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

Abstract i

Table of Contents iii

List of Figures vi

Nomenclature vii

List of Abbreviations viii

1 Introduction 1

2 Brownian motion 3

2.1 The Brownian motion discovery . . . 3
2.2 Einstein's Brownian theory . . . 4
2.3 The Langevin Equation . . . 8
2.4 Numerical simulation of bulk Brownian motion . . . 13
2.4.1 The numerical Langevin Equation . . . 13
2.4.2 Simulating Brownian motion using Python . . . 15
2.4.3 Speedup using Cython . . . 18
2.5 Conclusion . . . 19

3 Particle characterization and particle tracking using interference properties 20

3.1 Introduction . . . 20
3.2 Reflection Interference Contrast Microscopy . . . 20
3.3 Lorenz-Mie Fit . . . 23
3.3.1 Hologram dependance on the particule characteristics . . . 27
3.3.2 Lorenz-Mie conclusion . . . 30
3.4 Rayleigh-Sommerfeld back-propagation . . . 34
3.4.1 Numerical Rayleigh-Sommerfeld back-propagation . . . 35
3.5 Experimental setup . . . 36
3.6 Experimental pipeline . . . 38
3.6.1 Recording the holograms . . . 38
3.6.2 Fitting the holograms . . . 39
3.6.3 Radius and optical index characterization . . . 40
3.6.4 Conclusion . . . 42

A Appendix 43

A.1 Simulation of the full Langevin equation . . . 44

A.2 Pipeline tracking using pylorenzmie 52

References **65**

List of Figures

Fig. 1: Brownian motion of 1 μm particles in water tracked by hand by Jean Perrin and his colleagues. The points are spaced in time by 30 seconds and 16 divisions represents 50 μm. 4

Fig. 2: Simulation of the bulk Brownian motion 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 or shown. On the bottom, bullets represents the Mean Square Displacement (MSD) computed from the simulated trajectories. The black plain line represents Einstein's theory, which is computed from the square of Eq.2.2.11. 7

Fig. 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 reprensents a gaussian of zero mean and a τ variance, Eq.2.4.3. On the first line simulation is done with $\tau = 10^{-3}$ s and $\tau = 1$ s on the second one. Each column correspond to a number of draw N, from the left to the right $N = 10^2,\ 10^3$ and 10^4. 13

Fig. 4: Mean Relative Squared Error (MRSE) of the PDF measured from a generation of N Gaussian random number w_i, and, the actual Gaussian over which the generatio n is done, Eq.2.4.3. The generation is done over a Gaussian which has a mean value $\langle w_i \rangle = 0$ and the variance $\langle w_i^2 \rangle = \tau$. We explore generation ranging form $N = 10$ to 10^7 and $\tau = 10^{-2}$ to 10 s 15

Fig. 5: a) Set of 100 trajectories simulated using the full Langevin equation for particle of a radius $a = 1$ μm and a mass $m = 10$ μg in water, $\eta = 0.001$ Pa.s. The simulations are done with a time step $\tau = 0.01$ s. b) Bullets represents the measured Mean Squared Displacement (MSD) of the simulated trajectories. The plain black line represents the characteristic time of the diffusion, here $\tau_B = m/\gamma = 0.53$ s. The dotted line represents the MSD theory when $t << \tau_B$, MSD $= \tau^2 k_B T/m$. The dashed line when $t >> \tau_B$, MSD $= 2D\tau$. A detailed explanation of the simulation process can be found in the appendix.A.1. 18

Fig. 6: Figure from [43] 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.3.2.8 to measure the height of the particle (here h). (b) Same as (a) with a wavelength $\lambda_2 = 635$ nm. (c) Time series of the height of a particle h (green: λ_1, magenta: λ_2) and the particle velocity measured along the flow in blue. 21

Fig. 7: a) Raw hologram of a 2.5 μm polystyrene particle measured experimentally with the setup detailed in the chapter 3.5. b) Background obtained by taking the median value of the time series of images of the diffusing particle. c) Normalized hologram given by dividing a) by b). d) Result of the fit of c) using Eq.3.3.4 the particle is found to be at a height $z = 14.77$ μm. e) Comparison of the normalized radial intensity, obtained experimentally form c) and theoretically from d). 26

Fig. 8: Radial intensity profile stack as a function of the distance between the particle center and the focal place of the objective lens, generated for a particle of radius $a = 1.5$ μm and optical index $n = 1.59$. 27

Fig. 10: Radial intensity profile stack as a function of the particle radius, generated for a particle of optical index $n = 1.59$ with a distance between the particle center and the focal place of the objective lens $z = 15$ μm. . . . 28

Fig. 9: Radial intensity profile for of the particle radius $a << \lambda$, generated for a particles of optical index $n = 1.59$ with a distance between the particle center and the focal place of the objective lens $z = 15$ μm and $\lambda = 532$ nm. 28

Fig. 11: Radial intensity profile stack as a function of the particle optical index, generated for a particle of radius $a = 1.5$ μm with a distance between the particle center and the focal place of the objective lens $z = 15$ μm. . 29

Fig. 12: On the left, experimentally measured holograms' radial intensity profile stack, generated from a polystyrene bead of nominal radius $a = 1.5 \pm 0.035$ μm using the experimental setup explained in chapter 3.5. The calibration of this particle radius and optical index is shown in Fig.18. On the right, the corresponding theoretical stack using the result of each individual hologram's fit. 31

Fig. 13: On the left, experimentally measured holograms' radial intensity profile stack, generated from a polystyrene bead of nominal radius $a = 1.5 \pm 0.035$ μm using the experimental setup explained in chapter 3.5. The calibration of this particle radius and optical index is shown in Fig.18. On the right, the corresponding theoretical stack using the result of each individual hologram's fit. 32

Fig. 14: On the left, experimentally measured holograms' radial intensity profile stack, generated from a polystyrene bead of nominal radius $a = 1.5 \pm 0.035$ μm using the experimental setup explained in chapter 3.5. The calibration of this particle radius and optical index is shown in Fig.18. On the right, the corresponding theoretical stack using the result of each individual hologram's fit. 33

Fig. 15: On the left: the original hologram on the top and propagated along 15 μm on the bottom. On the right: Reconstruction using Eq.3.4.5 of the scattered intensity of single colloidal sphere of a radius $a = 1.5$ μm polystyrene spheres, $n_p = 1.59$ in water at a original height of 15 μm. [NEED TO CHECK IF THE PARTICLE HERE IS NOT TO LARGE] 36

Fig. 16: Photo of the custom build microscope used along my thesis. It is mainly composed of 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 colimated 521 μm wavelength laser. 37

Fig. 17: Schematic of the experimental setup. A laser plane wave of intensity I_0 illuminates the chamber containing a dilute suspension of micro-spheres in water. The light scattered by a particle interferes with the incident beam onto the focal plane of an objective lens, that magnifies the interference patten and relays it to a camera. 38

Fig. 18: 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 $n_p = 1.585 \pm 0.002$ and $a = 1.514 \pm 0.003$ μm. 41

Fig. 19: 3D plot of an experimental trajectory measured in water for a particle of optical index $n_p = 1.585$ and radius $a = 1.514$ μm. 42

Nomenclature

α	Noise amplitude
ℓ_{B}	Boltzmann length
ℓ_{B}	Debye length
η	Fluid viscosity
$\eta_{\perp}$	Viscosity orthogonal to a wall, see Eq.??
$\eta_{\parallel}$	Viscosity parrallel to a wall, see Eq.??
γ	Friction coefficient
ρ_{F}	Fluid density
ρ_{P}	Particle density
B	Amplitude of the electrostatic interactions
D	Diffusion coefficient, see Eq.2.2.12
g	Gravity constant
k_{B}	Boltzmann Constant
m	Mass of a particle
N_{A}	Avogrado constant
R	Gas constant
T	Temperature
V_t	Velocity of a particle
X_t	Particle position, see Eq.2.3.19

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
SDE	Stochastic Differential Equations

1 Introduction

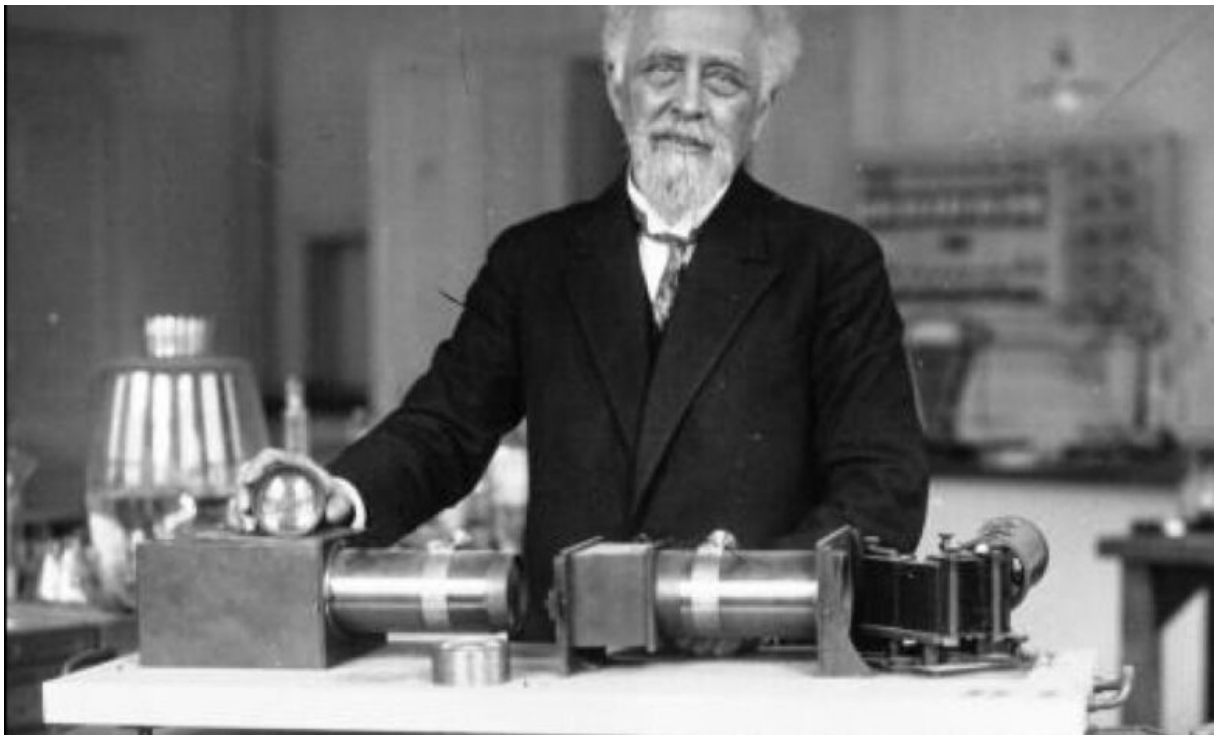
Since the observations of Gordon Moore in the 60's we know that the technological progress is bound to our ability to miniaturize. It's indeed due to the miniaturization that we are able to have more computational power leading to the rise of knew technologies like the Deep Learning [1] that showed the need of large computational capabilities by having the computer program *AlphaGo* beating *Lee Sedol* one of the greatest player of *Go* in 2016. Since this powerful demonstration AIs using the same technologies are showing up in every field, from the language translator to autonomous cars and is know starting to be extensively used in physics with in 2020 the first focus session on machine learning at the *March Meeting* that continued this year with presentations at every sessions. The success of Deep learning is not due to the fact that it's new and fancy algorithm since it known for several decade but only the fact that the miniaturization permitted to do the stunning amount of computation needed to have a smart AI. Our ability to use this technologies is finally bound to our ability to understand the surface physics at the manometer scale.

On another side we have microfluidic since the 80s which is an incredible multidisciplinary field involving chemistry, engineering, soft matter physics and also biotechnology. Microfluidic permitted the development of daily life technologies like the ink-jet printers or more advanced tools such as DNA chips [2] or lab-on-a-chip technology [3]. The ability to compose with a lot of different system to build microfluidic systems is a wonderful playground for physicists which gave a lot of complex systems in confinement to study and understand how different boundaries can change the dynamic properties of a system. At a time of miniaturization and nanotechnologies, the need of tools permitting the systematic study of complex confined system is a key.

In order to address these challenges my work in the past three years focused on using the confined Brownian motion. Brownian Motion is a central paradigm in modern science. It has implications in fundamental physics, biology, and even finance, to name a few. By understanding that the apparent erratic motion of colloids is a direct consequence of the thermal motion of surrounding fluid molecules, pioneers like Einstein and Perrin provided decisive evidence for the existence of atoms [4, 5]. Specifically, free Brownian motion in the bulk us characterized by a typical spatial extent evolving as the square root of time, as well as Gaussian displacements. At a time of miniaturization and interfacial science, and moving beyond the idealized bulk picture, it is relevant to consider the added roles of boundaries to the above context. Indeed, Brownian motion at interfaces and in confinement is a widespread practical situation in microbiology and nanofluidics. In such case, surface effects become dominant and alter drastically the Brownian statistics, with key

implications towards: i) the understanding and smart control of the interfacial dynamics of microscale entities; and ii) high-resolution measurements of surfce forces at equilibrium. Intertingly, a confined colloid will exhibit non-Gaussian statistics in displacements, due to the presence of multiplicative noises induced by the hindered mobility near the wall [6–8]. Besides, the particle can be subjected to electrostatic or Van der Waals forces [9] exerted by the interface, and might experience slippage too [10, 11]. Considering the two-body problem, the nearby boundary can also induce some effective interaction [12]. Previous studies have designed novel methods to measure the diffusion coefficient of confined colloids [13–18], or to infer surface forces [19–24].

In the the first part of the manuscript I will present the history of the Brownian motion and it's basic theory. In a second part I will present particle tracking using Mie holography and our experimental setup. Then the third part will focus on one trajectory analysis in order to infer the surface induced effects on the Brownian motion. In a last chapter I will present more complex inference.



2 Brownian motion

2.1 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 thought 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 movement was to test the presence of a male organ. In order 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 movement was still observable made him invalidate his theory. Interestingly each time that he encountered a material that he was able to 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 today Brownian motion quite difficult and the first theoretical work was actually 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 friend 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 was able to 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 by hand, 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 the results this work that Perrin received the Nobel award in 1926.

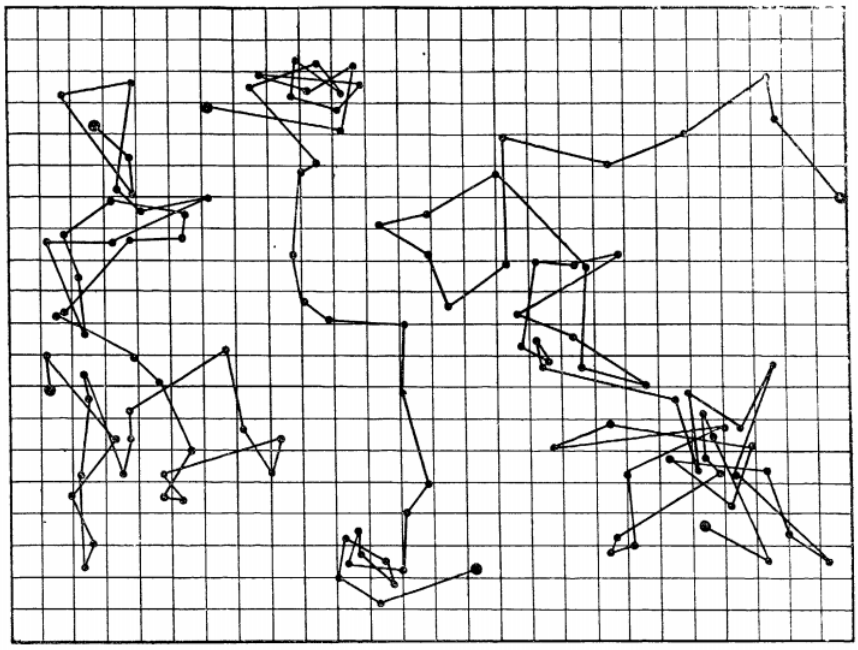


Figure 1: Brownian motion of 1 μm particles in water tracked by hand by Jean Perrin and his colleagues. The points are spaced in time by 30 seconds and 16 divisions represents 50 μm.

2.2 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 individual 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:

$$\mathrm{d}n = n\varphi(\Delta)\mathrm{d}\Delta \ , \tag{2.2.1}$$

where

$$\int_{-\infty}^{\infty} \varphi(\Delta)\mathrm{d}\Delta = 1 \ , \tag{2.2.2}$$

and φ is nonzero only for very small displacement Δ and 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:

$$f(x, t+\tau)\mathrm{d}x = \mathrm{d}x \int_{\Delta=-\infty}^{\Delta=+\infty} f(x+\Delta, t)\varphi(\Delta)\mathrm{d}\Delta \ . \tag{2.2.3}$$

Since τ is very small, we have:

$$f(x, t+\tau) = f(x,t) + \tau\frac{\partial f}{\partial t} \ . \tag{2.2.4}$$

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

$$f(x+\Delta, t) = f(x,t) + \Delta\frac{\partial f(x,t)}{\partial x} + \frac{\Delta^2}{2!}\frac{\partial^2 f(x,t)}{\partial x^2} \ \text{... ad inf.} \tag{2.2.5}$$

Putting all together, in Eq.2.2.3 we obtain:

$$f + \frac{\partial f}{\partial t}\tau = f\int_{-\infty}^{+\infty}\varphi(\Delta)\mathrm{d}\Delta + \frac{\partial f}{\partial x}\int_{-\infty}^{+\infty}\Delta\varphi(\Delta)\mathrm{d}\Delta + \frac{\partial^2 f}{\partial x^2}\int_{-\infty}^{+\infty}\frac{\Delta^2}{2}\varphi(\Delta)\mathrm{d}\Delta \ \text{...} \tag{2.2.6}$$

On the right-hand side, since $\varphi(x) = \varphi(-x)$ all even terms will vanish, moreover, all the odd terms will be very small compared to the precedent. Taking into account Eq.2.2.2 and invoking the definition:

$$\frac{1}{\tau}\int_{-\infty}^{+\infty}\frac{\Delta^2}{2}\varphi(\Delta)\mathrm{d}\Delta = D \ , \tag{2.2.7}$$

Eq.2.2.6 finally becomes:

$$\frac{\partial f}{\partial t} = D\frac{\partial^2 f}{\partial x^2} \ . \tag{2.2.8}$$

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:

$$f(x \neq 0, t = 0) = 0 \text{ and } \int_{-\infty}^{+\infty} f(x,t)dx = n \ . \tag{2.2.9}$$

The solution Eq.2.2.8 is then the Green's function of the heat equation in the bulk:

$$f(x,t) = \frac{1}{\sqrt{4\pi D}} \frac{\exp\left(\frac{-x^2}{4Dt}\right)}{\sqrt{t}} \ . \tag{2.2.10}$$

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:

$$\lambda_x = \sqrt{\langle \Delta^2 \rangle} = \sqrt{2Dt}. \tag{2.2.11}$$

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 will be 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:

$$D = \frac{RT}{N_\mathrm{A}} \frac{1}{6\pi\eta a} = \frac{k_\mathrm{B}T}{6\pi\eta a} \ , \tag{2.2.12}$$

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:

$$N_{\mathrm{A}} = \frac{t}{\lambda_x^2} \frac{RT}{3\pi\eta a} \ . \tag{2.2.13}$$

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 $\frac{16}{N_{\mathrm{A}}}$ and the mass a water molecule by $\frac{18}{N_{\mathrm{A}}}$. 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 theories 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 theoretical 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.

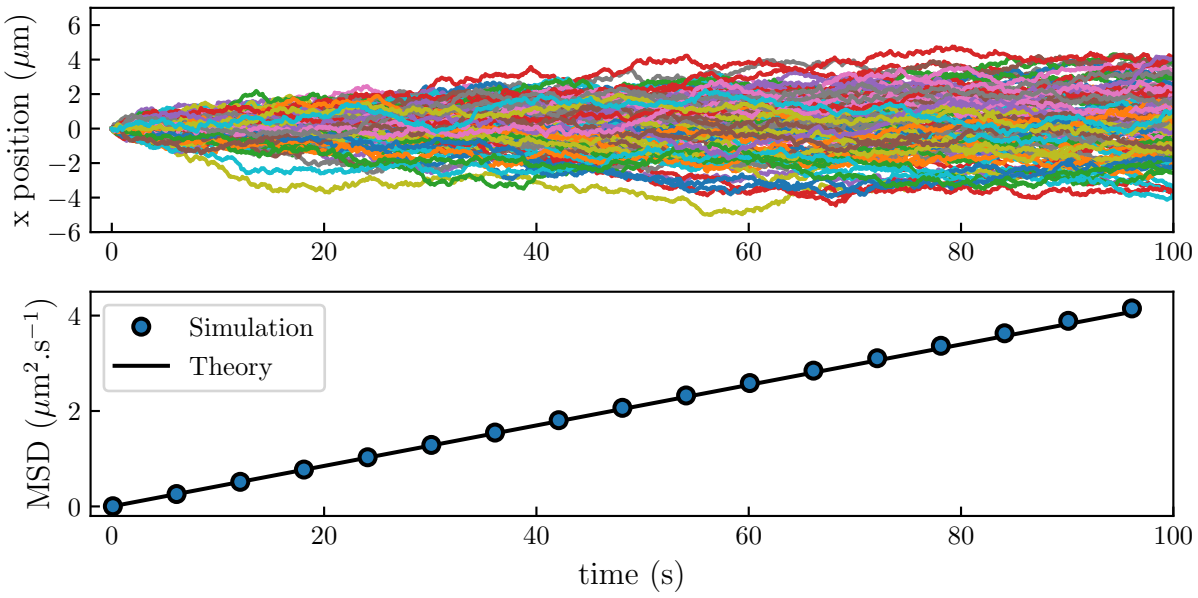


Figure 2: Simulation of the bulk Brownian motion of 1 μm particles in water. On the top each line represents the trajectory of a Brownian particle over 100 seconds. A total of 100 trajectories or shown. On the bottom, bullets represents the Mean Square Displacement (MSD) computed from the simulated trajectories. The black plain line represents Einstein's theory, which is computed from the square of Eq.2.2.11.

2.3 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:

$$m\mathrm{d}V_t = -\gamma V_t \mathrm{d}t + \alpha \mathrm{d}B_t \ , \tag{2.3.1}$$

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, dBt a random noise which has a Gaussian distribution of zero mean thus:

$$\langle \mathrm{d}B_t \rangle = 0 \ , \tag{2.3.2}$$

and variance equal to:

$$\langle \mathrm{d}B_t^2 \rangle = \mathrm{d}t \ . \tag{2.3.3}$$

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:

$$\langle \frac{\mathrm{d}V_t}{\mathrm{d}t} \rangle = -\frac{\gamma}{m}\langle V_t \rangle \mathrm{d}t + \frac{\alpha}{m}\langle \mathrm{d}B_t \rangle \ , \tag{2.3.4}$$

with the properties of dB_t given by Eq.2.3.2, it becomes:

$$\langle \mathrm{d}V_t \rangle = -\frac{\gamma}{m}\langle V_t \rangle \mathrm{d}t \ . \tag{2.3.5}$$

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:

$$\langle x \rangle = \frac{1}{N} \sum_{i=1}^{N} x_i \ , \tag{2.3.6}$$

one can then show that:

$$\frac{\mathrm{d}}{\mathrm{d}t}\langle x \rangle = \frac{\mathrm{d}}{\mathrm{d}t}\left[\frac{1}{N}\sum_{i=1}^{N} x_i\right] = \frac{1}{N}\sum_{i=1}^{N}\frac{\mathrm{d}}{\mathrm{d}t}x_i = \langle \frac{\mathrm{d}}{\mathrm{d}t}x \rangle \ . \tag{2.3.7}$$

The latter thus shows that it is possible to invert average value $\langle \cdot \rangle$ and a derivative. Therefor, Eq.2.3.5 becomes:

$$\frac{\mathrm{d}}{\mathrm{d}t}\langle V_t \rangle = -\frac{\gamma}{m}\langle V_t \rangle \ , \tag{2.3.8}$$

which has a familiar solution:

$$\langle V_t(t) \rangle = V_0 \mathrm{e}^{-\frac{\gamma}{m}t} \ , \tag{2.3.9}$$

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 micro-metric we have $\tau_{\mathrm{B}} \approx 10^{-7}$ s. This means 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:

$$-\gamma V_t \mathrm{d}t + \alpha \mathrm{d}B_t = 0 \ . \tag{2.3.10}$$

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.2.3.10, we can use Eq.2.3.4 to characterize further the unknown coefficient α. In order to do so we compute the mean square value of Eq.2.3.4, starting by taking the second order Taylor expansion:

$$\begin{aligned} \mathrm{d}\left(V_t^2\right) &\simeq \frac{\partial V_t^2}{\partial V_t}\mathrm{d}V_t + \frac{1}{2}\frac{\partial^2 V_t^2}{\partial V_t^2}\left(\mathrm{d}V_t\right)^2 \\ &= 2V_t\mathrm{d}V_t + \left(\mathrm{d}V_t\right)^2 \end{aligned} \tag{2.3.11}$$

combining Eqs.2.3.1 and 2.3.11, we obtain by only keeping the terms of order dt:

$$\mathrm{d}\left(V_t^2\right) = 2V_t\left(-\frac{\gamma}{m}V_t\mathrm{d}t + \frac{\alpha}{m}\mathrm{d}B_t\right) + \frac{\alpha^2}{m^2}\mathrm{d}B_t^2 \ . \tag{2.3.12}$$

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

$$\langle \mathrm{d}(V_t^2)\rangle = -2\frac{\gamma}{m}\langle V_t^2\rangle\mathrm{d}t + 2\frac{\alpha}{m}\langle V_t\mathrm{d}B_t\rangle + \frac{\alpha^2}{m^2}\langle \mathrm{d}B_t^2\rangle \ . \tag{2.3.13}$$

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.2.3.13 becomes:

$$\langle d(V_t^2)\rangle = \left[-2\frac{\gamma}{m}\langle V_t^2\rangle + \frac{\alpha^2}{m^2}\right]\mathrm{d}t \ . \tag{2.3.14}$$

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

$$\langle V_t^2\rangle = \frac{\alpha^2}{2\gamma m} \ . \tag{2.3.15}$$

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

$$\langle \frac{1}{2}mV_t^2\rangle = \frac{1}{2}k_{\mathrm{B}}T \ . \tag{2.3.16}$$

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

$$\alpha = \sqrt{2k_{\mathrm{B}}T\gamma} \ . \tag{2.3.17}$$

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

$$V_t \mathrm{d}t = \sqrt{2\frac{k_\mathrm{B}T}{\gamma}}\mathrm{d}B_t \tag{2.3.18}$$

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

$$X_t = \int_0^t V_{t'} dt' \ , \tag{2.3.19}$$

where we can suppose at the initial time $t = 0$ that $X_0 = 0$. Computing $\langle X_t^2 \rangle$ using Eqs.2.3.18,2.3.19 and 2.3.3 thus gives:

$$\langle X_t^2 \rangle = 2\frac{k_\mathrm{B}T}{\gamma}t = 2Dt \tag{2.3.20}$$

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

$$\mathrm{MSD} = \langle (X_0 - X_t)^2 \rangle = \langle X_t^2 \rangle \ , \tag{2.3.21}$$

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.2.2.11. 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.2.3.20 using the velocity correletor such as:

$$\langle X_t^2 \rangle = \int_0^t \int_0^t \langle V_{t'} V_{t''} \rangle dt' dt'' \ . \tag{2.3.22}$$

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.2.3.9, one has:

$$\langle V_t V_0 \rangle = \langle V_0^2 \rangle \mathrm{e}^{-t/\tau_\mathrm{B}} \ . \tag{2.3.23}$$

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:

$$\langle V_t V_t' \rangle = \frac{k_\mathrm{B}T}{m} \mathrm{e}^{-|t-t'|/\tau_\mathrm{B}} \ . \tag{2.3.24}$$

One can solve Eq.2.3.22 by splitting the integral in two parts, where $t' > t''$ and $t' < t''$:

$$\begin{aligned} \langle X_t^2 \rangle &= \frac{k_\mathrm{B}T}{m} \int_0^t dt' \int_0^{t'} dt'' \mathrm{e}^{-|t'-t''|/\tau_\mathrm{B}} = 2\frac{k_\mathrm{B}T}{\gamma} \left(\int_0^t dt' \left[1 - \mathrm{e}^{-t'/\tau_\mathrm{B}} \right] \right) \\ &= 2\frac{k_\mathrm{B}T}{\gamma} \left(t - \tau_\mathrm{B} \left[1 - \mathrm{e}^{-t/\tau_\mathrm{B}} \right] \right) \ . \end{aligned} \tag{2.3.25}$$

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

$$\begin{aligned} \langle X_t^2 \rangle &\simeq 2\frac{k_\mathrm{B}T}{\gamma} \left(t - \tau_\mathrm{B} \left[1 - 1 + \frac{t}{\tau_\mathrm{B}} - \frac{t^2}{2\tau_\mathrm{B}^2} \right] \right) \\ &= \frac{k_\mathrm{B}T}{m} t^2 \ . \end{aligned} \tag{2.3.26}$$

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:

$$\langle X_t^2 \rangle \simeq 2\frac{k_\mathrm{B}T}{\gamma} t = 2Dt \ . \tag{2.3.27}$$

This is the diffusive regime where the MSD, as found earlier, Eq.2.3.20 with the over-damped Langevin equation. To study this different results, it can be interesting to simulate the Brownian motion.

2.4 Numerical simulation of bulk Brownian motion

2.4.1 The numerical Langevin Equation

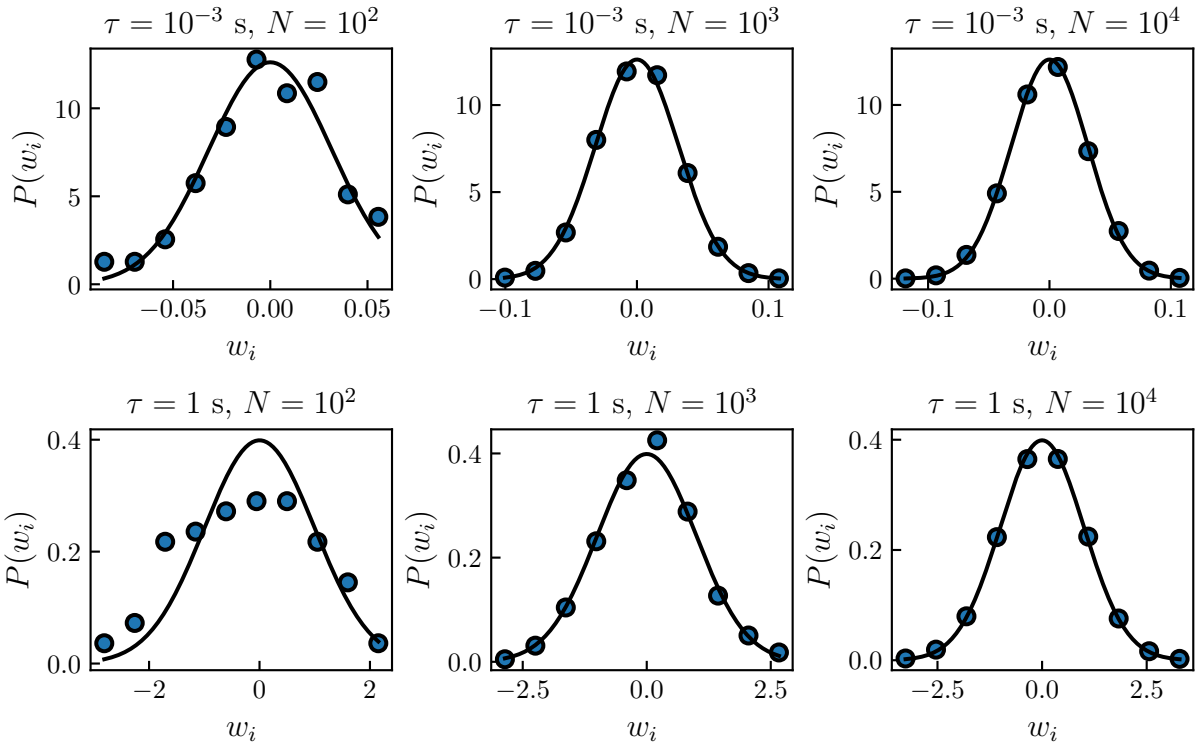


Figure 3: Bullets represents the probability density function of w_i, a Gaussian distributed number with a mean value $\langle w_i \rangle$ and a variance $\langle w_i^2 \rangle = \tau$. The plain black line reprensents a gaussian of zero mean and a τ variance, Eq.2.4.3. On the first line simulation is done with $\tau = 10^{-3}$ s and $\tau = 1$ s on the second one. Each column correspond to a number of draw N, 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. If one approximate the continuous position of a particle X_t at a time t by a discrete-time sequence x_i which is the solution of the equation at a time $t_i = i\tau$ with τ being the time step of the simulation. One can then use the Euler method to numerically write V_t as

$$V_t \simeq \frac{x_i - x_{i-1}}{\tau} \ , \tag{2.4.1}$$

and $\mathrm{d}V_t$ as

$$\begin{aligned}\frac{\mathrm{d}V_t}{\mathrm{d}t} &\simeq \frac{\frac{x_i - x_{i-1}}{\tau} - \frac{x_{i-1} - x_{i-2}}{\tau}}{\tau} \\ &= \frac{x_i - 2x_{i-1} + x_{i-2}}{\tau^2} \ .\end{aligned} \tag{2.4.2}$$

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[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:

$$P(w_i) = \frac{1}{\sqrt{2\pi\tau}} \mathrm{e}^{-\frac{w_i^2}{2\tau}}. \tag{2.4.3}$$

Such a number can be simply 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.2.4.1, 2.4.2 and w_i, the full Langevin equation becomes:

$$m\frac{x_i - 2x_{i-1} + x_{i-2}}{\tau^2} = -\gamma\frac{x_i - x_{i-1}}{\tau} + \sqrt{2k_\mathrm{B}T\gamma}w_i \ . \tag{2.4.4}$$

From the latter, one can write x_i as:

$$x_i = \frac{2 + \tau/\tau_\mathrm{B}}{1 + \tau/\tau_\mathrm{B}}x_{i-1} - \frac{1}{1 + \tau/\tau_\mathrm{B}}x_{i-2} + \frac{\sqrt{2k_\mathrm{B}T\gamma}}{m(1 + \tau/\tau_\mathrm{B})}\tau w_i \ , \tag{2.4.5}$$

where we can observe that we need two initial condition are needed, the two first positions of the particle. Numerically, those positions could be randomly generated or simply set to 0, if enough statics are generated it will not play much into the results.

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

2.4.2 Simulating Brownian motion using Python

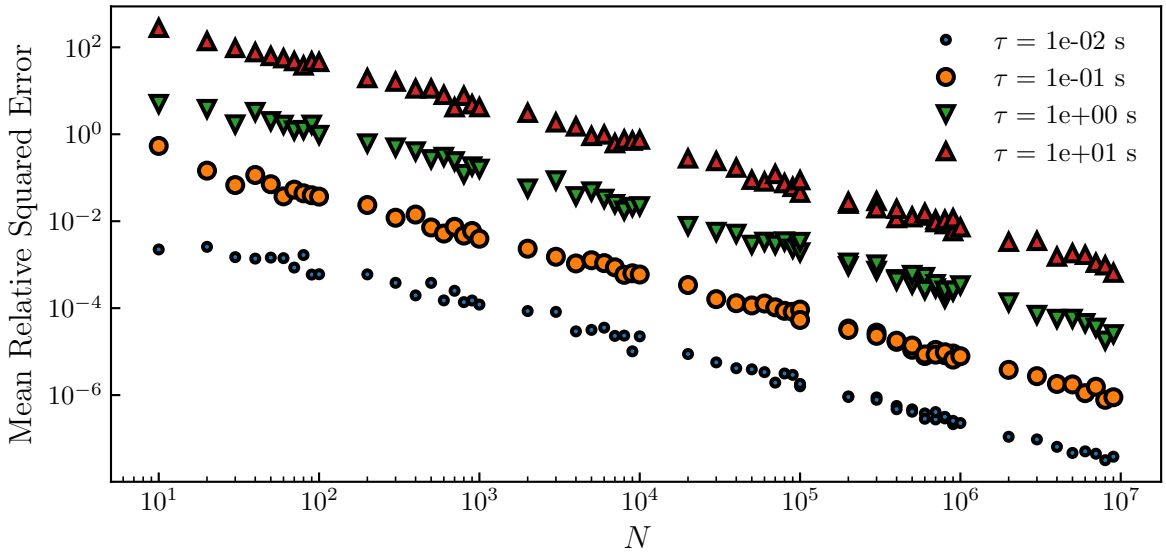


Figure 4: Mean Relative Squared Error (MRSE) of the PDF measured from a generation of N Gaussian random number w_i, and, the actual Gaussian over which the generatio n is done, Eq.2.4.3. The generation is done over a Gaussian which has a mean value $\langle w_i \rangle = 0$ and the variance $\langle w_i^2 \rangle = \tau$. We explore generation ranging form $N = 10$ to 10^7 and $\tau = 10^{-2}$ to 10 s

Before, deeping into the actual simulation, it could be interresting to ask ourselve about how long the simulation should be. Indeed, for the different observables' mean value to remain constant we should wait a suffisient amount of time. It is possible to have qualitative approach by generating N number w_i and measuring the resulting PDF $P_c(w_i)$ and looking how much we need to increase N to have $P_c(w_i) \simeq P(w_i)$. As we can see on the Fig.3, for simulation made with $\tau = 10^{-3}$ s and $tau = 1$ s, we oberve that as we increase the number of generated numbers N, the measured PDF, get closer to the real one given by Eq.2.4.3.

To have a more quantitative approach, one can compute the Mean Relative Squared Error (MRSE)) between the measured PDF $P_c(w_i)$ and $P(w_i)$ as a function of the number of generated numbers N, such as:

$$\text{MRSE} = \langle \frac{(P_c(w_i) - P(w_i))^2}{P(w_i)^2} \rangle \Big|_N \tag{2.4.6}$$

where the notation $|_N$ denotes that N numbers are generated. Additionaly, since we measure the $P_c(w_i)$ by doing an histogram, the question of how many bins is used should

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

$$\text{Bin width} = 2\frac{\text{IQR}(\{w_i\})}{\sqrt[3]{N}}, \tag{2.4.7}$$

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

```
import numpy as np

def _iqr(wi):
        """Function to compute interquartile range."""
        return np.subtract(*np.percentile(x, [75, 25]))

def optimal_bins(wi):
        """
        Function to compute the optinal number of bins using Freedan-diaconis rule.
        Input: list of random numbers | Output: optimal bins number
        """

        n = int(diff(wi) / (2 * _iqr(wi) * np.power(len(wi), -1 / 3)))

        if n <= 2:
                return 2
        else:
                return n
```

As we can see on the Fig.4, for τ varying between 10^{-2} and 10 seconds, and, N between 10 and 10^6, the MRSE decreases as N increases. More over, it is interesting to observe that the MRSE is greater as τ increases for a fix N value. Indeed, as an example we would need to only generate $N = 10^{-3}$ numbers to obain a 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 is known numerically, one could use it to simulate some Brownian trajectories. A simple way to do the simulation is shown using Python in the Jupyter notebook framework in the appendix.A.1. A set of trajectory 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 $\tau_B = 0.53s$. Moreover, as one can see on the Fig.5-b), the MSD is correctly modeled by the Eq.2.3.26 for $\tau << \tau_B$ and by the

Eq.2.3.27 for $\tau >> \tau_B$. Please note that for non-continuous data such as the simulation or experimental trajectories, and for a given time increment Δt, the MSD, is generally defined by:

$$\langle \Delta x^2 \rangle \big|_t = \langle (x(t + \Delta t) - x(t))^2 \rangle \big|_t \ , \tag{2.4.8}$$

where the average $\langle \rangle|_t$ is performed over a time t. Additionally, it can be numerically using the following Python function.

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

Additionally, as we have seen earlier, the Langevin Equation can be simplified to it's over-damped version Eq.2.3.10. In this case, the time step of the simulation τ should be greater than the diffusion time τ_B. Thus, one who is interested at the long time statistical properties of the Brownian motion can use the over-damped Langevin equation. In this case by putting $m = 0$ into Eq.2.4.5, one can write x_i as:

$$x_i = x_{i-1} + \sqrt{2D} w_i \ . \tag{2.4.9}$$

The statistical properties at long time could be retrieved by simulating Brownian motion using the full Langevin equation. But, since the integration scheme used for Eq.2.4.5 requires $\tau << \tau_B$ long simulation are necessary to retrieve the over-damped properties. As an example, for the Fig.2 I directly used Eq.2.4.9 for the simulation using the Python Snippet below.

```
        import numpy as np

        N = 1000  # trajectory length
        D = 1  # diffusion coefficient
        tau = 0.5  # time step
        trajectory = np.cumsum(np.sqrt(2 * D) * np.random.normal(0, np.sqrt(tau), N))
```

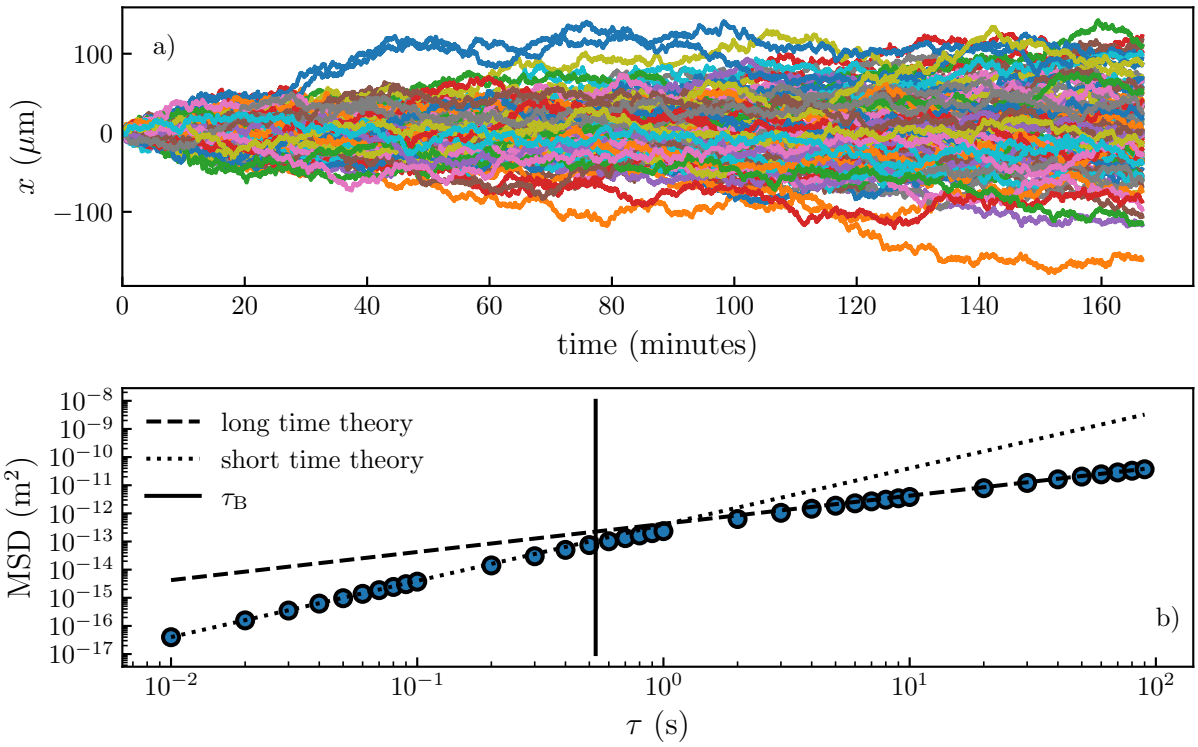


Figure 5: a) Set of 100 trajectories simulated using the full Langevin equation for particle of a radius $a = 1$ μm and a mass $m = 10$ μg in water, $\eta = 0.001$ Pa.s. The simulations are done with a time step $\tau = 0.01$ s. b) Bullets represents the measured Mean Squared Displacement (MSD) of the simulated trajectories. The plain black line represents the characteristic time of the diffusion, here $\tau_B = m/\gamma = 0.53$ s. The dotted line represents the MSD theory when $t << \tau_B$, MSD $= \tau^2 k_B T/m$. The dashed line when $t >> \tau_B$, MSD $= 2D\tau$. A detailed explanation of the simulation process can be found in the appendix.A.1.

2.4.3 Speedup using Cython

I would like to point out, that the optimization of a simple simulation of Brownian trajectory can be interesting. Indeed, using a pure Python code as presented in the first part of the appendix.A.1, the simulation of one trajectory of 10^6 steps, needs 6 s to be computed. Thus, more than 10 minutes should be required to compute the 100 trajectories showed in the Fig.5. This is long due to how Python always verify that what we do is allowed, it thus needs to verify at each step of the `for` loop the object type of each variable. This is in general the main drawback of the interpreted language, one solution is then to compile the part of the code where the `for` loops are computed. One can use the Cython package in order to specify the type of every single variable, once that done, it will convert the lengthy part in C and compile it. As presented in the appendix.A.1, doing that on full Langevin simulation reduces the time to generate a 10^6 steps trajectory from 6 s to 30 ms thus achieving a speedup of $\simeq 200\times$. Moreover, in the compiled version, we are here speed bound by the random number generation, indeed it takes 24.0 ± 0.8 ms using `numpy`.

Additionally, as shown at the end of the appendix.A.1 even pure C implementation of the random generation can be slower than the `numpy` one, thanks to impressive optimization. Thus, using those tools, this simulation is probably as optimize as it can get.

2.5 Conclusion

In this chapter, we have covered the history of the Brownian motion from the first observation of Robert Brown in the middle of 19th century to it's mathematical and experimental proofs in the early 20s. We have then described mathematicaly the bulk Brownian motion, and its important stastical properties. Finaly, we have used the later description in order to simulate Brownian motion using the both the full Langevin equation and it's over-damped version. In the following chapter, I will describe experimental methods using light interference properties in order to track colloidal particle.

3 Particle characterization and particle tracking using interference properties

3.1 Introduction

Properties of coherent light to produce interference is widely use in metrology for a long time with, for example, the famous Fabry-Pérot [32, 33] and Michelson interferometers [34]. The latter was initially used to measure earth's rotation and is still used today, in particular, for the recent measurement of gravitational waves [35]. 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 :

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

The first one, RICM, uses the principle of optical difference path as a Michelson interferometer. The other two, uses the interference between the light scattered by the colloid and the incident light. Generally, both of the sources are colinear, thus, we speak of in-line holography.

3.2 Reflection Interference Contrast Microscopy

Reflection Interference Contrast Microscopy was first introduced in cell biology by Curtis to study embryonic chick heart fibroblast [36] in 1964. RICM gained in popularity 40 years after both in biology and physics [37–42]. It was also used recently in soft matter physics to study elastohydrodynamic lift at a soft wall [43].

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 at the colloid's surface. The difference of optical path between two reflection create interference patterns. Let's take an interest at the mathematical description of this phenomenon. In the far field, we can describe two different one-dimensional electric field vectors of the same pulsation ω [44] as:

$$\vec{E}_1(\vec{r},t) = \vec{E}_{0_1}\cos(\vec{k}_1\cdot\vec{r}-\omega t+\epsilon_1)\ , \tag{3.2.1}$$

and

$$\vec{E}_2(\vec{r},t) = \vec{E}_{0_2}\cos(\vec{k}_2\cdot\vec{r}-\omega t+\epsilon_2)\ . \tag{3.2.2}$$

Where the k is the wave number $k = 2\pi n_{\mathrm{m}}/\lambda$, λ beeing the illumination wavelength, n_{m} the optical index of the medium, $\epsilon_{1,2}$ the initial phase of each wave and $\vec{r}$ the position from the source. Here, the origin ($\vec{r} = \vec{0}$) is taken at the position of the first reflection (on the glass slide). Thus at the particle, $\vec{r}$ is be given by the particle's height such that $|r| = z$, the particle-subtract distance.

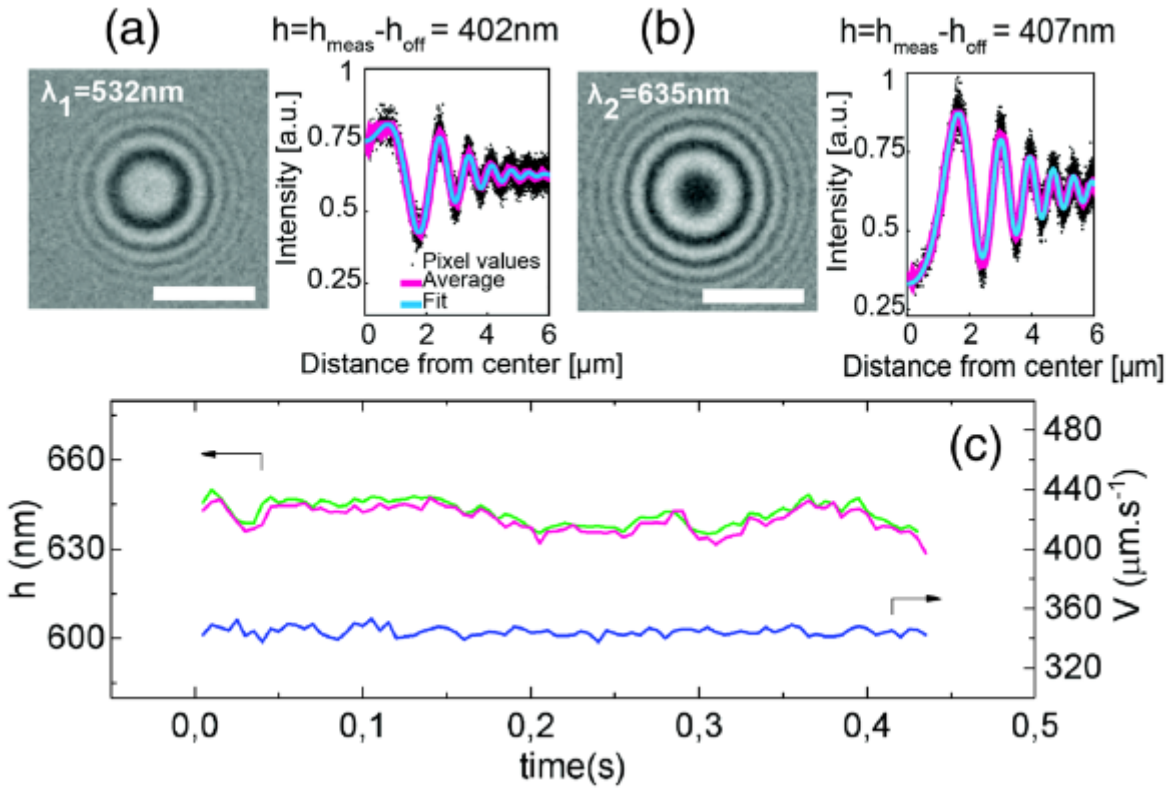


Figure 6: Figure from [43] representing RICM with two wavelengths. (a) Left: interference patterns created with a wavelength $\lambda_1 = 532$ nm (scale bar 5 μm). Right: radial intensity profile (black dots) extracted from the image, azimuthally averaged (magenta line) and fitted with Eq.3.2.8 to measure the height of the particle (here h). (b) Same as (a) with a wavelength $\lambda_2 = 635$ nm. (c) Time series of the height of a particle h (green: λ_1, magenta: λ_2) and the particle velocity measured along the flow in blue.

Experimentally, we measure the intensity of the interference patterns, those can be computed from the time averaged squared sum of the electric field $\vec{E} = \vec{E}_1 + \vec{E}_2$. The measured intensity is thus given by:

$$I = \langle \vec{E}^2 \rangle = \langle \vec{E}_1^2 + \vec{E}_2^2 + 2\vec{E}_1 \cdot \vec{E}_2 \rangle = \langle \vec{E}_1^2 \rangle + \langle \vec{E}_2^2 \rangle + 2\langle \vec{E}_1 \cdot \vec{E}_2 \rangle \tag{3.2.3}$$

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 light intensities. Using the trigonometric formula $2\cos(a)\cos(b) = \cos(a+b) + \cos(a-b)$ we have:

$$\langle \vec{E}_1 \cdot \vec{E}_2 \rangle = \langle \frac{1}{2} \vec{E}_{0_1} \vec{E}_{0_2} \left[\cos\left(\vec{k}_1 \cdot \vec{r} - \vec{k}_1 \cdot \vec{r} + \phi \right) + \cos\left(2\omega t + \phi'\right) \right] \rangle \ . \tag{3.2.4}$$

As we average over the time, the second cos will vanish since in general $\langle \cos(at+b) \rangle_t = 0$ thus:

$$\langle \vec{E}_1 \cdot \vec{E}_2 \rangle = \frac{1}{2} \langle \vec{E}_{0_1} \vec{E}_{0_2} \rangle \cos\left(\vec{k}_1 \cdot \vec{r} - \vec{k}_2 \cdot \vec{r} + \phi \right) \tag{3.2.5}$$

with ϕ the phase difference between the two fields, which is generally 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:

$$I = I_1 + I_2 + 2\sqrt{I_1 I_2} \cos\left(\vec{k}_1 \cdot \vec{r} - \vec{k}_2 \cdot \vec{r} + \phi \right) \tag{3.2.6}$$

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

$$I = I_1 + I_2 + 2\sqrt{I_1 I_2} \cos\left(\frac{4\pi n_{\mathrm{m}}}{\lambda} z + \phi \right) \tag{3.2.7}$$

If we now suppose that we have a spherical particle at a height z we can develop the radial interference intensity $I(x)$ as [42]:

$$I(x) = A_0 + A_1 \mathrm{e}^{-b_1 x^2} + A_2^{-b_2 x^2} \cos\left[\frac{4\pi n_m}{\lambda} \left(g(x) + z \right) + \phi \right] \tag{3.2.8}$$

Where A_i and b_i are fit parameters and $g(x)$ denotes the contour of the sphere. Finally, this method benefits of equations that are computationally light and permits to have a quick tracking of particles. However, as we can see on Eq.3.2.8, due to the periodicity of the cosinus, the interference pattern will be 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 limitation by using 2 different wavelength to $\simeq 1.2$ μm as used in [43]. Despite the precision of this method which can reach the 10 nm spatial resolution; the measurement ambiguity is not compatible with the study of micro-particle Brownian motion due to it's spacial extent, hence, RICM is not usable for our context. As a matter of fact, we experimentally reach height span of a few microns.

3.3 Lorenz-Mie Fit

When a colloid is illuminated with a plane wave, a part of the light is scattered. In consequence, the superimposition of 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 approximations [45] to describe the scattering of the particles. Indeed, one would need to use what we call the Lorenz-Mie theory which describes the scattering of dielectric spheres; this theory was found by Lorenz and independently by Mie in 1880 and 1908 [46, 47].

It is in the early 2000 that the Lorenz-Mie theory was first used in order to track and characterize particles [48, 49]. Since then, a lot of studies has been realized with this method [50]. In the following I will describe the Lorenz-Mie Fit method. In in this part, the height of the particle z 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 :

$$\vec{E}_0(\vec{r}, z) = E_0(\vec{r})\mathrm{e}^{ikz}\hat{e} \tag{3.3.1}$$

Let's consider a particle of radius a at a position $\vec{r}_p$, the scattered field can be written using the Lorenz-Mie theory [44] as:

$$\vec{E}_\mathrm{s}(\vec{r}, z) = \vec{f}_\mathrm{s}(k(\vec{r} - \vec{r}_p))E_0(\vec{r}) \exp(-ikz) \tag{3.3.2}$$

With $\vec{f}_\mathrm{s}$, the Lorenz-Mie scattering function [44]. The intensity I that we measure at $\vec{r}$ is given by the superimposition of incident and scattered waves. Since the measurements are done at the focal plane, i.e. $z = 0$, I is given by:

$$\begin{aligned} I(\vec{r}) &= |\vec{E}_{\mathrm{s}}(\vec{r},0) + \vec{E}_0(\vec{r},0)|^2 \\ &= E_0^2(\vec{r}) + 2E_0^2\,\mathrm{Re}\left(\vec{f}_{\mathrm{s}}(k(\vec{r}-\vec{r}_p))\hat{e}\right) + |\vec{f}_{\mathrm{s}}(k(\vec{r}-\vec{r}_p))|^2 \end{aligned} \tag{3.3.3}$$

The most of the experimental defects on the images are due to spacial illumination variation caused by dust particle and such. It 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 is 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 take the median of a stack of images. Additionally, for having the latter to work, the movie should be long enough to have the particle diffuse enough, if not a ghost of the particle will appear on the background. Moreover, This process also permits getting rid of the immobile particle that could generate any additional noise. An example of hologram before and after the normalization is shown in Fig.7 a-c).

Finally, we write the normalized intensity I/I_0:

$$\frac{I(\vec{r})}{I_0(\vec{r})} = 1 + 2\,\mathrm{Re}\left(\vec{f}_{\mathrm{s}}(k(\vec{r}-\vec{r}_p))\hat{e}\right) + |\vec{f}_{\mathrm{s}}(k(\vec{r}-\vec{r}_p))|^2 \tag{3.3.4}$$

Now that we have the analytical form of the holograms' intensity, it is possible to fit an experimental one to Eq.3.3.4 as shown in Fig.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:

$$\vec{f}_{\mathrm{s}}(k\vec{r}) = \sum_{n=1}^{n_c} \frac{i^n(2n+1)}{n(n+1)} \left(i a_n \vec{N}_{eln}^{(3)}(k\vec{r}) - b_n \vec{M}_{oln}^{(3)}(k\vec{r})\right) \tag{3.3.5}$$

where $\vec{N}_{eln}^{(3)}(k\vec{r})$ and $\vec{M}_{oln}^{(3)}(k\vec{r})$ are the vector spherical harmonics. a_n and b_n are some coefficients that depend 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 j_n and Hankel h_n functions as [44]:

$$a_n = \frac{\zeta^2 j_n(\zeta ka)kaj_n'(ka) - j_n(ka)[\zeta kaj_n(\zeta ka)]'}{\zeta^2 j_n(\zeta ka)kah_n^{(1)'}(ka) - h_n^{(1)}(ka)\zeta kaj_n'(\zeta ka)}\ , \tag{3.3.6}$$

and

$$b_n = \frac{j_n(\zeta ka)kaj_n'(ka) - j_n(ka)\zeta kaj_n'(mka)}{j_n(\zeta ka)kah_n^{(1)'}(ka) - h_n^{(1)}(ka)\zeta kaj_n'(mka)} \ , \tag{3.3.7}$$

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

Finally, an hologram is mainly given by the Lorenz-Mie scattering function Eq.3.3.5. Moreover, as we can oberve with the Eqs.3.3.6 and 3.3.7, an hologram will vary with a lot of parameters (λ, n_m, n_p, a and $\vec{r}_\mathrm{p}$) which can all be fitted. 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 precisely the position of the particle $\vec{r}_\mathrm{p}$ and in the same time characterize the radius and optical index of the colloid. As a side note, it is even possible to characterize a particle without a priori knowledge of its characteristics using Bayesian approach [51, 52].

Computing Eq.3.3.5 numerically brings another interesting question, as it is analytically written as a sum over n; one could ask after which number of terms n_c the series will converge. It has actually been found that the series converge after a number of terms [53]

$$n_c = ka + 4.05(ka)^{1/3} + 2 \ . \tag{3.3.8}$$

Consequently, larger particles' holograms will need 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 a number of terms $n_c = 55$ in water and and 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 take the same time to be computed; an image of a 2.5 μm particle's hologram will need $55/18 \simeq 3$ times more time to be fitted compared to the holgram of a 0.5 μm particle.

If a reader wants to evaluate an 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 using the following Python snippet which was used to make Fig.14 and 13:

```
1 import holopy as hp
2 from holopy.scattering import calc_holo, Sphere
3
4 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 it's 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 pixel 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 be directly be plotted using:
hp.show(holo)
```

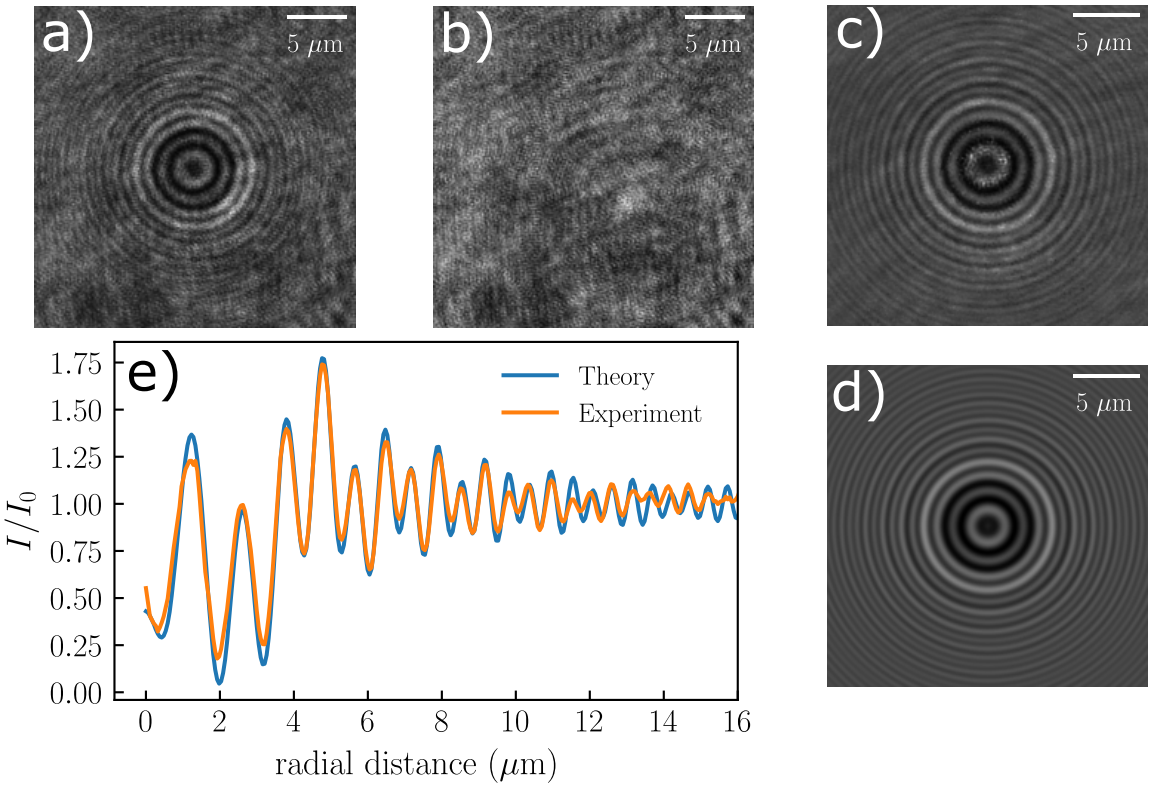


Figure 7: a) Raw hologram of a 2.5 μm polystyrene particle measured experimentally with the setup detailed in the chapter 3.5. b) Background obtained by taking the median value of the time series of images of the diffusing particle. c) Normalized hologram given by dividing a) by b). d) Result of the fit of c) using Eq.3.3.4 the particle is found to be at a height $z = 14.77$ μm. e) Comparison of the normalized radial intensity, obtained experimentally form c) and theoretically from d).

3.3.1 Hologram dependance on the particule characteristics

As we can see with the Eq.3.3.5, the in-line holograms vary with the position, radius and optical index of the particle. For in-line holograms, since the incident and scattered field, the x and y position of the particle will simply be given by the center of the hologram. Thus, it is possible to track only movement of the particle only in 2 dimensions by using algorithm such as the Hough transform 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 $a = 2.5$ μm and $z = 5$ μm.

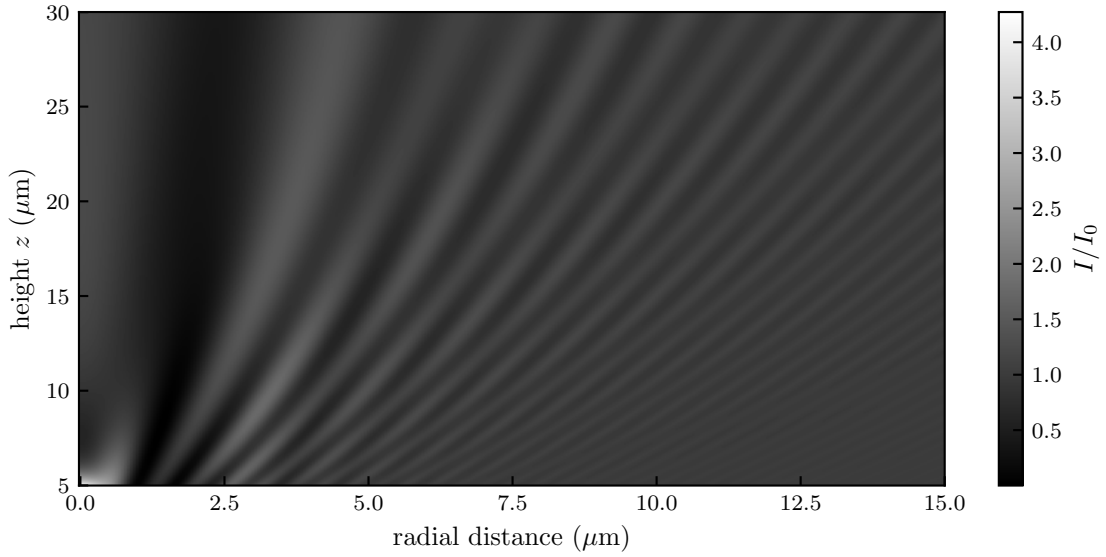


Figure 8: Radial intensity profile stack as a function of the distance between the particle center and the focal place of the objective lens, generated for a particle of radius $a = 1.5$ μm and optical index $n = 1.59$.

In order to gain some insights on how the holograms vary with the different parameters, one can to compute holograms for particles of different size and height. We will start by looking at the a particle of a radius $a = 1.5$ μm and $n_{\mathrm{p}} = 1.59$ as shown in the Fig.8. In this case, one can observe that as the distance between the particle and the focal plane z increases, the hologram's rings gets larger. Unlike a Michelson interferometer or RICM we do not observe the rings scrolling.

Additionally, this thickening of the rings can also be observed on the Fig.12, where hologram's intensity profile are plot 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 on the Fig.14.

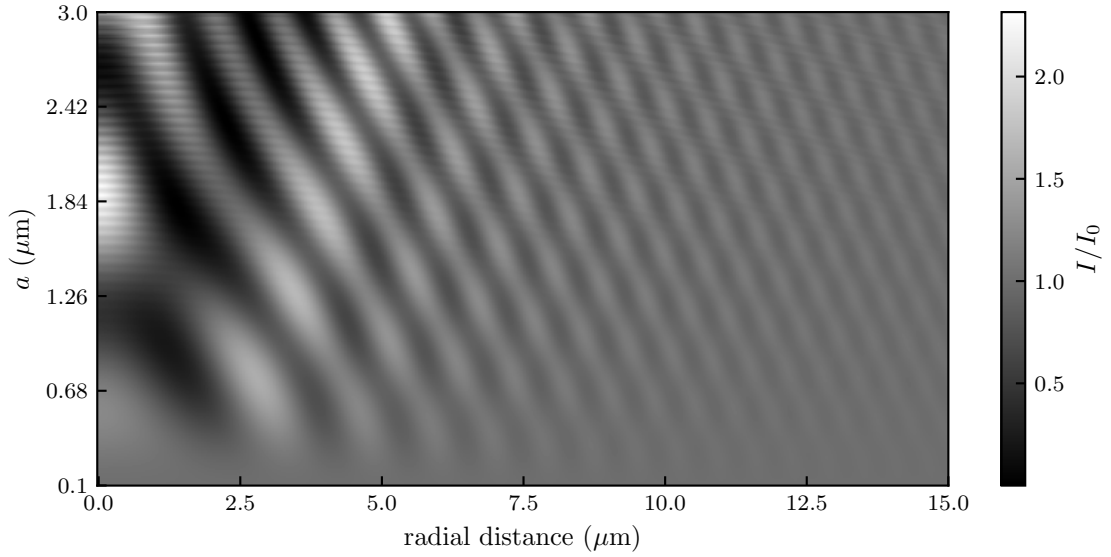


Figure 10: Radial intensity profile stack as a function of the particle radius, generated for a particle of optical index $n = 1.59$ with a distance between the particle center and the focal place of the objective lens $z = 15$ μm.

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

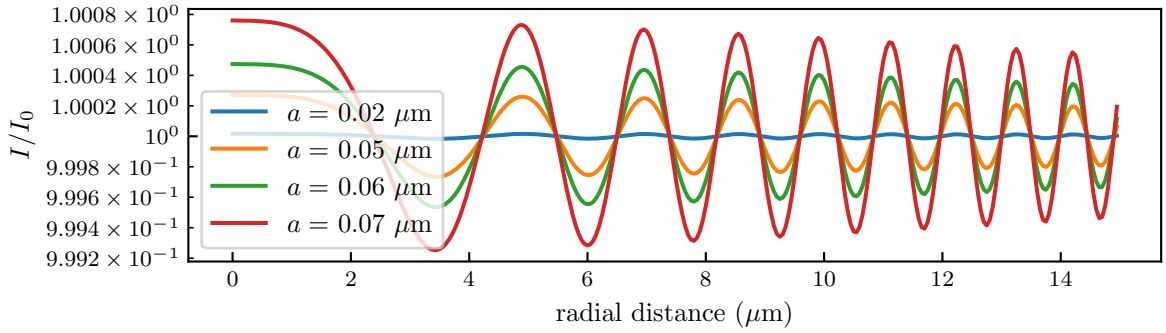


Figure 9: Radial intensity profile for of the particle radius $a << \lambda$, generated for a particles of optical index $n = 1.59$ with a distance between the particle center and the focal place of the objective lens $z = 15$ μm and $\lambda = 532$ nm.

We can now take a look at the variation with respect to the radius of the particle as shown on the Fig.10 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 $a << \lambda$ we do not observe the rings this is due to the fact that for the small particles, the scattering can be approximated using the Reyleigh theory which tells us that the scattering is isotropic.

Thus, the variation of intensity around I_0 will be smaller for small particle.

Also, in this small particle regime, the particle size will not affect the general shape of the hologram but just the intensity of the hologram as it can be seen on the Fig.9, for particle of radius $a = 0.02$ μm to $a = 0.07$ μm for a wavelength $\lambda = 0.532$ μm.

Additionally, since the noise to signal ratio will be lower than for bigger particles, it will be less precise to characterize small colloids compared to the wavelength.

As the particle gets bigger, the scattering become anisotropic and it mostly towards the incident plane wave direction. This effects leads to an increase of the amplitude of the rings I/I_0, as one can see on the Fig.10. Thus, the noise to signal ratio is high enough to easily discern the hologram on top of the noise as one can see on the experimental picture fig.7-a). One who wants to use this method should thus take care to use large enough particle for the holograms' intensity to be greater than the camera noise level.

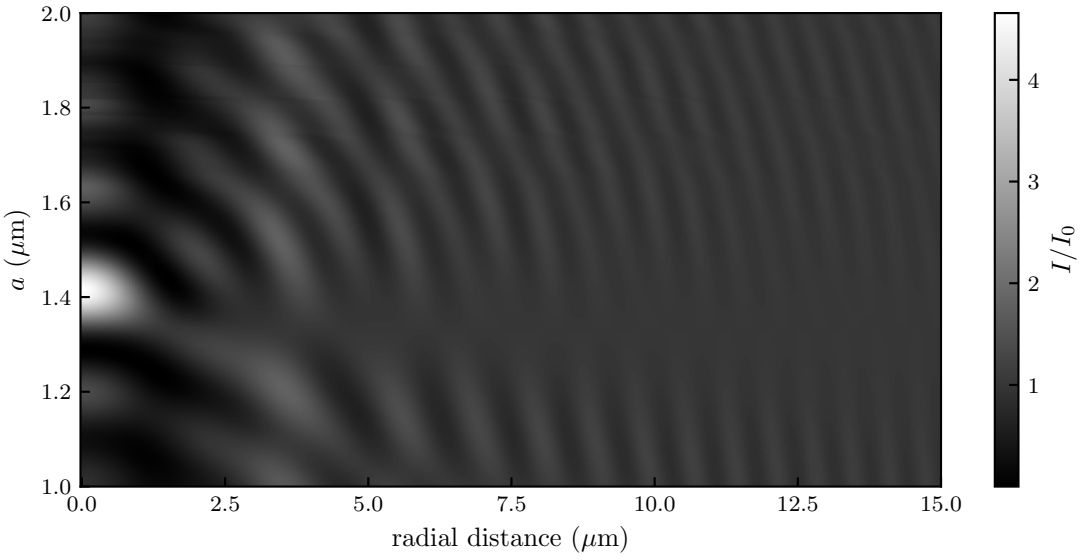


Figure 11: Radial intensity profile stack as a function of the particle optical index, generated for a particle of radius $a = 1.5$ μm with a distance between the particle center and the focal place of the objective lens $z = 15$ μm.

Finally, one can also check how the holograms are varying with the optical index a particle. In this case it is not the particle's optical index n_p which will matter the most but the ratio $\zeta = n_\mathrm{p}/n_\mathrm{m}$ which can be found in the a_n and b_n formulas, Eq.3.3.6 and 3.3.7. Indeed, for the scatter to happen, the optical index n_p of the colloid needs to be different from the optical index of the surrounding medium n_m. Additionally, the numerical solution of the Lorenz-Mie scattering is not stable for $n_\mathrm{p} \simeq n_\mathrm{m}$. In the Fig.11, we can observe holograms of a particle of radius $a = 1.5$ μm at fix distance between the particle and

the focal plane of the objective lens $z = 15$ μm with a varying colloid's optical index, in water $n_\mathrm{m} = 1.33$. On the Fig.11, one can thus observe than for $n_\mathrm{p} \simeq n_\mathrm{m}$ we do not oberve any holograms. Additionally, one can observe that the noise to signal ratio gradually increases as n_p became different from n_m. One who wants to use this method should thus take care to the particle material or the solvent to have n_m different enough to n_p for the holograms' intensity to be greater than the camera noise level.

3.3.2 Lorenz-Mie conclusion

The combination of the height, optical index and radius of colloid thus gives unique holograms. This uniqueness of the holograms permits extracting precisely the position, optical index and radius of a colloid. In order to see how holograms are for different couples of parameters on the Figs.13 and 14, one can see possible holograms for different size and height. Additionally, one can use the Jupyter Notebook on my github repository in order to plot any hologram .

Finally, Lorenz-Mie is the most versatile in-line holographic method, indeed, it permits tracking and characterize unique particles even without a priori knowledge. Besides, it is possible to write the Lorenz-Mie function $\vec{f_\mathrm{s}}$ for particular cases such as anisotropic [54], non-spherical particles [55] or particle clusters [54, 56] to name a few; such possibilities open the door to a lot of experimental studies. Additionally, it can reach really high precision as the tenth of nanometer on the position and radius as well as 10^{-3} on the optical index [49].

Unfortunately, the Lorenz-Mie fitting suffer 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 as been done to have faster tracking, such as random-subset fitting [57], GPU (graphical processing unit) acceleration, machine-learning [58, 59] and deep neural networks [60].

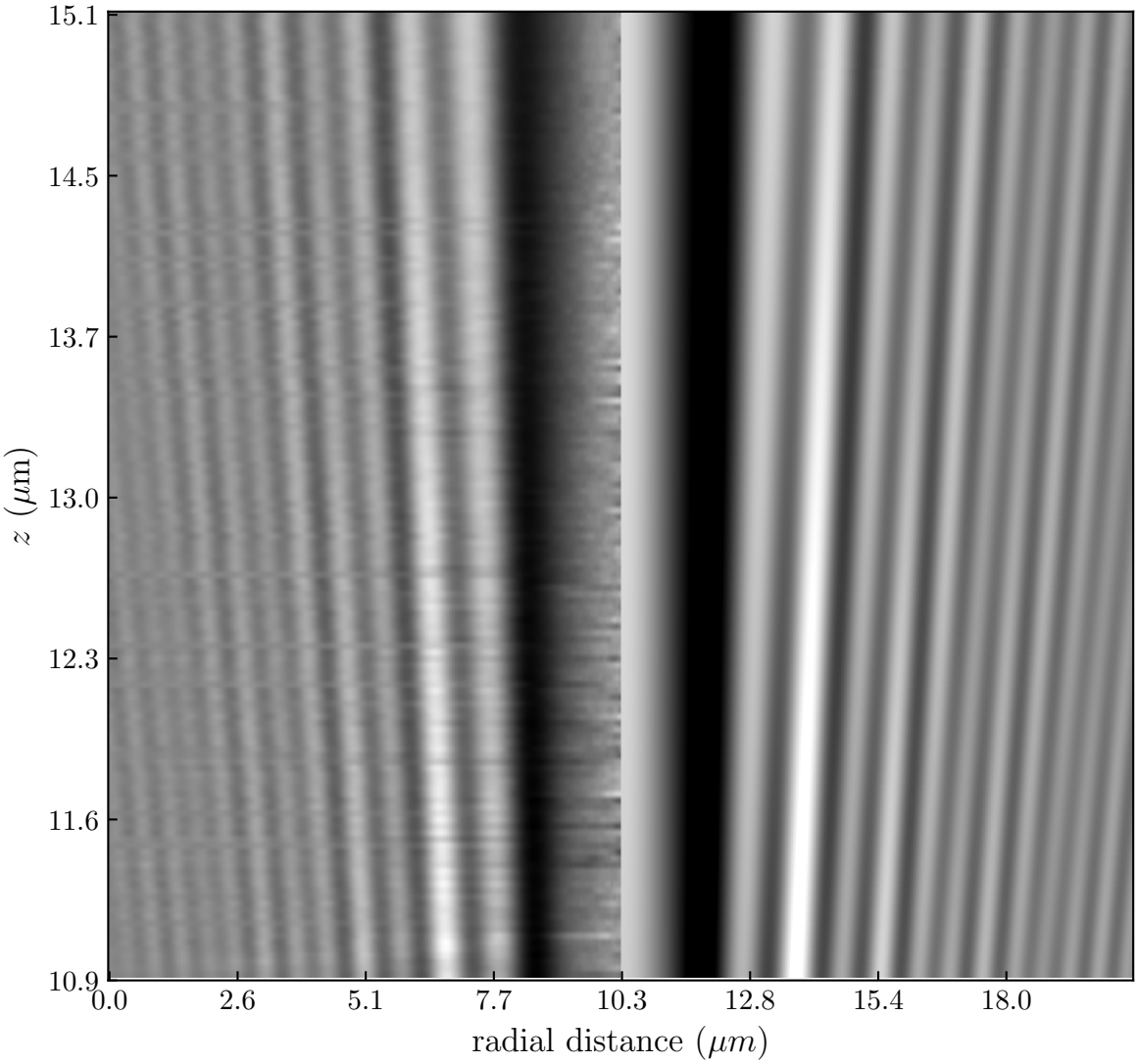


Figure 12: On the left, experimentally measured holograms' radial intensity profile stack, generated from a polystyrene bead of nominal radius $a = 1.5 \pm 0.035$ μm using the experimental setup explained in chapter 3.5. The calibration of this particle radius and optical index is shown in Fig.18. On the right, the corresponding theoretical stack using the result of each individual hologram's fit.

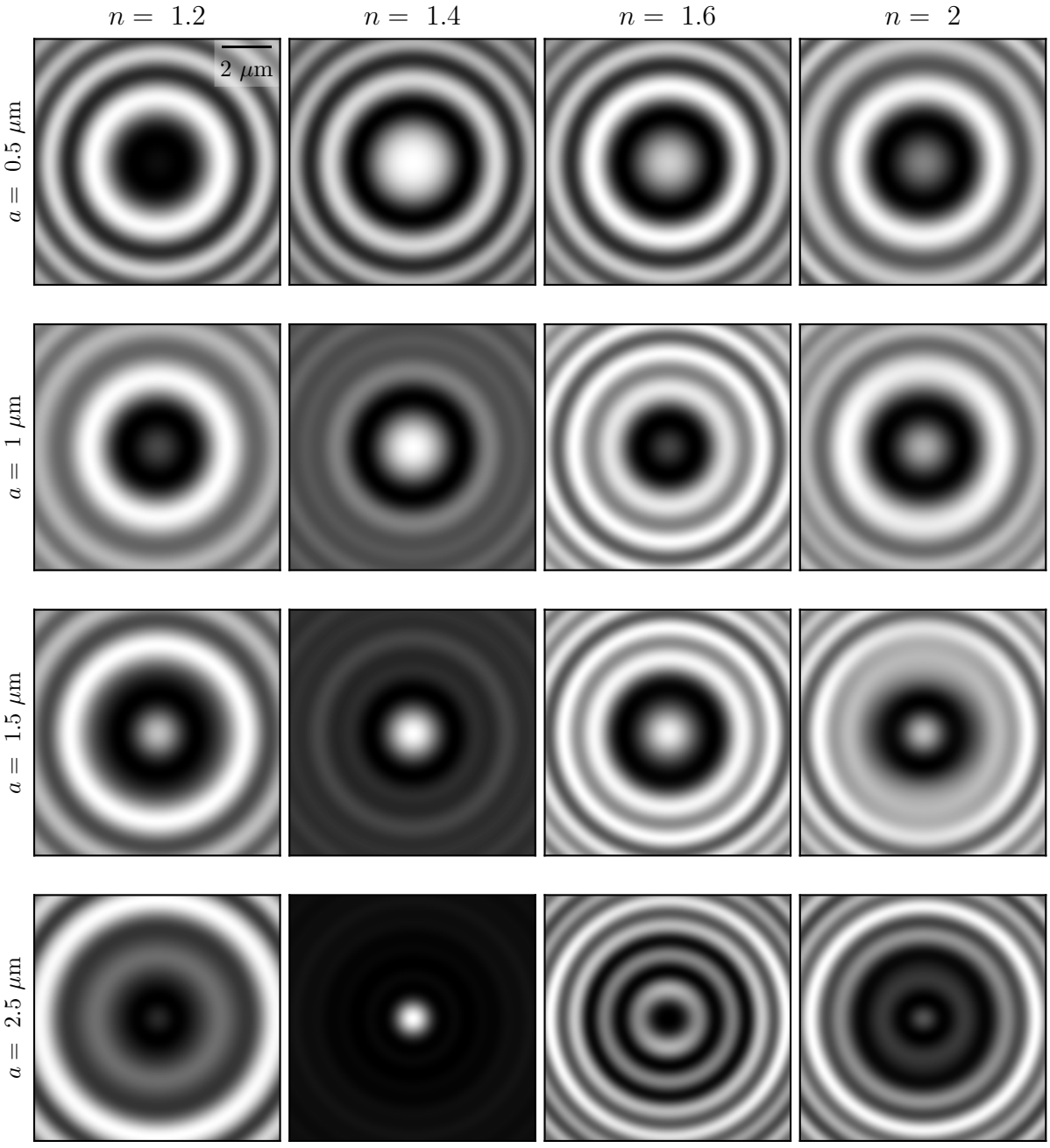


Figure 13: On the left, experimentally measured holograms' radial intensity profile stack, generated from a polystyrene bead of nominal radius $a = 1.5 \pm 0.035$ μm using the experimental setup explained in chapter 3.5. The calibration of this particle radius and optical index is shown in Fig.18. On the right, the corresponding theoretical stack using the result of each individual hologram's fit.

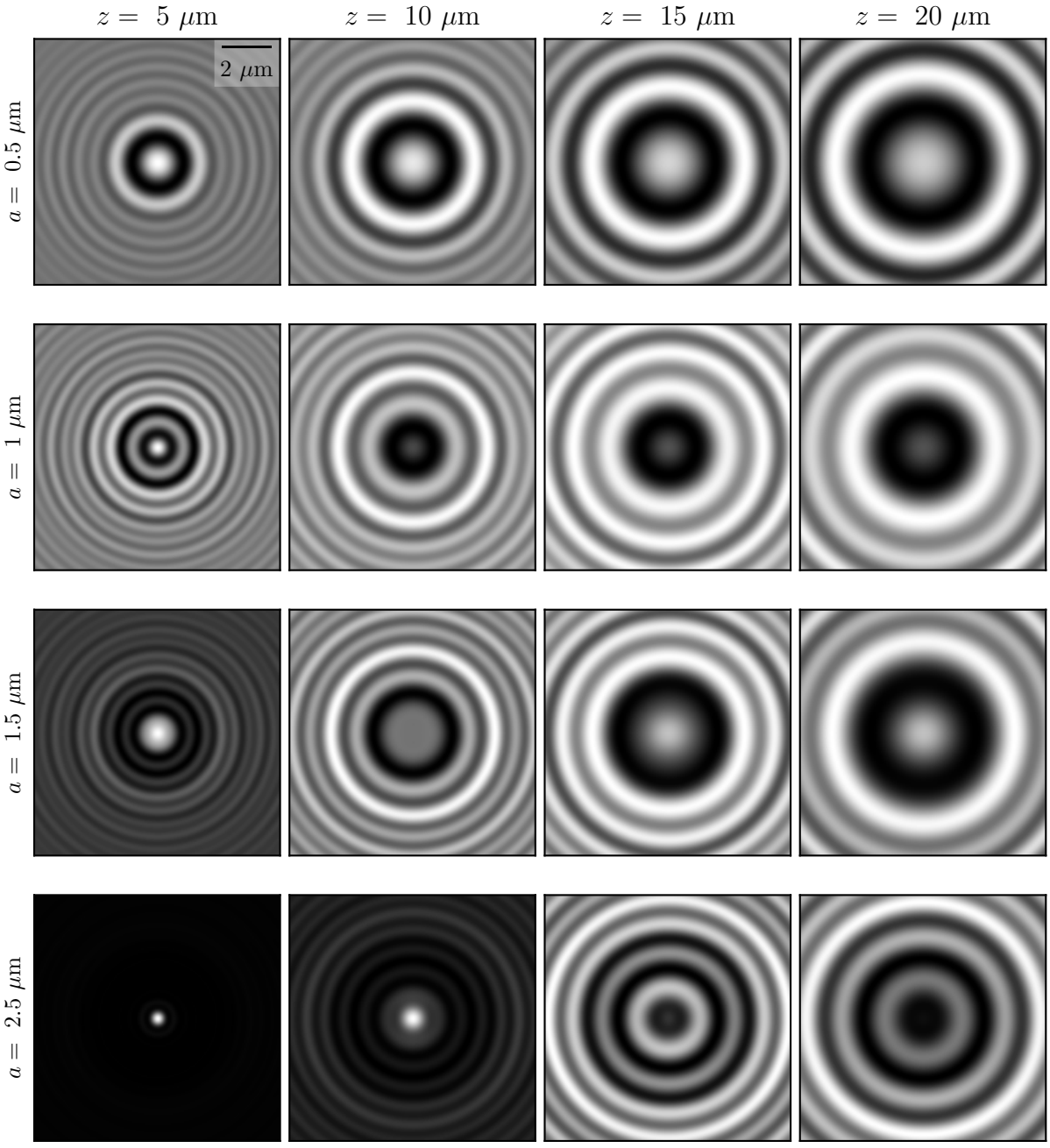


Figure 14: On the left, experimentally measured holograms' radial intensity profile stack, generated from a polystyrene bead of nominal radius $a = 1.5 \pm 0.035$ μm using the experimental setup explained in chapter 3.5. The calibration of this particle radius and optical index is shown in Fig.18. On the right, the corresponding theoretical stack using the result of each individual hologram's fit.

3.4 Rayleigh-Sommerfeld back-propagation

Rayleigh-Sommerfeld back-propagation [61] works on the same principle as the Lorenz-Mie fitting but assumes that we have small scatterers., such as :

$$|\zeta - 1| << 1 \text{ and } ka|\zeta - 1| << 1 \ . \tag{3.4.1}$$

In this case, at the focal plane, the intensity of the scattered field is smaller than the incident field, hence, the term $|\vec{E}_s|^2$ can be ignored. Thus, the normalized intensity, Eq.3.3.4 can be rewritten as:

$$\frac{I(\vec{r})}{I_0(\vec{r})} = 1 + 2\,\mathrm{Re}\left(\frac{E_s(\vec{r},0)}{E_0(\vec{r})}\right) \ . \tag{3.4.2}$$

If one can retrieve completely the scattered field from an image, it is possible to reconstruct it above the focal plane by convolution using the Rayleigh-Sommerfeld propagator [62]:

$$h_{-z}(\vec{r}) = \frac{1}{2\pi}\frac{\partial}{\partial z}\frac{\mathrm{e}^{ikR}}{R} \ , \tag{3.4.3}$$

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

$$E_s(\vec{r},z) = E_z(\vec{r},0) \otimes h_{-z}(\vec{r}) \tag{3.4.4}$$

By using the convolution theorem [62–65] and supposing a uniform illumination, one can write the reconstructed scattered field at a height z as:

$$E_s(\vec{r},z) \approx \frac{\mathrm{e}^{ikz}}{4\pi^2}\int_{-\infty}^{\infty} B(\vec{q})H(\vec{q},-z)\mathrm{e}^{i\vec{q}\cdot\vec{r}}d^2q \ , \tag{3.4.5}$$

where $B(\vec{q})$ is the Fourier transform of I/I_0 and $H(\vec{q},-z)$ is given by

$$H(\vec{q},-z) = \mathrm{e}^{iz\sqrt{k^2-q^2}} \ . \tag{3.4.6}$$

Finally, using Eq.3.4.5 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 Fig.15, we measure the position of the particle. Those equations are way less computational intensive than the Lorenz-Mie function Eq.3.3.5. Thus tracking can be way faster, moreover, Fourier transforms can be largely accelerated using GPU.

Additionally, as the propagator Eq.3.4.3 take 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.3.4.5, one just need to assume that we have spherical colloids. Thus, this method is great to reconstruct the 3D position of a lot of particles or clusters formations.

However, the major drawback is that it is the less precise of the presented measurements and that we can't use it to characterize the particles generating the holograms.

3.4.1 Numerical Rayleigh-Sommerfeld back-propagation

The `holopy` Python module also provide a set of method that permits to user the Rayleigh-Sommerfed 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 and the actual image. Then, one can use the `propagate` method to back-propagate an hologram over a set of height `zstack` 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)
```

Please note that using the `propagate` each propagation will be done by performing a convolution of the reference hologram over the distance to be propagated. However, better reconstruction can be obtained iteratively propagating holograms over short distance. 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 [66]. It can be done by specifying the argument `cfsp` to the `propagate` method. For example, to propagate three steps over each distance, we can use `hp.propagate(holo, zstack, cfsp=3)`.

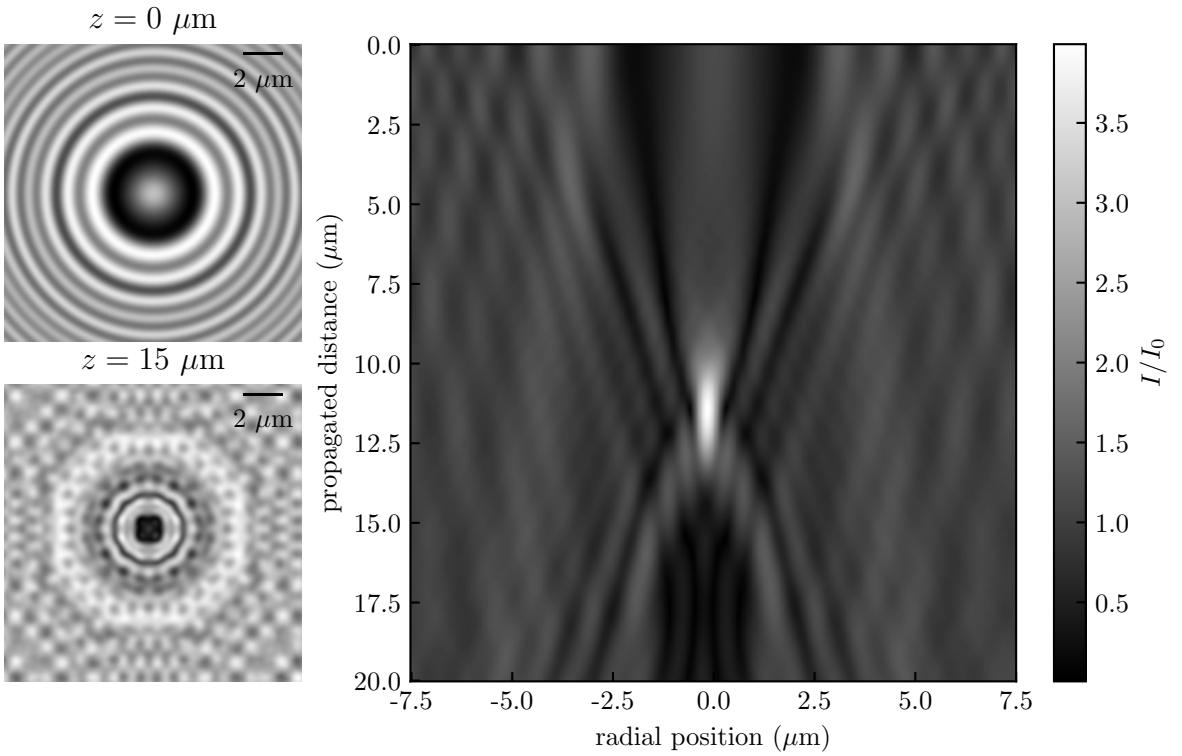


Figure 15: On the left: the original hologram on the top and propagated along 15 μm on the bottom. On the right: Reconstruction using Eq.3.4.5 of the scattered intensity of single colloidal sphere of a radius $a = 1.5$ μm polystyrene spheres, $n_{\mathrm{p}} = 1.59$ in water at a original height of 15 μm. [NEED TO CHECK IF THE PARTICLE HERE IS NOT TO LARGE]

3.5 Experimental setup

The experimental setup I used during my PhD could be employed in order to use both Lorenz-Mie and Rayleigh-Sommerfeld Back progation methods. In order to observe the holograms we use an homemade inverted microscope as shown on the Fig.16 and shematized in Fig.17. This microscope is built using Thor Labs cage system, as it permits doing vertical construction easily. Using this microscope, we observe the holograms coming form the interactions between a laser sourced 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 the evaporation, this permits to reduce 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 viscosity $\eta = 1$ mPa.s. The

particles are made of polystyrene of density $\rho = 1050\text{kg.m}^{-3}$ and optical index $n_\text{p} = 1.598$ for a wavelength of 532 nm.

The sample is illuminated by a collimated laser beam with a 532 nm wavelength. The used laser delivers a power of 4 W and has a centimetric waist. Since the laser is collimated, it has a near zero exentricity such as it can be seen as a plane wave. With this characteristics, we can assume that there is no optical force exerted on the particles [NEED TO CHECK HOW TO EVALUATE THAT]. As depicted in the chapter 3.3, 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.7a)).

The exposure time of the camera is set to $\tau_\text{expo} = 3$ 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 here $2D\tau_\text{expo} < 51.6$ nm.

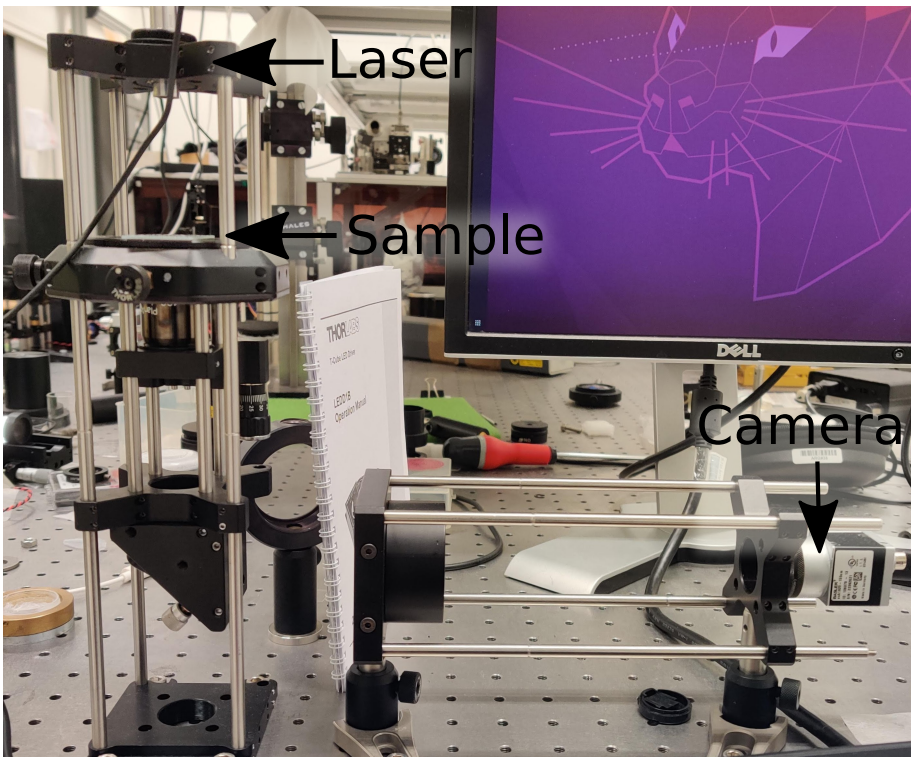


Figure 16: Photo of the custom build microscope used along my thesis. It is mainly composed of Thorlabs cage system. The camera used is a Basler acA1920-155um, we use a x60 magnification and 1.30 numerical aperture oil-immersion objective lens. The light source is a colimated 521 μm wavelength laser.

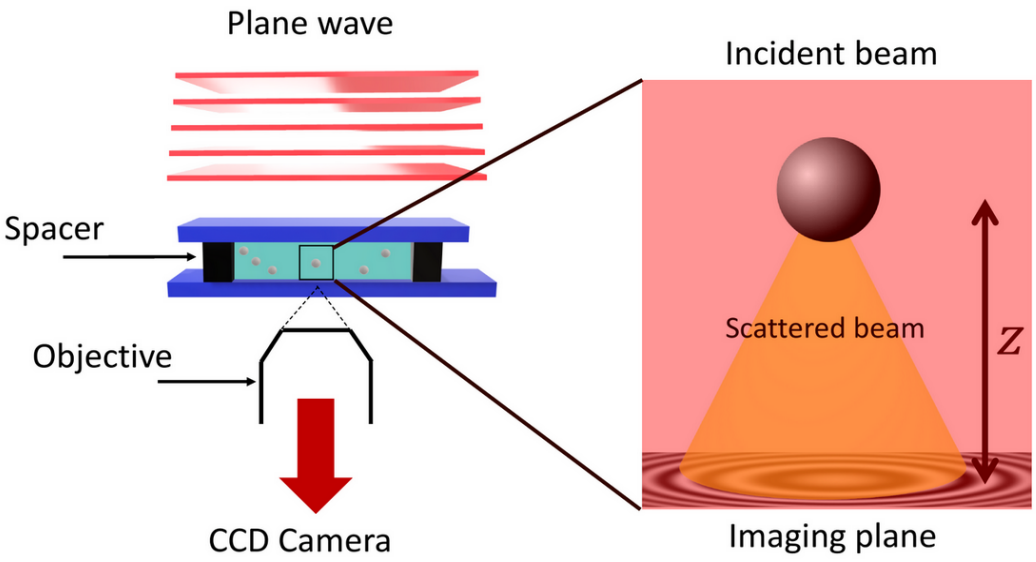


Figure 17: Schematic of the experimental setup. A laser plane wave of intensity I_0 illuminates the chamber containing a dilute suspension of micro-spheres in water. The light scattered by a particle interferes with the incident beam onto the focal plane of an objective lens, that magnifies the interference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

3.6 Experimental pipeline

The method we choose is the Lorenz-Mie fitting method, since this it permits the characterization of single particles. Indeed, since we are interested to fine effects near the surface, we need to know perfectly the radius of the particle we have recorded. This feature also make our all process calibration free, as we don't need to assume any physical properties. An exemple of the pipeline that pemits to track a single particle trajectory is shown in the appendix A.2. In the following, the different steps are the pipe line are going to be described.

3.6.1 Recording the holograms

Since we use a Basler camera, we use the provided Pylon software in order to record the holograms. The latter permits to adjust the parameters of the camera, such as the region of interest (ROI), frames per second (fps) or the opturation time to name a few. Also, movies can be recorded as a time series of images, AVI or MP4 files. AVI files or times series are a great way to save the movies since they are lossless. However, in general we use an ROI of 1000×1000 pixels to record the particle for a long time enough.

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 thus see that image sequences and AVI files are not suitable for our case because of several points: i) at 100 fps one would need 108 GB to store a 30 minutes film, it could lead to TB of data per experiment which was not manageable. ii) it would require a sequential writing speed of 60 MB/s, which is just below the limit of the better HDD. 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 choosed to use the MP4 files 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 will size $\simeq 3$ GB.

3.6.2 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 choosed to use the Pylorenzmie module developed by the Grier's lab at the New York University. Although this module present a lot of capabilities, it is quite cumbersome to use, thus, I developed a wrapper around the module 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. While fitting a series of images, it uses the results of the previous image as the initial fit parameters.

However, as presented in the section 3.3 about the Lorenz-Mie fitting, 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, indeed, if one want to track one trajectory made of 100 000 images one would need to wait a minimum of $\simeq 70$ days; for a series of images that need only a few minutes to be shot experimentally.

When I started my PhD, two groups, the Grier's lab and the Manoharan's lab, had already introduced python packages, respectively, Pylorenzmie and Holopy in order to inverse holograms. They had introduced ways to only fit a set of randomly chosen pixels,

[2] An uppercase B denotes Byte which is equivalent to 8 bits, such as 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 unit in most of the computer architectures. As an example, in binary "LOMA" would be encoded by "01001100 01001111 01001101 01000001"

and, demonstrated that taking only 1 % of the image pixels, could lead to similar precision and improve considerably the fit's execution time [57].

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 wanted to have. Ironically, this part of my project is certainly the one where I spent the most my time, and I certainly learned a lot about code optimization and computer cluster usage.

It is around the half of my thesis, that Pylorenzmie got a new commit on their github repository which was telling that they succeeded on using GPU acceleration using CUDA[3]. This was not an easy task since they needed to reconstruct the Bessel functions in an understandable way for the GPU, fortunately it is possible to do so by using continued fractions [53]. This humongous update permits fitting whole images at a whooping speed improvement of 20 fps. At this speed, we fit the tridimension position of the particle, the radius and optical index.

As a remark, the fits are done by soliving a least square problem using the Levenberg-Marquardt algorithm [67]. This algorithm, is largely used in any curve-fitting applications due to it capabilities to find a minimum even by starting far from it. As mentionned, it is also possible to use various models of Machine Learning or Deep Learning to do the fits [60]. However since we can write analyticaly the holograms the deep learnings models can't be more precise than a standard least-square fitting process. Deep Learning however, could be a great option is one wants to prioritize the computation time over the fit's precision.

Finally, to have a more reliable and fast tracking, we first fit the first 10 000 with $\vec{r}_\mathrm{p}$, a and n_p as free parameters. Using the results of this fit we can characterize the physical properties the observed colloid with high precision. Then, using the later result we can fit all the images with only the particle position $\vec{r}_\mathrm{p}$ as a free parameter.

3.6.3 Radius and optical index characterization

Once the data of the radius and optical index retrieved, the quantity we can look at is the the distribution of measurements. Using 10 000 measurements one can do a 2D histogram of the a and n_p as presented in the fig.18 here smoothed using a Gaussian KDE (Kernel

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

Density Estimator. Using the later using the top 10% measurements couple possibility, we measure that the radius of the observed particle is $a = a = 1.514 \pm 0.003$ μm and its optical index $n_{\mathrm{p}} = 1.585 \pm 0.002$.

Finally, using this measurement of radius and optical index, we fit the whole movie by removing them from the free parameters. Doing so, we have measured the trajectory of the particle as shown in Fig.19 in tridimension for the particle previously characterized.

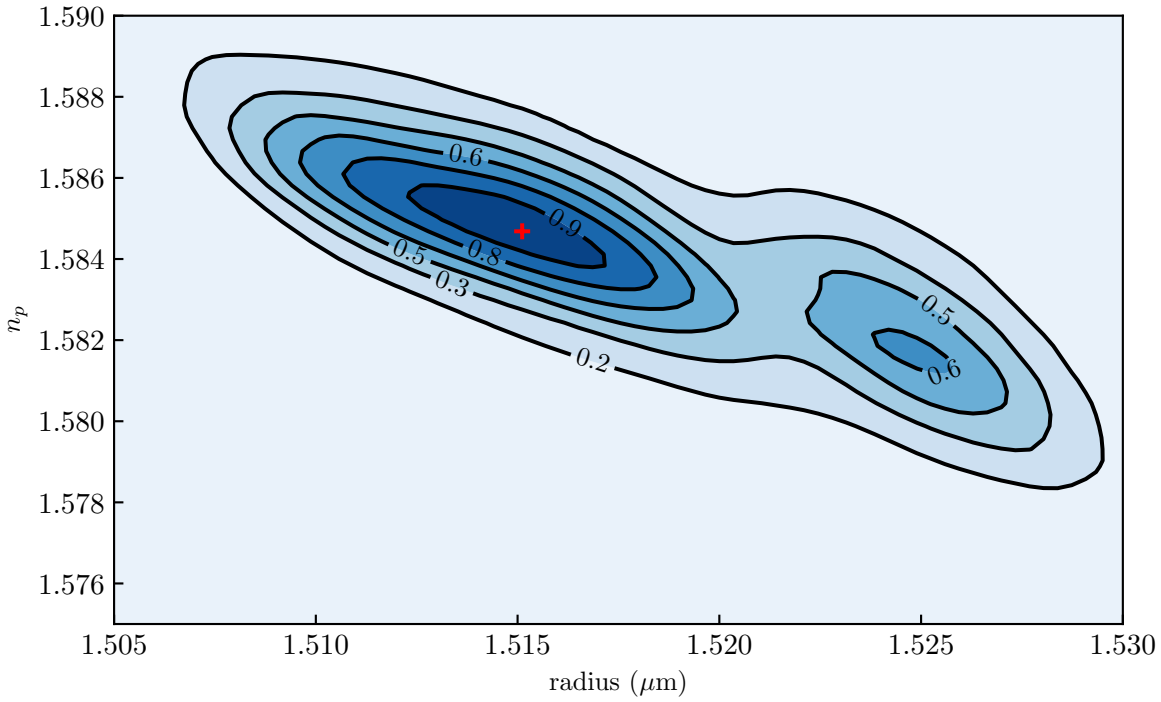


Figure 18: 2D Probability density function of the measurements of the optical index n_{p} and radius a. Black lines indicate iso-probability. Taking the 10% top probability, we measure $n_{\mathrm{p}} = 1.585 \pm 0.002$ and $a = 1.514 \pm 0.003$ μm.

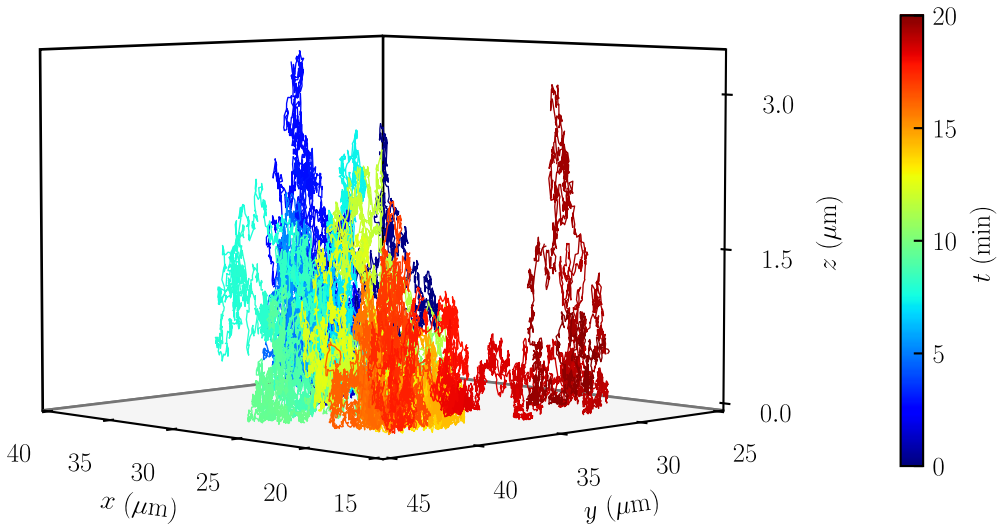


Figure 19: 3D plot of an experimental trajectory measured in water for a particle of optical index $n_{\mathrm{p}} = 1.585$ and radius $a = 1.514$ μm.

3.6.4 Conclusion

In this chapter, we have covered different method that enable the tracking of individual particles. All the different method each come with pros and cons. We decided to use Lorenz-Mie since it is requires no calibration. Then we have shown how we use it in practice, from the experimental setup to the numerical treatment. Also an exemple of the Jupyter notebooks used for the traking can be found in the appendix A.2. We have discussed on how to have fast and accurate fits to retrieve the particle trajectory. To do so, we first characterize fully the particle, namely, its radius and optical index; before tracking a whole movie.

Now that we have an understanding on the tracking of single colloids, we can use the later measured trajectory in order to understand how the Brownian motion is affected in various configurations. In the following, use confined Brownian motion in order to do a stochastic inference of surface-induced effects.

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
```

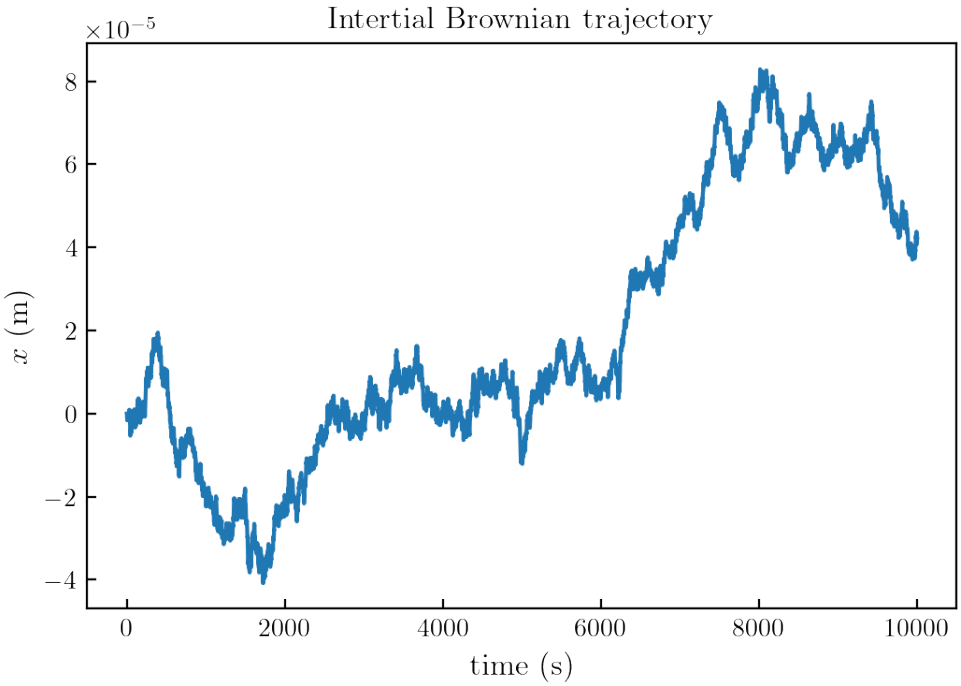
[5]:
```
def xi(xi1, xi2):
    """
    Function that compute the position of a particle using the full Langevin␣
 ↪Equation
    """
    t = tau / tauB
    wi = np.random.normal(0, np.sqrt(tau))
    return (
        (2 + t) / (1 + t) * xi1
        - 1 / (1 + t) * xi2
        + np.sqrt(2 * kbT * gamma)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

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

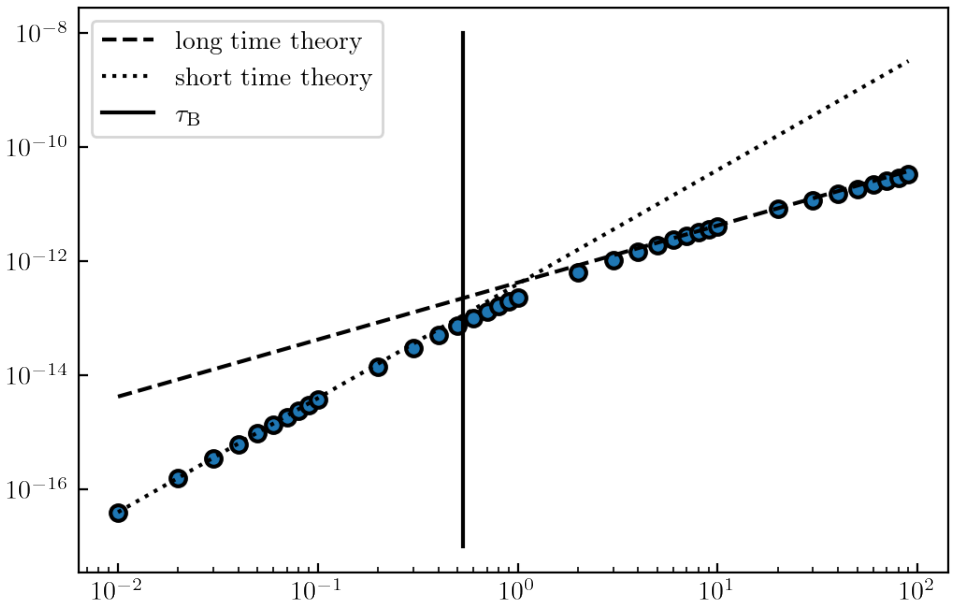
and for $\tau >> \tau_B$:

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

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
```

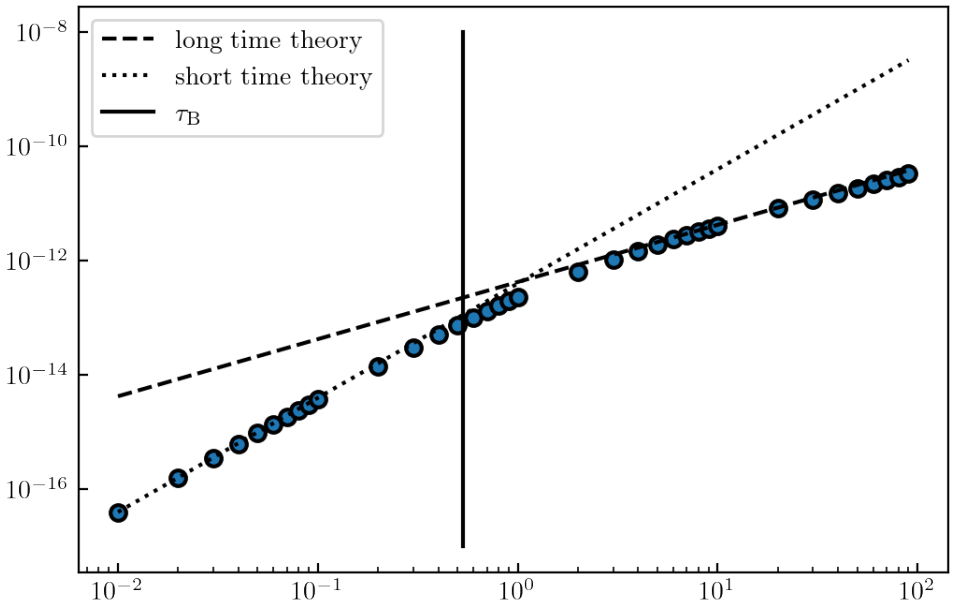
```
def trajectory_cython():

    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
```

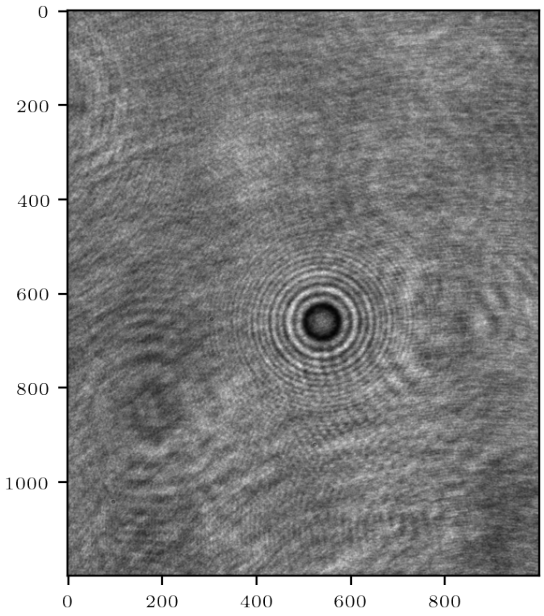
```
No module named 'pylorenzmie.fitting.cython.cminimizers'
```

[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
```

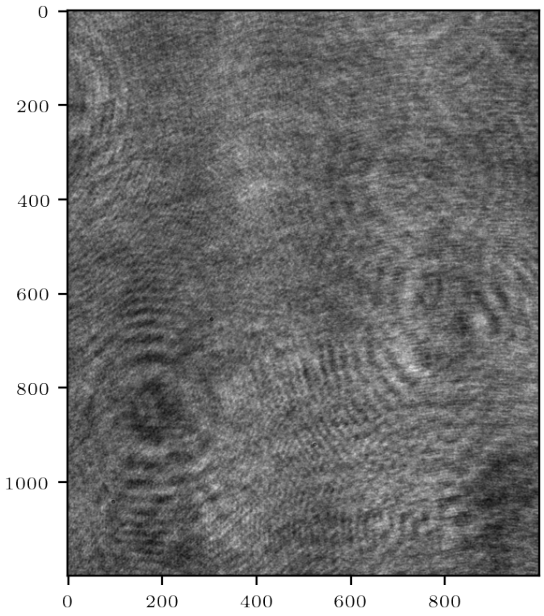
[4]:
```
# We take a look at the first image of the movie
image = vid.get_image(1)
plt.imshow(image,cmap="gray")
```

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



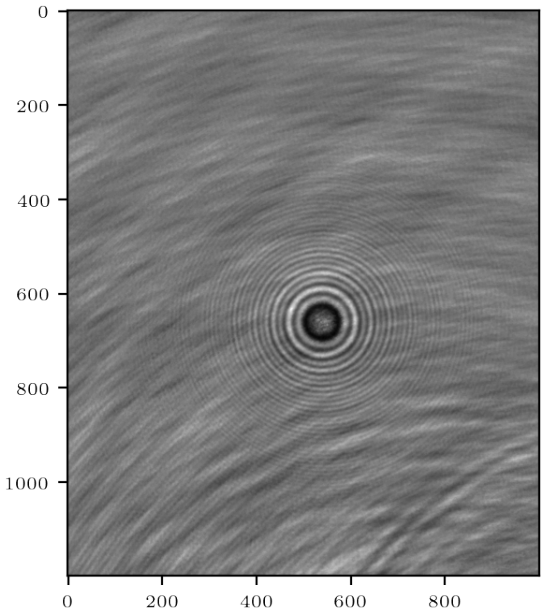
[5]:
```
# set the background image (it can also be computed using vid.
 ↪get_background method)
vid.number = 125000
vid.background = np.array(imageio.imread("background.tiff"))
#vid.background = vid.get_background(n=50) # n is the number of image to␣
 ↪use to compute the background
plt.imshow(vid.background,cmap="gray")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

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

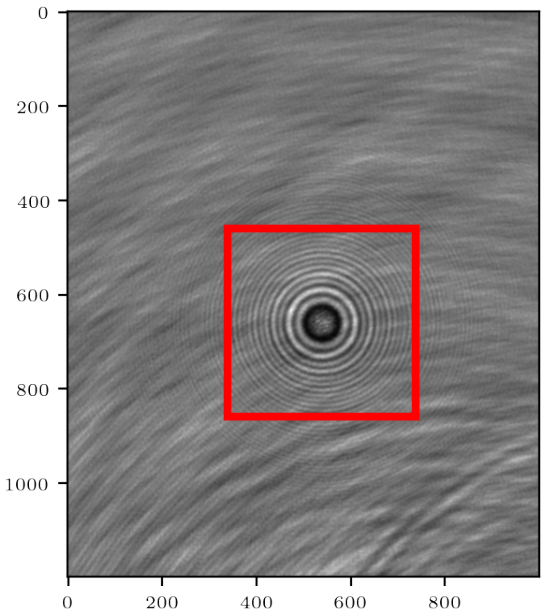


[6]:
```
imageio.imwrite("background.tiff",vid.background)# We save the background␣
↪for possible later use.
```

[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)
```

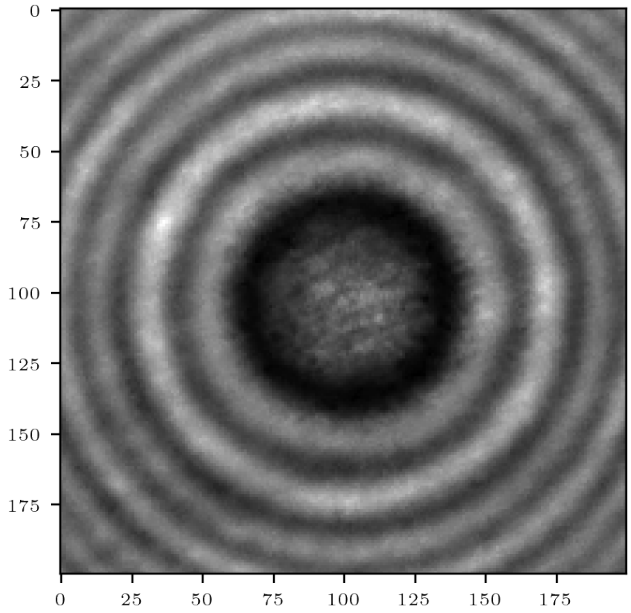


1.1 Fitting the first image

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.

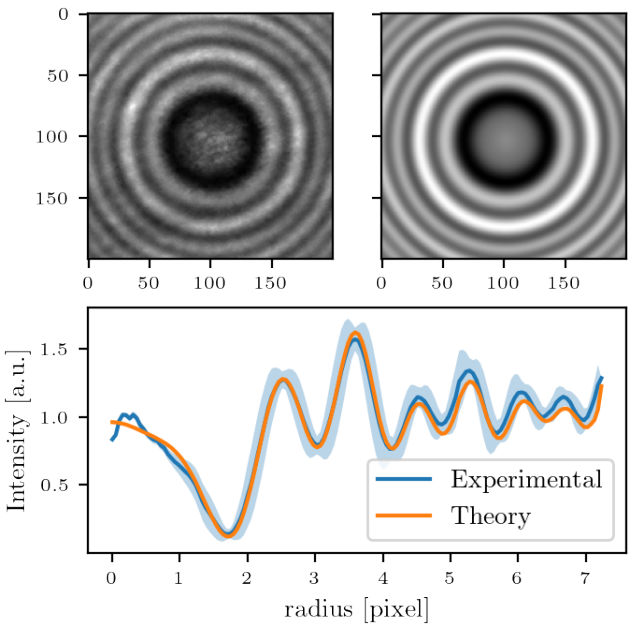
[13]:
```
center = np.array(np.shape(fitter.image))
```

[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`

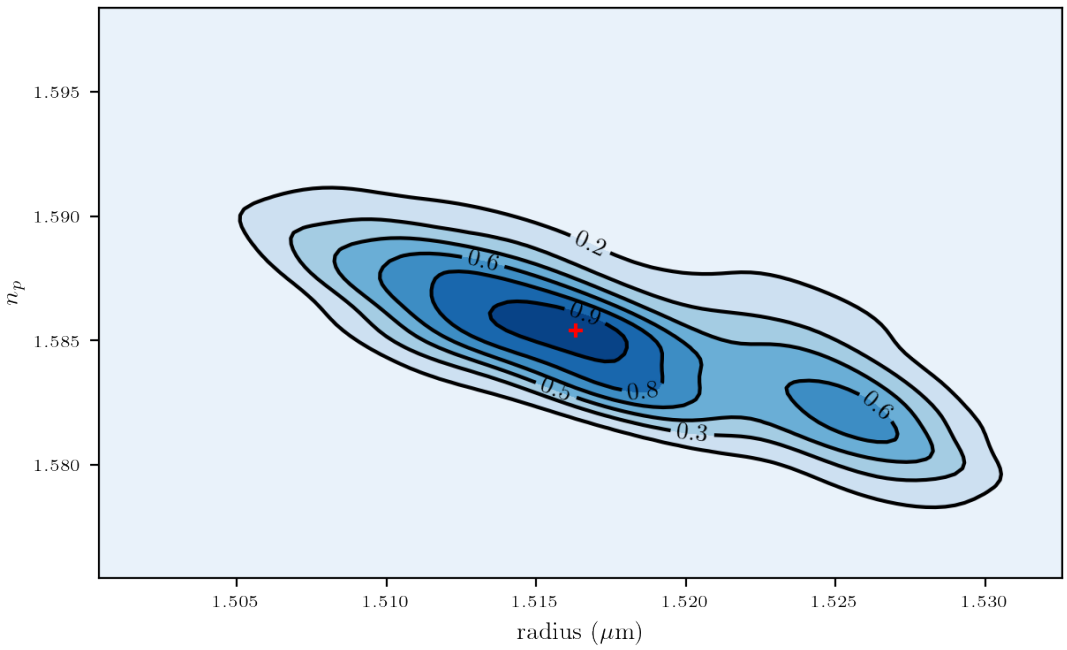
[18]:
```
import numpy as np
import scipy.stats as st
import matplotlib.ticker as ticker

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")
```

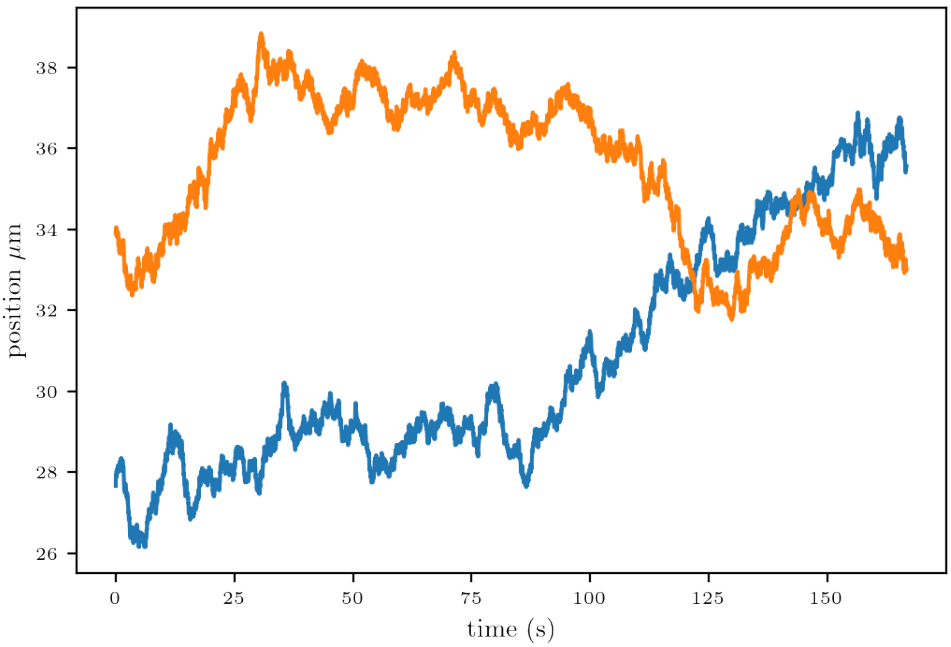
```
100% 9999/9999 [07:24<00:00, 22.47it/s]
```

[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

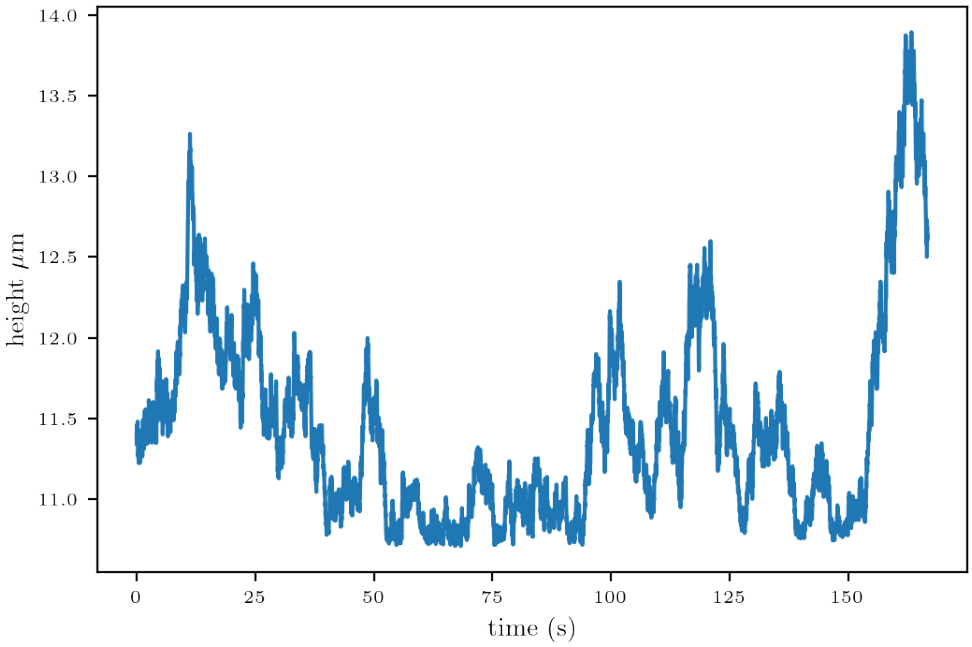
[25]:
```
plt.plot(np.arange(len(z))/60, x)
plt.plot(np.arange(len(z))/60, y)
plt.ylabel("position $\mathrm{\mu m}$")
plt.xlabel("time (s)")
```

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

[33] 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).

[34] 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).

[35] 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).

[36] 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).

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

[38] 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).

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

[40] 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).

[41] 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).

[42] 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).

[43] 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).

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

[45] 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).

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

[47] 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).

[48] 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).

[49] 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).

[50] 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).

[51] 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.

[52] 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).

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

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

[55] 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).

[56] 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).

[57] 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).

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

[59] 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).

[60] 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).

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

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

[63] 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).

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

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

[66] 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).

[67] 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.