hence, the drag force acting on the particle will increase inversely with the distance to the wall, due to additional hydrodynamic pressure arising from velocity gradients. At the macro scale, this effect can be seen with a frisbee, indeed, as it

gets closer to the grounds, hydrodynamic pressure increases due to air velocity gradient and one can observe that the frisbee seems to stop falling and continue to glide really near the ground's surface.

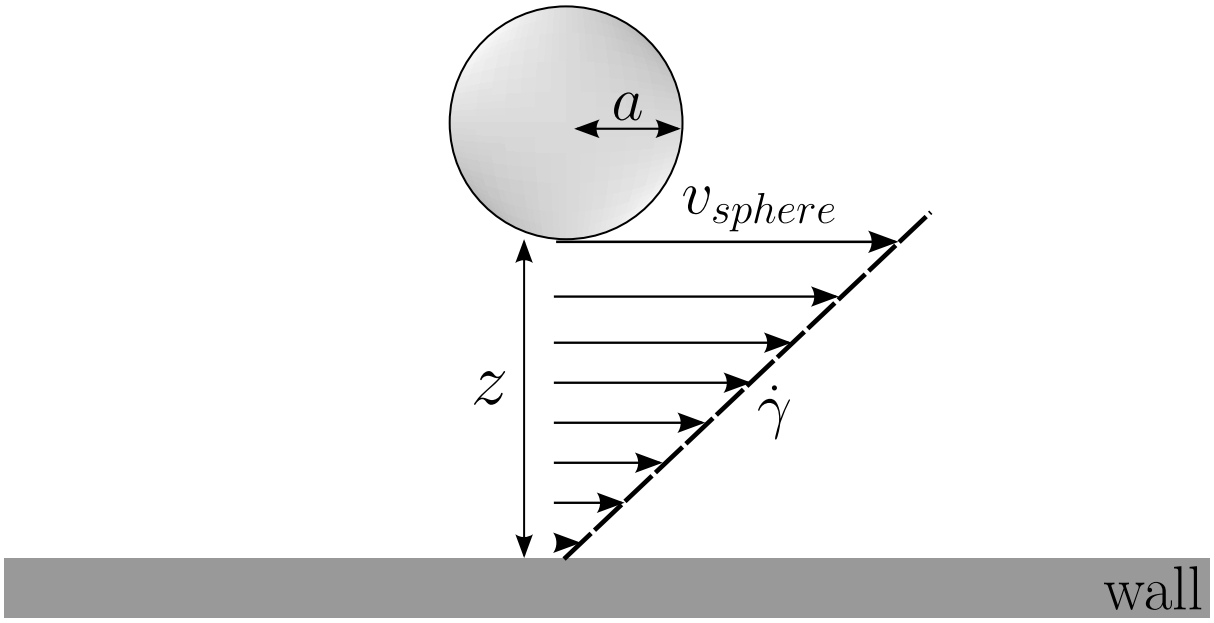


Figure 27: Schematic representation of a spherical colloid moving near a wall and the induced shear rate $\dot{\gamma}$.

A colloid diffusing near a wall thus experience a local drag force that depends on both its distance to the wall z and the movement direction. Thanks to the linearity of the Stokes equation, one can separate this local drag force for motion parallel and perpendicular to the wall. As the presence of the wall correct the drag force from a multiplicative coefficient, the confinement effect often expressed as an effective viscosity:

$$\eta_{\perp}(z) = \eta\lambda_{\perp}(z) \text{ , and } \eta_{\parallel}(z) = \eta\lambda_{\parallel}(z) \text{ ,} \tag{3.1.39}$$

where $\lambda_{\perp}$ and $\lambda_{\parallel}$ are respectively perpendicular and parallel correction factor to the drag force due to the presence of the wall. Taking into account this correction, the diffusion coefficient for parallel and perpendicular motion relative to the wall writes:

$$D_{\perp}(z) = \frac{D_0}{\lambda_{\perp}(z)} \text{ , and, } D_{\parallel}(z) = \frac{D_0}{\lambda_{\parallel}(z)} \text{ .} \tag{3.1.40}$$

For a no-slip boundary conditions imposed at the wall and the surface of the colloid, Brenner [74] obtained for perpendicular motion:

$$\lambda_{\perp}(z) = \frac{4}{3}\sinh\beta \sum_{n=1}^{\infty} \frac{n(n+1)}{(2n-1)(2n+3)} \left[\frac{2\sinh(2n+1)\beta + (2n+1)\sinh 2\beta}{4\sinh^2(n+1/2)\beta - (2n+1)^2\sinh^2\beta} - 1 \right] , \tag{3.1.41}$$

where $\beta = \cosh^{-1}((z+a)/a)$. The solution for the diffusion parallel to the wall, Faxén found [75]:

$$\lambda_{\parallel}(z) = \left[1 - \frac{9}{16}\xi + \frac{1}{8}\xi^3 - \frac{45}{256}\xi^4 - \frac{1}{16}\xi^5 \right]^{-1} , \tag{3.1.42}$$

where $\xi = a/(z+a)$. Eqs. (3.1.41) and (3.1.42) are precise for all z. However, the solution for the perpendicular motion can be quite complex to compute as it is an infinite series, to be computed numerically it requires a software that enables arbitrary-precision floating-point arithmetic[9] — such as Mathematica or the `mpmath` Python's module, for example. $D_{\perp}$ can be evaluated using the following Python snippet, where `nsum` function is used to compute the infinite sum:

```
from mpmath import nsum

def Dz(eta, z, a):
a = (z + a) / a
beta = float(acosh(a))
summ = nsum(
lambda n: (n * (n + 1) / ((2 * n - 1) * (2 * n + 3)))
* (
(
(2 * sinh((2 * n + 1) * xi) + (2 * n + 1) * sinh(2 * beta))
/ (
4 * (sinh((n + 1 / 2) * beta) ** 2)
- ((2 * n + 1) ** 2) * (sinh(beta) ** 2)
)
)
- 1
),
[0, inf],
)
summ = float(summ)
return kT / (6 * pi * eta * 4 / 3 * float(sinh(beta)) * summ * a)
```

[9] Arbitrary precision floating-point arithmetic enables to evaluate mathematical expressions with any precision, in other words, any number of digits.

To simplify the computation of $\lambda_\perp$, Goldman *et al.* [76] showed that Eq. (3.1.41) can be Padé approximated, giving:

$$\lambda_\perp = \frac{6z^2 + 9az + 2a^2}{6z^2 + 2az} \,. \tag{3.1.43}$$

In the near-wall regime, such that $z \ll a$, it is also possible to further approximate $\lambda_\perp$ to:

$$\lambda_\perp = \frac{a}{z} \,. \tag{3.1.44}$$

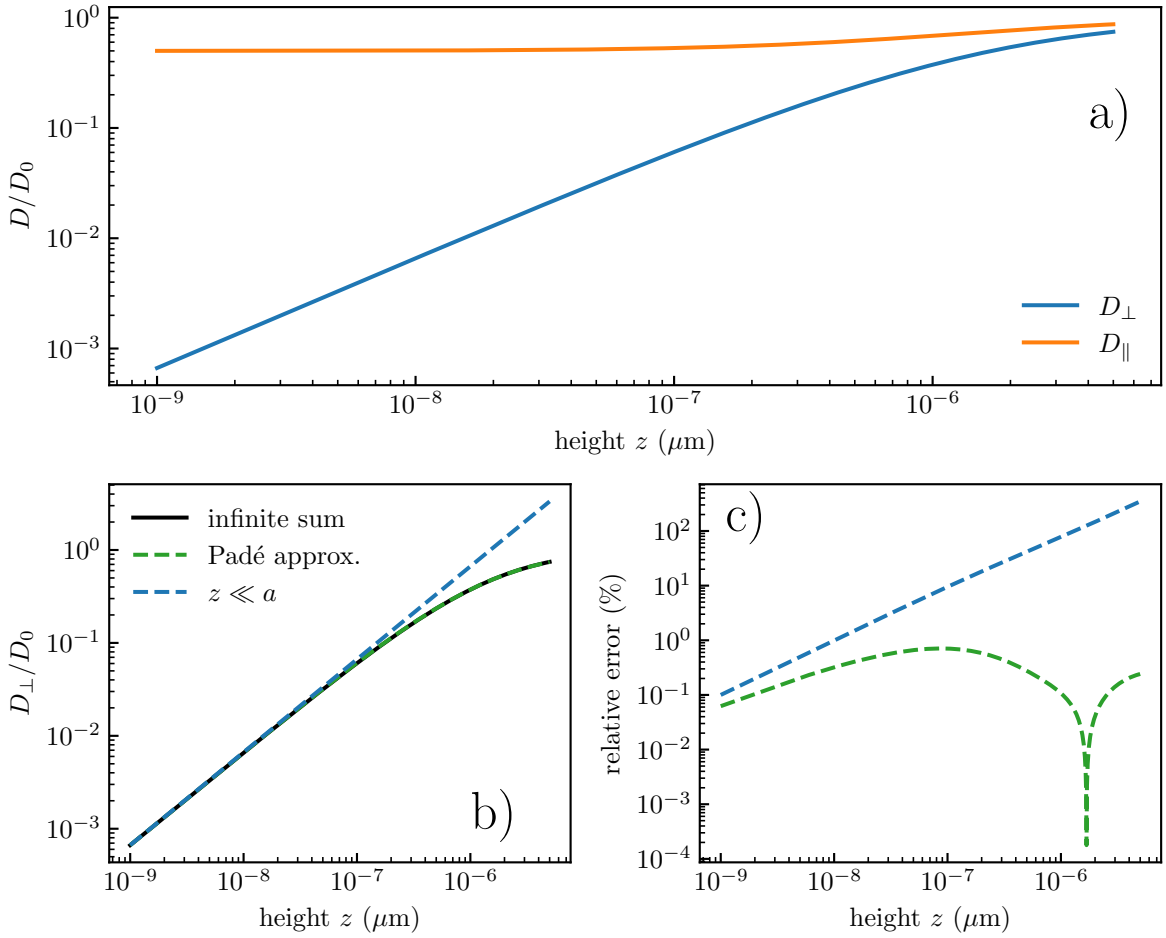


Figure 28: a) Parallel and perpendicular hindered relative diffusion coefficient for a colloidal particle of radius $a = 1.5\ \mu$m. b) Perpendicular hindered relative diffusion coefficient for a colloid particle of radius $a = 1.5$mum. In black the exact solution given by the infinite sum Eq. (3.1.41). In green the Padé approximation, Eq. (3.1.43) and in blue the near-wall regime Eq. (3.1.44). c) Relative error of the perpendicular hindered coefficient.

The Padé approximation and the near-wall approximation for the hindered diffusion are

plotted on Fig. 28-b). The Padé approximation fits really well with the exact solution given by the infinite sum Eq. (3.1.41), the near-wall approximation fits well when $z < a/10$. To check how precise the approximation are, it is possible to plot the relative error as in the Fig. 28-c). The Padé approximation shows precision up to 1%, thus, in the following of this work, when we refer to any evaluation of hindered perpendicular diffusion, the Padé approximation will be used.

3.1.4 Langevin equation for the confined Brownian motion

Now that the external forces acting on the particle and hindered diffusion coefficient is known, we rewrite the overdamped Langevin Eq. (1.3.10) as:

$$V_t^i \mathrm{d}t = -\frac{1}{\gamma(z)}\frac{\partial U(z)}{\partial x_i}\mathrm{d}t + \sqrt{2D_i(z)}\mathrm{d}B_t \ . \tag{3.1.45}$$

where $\gamma(z) = 6\pi\eta_i(z)a$ and i denotes three spatial directions, x, y and z[10], and $\mathrm{d}B_t$ is a Gaussian distribution satisfying $\langle \mathrm{d}B_t \rangle = 0$ and $\langle \mathrm{d}B_t^2 \rangle = \mathrm{d}t$. As we have discussed previously, the potential U only varies along the z-axis, thus, the external forces only acts on the particle on the z-axis while the particle diffuses freely along the x- and y-axis.

3.1.5 Spurious drift

It is interesting to observe that due to the hindered diffusion the magnitude of the Langevin force, $\sqrt{2D_i(z)}$, is not anymore constant, but varies with the height of the particle. This effect is called multiplicative noise and will have some interesting effects on the dynamic properties of the Brownian motion. To show the effects of the multiplicative noise, one can integrate over a time τ the Eq. (3.1.45), in the absence of the external force one as:

$$\Delta x_i = \int_{t_0}^{t_0+\tau} \sqrt{2D_i(z)}\mathrm{d}B_t \tag{3.1.46}$$

where Δx_i is a space increment. However, the noise term is not well defined and the time at which the magnitude of the force $\sqrt{2D_i(z)}$ in the integration Eq. (3.1.46) needs

[10] Where the previously determined $\eta_\parallel$ and $D_\parallel$ correspond to the x- and y-axis and $\eta_\perp$ and $D_\perp$ correspond to the z-axis.

to be specified. It is thus necessary to determine where the diffusion coefficient $D(z)$ in Eq. (3.1.46) is evaluated in the time interval $[t_0, t_0 + \tau]$. In general, $D(z)$ is represented by $D(z + \alpha\Delta z)$, or in the same way $D(z(t_0 + \alpha\tau))$, with $\alpha \in [0, 1]$. The value of α determines at which time in the interval $[t_0, t_0 + \tau]$ the local diffusion is evaluated, hence, the magnitude of the random force. The physics will thus change on how the noise is calculated; however, the requirement is that the long-time thermal equilibrium must be consistent with the Boltzmann distribution and, hence, constrain the choice of α. Taking into account α, removing the external forces, the Langevin equation along the z-axis becomes:

$$\Delta z = \sqrt{2D_{\parallel}(z + \alpha\Delta z)}dB_t \ . \tag{3.1.47}$$

By Taylor expanding $D_{\parallel}$ to the first order, one has:

$$D_{\parallel}(z + \alpha\Delta z) \simeq D_{\parallel}(z) + \alpha\frac{\mathrm{d}D_{\parallel}(z)}{\mathrm{d}z}\Delta z \ . \tag{3.1.48}$$

Substituting the later in Eq. (3.1.47) leads to:

$$\begin{aligned} \Delta z &\simeq \sqrt{2D_{\parallel}(z) + \alpha\frac{\mathrm{d}D_{\parallel}(z)}{\mathrm{d}z}\Delta z}dB_t \\ &= \sqrt{2D_{\parallel}(z)}\left[1 + \alpha\frac{\mathrm{d}D_{\parallel}(z)}{\mathrm{d}z}\frac{\Delta z}{D_{\parallel}(z)}\right]^{-1/2}\mathrm{d}B_t \ . \end{aligned} \tag{3.1.49}$$

Additionally, since the last term satisfies:

$$\alpha\frac{\mathrm{d}D_{\parallel}(z)}{\mathrm{d}z}\frac{\Delta z}{D_{\parallel}(z)} \ll 1 \ , \tag{3.1.50}$$

one can thus Taylor expend at the first order the last term of Eq. (3.1.49) as:

$$\left[1 + \alpha\frac{\mathrm{d}D_{\parallel}(z)}{\mathrm{d}z}\frac{\Delta z}{D_{\parallel}(z)}\right]^{-1/2} \simeq 1 + \frac{\alpha}{2}\frac{\mathrm{d}D_{\parallel}(z)}{\mathrm{d}z}\frac{\Delta z}{D_{\parallel}(z)} \ . \tag{3.1.51}$$

By finally substituting the first-order expansion of $\Delta z \simeq \sqrt{2D(z)}\mathrm{d}B_t$, we obtain:

$$\begin{aligned}\Delta z &\simeq \sqrt{2D_{\parallel}(z)}\left[1+\frac{\alpha}{2}\frac{\mathrm{d}D_{\parallel}(z)}{\mathrm{d}z}\frac{\Delta z}{D_{\parallel}(z)}\right]\mathrm{d}B_t \\ &\simeq \sqrt{2D_{\parallel}(z)}\left[1+\frac{\alpha}{2}\frac{\mathrm{d}D_{\parallel}(z)}{\mathrm{d}z}\frac{\sqrt{2D(z)}\mathrm{d}B_t}{D_{\parallel}(z)}\right]\mathrm{d}B_t \\ &= \sqrt{2D_{\parallel}(z)}\mathrm{d}B_t+\alpha\frac{\mathrm{d}D_{\parallel}(z)}{\mathrm{d}z}\mathrm{d}B_t^2\end{aligned} \tag{3.1.52}$$

Since for a short time $\mathrm{d}B_t \to 0$ it is possible to replace $\mathrm{d}B_t^2$ by its average value $\langle \mathrm{d}B_t^2\rangle_t = \tau$, the latter thus become [77]:

$$\begin{aligned}\Delta z &= \alpha\frac{\mathrm{d}D_{\parallel}(z)}{\mathrm{d}z}\tau-\frac{1}{\gamma(z)}\frac{\partial U(z)}{\partial z}\tau+\sqrt{2D_{\parallel}(z)}\mathrm{d}B_t \\ &= \left(-\frac{1}{\gamma(z)}\frac{\partial U(z)}{\partial z}+\alpha\frac{\mathrm{d}D_{\parallel}(z)}{\mathrm{d}z}\right)\tau+\sqrt{2D_{\parallel}(z)}\mathrm{d}B_t \\ &= \bar{v}_{\mathrm{d}}\tau+\sqrt{2D_{\parallel}(z)}\mathrm{d}B_t\ ,\end{aligned} \tag{3.1.53}$$

where:

$$\bar{v}_{\mathrm{d}} = -\frac{1}{\gamma(z)}\frac{\partial U(z)}{\partial z}+\alpha\frac{\mathrm{d}D_{\parallel}(z)}{\mathrm{d}z} = v_{\mathrm{d}}+v_{\mathrm{noise}}\ , \tag{3.1.54}$$

are the total drifts acting on the particle. The first term v_{d} is the drift due to "real" deterministic forces due to the electrostatic double-layer and gravity interactions, while the second term v_{noise} represents a noise-induced drift. This spurious drift does disappear when the diffusion coefficient is homogeneous, as for the x- and y-axis since the diffusion coefficient only depends on the colloid's height. Theoretically, α can take any value between 0 and 1. However, α generally takes 3 different values [78]: $\alpha = 0$, the Itô integral, corresponding to the use of the initial value of $D(z)$; $\alpha = 1/2$, the Stratonovich integral, corresponding to the mid-point value; and $\alpha = 1$ the anti-Itô or isothermal integral, corresponding to the use of the final value.

In particular, the Itô integral $\alpha = 0$ is commonly used in economics and biology as integrals are approximated by a sum, the first point of the integral is chosen. The Stratonovich integral $\alpha = 1/2$ arises in physical systems where noise correlation time $\tau_{\mathrm{c}} > 0$, and where the velocities are often computed at mid-points using three measurements. Finally, the isothermal integral $\alpha = 1$ is used in physical systems at equilibrium with a heat bath [22] such as our case with Brownian motion. Using the Padé approximation Eq. (3.1.43), the spurious drift v_{noise} writes:

$$v_{\text{noise}}(z) = 2\alpha D_0 a \frac{2a^2 + 12az + 21z^2}{(2a^2 + 9az + z^2)^2} \ , \tag{3.1.55}$$

and the deterministic part of the drift writes:

$$v_{\text{d}} = -\frac{k_{\text{B}}T}{\gamma(z)} \left[-\frac{1}{\ell_{\text{D}}} B \exp\left(-\frac{z}{\ell_{\text{D}}} \right) + \frac{1}{\ell_{\text{B}}} \right] \ . \tag{3.1.56}$$

Finally, the Langevin equation for a confined colloid near a wall writes:

$$V_t^i \text{d}t = \bar{v}_{\text{d}}^i \text{d}t + \sqrt{2D_i(z)} \text{d}B_t \ , \tag{3.1.57}$$

where the drifts $\bar{v}_{\text{d}}^i$ are non-zero only for the z-axis. A typical example of drift velocity for a colloidal particle of radius $a = 1.5$ μm in water, confined in interaction potential with a Debye length $\ell_{\text{D}} = 50$ nm, $B = 4$ $k_{\text{B}}T$ and a Boltzmann length $\ell_{\text{B}} = 500$ nm, is plotted in Fig. 29. As one can observe, the spurious drift is not negligible.

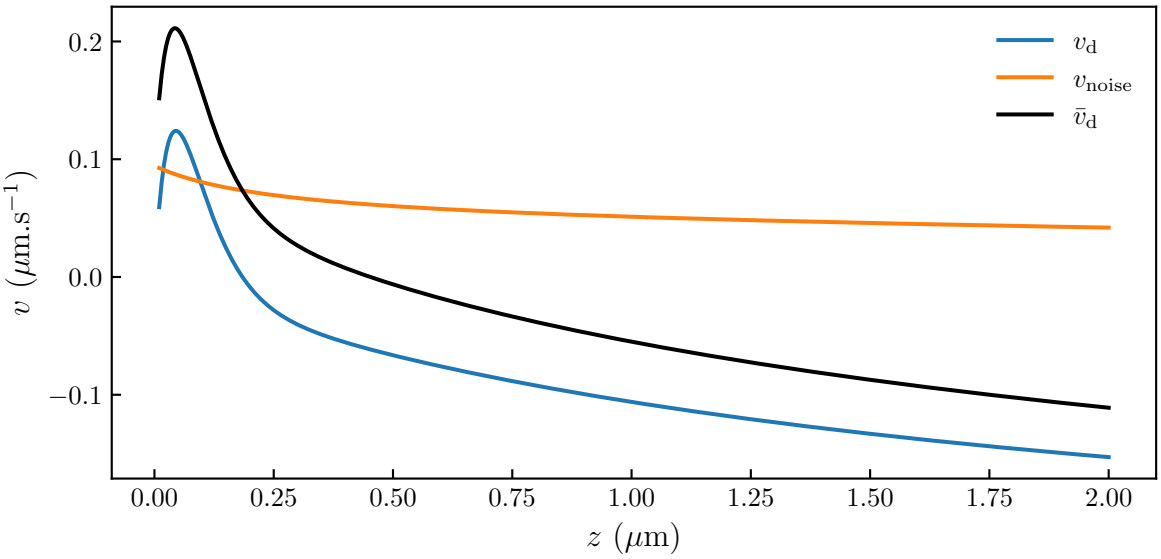


Figure 29: Typical drift velocity for a confined colloidal particle of radius $a = 1.5$ μm in water. The physical properties of the interaction are $\ell_{\text{D}} = 50$ nm, $B = 4$ $k_{\text{B}}T$ and $\ell_{\text{B}} = 500$ nm.

3.1.6 Fokker-Plank equation

The Fokker-Plank equation is an alternative way to describe Brownian motion. Instead of explicitly calculating a Brownian trajectory by solving the Langevin equation, Fokker-Plank equation describes the particle distribution function $P(x, x_0; t)$ where x denotes the particle position and x_0 its initial position. To derive the Fokker-Plank equation, let us start by taking generic Langevin equation:

$$\mathrm{d}X_t = u(X_t)\mathrm{d}t + a(X_t)\mathrm{d}B_t \ , \tag{3.1.58}$$

where X_t is the particle position, u is the drift due to external forces and a the magnitude of the random force. Let consider the average value of an arbitrary function $f(X_t)$ for a stochastic process obeying Eq. (3.1.58), which started at position x_0 at time $t = 0$, by definition [79]:

$$\langle f(X_t)\rangle = \int \mathrm{d}x \ p(x, x_0; t) f(x) \ , \tag{3.1.59}$$

with the initial conditions that can be written as:

$$p(x, x_0; 0) = \delta(x - x_0) \ . \tag{3.1.60}$$

We now take the time derivative of Eq. (3.1.59), starting by expending f at the $\mathrm{d}t$ as:

$$\begin{aligned}\left\langle \frac{\mathrm{d}f(X_t)}{\mathrm{d}t} \right\rangle &\simeq \frac{\mathrm{d}}{\mathrm{dt}} \left\langle \frac{\partial f(X_t)}{\partial x} u(X_t)\mathrm{d}t + \frac{1}{2}\frac{\partial^2 f(X_t)}{\partial x^2} a^2(X_t)\mathrm{d}B_t^2 \right\rangle \\ &= \frac{\mathrm{d}}{\mathrm{dt}} \left(\frac{\partial f(X_t)}{\partial x} u(X_t)\mathrm{d}t + \frac{1}{2}\frac{\partial^2 f(X_t)}{\partial x^2} a^2(X_t)\mathrm{d}t \right) \\ &= \frac{\partial f(X_t)}{\partial x} u(X_t) + \frac{1}{2}\frac{\partial^2 f(X_t)}{\partial x^2} a^2(X_t) \ .\end{aligned} \tag{3.1.61}$$

By combining Eqs. (3.1.59) and (3.1.61), we get:

$$\begin{aligned}\left\langle \frac{\mathrm{d}f(X_t)}{\mathrm{d}t} \right\rangle &= \int \mathrm{d}x \ \frac{\partial p(x, x_0; t)}{\partial t} f(x) \\ &= \frac{\partial f(X_t)}{\partial x} u(X_t) + \frac{1}{2} \frac{\partial^2 f(X_t)}{\partial x^2} a^2(X_t) \\ &= \int \mathrm{d}x \ p(x, x_0; t) G f(x) \ ,\end{aligned} \tag{3.1.62}$$

where G is an operator called the generator and is defined by its action on a function f as:

$$Gf = \frac{1}{2} a^2(x) \frac{\partial^2 f(x)}{\partial x^2} + u(x) \frac{\partial f(x)}{\partial x} \ . \tag{3.1.63}$$

Using the definition of the adjoint of G, denoted by $G^\dagger$, one has:

$$\begin{aligned}\int \mathrm{d}x \ \frac{\partial p(x, x_0; t)}{\partial t} f(x) &= \int \mathrm{d}x \ p(x, x_0; t) G f(x) \\ &= \int \mathrm{d}x \ G^\dagger p(x, x_0; t) f(x) \ .\end{aligned} \tag{3.1.64}$$

From the latter, we thus have:

$$\frac{\partial p(x, x_0; t)}{\partial t} = G^\dagger p(x, x_0; t) \ , \tag{3.1.65}$$

which leads to the Forward-Fokker-Planck equation:

$$\frac{\partial p(x, x_0; t)}{\partial t} = \frac{\partial^2}{\partial x^2} \left[\frac{a^2(x)}{2} p(x, x_0; t) \right] - \frac{\partial}{\partial x} \left[u(x) p(x, x_0; t) \right] \ . \tag{3.1.66}$$

The latter is called Forward because the partial differential equation is in terms of the variable x, the position of the particle at which the stochastic process ends up, at time t. For a confined Brownian motion near a wall, using the previously determined drifts $\bar{v}_\mathrm{d}$ Eq. (3.1.54), the Fokker-Plank equation for the movement along the z-axis, perpendicular to the wall writes:

$$\frac{\partial p(z, z_0; t)}{\partial t} = \frac{\partial^2}{\partial z^2} \left[D(z) p(z, z_0; t) \right] - \frac{\partial}{\partial z} \left[\bar{v}_\mathrm{d} p(z, z_0; t) \right] \ . \tag{3.1.67}$$

As an example, the stationary solution of the latter, $\partial p/\partial t = 0$, is given by the Gibbs-Boltzmann distribution $P_{\mathrm{eq}}(z)$ (see Eq. (3.1.30)). The solution for the Eq. (3.1.65) with the particle starting at position z_0, at time $t_0 = 0$ writes:

$$\begin{aligned} P(z, z_0; t) &= \exp(G^{\dagger} t) p(z, z_0, 0) \\ &= \exp\left(\frac{\partial^2}{\partial z^2} D(z) t - \frac{\partial}{\partial z} \bar{v}_{\mathrm{d}} t\right) \delta(z - z_0) \end{aligned} \tag{3.1.68}$$

3.1.7 Numerical simulations of confined Brownian motion

We had previously determined that the simulation of a bulk Brownian motion without external forces can be simulated using Eq. (1.4.9):

$$x_i = x_{i-1} + \sqrt{2D} w_i \ . \tag{3.1.69}$$

However, with the confined Brownian motion one needs to take into account the hindered diffusion, external forces due to gravity and double-layer interactions and the noise-induced drift. Putting all that together, leads to a new equation for x_i which writes for the movement parallel to wall:

$$x_i = x_{i-1} + \sqrt{2D_{\parallel}} w_i \ , \tag{3.1.70}$$

For the particle movement perpendicularly to the wall (z-axis), one needs to add the total drift $\bar{v}_{\mathrm{d}}$ Eq. (3.1.54), such that:

$$z_i = z_{i-1} + \bar{v}_{\mathrm{d}}(z_{i-1}) \tau + \sqrt{2D_{\perp}} w_i \ , \tag{3.1.71}$$

where τ is the simulation time step and w_i being a Gaussian distributed number of mean value $\langle w_i \rangle = 0$ and variance $\langle w_i^2 \rangle = \tau$. Unlike the bulk Brownian motion where the time step τ can be chosen for the desired precision as shown previously on Fig. 4. For an accurate simulation, the time step should be short enough for the drifts $\bar{v}_{\mathrm{d}}$ and local diffusion coefficient to be relatively constant in the time period $t_{i+1} - t_i = \tau$ and in the displacement range $\Delta z = z_{i+1} - z_i$, such that:

$$\bar{v}_{\mathrm{d}}(z \in [z_i, z_{i+1}]) \simeq \bar{v}_{\mathrm{d}}(z_i) \ , \tag{3.1.72}$$

and:

$$D_{\perp,\parallel}(z \in [z_i, z_{i+1}]) \simeq D_{\perp,\parallel}(z_i) \ . \tag{3.1.73}$$

Since that the diffusion coefficient varies less for the movement parallel, one can only consider the perpendicular motion to determine the simulation time step. Also, as it can be seen in Fig. 30 where the drift v_{d} and diffusivity gradient are plotted, the diffusion gradient is negligible compared to the drift gradient, we thus focus on the drifts condition Eq. (3.1.72). Moreover, the drifts vary more when the colloid is near the surface where one can approximate the diffusion coefficient $D_\perp$ using Eq. (3.1.44) such that:

$$D_\perp(z)|_{z \ll a} = D_0 \frac{z}{a} \ . \tag{3.1.74}$$

Near the surface, the gravitational interaction can be neglected as it is smaller than the electrostatic interactions as shown in Fig. 25. In that case, by taking $\alpha = 1$, the total drifts Eq. (3.1.54) near the surface simplifies to:

$$\begin{aligned} \bar{v}_{\mathrm{d}} &= \frac{k_{\mathrm{B}}T}{\gamma_0}\frac{z}{a}\left[\frac{B}{\ell_{\mathrm{D}}}\exp\left(-\frac{z}{\ell_{\mathrm{D}}}\right)\right] + \frac{\partial}{\partial z}D_0\frac{z}{a} \\ &= D_0\frac{z}{a}\left[\frac{B}{\ell_{\mathrm{D}}}\exp\left(-\frac{z}{\ell_{\mathrm{D}}}\right)\right] + \frac{D_0}{a} \\ &= \frac{D_0}{a}\left[1 + \frac{Bz}{\ell_{\mathrm{D}}}\exp\left(-\frac{z}{\ell_{\mathrm{D}}}\right)\right] \end{aligned} \tag{3.1.75}$$

By expending the exponential term to the order z/ℓ_{D} order, we get:

$$\begin{aligned} \bar{v}_{\mathrm{d}} &= \frac{D_0}{a}\left[1 + \frac{Bz}{\ell_{\mathrm{D}}}\exp\left(-\frac{z}{\ell_{\mathrm{D}}}\right)\right] = \frac{D_0}{a}\left[1 + \frac{Bz}{\ell_{\mathrm{D}}}\left(1 - \frac{z}{\ell_{\mathrm{D}}}\right)\right] \\ &= \frac{D_0}{a}\left(1 + \frac{Bz}{\ell_{\mathrm{D}}}\right) \end{aligned} \tag{3.1.76}$$

To satisfy Eq. (3.1.72), we need to have small relative change of drifts in an interval $[z, z + \Delta z]$ which can be written as [82]:

$$\frac{|\bar{v}_{\mathrm{d}}(z+\Delta z)-\bar{v}_{\mathrm{d}}(z)|}{\bar{v}_{\mathrm{d}}(z)} \ll 1 \tag{3.1.77}$$

Combining Eqs. (3.1.76) and (3.1.77), we get:

$$|\Delta| z \ll \frac{\ell_{\mathrm{D}}}{B} + z \ . \tag{3.1.78}$$

Using the MSD, it is possible to link the latter to the simulation time step τ by using the average value of Δz^2, such that:

$$\langle \Delta z^2 \rangle (z) = 2D_{\perp}(z)\tau = 2D_0 \frac{z}{a}\tau \tag{3.1.79}$$

Combining Eqs. (3.1.78) and (3.1.79) thus leads to:

$$\tau = \frac{a\langle \Delta z^2 \rangle}{2D_0 z} < \frac{a}{2D_0}\frac{(\frac{\ell_{\mathrm{D}}}{B}+z)^2}{z} = \tau_{\max}(z) \ , \tag{3.1.80}$$

where $\tau_{\max}$ is thus the maximal time step that satisfies Eq. (3.1.78). At this point, there are two different ways to do the simulation: the first one is to do an adaptive time step where $\tau_{\max}$ for each step of the simulation ; the second one is to find the smallest $\tau_{\max}(z)$ and use for all the simulation a time step τ satisfying $\tau < \min(\tau_{\max})$. The latter can be evaluated by finding the height $z_{\min}$, at which the derivative of $\tau_{\max}$ nullifies, such that:

$$\left.\frac{\partial \tau_{\max}}{\partial z}\right|_{z_{\min}} = 0 \ . \tag{3.1.81}$$

Solving the latter gives:

$$z_{\min} = \frac{\ell_{\mathrm{D}}}{B} \ . \tag{3.1.82}$$

which finally gives a maximal simulation time step, $\min(\tau_{\max})$:

$$\min(\tau_{\max}) = \frac{2a}{D_0}\frac{\ell_{\mathrm{D}}}{B} \ , \tag{3.1.83}$$

this time should be the maximal time step τ used for a confined Brownian simulation near a wall to ensure an accurate simulation of the near-wall effects. In the Fig. 30-b) some typical $\tau_{\max}$ are plotted for a colloid of radius $a = 1.5$ μm, $B = 4$ and ℓ_{D} varying between 20 and 100 nm. We can observe that for this range of values that well represents the experiments that I have done during my thesis, taking a simulation time step $\tau < 0.01$ s can be used for all set of parameters.

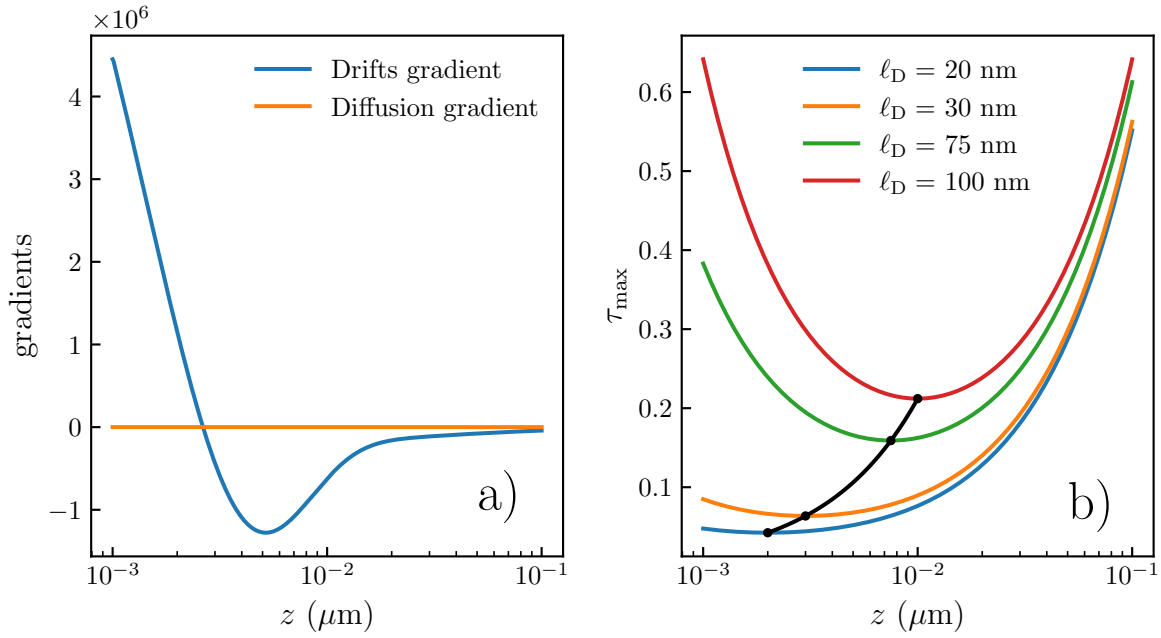


Figure 30: a) Drift and diffusive gradients for a confined colloidal particle of radius $a = 1.5$ μm in water. The physical properties of the interaction are $\ell_{\mathrm{D}} = 50$ nm, $B = 4\, k_{\mathrm{B}}T$ and $\ell_{\mathrm{B}} = 500$ nm. b) $\tau_{\max}$ for a particle of radius $a = 1.5$ μm and $B = 4$ for different Debye length. The black line represents the minimum value $\tau_{\max}$ for ℓ_{D} varying between 20 nm and 100 nm, this minimal time represents the maximal simulation time step that should be used for an accurate simulation.

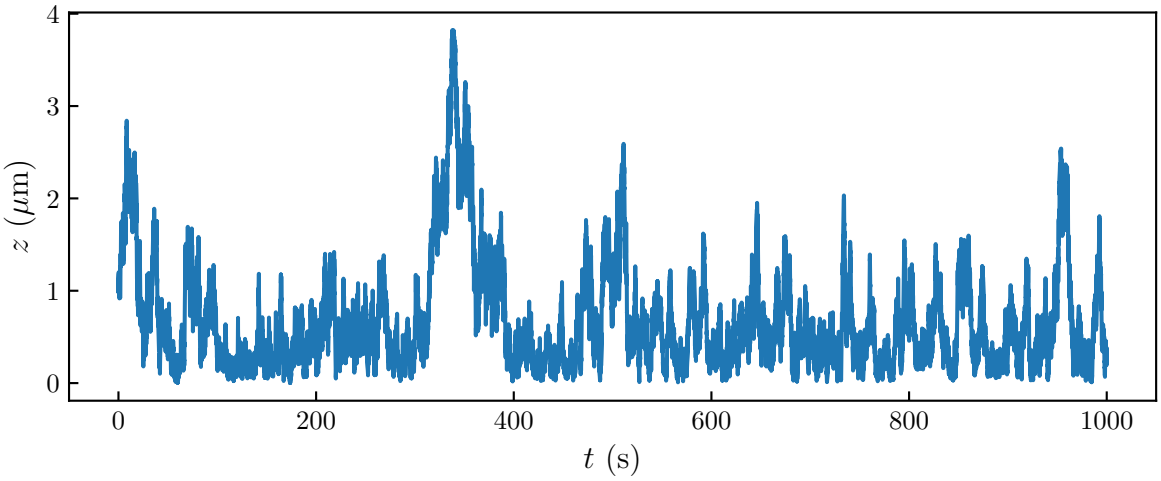


Figure 31: Simulated confined Brownian motion height trajectory of a colloidal particle of radius $a = 1.5$ μm of density $\rho_{\mathrm{p}} = 1050$ kg.m^{-3}, $\alpha = 1$ and the potential is characterized by $\ell_{\mathrm{D}} = 50$ nm and $B = 4$.

We have developed the simulation of the Confined Brownian motion using Python, as part of the Master's internship of Élodie Millan, the interested reader will find more information on confined Brownian motion in more complex systems in her forthcoming thesis. A trajectory of a colloidal particle of radius $a = 1.5$ μm of density $\rho_{\mathrm{p}} = 1050$ kg.m^{-3} and the potential characterized by $\ell_{\mathrm{D}} = 50$ nm and $B = 4$, is plotted in the Fig. 31 which does look like the experimental trajectory that was shown in Fig. 22 as an introduction to the chapter. In this trajectory, the noise induced lift is taken into account by using the isothermal approach $\alpha = 1$. However, to check if it is the correct way to take into account the spurious drift, the constraint we have is that the long-time statistics should satisfy the Gibbs-Boltzmann equation Eq. (3.1.30). To compute an experimental probability density function from a set of points, one can use the following Python snippet.

```
1        def pdf(data, bins=10, density=True):
2
3            pdf, bins_edge = np.histogram(data, bins=bins, density=density)
4            bins_center = (bins_edge[0:-1] + bins_edge[1:]) / 2
5
6            return pdf, bins_center
```

The Gibbs-Boltzmann distribution for $\alpha = 0, 0.5$ and 1 is shown in Fig. 32. We see that the Isothermal integral of the noise gives the correct distribution, in the other two cases, we observe that the particle is more likely to be found nearer the surface, this is corrected by the additional spurious drift Eq. (3.1.55).

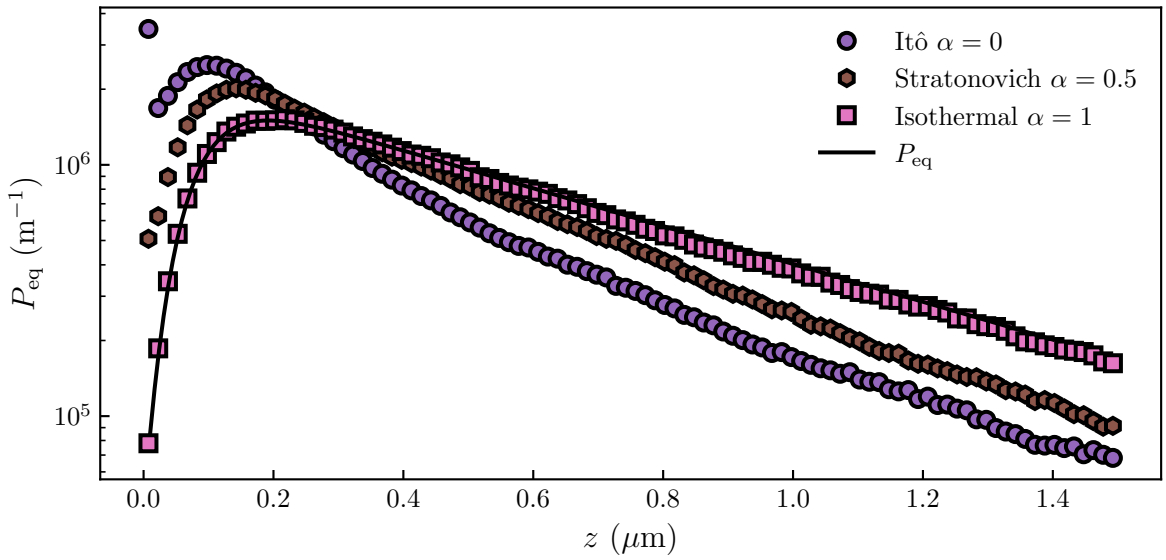


Figure 32: Probability Density Function of the height of the particle for different computation of the spurious drift, Itô $\alpha = 0$, Stratonovich $\alpha = 0.5$ and Isothermal $\alpha = 1$. The plain black line represents the expected Gibbs-Boltzmann distribution. The simulation parameters : $a = 1.5$ μm, $\rho_{\mathrm{p}} = 1050$ kg.m^{-3}, $\ell_{\mathrm{D}} = 50$ nm, $B = 4$ and $\ell_{\mathrm{B}} = 577$ nm.

3.2 Experimental study

Let us now analyze the experimental data obtain through the Mie tracking. In the theory, we wrote the height of the particle z being the shortest distance between the wall and the colloidal particle surface. However, it is not the height measured by the Mie tracking, since it measures the distance between the objective lens focal plane and the particle center. To have the correct measured height, we suppose that the particle does approach very closely to the wall. From that assumption, we then a moving-minimum along the trajectory to set the minimal value of the trajectory to zero. The moving minimum can be calculated using the following Python function:

```
1         def movmin(z, window):
2         result = np.empty_like(z)
3         start_pt = 0
4         end_pt = int(np.ceil(window / 2))
5
6         for i in range(len(z)):
7         if i < int(np.ceil(window / 2)):
8         start_pt = 0
9         if i > len(z) - int(np.ceil(window / 2)):
10        end_pt = len(z)
11        result[i] = np.min(z[start_pt:end_pt])
12        start_pt += 1
```

```
13          end_pt += 1
14
15          return result
```

In the latter, `window` represent the number of points is used to compute the minimum. As an example, if one chose `window = 10000`, the first value of `result` is the minimum of the first 10000 points of `z`. If there is enough data around the point where the minimum is calculated, the ensemble is centered, with a window of 100, the minimal value at the 100th elements is between the 50th and 150th (*e.g.* `result[100] = np.min(z[50:150])`. The raw and rescaled trajectory are shown Fig. 33. Of course, that technique is not perfect and we are working on a method that could directly measure the wanted wall-particle distance, also using Mie. However, subtract the moving minimum has a benefit, indeed it can remove some experimental drift that can be due to the physical movement of the optical pieces of the microscope. Moreover, as the exact location of the $z = 0$ origin is thus *a priori* undetermined we add to the physical parameters B, ℓ_D and ℓ_B a height offset z_{off} that accounts for the correction of the wall position.

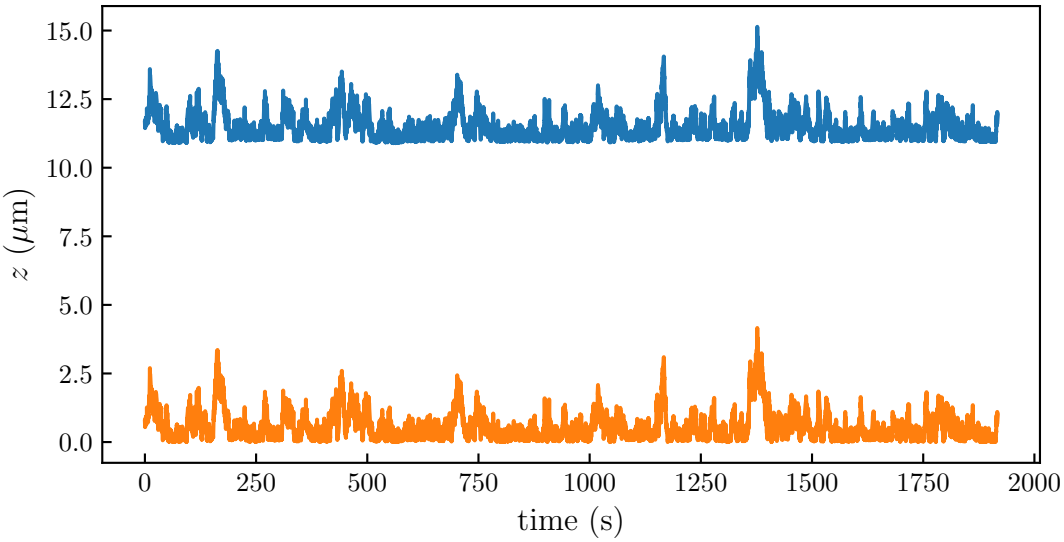


Figure 33: Raw trajectory measured using the Mie tracking technique, and it's rescaled value using moving minimum algorithm with a window of 10000 points.

3.2.1 Equilibrium distribution

As we have done for the simulated trajectory, one can construct the equilibrium probability density function $P_{eq}(z)$ of the position of the particle. As seen in Fig. 34, and explain in the previous section, an exponential tail is observed at large distance, which is identified

to the sedimentation contribution in Perrin's experiment [5], but here with the probability density function of a single particle instead of the concentration field. In contrast, near the wall, we observe an abrupt depletion, indicating a repulsive electrostatic contribution. Additionally, we see that the Gibbs-Boltzmann distribution Eq. (3.1.29) fits the data very well.

Moreover, as shown in Fig. 35, we verified that we recover the Debye relation $\ell_{\mathrm{D}} = 0.304/\sqrt{[\mathrm{Nacl}]}$, with ℓ_{D} in nm, and where [NaCl] is the concentration of salt in mol/L, with a prefactor corresponding to a single monovalent salt in water at room temperature [70]. Besides, we have verified, as shown in Fig. 35, that the dimensionless parameter B related to surface charges is constant in the studied salt-concentration range, thus excluding any nonlinear effect [80, 81] in our case.

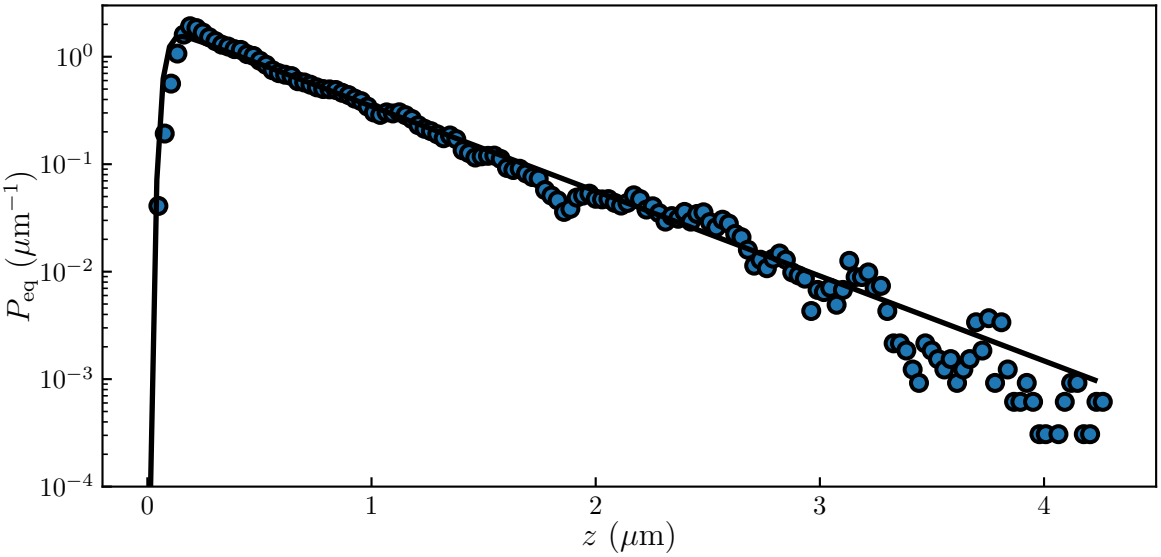


Figure 34: Measured equilibrium probability density function P_{eq} of the distance z between the particle and the wall. The solid line represents the best fit to the normalized Gibbs-Boltzmann distribution in position, using the total potential energy $U(z)$ of Eq. (3.1.29), with $B = 4.8$, $\ell_{\mathrm{D}} = 21$ nm, and $\ell_{\mathrm{B}} = 530$ nm.

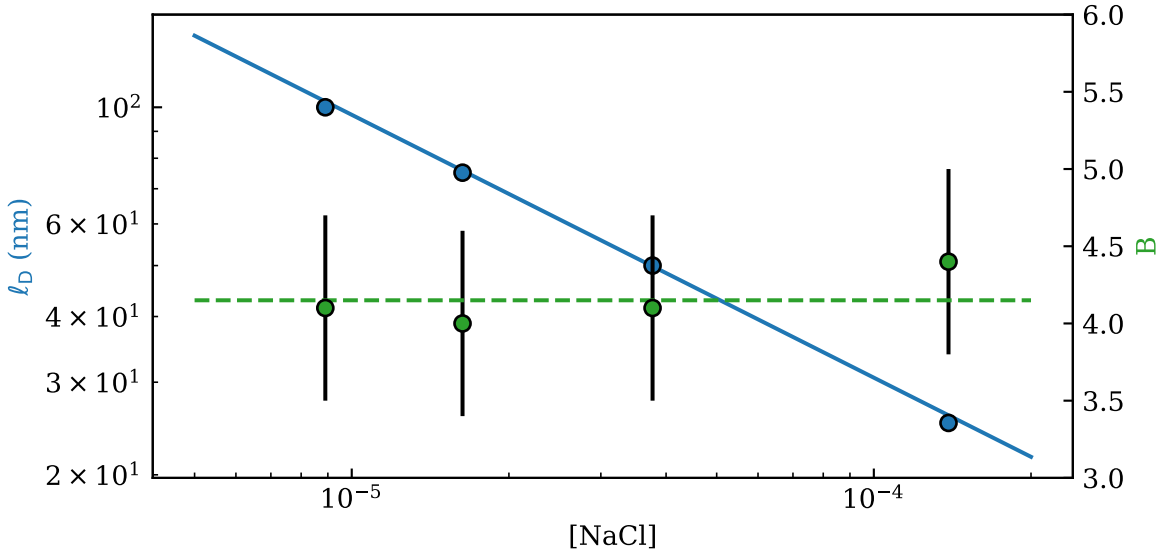


Figure 35: In blue, left axis, measured Debye length ℓ_D as a function of salt concentration [NaCl]. The solid line is the expected Debye relation $\ell_D = 0.304/\sqrt{[\text{Nacl}]}$, for a single monovalent salt in water at room temperature. In green, right axis, measured B as a function of salt concentration [NaCl]. The dashed line represents the mean value of the measured B values.

3.2.2 Mean Square Displacement

We now turn to dynamic aspects, by considering the mean-squared displacement (MSD). For the three spatial directions, indexed by $i = x$, y, and z, corresponding to the coordinates $r_x = x$, $r_y = y$, and $r_z = z$, of the position $\vec{r}$, and for a given time increment Δt, the MSD is defined as:

$$\langle \Delta r_i(t)^2 \rangle_t = \langle [r_i(t + \Delta t) - r_i(t)]^2 \rangle_t \ , \tag{3.2.1}$$

where the average $\langle \rangle_t$ is performed over time t. For a free Brownian motion in the bulk, and in the absence of other forces than the dissipative and random ones, the MSD is linear in time, *i.e.* $\langle \Delta r_i(t)^2 \rangle_t = 2D_0 \Delta t$, where $D_0 = k_B T/(6\pi\eta a)$ is the bulk diffusion coefficient given by the Stokes-Einstein relation [4], and η is the liquid viscosity. Further including sedimentation restricts the validity of the previous result along z to short times only, *i.e.* for $\Delta t \ll \ell_B^2/D_0$ such that the vertical diffusion is not yet affected by the gravitational drift.

The presence of a rigid wall at $z = 0$ adds a repulsive electrostatic force along z. It also decreases the mobilities nearby through hydrodynamic interactions, leading to effective viscosities $\eta_\parallel(z) = \eta_x(z) = \eta_y(z)$, and $\eta_\perp(z) = \eta_z(z)$. Interestingly, despite the previous

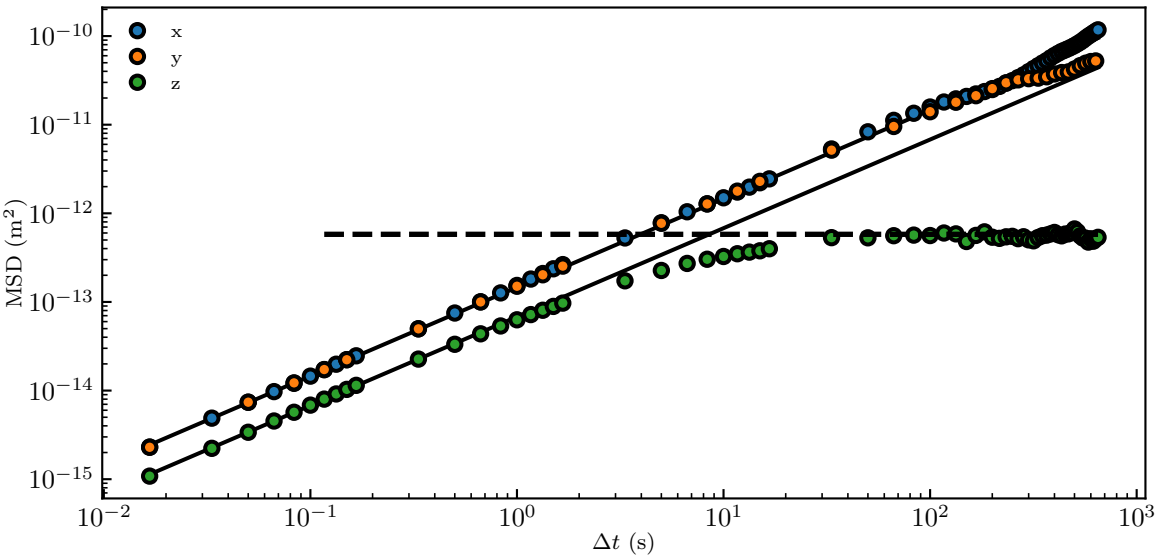


Figure 36: Measured mean-squared displacements (MSD, see Eq. (3.2.1)) as functions of the time increment Δt, for the three spatial directions, x, y, and z. The solid lines are best fits to Eq. (3.2.2), using Eqs. (3.1.29), (3.1.41) and (3.1.42), with $B = 4.8$, $\ell_{\mathrm{D}} = 21$ nm, and $\ell_{\mathrm{B}} = 530$ nm, providing the average diffusion coefficients $\langle D_{\parallel} \rangle = \langle D_x \rangle = \langle D_y \rangle = 0.52\, D_0$ and $\langle D_z \rangle = 0.24\, D_0$. The dashed line is the best fit to Eq. (3.2.10), using Eq. (3.1.29), with $B = 4.8$, $\ell_{\mathrm{D}} = 21$ nm, and $\ell_{\mathrm{B}} = 530$ nm.

modifications, the temporal linearity of the MSD is not altered by the presence of the wall [7, 19] for x and y, as well as at short times for z. In such cases, the MSD reads:

$$\langle \Delta r_i(t)^2 \rangle_t = 2 \langle D_i \rangle \Delta t \ , \tag{3.2.2}$$

where for each spatial direction we introduced the local diffusion coefficient $D_i(z) = D_0 \eta / \eta_i(z)$, and its average

$$\langle D_i(z) \rangle = \int_0^\infty \mathrm{d}z \, D_i(z) P_{\mathrm{eq}}(z) \ , \tag{3.2.3}$$

against the Gibbs-Boltzmann distribution in position. As shown in Fig. 36, the MSD measured along x or y is indeed linear in time. By fitting the data to Eq. (3.2.2), using Eqs. (3.1.29) and (3.1.42), we extract an average transverse diffusion coefficient $\langle D_{\parallel} \rangle = \langle D_x \rangle = \langle D_y \rangle = 0.52\, D_0$. In contrast, along z, we identify two different regimes: one at short times, where the MSD is still linear in time, with a similarly obtained best-fit value of $\langle D_z \rangle = 0.24\, D_0$; and one at long times, where the MSD saturates to a plateau. This latter behavior indicates that the equilibrium regime has been reached, with the particle having essentially explored all the relevant positions given by the Gibbs-Boltzmann distribution.

3.2.3 Non-Gaussian dynamics - Displacement distribution

Having focused on the MSD, *i.e.* on the second moment only, we now turn to the full probability density function P_i of the displacement Δr_i. Since the diffusion coefficient $D_i(z)$ varies as a result of the variation of z along the particle trajectory, P_i exhibits a non-Gaussian behavior, as seen in Figs. 37-a,b,c,d). We even resolve the onset of a non-Gaussian behavior in P_x, by zooming on the large-$|\Delta x|$ wings. At short times, the diffusion coefficient D_i and the drift velocity $\bar{v}_\mathrm{d}$, can be considered constant. By writing the initial condition $\delta(x_i - x_i^0)$, the solution of the Forward Fokker-Plank Eq. (3.1.68), becomes:

$$\begin{aligned} P(x_i, z_0, \Delta t) &= \exp\left[\frac{\partial^2}{\partial z^2} D_i(z_0)\Delta t - \frac{\partial}{\partial z}\bar{v}_\mathrm{d}^i(z_0)\Delta t\right] \frac{1}{2\pi}\int_{-\infty}^{\infty} \mathrm{d}u \exp(ju(z - z_0)) \\ &= \frac{1}{2\pi}\int_{-\infty}^{\infty} \mathrm{d}u \left[D_i(z_0)\Delta t \frac{\partial^2}{\partial z^2}\exp(ju(z - z_0)) - \bar{v}_\mathrm{d}^i(z_0)\Delta t\frac{\partial}{\partial z}\exp(ju(z - z_0))\right] \\ &= \frac{1}{2\pi}\int_{-\infty}^{\infty} \mathrm{d}u \exp\left[-u^2 D(z_0)\Delta t + ju(z - z_0) - ju\bar{v}_\mathrm{d}^i(z_0)\Delta t\right] \ , \end{aligned} \tag{3.2.4}$$

where $\bar{v}_\mathrm{d}^i(z_0)$ is non-zero only for the z-axis. The latter can be reduced to [82, 83]:

$$P_i(\Delta r_i, z_0, \Delta t) = \frac{1}{\sqrt{4\pi D_i(z_0)\Delta t}} \exp\left[-\frac{(\Delta r_i - \bar{v}_\mathrm{d}^i\Delta t)^2}{4D_i(z)\Delta t}\right] \ . \tag{3.2.5}$$

Which is a Gaussian distribution with a 0 mean value for the x- and y-axis and

$$\langle P_z(\Delta z, z_0, \Delta t)\rangle = \bar{v}_\mathrm{d}\Delta t \ , \tag{3.2.6}$$

for the z-axis. Additionally, it has a variance $\sigma_i(z_0) = \sqrt{2D_i(z_0)\Delta t}$. From Eq. (3.2.5), we can observe than the total drift $\bar{v}_\mathrm{d}$ induces an asymmetry on the displacement along the z-axis. However, in our experiment, as we have access to long enough trajectory, we are interested in the distribution which is not conditioned by the initial position but by the Gibbs-Boltzmann distribution Eq. (3.1.29). At short times, P_i can thus be modeled by the averaged diffusion Green's function [18, 84]:

$$\begin{aligned} P_i(\Delta r_i, \Delta t) &= \int_0^{\infty} \mathrm{d}z_0 P_\mathrm{eq} P(x_i, z_0, \Delta t) \\ &= \int_0^{\infty} \mathrm{d}z \, P_\mathrm{eq}(z) \frac{1}{\sqrt{4\pi D_i(z)\Delta t}} \mathrm{e}^{-\frac{\Delta r_i^2}{4D_i(z)\Delta t}} \ , \end{aligned} \tag{3.2.7}$$

against the Gibbs-Boltzmann distribution. Which can alternatively be written as an integral over the diffusion such that:

$$P(\Delta r_i, \Delta t) = \int_0^\infty dD_i P(D_i) \frac{1}{\sqrt{4\pi D_i \Delta t}} \mathrm{e}^{\frac{-\Delta r_i^2}{4D_i \Delta t}} \tag{3.2.8}$$

The latter can be evaluated using the following Python snippet.

```
def P_D(B, ld, lb):
# Computing the D PDF.
z = np.linspace(1e-9, 15e-6, 1000)
P_D = Dz(z) * P_b_off(z, offset, B, ld, lb)
P_D = P_D / np.trapz(P_D, z)  # extra step to ensure PDF normalization
return Dz, P_D


def _P_Dz_short_time(Dz, Dt, B, ld, lb):
# Using the D PDF to compute the P()
D_z, P_D = P_D(B, ld, lb)
P = P_D / np.sqrt(4 * np.pi * D_z * Dt) * np.exp(-(Dz ** 2) / (4 * D_z * Dt))
P = np.trapz(P, D_z)
return P


# Wrapping the previous function of easier use for Dz arrays.
def P_Dz_short_time(Dz, Dt, B, ld, lb):
P = np.array([_P_Dz_short_time(i, Dt, B, ld, lb) for i in Dz])
P = P / np.trapz(P, Dz) # extra step to ensure PDF normalization
return P
```

In the latter snippet, the evaluation is done for Δz, however, to compute $P_x(\Delta x)$ one should just change the `Dz(z)` function to compute the parallel diffusion coefficient $D_\parallel$. Since P is a PDF, it should be normalized such that $\int P = 1$, I added an extra step to ensure PDF normalization along the evaluation. Since, we have reached equilibrium, the averaged particle's drift should be equal to 0 thus leading to a mean value of the distribution Eq. (3.2.7), $\langle P_i(\Delta r_i, \Delta t) \rangle = 0$. As shown in Figs.37-a,c,b,d) Eq. (3.2.7) captures the early data very well. At long times, Eq. (3.2.7) remains valid only for P_x and P_y. Nevertheless, the equilibrium regime being reached, P_z only depends on the Gibbs-Boltzmann distribution and can eventually be written as:

$$\lim_{\Delta t \to \infty} P_z(\Delta z) = \int_0^\infty \mathrm{d}z \, P_{\mathrm{eq}}(z + \Delta z) P_{\mathrm{eq}}(z) \ , \tag{3.2.9}$$

which contains in particular the second moment:

$$\lim_{\Delta t\to\infty}\langle\Delta z^2\rangle = \int_{-\infty}^{+\infty} \mathrm{d}\Delta z\, \Delta z^2 \int_0^{\infty} \mathrm{d}z\, P_{\mathrm{eq}}(z+\Delta z)P_{\mathrm{eq}}(z)\ . \tag{3.2.10}$$

As shown in Fig. 37-e), Eq. (3.2.9) captures the long-term data along z very well. Additionally, Eq. (3.2.10) permits to fit the plateau of the MSD as shown in the Fig. 36. Eq. (3.2.9) can be evaluated using the following Python function:

```
1         def _Pdeltaz_long(DZ, B, ld, lb):
2         z = np.linspace(0, 20e-6, 1000)
3         dP = P_eq(z, B, ld, lb) * P_eq(z + DZ, B, ld, lb)
4         P = trapz(dP,z)
5         return P
6
7         def Pdeltaz_long(DZ, B, ld, lb):
8         pdf = np.array([_Pdeltaz_long(i,B, ld, lb) for i in DZ])
9         pdf = pdf / trapz(pdf,DZ)
10        return pdf
11
```

Where the `P_eq` function has been described in the section 3.1.2.

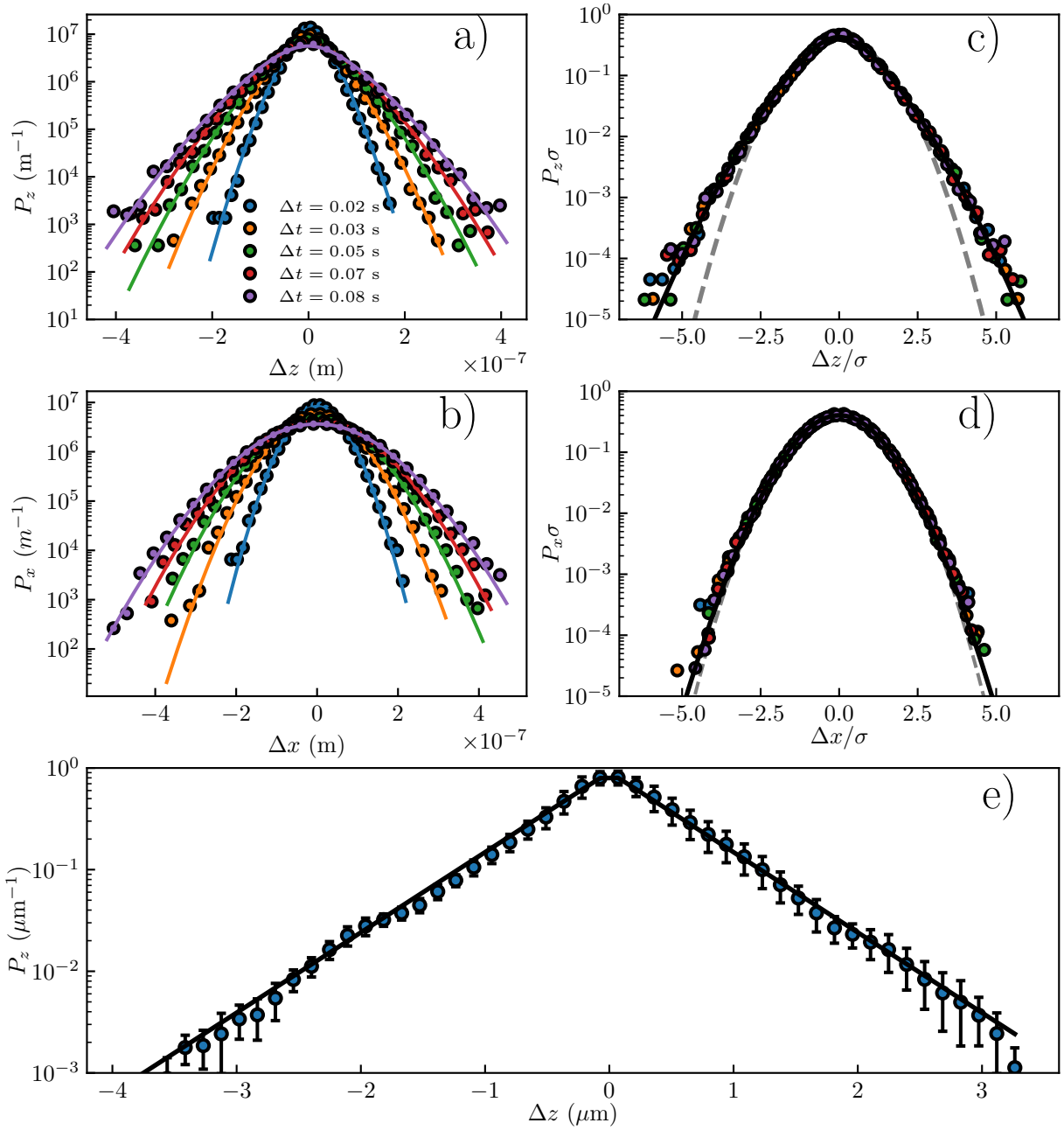


Figure 37: a, b) Probability density functions P_i of the displacements Δx and Δz, at short times. The solid lines are the best fits to Eq. (3.2.7), using Eqs. (3.1.29), (3.1.41), and (3.1.42), with $B = 4.8$, $\ell_D = 21$ nm, and $\ell_B = 530$ nm. c,d) Normalized probability density functions $P_i\,\sigma$ of the normalized displacements $\Delta x/\sigma$ and $\Delta z/\sigma$, at short times, with σ^2 the corresponding MSD (see panel Fig. 36), for different time increments Δt ranging from 0.0167 s to 0.083 s, as indicated with different colors. The solid lines are the best fits to Eq. (3.2.7), using Eqs. (3.1.29), (3.1.41), and (3.1.42), with $B = 4.8$, $\ell_D = 21$ nm, and $\ell_B = 530$ nm. For comparison, the gray dashed lines are normalized Gaussian distributions, with zero means and unit variances. d) Probability density function P_z of the displacement Δz, at long times, averaged over several values of Δt ranging between 25 s and 30 s. The solid line is the best fit to Eq. (3.2.9), using Eq. (3.1.29), with $B = 4.8$, $\ell_D = 21$ nm, and $\ell_B = 530$ nm.

3.2.4 Local diffusion coefficient inference

We now wish to go beyond the previous average $\langle D_i \rangle$ of Eq. (3.2.2), and resolve the local diffusion coefficient $D_i(z)$. To measure local viscosity from experimental trajectories, a binning method is generally employed [85]. This method consists of constructing the displacement PDF conditioned on the particle height, a measure the distribution's variance $\sigma_i(z_0) = \sqrt{2D_i(z_0)\Delta t}$ as in Eq. (3.2.5).

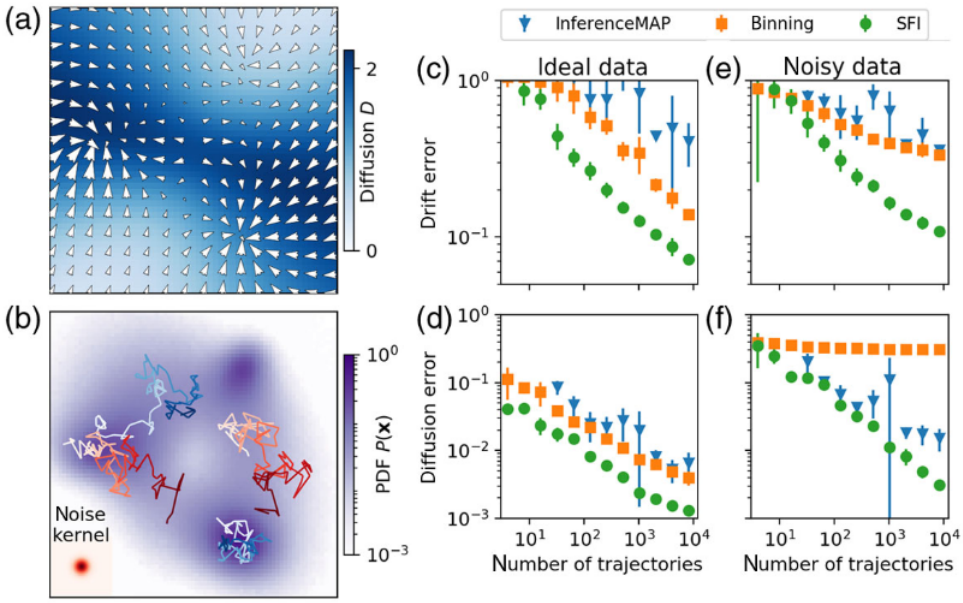


Figure 38: Figure from [86]. Quantitative comparison of Surface Force Inference (SFI with other methods on a simulated system mimicking 2D single-molecule trajectories in a complex environment with space-dependent isotropic diffusion. a) The diffusion field (blue gradient) and drift field (white arrows). b) The steady-state distribution function (PDF) of the process. The traces are representative trajectories of 100 time steps. c-f) Comparison of the performance of SFI and two widely used inference methods: InferenceMAP, a method for single-molecule inference (blue triangles) [87], and grid-based binning with maximum-likelihood estimation [85, 88] (orange squares). They evaluated the performance of these methods on the approximation of the drift field (c),e)) and diffusion field (d)f)) as a function of the number N of single-molecule trajectories (similar to the ones in panel b)) used. With ideal data (c),d)) and in the presence of measurement noise(e),f)). The performance is evaluated as the average mean-squared error on the reconstructed field along trajectories. SFI outperforms both other methods in all cases. More information about the parameters of their simulation and analysis can be found in their work [86].

Although the binning method is well suited for drift measurements, it suffers from a lack of convergence and precision when second moments or local diffusion coefficients have to be extracted. In particular, the binning method did not allow us to measure specifically the local diffusion coefficient in the key interfacial region corresponding to

$z < 100$ nm. Indeed, as we can observe in the Fig. 38-f) the diffusion error on noisy (such as experimental) data does saturate of the binning method is outperformed by a robust developed method recently developed by Frishman and Ronceray [86]. This method uses Stochastic Force Inference (SFI), in order to evaluate spatially varying force fields and diffusion coefficients, from the information contained within the trajectories.

In practice SFI can reconstruct force and diffusion fields and measure entropy production from Brownian trajectories. SFI is based on the communication-theory notion of capacity which is an information-theoretic bound, when the system is viewed as a communication channel, that limit at which rate information about the fields can be extracted from a Brownian trajectory. To explain the method, let us consider a Brownian particle that obey to the equation similar to Eq. (3.1.57), using μ to denotes the x-, y- and z-axis:

$$\dot{x}_\mu = F_\mu(x) + \partial_\nu D(x)_{\mu\nu} + \sqrt{2D(x)}_{\mu\nu} \mathrm{d}B_t \ , \tag{3.2.11}$$

where the Einstein summation is used over repeated indices, and the force field $F_\mu(x)$ and the diffusion tensor $D_{\mu\nu}$ are assumed to be time-independent. This method is built to measure the entire diffusion matrix, which for a particle diffusing above a wall has non-zero the diagonal term (*i.e.* D_{xx}, D_{yy} and D_{zz}). However, for more complex environment such as cellular environment as shown on Fig. 38-a), non-diagonal terms exist leading to correlations between the displacement axis. As an example, it has been observed in elastohydrodynamics [89] that the parallel movement near a soft wall can induce perpendicular forces, hence leading in that peculiar case, to correlation between the x and z movement. The SFI idea is to approximate the diffusion field as a linear combination of a basis of n_b known functions $b = b_\alpha(x)_{\alpha=1,\dots,n_b}$. For the diffusion of spherical spheres near a wall where the diffusion coefficient is given by the Eqs. (3.1.41) and (3.1.42) we got robust results using their built-in polynomial basis:

$$b = \{b_\alpha(x)\}_{\alpha=1,\dots,n_b} = \{1, x_\mu, x_\mu x_\nu, ...\} \text{ (up to the order } n_b \text{)}. \tag{3.2.12}$$

They perform this approximation by projecting the diffusion field onto the space spanned by $b_\alpha(x)$ using the position PDF, P_{eq} (see Eq. (3.1.29)) as a measure. This corresponds to a least-squares fit of the diffusion field by linear combinations of the b_α's. To do this fit, they define a projector:

$$c_\alpha(x) = B^{1/2}_{\alpha\beta} b_\beta(x) \ , \tag{3.2.13}$$

where $B^{1/2}_{\alpha\beta}$ is an orthonormalization matrix such that $\int c_\alpha c_\beta P_{\mathrm{eq}}(x)\mathrm{d}x = \delta_{\alpha\beta}$. They then approximate $D_{\mu\nu}$ by its projection as a linear combination of known functions:

$$D_{\mu\nu}(x) \simeq D_{\mu\nu\alpha} c_\alpha(x) \ , \tag{3.2.14}$$

with,

$$D_{\mu\nu\alpha} = \int D_{\mu\nu}(x) c_\alpha(x) P_{\mathrm{eq}} \mathrm{d}x \ . \tag{3.2.15}$$

As the equation here are over a whole trajectory which is long enough to satisfy P_{eq}, the empirical projector $\hat{c}_\alpha$ can be approximated using the trajectory averages. To measure the local diffusion they construct a local estimator $\hat{d}_{\mu\nu} = \Delta x_\mu(t_i)\Delta x_\nu(t_i)/2\Delta t$, so that $D_{\mu\nu\alpha}$ now reads:

$$\hat{D}_{\mu\nu\alpha} = \frac{1}{\tau}\sum_i \hat{d}_{\mu\nu}(t_i)\hat{c}_\alpha(x(t_i))\Delta t). \tag{3.2.16}$$

The latter equation corresponds to a linear regression of $\hat{d}_{\mu\nu}(t_i)$, and permits to project the diffusion tensor onto a finite set of basis functions. We implemented this method, using fourth-order polynomials in our case. to simplify the use of their method with our data, I developed a simple Python function around their method which can infer the local diffusion coefficient by only one function call:

```
1
2       Dx, Dy, Dz, z_D = Compute_diffusion(pos)
```

Where in the latter `pos`, is the 3D trajectory of a Brownian colloid. It allowed us to infer the local diffusion coefficients $D_i(z)$, down to $z = 10$ nm, as shown in Fig. 39. The results are in excellent agreement with the theoretical predictions, $D_\parallel(z)$ and $D_z(z)$, using Eqs. (3.1.41) and (3.1.41), thus validating the method.

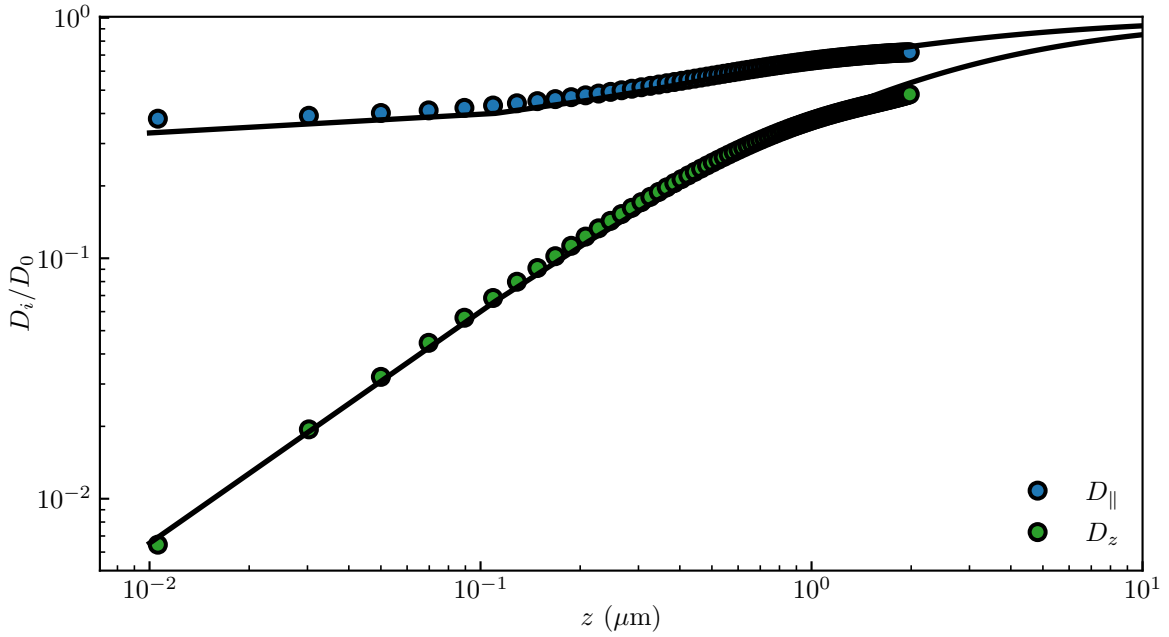


Figure 39: Measured local short-term diffusion coefficients D_i of the microparticle, normalized by the bulk value D_0, as functions of the distance z to the wall, along both a transverse direction x or y ($D_i = D_\parallel = D_x = D_y$, blue) and the normal direction z ($D_i = D_z$, green) to the wall. The solid lines are the theoretical predictions, $D_\parallel(z) = D_0\eta/\eta_\parallel(z)$ and $D_z(z) = D_0\eta/\eta_z(z)$, using the local effective viscosities $\eta_\perp(z)$ and $\eta_\parallel(z)$ of Eqs. (3.1.41) and (3.1.41), respectively.

3.2.5 Precise potential inference using multi-fitting technique

So far, through Figs.34 to 39, we have successively presented the various measured statistical quantities of interest, as well as their fits to corresponding theoretical models. Therein, we have essentially three free physical parameters, B, ℓ_B, ℓ_D, describing the particle and its environment, as well as the *a priori* undetermined location of the $z = 0$ origin. These four parameters are actually redundant among the various theoretical models. Therefore, in order to measure them accurately, we in fact perform all the fits simultaneously, using a Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm that is well suited for unconstrained nonlinear optimization [90]. To do so, we construct a global minimizer:

$$\chi^2 = \sum_{n=1}^{N} \chi_n^2 \ , \tag{3.2.17}$$

where we introduce the minimizer χ_n^2 of each set n among the N sets of data, defined as:

$$\chi_n^2 = \sum_{i=1}^{M_n} \frac{[y_{ni} - f_n(x_{ni}, \mathbf{b})]^2}{f_n(x_{ni}\ , \mathbf{b})^2} \ , \tag{3.2.18}$$

with $\{x_{ni}, y_{ni}\}$ the experimental data of set n, M_n the number of experimental data points for set n, f_n the model for set n, and $\mathbf{b} = (b_1, b_2, ..., b_p)$ the p free parameters. In our case, $p = 4$, and $\{x_{ni}, y_{ni}\}$ represent all the experimental data shown in Figs.34 to 39.

Due to strong dependence of the normal diffusion coefficient D_z with z, it is possible to find the wall position with a 10 nm resolution, thus overcoming a drawback of the Lorenz-Mie technique which only provides the axial distance relative to the focus of the objective lens. Besides, the three physical parameters globally extracted from the multifitting procedure are: $B = 4.8 \pm 0.6$, $\ell_{\mathrm{D}} = 21 \pm 1$ nm, and $\ell_{\mathrm{B}} = 530 \pm 2$ nm. Using the particle radius $a = 1.518 \pm 0.006$ μm calibrated from the preliminary fits of the interference patterns to the Lorenz-Mie scattering function (see section 2.7.3), and the 1050 kg.m^{-3} tabulated bulk density of polystyrene, we would have expected $\ell_{\mathrm{B}} = 559$ nm instead, which corresponds to less than 2 % error, and might be attributed to nanometric offsets, such as *e.g.* the particle and/or wall rugosity.

3.2.6 Measuring external forces using the local drifts

Finally, we investigate the total conservative force $F_z(z)$ acting on the particle along z. The first method way to measure it is to calculate the gradient of the potential U which is experimentally measured from the position PDF giving:

$$F_z^{\mathrm{eq}} = -\nabla U = k_{\mathrm{B}}T \frac{\partial}{\partial z} \ln(P_{\mathrm{eq}}) \ , \tag{3.2.19}$$

where one can use the experimentally measured P_{eq} (see Fig. 34) the results of this method is shown in fig.40. However, it can be interesting to measure the forces using the local drifts as for more complex systems, some non-conservative forces could arise. As the Eq. (3.2.19) only take into account to the potential U, only conservative forces can be extracted from the measurement of P_{eq}. For more complex system, non-conservative forces could be measured by the difference between forces obtained through P_{eq} and the local drifts. By averaging the overdamped Langevin equation over a fine-enough z-binning grid and short enough time interval Δt, one gets in the Itô convention (corresponding to our definition of Δz):

$$F_z(z) = 6\pi\eta_z(z)a\frac{\langle \Delta z \rangle}{\Delta t} - k_{\mathrm{B}}T\frac{D_z'(z)}{D_z(z)} \ , \tag{3.2.20}$$

where the last term corresponds to the additional contribution due to the non-trivial integration of the multiplicative noise [22, 91–93], with the prime denoting the derivative with respect to z. From the averaged measured vertical drifts $\langle \Delta z \rangle$, and invoking Eq. (3.1.43),

one can reconstruct $F_z(z)$ from Eq. (3.2.20), as shown in Fig. 40. We stress that the statistical error on the force measurement is comparable to the thermal-noise limit [94]:

$$\Delta F = \sqrt{24\pi k_{\mathrm{B}} T \eta_z(z) a / \tau_{\mathrm{box}}(z)} \ , \tag{3.2.21}$$

where $\tau_{\mathrm{box}}(z)$ is the total time spent by the particle in the corresponding box of the z-binning grid. To corroborate these measurements, we invoke Eq. (3.1.29) and express the total conservative force $F_z(z) = -U'(z)$ acting on the particle along z:

$$F_z(z) = k_{\mathrm{B}} T \left(\frac{B}{\ell_{\mathrm{D}}} \mathrm{e}^{-\frac{z}{\ell_{\mathrm{D}}}} - \frac{1}{\ell_{\mathrm{B}}} \right) \ . \tag{3.2.22}$$

Using the physical parameters extracted from the above multifitting procedure, we plot Eq. (3.2.22) in Fig. 40. The agreement with the data is excellent, thus showing the robustness of the force measurement. In particular, we can measure forces down to a distance of 40 nm from the surface. Besides, far from the wall, we are able to resolve the actual buoyant weight $F_{\mathrm{g}} = -7 \pm 4$ fN of the particle. This demonstrates that we reach the femtoNewton resolution, and that this resolution is solely limited by thermal noise.

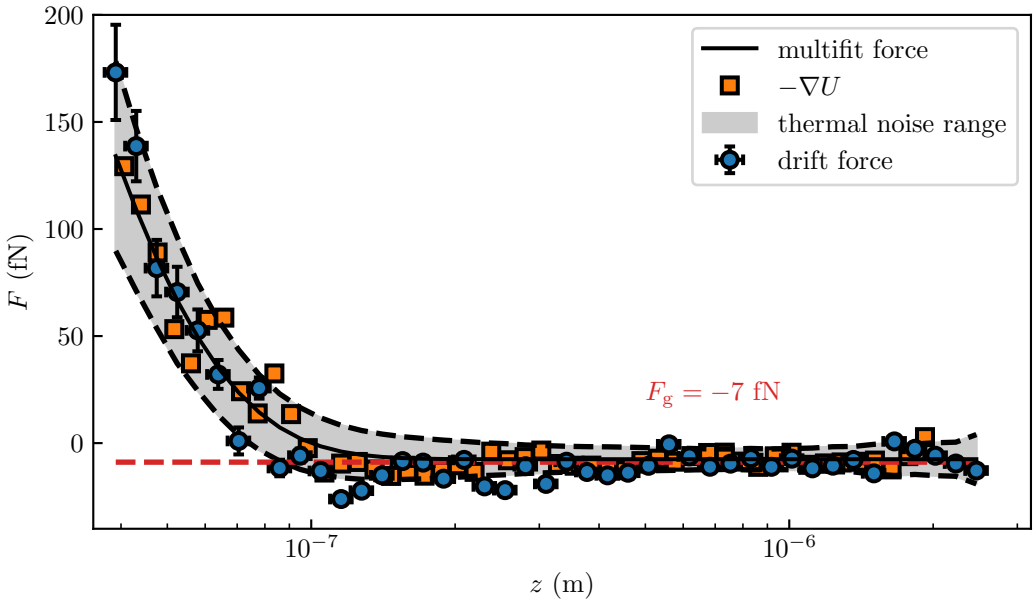


Figure 40: Total normal conservative force F_z exerted on the particle as a function of the distance z to the wall, reconstructed from Eq. (3.2.20), using Eq. (3.1.43) in circles and using Eq. (3.2.19) in squares. The solid line corresponds to Eq. (3.2.22), with $B = 4.8$, $\ell_{\mathrm{D}} = 21$ nm and $\ell_{\mathrm{B}} = 530$ nm. The black dashed lines and gray area indicate the amplitude of the thermal noise computed from Eq. (3.2.21). The horizontal red dashed line indicates the buoyant weight $F_{\mathrm{g}} = -7$ fN of the particle.

3.3 Conclusion

To conclude, we have successfully built a multi-scale statistical analysis for the problem of freely diffusing individual colloids near a rigid wall. Combining the equilibrium distribution in position, time-dependent non-Gaussian statistics for the spatial displacements, a novel method to infer local diffusion coefficients, and a multifitting procedure, allowed us to reduce drastically the measurement uncertainties and reach the nanoscale and thermal-noise-limited femtoNewton spatial and force resolutions, respectively. The ability to measure tiny surface forces, locally, and at equilibrium, as well the possible extension of the method to non-conservative forces and out-of-equilibrium settings [95, 96], opens fascinating perspectives for nanophysics and biophysics.

A Appendix

1 Intertial Brownian motion simulation

One can write the Langevin equation as:

$$m\ddot{x} = -\gamma\dot{x} + \sqrt{2k_\mathrm{B}T\gamma}\mathrm{d}B_t \tag{1}$$

By replacing with the Euler method $\dot{x}$ by:

$$\dot{x} \simeq \frac{x_i - x_{i-1}}{\tau}\,, \tag{2}$$

$\ddot{x}$ by:

$$\begin{aligned}\ddot{x} &\simeq \frac{\frac{x_i - x_{i-1}}{\tau} - \frac{x_{i-1} - x_{i-2}}{\tau}}{\tau} \\ &= \frac{x_i - 2x_{i-1} + x_{i-2}}{\tau^2}\,.\end{aligned} \tag{3}$$

and finally, $\mathrm{d}B_t$ by a Gaussian random number w_i with a zero mean value and a τ variance, on can write x_i as:

$$x_i = \frac{2 + \tau/\tau_\mathrm{B}}{1 + \tau/\tau_\mathrm{B}}x_{i-1} - \frac{1}{1 + \tau/\tau_\mathrm{B}}x_{i-2} + \frac{\sqrt{2k_\mathrm{B}T\gamma}}{m(1 + \tau/\tau_\mathrm{B})}\tau w_i\,, \tag{4}$$

We will in the following use Python to simulate such a movement and check the properties of the mean squared displacement. In the end I will propose a Cython implementation that permits a 1000x speed improvement on the simulation.

[1]:
```
# Import important libraries
import numpy as np
import matplotlib.pyplot as plt
```

[2]:
```
# Just some matplotlib tweaks
import matplotlib as mpl

mpl.rcParams["xtick.direction"] = "in"
mpl.rcParams["ytick.direction"] = "in"
mpl.rcParams["lines.markeredgecolor"] = "k"
mpl.rcParams["lines.markeredgewidth"] = 1.5
mpl.rcParams["figure.dpi"] = 200
from matplotlib import rc

rc("font", family="serif")
rc("text", usetex=True)
rc("xtick", labelsize="medium")
rc("ytick", labelsize="medium")
rc("axes", labelsize="large")
```

```
def cm2inch(value):
    return value / 2.54
```

[3]:
```
N = 1000000  # length of the simulation
tau = 0.01  # simulation time step
m = 1e-8  # particle mass
a = 1e-6  # radius of the particle
eta = 0.001  # viscosity (here water)
gamma = 6 * np.pi * eta * a
kbT = 4e-21
tauB = m / gamma
```

[4]:
```
print(
    "With such properties we have a characteristic diffusion time of {:.2f} s".
 ↪format(
        tauB
    )
)
```

```
With such properties we have a characteristic diffusion time of 0.53 s
```

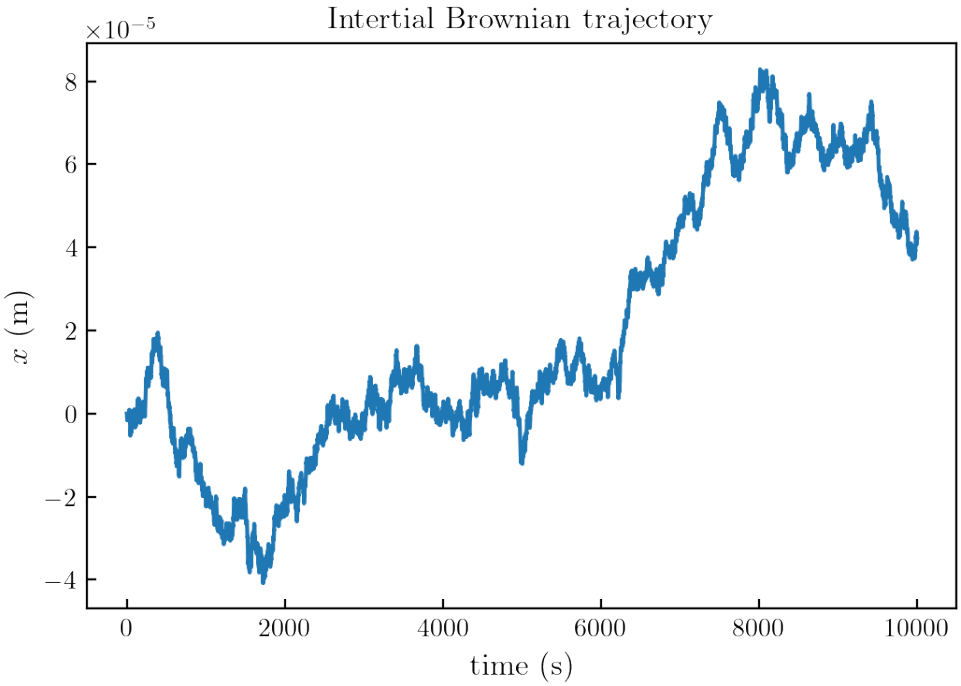
[5]:
```
def xi(xi1, xi2):
    """
    Function that compute the position of a particle using the full Langevin␣
 ↪Equation
    """
    t = tau / tauB
    wi = np.random.normal(0, np.sqrt(tau))
    return (
        (2 + t) / (1 + t) * xi1
        - 1 / (1 + t) * xi2
        + np.sqrt(2 * kbT * gamma) / (m * (1 + t)) * np.power(tau,1) * wi
    )
```

[6]:
```
def trajectory(N):
    x = np.zeros(N)
    for i in range(2, len(x)):
        x[i] = xi(x[i - 1], x[i - 2])
    return x
```

Now that the functions are setup one can simply generate a trajectory of length N by simply calling the the function `trajectory()`

[7]:
```
# Generate a trajectory of 10e6 points.
x = trajectory(1000000)
```

[8]:
```
plt.plot(np.arange(len(x))*tau, x)
plt.title("Intertial Brownian trajectory")
plt.ylabel("$x$ (m)")
plt.xlabel("time (s)")
plt.show()
```



1.1 Cross checking

As we are dealing with intertial Brownian motion, the later is characterize by a characteristic time $\tau_B = m/\gamma$. We will check that the simulated trajectory gives us the correct MSD properties to ensure the simulation si done properly. The MSD given by:

$$\mathrm{MSD}(\tau) = \langle (x(t) - x(t+\tau)^2 \rangle \Big|_t \,, \tag{5}$$

with Δt a lag time. The MSD, can be computed using the function defined in the cell below. For times $\tau << \tau_B$ we should have:

$$\mathrm{MSD}(\tau) = \frac{k_B T}{m} \tau^2 \,, \tag{6}$$

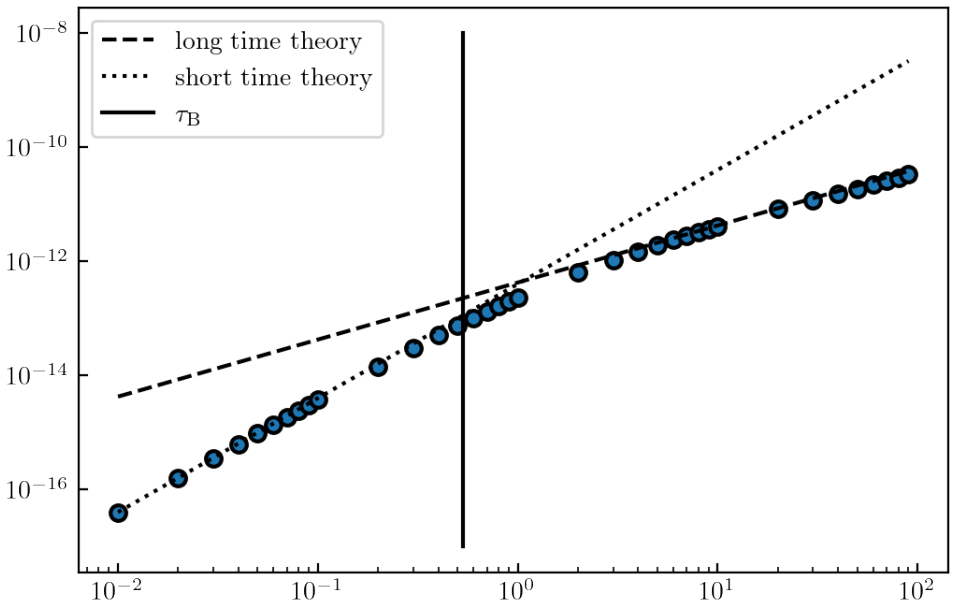
and for $\tau >> \tau_B$:

$$\mathrm{MSD}(\tau) = 2D\tau \,, \tag{7}$$

with $D = k_B T/(6\pi\eta a)$.

[9]:
```
t = np.array([*np.arange(1,10,1), *np.arange(10,100,10), *np.
 ↪arange(100,1000,100), *np.arange(1000,10000,1000)])
def msd(x,t):
    _msd = lambda x, t : np.mean((x[:-t] - x[t:])**2)
    return [_msd(x,i) for i in t]
MSD = msd(x,t)
```

[10]:
```
D = kbT/(6*np.pi*eta*a)
t_plot = t*tau
plt.loglog(t*tau,MSD, "o")
plt.plot(t*tau, (2*D*t_plot), "--", color = "k", label="long time theory")
plt.plot(t*tau, kbT/m * t_plot**2, ":", color = "k", label="short time theory")

horiz_data = [1e-8, 1e-17]
t_horiz = [tauB, tauB]
plt.plot(t_horiz, horiz_data, "k", label="$\\tau_\mathrm{B}$")
plt.legend()
plt.show()
```



Our simulation is giving the expeted results but how much time do we need to generate this trajectory of 1000000 points

[11]:
```
%timeit trajectory(1000000)
```

```
6.32 s ± 101 ms per loop (mean ± std. dev. of 7 runs, 1 loop each)
```

So we need about 6 seconds to generate this trajectory, which is in the cases of someone who want to look at fine effects and need to generate millions of trajectories is too much, in order to fasten the process i will in the following use Cython to generate the trajectory using C.

1.2 Cython acceleration

[12]:
```
%load_ext Cython
```

[13]:
```
%%cython
import cython
cimport numpy as np
import numpy as np
from libc.math cimport sqrt
ctypedef np.float64_t dtype_t

cdef int N = 1000000 # length of the simulation

cdef dtype_t tau = 0.01 # simulation time step
cdef dtype_t m = 1e-8 # particle mass
cdef dtype_t a = 1e-6 # radius of the particle
cdef dtype_t eta = 0.001 # viscosity (here water)
cdef dtype_t gamma = 6 * 3.14 * eta * a
cdef dtype_t kbT = 4e-21
cdef dtype_t tauB = m/gamma
cdef dtype_t[:] x = np.zeros(N)

@cython.boundscheck(False)
@cython.wraparound(False)
@cython.nonecheck(False)
@cython.cdivision(True)
cdef dtype_t xi_cython( dtype_t xi1, dtype_t xi2, dtype_t wi):
    cdef dtype_t t = tau / tauB
    return (
        (2 + t) / (1 + t) * xi1
        - 1 / (1 + t) * xi2
        + sqrt(2 * kbT * gamma) / (m * (1 + t)) * tau * wi
    )

@cython.boundscheck(False)
@cython.wraparound(False)
@cython.nonecheck(False)
cdef dtype_t[:] _traj(dtype_t[:] x, dtype_t[:] wi):
    cdef int i
    for i in range(2, N):

        x[i] = xi_cython(x[i-1], x[i-2], wi[i])
    return x
```

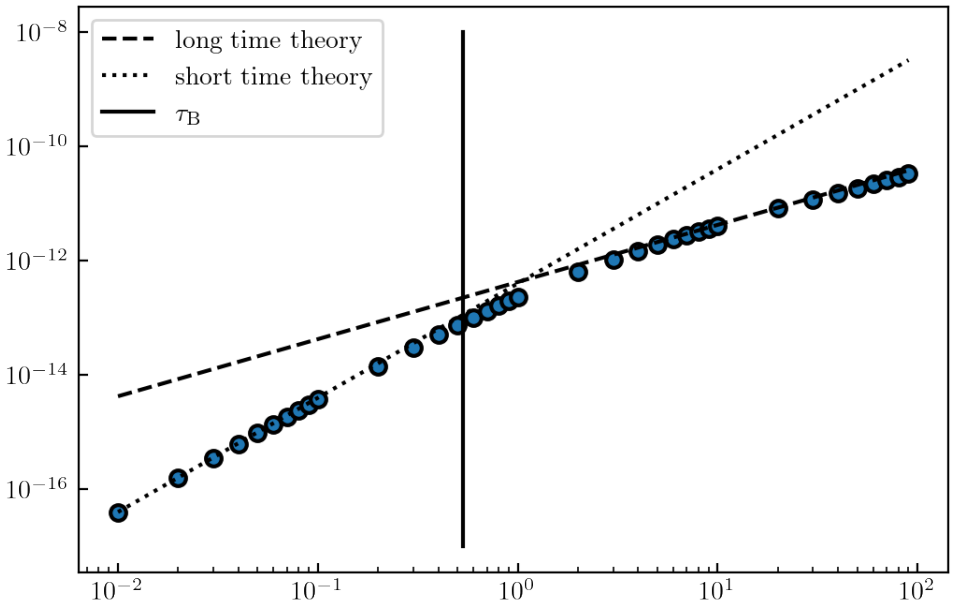
```
def trajectory_cython():

    cdef dtype_t[:] wi = np.random.normal(0, np.sqrt(tau), N).astype('float64')

    return _traj(x, wi)
```

[14]:
```
%timeit trajectory_cython()
```

```
28.9 ms ± 416 µs per loop (mean ± std. dev. of 7 runs, 10 loops each)
```

Rapid check if the Cython code properly works.

[15]:
```
x=np.asarray(trajectory_cython())
D = kbT/(6*np.pi*eta*a)
t_plot = t*tau
plt.loglog(t*tau,MSD, "o")
plt.plot(t*tau, (2*D*t_plot), "--", color = "k", label="long time theory")
plt.plot(t*tau, kbT/m * t_plot**2, ":", color = "k", label="short time theory")

horiz_data = [1e-8, 1e-17]
t_horiz = [tauB, tauB]
plt.plot(t_horiz, horiz_data, "k", label="$\\tau_\mathrm{B}$")
plt.legend()
plt.show()
```



1.2.1 Conclusion

We finally only need $\simeq$ 6 ms to generate the trajectory instead of $\simeq 6s$ which is a $\simeq 1000\times$ improvement speed. The simulation si here bound to the time needed to generate the array of random numbers which is still done using numpy function. After further checking, Numpy random generation si as optimize as one could do so there is no benefit on cythonizing the random generation. For the sake of completness one could fine a Cython version to generate random numbers. Found thanks to Senderle: https://stackoverflow.com/questions/42767816/what-is-the-most-efficient-and-portable-way-to-generate-gaussian-random-numbers

[16]:
```
%%cython
from libc.stdlib cimport rand, RAND_MAX
from libc.math cimport log, sqrt
import numpy as np
import cython

cdef double random_uniform():
    cdef double r = rand()
    return r / RAND_MAX

cdef double random_gaussian():
    cdef double x1, x2, w

    w = 2.0
    while (w >= 1.0):
        x1 = 2.0 * random_uniform() - 1.0
        x2 = 2.0 * random_uniform() - 1.0
        w = x1 * x1 + x2 * x2

    w = ((-2.0 * log(w)) / w) ** 0.5
    return x1 * w

@cython.boundscheck(False)
cdef void assign_random_gaussian_pair(double[:] out, int assign_ix):
    cdef double x1, x2, w

    w = 2.0
    while (w >= 1.0):
        x1 = 2.0 * random_uniform() - 1.0
        x2 = 2.0 * random_uniform() - 1.0
        w = x1 * x1 + x2 * x2

    w = sqrt((-2.0 * log(w)) / w)
    out[assign_ix] = x1 * w
    out[assign_ix + 1] = x2 * w
```

```
@cython.boundscheck(False)
def my_uniform(int n):
    cdef int i
    cdef double[:] result = np.zeros(n, dtype='f8', order='C')
    for i in range(n):
        result[i] = random_uniform()
    return result

@cython.boundscheck(False)
def my_gaussian(int n):
    cdef int i
    cdef double[:] result = np.zeros(n, dtype='f8', order='C')
    for i in range(n):
        result[i] = random_gaussian()
    return result

@cython.boundscheck(False)
def my_gaussian_fast(int n):
    cdef int i
    cdef double[:] result = np.zeros(n, dtype='f8', order='C')
    for i in range(n // 2):  # Int division ensures trailing index if n is odd.
        assign_random_gaussian_pair(result, i * 2)
    if n % 2 == 1:
        result[n - 1] = random_gaussian()

    return result
```

[17]:
```
%timeit my_gaussian_fast(1000000)
```

```
28.7 ms ± 963 µs per loop (mean ± std. dev. of 7 runs, 10 loops each)
```

[18]:
```
%timeit np.random.normal(0,1,1000000)
```

```
24 ms ± 768 µs per loop (mean ± std. dev. of 7 runs, 10 loops each)
```

One can thus see, that even a pure C implementation can be slower than the Numpy one, thanks to impressive optimization.

[ ]:
```

```

1 Fitting pipeline using pylorenzmie

In order to fit an hologram, I used the pylorenzmie model which provides a set of python classes in order to analyse holographic microscopy data.

Pylorenzmie can be download on the David Grier's github repository: https://github.com/davidgrier/pylorenzmie.

What I actually get from the experiments are mp4 movies, in order to analyze them easily, I constructed a wrapper around the pylorenzmie module which can be found on my repository: https://github.com/eXpensia/wraplorenzmie.

This wrapper permits to do the following pipeline:

- Directly load the movies
- Compute the back ground.
- Use the first image in order to get the pre guesses
- Fit the 10 000 first images to determine precisely the radius and index of a particle.
- Use the later information in order to fit the whole movie (and save the data in the same time)

One that done, the trajectory be analyzed separately.

[1]:
```
# We first start by import the important modules

import wraplorenzmie.utilities.utilities as utilities
import wraplorenzmie.fits.fit as fit
import imageio
# For Plotting.
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
#sns.set(style='white', font_scale=2)
%matplotlib inline
import matplotlib as mpl


mpl.rcParams["figure.dpi"] = 200
from matplotlib import rc
rc('font', family='serif')
rc('text', usetex=True)
rc('xtick', labelsize='x-small')
rc('ytick', labelsize='x-small')

def cm2inch(value):
    return value/2.54
```

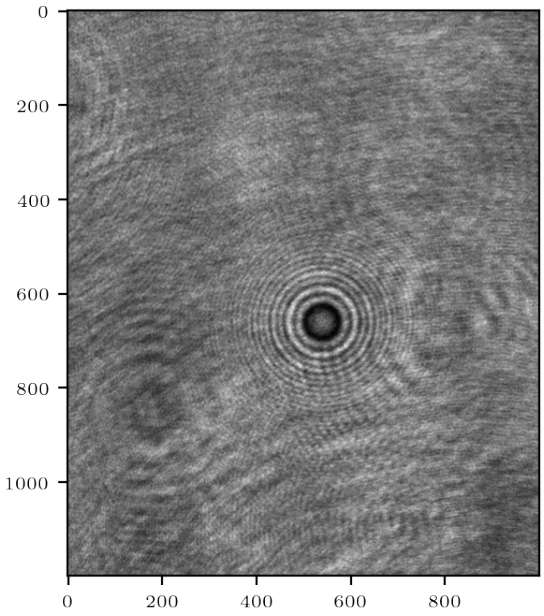
```
No module named 'pylorenzmie.fitting.cython.cminimizers'
```

[2]:
```
#We load the movie
vid = utilities.
 ↪video_reader("Basler_acA1920-155um__22392621__20200527_162231224.mp4")
```

[3]:
```
# A function that permits to compute de radial profile of an image this␣
 ↪will later be used in order to see if the fits are done correctly
def radial_profile(data, center=None):
    if center==None:
        center = np.array(np.shape(data)) / 2

    y, x = np.indices((data.shape))
    r = np.sqrt((x - center[0])**2 + (y - center[1])**2)
    r = r.astype(int)

    tbin = np.bincount(r.ravel(), data.ravel())
    nr = np.bincount(r.ravel())
    radialprofile = tbin / nr

    T = data.ravel()
    V = r.ravel()

    err = [np.std(T[V == u]) for u in np.unique(V)]

    return radialprofile, err
```

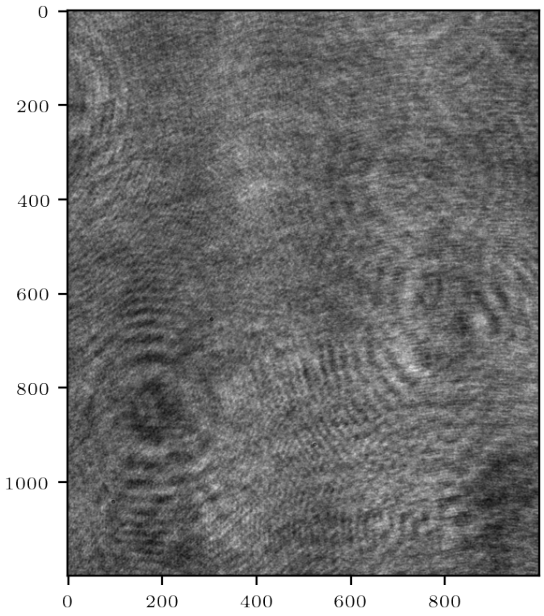
[4]:
```
# We take a look at the first image of the movie
image = vid.get_image(1)
plt.imshow(image,cmap="gray")
```

[4]:
```
<matplotlib.image.AxesImage at 0x1a70b337be0>
```



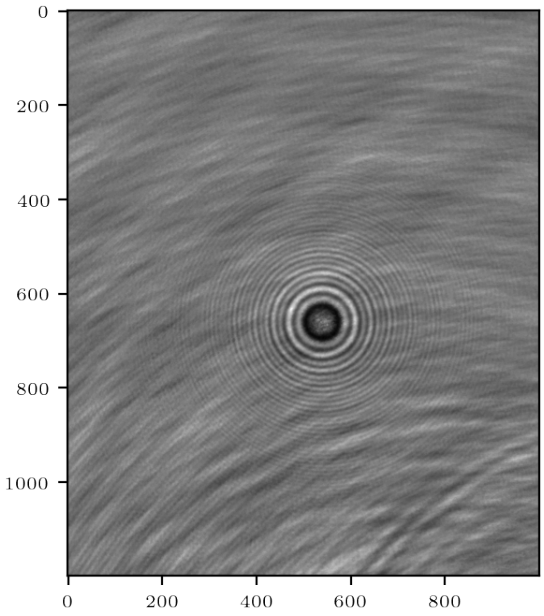
[5]:
```
# set the background image (it can also be computed using vid.
 ↪get_background method)
vid.number = 125000
vid.background = np.array(imageio.imread("background.tiff"))
#vid.background = vid.get_background(n=50) # n is the number of image to␣
 ↪use to compute the background
plt.imshow(vid.background,cmap="gray")
```

[5]: <matplotlib.image.AxesImage at 0x1a70bc56490>

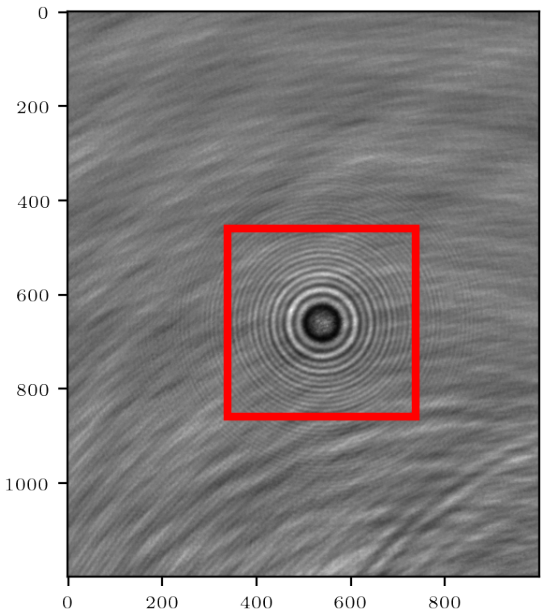


[6]:
```
imageio.imwrite("background.tiff",vid.background)# We save the background␣
↪for possible later use.
```

[7]:
```
# the normalized image, we can see that their is some movement in the␣
↪background.
# This could be avoided by computed the background as a function of the␣
↪time, if the particle diffuses enough.
normed_image = utilities.normalize(image,vid.background)
plt.imshow(normed_image,cmap="gray")
normed_image = normed_image
```



[8]:
```
# We found the possition of the particle
feature = utilities.center_find(image)[0]
utilities.plot_bounding(normed_image,feature)
```

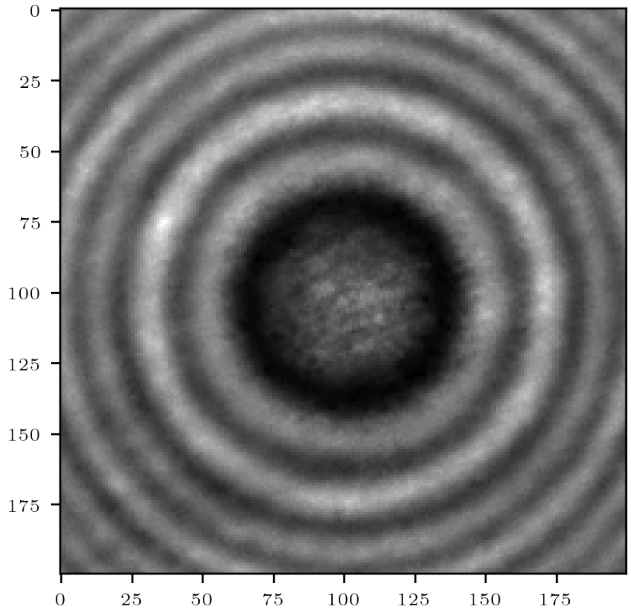


1.1 Fitting the first image

We fit the first image in order to get the preguess. We first start by croping the hologram.

[9]:
```
xc, yc, w, h = feature[0]
x_center = xc
y_center = yc
h=200
im_c = fit.crop(image, int(xc), int(yc), int(h))
bk_c = fit.crop(vid.background, int(xc), int(yc), int(h))
cropped = utilities.normalize(im_c,bk_c, dark_count = np.min(im_c))
cropped = cropped / np.mean(cropped)
plt.imshow(cropped,cmap = "gray")
```

[9]:
```
<matplotlib.image.AxesImage at 0x1a71d7e6d00>
```



[10]:
```
# We setup the fitting method.
fitter = fit.fitting(cropped,0.532,0.0513)
fitter.make_guess(1.50,1.59,12,alpha = 1,fit_r=True,␣
 ↪fit_n=True,fit_alpha=True)
```

[11]:
```
# We do the actual fit.
result = fitter.fit_single(cropped, method = "lm")
```

[12]:
```
zo = result.result["x"][2]*0.0513
print(result.result["x"][2]*0.0513)
print(result.redchi)
print(result.result["x"])
```

```
11.427616273713154
7.2459765196825305
[101.23514587 103.00299474 222.76055114   1.5310255    1.58239091
   1.00198476]
```

We can plot the result to see if the fit worked properly, and, for a more quantitative comparison we can compute the radial intensity profile of both hologram and compare them.

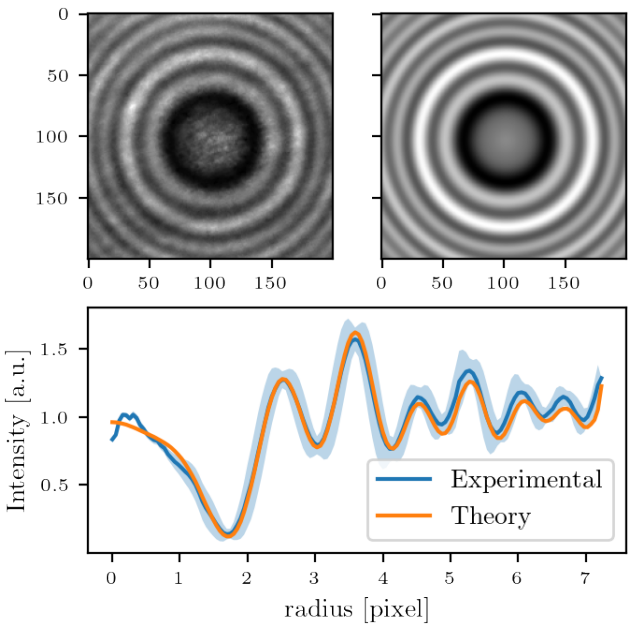
[13]:
```
center = np.array(np.shape(fitter.image))
```

[14]:
```
radial_exp, err = radial_profile(fitter.image)
theo_exp, err = radial_profile(fitter.fitter.model.hologram().
 ↪reshape(fitter.shape))
# computing first the holgram using the fit resutlt
```

[15]:
```
fit_data = {}
radius_radial = np.arange(len(radial_exp)) * 0.0513
plt.figure(figsize = (15,15))
fig = plt.figure(figsize=(cm2inch(8.6),1.65*cm2inch(8.6)))
fig.subplots_adjust(left=0.14, bottom=.12, right=.99, top=.98)

plt.subplot(2,2,1)
plt.imshow(fitter.image, cmap = "gray")
#plt.title('subplot(2,2,1)')

fit_data["exp_image"] = fitter.image

plt.subplot(2,2,2)
plt.imshow(fitter.fitter.model.hologram().reshape(fitter.shape), cmap =␣
 ↪"gray")
frame1 = plt.gca()
frame1.axes.yaxis.set_ticklabels([])

fit_data["th_image"] = fitter.fitter.model.hologram().reshape(fitter.
 ↪shape)


#plt.title('subplot(2,2,2)')

plt.subplot(2,2,(3,4))
```

```
plt.plot(radius_radial, radial_exp, label="Experimental")
plt.fill_between(radius_radial, radial_exp - err, radial_exp + err, alpha␣
↪= 0.3)
plt.plot(radius_radial, theo_exp, label="Theory")
plt.legend()
plt.xlabel("radius [pixel]")
plt.ylabel("Intensity [a.u.]")

fit_data["I_r_exp"] = radial_exp
fit_data["I_errr_exp"] = err

fit_data["theo_exp"] = theo_exp
fit_data["I_radius"] = radius_radial

fig.set_size_inches(cm2inch(8.6), cm2inch(1.6 * 8.6/1.618))
plt.savefig("fit_fig.pdf")
```

```
<Figure size 3000x3000 with 0 Axes>
```



[16]:
```
fitter.fit_video(vid =␣
↪vid,savefile="find_nrfit_result_dur_27052020_n_r_fix_0p0513_wav532.
↪dat",xc = x ,yc= y, h = 200, n_end=10000,method = "lm")
```

```
100% 9999/9999 [12:39<00:00, 13.17it/s]
```

[17]:
```
# Since the measurement or not saved into the ram we need to load it
n_r = np.fromfile('find_nr_exame.dat', dtype=np.float64)
n_r = n_r.reshape(len(n_r)//10,10)
r = n_r[:,3]
n = n_r[:,4]
```

1.2 Fitting the n, r distributiton using a KDE estimator

To find the most probable couple of r/n we use a kde estimator using `seaborn`

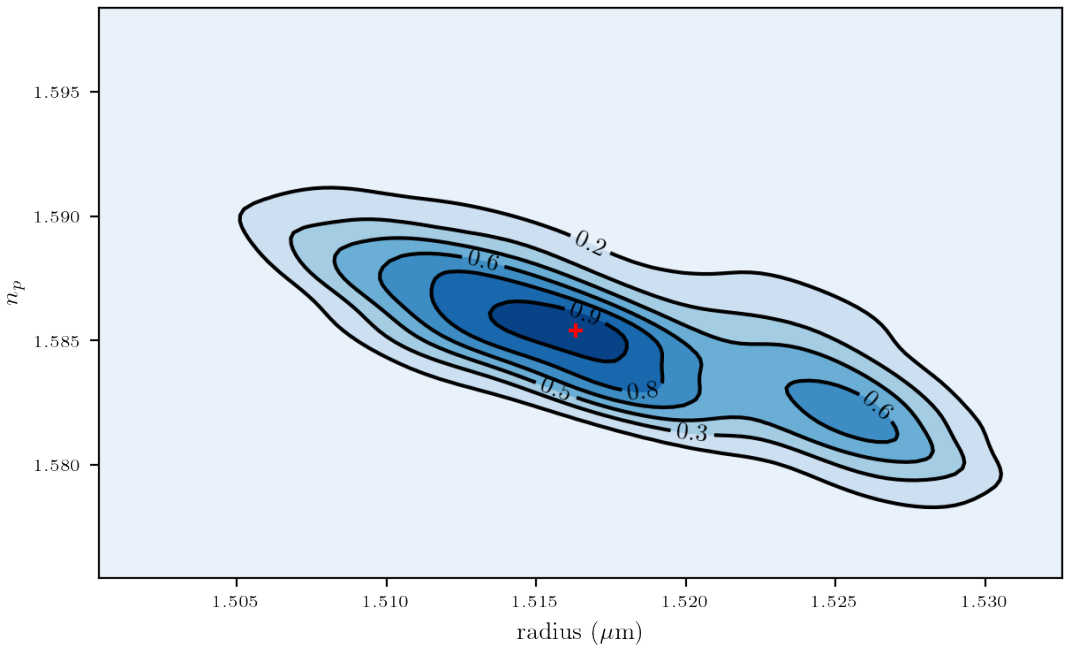
[18]:
```
import numpy as np
import scipy.stats as st
import matplotlib.ticker as ticker

data = np.random.multivariate_normal((0, 0), [[0.8, 0.05], [0.05, 0.7]],␣
↪100)
x = r[(r>1.5) & (r<1.555)]
y = n[(r>1.5) & (r<1.555)]
xmin, xmax = np.min(x), np.max(x)
ymin, ymax = np.min(y), np.max(y)

# Peform the kernel density estimate
xx, yy = np.mgrid[xmin:xmax:100j, ymin:ymax:100j]
positions = np.vstack([xx.ravel(), yy.ravel()])
values = np.vstack([x, y])
kernel = st.gaussian_kde(values)
f = np.reshape(kernel(positions).T, xx.shape)
f = f/np.max(f)
```

[19]:
```
np.round(np.max(f))
```

[19]:
```
1.0
```

[20]:
```
fig = plt.figure()
fig.subplots_adjust(left=0.16, bottom=.20, right=.99, top=.99)
ax = fig.gca()
#ax.set_xlim(1.505, 1.53)
#ax.set_ylim(1.575, 1.6)
# Contourf plot
cfset = ax.contourf(xx, yy, f, cmap='Blues')
## Or kernel density estimate plot instead of the contourf plot
#ax.imshow(np.rot90(f), cmap='Blues', extent=[xmin, xmax, ymin, ymax])
# Contour plot
cset = ax.contour(xx, yy, f, colors='k', levels=6)
```

```
# Label plot
ax.xaxis.set_major_formatter(ticker.FormatStrFormatter('%1.3f'))
ax.yaxis.set_major_formatter(ticker.FormatStrFormatter('%1.3f'))
ax.clabel(cset, inline=1, fontsize=10, fmt="%1.1f")
plt.scatter(xx[np.where(f == 1)],yy[np.where(f == 1)], color = "red",␣
 ↪marker="+")
ax.set_xlabel("radius ($\mathrm{\mu m}$)")
ax.set_ylabel("$n_p$")
#plt.title("KDE r n")
fig.set_size_inches(cm2inch(16), cm2inch(9.9))

plt.tight_layout()
fig.savefig('KDErn.pdf')
#pl.show()
```



[21]:
```
print(" n determined with : mu={0}, sigma={1}".format(np.mean(yy[np.
 ↪where(f > 0.1)]), np.std(yy[np.where(f > 0.1)])))
print(" r determined with : mu={0}, sigma={1}".format(np.mean(xx[np.
 ↪where(f > 0.1)]), np.std(xx[np.where(f > 0.1)])))
```

```
n determined with : mu=1.5851200393768743, sigma=0.003267685282504072
r determined with : mu=1.5181266656310368, sigma=0.00682411690457934
```

[22]:
```
(mu_n, mu_r) = np.mean(yy[np.where(f > 0.1)]) ,  np.mean(xx[np.where(f >␣
↪0.1)])
```

1.3 Fitting the whole movie

Now that the measurement of n and r is one we can move on the measurement of the whole trajectory by simply using `fitter.fit_video`. For demonstration purposes, I only fit here at $\simeq$ 22 image per seconds, if can goes up to at least 60 with recent GPU.

[23]:
```
del fitter
fitter = fit.fitting(cropped,0.532,0.0513)
fitter.make_guess(mu_r,mu_n,zo,alpha = 1,fit_r=False,␣
↪fit_n=False,fit_alpha=False)
#result = fitter.fit_single(cropped, method = "lm")
fitter.fit_video(vid =␣
↪vid,savefile="fit_result_1p5kPa_18122019_n_r_fix_0p0883_wav_532_ex1.
↪dat",xc = xc ,yc= yc, h = 200, n_end=10000,method = "lm")
```

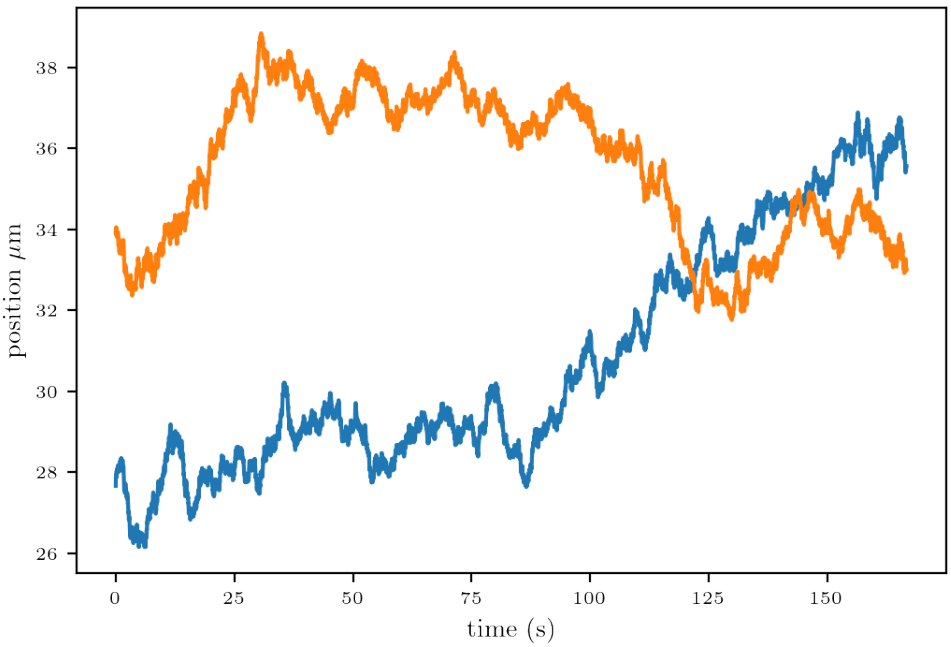
```
100% 9999/9999 [07:24<00:00, 22.47it/s]
```

[24]:
```
import numpy as np
data = np.fromfile('fit_result_1p5kPa_18122019_n_r_fix_0p0883_wav_532_ex1.
↪dat', dtype=np.float64)
data = data.reshape(len(data)//10,10)
x = data[:,0]*0.0513
y = data[:,1]*0.0513
z = data[:,2]*0.0513
```

1.4 Plot the trajectory

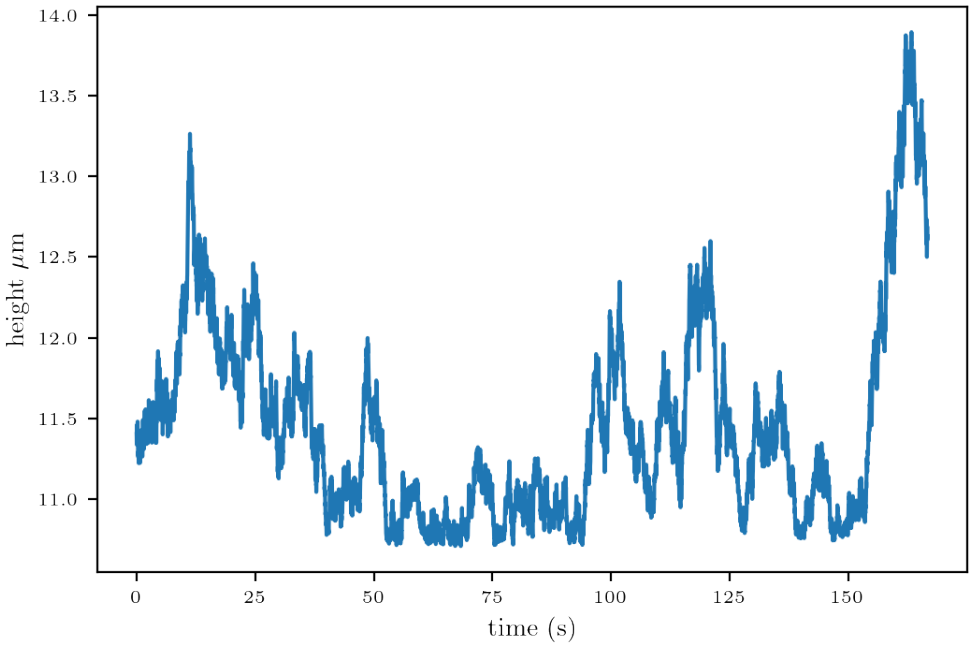
[25]:
```
plt.plot(np.arange(len(z))/60, x)
plt.plot(np.arange(len(z))/60, y)
plt.ylabel("position $\mathrm{\mu m}$")
plt.xlabel("time (s)")
```

[25]:
```
Text(0.5, 0, 'time (s)')
```



[26]:
```
plt.plot(np.arange(len(z))/60, z)
plt.ylabel("height $\mathrm{\mu m}$")
plt.xlabel("time (s)")
```

[26]: `Text(0.5, 0, 'time (s)')`



References

[1] R. Baraniuk, D. Donoho, and M. Gavish, "The science of deep learning", Proceedings of the National Academy of Sciences **117**, Publisher: National Academy of Sciences Section: Introduction, 30029–30032 (2020).

[2] B. Lemieux, A. Aharoni, and M. Schena, "Overview of DNA chip technology", Molecular Breeding **4**, 277–289 (1998).

[3] Á. Ríos, M. Zougagh, and M. Avila, "Miniaturization through lab-on-a-chip: utopia or reality for routine laboratories? a review", Analytica Chimica Acta **740**, 1–11 (2012).

[4] A. Einstein, "Über die von der molekularkinetischen Theorie der Wärme geforderte Bewegung von in ruhenden Flüssigkeiten suspendierten Teilchen", Annalen der Physik **vol. 4, t. 17** (1905).

[5] J. Perrin, *Les Atomes*, Google-Books-ID: A0ltBQAAQBAJ (CNRS Editions, Nov. 14, 2014), 199 pp.

[6] B. U. Felderhof, "Effect of the wall on the velocity autocorrelation function and long-time tail of brownian motion †", The Journal of Physical Chemistry B **109**, 21406–21412 (2005).

[7] M. V. Chubynsky and G. W. Slater, "Diffusing diffusivity: a model for anomalous, yet brownian, diffusion", Physical Review Letters **113**, 098302 (2014).

[8] A. V. Chechkin, F. Seno, R. Metzler, and I. M. Sokolov, "Brownian yet non-gaussian diffusion: from superstatistics to subordination of diffusing diffusivities", Physical Review X **7**, 021002 (2017).

[9] C. I. Bouzigues, P. Tabeling, and L. Bocquet, "Nanofluidics in the debye layer at hydrophilic and hydrophobic surfaces", Physical Review Letters **101**, Publisher: American Physical Society, 114503 (2008).

[10] L. Joly, C. Ybert, and L. Bocquet, "Probing the nanohydrodynamics at liquid-solid interfaces using thermal motion", Physical Review Letters **96**, Publisher: American Physical Society, 046101 (2006).

[11] J. Mo, A. Simha, and M. G. Raizen, "Brownian motion as a new probe of wettability", The Journal of Chemical Physics **146**, Publisher: American Institute of Physics, 134707 (2017).

[12] E. R. Dufresne, T. M. Squires, M. P. Brenner, and D. G. Grier, "Hydrodynamic coupling of two brownian spheres to a planar surface", Physical Review Letters **85**, Publisher: American Physical Society, 3317–3320 (2000).

[13] L. P. Faucheux and A. J. Libchaber, "Confined brownian motion", Physical Review E **49**, 5158–5163 (1994).

[14] E. R. Dufresne, D. Altman, and D. G. Grier, "Brownian dynamics of a sphere between parallel walls", EPL (Europhysics Letters) **53**, Publisher: IOP Publishing, 264 (2001).

[15] H. B. Eral, J. M. Oh, D. v. d. Ende, F. Mugele, and M. H. G. Duits, "Anisotropic and hindered diffusion of colloidal particles in a closed cylinder", Langmuir **26**, Publisher: American Chemical Society, 16722–16729 (2010).

[16] P. Sharma, S. Ghosh, and S. Bhattacharya, "A high-precision study of hindered diffusion near a wall", Applied Physics Letters **97**, Publisher: American Institute of Physics, 104101 (2010).

[17] J. Mo, A. Simha, and M. G. Raizen, "Broadband boundary effects on brownian motion", Physical Review E **92**, 062106 (2015).

[18] M. Matse, M. V. Chubynsky, and J. Bechhoefer, "Test of the diffusing-diffusivity mechanism using near-wall colloidal dynamics", Physical Review E **96**, 042604 (2017).

[19] D. C. Prieve, "Measurement of colloidal forces with TIRM", Advances in Colloid and Interface Science **82**, 93–125 (1999).

[20] A. Banerjee and K. Kihm, "Experimental verification of near-wall hindered diffusion for the brownian motion of nanoparticles using evanescent wave microscopy", Physical review. E, Statistical, nonlinear, and soft matter physics **72**, 042101 (2005).

[21] S. Sainis, V. Germain, and E. Dufresne, "Statistics of particle trajectories at short time intervals reveal fN-scale colloidal forces", Physical review letters **99**, 018303 (2007).

[22] G. Volpe, L. Helden, T. Brettschneider, J. Wehr, and C. Bechinger, "Influence of noise on force measurements", Physical Review Letters **104**, 170602 (2010).

[23] M. Li, O. Sentissi, S. Azzini, G. Schnoering, A. Canaguier-Durand, and C. Genet, "Subfemtonewton force fields measured with ergodic brownian ensembles", Physical Review A **100**, 063816 (2019).

[24] S. K. Sainis, V. Germain, and E. R. Dufresne, "Statistics of particle trajectories at short time intervals reveal fN-scale colloidal forces", Physical Review Letters **99**, 018303 (2007).

[25] B. Robert, "XXVII. a brief account of microscopical observations made in the months of june, july and august 1827, on the particles contained in the pollen of plants; and on the general existence of active molecules in organic and inorganic bodies", The Philosophical Magazine **4**, Taylor & Francis, 161–173 (1828).

[26] S. Peter, "Brownian motion", **Brownian motion**.

[27] J. Perrin, "Mouvement brownien et molécules", J. Phys. Theor. Appl. **9**, 5–39 (1910).

[28] A. Genthon, "The concept of velocity in the history of brownian motion", The European Physical Journal H **45**, 49–105 (2020).

[29] P. Langevin, "Sur la théorie du mouvement brownien", C. R. Acad. Sci. (Paris) 146, 530–533 **65**, 1079–1081 (1908).

[30] R. Durrett, *Probability: theory and examples*, 5th ed., Cambridge Series in Statistical and Probabilistic Mathematics (Cambridge University Press, Cambridge, 2019).

[31] D. Freedman and P. Diaconis, "On the histogram as a density estimator:l 2 theory", Zeitschrift für Wahrscheinlichkeitstheorie und Verwandte Gebiete **57**, 453–476 (1981).

[32] E. Ampomah, E. Mensah, and A. Gilbert, "Qualitative assessment of compiled, interpreted and hybrid programming languages", Communications on Applied Electronics **7**, 8–13 (2017).

[33] C. Fabry and A. Pérot, "Théorie et application d'une nouvelle méthode de spectroscopie interferentielle.", Ann. Chim. Phys., 7 (1899).

[34] A. Perot and C. Fabry, "On the application of interference phenomena to the solution of various problems of spectroscopy and metrology", The Astrophysical Journal **9**, 87 (1899).

[35] A. A. Michelson and E. W. Morley, "On the relative motion of the earth and the luminiferous ether", American Journal of Science **s3-34**, Publisher: American Journal of Science Section: Extraterrestrial geology, 333–345 (1887).

[36] LIGO Scientific Collaboration and Virgo Collaboration et al., "GW151226: observation of gravitational waves from a 22-solar-mass binary black hole coalescence", Physical Review Letters **116**, in collab. with B. P. Abbott, Publisher: American Physical Society, 241103 (2016).

[37] A. S. Curtis, "The mechanism of adhesion of cells to glass. a study by interference reflection microscopy", The Journal of Cell Biology **20**, 199–215 (1964).

[38] T. J. Filler and E. T. Peuker, "Reflection contrast microscopy (RCM): a forgotten technique?", The Journal of Pathology **190**, 635–638 (2000).

[39] P. A. Siver and J. Hinsch, "The use of interference reflection contrast in the examination of diatom valves", Journal of Phycology **36**, 616–620 (2000).

[40] I. Weber, "[2] reflection interference contrast microscopy", in *Methods in enzymology*, Vol. 361, Biophotonics, Part B (Academic Press, Jan. 1, 2003), pp. 34–47.

[41] L. Limozin and K. Sengupta, "Quantitative reflection interference contrast microscopy (RICM) in soft matter and cell adhesion", Chemphyschem: A European Journal of Chemical Physics and Physical Chemistry **10**, 2752–2768 (2009).

[42] F. Nadal, A. Dazzi, F. Argoul, and B. Pouligny, "Probing the confined dynamics of a spherical colloid close to a surface by combined optical trapping and reflection interference contrast microscopy", Applied Physics Letters **79**, 3887–3889 (2002).

[43] J. Raedler and E. Sackmann, "On the measurement of weak repulsive and frictional colloidal forces by reflection interference contrast microscopy", Langmuir **8**, 848–853 (1992).

[44] H. S. Davies, D. Débarre, N. El Amri, C. Verdier, R. P. Richter, and L. Bureau, "Elastohydrodynamic lift at a soft wall", Physical Review Letters **120**, 198001 (2018).

[45] C. F. Bohren and D. R. Huffman, *Absorption and scattering of light by small particles* (Wiley, Apr. 1998).

[46] H. J. W. Strutt, "LVIII. on the scattering of light by small particles", The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science **41**, Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/14786447108640507, 447–454 (1871).

[47] L. Lorenz, *Lysbevægelsen i og uden for en af plane Lysbølger belyst Kugle*, Google-Books-ID: hnE7QwAACAAJ (1890), 62 pp.

[48] G. Mie, "Beiträge zur optik trüber medien, speziell kolloidaler metallösungen", Annalen der Physik **330**, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/andp.19083300302 377–445 (1908).

[49] B. Ovryn and S. H. Izen, "Imaging of transparent spheres through a planar interface using a high-numerical-aperture optical microscope", JOSA A **17**, Publisher: Optical Society of America, 1202–1213 (2000).

[50] S.-H. Lee, Y. Roichman, G.-R. Yi, S.-H. Kim, S.-M. Yang, A. v. Blaaderen, P. v. Oostrum, and D. G. Grier, "Characterizing and tracking single colloidal particles with video holographic microscopy", Optics Express **15**, Publisher: Optical Society of America, 18275–18282 (2007).

[51] J. Katz and J. Sheng, "Applications of holography in fluid mechanics and particle dynamics", Annual Review of Fluid Mechanics **42**, _eprint: https://doi.org/10.1146/annurev-fluid-121108-145508, 531–555 (2010).

[52] P. Gregory, *Bayesian logical data analysis for the physical sciences: a comparative approach with mathematica® support*, Google-Books-ID: idkLAQAAQBAJ (Cambridge University Press, Apr. 14, 2005), 496 pp.

[53] T. G. Dimiduk and V. N. Manoharan, "Bayesian approach to analyzing holograms of colloidal particles", Optics Express **24**, Publisher: Optical Society of America, 24045–24060 (2016).

[54] W. J. Lentz, "Generating bessel functions in mie scattering calculations using continued fractions", Applied Optics **15**, Publisher: Optical Society of America, 668–671 (1976).

[55] J. Fung and V. N. Manoharan, "Holographic measurements of anisotropic three-dimensional diffusion of colloidal clusters", Physical Review E **88**, 020302 (2013).

[56] A. Wang, T. G. Dimiduk, J. Fung, S. Razavi, I. Kretzschmar, K. Chaudhary, and V. N. Manoharan, "Using the discrete dipole approximation and holographic microscopy to measure rotational dynamics of non-spherical colloidal particles", Journal of Quantitative Spectroscopy and Radiative Transfer **146**, 499–509 (2014).

[57] R. W. Perry, G. Meng, T. G. Dimiduk, J. Fung, and V. N. Manoharan, "Real-space studies of the structure and dynamics of self-assembled colloidal clusters", Faraday Discussions **159**, Publisher: The Royal Society of Chemistry, 211–234 (2013).

[58] T. G. Dimiduk, R. W. Perry, J. Fung, and V. N. Manoharan, "Random-subset fitting of digital holograms for fast three-dimensional particle tracking", Applied Optics **53**, G177 (2014).

[59] A. Yevick, M. Hannel, and D. G. Grier, "Machine-learning approach to holographic particle characterization", Optics Express **22**, 26884 (2014).

[60] M. D. Hannel, A. Abdulali, M. O'Brien, and D. G. Grier, "Machine-learning techniques for fast and accurate feature localization in holograms of colloidal particles", Optics Express **26**, Publisher: Optical Society of America, 15221–15231 (2018).

[61] L. E. Altman and D. G. Grier, "CATCH: characterizing and tracking colloids holographically using deep neural networks", The Journal of Physical Chemistry B **124**, Publisher: American Chemical Society, 1602–1610 (2020).

[62] L. Wilson and R. Zhang, "3d localization of weak scatterers in digital holographic microscopy using rayleigh-sommerfeld back-propagation", Optics Express **20**, 16735 (2012).

[63] J. W. Goodman, *Introduction to fourier optics*, Google-Books-ID: ow5xs_Rtt9AC (Roberts and Company Publishers, 2005), 520 pp.

[64] F. C. Cheong, B. J. Krishnatreya, and D. G. Grier, "Strategies for three-dimensional particle tracking with holographic video microscopy", Optics Express **18**, Publisher: Optical Society of America, 13563–13573 (2010).

[65] G. C. Sherman, "Application of the convolution theorem to rayleigh's integral formulas", JOSA **57**, Publisher: Optical Society of America, 546–547 (1967).

[66] U. Schnars and W. P. Jüptner, "Digital recording and reconstruction of holograms in hologram interferometry and shearography", Applied Optics **33**, 4373–4377 (1994).

[67] T. M. Kreis, "Frequency analysis of digital holography with reconstruction by convolution", Optical Engineering **41**, Publisher: International Society for Optics and Photonics, 1829–1839 (2002).

[68] X. Zhang, J. Qiu, J. Qiu, J. Qiu, X. Li, X. Li, J. Zhao, J. Zhao, L. Liu, and L. Liu, "Complex refractive indices measurements of polymers in visible and near-infrared bands", Applied Optics **59**, Publisher: Optical Society of America, 2337–2344 (2020).

[69] J. J. Moré, "The levenberg-marquardt algorithm: implementation and theory", in Numerical analysis, edited by G. A. Watson, Lecture Notes in Mathematics (1978), pp. 105–116.

[70] J. N. Israelachvili, *Intermolecular and surface forces*, Google-Books-ID: MVbWBhubrgIC (Academic Press, May 29, 2015), 706 pp.

[71] G. M. Bell, S. Levine, and L. N. McCartney, "Approximate methods of determining the double-layer free energy of interaction between two charged colloidal spheres", Journal of Colloid and Interface Science **33**, 335–359 (1970).

[72] S. H. Behrens and D. G. Grier, "The charge of glass and silica surfaces", The Journal of Chemical Physics **115**, 6716–6721 (2001).

[73] J. Gregory, "Approximate expressions for retarded van der waals interaction", Journal of Colloid and Interface Science **83**, 138–145 (1981).

[74] H. Brenner, "The slow motion of a sphere through a viscous fluid towards a plane surface", Chemical Engineering Science **16**, 242–251 (1961).

[75] H. Faxen, "Fredholm integral equations of hydrodynamics of liquids i", Ark. Mat., Astron. Fys **18**, 29–32 (1924).

[76] A. J. Goldman, R. G. Cox, and H. Brenner, "Slow viscous motion of a sphere parallel to a plane wall—i motion through a quiescent fluid", Chemical Engineering Science **22**, 637–651 (1967).

[77] N. Ikeda and S. Watanabe, *Stochastic differential equations and diffusion processes*, Google-Books-ID: QZbOBQAAQBAJ (Elsevier, June 28, 2014), 572 pp.

[78] G. Volpe and J. Wehr, "Effective drifts in dynamical systems with multiplicative noise: a review of recent progress", Reports on Progress in Physics **79**, 053901 (2016).

[79] M. Le Bellac, F. Mortessagne, and G. G. Batrouni, *Equilibrium and non-equilibrium statistical thermodynamics* (Cambridge University Press, Cambridge, 2004).

[80] W. Wang, J. S. Guasto, and P. Huang, "Measurement bias in evanescent wave nano-velocimetry due to tracer size variations", 10.1007/S00348-011-1188-X (2011).

[81] M. R. Oberholzer, N. J. Wagner, and A. M. Lenhoff, "Grand canonical brownian dynamics simulation of colloidal adsorption", The Journal of Chemical Physics **107**, Publisher: American Institute of Physics, 9157–9167 (1997).

[82] M. Matse, "State-dependent diffusion of brownian particles near a boundary wall", 84.

[83] H. Risken, *Fokker-planck equation: methods of solution and applications.* OCLC: 968506197 (Springer, Place of publication not identified, 2012).

[84] S. Hapca, J. W. Crawford, and I. M. Young, "Anomalous diffusion of heterogeneous populations characterized by normal diffusion at the individual level", Journal of The Royal Society Interface **6**, 111–122 (2009).

[85] R. Friedrich, J. Peinke, M. Sahimi, and M. Reza Rahimi Tabar, "Approaching complexity by stochastic methods: from biological systems to turbulence", Physics Reports **506**, 87–162 (2011).

[86] A. Frishman and P. Ronceray, "Learning force fields from stochastic trajectories", Physical Review X **10**, 021009 (2020).

[87] M. E. Beheiry, M. Dahan, and J.-B. Masson, "InferenceMAP: mapping of single-molecule dynamics with bayesian inference", Nature Methods **12**, Bandiera_abtest: a Cg_type: Nature Research Journals Number: 7 Primary_atype: Correspondence Publisher: Nature Publishing Group Subject_term: Computational biophysics;Image processing;Single-molecule biophysics;Software Subject_term_id: computational-biophysics;image-processing;single-molecule-biophysics;software, 594–595 (2015).

[88] N. Hoze, D. Nair, E. Hosy, C. Sieben, S. Manley, A. Herrmann, J.-B. Sibarita, D. Choquet, and D. Holcman, "Heterogeneity of AMPA receptor trafficking and molecular interactions revealed by superresolution analysis of live cell imaging", Proceedings of the National Academy of Sciences **109**, Publisher: National Academy of Sciences Section: Biological Sciences, 17052–17057 (2012).

[89] B. Saintyves, T. Jules, T. Salez, and L. Mahadevan, "Self-sustained lift and low friction via soft lubrication", Proceedings of the National Academy of Sciences **113**, 5847–5849 (2016).

[90] Y.-H. Dai, "Convergence properties of the BFGS algoritm", SIAM Journal on Optimization **13**, Publisher: Society for Industrial and Applied Mathematics, 693–701 (2002).

[91] R. Mannella and P. V. E. McClintock, "Comment on "influence of noise on force measurements"", Physical Review Letters **107**, 078901 (2011).

[92] R. Mannella and P. V. E. McCLINTOCK, "Itô versus stratonovich: 30 years later", Fluctuation and Noise Letters **11**, Publisher: World Scientific Publishing Co., 1240010 (2012).

[93] J. M. Sancho, "Brownian colloidal particles: ito, stratonovich, or a different stochastic interpretation", Physical Review E **84**, Publisher: American Physical Society, 062102 (2011).

[94] L. Liu, S. Kheifets, V. Ginis, and F. Capasso, "Subfemtonewton force spectroscopy at the thermal limit in liquids", Physical Review Letters **116**, Publisher: American Physical Society, 228001 (2016).

[95] Y. Amarouchene, M. Mangeat, B. V. Montes, L. Ondic, T. Guérin, D. S. Dean, and Y. Louyer, "Nonequilibrium dynamics induced by scattering forces for optically trapped nanoparticles in strongly inertial regimes", Physical Review Letters **122**, Publisher: American Physical Society, 183901 (2019).

[96] M. Mangeat, Y. Amarouchene, Y. Louyer, T. Guérin, and D. S. Dean, "Role of nonconservative scattering forces and damping on brownian particles in optical traps", Physical Review E **99**, Publisher: American Physical Society, 052107 (2019).

[97] M. P. Wolf, G. B. Salieb-Beugelaar, and P. Hunziker, "PDMS with designer functionalities—properties, modifications strategies, and applications", Progress in Polymer Science **83**, 97–134 (2018).

[98] N. Tucher, "Analysis of photonic structures for silicon solar cells", PhD thesis (Dec. 14, 2016).

[99] Y. Xia and G. M. Whitesides, "Soft lithography", Annual Review of Materials Science **28**, 153–184 (1998).

[100] Z. Wang, A. A. Volinsky, and N. D. Gallant, "Crosslinking effect on polydimethylsiloxane elastic modulus measured by custom-built compression instrument", Journal of Applied Polymer Science **131**, n/a–n/a (2014).