POLITECNICO DI TORINO

Master's degree Thesis

Degree of Master of Science in
Mechanical Engineering

Enzyme-powered nanomotors navigating in complex viscoelastic media

Supervisor:

Prof. Matteo Fasano

Co-Supervisor:

Prof. Pietro Asinari

Dr. Rafael Mestre Castillo (IBEC)

Dr. Marco De Corato (IBEC)

Prof. Samuel Sanchez (IBEC)

Candidate:

Chiara Greco

Academic Year. 2019-2020

Ogni forma di felicità e libertà,

se queste realmente le si possa considerare separatamente,

inizia da una forma d'amore.

La mia Tesi la dedico alla forma d'amore più pura

Che la vita ha voluto concedermi:

A mia madre,

per avermi insegnato a sorridere nel buio

ed a ballare nudi difronte la paura.

A mio padre,

perché possa essere fiero di me

almeno la metà di quello che io sono di lui.

Al loro amore raro e sincero,

per avermi dimostrato quanto possa esser dolce e semplice

la vita condivisa con la propria anima gemella.

Abstract

One of the most promising scientific goals that could change the medical field is to find an accurate drug delivery system on the human body, especially for cancer treatment. In the last years, researchers in fields like bioengineering and nanotechnology have designed biocompatible nanomotors for biomedical applications. After the demonstration of the possibility to adopt the self-propulsion by enzyme catalysis, the dynamics of nanoscale swimmers have been studied in simple fluids like water.

This thesis aims to understand the motion of enzyme-based nanoparticles, developed in a chemical laboratory, that navigate in a complex fluid: Hyaluronic Acid (HA). To characterize the motors in HA was necessary to do different steps starting from the particle synthesis. This part is represented by the synthesis of the particles of silica, the ammine functionalization, the addition of PEG (methoxy-polyethylene glycol amine) and at the end the urease enzyme, which allows that motors propel because of the decomposition of urea.

The entire project lasted approximately 6 months and was supported by Smart-Nano-Bio Devices Group from IBEC. The output of this study starts from Dynamic Light Scattering (DLS) measurements, where MSD (Mean Squared Displacement) is extracted, by a Phyton program, to find out if the enzymatic reaction alters the environment and thus the motion of the nanomotors. In this way it was possible to see microscopical changes while the macroscopic ones are studied with Rheometer, to understand the fluid properties, and are seen with the electronic microscope: SEM (Scanning Electron Microscope).

The results of all the experiments led to different conclusions. First, that the synthesis was successful, as supported by z-potential and readings DLS. From these last underlined also that adding PEG avoided extreme aggregation of particles and that the nanomotors with urease enzyme and urea move in water as proved by the diffusion of the MSD.

For the HA it was possible to see an increase in the motion parameters which indicates that there is an enhancement of it. There are also some indications of electrostatic interactions

between the particles and the HA depending on the charge, as supported by the DLS motion experiments.

Now, the work will continue with the experiments to make focus on these results and to correlate them to a real possible motion in a complex fluid. In this way, it will be possible to get close to a possible way to work on a system every time more like human tissue. However, the motion of micro- and nano-motors in complex fluids has not been extensively addressed.

Acknowledgments

The author would like to express his gratitude to IBEC for the opportunity to develop this project. I would like to thank everyone from the **Institute for Bioengineering of Catalonia, SPAGNA,** in particular Rafael Mestre and Marco Corato, my supervisors, for the help and support and Samuel Sanchez, my company tutor.

Special thanks to Prof. Matteo Fasano and prof. Pietro Asinari from Politecnico di Torino for them guidance and advice for the thesis development.

Index

List of Figures

List of tables

List of equations

Introduction

This chapter presents a little description of the institute, the Smart Nano-Bio-Devices Group in particular. After that, there is a little focus on the story of the thesis.

1.1 IBEC

IBEC is the acronym of the **Institute for Bioengineering of Catalonia**. It is an Interdisciplinary research bioengineering institute focused on medical applications.

This institute arose from the Centre of Reference for Biomedical Engineering of Catalonia (CREBEC). CREBEC was comprised of the Biomedical Engineering Research Centre (CREB) of the Technical University of Catalonia (UPC), the Research Centre on Bioelectronics and Nanobioscience (CBEN) of the University of Barcelona (UB) and the then newly created Laboratory of Nanobioengineering Research.

IBEC was established in 2005 and is located at the Barcelona Science Park (Parc Cientìfic de Barcelona – PCB) with the direction of the director of the institute is Prof. Josep Samitier. [1]

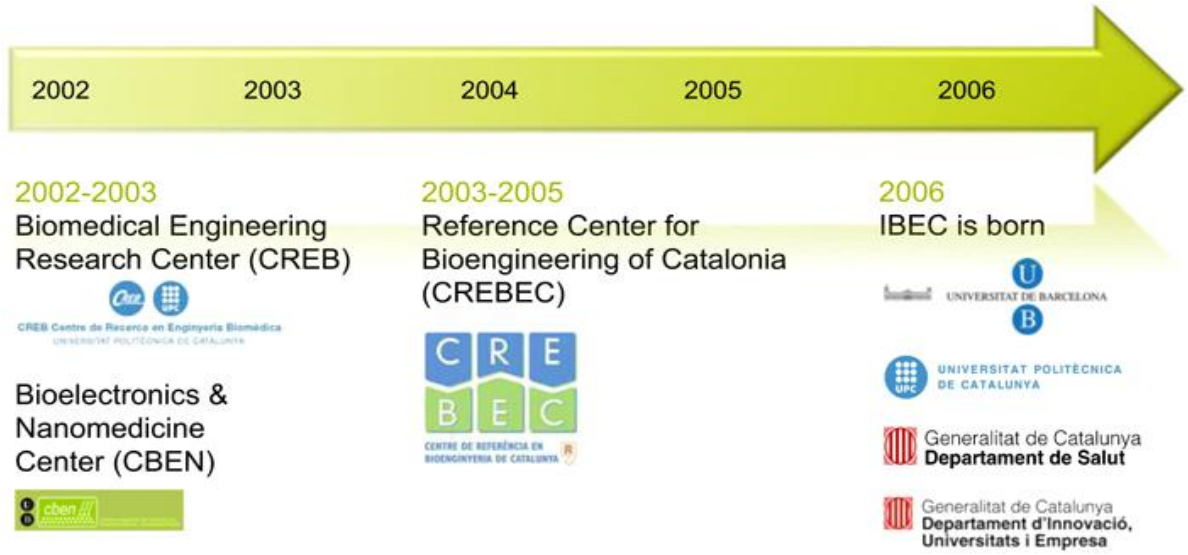

FIGURE 1 -HISTORY OF IBEC. [1]

In 2014 IBEC was named a "Severo Ocha Centre of Excellence" by the Spanish Ministry of Economy and Competitiveness.

IBEC's activities are divided into six research programs:

- **Cellular biotechnology:**
- **Biomechanics and cellular biophysics:**
- **Nanobiotechnology:**

  Biomaterials, implants and tissue engineering:
- **Medical signals and instrumentation:**
- **Robotic and biomedical imaging:**

The institute currently has 22 research groups and more than 250 researchers and staff from 20 different countries.

1.1.1 Smart Nano-bio-devices

The Smart Nano-bio-devices group is now composed of 15 people from different backgrounds. The Group Leader is the ICREA Research Professor: Samuel Sánchez Ordóñez.

The main aim of this group is to develop different Systems ranging from active nanoparticles (nanobots), 3D Bioprinted Actuators and flexible biosensors. These researches are focused on active matter, the use of nanobots for future nanomedicine and environmental applications and the bioengineering of new devices based on hybrid systems. In this group there are different Teams, each one with a topic:

- **NanoBio Team** that demonstrated the use of different enzymes, to generate active propulsion of nano and microparticles, paving the way towards new applications of artificial active matter in biomedicine.

- **3D BioPrinted Soft Robotics Team** that explores the integration of biological tissue and artificial materials at larger length scales with the advantage of 3D bioprinting technique to develop bio-robotic systems.

- **Environmental applications of micro-Nano motors**. Micromotors can remove a wide variety of pollutants from contaminated water. Artificial micromotors, based on bubble self-propulsion have demonstrated to be able to mix solutions and enhance chemical reactions while they swim. These micromotors are mostly based on two main structures, tubular and spherical.

- **(Flexible) Biosensors for non-invasive Point of Care diagnostics.** Point-of-care diagnostics allows decentralizing clinical diagnostic practices and monitoring health out of specialized hospital settings.

1.2 State of the Art

Before focusing on the project is necessarily doing a focus on his state of the art and, for this reason, are now explained all the knowledge and developments from the beginning of the idea of this work-study.

1.2.1 From the Physics of Active Motion to the Enzyme-powered Motors

Thanks to nanotechnology, microtechnology and nanoengineering it is now possible to control the structure and morphology of matter at the nano- and microscale. According to this, a big aim in the science community is the "drug delivery" for which different strategies are being investigated to provide the "activity" into nano- and microstructure. There are different studies and researches about the physics of active motion for micro- and nanoswimmers. In this field, the first goal was to discover the activation that could come from self-propulsion, converting chemical energy into kinetic. To go forward it was necessary to use biocompatible and not toxic fuel to facilitate sustained self-propulsion.

The use of enzymes as a power source has emerged as a response to the need for biocompatible fuels. Due to the high diversity of enzyme types in nature, this system offers unique versatility and the possibility to design specific swimmers that become active on-demand when and where the substrate is present. [2]

Although the field is still in its infancy, a great spectrum of applications has been proposed with functions in sensing, imaging, environmental remediation, nanosurgery, and drug delivery, and several milestones toward them have been achieved, such as facilitated drug transport to cells and tissues. Nonetheless, to fully understand and predict the performance of enzymatic micro- and nano swimmers, a deeper knowledge of the fundamental aspects underlying their motion behaviour is required.

There are now different artificial micro- and nano-swimmers already fabricated with their self-propulsion allowed by the exploitation of free chemical energy from chemical reactions and its conversion to mechanical work. As just said, another power source emerged is the enzymes integration used as catalytic engines. This alternative offer fuel biocompatibility, bioavailability, and versatility. In this view, different enzymes demonstrated their capacity to create active motion and so to break the biomedicine barrier.

Anyway, is not yet understood how fundamental properties like size, dimension, number of enzymes, distribution, etc..., could affect the motion. This problem becomes bigger in the short length scale, where it is difficult to understand whether the motion is generated by random fluctuations or enzymatic activity. So, it is very important to build a solid statistical approach to characterize the tracking of the particles. For all the reasons mentioned it is essential to design an efficient navigation of the enzymatic swimmers. The first prototypes of biocatalytic microswimmers were based on fibers and tubular structures, emerged successful for micro-nanojets, fabricated by Mano and Heller with a carbon fiber and able to self-propel at the water-air interface. [2]

In this view, bubble-free propulsion of silica nanojets powered by urease, by a synthesis approach created to control the enzyme localization. Depending on the enzyme position and nanojet length, different motion dynamics were observed like shown in the “Figure 2”.

Although there are many good results with the use of tubular shapes for biocatalytic self-propulsion of microswimmers, the spherical ones are more developed, because they are simpler to fabricated and more versatile. [2]

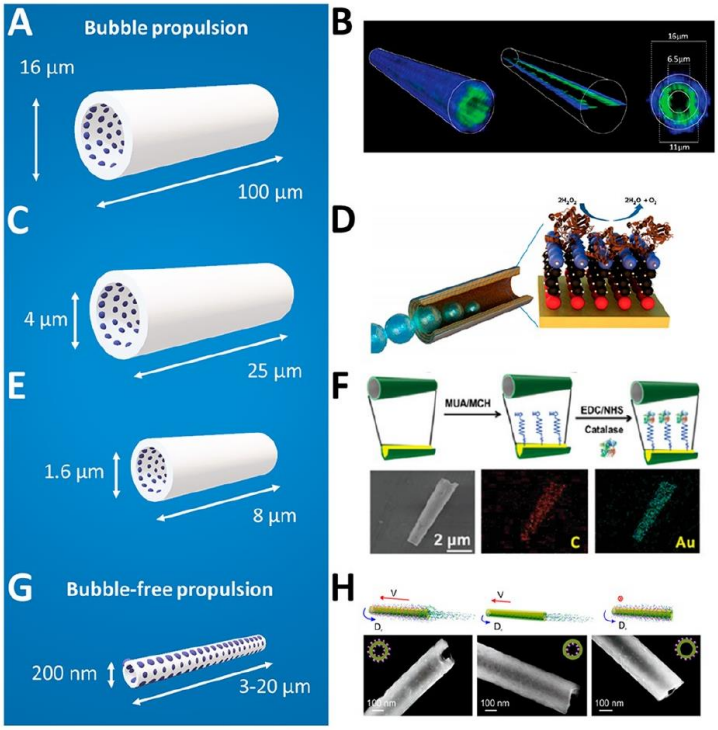

FIGURE 2 - TUBULAR MICRO- AND NANOJETS POWERED BY ENZYMES: (A,B); TUBULAR MICROJETS FABRICATED BY ELECTROSPINNING; (C,D) TI/AU; MICROJET FABRICATED BY ROLLING-UP TECHNOLOGY; (E, F) TUBULAR MICROJETS FABRICATED BY ELECTRODEPOSITION; (G, H) SILICA NANONETS FABRICATED BY TEMPLATE SYNTHESIS, WHERE DIFFERENT ENZYME CONFIGURATIONS WERE EXPLORED. [2]

The diameter of the spherical swimmers determinates the Motion dynamics like shows on the "Figure: 3". This is a schematic representation of micro- and nanoswimmers, where blue dots represent enzyme molecules and underline how the diffusion is more present for nanoparticles (from 100 to 800 nm) than microparticles where exists a ballistic motion.

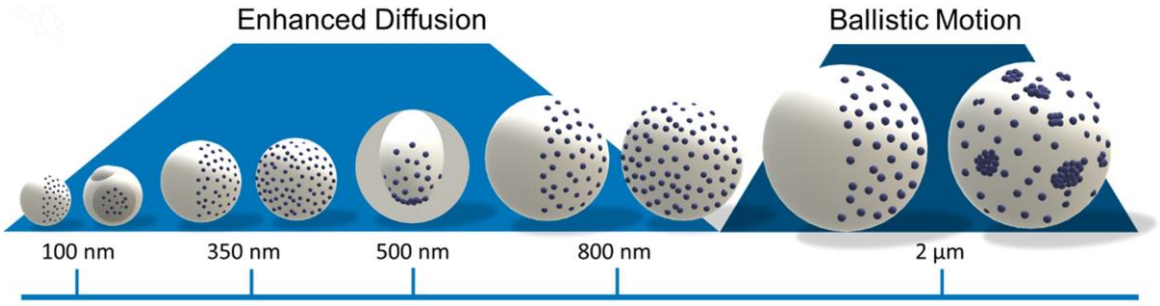

FIGURE 3 - SPHERICAL ENZYME-POWERED MICRO- AND NANOSWIMMERS CLASSIFICATION WITH RESPECT TO THEIR SIZE, STRUCTURE, AND ENZYME DISTRIBUTION. [2]

However, apart from their dimensions is needed an asymmetry to avoid null net forces. A typical approach used to obtain the asymmetric distribution of the enzymes is to create Janus spherical particles where enzyme cover only half particle, used **mesoporous silica particles** efficient for active motion driven by catalase too. These kinds of particles come from the choice of the **Silica** because of its biocompatibility and then was changed its frame becoming **porous** to allow the drug delivery allowing the encapsulation of multiple cargos into the mesoporous hollow.

After designed the particle, the investigation has moved on the biocompatible and non-toxic fuel. First, are presented many different enzymes used as power sources for micro- and nanoswimmers “Figure: 4”.

Among all these enzymes is chosen the urease as the best biological catalyst to power micro- and nanoswimmers by hydrolyzation of **urea** (CO(NH2)2) into carbon dioxide (CO2) and ammonia (NH3) with the reaction: CO(NH2)2+H2O → CO2+NH3. Urea is a natural oxidizing and one of the most promising platforms for future biomedical applications of active micro/nanomachines.

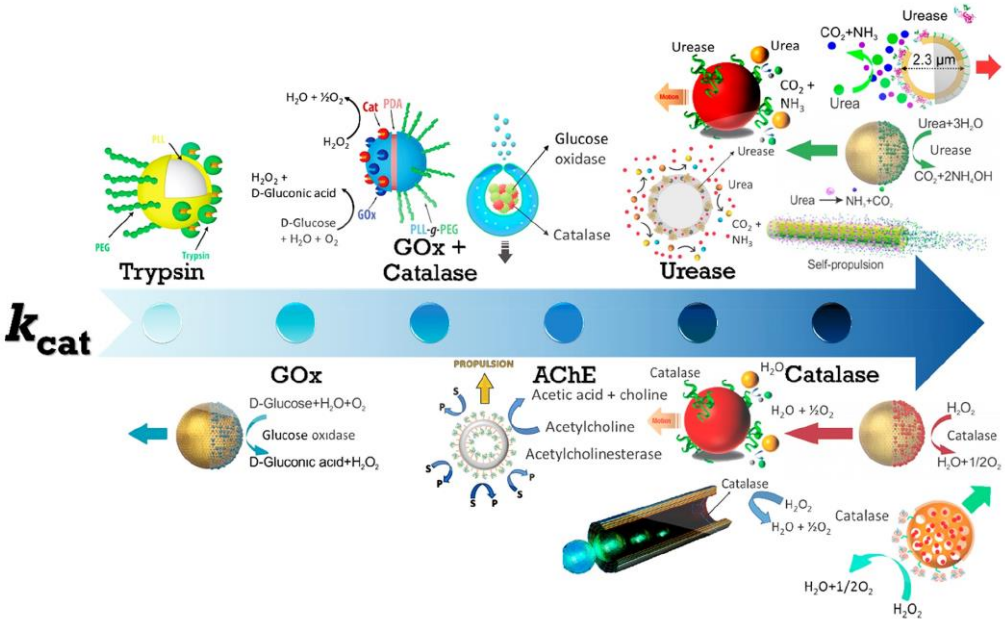

FIGURE 4 - SCHEMATIC OF ENZYMES REPORTED TO POWER MICRO- AND NANOSWIMMERS ARRANGED IN ASCENDING VALUE OF TURNOVER NUMBER (KCAT): TRYPSIN, GLUCOSE OXIDASE (GOX), GOX COUPLED WITH CATALASE, ACETYLCHOLINESTERASE (ACHE), UREASE, AND CATALASE. [2]

The following studies discovered the presence of a force that meant the possibility of movement that implied all the considerations about the motion dynamic analysis. The Samuel Sánchez group did considerations of motion dynamics analysis that showed the following equations. The dynamics of self-propelled particles are upper perpetually analysed by calculating the MSD of their positions over time. By assuming a constant speed over time and randomization of the particle's position and orientation due to Brownian fluctuations, one can obtain the following **MSD** [2]:

$$MSD(\Delta t) = \left\langle \left(\vec{r}(\Delta t) - \vec{r}(0)\right)^2 \right\rangle = 4D_t\Delta t + \frac{v^2\tau_r^2}{2}\left[\frac{2\Delta t}{\tau_r} + e^{-2\Delta t/\tau_r} - 1\right] \quad (1)$$

Where:

- Δt is the time interval considered;
- $\vec{r}(0)$ is the position of the particle at the initial time;
- $\vec{r}(\Delta t)$ is the position of the particle after a time interval Δt;
- D_t is is the translational diffusion coefficient;
- τ_r is the rotational diffusion time;
- v is the speed of the particle.

This equation only applies when particle dynamics are characterized by constant speed and the particle experiences no torques. However, it could be possible to distinguish two different regimes that simplify the equation [2]:

- At longer time scales ($\Delta t \gg \tau_r$) it could be written:

  $$MSD(\Delta t) = (4D_t + v^2\tau_r)\Delta t = 4D_e\Delta t \quad (2)$$

  Where D_e represents the effective diffusion coefficient and which is analogous to the case of a **passive Brownian particle** and is referred to as enhanced diffusion.

- At shorter time scales ($\Delta t \ll \tau_r$) it takes the form:

  $$MSD(\Delta t) = 4D_t + v^2\Delta t^2 \quad (3)$$

  which is called the **propulsive or ballistic regime**, where it would be possible to see an effective directional movement where the particle seems to continuously propel in a specific direction.

These equations are commonly used for the motion analysis of catalytic and biocatalytic (enzymatic) micro- and nanoswimmers, since they can give statistically averaged results.

The MSD is usually obtained by two methods, dynamic light scattering (DLS) and optical tracking. This study has a giant complexity and the analysis to obtain the MSD should be the same for any method to guarantee a correct evaluation of the results. Since real speed of a Brownian it is complex, the instantaneous speed should be used only for comparison. Therefore, for the nano-swimmers the enhanced diffusion coefficient it is the only proper metric to study the motion, while for micro-swimmers, it is possible to perform a quadratic fitting to have an average propulsive speed of the particle. This is a characteristic of the MSD that increase the Δt and therefore the variance of the date due to not enough point. By increasing the time sensitivity and considering just the initial part of the MSD the error is reduced. It is reminded that the assumptions are that the particles are spheres and the motion is at constant speed in a Newtonian fluid. If there are not this condition is necessary to consider the general equation:

$$MSD(\Delta t) \approx t^{\alpha} \tag{4}$$

α is the **scaling factor** that define the diffusivity of the regime as shown in "Figure 5" depending on its value:

- $\alpha = 1$ means a **diffusive regime;**
- $\alpha < 1$ a **subdiffusive regime;**
- $1 < \alpha < 2$ a **superdiffusive regime;**
- $\alpha = 2$ a **propulsive regime.**

All this underlines the importance of time to study the motion of the particle. In fact, for an artificial nanoscale swimmer that uses a chemical reaction catalyzed on its own is demonstrated that in a short time there is a substantial component of directed motion, with velocity depending on the concentration of fuel molecules.

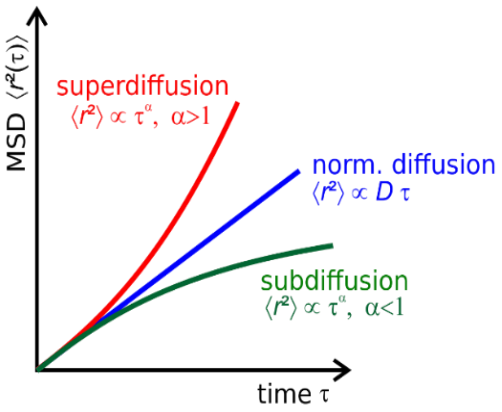

FIGURE 5 - SCALING FACTOR FUNDAMENTAL TO DEFINE THE DIFFUSIVITY OF THE REGIME

In a longer time, the motion reverts to a random walk with an improvement of the Diffusion coefficient. In this way is shown how at short times the particles move predominantly in a directed way, with velocity depending on the concentration of the fuel molecules. While, at longer times, the motion reverts to a random walk, in which runs of directed motion are interrupted by random changes of direction. It is possible too extracts the positional and the rotational diffusion coefficients and show that they are consistent with theoretical predictions, with the rotational diffusion featuring a moderate concentration dependence that could be attributed to directed rotational components in the motion.[3]

To analyse the motion is used the particle tracking and from the records of the particle, the trajectory is calculated the average value of the squared displacement as a function of time. For a purely Brownian particle of radius a, the squared displacement is linear in time with the slope controlled by the particle **Diffusion coefficient**:

$$D = \frac{k_B T}{6\pi\eta a} \quad \left[\frac{cm^2}{sec}\right] \tag{5}$$

where:

- $k_B T$ is the Thermal Energy where k_B is the constant of Boltzmann;
- η is the viscosity of the water.

The stabilization of the diffusion coefficient values in the presence of increasing concentrations of urea can be explained by the activity kinetics of urease.
It is demonstrated how is possible for urease-modified nanobots, composed of mesoporous silica shell, loading the anticancer drug Dox, presenting enhanced drug release profiles dependent on urea concentration. In this way, is been created an active and biocompatible micromotor realized on non-toxic silica with surface chemistry well known, fueled biologically by urea, and with complete motion control. [4]
Another important thing is the particles allows for the transportation of cargo of different sizes by the microcapsule motor to target locations on-demand, leading to novel possibilities for targeted drug delivery in biomedical applications.

Now, to get closer to a real treatment is necessary to work on a similar human body system that means start the study of the enzyme powered motors on a complex fluid. The motion of these particles in non-Newtonian fluids is still developing. [5]

Unlike in Newtonian fluids, the constitutive equation for stress is nonlinear in non-Newtonian fluids and as a result, a straightforward linear decomposition of the flow field into drag and thrust components fails. There is only one relevant report on the motion of enzymatically propelled motors in viscoelastic fluids, in the form of a screw-like motor. It shows how the pH changes due to the enzymatic reaction that changes the environment. However, no emphasis was given on these changes or the effects they could have on spherical nano-motors.

This works now aims to reproduce these particles and to study them into the most similar human tissue complex fluid: hyaluronic acid. This is willingly used in biomedicine because is a biomaterial. This term defines a bioactive material designed to interface with biological systems to guide cell growth or replace any tissue, fluid or organ by mimicking its natural function.

They can partially or completely replace the functions of a living organism, they must possess the required fundamental of biocompatibility, that is, they must not interact harmfully to the body. Biomaterials they are used as devices intended to be implanted in the human body, but also as materials that have contact extracorporeal with tissues or with blood.

2 Materials and Methods

This chapter represents a key one in the economics of this research work. It aims to illustrate in detail the materials on which the research was conducted, the process that allowed to select the elements object of the thesis and in what ways and times the elements of the case studies were treated.

2.1 Chemical Experiment

The chemical part is fundamental in this work because the results of the experiments depend on the chemicals preparations. The aim is to create well-prepared particles in every experiment to be studied well in the DLS.
Let's introduce all the chemical synthesis, the particles functionalization and the preparations of the solutions that are explained in the following chapters:

- *Synthesis of Silica seeds;*
- *Mesoporous Silica synthesis to create: CoreShell particles;*
- *Amine functionalization to create CoreShell-NH_2 particles;*
- *PEG functionalization to create: CoreShell-PEG particles;*
- *PEG-Urease functionalization to create: CoreShell-PEG-Ur particles;*
- *Urea preparation in the concentration of 100mM and 50mM;*
- *Hyaluronic Acid preparation*

2.1.1 Silica Seeds

The creation of silica seeds starts in a laboratory and is the first element of all the other experiments of this work. Silica was chosen for this case study because of its biocompatibility.

2.1.1.1 What is the Silica?

The first element of the experiments is Silicon dioxide, known as silica. The chemical formula of this oxide of silicon is SiO_2.

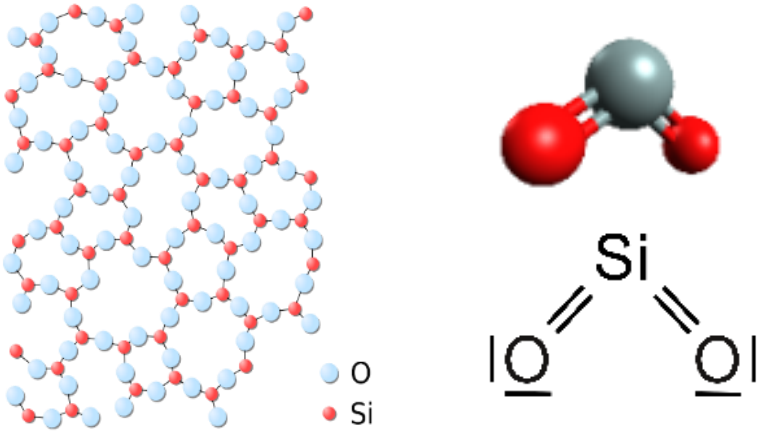

FIGURE 6 - SILICA

2.1.1.2 Protocol of SILICA SEEDS

1. **Purpose:** Synthesis of 100/120nm Silica seeds.
2. **Materials, Machines and Chemicals:**

TABLE 1 - EXPERIMENTAL INSTRUMENTS OF SILICA SEEDS

Materials	Machines	Chemicals
round -bottom flask with 3 neck	Centrifuge 5424	EtOH 99%
round -bottom flask support	Agitator TOC-X5	H_2O mq
Stirrer Bar	hot plate/stirrer	Ammonias Hydroxide NH_4OH
Eppendorf tubes		TEOS
Micropipettes		

3. **Methods:**

This synthesis starts putting into a **rounded flask with 3 neck** (all in 20mL of H2O total volume): a stirrer bar, 7mL EtOH 99% with a glass micropipette, 10 mL H2O and 2mL Ammonias Hydroxide NH4OH. After that, put in a hot plate/stirrer (600rpm for 15mins), add 125µL of TEOS dropwise with a filter micropipette, stir for 21h and collect

in Eppendorf. At the end wash 3 times in EtOH (Centrifuge 5424: 3500 rpm for 3:30 min) and leave in Et0H.

4. **Conservation:**

   Long.

2.1.2 Mesoporous Silica synthesis

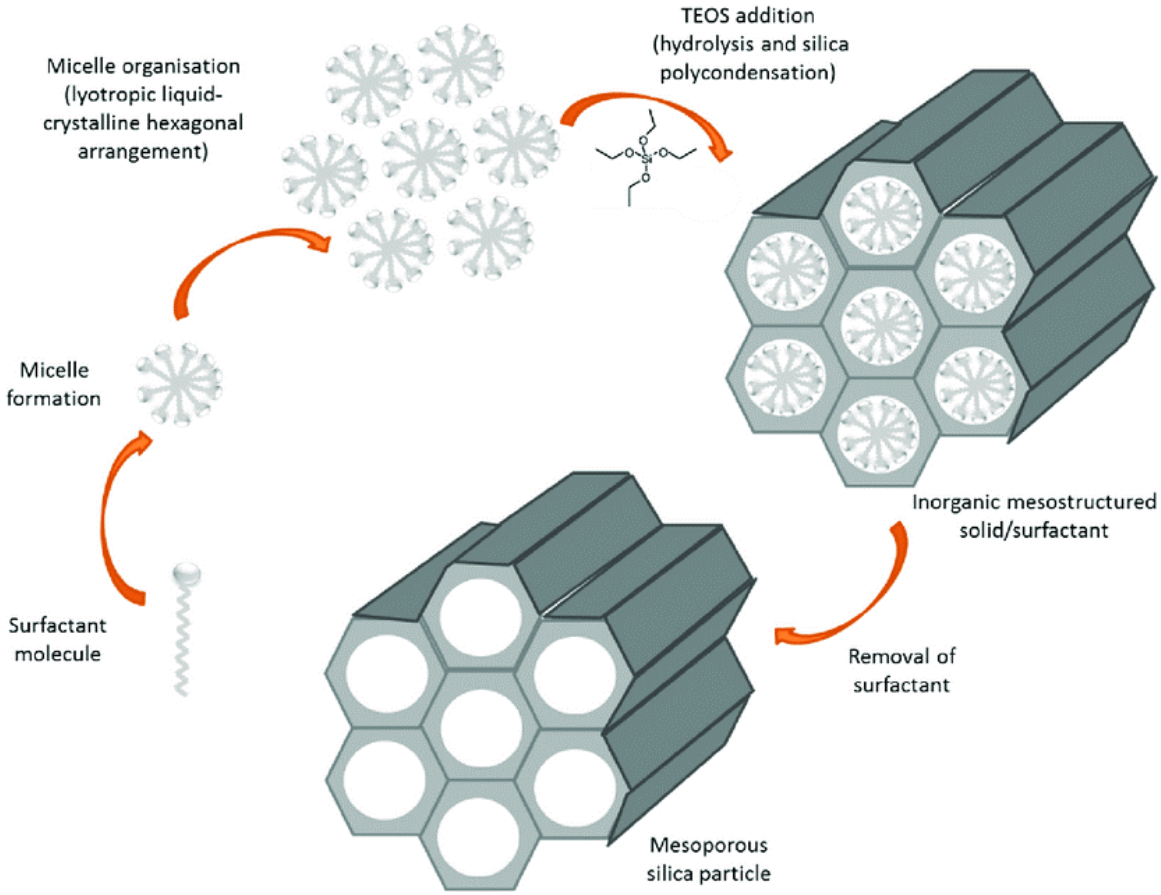

FIGURE 7 - SCHEMATIC PROCESS OF MESOPOROUS SILICA [6]

This synthesis aims to create a mesoporous structure and to fix it with the synthesis. This is useful to allocate chemical functions in the nanometric space. In this way, in the future, the particles could deliver soluble drugs.

2.1.2.1 Protocol of Silica Synthesis: Core Shell 400nm

1. **Purpose:** this process is a COVERAGE for a silica already synthesized: the growth of more silica around the nucleus to become more porous. In the future, this will be done to put drugs or medicines in the pores.
2. **Materials, Machines and Chemicals:**

TABLE 2 - EXPERIMENTAL INSTRUMENTS OF SILICA SYNTHESIS

Materials	Machines	Chemicals
Micropipettes and Pasteur pipette	Fume Hoods (Flores Valles)	Silica SEEDS: 250 nm in EtOH (53,60 mg/mL)
Round-bottom flask support	ultrasonic cleaner	H_2O mq
Stirrer Bars	Agitator TOC-X5	TEOA (Triethanolamine)
Eppendorf tubes	Synthesizer	CTAB
rounded bottom flask with 3 neck	Centrifuge 5424	Methanol
Beaker and Round-bottom flask	Sartorius Analytical Balance	Ethanol: EtOH
Graduated Cylinder		

3. **Methods:**

3.1 synthesis:

First put in a rounded flask with 3 neck (all in 20mL of H2O total volume): a stirrer bar, 60mg Silica seeds transformed in volume with a check of the concentration, 40 mg TEOA, and 75 mg CTAB. Then, put at 80° C in the synthesizer: hot plate/stirrer, waiting until it stabilized at 80°C, immersed in acid. At 80°C add 125µL of TEOS dropwise and stir for 2h at this temperature. Collect in Eppendorf (from a 20mL tot Vol to almost 10 tubes) and wash a time in water (Centrifuge 5424: 3000 rpm for 2:30 min). Then, re-suspend in Methanol in 30mL solution (working in Fume Hoods), putting into a flask: a stirrer bar, resulted mL of particles (10mL) took from each Eppendorf, left mL of Methanol (20mL) and 1.8 mL HCl (37% pure). Stir at 80° C in the synthesizer for 24h.

3.2 Removal of CTAB:

Working in Fume Hoods (Flores Valles), put solution from the flask to 8 Eppendorf with almost 2mL of solution for each one. Wash with Ethanol X3 (Centrifuge 5424: 3000 rpm for 2:30 min) and with H_20 X3 (Centrifuge 5424: 3000 rpm for 2:30 min). Leave in Et0H.

4. **Conservation:**

   long

2.1.3 Amine Functionalization

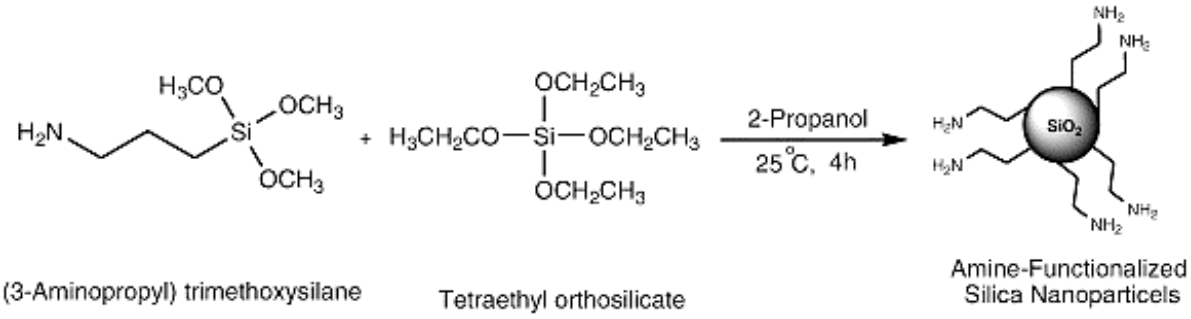

FIGURE 8 - AMINE FUNCTIONALIZATION

The functionalization with amine of the particles of silica is a necessary step to create a urease molecule.

Protocol of Amine functionalization: CoreShell 400nm-NH_2

1. **Purpose:** Add amine.
2. **Materials, Machines and Chemicals:**

TABLE 3 - EXPERIMENTAL INSTRUMENT OF AMINE FUNCTIONALIZATION

Materials	Machines	Chemicals
Eppendorf tubes (1.5-2 mL)	Centrifuge 5424	Core Shell 400nm particles
Becker (5-20mL)	Agitator TOC-X5	H_20
Micropipettes	Thermomixer Comfort	APTES (3-Aminopropyl)
Stirrer Bars	Synthesizer	PBS
syringe		Ethanol: EtOH

3. **Functionalization:** from mesoporous silica particles already synthesized is necessary to add amine to, then, attached the urease enzyme. Wash Core-Shell 400nm particles (4 Eppendorf) with H20 for three times (Centrifuge 5424: 3500rpm, 1150rcf, for 3:30min) and introduce all in a 5mL Becker. After calculating the concentration of the particles, calculate the particles volume in the function of measured concentration

(Ci*Vi = Cf*Vf). Then, take a 20mL Becker and put inside: a stirrer Bar, the necessary calculated particles volume taken from the 5mL Becker with a micropipette, the remaining water volume to have 5mL solution, subtracting the particles volume already introduced (5000µL - 710µL = 4290µL of water) and 30 µL of prepared reactive solution. To prepare the reactive solution put in an Eppendorf: 100 µL of reactive: APTES (3-Aminopropyl) taken with a syringe and 900 µL of water.

Close the Becker with parafilm and it on the Hot Plate (necessary to warm an acid) at 50°C and 140rpm, leaving for 15h minimum.

After all, insert 1000 µL of 5mL solution in 5 Eppendorf, wash with PBS X3 (Centrifuge 5424: 3500rpm, 1150rcf, for 3:30min) and leave in EtOH.

4. **Conservation:**

   2 weeks.

2.1.4 PEG Functionalization

Polyethylene glycol is a polyether used to many applications, especially medicine. In each laboratory is presented in different types and this experiment is used the Methoxy-Polyethylene glycol amine in 1mg/mL concentration.

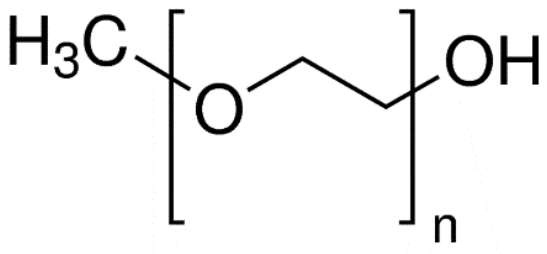

FIGURE 9 - METHOXY-POLYETHYLENE GLYCOL AMINE

This functionalization is necessary to reduce the aggregation of the particles.

2.1.5.1 PEG stock preparation

Before the functionalization is necessary to create a PEG stock:

1. **Purpose:** Create a PEG stock necessary for urease functionalization.

2. **Materials, Machines and Chemicals:**

TABLE 4 - EXPERIMENTAL INSTRUMENT OF PEG STOCK PREPARATION

Materials	Machines	Chemicals
5mL Becker	precision balance Sartorius CPA623S	PEG (Methoxy Polyethylene glycol amine)
spatula		PBS (1X)
Micropipette		

3. **1mg/mL Solution Preparation:**

   Put in a 5mL Becker: 3mg PEG weigh in Balance Sartorius CPA623S by a spatula and 3mL PBS with a micropipette.

4. **Conservation:**

   A day

2.1.5.2 Protocol of PEG Functionalization: CoreShell 400nm-PEG

1. **Purpose:** Add PEG to Core Shell particles.
2. **Materials, Machines and Chemicals:**

TABLE 5 - EXPERIMENTAL INSTRUMENT OF PEG FUNCTIONALIZATION

Materials	Machines	Chemicals
Eppendorf tubes	Centrifuge 5424	Core Shell 400nm (1mg/mL)
spatulas	Noria DLAB MX-RD-E	H_20 and PBS(1X)
Micropipettes	balance Sartorius CP224S	Glutaraldehyde 25%
		PEG stock (1mg/mL)

3. **Functionalization:**

   take a Core Shell 400nm (1mg/mL) Eppendorf and wash it with H_20 3 times (Centrifuge 5424: 3500rpm, 1150rcf, for 3:30min); at last wash don't introduce water but: 100 µL of Glutaraldehyde 25% and 900 µL of PBS. Then, put on gravitation rotation in Noria DLAB MX-RD-E for 2:30h. After, wash with PBS 3 times (Centrifuge 5424: 3500rpm, 1150rcf, for 3:30min); at last wash don't introduce only PBS but 1mL solution

calculated: 3 µL PEG and 997 µL PBS. Leave overnight rotating in Noria and the day after, wash in H_20 3 times (Centrifuge 5424: 3500rpm, 1150rcf, for 3:30min).

3. **Conservation:**
   - Time: A day in the refrigerator.

In "Figure 10" is shown the construction of the particles step by step:

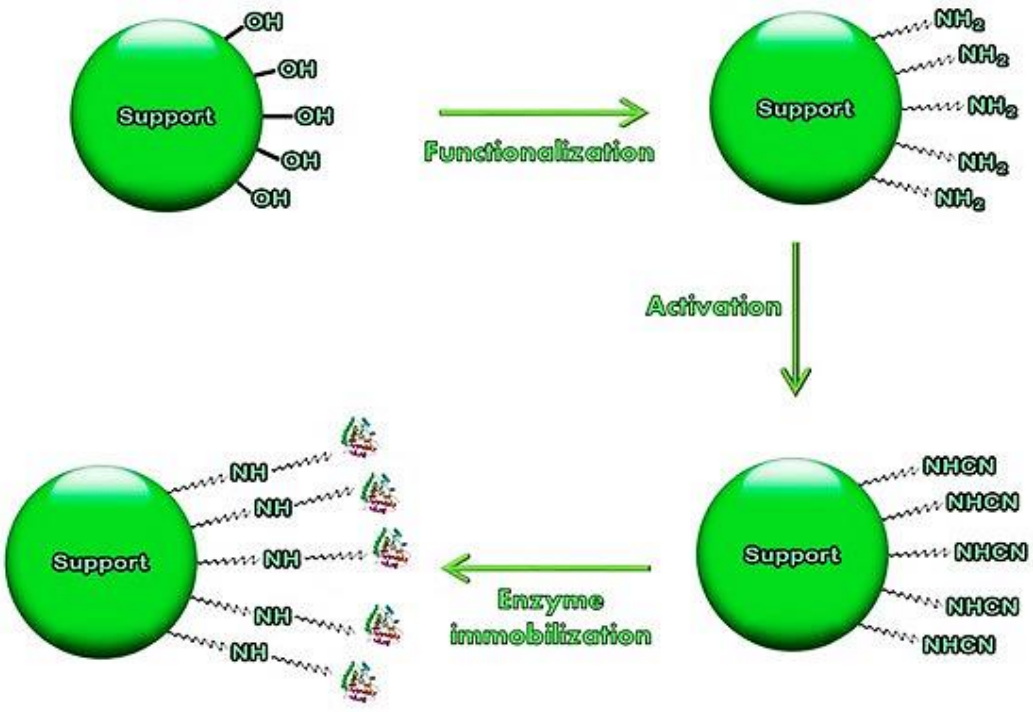

FIGURE 10 - CONSTRUCTION OF THE PARTICLES

2.1.5 PEG – Urease Functionalization

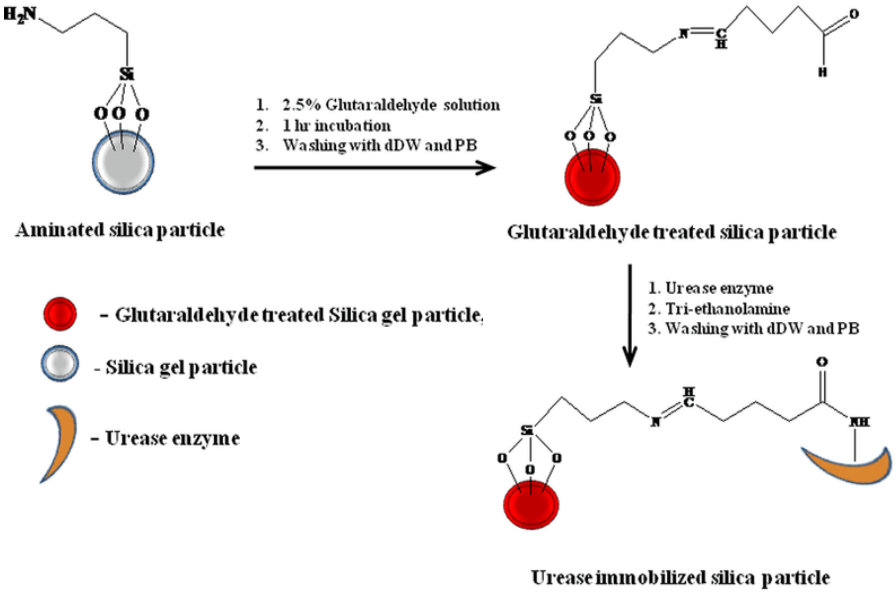

FIGURE 11 - UREASE FUNCTIONALIZATION

This enzyme catalyzed the urea hydrolysis into carbon dioxide and ammonia:

$$(NH_2)_2CO + H_2O \xrightleftharpoons{\text{Ureasi}} 2NH_3 \uparrow + CO_2$$

$$NH_3 + H^+ \rightleftharpoons NH_4$$

FIGURE 12 - UREASE REACTION

The urease enzyme is already studied, and its behaviour is understandable on the motion of the particles. The crystal structure of the urease is presented in “Figure 13”:

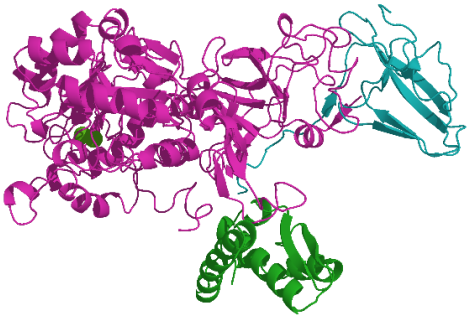

FIGURE 13 - UREASE ENZYME

The urease functionalization is un-useful without the PEG addition because of the aggregation of the particles.

Protocol of Core Shell 400nm-PEG-Ur

1. **Purpose:** Add PEG to Core Shell – NH2 particles to introduce urease enzyme later.
2. **Materials, Machines and Chemicals:**

TABLE 6 - EXPERIMENTAL INSTRUMENT OF PEG FUNCTIONALIZATION

Materials	Machines	Chemicals
Eppendorfs	precision balance Sartorius CP224S	Core Shell 400nm– NH2 particles
spatulas	Noria DLAB MX-RD-E	H_20
Micropipettes	Centrifuge 5424	Glutaraldehyde 25%
		PBS(1X)
		PEG stock (1mg/mL)
		urease

1. **Functionalization:**

   Take a Core Shell 400nm (1mg/mL) Eppendorf and wash it with H_20 X3 (Centrifuge 5424: 3500rpm, 1150rcf, for 3:30min); at last wash don't introduce water but: 100 µL of Glutaraldehyde 25% and 900 µL of PBS (1X). Put on gravitation rotation in Noria DLAB MX-RD-E for 2:30h and meanwhile prepare a "**PEG STOCK**" chose a concentration (1mg/mL) and wash with PBS X3 (Centrifuge 5424: 3500rpm, 1150rcf, for 3:30min); at last wash don't introduce only PBS but the urease Solution, put in an Eppendorf: 3mg urease weight on the precision balance Sartorius CP224S, 3 µg/mL PEG solution calculated: Vi*Ci = Vf*Cf (3 µL PEG and 997 µL PBS).

   Put the urease Solution in the Previous Eppendorf and leave overnight rotating in Noria DLAB MX-RD-E. At the end wash with H_20 X3 (Centrifuge 5424: 3500rpm, 1150rcf, for 3:30min).

3. **Conservation:**
   - A day in the refrigerator.

Below are shown the solvents preparations used for each experiment: water, urea (in different concentrations) and the complex fluid: hyaluronic acid.

2.1.6 Urea Preparation

Urea is an organic compound that has an important role in the metabolism of nitrogen-containing compounds by animals and is the main nitrogen-containing substance in the urine of mammals.

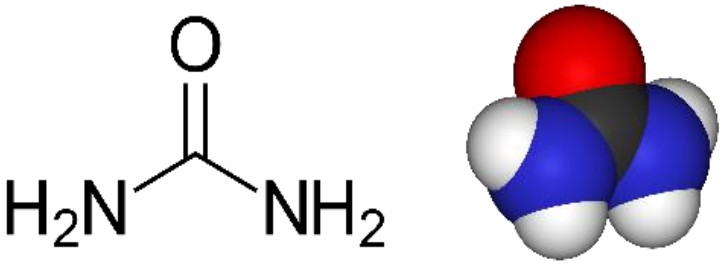

FIGURE 14 – UREA

Is used a lot in laboratory for is denaturant power that disrupts noncovalent bonds in the proteins, a property that increase the solubility of some proteins.

The urea is added to the solution because the urease enzyme catalyse the hydrolysis of urea into carbon dioxide and ammonia.

2.1.7.1 100mM Urea Preparation

1. **Purpose:** Obtain a 200mM of urea final concentration (X2)
2. **Materials, Machines and Chemicals:**

TABLE 7 - 200MM UREA PREPARATION

Materials	Machines	Chemicals
20 mL Becker	precision balance Sartorius CPA623S	urea
spatula		H_20
5mL Micropipette		

3. **Solution Preparation:**

a. **Dates:**
   - Final Concentration: Cf = 200mM (double concentration to have 1mM);
   - urea molecular weight: PM = 60,06 g/mol;
   - Final Volume: Vf = 15 mL;
   - Solvent: H_20.

b. **How many urea grams weigh?**
   - Molarity = mol/L = Cf = 200mM = 200µmol/mL = $200*10^{-6}$ mol/mL;
   - Grams = Cf * PM * Vf = $200*10^{-6}$ mol/mL * 60,06 g/mol * 15 mL = 0,18g;

c. **100mM urea Solution:**

   Knowing the urea grams put in a 20mL Becker:
   - 0,18g urea weigh in Balance Sartorius CPA623S by a spatula;
   - 15mL H_20, taken by a 5mL Micropipette.

   Obtaining a 6mg/mL Concentration.

4. **Conservation:**

   1/2 days.

2.1.7.2 50mM urea Preparation

1. **Purpose:** Obtain a 100mM of urea final concentration (X2)
2. **Materials, Machines and Chemicals:**

   On the same way of 2.7.1:

3. **Solution Preparation:**

   a. **50mM urea Solution:**

   Knowing the urea grams put in a 20mL Becker:
   - 0,09g urea weigh in Balance Sartorius CPA623S by a spatula;
   - 15mL H_20, taken by a 5ml micropipette.

4. **Conservation:**

   1/2 days.

2.1.7 Hyaluronic Acid

Hyaluronic acid is an anionic, no sulphated, glycosaminoglycan, chosen like the complex media in these experiments because is distributed widely throughout connective, epithelial, and neural tissues so good to replicate the human body in this work.

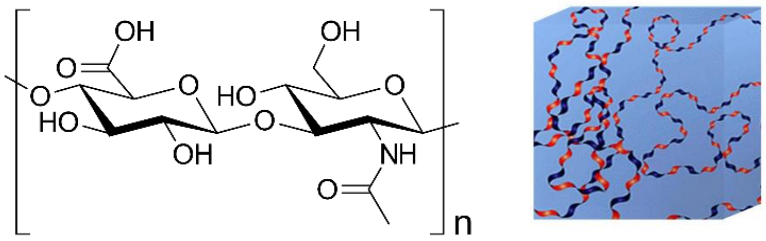

FIGURE 15 - HYALURONIC ACID MOLECULE ON THE LEFT AND THREE-DIMENSIONAL STRUCTURE ON THE RIGHT

Hylauronic Acid stock preparation

1. **Purpose:** Preparing an 2mg/mL Hylauronic Acid stock
2. **Materials, Machines and Chemicals:**

TABLE 8 - EXPERIMENTAL INSTRUMENT OF HYLAURONIC ACID PREPARATION

Materials	Machines	Chemicals
200 mL Becker	precision balance Sartorius CPA623S	Hylauronic Acid
Plastic spatula	stirrer	H_20
Plastic container		

3. **Solution Preparation:**
   a. **Dates**
      - Final Concentration: Cf = 2mg/mL (double concentration);
      - Final Volume: Vf = 100 mL;
      - Solvent: H_20.
   b. **Hylauronic Acid Solution:**

      Put in a 200mL Becker: 100mL H_20, a stirrer bar, 200mg hyaluronic acid weigh in Balance by a plastic spatula in a plastic container. Leave overnight in a stirrer.
4. **Conservation:**

   In refrigerator. Put out from refrigerator before to use it.

2.2 DLS Experiment

This is a fundamental Experimental part of the project where the DLS is the machine is used to determine the dimensions of the particles thanks to the thermal energy of the suspension particles. In this way is possible to detect the movements of the particles.

2.2.1 What is DLS?

Dynamic Light Scattering (DLS), also known as Quasi-Elastic Light Scattering (QELS) or Photon Correlation Spectroscopy (PCS), is a widely applied method to measure the size of particles dispersed in a liquid. DLS determines the Brownian motion of the particles suspended in the liquid. Embedded probe particle microrheology methods are based on DLS, a non-invasive technique for electrophoretic mobility and size measurements of nanoparticles. On the DLS measurement, every exposure is lighted up by a laser and the variation of the light intensity is measurement depending on a function of the time. These intensity variations are generated by the Brownian movement of the particles from the scattering start. For the same temperature and viscosity, the movement of "small" particles is fast like them intensity variation of scattering, while the movement of "large" particles is slow, like the variation too.

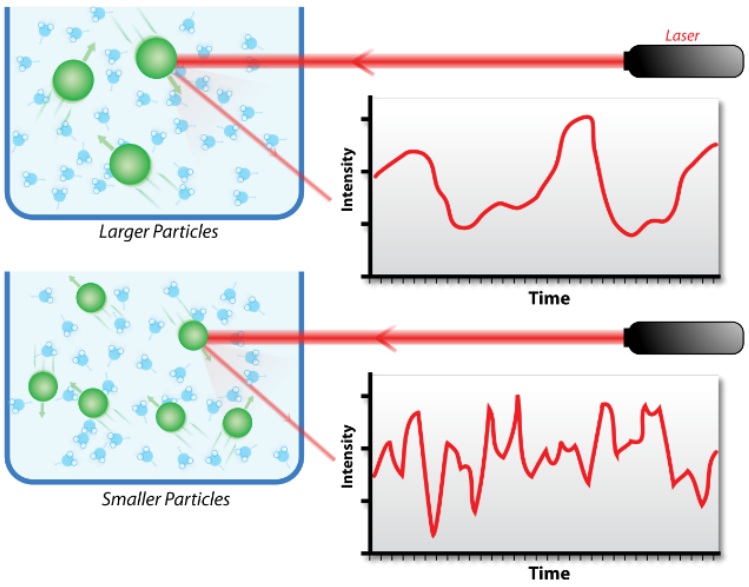

FIGURE 16 - DYNAMIC LIGHT SCATTERING

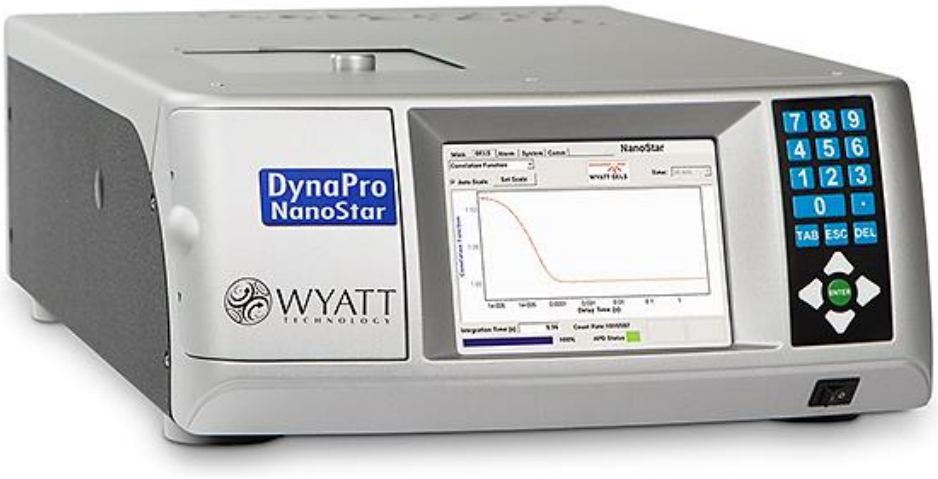

FIGURE 17 - DLS INSTRUMENT [3]

Thanks to autocorrelation, the velocity of intensity variation is given, and the diffusion coefficient of the particles calculated by the correlation function. Then, the **Stokes-Einstein equation** converts the diffusion coefficient D into a hydrodynamic diameter. On the other hand, the Zeta potential measurement is necessary to predict the stability of dispersion and electrostatic interactions.

In this work is used the DLS Machine: "Mobius" from the Wyatt Technology "Figure 17". This instrument is defined as *reliable*, *reproducible*, and *non-destructive* electrophoretic mobility measurements and is also compatible with DYNAMICS software that acquires and analyses the size, molar mass, and zeta potential. The software provides to calculate the percentage of the samples to create a cumulative size distribution.

The DLS Instrument is provided of a Kit for every kind of experiments. In this work are used the Disposable Cuvettes and the Mobius Dip Cell for the determination of the Zeta Potential:

FIGURE 18 – DLS KIT: DISPOSABLE CUVETTES ON THE LEFT AND THE MOBIUS DIP CELL ON THE RIGHT. [3]

2.2.2 Introduction to DLS physics

Motion and measurements of the particles are described by correlations function that are related by the **Siegert Relationship**:

$$G_2(\tau) = B[1 + \beta|g_1(\tau)|^2] \quad (6)$$

Where:

- $G_2(\tau)$ is the **intensity correlation function** that describes particle motion measuring change in the scattering intensity;
- $G_1(\tau)$ is the **electric field correlation function** that describes measured fluctuations: the correlated particle movements;
- B is the baseline;
- β is an instrumental response.

The size of the particle is calculated from the **decay constant** and experimentally determined:

$$\Gamma = -Dq^2 \quad (7)$$

where q^2 reflects the distance the particle travels and is defined as the **scattering wave factor**:

$$\left(q = \frac{4\pi n}{\lambda} sin\left(\frac{\theta}{2}\right)\right) \quad (8)$$

Where:

- n is the refractive index of the medium;
- λ is the wavelength of the incident beam;
- θ is the angle at which the photodetector is placed in relation with the incident beam.

The diffusivity is determinate by D that is the **diffusion coefficient** of the particles written as on the Equation (5) so, the inverse equation is possible to calculate the radius:

$$r = \frac{k_B T}{6\pi\eta D} \quad [cm] \quad (9)$$

2.2.3 Protocol of the DLS

For each exposure is used the same method to create a common protocol to detect better the differences.

This method is proved on the water first to search for the best way to experiment.

Each Experiment starts from a type of particles that is called for simplicity EPP.A:

1) **CoreShell 400nm particles [2.1.2.1]**;
2) **CoreShell 400nm-NH2 particles [2.1.3.1]** (trying to understand how the molecular charge can change the experiment because the CoreShell charge is negative while with -NH2 became positive. In this way could be differences in the H.A.);
3) **CoreShell 400nm-PEG particles [2.1.5.2]**;
4) **CoreShell 400nm-PEG-Ur particles [2.1.6.1]**.

For anyone of them is only necessary a few microlitres of particles, so is more than enough an Eppendorf for each, in 1mg/mL concentration.

Before starting a DLS experiment is necessary for washing the Eppendorf considered in the water on the centrifuge.

Take off from the refrigerator the Hylauronic acid [2.1.8.1] necessary for the experiments and have ready beakers of H_2O, 100mM urea [2.1.7.1] and 50mM urea [2.1.7.2].

Every exposure is studied with different solvents that are, called EPP.B:

a) **H_2O;**
b) **H_2O and urea 100mM;**
c) **H_2O and urea 50mM;**
d) **hyaluronic acid and H_2O;**
e) **Hyaluronic acid and urea 100mM;**
f) **Hyaluronic acid and urea 50mM.**

For each exposure is considered 3 measurement mixing by:

- **Only vortex**;
- **Vortex and sonication**.

FIGURE 19 – ULTRASONIC CLEANER FROM SAMARTH ELECTRONICS ON THE LEFT AND VORTEX AGITATOR FROM VWR COLLECTION ON THE RIGHT.

The mix is very important for nanoparticles to obtain a good result and the sonication does a better job than the vortex, using sound energy to agitate particles rather than only mix them. But the sonication is not allowed for the hyaluronic acid so is been created a single methodology repeatable for each exposure.

2.2.4 The method

To find the most suitable method was necessary to try a lot the protocol, perfectionated in water and then, focusing on the better ones in the meaning of concentration, time and type of mix. It is now presented the discovered method used to have the results studied.

First is necessary to prepare the DLS before starting and then work in the same way for each exposure. Every experiment presents two Eppendorf explained in the preview chapter: EPP.A and EPP.B. Before starting, sonicate the particle of 1mg/mL in the EPP.A for four minutes (maintaining the temperature of the sonicator between 22-27°C and, meanwhile, prepare the 1,5mL of solvent on the EPP.B. Introduce 15µL of the particle into the EPP.B to have a 10µg/mL solution. Then, vortex the mix for ten seconds and put into a CUBETTE being careful to not create bubbles.

Put into the DLS and wait that appears on it a sense curve and run the program. This last step takes almost twenty seconds when there is not H.A. and forty when there it is.

For each RUN is chosen to do seven measurements each of on average of three.

2.2.5 Phyton Program

Is been created a Phyton Program (visible in Annex A) that work on Newtonian system. In a Newtonian system, the Stokes-Einstein relationship can be used to represent the diffusion coefficient as a function of the viscosity of the fluid medium, the size of the particle, and the temperature as described on the equation (9).

There is a **linear increment in the MSD** with time for a particle that moves freely because of thermal fluctuations in a Newtonian fluid and the slope of this increment is represented by the diffusion coefficient of the particle as follows:

$$\langle \Delta r^2(t) \rangle = 2dD_t t^{\alpha} = MSD \tag{10}$$

Where:

- $\langle \Delta r^2(t) \rangle$ is the MSD in the time t;
- d is the dimensionality of the motion ("2" in a plane, "3" for motion in space);
- α is the scaling factor;
- D_t is the diffusion coefficient of the particles.

Considering a motion three-dimensional: $d = 3$ and it could be possible re-write the **MSD**:

$$MSD = 6D_t t^{\alpha} \tag{11}$$

The Phyton Program reads the **autocorrelation** that comes from the DLS in the form:

$$G_2(\tau) = B(1 + \beta |G_1(\tau)|^2) \tag{12}$$

Where:

- B is the Baseline (usually "1" but change in function of the asymptote of the autocorrelation);
- β is the instrumental response;
- $G_1(\tau)$ (**electric field correlation function**) is extracted by the program:

$$G_1(\tau) = \sqrt{G_2(\tau) - B} \tag{13}$$

Also, is written the **MSD in a logarithmic form**:

$$\langle \Delta r^2(t) \rangle = \frac{6}{q^2}[\log(g_1(0)) - \log(g_1(t))] = MSD$$

$$MSD = -\frac{6}{q^2}[\log(g_1(t)) - \log(g_1(0))] \quad (14)$$

In this equation q is the **scattering wave factor** [*equation (8)*] written as a constant. From where the program extracts:

$$MSD = -\frac{6}{q^2}\log(G_1(\tau))$$

Now is possible to calculate:

$$\log(G_1(\tau)) = -\frac{q^2}{6}MSD$$

$$G_1(\tau) = e^{-\frac{q^2}{6}MSD}$$

Considering the equation (11) and (14) the program starts to do the **fitting** of:

$$\frac{1}{6}\log(MSD) = \log(D_t) + \alpha\log(t) \quad (15)$$

From the MSD fitting come out the values of α and D_t that it could be possible analyse. But, before to obtain the results the first "RUN" of the program sent to video a Figure that shows the trend and where is possible to select automatically the range to analyse:

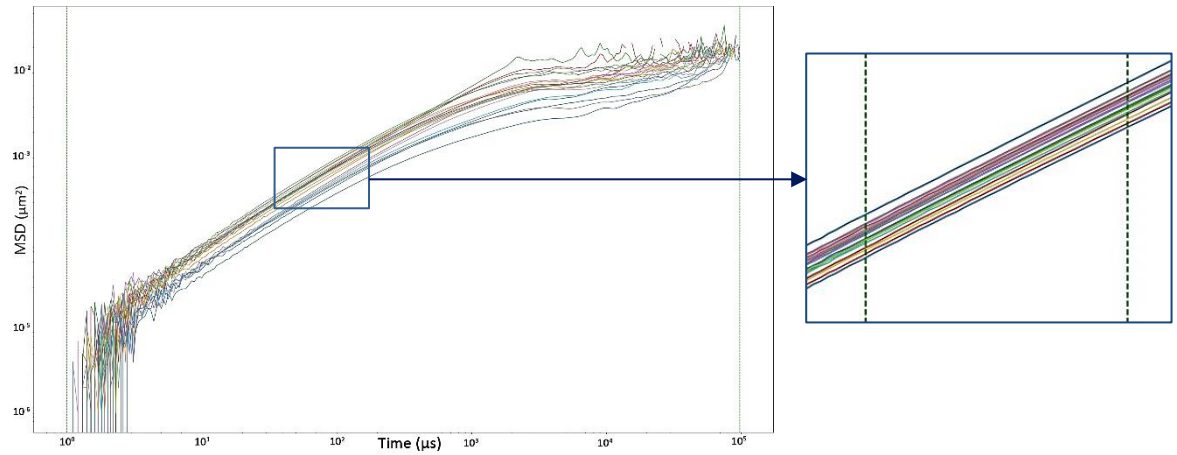

FIGURE 20 - SELECTION OF THE RANGE

2.3 Rheometer Experiment

This experimental part begins the macroscopical study thanks to rheology. This step underlines the differences seen on the microscopical measurements.

2.3.1 What is a Rheometer?

A rheometer is a laboratory device used to measure how a fluid responds to an applied force. It is necessary for complex fluid impossible to be defined with a single viscosity value, so it needs more parameters than a normal viscosimeter. This precision instrument measures the Rheology of the fluid. This last is considered as the study of stress-strain relationships in materials measures flow and deformation of materials and indicates the distance over which a body moves under the influence of an external force or stress. The rheometer contains different geometric configurations, controls the environment, applies and measures ranges of stress, strain, and strain rate.

This work uses the "Discovery Hybrid Rheometer HR-2" from TA instruments:

FIGURE 21 - DISCOVERY HYBRID RHEOMETER HR-2 [9]

The Discovery Series Hybrid Rheometer (DHR) has the best technology present on the rheological world.

In particular, the HR-2 is a rotational rheometer with an innovative technology that presents:

1- **Second Generation Magnetic Thrust Bearing:** a patented design with less than 70% of the friction;
2- **Advanced Drag Cup Motor:** A motor designed with digital control and extremely smooth acceleration and a minimum inertia, temperature and friction;
3- **Normal Force Rebalance Transducer:** to accurately measure the normal force up to 50N;
4- **New True Position Sensor (TPS):** that make real-time gap corrections for movement due to thermal expansion, working with all smart geometries.
5- **Active Temperature Control (ATC):** with wireless temperature across air gap in non-contact system.

In addition, with the Rheometer is provided a Technology Software: TRIOS, a software package that uses cutting-edge technology for instrument control, data collection, and data analysis for thermal analysis and rheology.

2.3.2 Macrorheology

As just said, the Rheometer works on the rheology describing the deformation and flow behaviour of all kinds of material. This branch of physics is related to the viscous behaviour of the fluids. These lasts contain molecules in different sizes but, larger particles are the reason for higher viscosity values. There are different methods for measuring viscosity that can only be determined with absolute systems.

The absolute values have a relatively narrow shear gap as defined by specific standards for measuring systems (ISO 3219 and DIN 53019) that describe the possible geometries shown on “Figure 22”: cone-plate, concentric cylinders and plate-plate.

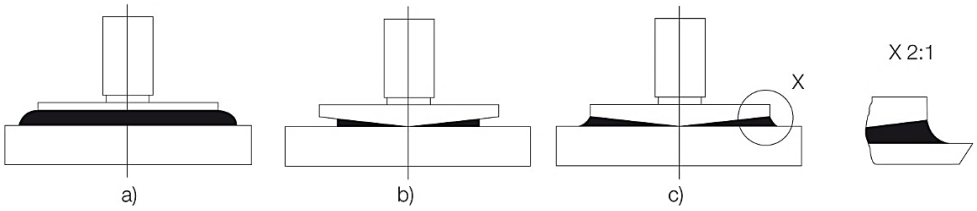

FIGURE 22 - FILLING OF CONE-PLATE MEASURING SYSTEM AFTER GAP SETTING: A) OVERFILLED, B) UNDERFILLED, C) CORRECT AMOUNT (ACCORDING TO DIN 51810-1) [10]

Rotational tests with a rheometer can be carried out simulating the processes that are dependent on flow velocity or volume flow rate or presenting the driving force via torque or shear stress.

Viscosity values are not constant values as they are affected by many conditions. For Newtonian flow like water, the viscous flow behaviour is easier: viscosity is independent of the shear rate. The situation is different for **pseudoplastic flow** or **dilatant flow** behaviour. The first case: Shear-thinning behaviour, is related to the internal structures of samples and it is characterized by a decrease of viscosity with an increase of shear rates. The second case: Shear-thickening, means an increase of viscosity with an increase of shear rates.

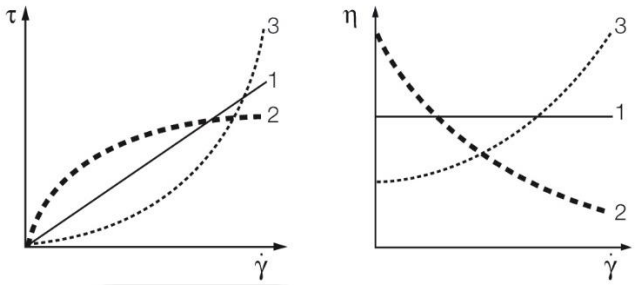

FIGURE 23 - FLOW CURVES (LEFT) AND VISCOSITY CURVES (RIGHT) FOR (1) IDEALLY VISCOUS, (2) SHEAR-THINNING, AND (3) SHEAR-THICKENING FLOW BEHAVIOUR. [10]

There are many materials with a mixture of viscous and elastic behaviour called viscoelastic that needs an appropriate study. In this way, the calculation of the viscosity is done after the necessary definition of the shear stress and the shear rate but to define the rheological parameters for the description of flow behaviour is used the two-plates model where shear is applied to a sample sandwiched between the two plates:

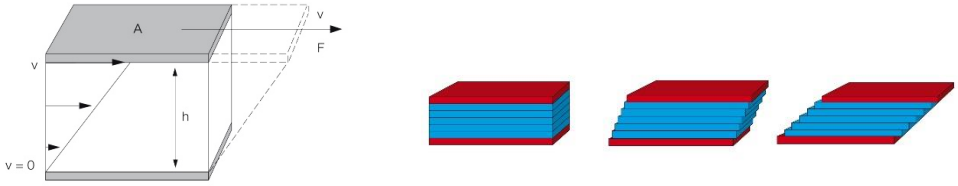

FIGURE 24 - CALCULATION OF SHEAR STRESS AND SHEAR RATE USING THE TWO-PLATES MODEL WITH SHEAR AREA A, GAP WIDTH H, SHEAR FORCE F, AND VELOCITY V [10]

- **Shear stress** is the component of stress coplanar with a material cross-section and is represented by the division between the shear Force F and the Shear Area A:

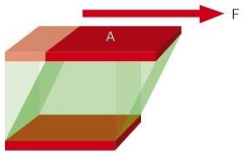

FIGURE 25 - SHEAR STRESS

$$\tau = \frac{F}{A} \quad \left[\frac{N}{m^2}\right] = [Pa] \tag{16}$$

- **Shear rate** is defined by the velocity v and the shear gap h as in the following equation:

$$\dot{\gamma} = \frac{v}{h} \quad [s^{-1}] \tag{17}$$

- **Law of viscosity**:

$$\eta = \frac{\tau}{\dot{\gamma}} \quad [Pa\ s] \tag{18}$$

- **Shear strain or shear deformation** formed by the deflection path s and the shear gap h:

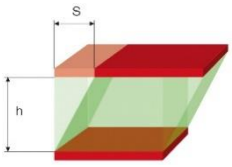

FIGURE 26 - SHEAR STRAIN

$$\gamma = \frac{s}{h} \quad [-] \tag{19}$$

- **Shear modulus** is used to measure the stiffness of material and is defined as:

$$G = \frac{\tau}{\gamma} \quad [Pa] \tag{20}$$

The two-plates model can also be used for explaining oscillatory tests that are controlled by sinusoidal strain.

A sine curve is described by its amplitude (maximum deflection) and its oscillation period or frequency on “Figure 27.a”

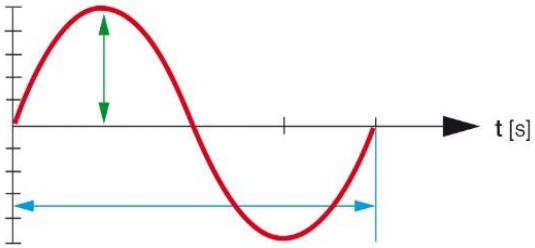

FIGURE 27.A - SINE CURVE FOR SINUSOIDAL STRAIN TEST

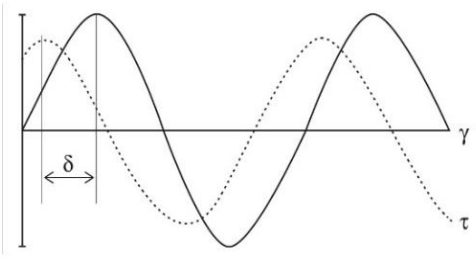

FIGURE 27.B- SINUSOIDAL FUNCTION FOR THE OSCILLATORY TEST

The force that acts is measured and required as a counter while the signal is rheologically evaluated as a shear stress τ. So, the oscillatory test for viscoelastic behaviour is presented as a sinusoidal function versus time "Figure 27.b"

For samples with a viscoelastic behaviour the sine curves show a time lag for the response signal called the **phase shift** δ. This phase is the time lag between the present and the resulting sinusoidal oscillation that is always between 0° and 90°.

The viscoelastic behaviour of a sample in oscillatory shear tests is represented by the **complex shear modulus** G^*, the shear stress amplitude τ_A and the strain amplitude γ_A. This takes the following form:

$$G^* = \frac{\tau_A}{\gamma_A} \quad [Pa] \tag{21}$$

The phase δ is correlated to the complex shear modulus, representing the angle determined for each measuring point. The vector diagram below in "Figure 28", shows this correlation and it demonstrates how the complex shear modulus G* could be defined as a vector that, so is formed by others two represented by its projection on the axis:

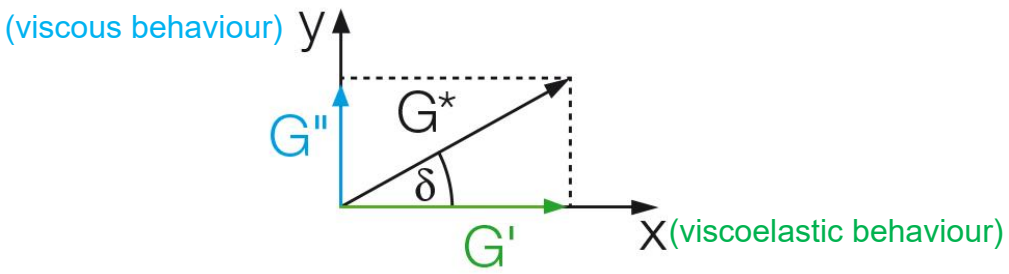

FIGURE 28 – CORRELATION BETWEEN THE COMPLEX SHEAR MODULUS AND THE PHASE SHIFT

- The **storage modulus** G' is the projection of the complex shear modulus on the x-axis and represents the elastic portion of the viscoelastic behaviour (stored deformation energy) describing the solid-state behaviour of the sample;
- The **loss modulus** G'' is the projection of the complex shear modulus on the y-axis and characterizes the viscous portion of the viscoelastic behaviour (deformation energy lost, dissipated, through internal friction when flowing) and means the liquid-state behaviour of the sample.

The **complex formula of dynamic modulus** is expressed in terms of imaginary and real parts:

$$G^*(\omega) = G'^{(\omega)} + G''(\omega) \tag{22}$$

Where both moduli can be calculated by measuring the lag time between stress and strain as it follows:

Storage modulus $$G' = \frac{\sigma_0}{\varepsilon_0}\cos(\delta) \tag{23}$$

Loss modulus $$G'' = \frac{\sigma_0}{\varepsilon_0}\sin(\delta) \tag{24}$$

Where Strain and stress are respectively expressed as:

Strain $$\varepsilon = \varepsilon_0 \sin(\mathrm{wt}) \tag{25}$$

Stress $$\sigma = \sigma_0 \sin(\mathrm{wt} + \delta) \tag{26}$$

They are correlated in the function of the materials: for Viscoelastic solids G' > G'' while for viscoelastic liquids G'' > G'.

For evaluating the time-dependent viscoelastic behaviour is necessary to consider:

- **Frequency sweeps** (Figure 29) to show the time-dependent behaviour of a sample in the non-destructive deformation range.
- **Amplitude sweeps** (Figure 30) to describe the deformation behaviour of samples in the non-destructive deformation range and at determining the upper limit of this range;

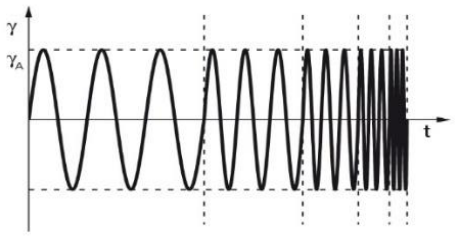

FIGURE 29 - FREQUENCY SWEEPS

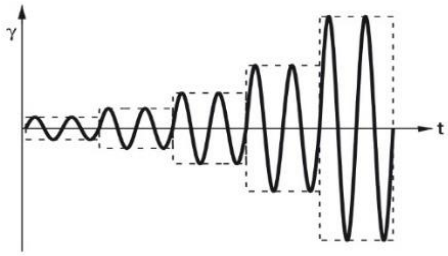

FIGURE 30 - AMPLITUDE SWEEPS

2.3.3 Protocol of the Rheometer experiments

The Experiments on the Rheometer start with the preparation of the samples in Eppendorf tubes, each one with 1,5mL of Solution at 2mg/mL of concentration:

(A) 0,75mL Hylauronic Acid and 0,75mL of water;
(B) 0,75mL Hylauronic Acid and 0,75mL of water with 15µL of particles: CoreShell-Ur (1mg/mL);
(C) 0,75mL Hylauronic Acid and 0,75mL of UREA 50mM with 15µL of particles: CoreShell-Ur (1mg/mL);
(D) 0,75mL Hylauronic Acid and 0,75mL of UREA 100mM with 15µL of particles: CoreShell-Ur (1mg/mL).

These lasts two samples are the ones with the Reaction.

After that is done the calibration of the Rheometer with the choice of different parameters on the Software TRIOS:

- Geometry: Flat-Plate certificate;
- GAP: 100µm;
- Diameter: 40nm;
- Strain Rate;
- Oscillatory Ramp.

When the calibration is completed is taken an amount of solution from an Eppendorf with a Micropipette to collocate them on the centre of the plate. Now, is possible to give the start to press the fluid and go forward with the RUN to the Software that allows the rotation.

2.4 SEM

This experimental part aims to take picture of the hyaluronic acid solution, searching differences between case without reaction and with it.

2.4.1 What is a SEM?

Scanning Electron Microscope SEM, it's an electronic microscope based on scanning the focused electron beam over a surface to create an image. This last is the product of the signal comes from the interaction between the electrons, that are in the beam, and the sample. This kind of microscope is used more than the one on light because of its shorter wavelengths and better resolution.

On specific, in this work is used the "Scanning Electron Microscope NOVA NanoSEM230" from FEI company:

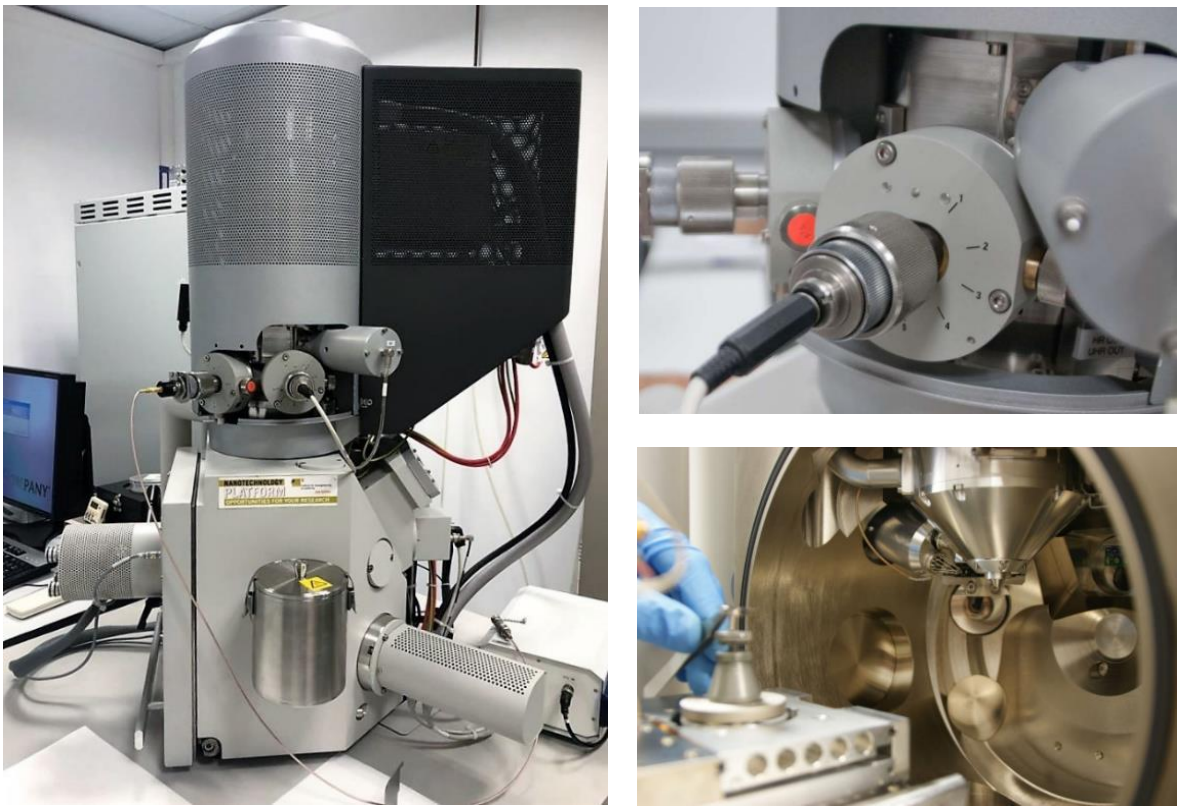

FIGURE 29 – NOVA NANOSEM 230 [11]

This microscope presents high resolution imagining with equipment that operates in two vacuum modes (high and low), to inspect conducting and insulating samples. The SEM components are:

- Source of electrons;
- Column down which electrons travel with electromagnetic lenses;
- Electron detector;
- Sample chamber;
- Computer and display to view the images.

At the top of the column, there is the production of the electrons that accelerate down to pass through a combination of lenses and apertures to produce the focused beam. This beam of electrons hits the surface of the sample that are scanned and collected images.

2.4.2 Protocol of the SEM measures

To use the Microscope SEM is chosen the technique of Freeze-drying or **lyophilization** of the samples. At the beginning are prepared three containers 3mL of Solution at 2mg/mL of concentration. Into each of them is put:

(A) 1,5mL of hyaluronic acid and 1,5mL of water;
(B) 1,5mL of hyaluronic acid and 1,5mL of water with 30µL of particles: CoreShell-Ur;
(C) 1,5mL of hyaluronic acid and 1,5ml of urea 100mM with 30µL of particles: CS-Ur.

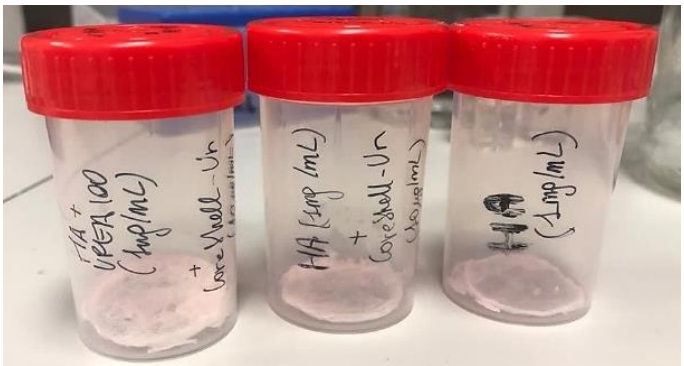

FIGURE 30 - CONTAINERS READY FOR LYOPHILIZATION

The three full containers are left for almost 4 hours before lyophilizing them.

The lyophilization is a dehydration system that consists of removing water from a liquid, pasty or solid product by freezing or freezing and evaporating under vacuum. When heating water in the solid-state and at very low pressure, the water sublimates and goes into the gaseous state (water vapour). This water vapour that is released from the solid product is trapped on a condenser or trap. This technology allows not to alter the quality of the product that is freeze-dried. It is the only drying method that perfectly preserves the molecular structure of the different molecules that make up the product.

There are different freeze-drying phases:

- **Freezing**: where the products are frozen, the water turned into ice;
- **Primary drying** or **sublimation** or **freeze-drying** consists in sublimating the interstitial ice; causes the evaporation of the water vapour which is fixed on a special coil which is at a temperature between -60 and -70°C;
- **Secondary drying** that is a heating operation. Here, the maximum of the temperature is + 50 ° C.

This freeze-drying technique allows us to obtain dehydrated products with an extremely low humidity rate between 1 and 5%. [12]

In this work the freezing phase consists on put the samples on liquid hydrogen (−252.87 °C), one container at a time, with the help of tweezers and cryo-gloves as in “Figure 31”.

FIGURE 31 - PUT THE SAMPLES ON THE LIQUID HYDROGEN

This technique is a cryogenic process that freezes samples instantly. The frozen samples are now ready to be put into the lyophilizer for 24 hours.

FIGURE 32 - VIRTIS GENESIS PILOT LYOPHILIZER

In this experiment is used the **Virtis Genesis Pilot Lyophilizer** from the SP Scientific that is shown in "Figure 32":

this freeze has shelf temperatures as low as -70°C and condenser temperatures to -85°C supports a wide range of applications. there is a selection of control formats and temperatures and vacuum sensors. then, The Genesis allows the configuration of a freeze dryer to meet any application requirement. [13]

Also, is provided with the **Wizard 2.0 controller** that is shown in "Figure 33", designed to meet the lyophilization requirements of a broad range of end-users. The control system utilizes a PICBASIC software program, custom hardware package and an embedded microprocessor system to provide full process automation, including complete recipe management and alarming. In many cases, the Wizard 2.0 controller can serve as a drop-in replacement for outdated controllers.

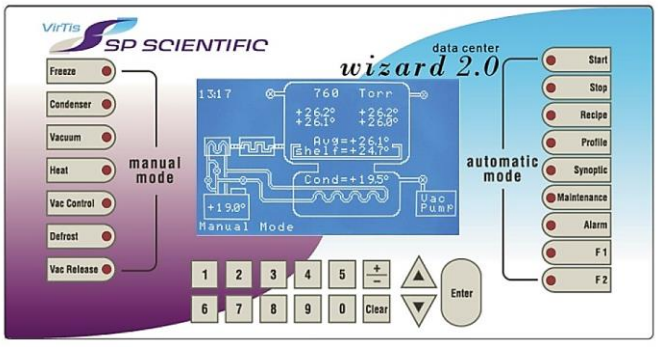

FIGURE 33 - THE WIZARD 2.0 CONTROLLER [13]

The day after the samples are dehydrated and ready to be seen on the microscope where are put on three different chambers. Moving the controller of the SEM is possible to see the sample and to take photos regularizing the focus.

3 Results and Discussions

This chapter is the core of the study, here are treated and analysed the initial situation, the final configuration and the results of the experiments. All the steps of the process are analysed considering constraints and possibilities.

3.1 Project purpose

The project aims to compare the navigation of the nanoparticles in water and complex media: hyaluronic acid. These nanoparticles are already studied as nanomotors because of their possible motion. This work explains how to create them in the laboratory to obtain the perfect nanomotor: CoreShell of mesoporous Silica with PEG and the urease enzyme (CoreShell-PEG-Ur) that allows their motion thanks to the reaction with the urea.

The first step of the project consists to underline the Hypothesis: the motion of the particles increases with the addition of urea because ammonia changes the pH. This value goes up changing the structure of the hyaluronic acid that became less viscous and could allow the motion of the nanomotors in a complex fluid media too. It is already known that the motion of nanomotors works in water, but the aim is to discover changes in a viscoelastic media too.

Focusing on this concept it was necessary to optimize the chemical part [2.1] doing the synthesis particles and then adding PEG to avoid the aggregation. When the chemical synthesis was clear it was possible to go forward doing the experiments on the DLS [2.2], calculating the MSD and extracting in the Phyton program the scaling factor “α” and Diffusion Coefficient “D” to see the motion of the particles. Thanks to these coefficients it was possible to see the accuracy of the synthesis or if the process needed a re-start. Only when the Autocorrelation, “α” and “D” resulted meaningful for the experiments was possible to move on the following experiments.

At this point, the microscopical system is defined so the next steps are referred to the macroscopical one. The macroscopical experiments are divided into two different parts: the

first one consists in using the Rheometer to demonstrate the change of the viscosity in the hyaluronic acid with the addition of urea while thanks to the second experiment is possible to see the macroscopical change of the structure on the SEM.

All the results for each experimental step are reported in the following sections.

3.2 DLS Results

The Results on DLS started taking all the Data from the machine and putting them on the Phyton Program explained in the chapter [2.2.5]. In the Output of the program there are the MSD fitting with the values of "α" and "D". The only thing that change in the program from the Water and the hyaluronic acid is the "CutOff2" that means more or less when the autocorrelation goes asymptotically to "1" and is $5*10^6$ for H.A. and $1*10^5$ for water.

- **AUTOCORRELATION G_2**

The first visible result that can immediately show the correctness of the experiment is the autocorrelation. It can be seen or on the DLS machine and in the output of the Phyton program. Of course, this concept is valid only for water because the autocorrelation in a complex fluid is unknown. Are now presented in "Figure 34" two extreme type of autocorrelation: on the left the best autocorrelation expected from an acquisition, on the right the worst one.

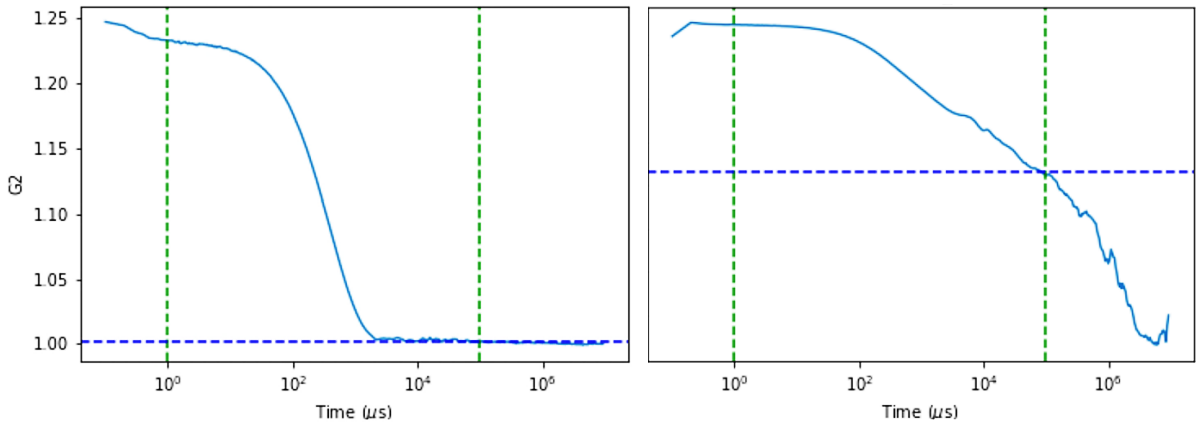

FIGURE 34 - AUTOCORRELATION RELIABLE ON THE LEFT AND UNRELIABLE ON THE RIGHT

3.2.1 DLS Data analysis

From the DLS are extracted different variables important to understand the correctness of the chemical synthesis and the reliability of the measurements, as the coefficients to see the possible motion of the particles.

- **RADIUS**

The nanomotors are created in the laboratory to have a radius of 200nm. At the beginning was simple to see and check the radius of the particles from data taken directly from the DLS. For each measure, DLS reads instantly the radius presents on the sample and shows in output different picks depending on the percentage of mass occupied as shown in "Figure 35". Usually is possible that the machine reads aggregations, powder or bubbles too. For this reason, the result with less picks are more reliable.

In general, was possible to see a similar constant radius for all the samples demonstrating a constant behaviour. The functionalization process did not alter the size of the particles and didn't cause any aggregation, otherwise there would be peaks of bigger radius. To underline this concept, it is created a gaussian distribution of the radius in "Figure 36".

The graph shows how each sample follow the same trend, with the maximum around 200nm or more, considering the range of the scattering too.

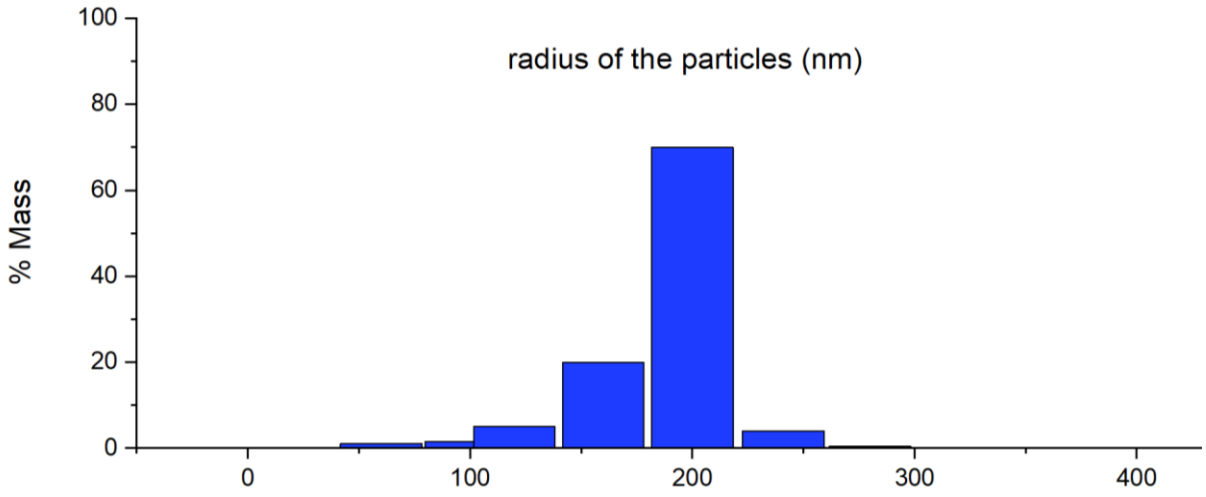

FIGURE 35 - RADIUS OF THE PARTICLES

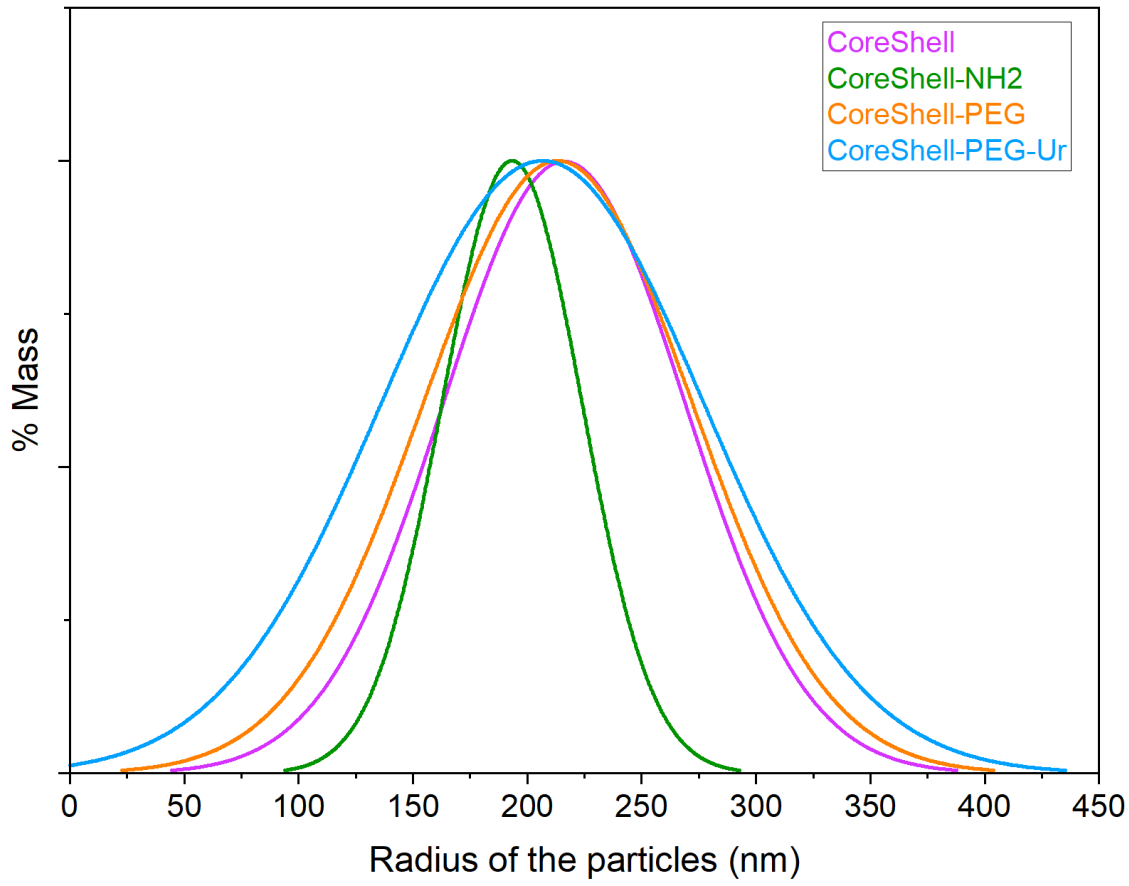

FIGURE 36 - DISTRIBUTION OF THE RADIUS OF THE PARTICLES

- **ZETA POTENZIAL**

The value of zeta potential is extracted from the DLS using a **Mobius Dip Cell** as explained in the chapter 2.2.1. The stability behaviour of a colloid depends on zeta potential that here is studied to understand how it change for each type of particle, in a correlation of their diffusions, and prove the validity of the particle functionalization.

Thanks to scientific literature is known the value of zeta potential for each examined sample. These values are showed in the "Table 9".

TABLE 9 - VALUES OF ZETA POTENTIAL

particle	CoreShell	CoreShell-NH_2	CoreShell-PEG-Ur
Zeta potential	-45 \ -35	+7 \ +15	-20 \ -30

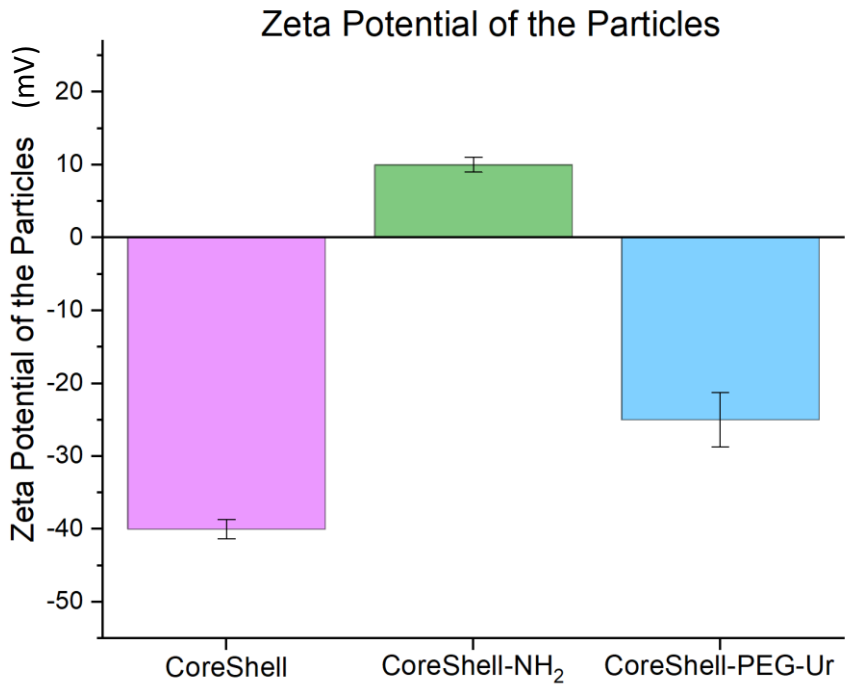

FIGURE 37 - ZETA POTENTIAL OF THE PARTICLES

In "Figure 37" is possible to see that the values are confirmed in sign and value. The sign resulted quite negative for the CoreShell, positive for the particles with the amino group (CoreShell-NH_2) and again negative for the CoreShell with the urease enzyme (CoreShell-PEG-Ur).

- **DIFFUSION COEFFICIENT**

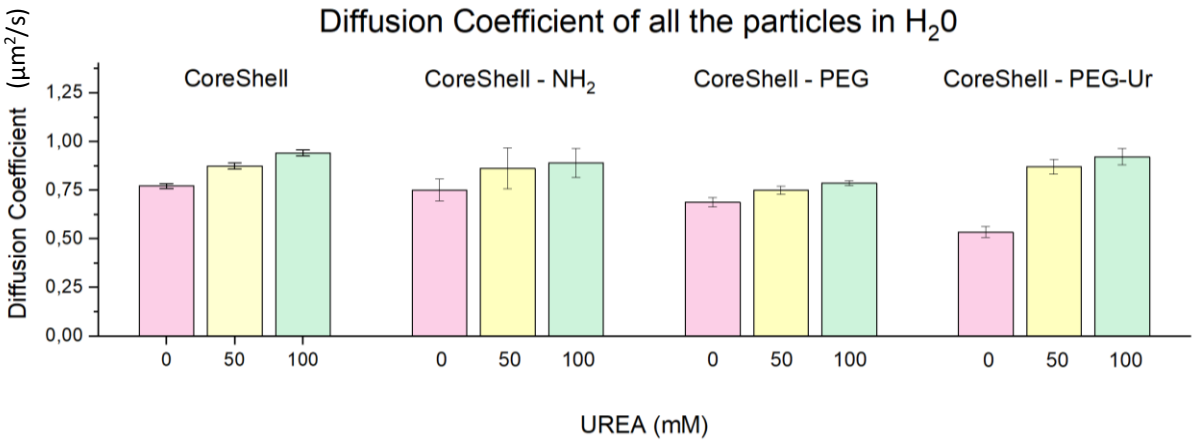

FIGURE 38 - DIFFUSION COEFFICIENT OF THE PARTICLES

The results of the diffusion coefficient are shown in "Figure 38" for each studied sample. From scientific literature are already known the expected values of the diffusion coefficient: "0,8 $\mu m^2/s$" for the particles without reaction and around "1,2 $\mu m^2/s$" when the enzymatic reaction exists. This means that for the particle with the enzyme with the urea is expected an increase of diffusion to demonstrate the possible motion of the particle. To obtain these results, the chemicals synthesis were repeated many times, but it was possible to see the correct trend in the end.

- **SCALING FACTOR α**

Firstly, is useful to see the results of the scaling factor in the water "Figure 39" to underline that α remain constant and almost "1" for every type of particles, with or without urea. This is important because, how explained, α define the diffusivity of the regime that is diffusive when the coefficient is "1", as in this case.

In these experiments are always present human errors, chemical errors or machine reading errors. These contribute to obtain a value a little bit different from one to another. It was demonstrated that removing the part of acquisition not expected, the coefficient is always equal to one or more. Even though, is decided to maintain all the data with the errors because are always present in experiments and because the deviation from the ideal value negligible. With a huge amount of data was performed the average of each sample and the calculation of the Standard Error of the Mean (SEM).

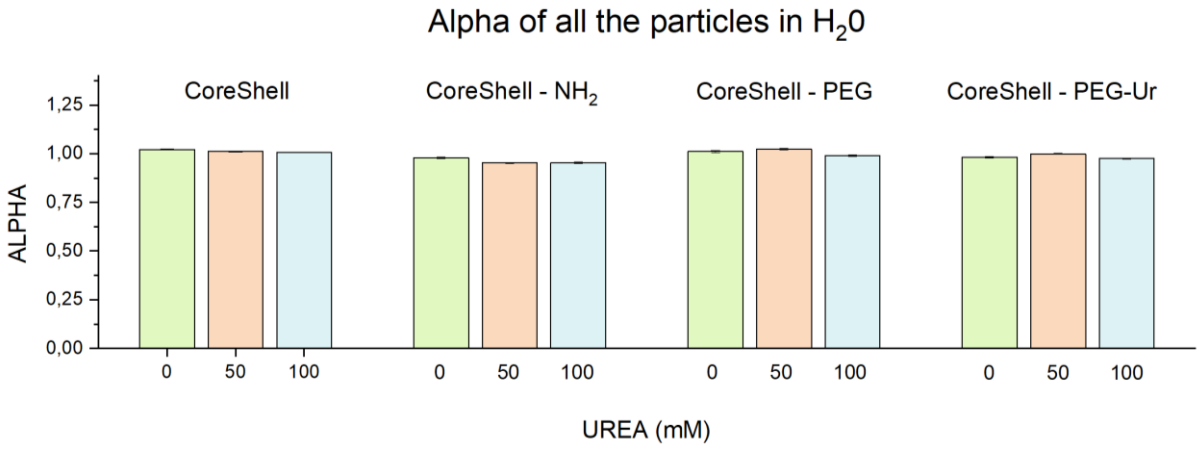

FIGURE 39 - SCALING FACTOR OF ALL THE PARTICLES

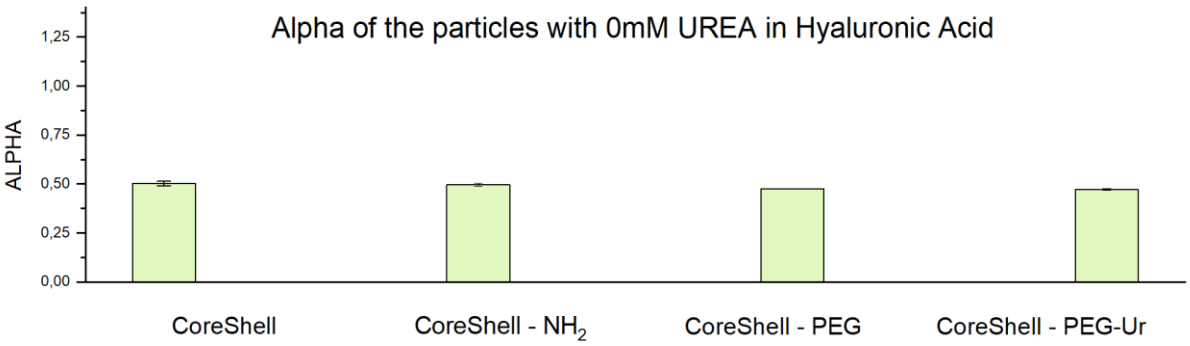

FIGURE 40 - SCALING FACTOR OF THE PARTICLES IN HYALURONIC ACID WITHOUT UREA

In "Figure 40" are shown the scaling factors for hyaluronic acid that were unknown, and they are resulted lower than "1" but almost the same for each kind of particle. In fact, the values of the scaling factor in H.A. were always approximately 0,5 and the average of each sample resulted in this range as it is shown in the first line of the "Table 11".

TABLE 10 - SCALING FACTOR OF THE PARTICLES IN H.A. WITH 0MM OF UREA

α of the Particle	**CoreShell**	**SEM error**	**CoreShell-NH_2**	**SEM error**
urea 0mM	0,501834	0,011486493	0,49636719	0,002590286
urea 50mM	0,481683	0,015085363	0,456043524	0,003671404
urea 100mM	0,491459	0,014564816	0,463144952	0,004260123
α of the Particle	**CoreShell-PEG**	**SEM error**	**CoreShell-PEG-Ur**	**SEM error**
urea 0mM	0,474704071	0,006414611	0,472050048	0,005876
urea 50mM	0,484786429	0,01001887	0,572389	0,003424
urea 100mM	0,481925214	0,007261243	0,544628857	0,00709

To study the cases with urea 50mM and 100mM the values of the scaling factor are firstly normalized to the case without urea and then are studied the increase of percentage of them to see a possible relevant change.

In the Figure “41” are showed the four groups of samples where it is possible to see, as in the “Table 11”, that the only relevant change of the values with the urea is present only in the group of the particles with the enzyme reaction. It could be mean a possible motion in this specific fluid. Of course, this is only a starting point to see a possible motion in H.A. that will need more experiments and proves but for now was possible to see an increase in the motion parameters which indicates that there is an enhancement of it. The percentage increase is represented as the scaling factor in the case with the urea divided to the case without it: $\frac{\propto_{50mM}}{\propto_{0mM}}$ or $\frac{\propto_{100mM}}{\propto_{0mM}}$.

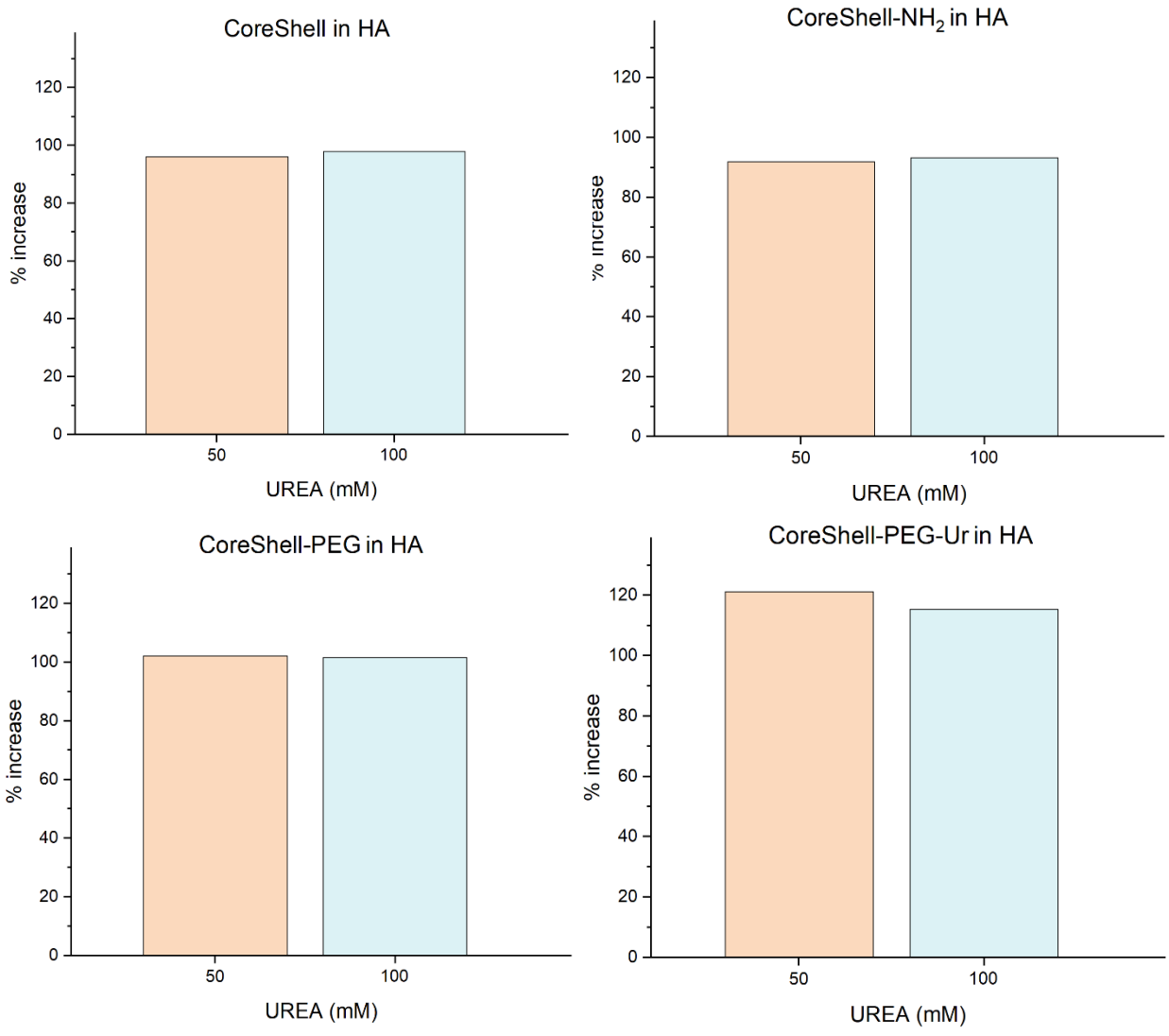

FIGURE 41 - GROUPS OF PARTICLES NORMALIZED AT 0MM UREA

3.2.2 DLS Microrheology

A possible improvement of the DLS experiment and its results is the discussion of the Microrheology.

Microrheology is the study with tracer particles can be performed using the dynamic light scattering (DLS) technique. For a DLS measurement, an incident light beam crosses a colloidal sample and a photodetector typically placed at 173° from the original beam. The scattering intensity at the detector is recorded and correlated with the intensity of the incident beam. The resulting auto-correlation function (g_1) could be used to extract information from the particles as size, hydrodynamic radius or diffusion coefficient of them. For studying the rheologic properties of the medium g_1 must be first associated to the MSD of the tracer particles as in the equation (15).

In the frequency domain, the MSD of a tracer particle is related to the **dynamic modulus** of its surrounding medium $G^*(\omega)$ with the expression:

$$G^*(\omega) = \frac{k_b T}{\pi a \langle \Delta r^2(1/\omega) \rangle \Gamma(1 + \alpha(\omega))} \tag{27}$$

Where:

- a is the radius of the tracer particle;
- Γ is the gamma function.

$\alpha(\omega)$ is the **power law exponent of the logarithmic slope** of $\langle \Delta r^2(t) \rangle$ at $t = \frac{1}{\omega}$. Thus:

$$\alpha(\omega) = \left. \frac{\mathrm{d} \ln \langle \Delta r^2(t) \rangle}{\mathrm{d} \ln t} \right|_{t=1/\omega} \tag{28}$$

For:

- $\alpha(\omega) = 1$: the tracer is embedded in a Newtonian fluid;
- $\alpha(\omega) \approx 0$: the probe diffuses in a pure elastic medium.

The complex formula of dynamic modulus is expressed in terms of its imaginary and real parts as in the equation (22).

With the **Kramers-Kronig relation**, the **loss and storage modulus** of the medium surrounding the tracer particles is calculated as:

$$G'(\omega) = G^*(\omega) \cdot \cos(\pi\alpha(\omega)/2)$$
$$G''(\omega) = G^*(\omega) \cdot \sin(\pi\alpha(\omega)/2) \quad (29)$$

Finally, these terms are used to calculate the **complex viscosity of the medium** as follows:

$$\eta^*(\omega) = \sqrt{\frac{\left(G''(\omega)\right)^2 + \left(G'(\omega)\right)^2}{\omega^2}} \quad (30)$$

3.3 Rheometer Results

The Rheometer was necessary for the study of hyaluronic acid to see the macroscopical changes of the system focusing on this viscoelastic media and to know better the fluid properties. To study the properties of the fluid was necessary to study the trend of the storage G' and the loss modulus G'. As explained these moduli represent respectively the elastic and the viscous portion of the viscoelastic behaviour.

Here, are studied four samples of hyaluronic acid: alone, with samples and with 50mM or 100mM concentration of UREA as explained in the chapter [2.3.3]. To evaluate the viscoelastic behaviour of the fluid, for each sample are done two graphs about the trend of the moduli and the complex viscosity as a function of the angular frequency of the Rheometer. Each graph resulted from the average of three experiments where there are also reported the errors.

For evaluating the behaviour in the low shear-rate range, it is beneficial to use a log-log plot for the diagrams of flow curves and viscosity curves. The advantage of these diagrams on a logarithmic scale is that a very large range of values (several orders of magnitude) can be illustrated clearly in one diagram. With a presentation on a linear scale, however, this range can only be depicted to a limited extent.

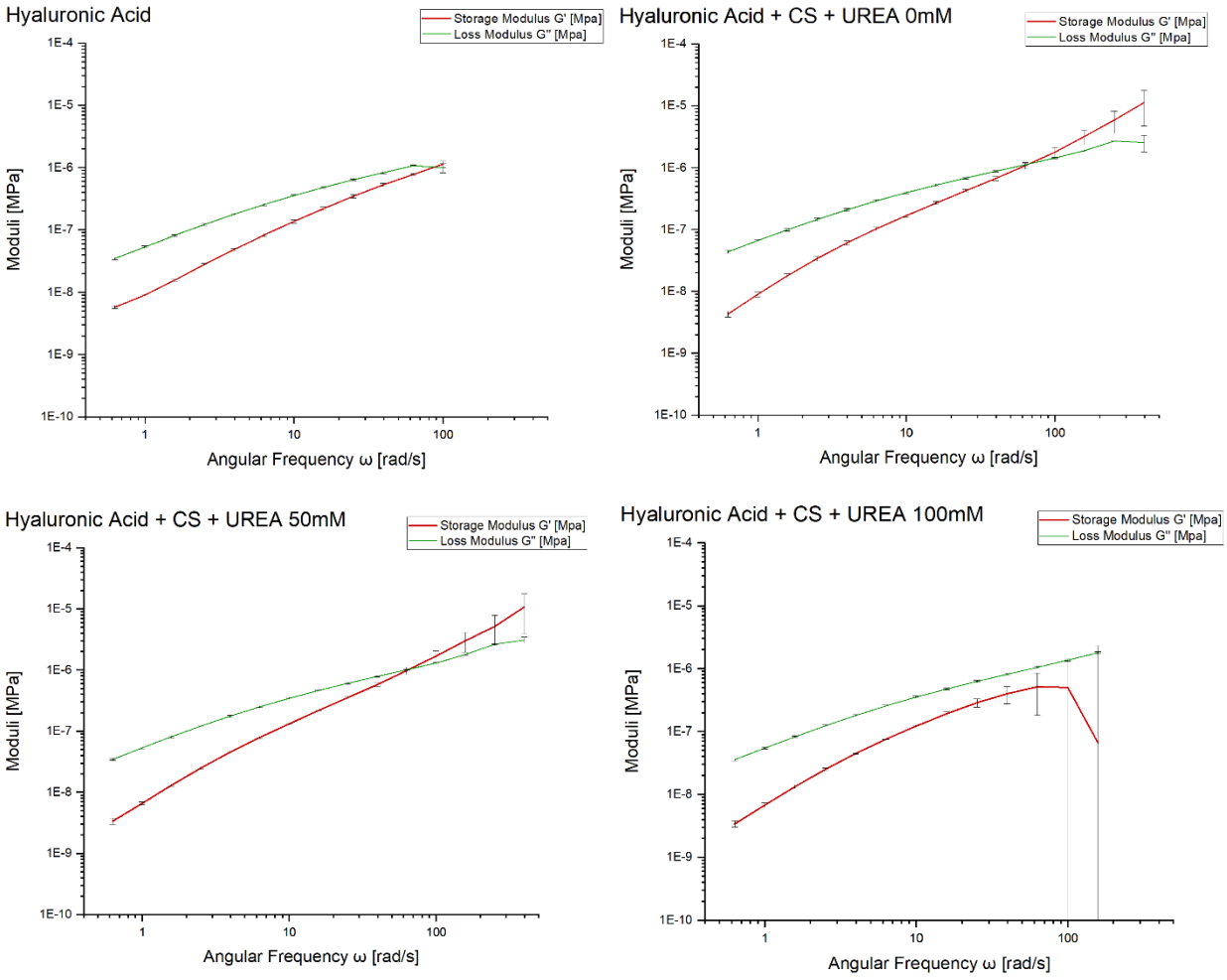

FIGURE 42 - GRAPHS OF THE STORAGE AND THE LOSS MODULUS OF THE SAMPLES IN FUNCTION OF THE ANGULAR FREQUENCY OF THE RHEOMETER

To have a less viscous behaviour, the elastic modulus of the structure should increase with the addition of the urea. It is expected an enhance of the storage modulus as a reduction of the complex viscosity.

The graphs in the “Figure 42” and in the “Figure 43” show that is present a change of the trend but is not so evident.

This could mean that there are local changes, as saw in the DLS microscopically results, but there are not globally changes as now demonstrated in the Rheometer. This could also can say that the particles are not changing the properties in the macroscopical scale but only in the microscopical one.

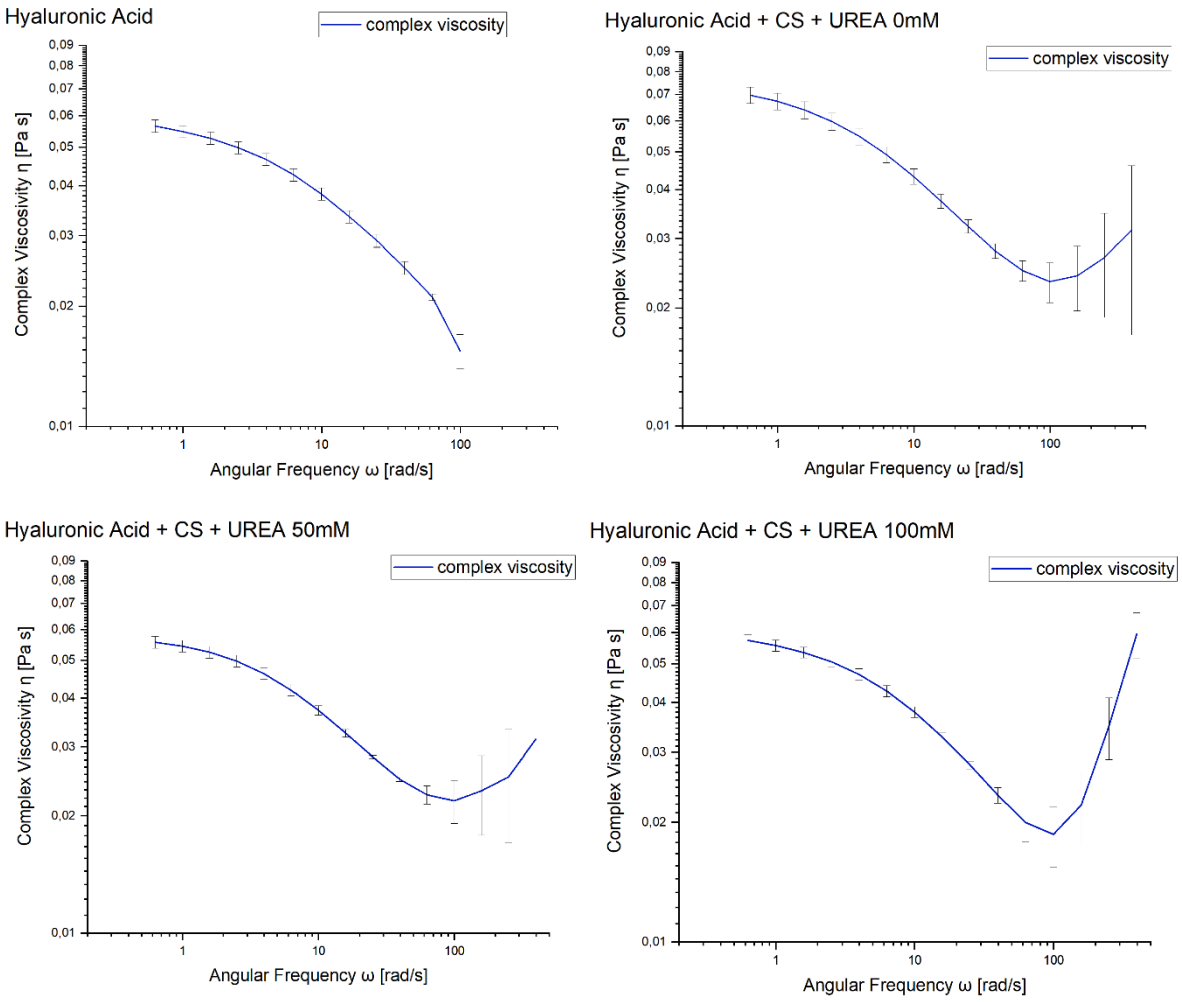

FIGURE 43 - GRAPHS OF THE COMPLEX VISCOSITY OF THE SAMPLES IN FUNCTION OF THE ANGULAR FREQUENCY OF THE RHEOMETER

3.4 SEM Results

The results of the microscope SEM are shown as photos of the different samples in the resolution of 300µm in "Figure 44" and 50 µm in "Figure 45".

These photos show immediately that for hyaluronic acid alone the structure is dense and there are not extremely difference with the addition of the particles. On the other hand, it is clear how the structure changes with the addition of the urea that, starting the reaction with the particle, modify the macromolecule.

Hylauronic Acid | H.A. and CoreShell-PEG-Ur | H.A, urea and CS-PEG-Ur

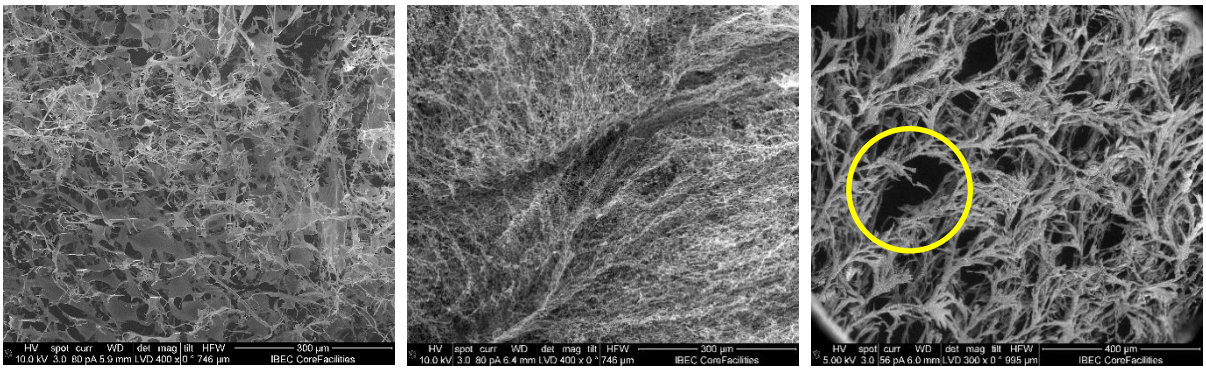

FIGURE 44 - SAMPLES SHOW IN SEM WITH A RESOLUTION OF 300 µM

Hylauronic Acid | H.A. and CoreShell-PEG-Ur | H.A, urea and CS-PEG-Ur

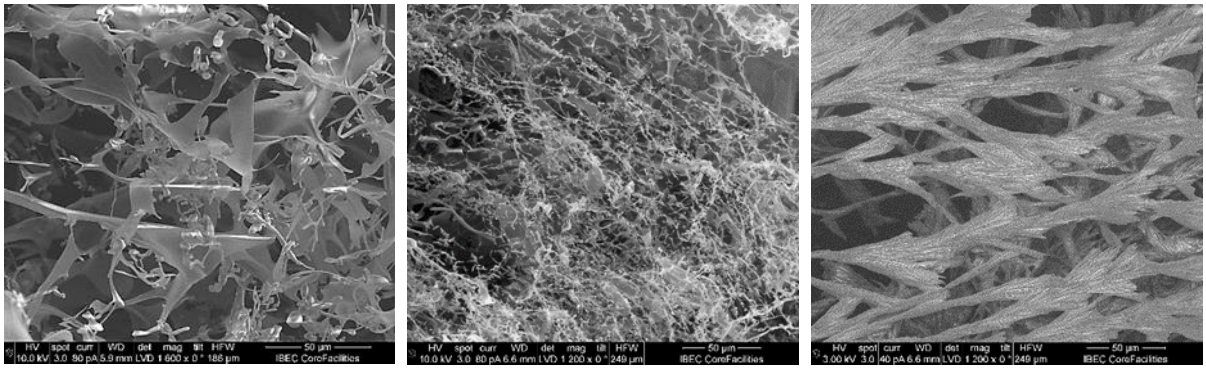

FIGURE 45 - SAMPLES SHOW IN SEM WITH A RESOLUTION OF 50 µM

The most important visible thing is that adding the urea the structure became less dense and it possible to see holes on the structure that is set in “Figure 46”. This might mean that with the introduction of the reaction it is creating the space to allow the motion of the particles.

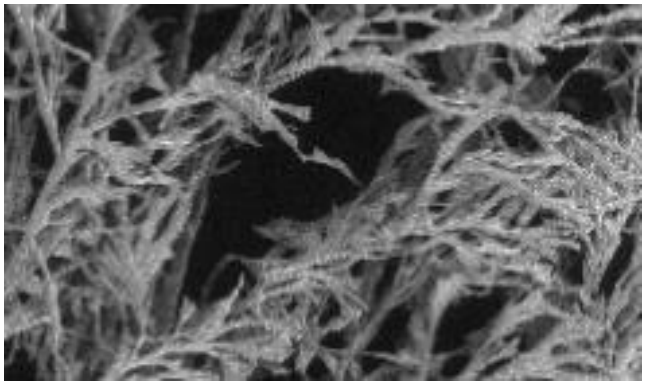

FIGURE 46 - ZOOM OF THE HOLE IN THE STRUCTURE OF H.A. AND UREA

4 Conclusion and considerations

The results of all the experiments led to different conclusions.

Firstly, the correctness of the synthesis is successfully supported by the z-potential and readings DLS. In this part it is demonstrated that the addition of PEG avoided extreme aggregation of particles and also that the nanomotors with urease enzyme and urea move in water, as proved by the diffusion of the MSD.

For the Hylauronic Acid it was possible to see an increase in the motion parameters which indicates that there is an enhancement of it. There were also some indications of electrostatic interactions between the particles and the HA depending on the charge, as supported by the DLS motion experiments.

To support the microscopical experiments Rheometer and SEM demonstrated how the enzymatic reaction can change the structure of the complex media, making possible a probable motion on it. In particular, the Rheometer showed an irrelevant global change that could mean the particles are not changing the properties in the macroscopical scale but only in the microscopical one. Moreover, the SEM microscope showed that the structure changes when there is the enzymatic reaction with a creation of holes that might allow a probable motion of the particles.

A possible answer could be that these nanomotors might work microscopically in a complex media, without creating global changes in the system. In this way it will possible avoiding secondary effects in the entire system that will be the human body.

Now, the work will continue with the experiments focusing on these results and correlating them to a real possible motion in a complex fluid. Therefore, it will be possible to get close to a possible way to work on a system every time more like human tissue.

4.1 Possible improvements and Utopias

A nanorobot is a little particle or a little device that it can move on its own. The smallest is thousand time little than a hair's diameter. It is possible to Imagine a submarine that could go through a fluid by itself, without any interaction form outside, just with the energy that it has inside. In This way, nanorobots will fight cancer from inside of the human body.

A tumor has a structure like an onion where the mean cell is on the center and it grow around itself by adding layers. The actual medicine kills layers but if could possible to exist a powerful robot that could go on the center, through layers, where the mean cell is, that would be the most effective way to fight cancer.

There are already go-through robots experimented inside human-like tissue, in carcinogenic cells in laboratory. It has now possible to think that autonomous and self-propelling robots will navigate inside our body.

The nanorobots has simple structure tube, cylinder and sphere similar to what we naturally have inside our body and they are little particles that can move medicines in the body. The nanorobots have different characteristics: they need to move on their own, with no one that do it from the outside; they have to react to some incitement, and they have to be "intelligent". A cancer usually take place where there is a change of pH, chemical composition or temperature. They must be able to detect these changes. Once they arrive where is the mean cell there are different ways to release the medicine: it could be a nanorobot that can blow up and release what it carry, or it could open a little gate to release the medicine in the exact place where the tumor is or it could have a magnetic field to control the direction of the medicine.

The research aims to discover applications that are not available today and to arrive to have medicines that will have less side effects. Those how already have some form of disease instead of injecting the medicine directly inside the body and let it spread everywhere in the body, could inject nanomotors that go just to the sick part.

For example, those who have diabetes needs a little injection that can be replaced with nanomotors.

Another way is to encapsulate nanorobots that must arrive somewhere and free them in the exact place where they need to operate. This can reduce hundred time the side effects.

The challenge it is find the perfect combustible for nanomotors that make them unique.

In the last years the main object it is find a biocompatible combustible that does not produce any damage to the organism. The last combustible used is hydrogen peroxide but at high concentration it kills cells. It is obvious that using this type of fuel on the nanorobots is not a solution.

The nanomotor's world is now making ten years and we are just in the middle of way…

Annex A – Phyton Program

- **Microrheology reader plotter**

```
import matplotlib

#matplotlib.use('agg') #doesn't show plots (before importing plt)

import numpy as np

import matplotlib.pyplot as plt

plt.switch_backend('agg') #doesn't show plots

#backend Qt5Agg is the default

import openpyxl

from scipy import special

import sys

from tqdm import tqdm

import csv

from scipy.interpolate import interp1d

from scipy.signal import savgol_filter

from scipy.interpolate import splrep, splev, InterpolatedUnivariateSpline

from scipy.interpolate import UnivariateSpline

import math

from scipy.optimize import curve_fit

import glob

import os

import easygui

#DLS Measurements Constants

#theta= 173*2*np.pi/360 #Angle Detection of Malvern (cluster DLS)

theta = 163.5*2*np.pi/360 #Of Mobius (helix DLS)
```

```
theta = 163.5*2*np.pi/360 #Of Mobius (helix DLS)

#wave= 633e-9 #DLS Wavelenght of Malvern (cluster DLS)

wave = 532e-9 #Of Mobius (helix DLS)

#Experimental Constants

Kb= 1.3806488e-23 #Boltzmann Constant

T= 310 #Temperture

r= 400e-9 #Particle Radius

#n= 0.170 #Refractive Index HA

n = 1.3325 #Refractive index water

q = ((4*np.pi*n)/wave)*np.sin(theta/2) #Wavevector

'''Global variables defined'''

global corr, delayTime, header

def derivate(x,t):

'''Function that derivates a list x with respect to t'''

d = list()

for i in range(1,len(x)-1):

h = t[i+1] - t[i-1]

d.append((x[i+1]-x[i-1])/h)

return d, t[1:-1]

def eliminateValue(x,t):

'''This function reads the values of the array x and, if there are NaN's or
inf's in it it eliminates it from the list, together with the corresponding t''

if type(x) is list:

if type(x[0]) is not list:

xx = [x[i] for i in range(len(x)) if (np.isnan(x[i]) | np.isinf(x[i])) != True]

tt = [t[i] for i in range(len(x)) if (np.isnan(x[i]) | np.isinf(x[i])) != True]

x = xx
```

```
t = tt

else:

xAux = list()

tAux = list()

for k in range(len(x)):

xx = [x[k][i] for i in range(len(x[k])) if (np.isnan(x[k][i]) |
                                                np.isinf(x[k][i])) != True]

tt = [t[k][i] for i in range(len(x[k])) if (np.isnan(x[k][i]) |
                                                np.isinf(x[k][i])) != True]

xAux.append(xx)

tAux.append(tt)

x = xAux

t = tAux

return x, t

def readData(fname, filedir):

'''Function that reads the data given a filename and directory.
Autocorrelation and delay time are read.'''

global corr, delayTime, header, nbMeas

corr, delayTime = [list() for i in range(2)]

global index1,index2, cutoff1, cutoff2

# files = glob.glob(fname+'.csv')

# numbers = [int(files[i].split('(')[1].split(')')[0]) for i in range(len(files

# #This function sorts the files ordering it by the numbers

# numbers, files = (list(t) for t in zip(*sorted(zip(numbers, files))))

'''G2 folder is created in case it doesn't exist already'''

if not os.path.exists(filedir+'\\Data\\'+fname+'\\G2\\'):

os.makedirs(filedir+'\\Data\\'+fname+'\\G2\\')


'''.csv file is read'''
```

```
with open(filedir+'\\'+fname+'.csv',"r") as file:

reader = csv.reader(file, delimiter=";")

data = list(reader)

header = data[0]

'''It considers that the first column is the time. The second and following
columns are data from different measurements. Therefore, the number of
measurments is the number of columns - 1'''

nbMeas = len(header) - 1

if header[-1] == '':

nbMeas = nbMeas - 1

globals()['delayTime'].append([float(data[i][0].replace(',','.')) for i in
                                                    range(1,len(data))])

for j in range(1,nbMeas+1):

aux = list()

# print(j)

if j > len(data):


nbMeas = j - 1

break

for i in range(1,len(data)):

if data[i][j] == '':

aux.append(1)

else:

aux.append(float(data[i][j].replace(',','.')))

globals()['corr'].append(aux)

'''Cutoff times in us.
This cut-offs are selected MANUALLY.
The second cut-off can be very different depending on the sample.
This could be automated for the user to select it
manually as in "msd_analyser".'''

cutoff1 = 1
```

```
cutoff2 = 1e5

index1 = min(range(len(delayTime[0])), key=lambda i:
                                                abs(delayTime[0][i]-cutoff1))

index2 = min(range(len(delayTime[0])), key=lambda i:
                                                abs(delayTime[0][i]-cutoff2))

B = [np.mean([corr[i][index2-j] for j in range(10)]) for i in range(len(corr))]

for i in range(nbMeas):

fig0 = plt.figure(0)

plt.plot(delayTime[0],corr[i])

plt.axvline(cutoff1, color='g', linestyle='dashed')

plt.axvline(cutoff2, color='g', linestyle='dashed')

plt.axhline(B[i], color = 'b', linestyle = 'dashed')


plt.xscale('log')

plt.title(header[i+1])

plt.xlabel('Time ($\mu$s)')

plt.ylabel('G2')

#Part of the code to deal with columns having the same name, so that it
#                                          doesn't overwrite the files

fs = glob.glob(filedir + '\\Data\\'+fname+'\\G2\\G2_'+fname+'_*.png')

loop = True

filename = 'G2_'+fname+'_'+header[i+1]

aux = 1

while loop:

repeated = False

for n in range(len(fs)):

if filename in fs[n]:

repeated = True

if repeated:

filename = 'G2_'+fname+'_'+header[i+1]+'('+str(aux)+')'
```

```
aux += 1

else:

fig0.savefig(filedir + '\\Data\\'+fname+'\\G2\\'+filename+'.png')

fig0.savefig(filedir + '\\Data\\'+fname+'\\G2\\'+filename+'.svg',
                                                    format='svg',dpi=1200)
plt.close()

with open(filedir + '\\Data\\'+fname+'\\G2\\'+filename+'.txt', 'w') as f:

f.write('Delay Time (us)\tG2 autocorrelation\n')

for j in range(len(corr[i])):

f.write("%f\t%f\n" % (delayTime[0][j],corr[i][j]))

f.close()

loop = False

#End of code to deal with columns having same names

def getMSD(fname, filedir):

'''This function obtains the MSD from G2 autocorrelation'''

if not os.path.exists(filedir+'\\Data\\'+fname+'\\MSD\\'):

os.makedirs(filedir+'\\Data\\'+fname+'\\MSD\\')

if not os.path.exists(filedir+'\\Data\\'+fname+'\\G1\\'):

os.makedirs(filedir+'\\Data\\'+fname+'\\G1\\')

global corr, delayTime, header, nbMeas

global msd, mean_msd, std_msd, sem_msd, time_msd, g1

msd, time_msd, g1 = [list() for i in range(3)]

'''If the data was taken with the Malvern DLS, the correlation data is g2-1
But if it was taken with the Mobius, the correlation data is just g2'''

B = [np.mean([corr[i][index2-j] for j in range(10)]) for i in range(len(corr))]

g1 = [[corr[i][j]-B[i] for j in range(len(corr[i]))] for i in range(len(corr))]

'''One way of calculating the msd
Not normalizing g2 and doing the logarithm of the difference
of the intercept of g1 and g1(g). For the intercept, the first N (10) points
are used to calculate it.'''
```

```
g1 = [[np.sqrt(g1[i][j]) for j in range(len(g1[i]))] for i in range(len(g1))]


intercept = [np.mean([g1[i][index1+j] for j in range(10)])
                                                    for i in range(len(g1))]

'''MSD is calculated in m^2, so we multiply by 1e12 to convert it to um^2'''

msd = [[1e12*(6.0/q**2)*(np.log(intercept[i])-np.log(g1[i][j]))
            for j in range(len(g1[i]))][index1:index2] for i in range(len(g1))]

time_msd = delayTime[0][index1:index2]

'''Plotting G1'''

for i in range(nbMeas):

fig10 = plt.figure(10)

plt.plot(delayTime[0], g1[i])

plt.xscale('log')

plt.title(header[i+1])

plt.xlabel('Time ($\mu$s)')

plt.ylabel('G1 correlation function')

fig10.savefig(filedir + '\\Data\\'+fname+'\\G1\\G1_'+fname+
                                                    '_'+header[i+1]+'.png')

#Part of the code to deal with columns having the same name, so that it
#                                                 doesn't overwrite the files

fs = glob.glob(filedir + '\\Data\\'+fname+'\\G1\\G1_'+fname+'_*.png')

loop = True

filename = 'G1_'+fname+'_'+header[i+1]

aux = 1

while loop:

#This loops deals with measurements having the same name.
#If the current filename occurs more than once (it's already in
#                                                 the list "filenames")


#a parenthesis with a number is added at the end. The number
#inside the parenthesis is also iterated starting from 1 until
#the filename is not repeated
```

```
repeated = False

for n in range(len(fs)):

if filename in fs[n]:

repeated = True

if repeated:

filename = 'G1_'+fname+'_'+header[i+1]+'('+str(aux)+')'

aux += 1

else:

fig10.savefig(filedir + '\\Data\\'+fname+'\\G1\\'+filename+'.png')

fig10.savefig(filedir + '\\Data\\'+fname+'\\G1\\'+filename+'.svg',
                                                format='svg',dpi=1200)

plt.close()

with open(filedir + '\\Data\\'+fname+'\\G1\\'+filename+'.txt', 'w') as f:

f.write('Delay Time (us)\tG1 autocorrelation\n')

for j in range(len(corr[i])):

f.write("%f\t%f\n" % (delayTime[0][j],g1[i][j]))

f.close()

loop = False

#End of code to deal with columns having same names

'''Plotting MSD'''

for i in range(nbMeas):


fig1 = plt.figure(1)

plt.plot(time_msd, msd[i])

plt.xscale('log')

plt.yscale('log')

# plt.xlim([0,2e4])

plt.title(header[i+1])
```

```
plt.xlabel('Time ($\mu$s)')

plt.ylabel('MSD ($\mu$m$^2$)')

#Part of the code to deal with columns having the same name, so
#                                    that it doesn't overwrite the files

fs = glob.glob(filedir + '\\Data\\'+fname+'\\MSD\\MSD_'+fname+'_*.png')

loop = True

filename = 'MSD_'+fname+'_'+header[i+1]

aux = 1

while loop:

repeated = False

for n in range(len(fs)):

if filename in fs[n]:

repeated = True

if repeated:

filename = 'MSD_'+fname+'_'+header[i+1]+'('+str(aux)+')'

aux += 1

else:


fig1.savefig(filedir + '\\Data\\'+fname+'\\MSD\\'+filename+'.png')

fig1.savefig(filedir + '\\Data\\'+fname+'\\MSD\\'+filename+'.svg',
                                                    format='svg',dpi=1200)

plt.close()

with open(filedir + '\\Data\\'+fname+'\\MSD\\'+filename+'.txt', 'w') as f:

f.write('Time (us)\tMSD (um^2)\n')

for j in range(len(msd[i])):

f.write("%f\t%f\n" % (time_msd[j],msd[i][j]))

f.close()

loop = False

#End of code to deal with columns having same names
```

```
if __name__ == "__main__":
filepath = easygui.fileopenbox()
filedir = os.path.splitext(filepath)[0]
fname = filedir.split('\\')[-1]
filedir = filedir.split(fname)[0]
if not os.path.exists(filedir + '\\Data\\'+fname+'\\'):
os.makedirs(filedir + '\\Data\\'+fname+'\\')
# sys.exit()
readData(fname, filedir)
getMSD(fname, filedir)
```

- **Microrheology msd analyser**

```
import matplotlib
#matplotlib.use('agg') #doesn't show plots (before importing plt)
import numpy as np
import matplotlib.pyplot as plt
plt.switch_backend('agg') #doesn't show plots
#backend Qt5Agg is the default
import openpyxl
from scipy import special
import sys
from tqdm import tqdm
import csv
from scipy.interpolate import interp1d
from scipy.signal import savgol_filter
from scipy.interpolate import splrep, splev, InterpolatedUnivariateSpline
from scipy.interpolate import UnivariateSpline
```

```
import math

from scipy.optimize import curve_fit

import glob

import os

import easygui

from matplotlib.offsetbox import AnchoredText

'''Class adapted from https://matplotlib.org/users/event_handling.html'''

class LineBuilder:

    '''Class that creates two vertical lines when the button is pressed and
    then released This class is used to select the two cut-off
    regions from the MSD'''
    def __init__(self, v1,v2):

        self.v1 = v1

        self.v2 = v2

        self.v1_y = list(v1.get_ydata())

        self.v2_y = list(self.v2.get_ydata())

        self.v1_x = list(v1.get_xdata())

        self.v2_x = list(self.v2.get_xdata())

        self.minValue = self.v1_x[0]

        self.maxValue = self.v2_x[0]

        self.cid_press = v1.figure.canvas.mpl_connect
            ('button_press_event', self.on_press)

        self.cid_release = self.v1.figure.canvas.mpl_connect
            ('button_release_event', self.on_release)

        self.cid_close = self.v1.figure.canvas.mpl_connect
            ('close_event', self.close)

        v1.figure.canvas.start_event_loop(0)

    def on_press(self, event):

        '''When the button is pressed, the vertical line v1 is created,
        indicating the min value of the fitting region'''

        print('Pressed at: x=%f, y=%f' % (event.xdata, event.ydata))
```

```
        if event.inaxes != self.v1.axes: return

        self.v1_x = [event.xdata,event.xdata]

        self.v1.set_data(self.v1_x,self.v1_y)

        self.line.figure.canvas.draw()
    def on_release(self, event):

        '''When the button is released, the vertical line v2 is created,
        indicating the max value of the fitting region'''

        print('Released at: x=%f, y=%f' % (event.xdata, event.ydata))

        if event.inaxes != self.v2.axes: return

        self.v2_x = [event.xdata,event.xdata]

        self.v2.set_data(self.v2_x,self.v2_y)

        self.v2.figure.canvas.draw()

    def close(self,event):

        '''The values of v1 and v2 are ordered before exiting'''

        if self.v1_x[0] > self.v2_x[0]:

            self.minValue = self.v2_x[0]

            self.maxValue = self.v1_x[0]

        else:

            self.minValue = self.v1_x[0]

            self.maxValue = self.v2_x[0]

        self.v1.figure.canvas.mpl_disconnect(self.cid_press)

        self.v1.figure.canvas.mpl_disconnect(self.cid_release)

        self.v1.figure.canvas.mpl_disconnect(self.cid_close)

        self.v1.figure.canvas.stop_event_loop()

        print('Closing figure')

def fitFuction(t,a,b):

    '''If the MSD is of the form:
            MSD = 6D*t^alpha
    then
```

```
        log(MSD) = log(6D) + alpha*log(t)'''

    return a*t + b

def eliminateNaN(x,t):

    '''This function reads the values of the array x and, if there are
    NaN's or inf's in it eliminates it from the list, together
    with the corresponding t'''

    if type(x) is list:

        if type(x[0]) is not list:

            xx = [x[i] for i in range(len(x)) if (np.isnan(x[i]) |
                    np.isinf(x[i])) != True]

            tt = [t[i] for i in range(len(x)) if (np.isnan(x[i]) |
                    np.isinf(x[i])) != True]

            x = xx

            t = tt

        else:

            xAux = list()

            tAux = list()

            for k in range(len(x)):

                xx = [x[k][i] for i in range(len(x[k])) if (np.isnan(x[k][i]) |
                        np.isinf(x[k][i])) != True]

                tt = [t[k][i] for i in range(len(x[k])) if (np.isnan(x[k][i]) |
                        np.isinf(x[k][i])) != True]

                xAux.append(xx)

                tAux.append(tt)

            x = xAux

            t = tAux
```

```
    return x, t

def fitMSD(time_msd, msd, measname):

    '''First the time and msd are converted into log for the fitting
    IMPORTANT!!: Time is in us, need to multiply the diffusion by 1e6 later'''

    logtime = [np.log(t) for t in time_msd]

    logmsd = [np.log(msd) for msd in msd]

    '''NaN's from the logarithm are eliminated so that the
            fitting can be performed'''

    logmsd, logtime = eliminateNaN(logmsd,logtime)

    alpha = np.nan

    D = np.nan

    try: #Try to do the fitting

        popt, pcov = curve_fit(fitFuction, logtime, logmsd)

    except ValueError: #Different errors are considered

        print('There are NaNs in the data sets or incompatible data.')

        with open(filedir + '\\Data\\'+fname+'\\MSD\\MSDFitting_'
                  +fname+'_'+measname+'.txt', 'w') as f:

            f.write('Fitting could not be performed.')

            f.write('ValueError exception: There are NaNs in the data
                    'sets or incompatible data')

            f.close()

        fig = plt.figure()

        fig.savefig(filedir + '\\Data\\'+fname+'\\MSD\\MSDFitting_'
                    +fname+'_'+measname+'.png')

        fig.savefig(filedir + '\\Data\\'+fname+'\\MSD\\MSDFitting_'
                    +fname+'_'+measname+'.svg',format='svg',dpi=1200)

        plt.close()
    except RuntimeError:

        print('Least-squares minimization failed.')

        with open(filedir + '\\Data\\'+fname+'\\MSD\\MSDFitting_'
                  +fname+'_'+measname+'.txt', 'w') as f:
```

```
            f.write('Fitting could not be performed.')

            f.write('RuntimeError exception: least-squares
                    'minimization failed.')

            f.close()

        fig = plt.figure()

        fig.savefig(filedir + '\\Data\\'+fname+'\\MSD\\MSDFitting_'
                    +fname+'_'+measname+'.png')

        fig.savefig(filedir + '\\Data\\'+fname+'\\MSD\\MSDFitting_'
                    +fname+'_'+measname+'.svg',format='svg',dpi=1200)

        plt.close()

    except OptimizeWarning:

        print('Covariance of the parameters could not be estimated.')

        with open(filedir + '\\Data\\'+fname+'\\MSD\\MSDFitting_'
                  +fname+'_'+measname+'.txt', 'w') as f:

            f.write('Fitting could not be performed.')

            f.write('OptimizeWarning exception: Covariance of the
                    'parameters could not be estimated.')

            f.close()

        fig = plt.figure()

        fig.savefig(filedir + '\\Data\\'+fname+'\\MSD\\MSDFitting_'
                    +fname+'_'+measname+'.png')

        fig.savefig(filedir + '\\Data\\'+fname+'\\MSD\\MSDFitting_'
                    +fname+'_'+measname+'.svg',format='svg',dpi=1200)

        plt.close()
    else: #If the fitting can be done, the parameters are extracted

        alpha = popt[0]

        D = 1e6*np.exp(popt[1])/6 #Need to pass to um^2/s, because time is [us]

        #The data is saved and plotted

        with open(filedir + '\\Data\\'+fname+'\\MSD\\MSDFitting_'
                  +fname+'_'+measname+'.txt', 'w') as f:

            f.write('Alpha\tDiffusion coefficien
                    '(um^2/s)t\tInitial cutoff (us)\tFinal cutoff (us)\n')
```

```
            f.write("%f\t%f\t%f\t%f\n" % (alpha, D,time_msd[0], time_msd[-1]))

            f.close()

        auxT = np.linspace(time_msd[0],time_msd[-1],num = 10)

        auxX = [1e-6*6*D*(t**alpha) for t in auxT] #Time [us] and D [um^2/s]

        fig1 = plt.figure(1)

        ax = fig1.gca()

        plt.plot(time_msd,msd, label = 'Data')

        plt.plot(auxT,auxX,'g--', label = 'Fitting')

        text = "Fitting details\n" + r"${\alpha}$ = %.3f" % (alpha)
                                                +"\n" + r"D = %.6f" % (D)

        anchored_text = AnchoredText(text, loc=4)

        ax.add_artist(anchored_text)

        plt.legend(loc = 'upper left')

        plt.xscale('log')

        plt.yscale('log')

        plt.title(measname)

        plt.xlabel('Time ($\mu$s)')

        plt.ylabel('MSD ($\mu$m$^2$)')

        fig1.savefig(filedir + '\\Data\\'+fname+'\\MSD\\MSDFitting_'
                     +fname+'_'+measname+'.png')

        fig1.savefig(filedir + '\\Data\\'+fname+'\\MSD\\MSDFitting_'
                     +fname+'_'+measname+'.svg',format='svg',dpi=1200)

        plt.close()

    return alpha, D

def derivate(x,t):

    '''Function that derivates a list x with respect to t'''

    d = list()

#     d.append((x[1]-x[0])/(t[1]-t[0]))

    for i in range(1,len(x)-1):
```

```
        h = t[i+1] - t[i-1]

        d.append((x[i+1]-x[i-1])/h)

#    d.append((x[-1]-x[-2])/(t[-1]-t[-2]))

    return d, t[1:-1]

def getDiffusion(timeMSD,msd, fname, filedir, measname):

    '''Function that calculates and saves the local diffusion coefficient
        at each time point by derivating the MSD'''

    '''THIS FUNCTION IS NOT UPDATED'''

    if not os.path.exists(filedir+'\\Data\\'+fname+'\\Diffusion\\'):

        os.makedirs(filedir+'\\Data\\'+fname+'\\Diffusion\\')

    #Diffusio is the derivative of the MSD divided by 6 (in 3D)

    diffusion, time = derivate(msd,timeMSD)

    diffusion = [diffusion[i]*1/6 for i in range(len(diffusion))]

    fig2 = plt.figure(2)

    plt.plot(time,diffusion)

    plt.xscale("log")

    plt.yscale("log")

    plt.title(measname)

    plt.xlabel('Time ($\mu$s)')

    plt.ylabel('Diffusion coefficient ($\mu$m$^2$/$\mu$s)')

    plt.tight_layout()

    fig2.savefig(filedir+'\\Data\\'+fname+'\\Diffusion\\Diffusion_'
                 +fname+'_'+measname+'.png')

    fig2.savefig(filedir+'\\Data\\'+fname+'\\Diffusion\\Diffusion_'
                 +fname+'_'+measname+'.svg',format='svg',dpi=1200)

    plt.close()

    with open(filedir+'\\Data\\'+fname+'\\Diffusion\\Diffusion_'
              +fname+'_'+measname+'.txt', 'w') as f:

        f.write('Time (us)\tDiffusion (um^2/us)\n')

        for j in range(len(diffusion)):
```

```
            f.write("%f\t%.10f\n" % (time[j],diffusion[j]))
        f.close()
def getCoordinates(fname, filedir):
    '''Function that reads a file and...'''
    global cutoff1, cutoff2
    global data, time_msd, msd
    with open(filedir + '\\'+fname+'.csv',"r") as file:
        reader = csv.reader(file, delimiter=";")
        data = list(reader)
    header = data[0]
    '''It considers that the first column is the time. The second and
    following columns are data from different measurements. Therefore,
    the number of measurments is the number of columns - 1'''
    nbMeas = len(header) - 1
    if header[-1] == '':
        nbMeas = nbMeas - 1
    timeRead = False
    time_msd = list()
    msd = list()
    filenames = list()
    for i in range(nbMeas):
        fn = header[i+1] #The measurement name is in the header
        if fn in filenames:
            loop = True
            aux = 0
            while loop:
```

```
                #This loops deals with measurements having the same name.
                #If the current filename occurs more than once (it's already
                #in the list "filenames") a parenthesis with a number is added
                #at the end. The number inside the parenthesis is also
                #iterated starting from 1 until the filename is not repeated

                repeated = False

                for k in range(len(filenames)):

                    if aux > 0:

                        fn = header[i+1]+'('+str(aux)+')'

                    if fn == filenames[k]:

                        aux += 1

                        repeated = True

                        break

                if not repeated:

                    loop = False

        #The filename (with or without number in parenthesis) is saved in the
        #filenames list

        filenames.append(fn)

        try:

            with open(filedir + '\\Data\\'+fname+'\\MSD\\MSD_'+fname+'_'
                      +fn+'.txt','r') as f:

                data = [row for row in csv.reader(f,delimiter='\t')]

#                print(data[1])

                if timeRead == False:

                    time_msd = [float(data[j][0]) for j in range(1,len(data))]

                    timeRead = True

                msd.append([float(data[j][1]) for j in range(1,len(data))])
#                print(time_msd)
        except:

            '''Need to deal with this exception'''

            print('asf')
```

```
    '''Allows showing figures'''
    plt.switch_backend('Qt5Agg')
    '''Creates figure'''
    fig = plt.figure()
    #All the MSDs are plotted in the same file, so that the user can select
    #the cut-offs to perform the fittings
    for i in range(nbMeas):
        try:
            plt.plot(time_msd,msd[i])
        except:
            '''Need to deal with exception'''
            print('adsf')
    plt.xscale('log')
    plt.yscale('log')
#    plt.title(header[i+1])
    plt.xlabel('Time ($\mu$s)')
    plt.ylabel('MSD ($\mu$m$^2$)')
    #The cut-offs are set by default to the whole MSD region
    cutoff1 = np.min(time_msd)
    cutoff2 = np.max(time_msd)
#    line, = ax.plot([0], [0])  # empty line
    v1 = plt.axvline(cutoff1, color='g', linestyle='dashed')
    v2 = plt.axvline(cutoff2, color='g', linestyle='dashed')
#    plt.ion()
    #A LineBuilder cass object is created
    #The cut-offs are selected by the user by pressing and releasing the mouse.
    #Once they are selected, pressing ESC exits the plot and the cut-offs
    #are stored in index1 and index2
    linebuilder = LineBuilder(v1,v2)
```

```
index1 = min(range(len(time_msd)), key=lambda i: abs(time_msd[i]
                                                     -linebuilder.minValue))

index2 = min(range(len(time_msd)), key=lambda i: abs(time_msd[i]
                                                     -linebuilder.maxValue))

'''Hides plots again'''

plt.switch_backend('agg')

for i in range(nbMeas):

    getDiffusion(time_msd[index1:index2],msd[i][index1:index2],
                 fname, filedir, header[i+1])

alphas = list()

diffusions = list()

for i in range(nbMeas):

    #The MSD from the region indicated by index1 and index2 is fit,
    #and the parameters extracted

    alpha, D = fitMSD(time_msd[index1:index2],msd[i][index1:index2],
                      filenames[i])

    alphas.append(alpha)

    diffusions.append(D)

'''Makes summary file of MSD fittings'''

with open(filedir + '\\Data\\'+fname+'\\MSD\\MSDFitting_'+fname
          +'_summary.csv', 'w') as f:

    f.write(',')

    for i in range(nbMeas):

        f.write(filenames[i]+',')

    f.write('\nAlpha,')

    for i in range(nbMeas):

        if alphas[i] == None:

            f.write('-,')

        else:

            f.write('%.6f,' % (alphas[i]))
```

```
        f.write('\nD (um^2/s),')
        for i in range(nbMeas):
            if diffusions[i] == None:
                f.write('-,')
            else:
                f.write('%.6f,' % (diffusions[i]))
        f.write('\n\n,Mean,STD,SEM,n\n')
        nD = len(diffusions) - np.count_nonzero(np.isnan(diffusions))
        f.write('D (um^2/s),%.6f,%.6f,%.6f,%d\n' % (np.nanmean(diffusions),
                            np.nanstd(diffusions,ddof=1),
                            np.nanstd(diffusions,ddof=1)/np.sqrt(nD),
                            nD))
        nAlpha = len(alphas) - np.count_nonzero(np.isnan(alphas))
        f.write('Alpha,%.6f,%.6f,%.6f,%d\n' % (np.nanmean(alphas),
                            np.nanstd(alphas,ddof=1),
                            np.nanstd(alphas,ddof=1)/np.sqrt(nAlpha),
                            nAlpha))
#       f.write('Alpha\tDiffusion coefficient\tInitial cutoff (us)\tFinal
#                                                           cutoff (us)\n')


#           f.write("%f\t%f\t%f\t%f\n" % (alpha, D,time_msd[0], time_msd[-1]))
        f.close()
if __name__ == "__main__":
    '''Main function. The easygui module lets the user open a .csv file
    to extract the data'''
    filepath = easygui.fileopenbox()
    filedir = os.path.splitext(filepath)[0]
```

```
fname = filedir.split('\\')[-1]
filedir = filedir.split(fname)[0]
#    if not os.path.exists(filedir + '\\Data\\'+fname+'\\'):
#        os.makedirs(filedir + '\\Data\\'+fname+'\\')
getCoordinates(fname, filedir)
```

References

[1] IBEC website: http://www.ibecbarcelona.eu/

[2] Tania Patiño, Xavier Arqué, Rafael Mestre, Lucas Palacios, and Samuel Sánchez *Accounts of Chemical Research* **2018** *51* (11), 2662-2671 DOI: 10.1021/acs.accounts.8b00288

[3] Jonathan R. Howse, Richard A. L. Jones, Anthony J. Ryan, Tim Gough, Reza Vafabakhsh, and Ramin Golestanian Phys. Rev. Lett. 99, 048102 – 27 July 2007

[4] Hortelão, A. C., Patiño, T., Perez-Jiménez, A., Blanco, À., Sánchez, S., *Adv. Funct. Mater.* 2018, 28, 1705086. https://doi.org/10.1002/adfm.201705086

[5] Datt, C., Natale, G., Hatzikiriakos, S., & Elfring, G. (2017). An active particle in a complex fluid. *Journal of Fluid Mechanics, 823*, 675-688. doi:10.1017/jfm.2017.353

[6] Nam, L.; Coll, C.; Erthal, L.C.S.; De la Torre, C.; Serrano, D.; Martínez-Máñez, R.; Santos-Martínez, M.J.; Ruiz-Hernández, E. Drug Delivery Nanosystems for the Localized Treatment of Glioblastoma Multiforme. *Materials* **2018**, *11*, 779.

[7] WYATT TECHNOLOGY's Mobius: https://www.wyatt.com/products/instruments/mobius-zeta-potential-detector.html#mobius-1

[8] Autocorrelation function: https://weitzlab.seas.harvard.edu/files/weitzlab/files/dynamiclightscattering.pdf

[9] TA instrument-Discovery Hybrid Rheometer HR-2: https://www.tainstruments.com/dhr-2/

[10] Macrorheology: https://wiki.anton-paar.com/en/basics-of-rheology/

[11] Scanning Electron Microscope NOVA NanoSEM230: www.ibecbarcelona.eu/wp-content/uploads/2018/12/SEM.pdf

[12] Scanning Electron Microscopy: https://www.nanoscience.com/techniques/scanning-electron-microscopy/

[13] Genesis Pilot Lyophilizer: https://www.spscientific.com/Products/Freeze_Dryers_/_Lyophilizers/VirTis/Floor_Model_Tray/Pilot_Lyophilizers/Genesis_Pilot_Lyophilizer/