Towards nanoparticle rational design: a multi-scale simulation investigation of the molecular mechanisms underpinning polymer based drug-delivery vehicles

Lopez-Rios De Castro, Raquel

Awarding institution:
King's College London

Towards nanoparticle rational design: a multi-scale simulation investigation of the molecular mechanisms underpinning polymer based drug-delivery vehicles

Raquel López-Ríos de Castro

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy in Computational Biophysics

of

King's College London.

Department of Chemistry and Department of Physics

King's College London

March 28, 2024

Abstract

Polymer-based nanoparticles (PNP) are receiving increasing attention as potential cancer therapeutics to replace conventional cancer treatments. What differentiates PNP from current cancer therapeutics is their specificity towards cancer cells, which, among many advantages, increases the efficacy of the drug, lowers systemic toxicity, and allows for higher tolerable doses. Despite these advantageous characteristics, PNP still have a long way to go before becoming standardized cancer therapeutic delivery vehicles, as their mechanisms of action remain unknown. This thesis employs a multi-scale simulation approach to elucidate the mechanisms of action of PNP formed by amphiphilic block co-polymers. In particular, all-atom (AA) and coarse-grain (CG) simulations are used to understand key processes of PNP that will contribute to their application as cancer therapeutics, such as their self-assembly process, drug encapsulation, and selectivity towards cancer cells. These processes are extremely dynamic, making them either impossible or very hard to study using experimental techniques. The motivation behind this thesis is to contribute to the knowledge of the rational design of PNP and to create platforms to standardize and automate the analysis of PNP simulations, enabling scientists to apply these methods to any other PNP of interest.

A fundamental problem in PNP rational design is to deduce the overall micelle characteristics from the individual polymers, which are normally not the same. To address this, in this thesis AA MD simulations of the same PEO-PMA polymers (same monomer and polymer numbers) but arranged in a different topology were performed. Here, it was demonstrated that polymer topology plays a key role in the overall micelle physical characteristics, drawing a link between topologies and specific micelle characteristics such as size or hydration. It was also shown that polymers that form a micelle with a clear hydrophobic core and hydrophilic corona, adopt location-specific polymer conformations. Furthermore, in this thesis, CG simulations were conducted on an experimentally validated PEG-PLGA NP loaded with anti-cancer peptides, aiming to understand its cargo encapsulation and selectivity toward cancer cells. Regarding drug storage, it was shown that the polymers in the PEG-PLGA NP also take location specific conformation, and that these conformations form local microenvironments within the NP that lead to the solubilization of the peptide in several storage locations. Additionally, to study the experimentally validated selectivity of this NP towards cancer cells, the NP was simulated with a model cancer and healthy membrane. From these simulations, the changes induced by the NP-membrane interactions on both, the NP physicochemical characteristics and membrane disruption, were quantified and compared. It was found that changes in both -NP and membrane- were more substantial in the cancer simulation, meaning that the selectivity of this NP could also be observed *in silico*. The key force driving the NP-membrane interactions was the interactions between the PEO polymers and a specific lipid species, that is present in higher percentages in cancer membranes. This suggests that preferential polymer-lipids interactions may play a vital role in the PNP selectivity towards cancer cells. Moreover, a set of parameters to assess the selectivity of PNP towards cancer cells *in silico* are proposed, which can be applied to other NP-membrane systems.

To conduct such a detailed analysis of polymer systems, it was necessary to develop novel analysis tools. To this end, a graph theoretical cluster algorithm was developed to track changes in polymer aggregation throughout a simulation. Also, to study the specific polymer conformations within the micelle, dimensionality reduction unsupervised machine learning and clustering techniques were applied. Finally, to be able to analyse simulation properly, an algorithm to make molecular structures whole across the periodic boundary when its size is greater than half the box size was created. These analysis tools, along with others developed in this thesis, have been incorporated into the publicly available PySoftK software package. This way, PySoftK aims to provide an automated computational analysis workflow to study complex properties of soft-matter.

Acknowledgements

My PhD journey began in September 2019, a mere six months before the onset of a global pandemic. This marked the commencement of what was, for many of us, our inaugural experience with lockdowns. Personally, I was pretty much locked down for almost two years, the first half of my PhD. However, despite spending the first two years of my PhD living within a flat, followed by the stress of catching up with work upon my return to London, I have loved my PhD and I would not change a thing (except of course, avoiding the circumstances of a pandemic). The people that I am acknowledging in this section are the reason why I have enjoyed every single moment of the last four years, irrespective of the challenges.

The first person I would like to thank is Prof Christian. D. Lorenz, my PhD supervisor, mentor, and who I hope will be a life-long collaborator and friend. Chris, you have given me strength to believe in my own ideas and to pursue every goal. But most importantly, you have taught me how to be resilient in the face of failure. I hold deep admiration for you and I really hope that I get to be an academic like you are. I also want to thank my other PhD supervisor, Prof. Martin B. Ulmschneider. Martin, your steadfast support for all of my ideas has been instrumental in my scientific journey. From you, I've also learned the importance of maintaining a positive outlook even in challenging situations, and I will forever cherish this lesson.

Another reason why I loved my PhD, is because I had the privilege to work and learn from two of the most brilliant individuals I know: Rob M. Ziolek and Alejandro Santana-Bonilla. Collaborating with both of you (and hanging out together) has been a genuine pleasure. Alejandro, I will always be thankful for you teaching me how to code properly. Rob, you have shaped my scientific thinking. Your unwavering support, life and work advice, offered at any hour, are treasures I hold dear. My gratitude for this is boundless. Furthermore, I extend my thanks to all the members of both my labs: Dominique, Eddie, Alice, Miruna, Joanna, Zhiwen, Javier, Alex and Melissa. Creating a sense of belonging in a virtual environment is no small feat, but thanks to the collective effort we invested, I believe we successfully fostered that group camaraderie.

I must also acknowledge four young scientists who have been a constant source of inspiration throughout my PhD journey, without whom these years would have not been the same. First, I want to thank Mahnoor. You were the first student I supervised, and now you are teaching me, your passion for science is contagious. It is thanks to you (and our pipetting strategies) that I endured the challenges of experimental work. I am thankful that our collaboration has grown into a friendship. I also want to thank Favour, Ashfeen and Oluebube. While I initially guided your transition into the PhD journey, towards the end, you became my steadfast support system, you were the best part of going in to work. The resources I was born with ease my scientific journey, that is why I greatly admire your resilience and determination, as you've navigated a path in science that, while not without its difficulties, you have truly fought for and owned. The four of you are all exceptional scientists, and I am deeply grateful to witness your future successes, as Little Simz puts it,"Woman to woman, I just want to see you glow".

No matter how much I want to work in academia to live a less 'capitalistic' life, research is not possible without funding. Hence, I express my gratitude to BBSRC and my PhD program, LIDo DTP, for their financial support and for fostering a strong sense of cohort unity.

For the last year, I have been talking a lot to a neuroscientist. So, now I know better than ever the impact of our relationships on us. Therefore, my enjoyment of my PhD isn't solely attributed to my work, but to the people that have been with me, that I have met and that I have lost along the way. A world-wide pandemic has been manageable thanks to you and I would like to acknowledge some of this people.

First, I want to thank my friends from Vera, particularly María, Elina, Gonzalo, Mario, Arancha, Khalifa, Marta, Bea, Maria (R), Olea, Clara and Jorge. I feel like no one believes in me as much as you do, which makes me feel very loved and gives me strength. Also, thank you for showing me that ‘having fun’ is also necessary (specially when you are stuck with a problem), you are definitely the people I have the most fun with.

I would also want to thank my two best friends, Robert and Elena (Kori). Robert, thank you for our enriching conversations and for teaching me a more stoic way of life, which I really needed during these past years. Kori, everything I am thankful for about our relationship cannot be summarised in a few sentences, so let me just say that I feel extremely lucky to have crossed paths with you. From the LIDo induction day, until now, you have become my best friend.

I also want to thank Iván, the neuroscientist I met along this journey. Thank you for helping me rediscover life after a two-year lockdown. I am very excited about the next Chapter ahead of us (one that extends beyond the confines of my thesis Chapters).

Finally, I want to thank my family, with who I survived the global pandemic. To my parents, I am very thankful for your trust in my choices, even when they led me down unknown paths for you. Dad and Mum, you’ve taught me both, strength and sensitivity, shaping who I am today. I also want to thank my grandma, thanks to you I am both, a semi-professional checkers players and a great chef, since from you I learnt that ‘there is no such thing as too much garlic powder’. Nala, your unconditional love kept me grounded during the pandemic and you watching me write my thesis made the journey less lonely. Pedro, thank you for your candid life advice, even when impartiality isn’t your preference. I consider myself fortunate to have you as my confidant. Juan, you are the second funniest person I know, thank you for making me laugh, even in moments when I thought I didn’t want to. Your trust in me bolsters my self-confidence. And last, but certainly not least, my sister Lucia (the funniest person I know). Despite being my younger sister, your influence has helped me grow and made me a happier person.

List of Publications

The following articles were prepared during the course of this PhD. Work is included within this thesis from those highlighted [∗].

- [∗] **López-Ríos De Castro, R.**, Ziolek, R. and Lorenz, C.D. 'Topology-Controlled Self-Assembly of Amphiphilic Block Copolymers', *Nanoscale*, **2023**.

- Santana-Bonilla, A., **López-Ríos De Castro, R.**, Sun, P., Ziolek, R.M. and Lorenz, C.D. 'Modular Software for Generating and Modeling Diverse Polymer Databases'. *Journal of Chemical Information and Modeling*, **2023**.

- Ziolek, R.M., Santana-Bonilla, A., **López-Ríos De Castro, R.**, Kuhn, R., Green, M. and Lorenz, C.D. 'Conformational heterogeneity and interchain percolation revealed in an amorphous conjugated polymer'. *ACS nano*, **2022**, 16(9), pp.14432-14442.

- Bjerrum, E. J., Margreitter, C., Blaschke, T., Kolarova, S., & **López-Ríos De Castro, R.** 'Faster and more diverse de novo molecular optimization with double-loop reinforcement learning using augmented SMILES', *Journal of Computer-Aided Molecular Design*, **2023**, 1-22.

- Paez-Perez, M., Vyšniauskas, A., López-Duarte, I., Lafarge, E.J., **López-Ríos De Castro, R.**, Marques, C.M., Schroder, A.P., Muller, P., Lorenz, C.D., Brooks, N.J. and Kuimova, M.K., 'Directly imaging emergence of phase sep-

aration in peroxidized lipid membranes'. *Communications Chemistry 6.1* **2023**, 6(1),15.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation of the thesis

Polymer-based nanoparticles (PNP) are promising candidates to replace current cancer therapies, primarily due to their high specificity toward cancer cells, which reduces overall systemic toxicity. However, there has been limited clinical translation of NP-based cancer therapy, mainly because our understanding of their mechanisms of action remains incomplete. This includes aspects ranging from predicting PNP cargo storage location to their selective targeting of tumorous tissue. The results presented in this thesis aim to provide new insights into the molecular mechanisms by which PNP self-assemble, store their cargo, and deliver therapeutics targeting cancer cells over healthy cells. This has been achieved by a multi-scale study which combines all-atomistic and coarse-grained molecular dynamics simulations, combined with a wide variety of novel analysis techniques that utilise unsupervised machine learning and graph theory. These analysis tools have been included into the newly published software package, PySoftK, so that the entire soft-matter community can benefit from their development. Overall, the work presented in this thesis proposes new parameters to consider for the rational design of PNP, new key interactions between PNP and cancer membranes to assess PNP preference toward cancer cell lines (which are vital for the *in silico* screening of NP) and new analysis techniques to better understand and quantify the physicochemical characteristics of polymer-based PNP.

1.2 Cancer Nanotechnology

1.2.1 Cancer and current therapies

There are over 100 different types of cancers, but they are all characterised by anomalous and uncontrolled cell growth with the ability to invade other tissues [3]. This unrestrained replication occurs due to mutation of genes that alter the replication and death rate of cells. Most of these mutations are not inherited, but are caused by DNA damage, such as exposure to UV radiation or endogenous errors in DNA replication [4]. The scientific journey of finding therapies to cure cancer started back in the early XX century [5]. Only 50 years later, the first nation-wide effort to tackle this disease materialized with the creation of the Cancer Chemotherapy National Service Center in the USA [6]. However, more than 100 years later, cancer is still the second leading cause of death world-wide, causing one of every six deaths in 2008 [7]. Also, it is estimated that in 2020 in the US alone, $1,806,590$ people were diagnosed with cancer, and in the same year there were more than 600 thousand deaths [8] due to this disease. Furthermore, global demographics characteristics point towards a steep increase in cancer incidences within the next decades. It is estimated that by 2025 more than 20 million cancer cases will be diagnosed annually worldwide [9]. Figure 1.1 displays the statistics for the most common types of cancer deaths for women and men in 2023 in the USA.

The main reason why the mortality of cancer remains high is due to the tendency of cancer to spread across the body (moving away from the primary tumour), also known as metastasis, until it becomes incurable. Metastasis is the cause of 90% of deaths from solid tumours [10]. Furthermore, the elevated mortality is a result of the limited treatment options for cancers that are inoperable or do not respond to targeted or hormonal therapy. For example, one of the cancers with the highest mortality is lung cancer [11], which is often challenging to operate. Lung cancer is the first leading cause of cancer death for men and women in the

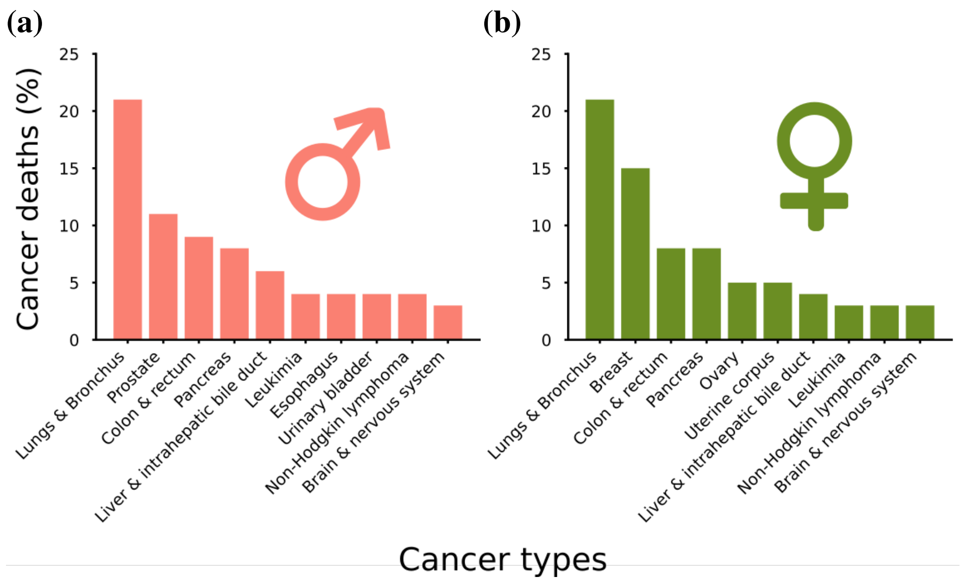


Figure 1.1: Ten leading cancer types for new deaths in the USA in 2023. (a) For men and (b) for women. Statistics obtained from Siegel *et al.*[1]

U.S.A [1]. For these types of cancer and for any advanced cancer, the most common treatment option is a combination of surgery, radiation and/or chemotherapy [3, 12].

Surgery aims to remove the tumorous tissue, while radiation therapy uses a targeted ionising radiation on the cancerous tissue either to directly kill the cancer cells or to induce genetic changes that lead to cell death [13]. Even though radiation therapy tries to minimize the damage to the surrounding healthy tissue, normally the radiation dose required to achieve sufficient cancer cell death exceeds the dose tolerated by the healthy tissue [14]. Chemotherapy consists of the administration of cytotoxic chemicals to eradicate the cancer cells, or reduce the tumour. These cytotoxic chemicals are normally administered to the patient in combinations of several drugs to avoid tumour cell resistance and to try to reduce the toxicity to healthy cells [15]. However, chemotherapy is not typically curative, as it has low specificity towards cancer cells [15, 16, 17], limiting the maximum delivery dose tolerable for the patient. It is also common to combine these therapies with cancer inmunotherapy, which aims to eradicate the disseminated tumor cells in the blood stream and organs to reduce the overall risk of distant metastases [12]. This is achieved by stimulating the patient's immune system. Unfortunately, only a small amount of patients respond to this type of treatment, since many types of tumors quickly develop drug resistance [18]. Thus, there is a critical need to find therapeutics with high selectivity towards cancer cells [19, 20, 21] to reduce the side-effects of cancer treatment and to increase the maximum tolerated drug dose. These therapeutics would improve the patient's quality of life during treatment and also boost the chances of total recovery from cancer for the patient.

1.2.2 Introduction to Cancer Nanotechnology

Nanotechnology is an interdisciplinary field dedicated to creating and manipulating novel products at the nano-scale level [20, 21], encompassing biology, physics, chemistry, engineering, and medicine. In the last 30 years, nanomedicine has attracted a lot of attention as a potential candidate for cancer therapy [19, 20]. Nanotechnology for cancer involves the use of biocompatible semiconductors, metals and polymeric particles with novel and complex optical, electronic, magnetic and structural properties that cannot be achieved with individual molecules or bulk solids [22]. These properties are pivotal for applications in cancer detection, diagnosis and treatment [20].

There are many different nano-structures that are used in cancer nanotechnology, such as liposomes; which are the simplest nanovector and are used to encapsulate therapeutics [20, 23]. There are already approved liposome-based cancer therapies. For example, Doxil, which is doxorubicin encapsulated PEGylated nanoliposomes [20, 24]. Another example of cancer nanotechnology devices are carbon nanotubes, these are carbon cylinders composed of benzene rings mainly used for high-specificity sensing of DNA and proteins [20, 23].
One of the nano-structure that has attracted the most attention as a potential cancer drug delivery vehicle, and that will be the main focus of this thesis are nanoparticles (NP) [19]. NP are submicron-sized particles that are able to encapsulate and transport therapeutics [20]. NP have several characteristics that make them ideal delivery vehicles for cancer therapeutics: the ability to transport drugs across biologial

barriers; such as the blood-brain barrier [20], the increased circulation time of the encapsulated drug [25], the enhanced protection of the drug from bio-degradation, the capacity to deliver a large drug concentration in a specific site [17], and in the case of polymeric NP the biodegradability of their components [26]. Nevertheless, the most attractive characteristics of NP as potential cancer therapeutics is their potential to be engineered to posses high specificity towards cancer cells, resulting in lower toxicity [19, 23].

1.2.2.1 Polymeric nanoparticles for cancer cell delivery

Polymeric NP (PNP) are biodegradable and biocompatible polymeric colloidal particles with submicron diameters [19, 27]. These NP are able to carry drugs within their polymeric matrix or have therapeutics adsorbed or conjugated onto their surface [19, 20]. Furthermore, polymeric NP are capable of sustained released of their cargo by tuning polymers to respond to specific cell environment changes that appear in diseased cells, such as acidic pH or high oxidative stress, which enhances tissue targeting [27, 28]. They can also be designed to deliver drugs across several biological barriers. For example, PNP coated with polysorbates have been used to deliver therapeutics into the brain across the blood-brain barrier [19, 20]. In general, polymeric NP have been shown to greatly increase the concentration of anticancer drugs in brain tumours [19]. Once the PNP has delivered its cargo, due to the biocompatibility and biodegradability of its polymeric components, the NP degrades into molecules that are harmless to the human body. Therefore, polymeric NP-based cancer therapy enhances drug specificity and reduces systemic toxicity. For illustration purposes, examples of snapshots from computer simulations of polymer-based NP are represented in Figure 1.2.

PNP can encapsulate hydrophilic and hydrophobic drugs. The polymer shell of PNP serves a dual purpose – it shields the encapsulated drugs from enzymatic degradation [27] and safeguards them against external agents within the human body. In addition to preserving drug integrity and facilitating targeted delivery, NP reduce drug clearance, which extends the circulation time of drugs. This effect is achieved through the incorporation of hydrophilic polymers like polyethylene glycol (PEG)

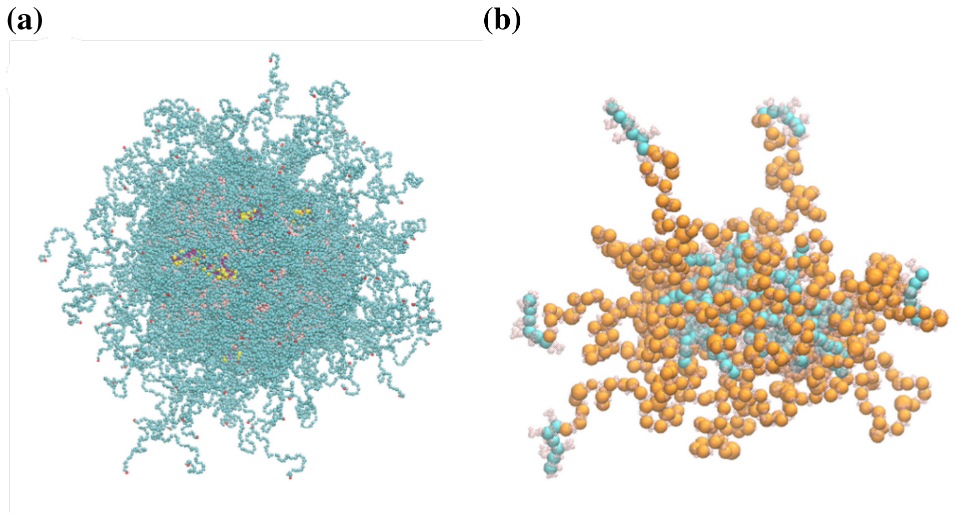


Figure 1.2: Snapshot of polymer based NP from computer simulations performed as part of this thesis. (a) Coarse-grained representation of a PEG-PLGA NP loaded with an anticancer peptide. PEG is in blue, PLGA in pink and the anticancer peptide in yellow and purple. (b) All-atomistic representation of micelle made up of PMA-PEO-PMA polymers. PMA is in sky blue and PEO in orange. Representations are not to scale

or polyvinyl alcohol (PVA), which are both non-toxic and blood-compatible. These hydrophilic polymers form a protective corona on the outer surface of the PNP, inhibiting phagocytosis by non-targeted cells [27]. This corona, primarily composed of hydrophilic polymers, envelops the hydrophobic core of the NP. Notably, several hydrophobic polymers have been extensively studied for use in NP loaded with cancer therapeutics, including poly-lactide-co-glycolide (PLGA), polyhydroxyalkanoates (PHAs), and polylactic acid (PLA) [27, 29]. Since PNP are able to deliver a targeted and controlled release of cancer therapeutic drugs, which reduces the toxicity and improves the pharmacokinetics of the cancer formulations, they are a powerful alternative to conventional cancer therapies.

1.2.2.2 PNP synthesis methods

NP synthesis typically allows for straightforward and scalable production in the laboratory, enabling the synthesis of large quantities in a single batch [19, 20, 27]. There are several methods for polymeric NP synthesis. Some of the most common methods are emulsification-solvent evaporation or extraction, double emulsion and evaporation, and precipitation [27]. Emulsion-solvent evaporation is divided

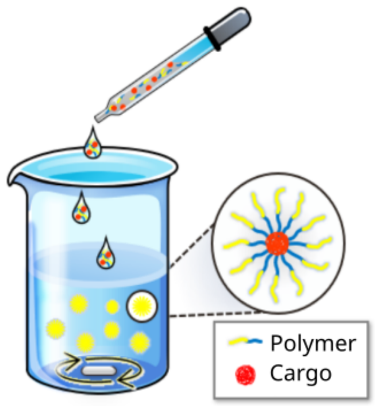


Figure 1.3: Schematic representation of PNP formation via precipitation. Schematic representation of a drug-loaded polymer-based NP formed via precipitation. Figure adapted from Chen *et al.*[2]

into two steps. The first step is the emulsification of the polymer solution into an aqueous phase. In the second step the solvent is evaporated and the polymers are precipitated as nanopsheres [27, 30]. On the other hand, the double emulsion and evaporation method is used to synthesize PNP loaded with hydrophilic drugs. This is done by adding aqueous drug solutions to the polymer solution while stirring to form an emulsion[30]. Finally, the PNP studied experimentally in this thesis were formed via precipitation. This method consists of adding drop by drop the polymer dissolved in a water-miscible solvent (acetone, ethanol, etc..), and if needed also the drug, into the aqueous phase [27, 30, 31]. A schematic representation of this method is illustrated in Figure 1.3.

1.2.3 Mechanisms behind NP cancer selectivity

As mentioned above, a major challenge in current cancer therapies is the lack of selectivity in affecting only cancer cells while sparing healthy ones. NP, including PNP, offer a solution to this problem since they can selectively target cancer cells. This not only reduces systemic toxicity, but also lowers the necessary drug dose, since the majority of the drug will be delivered to the target tissue or cells [32]. NP achieve this selective delivery to tumor tissue via passive or active targeting [19, 20, 23].

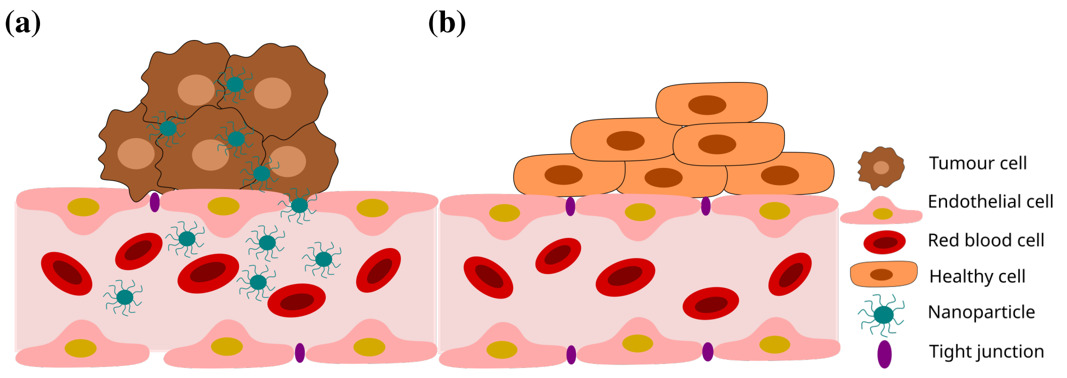


Figure 1.4: Schematic representation of the vasculature of cancer and healthy tissue. (a) Representation of the ’leaky vasculature’ of tumors. The lack of tight junctions and the bigger spacing between adjacent endothelial cells (EPR effect) allow the NP entry into tumorous tissues. (b) Representation of the vasculature of healthy cells, the spaces between the adjacent endothelial cells are quite small and with tight junctions, which hinder the entry of NP into these tissues.

1.2.3.1 Passive targeting

Passive targeting is the accumulation of the NP in the cancer tissue due to the NP physico-chemical and pharmacological properties. It has long been believed that the most important factor that affects NP passive targeting of cancer cells is the Enhanced Permeability and Retention (EPR) effect [19, 23, 27, 33], also known as the ’leaky vasculature of tumours’. The EPR effect cause NP to accumulate more in cancer tissue than in healthy tissue due to two main reasons. First, growing tumors produce endothelial growth factors that induce angiogenesis. This new vasculature has abnormally enlarged junctions (between 600 – 800 nm [20]) between adjacent endothelial cells in tumor tissue, allowing NP to easily pass through these junctions [20, 22]. Figure 1.4 shows a schematic representation of the EPR effect and also of the vasculature of healthy cells for comparison. From this Figure, it is clear that the enlarged spaces between adjacent endothelial cells favour the entry of NP into tumorous tissue. Additionally, the second reason why the EPR effect leads to a higher accumulation of NP in cancerous tissues is that tumors have a faulty lymphatic drainage system, which facilitates particle accumulation [22, 27].

Therefore, these characteristics of the vasculature of tumours lead to a higher accumulation of NP in cancer tissue than healthy tissue, increasing the bioavailability and efficacy of anti-cancer drug loaded NP [20]. Typically, after entering through the tight-junctions, NP navigate the trans-endothelial transport pathways that support tumor growth and nutrition [34, 35], allowing them to reach various regions within the tumorous tissue. Furthermore, recent research shows that the NP entry into tumours may also be enhanced by a specific type of endothelial cell found in cancer tissue [36].

Apart from the EPR effect, there are other mechanisms of passive targeting that contribute to the specificity of NP towards cancer cells. These mechanisms of selective targeting rely on the tumor microenvironment, which is different to the one of healthy cells. For example, since cancer cells rapidly divide, they have a higher metabolitic rate, needing a higher supply of oxygen and nutrients than their healthy counterparts. To obtain that extra energy, cancer cells use glycolysis. This leads to a change in the pH, making the environment more acidic [20]. Therefore, a NP that is stable at physiological pH but degrades at a lower pH, would also be a mechanism of passive targeting, since the physicochemical properties of the NP contribute to the delivery of the drug.

1.2.3.2 Active targeting

Active targeting involves the entry of NP into tumor tissue via receptor mediated endocytosis [27]. This process is achieved by functionalizing NP with attached ligands, which can be small molecules, peptides, antibodies, and more [20, 37]. When a NP is loaded with a drug and have ligands, they are referred to as 'ternary-structured NP' [20, 27]. The choice of ligand depends on various parameters, including the receptors exclusively expressed by the target cancer cells, their distribution within the cells [20, 27], and their abundance [19]. This mechanism is effective because, upon the initiation of the interaction between the ligand and receptors, the plasma membrane invaginates around the NP, forming an endosome [19, 20]. Subsequently, the endosome is transported to a specific target organelle. Examples of receptors targeted with ligands include folate receptors, epidermal growth factor receptors, and transferrin receptors [27].

1.2.3.3 Differences between cancer and healthy cells: lipidomics

As stated above, an ideal cancer therapeutic should be selective towards cancer cells even at the early stages of the disease [23], and leave healthy cells unaltered. To achieve this, it is necessary to find clear cancer biomarkers that can be targeted by anti-cancer drugs. This allows cancer therapeutics to treat the disease even at initial stages, and also reduces the damage produced to healthy cells. There are several differences between cancer and healthy cells that could be cancer biomarkers, such as the tumour microenvironment or the metabolic rate [20]. This thesis will focus on the differences in the cell membrane lipid composition across cancer and healthy cells.

Lipid metabolism can be altered by different diseases, such as diabetes, Alzheimer's disease and cancer[38]. Cancer cells have a different lipid composition than their healthy counterparts. Several studies have shown that many lipid species increase in cancer cells, for example, lysophospholipids in ovarian cancer or gylcerophospholipids in prostate cancer [39]. Therefore, lipidomics, which is the quantification of lipids confined in a biological entity [39], e.g cell membrane, is a promising cancer biomarker.

In cancer, the cell membrane of a tumor influences its ability to grow and attach to neighbouring cells, as well as its metastasis speed[3]. In general, cancer cells possess an overall negative charge due to higher percentages of negatively charged lipids such as phosphatidylserine (PS), O-glycosylated mucins, or sialyated gangliosides [40], while healthy cells tend to have a more neutral overall charge due to the zwitterionic nature of their major lipid components such as phospatodylcholine (PC) or phosphatidylethanolamine (PE). Furthermore, it is hypothesized that cancer cells exhibit greater membrane fluidity compared to their healthy counterparts. This increased fluidity, attributed to lower percentages of cholesterol in cancer membranes, may enhance the activity of certain drugs by facilitating membrane destabilization [3]. Cholesterol protects healthy cells from the insertion of therapeutics, as cholesterol makes healthy membranes less fluid and more rigid

[41, 42]. Finally, cancer lipids also contribute to making cancer cells have a larger surface area than healthy cells, increasing the possible number of contacts between therapeutics and the cancer membrane [43].

All these characteristics favour NP selectivity towards cancer cells. Furthermore, there are specific polymers that are more likely to interact with cancer cells lipids. For example, previous studies on PEG polymers, have shown the preference for PEG to interact with certain lipid types such as 1-palmitoyl-2-oleoyl-sn-glycero-3-phospho-(1'-rac-glycerol) (POPG) which are present in higher percentages in some types of cancer membranes than in healthy cell membranes [44].

1.2.4 Limitation of polymeric NP for drug delivery

Despite successful NP drug delivery to cancer cells *in vitro* and mouse models, there has been very little clinical translation [45] due to various limitations. Regarding limitations of NP synthesis, many fabrication methods for NP are only suitable for lab-scale synthesis, so better methods need to be designed for large-scale production. Also, dry forms of NP easily aggregate, making them hard to handle [46]. Moreover, there are also issues with respect to the NP distribution within the body. For example, some NP tend to accumulate in the liver, spleen and lung [46, 47]. If NP release their cargo in any of these organs, it could increase the overall systemic toxicity of NP. Other common issue when using NP as drug delivery system is the lack of EPR effect in regions with very low permeability or microvessels, making these tissues harder to target [46].

Nevertheless, one of the main reasons why NP based cancer therapy is being hindered is due to the limited understanding of NP mechanisms of action [48]. Many studies have focused on understanding the various mechanisms involved in the delivery of NP-loaded drugs into cancer cells, such as the forces that drive NP-cancer membrane interactions [35, 45, 49, 50, 51, 52], how changes in the physichochemical properties of NP [49, 53] tune their specificity towards cancer cells [37], the mechanisms behind successful NP cargo storage [54] and delivery into the cells [55]. Unfortunately, there is still no clear comprehension of these NP processes and characteristics, impeding the design of novel cancer-specific NP for efficient drug delivery into tumors. The processes mentioned above are still not fully understood because they are highly dynamic, making them challenging to study experimentally. However, a deep understanding at the molecular level is possible with molecular dynamics (MD) simulations [49, 50, 56] as it will be discussed in Section 1.4.

1.2.5 NP cargo: antimicrobial peptides as anticancer peptides

Antimicrobial peptides (AMPs) belong to the innate immune system of several species, protecting them against bacteria, fungi, protozoa and viral infections. AMPs are short proteins, normally between 5 and 40 aminoacids in length, typically positively charged, amphiphatic and normally take an α-helical or β-sheet structure when they come in contact with a cell membrane [57]. These characteristics allow them to disrupt bacterial membranes [3, 57, 58], which, among other traits, tend to be negatively charged. Thus, AMPs first became of interest as a potential alternative to antibiotics during the mid 1990s [59]. One of the mechanisms of AMPs to disturb the membrane is by pore formation. There are different types of pore that AMPs can form. The most common is known as the barrel-stave model: a highly ordered cylindrical water pore formed by trans-membrane inserted peptides. AMPs can also act via the toroidal pore model: where they absorb on the surface of the membrane and lead to membrane thinning and expansion of the head group region, inducing membrane curvature [60]. Finally, another mechanism is membrane dissolution; where AMPs assemble on the surface of the membrane and disrupt it forming a smaller pore, or lipid-peptide domain formation [3, 40, 57]. It is hypothesized that AMPs are able to overcome antimicrobial resistance because they target cells depending on their lipid membrane composition. This makes it harder for bacteria to develop resistance against them, due to the difficulty to change their membrane lipid composition.

Similarly to bacterial membranes, cancer membranes also have a more negatively charged lipid composition compared to healthy cells [40]. Therefore, some of the peptides that have been proven to be AMPs can also serve as anti-cancer peptides (ACPs) [3, 58] and most of them have either an α-helical or β-sheet secondary structure [40]. ACPs are able to kill cancer cells using the same mechanisms that they use to disrupt bacterial membranes. The main reason behind cancer cell targeting characteristic of ACPs is due to the presence of more negatively charged lipids in cancer cells membranes compared to healthy membrane, which promotes the initial electrostatic interaction between the naturally cationic ACP and the cancer membrane. Research also shows that ACPs may not only kill cancer cells by disrupting the cell membrane, but also by disrupting the membranes of organelles such as the mitochondrial membrane, inducing apoptosis [58]. ACPs are known for their low toxicity, attributed to their precise targeting and high specificity. They also exhibit excellent tumor penetration. Similar to how bacteria have lower resistance to AMPs, cancer cells show reduced resistance to ACPs, making them promising chemotherapeutics with high potential [58].

However, AMPs suffer from poor stability, solubility, and salt sensitivity in vivo, which has hindered their success in clinical trials [61, 62]. Yet, when encapsulated within PNP, their stability increases and they are protected from proteolytic degradation [62]. PNP also prevent AMPs from being recognized by the immune system, achieved by, for example, PEG polymers shielding the positive charges of AMPs [61]. Furthermore, polymeric NP increase the circulation time of AMPs in the body [61] and enable the delivery of a high dose to the target tissue [62]. The combined specificity of PNP and AMPs makes them a potential therapeutic option with high specificity and low toxicity for cancer therapy.

1.3 Polymers

Polymer science officially started in 1920s. It was Herman Staudinger who in 1922 published an article setting the basis of polymerization. He defined polymers as repeating units of small chemical units (monomers) linked via covalent bonds. In 1953 he received the Nobel prize for his discoveries on these macromolecules [63, 64]. Since then, polymers have been used for a wide variety of applications. For instance, the use of polymers like nylon, poly(methylmethacrylate), or polyvinyl chloride in surgical sutures, known as Nylon sutures, began in the mid-1940s [65]. Currently, advances in synthetic chemistry have enabled the creation of increasingly complex polymers [66]. These versatile materials are now employed in various cutting-edge applications, including sensing, soft robotics, and self-healing technologies [67].

1.3.1 Block Copolymers

Block copolymers (BCPs) have been a key research topic for many years now, mainly due to their ability to self-assemble and form complex structures [66, 68]. They are made of at least two different monomers, for instance A and B, that can be arranged in several morphologies. Examples include diblock AB, triblock ABA, alternating AB blocks (also knowns as multiblock), pentablock (ABABA) and so on [66, 68]. BCPs give raise to materials with unique interesting properties that differ from those of the individual constituent polymers.

The specific spacial arrangement of the blocks within a polymer is known as topology. Polymers with an identical number of monomer species, but differing in topology, exhibit very different behaviors, especially when they form larger structures [69]. Figure 1.5 shows polymers made up of ethylene oxide (EO) and methyl acrylate (MA), with the same number and type of monomer species, but arranged in a different topology. The most common polymer topologies are: diblock linear, which only has two types of blocks, A and B, and they are positioned in a single chain. This topology is depicted for a PEO-PMA polymer in Figure 1.5 (a). Also,

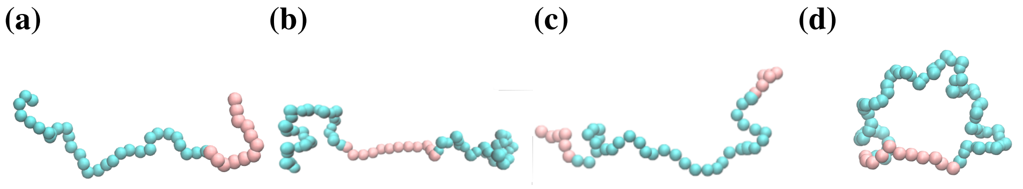


Figure 1.5: AA representation of block co-polymers with different topology. Snapshots of PEO-PMA polymers with the same number and type of monomer species, but arranged in a different topology. The topologies are (a) diblock PEO-PMA, (b) triblock PEO-PMA-PEO, (b) triblock PMA-PEO-PMA and (c) PEO-PMA ring polymer. MA is shown in pink and EO in blue. All topologies have 15 MAs and 30 EOs.

linear triblock; three blocks position wihtin a single chain, two triblock topologies (A-B-A and B-AB) are depicted in Figures 1.5 (b) and (c). Furthermore, the cyclic topology, depicted in Figure 1.5 (d), forms a ring-shaped structure within a linear chain by connecting the ends. Finally, the last most common type of topology is the branched topology; where several chains or branches extend from the main polymer backbone. Examples of branched topologies are star polymers, with multiple branches radiating out from a central core or dendritic polymers with a tree-like structure [70].

BCPs are also interesting because most of them undergo microphase separation, which consists in spontaneously forming phase-separated structure with microscopic length when in bulk or in solvent. In other words, for the case of diblock AB polymers, micro domains of A and B type are formed and arranged in a regular periodic structure [71]. The macrophase separation of BCPs depends on three parameters: the degree of polymerization (N), which is the number of a specific monomer species in a polymer. N is proportional to the molecular weight [68]. The volume fractions of the blocks (f), which is a way of defining the composition of the BCP, for the case of a diblock (AB) polymer it would be: $f_A = \frac{N_A}{N_{total}}$, such that $f_A + f_B = 1$, and where N_{total} is the total number of monomer species of the polymer of interest, and N_A is the total number of monomer species A of the same polymer. Finally, the last parameters is the Flory-Huggins parameter (χ_{AB}), which describes the thermodynamic interactions between two different monomers [68, 66] that drive phase separation. For a diblock copolymer χ_{AB} in bulk is given by:

$$\chi_{AB} = \frac{z}{k_B T}[\varepsilon_{AB} - \frac{1}{2}(\varepsilon_{AA} + \varepsilon_{BB})] \tag{1.1}$$

Where z is the number of nearest neighbours per repeat unit in the polymer, $k_B T$ is the thermal energy and $\varepsilon_{AA}, \varepsilon_{BB}$ and ε_{AB} are the interaction energies per repeat unit of A-A, B-B and A-B respectively [66]. Basically, this equation represents the exchange energy required to interchange two different monomers, divided by the thermal energy. Monomer species that have low affinity for each other, so a lower tendency for polymers to phase separate, will be given by lower values of χ and vice versa.
The degree of the microphase separation is given by the segregation product χN. For BCPs, as χN decreases or the temperature increases, the polymers become more disorder, since the combinatorial entropy increases [66]. It is important to note that when phase separation of polymers occurs in solution, Equation 1.1 gets more complex since the solvent and its interactions with the different polymer monomers need to be taken into account.

One of the block copolymers studied in this thesis is poly(ethylene oxide)-poly(methyl acrylate) (PEO-PMA), which has a diblock topology composed of two building blocks: ethylene oxide (EO) and methyl acrylate (MA). The PEO-PMA structure involves repeating units of EO followed by all MA monomers. Another block copolymer examined in this thesis, poly(ethylene glycol)-poly(lactic acid-co-glycolic acid) (PEG-PLGA), has a more complex structure, featuring alternating units of lactic acid (LA) and glycolic acid (GA) after the PEG monomers.

1.3.2 Amphiphilic polymers

Amphiphilic molecules are those that have a hydrophobic (lacking affinity for water) and hydrophilic (affinity for water) region. Therefore, an amphiphilic BCPs present both hydrophobic and hydrophilic building blocks [70]. If the ratios of

hydrophobic to hydrophilic are right and their concentration is above the critical aggregate concentration (CAC) [72], amphiphilic polymers can self-assemble in solution. The self-assembled structure can have different shapes such as spherical, cylinder-like, lamellar and vesicular [70]. Having hydrophobic and hydrophilic domains makes them ideal candidates for drug delivery, since this enables the encapsulation of hydrophobic or hydrophilic drugs within them.
The polymers PEO-PMA and PEG-PLGA are also amphiphilic polymers and they self-assemble into micelles or PNP. In the case of PEO-PMA, PEO is the hydrophilic part of the polymer since it has a high affinity for water and PMA constitues the hydrophobic block. Similarly for PEG-PLGA, PEG is the hydrophilic block while PLGA is hydrophobic.

1.3.3 Self-assembly of block copolymers

As mentioned above, amphiphilic block copolymers can self-assemble into complex structure if the concentration in solution is above the CAC. Self-assembly is the spontaneous process by which discrete components organize into a larger ordered structure, driven by their inter-molecular interactions and without any external forces. The fact that self-assembly leads to ordered structures is important, since it sets it appart from other aggregation processes that yield disordered structures, such as precipitation [73].

In the case of BCPs, the physicochemical characteristics of these higher-order structures depend on the properties of their individual components. However, the properties of the larger aggregate differ from those of the individual blocks. Consequently, deducing the properties of the larger structure from those of the individual polymers can be challenging. Nonetheless, this presents the advantage that achieving specific complex characteristics or behaviors in the self-assembled structure only requires tuning the individual polymers [73] rather than altering the overall structure.

Furthermore, polymers can be engineered to respond to external factors, such as pH or temperature, to induce structural changes in the self-assembled polymers

[74, 75]. The easy tuning combined with the large variety of morphologies that arise from self-assembled BCPs structures, and the complex physicochemical characteristics of these ensembles, place amiphiphilic BCPs at the center of contemporary polymer research.

There are two types of self-assembly: equilibrium self-assembly (ESA) and dynamic self-assembly (DySA). ESA occurs when the structure reached is in stable equilibrium, meaning that it is an entropy maximum for that system. On the other hand, DySA leads to non-equilibrium structures, so they need supply and dissipation of energy to be maintained [73]. In this thesis, we will focus on ESA, since amphiphilic BCPs follow this process. Interestingly, BCPs not only self-assemble into ordered structures, but can also undergo liquid-liquid phase (LLP) separation. LLP leads to the formation of a polymer-rich liquid phase coexisting in equilibrium with a polymer-depleted liquid phase [76]. However LLP is not very common, since amphiphilic BCPs tend to self-assemble, particularly when polymer amphiphilicity or the degree of polymerization are increased [75].

When amphiphilic BCPs self-assemble in solution, they can form structures with varying shapes. Some examples of these morphologies are spherical micelles or vesicles. A spherical micelle is an ordered spherical structure with a well-defined core consisting of the hydrophobic part of the polymer and an outer corona formed by the hydrophilic polymer species. At this point, it is necessary to make a distinction between NP and micelle. Micelles have a distinct hydrophobic-core and hydrophilic shell. NP do not have such clearly separated hydrophobic-hydrophilic regions, but tend to be homogeneous in composition throughout. Furthermore, micelles are smaller in size (0-100 nm) when compared to NP which can reach hundreds of nanometers. [77, 78] As we have discussed in the previous section, NP and micelles made of amphiphilic polymers have the potential to carry both, lipophilic and lipophobic drug, so they show great potential as drug delivery vehicles. On the other hand, BCPs can adopt a rod morphology. These are cylindrical-like micelles, with a cylindrical core and corona [66]. Rods can also be used as drug delivery

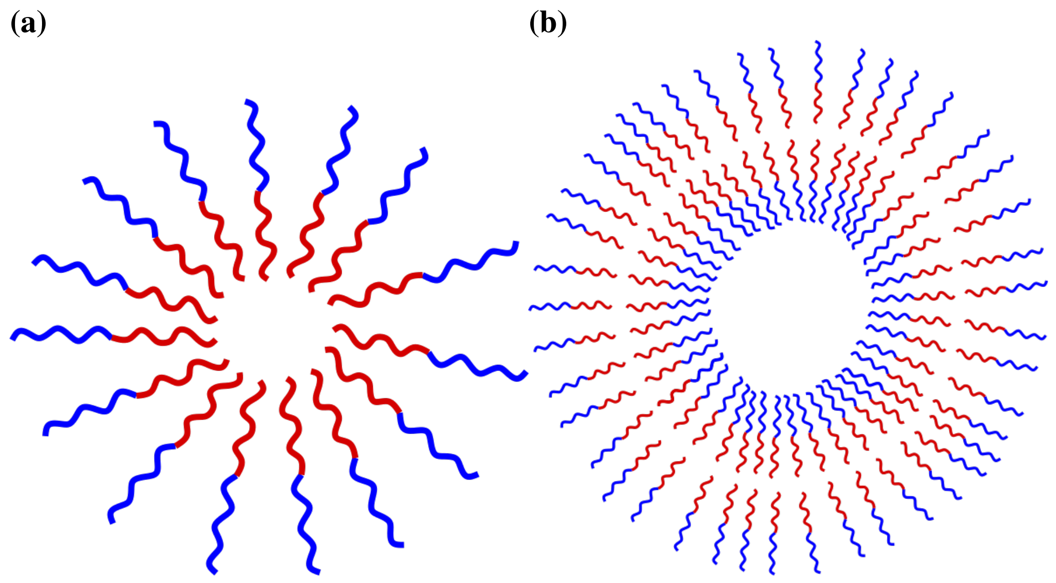


Figure 1.6: Schematic representation of self-assembled structures of amphiphilic BCPs Schematic representation of (a) micelle and (b) vesicle. The hydrophobic block of the polymer is represented in red and the hydrophilic in blue.

vehicles, since their cylindrical shape allows them to encapsulate therapeutics and enhances endocytosis [79]. Finally, another morphology are vesicles. These are hollow spheres formed by a bilayer wall. In vesicles, polymers arrange such that the hydrophilic part is at the external interfaces of the wall, and the hydrophobic blocks in the inside. Schematic representations of a NP and a vesicle formed by BCPs are depicted in Figure 1.6.

The theory underlying ESA self-assembly of amphiphilic BCPs is a combination of intermolecular forces and energetic considerations, leading to the formation of the above mentioned nanoarchitectures. As mentioned previously, entropy gain is a key factor in this process. The increase in entropy allows polymers to organize into distinct phases, enabling them to explore a vast number of microstates and spontaneously create ordered structures [80]. Additionally, the hydrophobic effect also has a crucial role in the self-assembly of amphiphilic BCPs. Hydrophobic segments tend to minimize their exposure to the aqueous environment, leading to the segregation of these blocks into hydrophobic-rich domains. Conversely, hydrophilic segments interact with the surrounding solvent and position themselves in hydrophilic-rich domains. Precisely controlling this effect is essential for the formation of nanoparticles and other structures. Furthermore, there are other interactions and forces that exert a smaller influence on self-assembly, such as Van der Waals interactions, which help stabilize neighboring polymer chains within the self-assembled domains [80]. Other interactions are hydrogen bonds between polymer segments, which can shape secondary structures and impact the overall architecture. Also, electrostatic interactions, arising from interactions between polar or charged segments, can also influence how polymers arrange themselves during self-assembly. Furthermore, thermal fluctuations in energy, resulting from temperature changes, affect the equilibrium between the self-assembled components. Lastly, chemical specificity enables precise coordination and alignment among polymer segments. It is the combined action of these forces that allows amphiphilic BCPs to spontaneously self-assemble into complex nanostructures.

As mentioned before, the hydrophobic effect is key for the nanostructures that arise from the self-assembly of BCPs, since it influences the packing of the polymer chains. Because of this, predicting the morphology of a self-assembled nanostructure is possible by calculating its "critical packing parameter" (*CPP*). The *CPP* describes the balance between the hydrophobic volume, hydrophilic surface area, and hydrophobic tail length of amphiphilic molecules, and how this influences their self-assembly into various structures. The *CPP* is given by the following formula:

$$CPP = \frac{V}{a_o l_c} \tag{1.2}$$

Here, V is the volume of the hydrophobic chains, a_o is the surface of the head group and l_c is the length of the hydrophobic tail. The *CPP* value determines the morphology of the self-assembled structure in the following way [81]:

$$
\begin{aligned}
&\text{if } 0 < CPP \leq \frac{1}{3} \ : \text{Spherical micelle}\\
&\text{if } \frac{1}{3} < CPP \leq \frac{1}{2} \ : \text{Cylindrical micelle}\\
&\text{if } \frac{1}{2} < CPP \leq 1 \ : \text{Vesicle}
\end{aligned}
$$

So for polymers with the same hydrophobic chain length, a big headgroup would mean that they self-assemble into spherical micelles, while a smaller headgroup will lead to vesicles.

BCPs are not the only molecules that self-assemble, surfactants and lipids can also self-assemble and form different structures. Nevertheless, amphiphilic BCPs are the preferred material for self-assembly due to four main reasons. First, BCPs allow the creation of structures within a wide nanometer range just by changing parameters such as molecular weight or monomer structure. Secondly, it is easy to control the morphology of BCPs architectures, since just by altering the polymer topology we can get different morphologies. Furthermore, also thanks to the development of synthetic chemistry techniques, we have a precise control over polymer synthesis, so we can tune individual polymers to adopt specific properties. Finally, self-assembled BCPs structure also benefit from the advantages of polymers over other materials, such as their flexibility, cost effectiveness, toughness and permeability control [68].

1.4 MD simulations for the study of polymer NP

In 1977 the first molecular dynamics (MD) simulation of a biomolecule was performed. This simulation consisted of just one globular protein and it ran for 0.5 ps [82]. Thirty five years later, MD allows us to simulate objects in the nanometer range for up to hundreds of micro seconds, primarily due to advancements in computational power [83]. As a result, MD is now a simulation technique that allows us to understand biological processes at the molecular scale, providing insight at a level that currently no experimental technique can. For example, crystallography and nuclear magnetic resonance (NMR) techniques can determine structural characteristics of biomolecules [83, 84]. However, MD can capture the dynamics and kinetics between the different atoms involved in the biological process being simulated, by using simple approximations used in classical physics [84]. Therefore, MD simulations are a powerful tool to understand the molecular mechanisms behind polymeric NP formation, cargo loading and the mechanisms of action behind PNP drug delivery to cancer cells. Nevertheless, it is important to note that experiments and MD simulations should be used together to advance faster in the polymeric NP field. Simulations will always need to be corroborated by experiments, and also experiments are needed to study longer and larger processes that exceed what is currently computationally possible. Similarly, simulations provide a zoomed-in molecular understanding and real-time picture in a way that experimental techniques cannot capture. Therefore, their combination will lead to a more complete understanding of PNP for cancer therapy.

This section explores how MD simulations have been used to study polymer self-assembly into PNP, as well as how PNP encapsulate their cargo and interact with model cancer and healthy membranes. Before diving into the details, it is necessary to differentiate two different scales for MD simulations used in this thesis, all-atom (AA) and corse-grain (CG) simulations. AA is the most traditional type of MD simulation. Here, each particle being simulated is an atom. Therefore, each force and motion of the system is calculated for every atom. The time span of AA simulations normally range between nanoseconds to a few microseconds. As AA MD simulates every particle, they are particularly good in capturing polymer conformational changes and peptide folding and partitioning into membranes [83]. On the other hand, CG MD simulations are able to capture processes in soft matter at the micrometer length scale, such as phase transformations that take longer than hundreds nanoseconds [85, 86]. For this type of phenomena, it would be impossible to compute AA simulations, due to the sheer computational load involved. In CG simulations, atoms are grouped together into a single bead or interaction site, effectively reducing the degrees of freedom in the system [86]. This simplification results in fewer computations during the simulations, as there are fewer particles and interactions to consider. As a result, CG simulations are well-suited for simulating larger systems over longer timescales. Further details on the scales of simulations can be found in Section 2.1.2. In this thesis, both methods, AA and CG MD simulations, have been used and combined to better understand the molecular mechanisms behind PNP formation and selectivity towards cancer cells.

1.4.1 MD simulations of BCP self-assembly

The physicochemical characteristics of BCP self-assembled structures can be studied experimentally with techniques such as Small-Angle X-ray Scaterring (SAXS) [87], Cryo-Transmission Electron Microscopy (Cryo-TEM) [88], Dynamic Light Scattering (DLS) [89] or Nuclear Magnetic Resonance (NMR) [90]. However, these techniques cannot be used to monitor the self-assembly process. Fortunately, classical MD simulations can track this process at an atomistic level, which currently no experimental technique can [91]. Furthermore, there have been numerous studies that have validated that the nanostructures formed by the self-assembly of specific BCPs in simulations agree in morphology, size and shape with those obtained experimentally [91, 92]. Therefore, MD simulations are a powerful tool to gain a molecular insight of the self-assembly of BCP aggregates.

There are numerous advantages of using MD simulations to study BCP self-assembly. For instance, AA simulations allow the study of packing motifs within structures, that is to say, how the polymer chains arranged within the structure and how they interact with each other. Also, it is possible to study the initial self-assembly pathways with AA simulations, which provide important information about the initial driving forces and stacking motifs of the self-assembly [91]. Finally, AA simulations facilitate the study of dynamic evolution and relaxation within idealized self-assembled structures, yielding a more realistic depiction. Some examples of AA MD simulation studies of the self-assembly of NPs are: Stipa *et al.* [29], who delved into the self-assembly of poly (lactic acid) (PLA) and PLGA polymers, shedding light onto their assembly process. In another study, Jafari *et al.* [93] examined various combinations of polyethylene glycol (PEG) and PLGA, (PEG, PEG-PLGA, PEG-PLGA-PEG, and PLGA-PEG-PLGA) to understand the differences in their packing to form NP.

Regarding CG simulations, the most clear advantage is that we can study larger systems for longer, since CG allows the study of self-assembly processes at the microsecond scale [92]. Also, due to their higher computational efficiency, CG can be used for high-throughput simulations [91], which is key in the *in silico* screening of therapeutics. Additionally, CG simulations also provide information about the packing motifs and can be used to study idealized structures. Although it is important to note that due to their lower number of degrees of freedom, the molecular picture is less detailed. Nevertheless, CG simulations have been widely used in the literature to study the self-assembly of BCPs. For example, Srinivas *et al.* [92] studied how varying the molecular weight of diblock co-polymers leads to different nanostructures. Their computational results agreed with experiments. Also, Park*et al.* [94] used CG simulations to study the self-assembly of polymers with a bottle-brush topology.

1.4.1.1 Currently available software for polymer simulations and its limitations

One of the most technically complex steps in using MD simulation to study polymer systems is obtaining the initial polymer models. This is particularly true for the set up of simulations with several types of polymers or different variations of the same polymer (topology, molecular weight, etc.) [95]. The reason is that large molecules with complex architectures are hard to parameterize and to structurally model them. Because of this, many computational platforms have been developed to try to make this process easier. For example, PolyParGen [96] is able to parameterize polymers, once the structure file is given. Nevertheless, it has limitations to which types and length of polymers it can do. Other softwares such as PolyMaps [97], MoSDeF [98], pysimm [99], RadonPy [100], CHARMM-GUI polymer builder [101] or Polymer Structure Predictor [102] also aim to generate the input structures and parameters needed for simulations. However, all of these software require a lot of chemical input information from the user, making them quite manual. Furthermore most of these software only allow the user to create homopolymers and simple topologies [95]. In order to try to fill this big gap in computational polymer research, and to provide a user-friendly platform for scientist to build, simulate and analyse complex polymer simulations, the modular python software PySoftK [95] was developed. PySoftK is presented in this thesis in Chapter 4.

1.4.2 MD simulations of NP drug encapsulation

MD simulations are valuable not only for capturing the self-assembly of BCPs into NP but also for studying the encapsulation of therapeutics within them. Experimental techniques, such as Cryo-EM [103] and Energy-Dispersive X-ray Spectroscopy (EDS) [104], are commonly used to examine the distribution of cargo and other components within NP. However, these techniques provide static snapshots of NP, (frozen samples during analysis), and lack atomistic resolution. In contrast, MD simulations offer several advantages over these methods. MD simulation enable the observation of the dynamic process of drug encapsulation, whether during NP self-assembly [93] or the adsorption onto pre-formed NP [54]. MD simulations provide a detailed, time-resolved view of drug encapsulation and behavior within NP at an atomistic level.

As mentioned earlier, NP formed by amphiphilic BCPs can encapsulate both types of drugs, hydrophobic drugs in the core and hydrophilic ones at the core-shell interface. The atomistic insights provided by MD simulations are of particular significance, as they offer essential information regarding the encapsulation process. This information includes whether a specific drug can be encapsulated and stably maintained within a NP [29], the cargo storage location of a specific drug in a particular NP [54] or how protected the cargo is from the external environment. However, there is a scarcity of computational studies focused on how experimentally validated polymer-based NP self-assemble encapsulating cancer therapeutics. This knowledge gap is addressed in Chapter 5.

1.4.3 MD simulations of NP membrane interactions

The mechanisms of action behind NP selectivity towards cancer cells are not well understood [48]. Several experimental techniques allow scientists to study how NP interact with cells. These methods include fluorescence microscopy [2], which permits real-time tracking of interactions, confocal microscopy [105], offering higher spatial resolution than fluorescence microscopy, and atomic force microscopy (AFM) [106], providing high-resolution images of nanoparticle-cell interactions, including surface interactions, mechanical changes, and potential membrane disruptions. These technique are highly valuable and can be used to study NP-membrane interactions for periods much longer than MD simulation can currently achieve, from minutes to days. However, MD simulations still present some advantages to study NP-membrane interactions over the aforementioned experimental techniques. The main three benefits are: first, as in the other cases, the atomic-level or molecular-level details of MD simulations. This level of resolution enables the study of how individual atoms, residues, and molecules interact at the interface between NP and lipid membranes, providing insights into the binding mechanisms and forces involved [107]. Secondly, MD simulations provide a mechanistic understanding, by simulating the dynamic behaviour of NP as they interact with lipid membranes. Therefore, MD simulations can elucidate key steps such as NP insertion, membrane disruption, and deformation [108], leading to a deeper mechanistic understanding of NP-membrane interactions. Finally, MD simulations of NP-membrane interactions offer predictive insights into how different physicochemical characteristics of NP affect their interactions with lipid membranes [109].

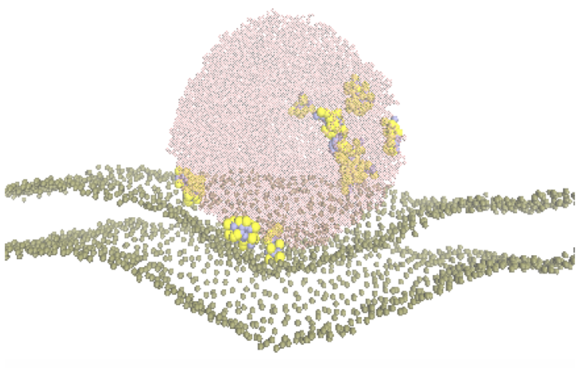


Figure 1.7: CG snapshot of a NP interacting with a cancer membrane. Snapshot of a CG simulation of a PLGA NP (pink) loaded with proteins as its cargo (yellow), interacting with a cancer membrane. For clarity, only the phosphate group of the lipids in the membrane are represented (green).

Figure 1.7 is a snapshot of an MD simulations of a CG peptide-loaded NP interacting with a cancer membrane. In this snapshot it is clear that the NP is inducing curvature on the membrane.

There are numerous studies that have utilised MD simulations to understand NP-cell interactions. For instance, Das *et al.* [110] compared AA and CG simulations of a NP with a pure DMPC (1,2-dimyristoyl-sn-glycero-3-phosphocholine) and a mixture of DMPC and DMPG (1,2-dimyristoyl-sn-glycero-3-phospho-(1-rac-glycerol)) membranes, to determine the most suitable simulation method based on the research objective. Also Ding *et al.* [51], among many other aspects, studied computationally how membrane curvature affects NP translocation. However, most MD studies aiming to understand NP-membrane interactions, use overly simplistic lipid membranes, typically composed of one, two or at most three lipids types. These reduced models overlook the significant differences in lipid composition between cancer cell membranes and those of healthy cells, as elucidated in Section 1.2.3.3. To achieve a more comprehensive understanding of the molecular mechanisms that drive PNP selectivity toward cancer cells, it is necessary to conduct simulations that incorporate complex and realistic lipid compositions, reflecting the distinct lipid profiles of cancer and healthy cells. Chapter 6 of this thesis delves into this area of study.

Chapter 2

Methods

2.1 Molecular Dynamics Simulations

Molecular Dynamics (MD) simulations are a powerful computational technique used to investigate the dynamic behavior of atoms and molecules at the molecular scale. The theoretical idea behind MD is to numerically solve the classical equations of motion for every atom or bead at each time step. The integration of the classical equations can get very complicated, since the force on each atom is the result of all the inter and intra-molecular forces that are acting upon on it, and as systems get larger the complexity of these calculations also increases. The general workflow of an MD simulation would be the following: first it is necessary to set the initial coordinates and velocities of the molecules in the system at the initial time unit or time step. Secondly, the force on each atom is calculated. Finally, each atom should move accordingly to those forces, the velocities are updated, and the simulation time advances by one time step. These three steps are repeated until the desired simulation time is reached. The resultant trajectory provides insights into the temporal evolution of molecular systems and using statistical mechanisms it is possible to calculate any thermodynamic observable. In this Section, the theoretical background of MD simulations will be discussed.

2.1.1 Newton's second law

The most accurate description of molecules is based on quantum mechanics. However, computations taking into account these effects, are very expensive as it is necessary to include additional calculations at each time step, such as electronic structure calculations. The increased computational cost limits the system size and simulation length. MD overcomes this issue by calculating the motion of atoms using a Newtonian approximation, which is less computationally intensive. This approximation provides accurate results on the dynamic and structure of molecules from their microscopic interactions. Therefore, in MD simulations Newton's second law of motion is calculated on every atom [82, 83, 84, 111], as shown in Equation 2.1:

$$\vec{F_i} = m_i \vec{a_i} = m_i \frac{d^2 \vec{r_i}}{dt^2} \tag{2.1}$$

Where $\vec{F}$ is the total force on the particle, i is the specific particle index, m is the mass of the particle, and a is the acceleration of the particle, which is the same as the second derivative of its position r with respect to time t. Note that the arrow denotes a vector quantity. The total force ($\vec{F_i}$) is calculated as the negative of the gradient of the potential:

$$\vec{F_i} = -\nabla U(\vec{r_i}) \tag{2.2}$$

Generally, the velocity ($\vec{v}$) and positions ($\vec{r}$) of any body can be obtained from its acceleration by following the classical equations of motion:

$$\vec{v_i} = \vec{v_{i0}} + \frac{1}{m} \int_0^t \vec{F_i} dt \tag{2.3}$$

$$\vec{r_i} = \vec{r_{i0}} + \int_0^t \vec{v_i} dt \tag{2.4}$$

2.1.2 Molecular Scale of Computer Simulations

According to the phenomena of interest, it may be desired for a simulation to span just a few nanoseconds with very few molecules, or to occur within the microsecond and micrometer range. Depending on the time and length scale that one wants to study, simulations can be performed at different molecular levels. Also, another

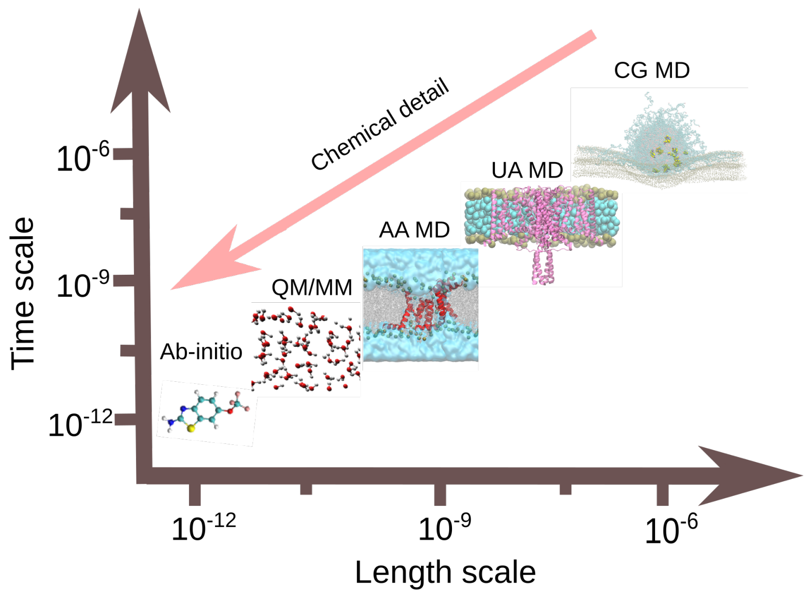


Figure 2.1: Schematic representation of the molecular scale of MD simulations. The main molecular techniques to simulate different scales are displayed in increasing order of time and space range.

aspect to take into consideration would be the computational cost of the simulation, of course it will always be more accurate to stimulate at the smaller scale, but it is currently computationally impossible to simulate large systems with significant chemical detail. There are five main scales to run MD simulations on. The different molecular scales that will be discussed in this section are depicted in Figure 2.1.

From smaller to larger ranges, the first one would be *ab initio* MD simulations, also referred to as first-principle simulations, which use quantum-mechanical descriptors of molecules without relying on empirical parameters. These simulations solve the Schrödinger equation for the electrons in the system using methods such as Hartree-Fock or density functional theory (DFT). Therefore, they provide very accurate results but are extremely computationally expensive, so it can only simulate very small systems of a few tens or hundreds of atoms and normally in the tens of picoseconds range [112]. Next would be the Quantum Mechanics-classical simulations (QM/MM), these combine *ab initio* with classical MD. Here, the electronic structure of a subset of atoms is treated with *ab initio* methods, while the remaining part of the system follows classical MD forcefields. This is normally used for systems where the quantum effects are important, but due do the size and length of the system, the rest of the atoms need to be treated classically [113]. For example, QM/MM can be used to study photo-induced electron transfer [114]. Continuing with the increasing order of the molecular scale, the next one would be all-atomistic (AA) simulations. AA is the most traditional type of MD simulations. In this type of simulation, each particle being simulated is an atom. Therefore, each force and motion of the system is calculated for every atom. The time span of AA simulations normally range between nanoseconds to few microseconds. As AA MD simulates every particle, they are particularly good in capturing molecular conformational changes and peptide folding and partitioning into membranes [83]. Chapter 3 in this thesis uses AA MD simulations. The next simulation technique is United-atom (UA) MD. This method consists of reducing the degrees of freedom of the system by grouping hydrogens with their corresponding heavy atoms into single interaction sites [115], which allows systems to run for longer compared to AA simulations. This computationally more efficient method can be used in systems where the behaviour of hydrogen atoms is not particularly important, such as lipid membranes, where the hydrogens in the hydrocarbon region have low biophysical importance [116]. Finally, the MD method that allows the simulation of larger systems for longer time scales is CG. There are processes in soft matter that require structures in the micrometer length scale, such as self-assembly of macromolecules, or phase transformations that take longer than hundreds of nanoseconds [85, 86]. For this type of phenomena, it would be impossible to use AA simulations, as the computations would be extremely resource intensive. To deal with this issue, CG models have been developed. This simulation method consists of grouping heavy atoms together into one bead or interaction site. Particularly, the forcefield MARTINI groups every four heavy atoms and their corresponding hydrogens into one bead, lowering the degrees of freedom of the system [86]. This atom grouping lessens the number of computations that will be performed during the simulations, as the number of particles is smaller and there is a smaller number of neughbouring

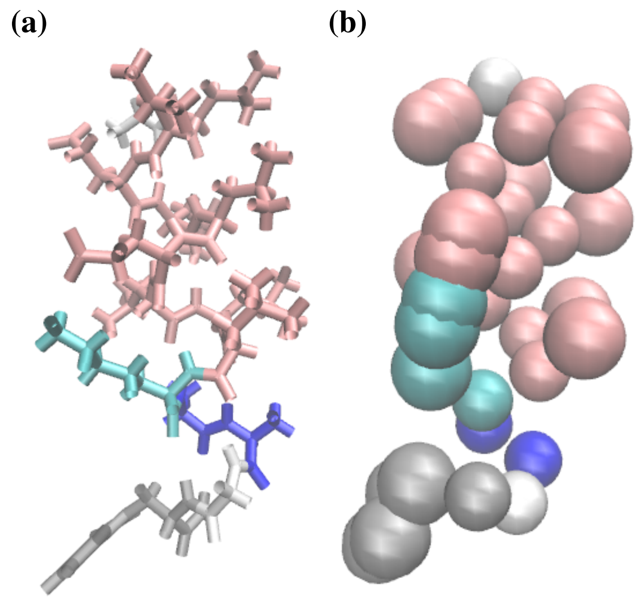


Figure 2.2: Same protein in AA and CG representation. (a) AA representation of the EEK cancer peptide. (b) CG representation of the EEK cancer peptide. The coloring of the residues is the same in both representations for a better comparison between the two. Representations are not to scale

particles to consider. For example, the aminoacid glycine which consist in 10 atoms in AA forcefields, it would be transformed into only one bead in CG forcefields. Therefore, the difference in the number of particles is of roughly of one order of magnitude between these two simulation methods. Due to this atom grouping, CG also operates on a larger time step than AA, AA simulations tend to have a time step of around 1 to 0.5 fs, while CG operates with a time step of at leas 10 fs, again the difference is of around one order of magnitude between the two. Furthermore, the reduced degrees of freedom due to the atom grouping in CG, contributes to a smooth energy landscape resulting in faster diffusion. The merger of these characteristics speed up the simulations [85]. Several simulations presented in this thesis are performed using CG MD. The conversion from AA to CG of a protein is shown in Figure 2.2.

2.1.3 System set-up

The first step to run a MD simulation is to set up the system, that is to say; to define all the molecules that will be simulated, including the solvent and ions, with their initial positions. Generating the initial system can be challenging, specially if the molecules are very large or have very complex architectures. More importantly, molecules must be placed with care, since it is always better to have an initial structure that is closed to the equilibrated state, to reach and speed up the simulation convergence and also to avoid steric clashes. A steric clash is an unphysical effect and occurs if a group of non-bonding atoms overlap or come into very close proximity within a molecular structure, that is to say, that their van der Waals radii are overlapping [117]. This must be avoided in MD simulations since it leads to unphysical behaviors and introduces artefacts in the simulations, which all lead to nonphysical values in the energy and dynamics of the system. Doing this manually can be tedious and long, so there are computational platforms that have been developed to make this process of initialising the system fast and straightforward. Depending on the type of molecules to be simulated, there are different ways to obtain a close-to-equilibrium initial structure. For example, in the case of proteins, the Protein Data Bank (http://www.rcsb.org/pdb/) contains the structural data of many macromolecules, obtained with crystallography or cryo-EM, which can then be used as starting configurations in MD simulations. In the case of lipid membranes, there are many platforms, the ones used in this thesis are CHARMM-GUI membrane builder [118] for AA MD simulations and CHARMM-GUI MARTINI maker [119] for CG membranes. They are used to create planar lipid bilayers, and the user can specify the lipid types. CHARMM-GUI works by first placing spheres mimicking the headgroup of the different phospholipids on the bilayer space, and once everything fits, then they are replaced by the equilibrated lipid structures, making sure the tails do not overlap and getting rid of any undesirable interaction. Finally, some of the polymers generated in this thesis were created using, PolyParGen [96]. In a nutshell, PolyParGen obtains the simulation input parameters for soft polymers by dividing up the polymer, acquiring the parameters of each structure and then combining all the parameters for the whole polymer [96]. However, PolyParGen has a limit on which types of topologies and length of polymers it can generate. That is why, polymers with a more complicated topology in this thesis were built with Python Scripts and Pysoftk [95]. More details on Pysoftk will be provided in Chapter 4. Alternatively, short CG polymers can also be generated with MARTINI maker [119].

2.1.4 Initialising velocities

Once the system has been created, before running any MD simulations, it is important to provide all atoms in the system with a certain initial velocity, since it would be nonphysical to have a system with no motion. The *principle of equipartition of kinetic energy* states that each momentum coordinate in the canonical phase space has $\frac{kT}{2}$ of thermal energy on average, where k is the Boltzmann constant, and T the temperature of the system. That is to say, each degree of freedom of a molecule contributes equally to the average kinetic energy. This principle is used in MD simulations to initialize the atomic velocities according to the *Maxwell-Boltzmann distribution* [120, 121]. This distribution describes the velocity of atoms in three-dimensions at equilibrium and it is given by the following formula:

$$f(v) = 4\pi \left(\frac{m}{2\pi kT}\right)^{\frac{3}{2}} v^2 e^{-\frac{mv^2}{2kT}} \tag{2.5}$$

Here, $f(v)$ is the probability density function for a velocity v, m is the mass of the particle, k is the Boltzmann constant and T is the temperature of the system. This equation shows that the velocity of particles follow a Gaussian distribution around a mean velocity at a given temperature, therefore this distribution accounts for the various kinetic energies of all particles that all together define the temperature of a system. Therefore, it is necessary to initialise the velocity of all atoms with values that will ensure that the simulation follows the *Maxwell-Bolztmann distribution*. At thermal equilibrium, this can be achieved by using the *principle of equipartition of kinetic energy*, which leads to the equation of $\frac{1}{2}mv^2 = \frac{3}{2}kT$ for a 3-dimensional system. Since in a 3-dimensional space, we can assume every atom has 3 degrees of freedom and each degree of freedom contributes $\frac{kT}{2}$ to the average kinetic energy, such that $KE = 3 \cdot \frac{1}{2}kT = \frac{3}{2}kT$. If the atoms are initialised with the previous equation, the atoms will follow the *Maxwell-Bolztmann distribution*. Note that in order to remove the system's drift (net system momentum) the initial velocities are scaled to ensure the mean kinetic energy reflects the desired temperature.

2.1.5 Integrating the equations of motion

Once the system is set up, and the initial velocities given, the computations of the velocities and positions of each atom can begin. However, it is quite complex to use Equations 2.3 and 2.4, since for systems with many particles it is impossible to solve these integrals analytically, so a different method must be used. Finite difference methods are numerical techniques used to approximate the derivatives of functions. They provide a way to numerically solve ordinary differential equations (ODEs), and it can be used in MD simulations to integrate the equations of motion over discrete finite time steps. This is achieved by discretizing time and approximating the trajectories over these small time steps. A common finite difference method integrator used in MD simulations is the velocity-Verlet algorithm [122, 123], which is based on the Taylor series expansions of the particle's trajectories. This algorithm integrates the equations of motion over a finite time interval δt, also known as the time step, following these steps:

$$\text{Update positions:} \quad \vec{r}(t+\delta t) = \vec{r}(t) + \vec{v}(t)\cdot\delta t + \frac{1}{2}\vec{a}(t)\cdot\delta t^2 \tag{2.6}$$

$$\text{Update velocities:} \quad \vec{v}(t+\frac{\delta t}{2}) = \vec{v}(t) + \frac{1}{2}\vec{a}(t)\cdot\delta t \tag{2.7}$$

$$\text{Calculate accelerations:} \quad \vec{a}(t+\delta t) = \frac{\vec{F}(\vec{r}(t+\delta t))}{m} \tag{2.8}$$

$$\text{Update velocities again:} \quad \vec{v}_(t+\delta t) = \vec{v}(t+\frac{\delta t}{2}) + \frac{1}{2}\vec{a}(t+\delta t)\cdot\delta t \tag{2.9}$$

Basically, the current force is evaluated, then $\vec{r}$ is computed for the next time step. Then, the current force term is added to the $\vec{v}$ (which updates the velocity midway through the time step). This is followed by the computation of the new force, and then the $\vec{v}$ is updated again, resulting in the velocity of the next time step. The fact that this algorithm updates the velocity at the middle of the time step improves it accuracy since it takes into account the updated positions when computing the velocities. This algorithm is second-order accurate, time-reversible and easy to implement [122].

From looking at the above equations, it is clear that choosing an appropriate δt will be key to model the correct motion of the particles. Equations 2.6-2.9 show that the velocity and accelerations are constant over the time step (δt). Therefore, the time step needs to be as large as possible so that the simulation is computationally efficient (if the time step is very small, it will take a large number of computations to reach the desired time length of the simulation) but it also needs to be short enough to produce relevant data. Normally, the time step needs to be approximately an order of magnitude smaller than the highest frequency of vibration in the system, which is C-H bonds in most atomistic systems [124].

2.1.6 Force field

Once the method for the integration of the equations of motion is set, the next step is to define the equation that will govern the forces being applied on the atoms, this equation is called the “force field”. Force fields rely on three main assumptions. First, force fields assume electronic motion can be ignored, computing only the motion for the nuclei. Therefore, atoms are treated as point particles described by classical motion, and the microscopic state of the system is described as a function of their position and momenta. This assumption is the *Born-Oppenheimer approximation* [125], which allows for the separation of nuclear and electronic motion. Since electronic motion is set to a stationary state, AA MD simulations (and larger simulation scales) can ignore the quantum mechanical calculations involving electrons, making it computationally much more efficient and simple. Furthermore, MD force fields assume that the total potential energy of a system can be written as a sum of different potentials. Because of this, the interactions between atoms are considered pairwise, and the total energy of the system is the sum of all these contributions. This approach simplifies the calculations needed to simulate many-body systems. The last assumption is transferability, which assumes that the potential energy function describing interactions for a small set of molecules can be applied to a wide range of molecules undergoing similar interactions (similar chemical groups). This means that the knowledge obtained from one simulation can be transferred to others, avoiding the need to parameterize every new system to be simulated. In classical MD force fields, the potential energy of the system is expressed as a combination of bonded and non-bonded interactions [126, 127]. The total potential energy ($U_{\text{total}}(\vec{r})$) of the simulation is obtained from the intra and inter molecular interactions of the different particles in the system as illustrated by Equation 2.10.

$$U_{\text{total}}(\vec{r}) = U_{\text{bonded}}(\vec{r}) + U_{\text{non-bonded}}(\vec{r}) \tag{2.10}$$

The bonded terms are associated with the deformation of bond and angle geometry (extension and compression of bonds and angle bending) and rotation around dihedral angles (torsions), so they cover intra-molecular interactions, while the non-bonded terms are associated to the electrostatic interactions, the dispersion interactions and the Van der Waals forces [125], covering inter-molecular interactions. These bonded and non-bonded terms are represented in Equation 2.11 and this splitting of the potentials contributing to the total energy is the same in all MD force fields:

$$\begin{aligned} U_{total} = & \underbrace{\sum_{bonds} U_{bond} + \sum_{angles} U_{angle} + \sum_{dihedrals} U_{dihedral} + \sum_{impropers} U_{impropers}}_{\text{bonded interaction}} \\ & + \underbrace{\sum_{atompair} U_{electrostatics} + \sum_{atompair} U_{LJ}}_{\text{non-bonded interaction}} \end{aligned} \tag{2.11}$$

Therefore, from Equation 2.11, the terms related to the bonds, angles, dihedrals and impropers are part of the bonded interactions, while the electrostatics and Lennard Jones potential terms are related to the non-bonded interactions. As stated above, Equation 2.11 is true for all MD force fields, however, the specific terms and functional forms to describe the bonded and non-bonded potential energy terms in various force fields can be different. In this section, I present the specific functional forms used in the MARTINI force field [126, 127], since this is the forcefield that has been used the most in the simulations presented in this thesis. Note that the type of function used to describe each term (harmonic, cosine...) is the same in all force fields, so the explanation given in this section applies to AA force fields. The MARTINI force field equation has the following form:

$$\begin{aligned}
U_{total} = & \underbrace{\sum_{bonds} \frac{K_r}{2}(r - r_{eq})^2}_{\text{covalent bonds}} + \underbrace{\sum_{angles} \frac{K_\theta}{2}[cos(\theta) - cos(\theta_{eq})]^2}_{\text{bond angles}} \\
& + \underbrace{\sum_{dihedrals} K_\phi[1 + \cos(n\phi - \phi_{eq})]}_{\text{bond dihedrals}} + \underbrace{\sum_{improper} K_d(\psi - \psi_{eq})^2}_{\text{improper dihedrals}} \\
& + \sum_{i<j} \left[\underbrace{4\varepsilon_{ij}\left[\left(\frac{\sigma_{ij}}{R_{ij}}\right)^{12} - \left(\frac{\sigma_{ij}}{R_{ij}}\right)^{6}\right]}_{\text{Lennard Jones interactions}} + \underbrace{\frac{q_i q_j}{\varepsilon_D R_{ij}}}_{\text{electrostatics}} \right]
\end{aligned} \tag{2.12}$$

where the K is the force constant of each term and the subscript eq denotes the equilibrium value of each quantity. For the sum over the energies of the bonds, r represents the intramolecular distance, θ is the intramolecular angle, ϕ is the dihedral angle and the n in the dihedral sum is the number of minima of the function. also known as the multiplicity. Likewise, ψ is the improper dihedral angle. Furthermore, for the non-bonded interactions in Equation 2.12, i and j refer to two different atoms for which the non-bonded interaction is being calculated. σ_{ij} is a Lennard-Jones parameter, which represents the distance between the two atoms at which the potential between them is zero. Similarly, ε_{ij} is the minimum potential energy, which occurs when the distance between the two particles is $r = 2^{\frac{1}{6}}\sigma$. Moreover, R_{ij} is the distance between the two atoms. For the Coulomb term, q is the partial charge of an atom and ε_D is the dielectric constant.

The parameters from Equation 2.12, such as σ, ε, the force constants K, the partial charges q and the equilibrium values are vital to reproduce real molecular behaviours. These are normally obtained from density functional theory (DFT), via experimental approaches, like spectroscopy [83, 111], or through simulations at different scales. The combination of these parameters are called a force field. For example, for AA simulations the most popular ones are: CHARMM [128], OPLS [129], GROMOS [130] and AMBER [131], which even though they have different parameters, they all give fairly similar results [111, 83]. In the next sections, Section 2.1.6.1 and 2.1.6.2 the bonded and non bonded terms will be explained in more detail.

2.1.6.1 Bonded potential

The bonded potential describes the interactions of atoms that belong to the same molecule. From Equation 2.12, it is clear that the bond term, which represents the covalent bond stretching, is modelled by a harmonic potential, and it describes the harmonic behavior of bonds around their equilibrium positions. A schematic representation of this term is depicted in Figure 2.3 (a). In Equation 2.12, K_r is the bond constant and it is normally around 100 kcal mol^{-1}, meaning that the amount of energy required to compress a bond is quite large, although this is dependent on the bond order. This ensures that bond lengths tend to return to r_{eq}, which is the reference bond length. Note that covalent bonds cannot be created nor broken during a MD simulation, so they need to be defined at the beginning as part of the initial topology of the molecules inputted into the simulation. In the case of simulating chemical reactions, there are force fields such as ReaxFF [132] that can account for bond breaking and creation.

The second term of the bonded interactions from Equation 2.12 describes the angle deformation and it is displayed in Figure 2.3 (b). This term is also described with a harmonic potential, again the quadratic nature of this energy function creates a restorative force that tends to bring angles back to their equilibrium value θ_0. However, the value for the angle constant K_θ is smaller than K_r [125], meaning that it takes less energy to deform an angle by a degree than it does to extend or compress a bond by an Angstrom. Note that some force fields, like CHARMM, include the Urey-Bradley term, which is an additional harmonic term that quantifies the potential energy of angle deformation associated to the bending between a central atom and its two surrounding atoms. It has the following form:

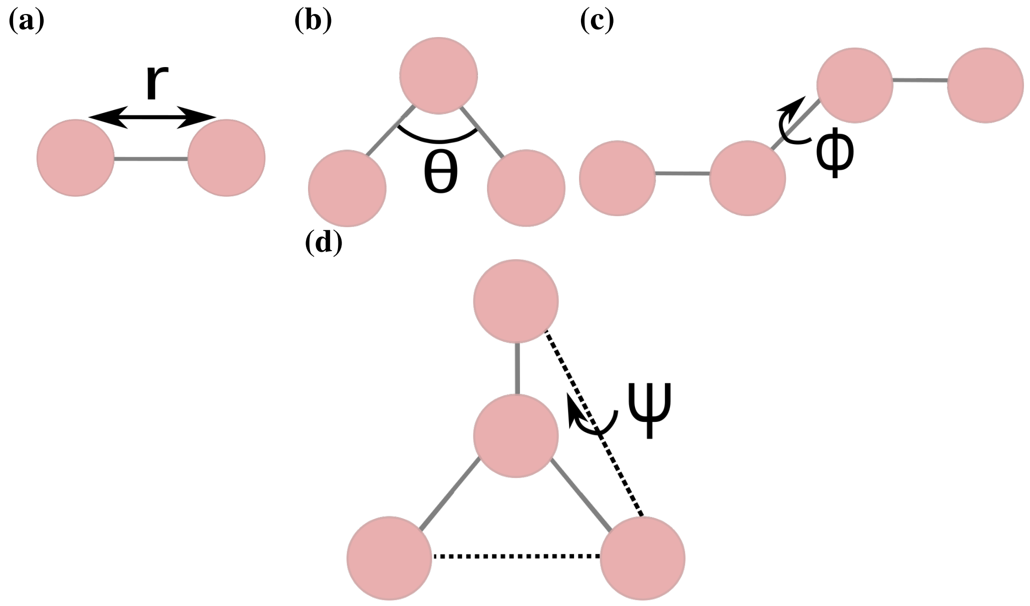


Figure 2.3: Schematic representation of bonded potential terms. Schematic representation of (a) bond term, (b) angle term, (c) dihedral term and (d) improper dihedral term. Particles are represented in pink and bonds in grey.

$U_{UB} = \sum K_u(u - u_{eq})^2$, where u is the distance between the outer atoms of a bonded triplet. Therefore the Urey-Bradley term describes the connection between angle bending and bond stretching. This term provides a more accurate description of vibrational spectra[111].

The next term in the bonded potential is the dihedral energy term, schematically represented in Figure 2.3 (c). From this Figure, it is clear that this term characterizes the rotation around a torsion angle. This term includes atoms that are separated by three bonds. The dihedral term is expressed using a periodic potential energy function, which captures the periodicity of torsional rotations, allowing for the simulation of molecular conformations with varying degrees of rotation around specific bonds. The energies involved in the dihedral torsions are much smaller than the bond and angle deformation energies. In this term, k_ϕ is the force constant, n is the number of minima in the energy function, since there can be several minima in a dihedral potential due to the possible torsions, and ϕ_{eq} is the equilibrium torsion angle, which is the position of the minima.

Finally, the last term of the bonded interactions is the improper dihedrals potential, depicted in Figure 2.3 (d). ‘Regular’ dihedrals describe the torsions of atoms around single bonds, while improper dihedrals preserve the planar alignment of (four) atoms, such that its equation has the corrective term with the constant K_d, which penalized if atoms get out of plane. This is of vital importance to fix incorrect molecular conformations.

2.1.6.2 Non-bonded potential

The remaining contributions to the total potential energy in Equation 2.12 are the non-bonded potentials, which describe the pair-wise interactions of particles that do not belong to the same molecule. The fifth term in Equation 2.12 is the Lennard-Jones potential. Figure 2.4 (a) shows a model Lennard-Jones potential with $\varepsilon = \sigma = 1$ and reduced units for clarity. The first term of the Lennard-Jones equation (R^{-12}) describes the repulsion of particles at short distances (e.g. when $R_{ij} < \sigma_{ij}$). The second term (R^{-6}) describes the attractive long range dispersion interaction between pairs of atoms. Therefore, the Lennard-Jones potential describes the physical phenomena by which if atoms are closer to each other than their characteristic size (as defined by σ_{ij} they repel each other (due to the Pauli exclusion principle) and it also characterizes the longer range interactions that arise from the fluctuating electron distributions of atoms, which leads to changes in their charge distribution inducing instantaneous dipoles, known as dispersion forces. The Lennard-Jones potential has an absolute minimum at $r_{ij} = 2\sigma_{ij}^{(1/6)}$, which determines the stable equilibrium distance between two interacting atoms. At this minimum, $U_{LJ} = -\varepsilon_{ij}$, which is the depth of the potential well, as depicted in Figure 2.4 (a). Therefore, ε_{ij} determines the strength of the interactions between two atoms. On the other hand, σ_{ij} is the r_{ij} value at which the potential crosses the x-axis, meaning that the potential changes from positive to negative, from describing an attractive interaction to a repulsive interaction. The values of ε_{ij} and σ_{ij} are different for each pair of chemical elements involved in the interaction since they are based on the chemistry of each atom. If the interaction is between two different atoms types, these parameters are calculated using the Lorentz-Berthelot mixing rule, which consists of combining the individual σ and ε parameters of the individual atom types to obtain the mixed interactions parameters. For two interacting atoms A and B, the calculation

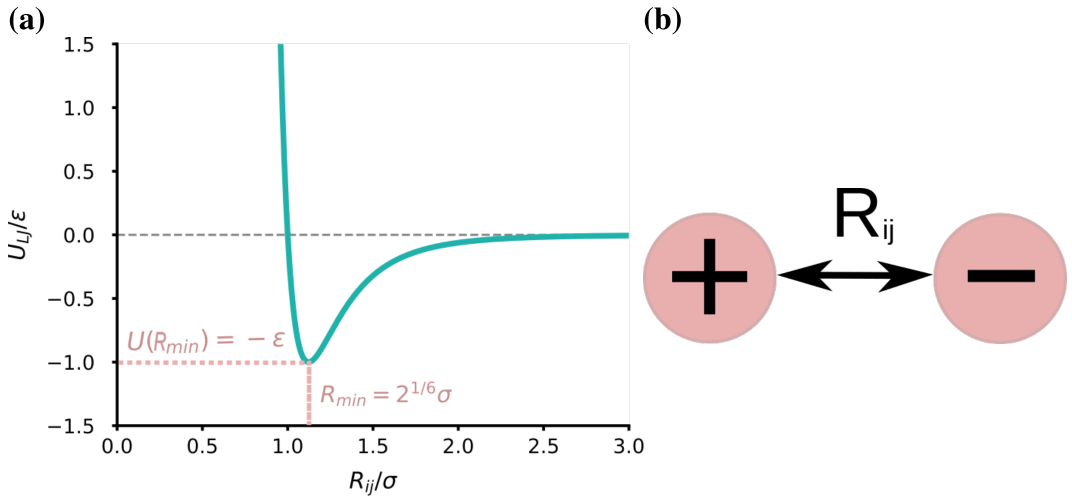


Figure 2.4: Non-bonded potential terms. (a) Lennard-Jones potential with $\varepsilon = \sigma = 1$. (b) Schematic representation of Coulomb interactions. The distance between the two particles is R_{ij} on both (a) and (b). In (b) the positive sign means the particle is positively charged and the negative sign negatively charged. Since their charges are opposite the force between them is attractive, represented by the black arrow.

of the mixed parameters using the Lorentz-Berthelot rule would be:

$$\varepsilon_{\text{AB}} = \sqrt{\varepsilon_A \varepsilon_B} \tag{2.13}$$

$$\sigma_{\text{AB}} = \frac{\sigma_A + \sigma_B}{2} \tag{2.14}$$

From Equations 2.13 and 2.14 it is clear that the Lorentz-Berthelot rule consists in taking the geometric mean of the interaction strengths (ε) and the arithmetic mean of the Van der Waals radii (σ). Also, it is important to note that when $r \to \infty$, $U_{LJ} \to 0$, which is shown in Figure 2.4 (a) by the dotted grey line. Nevertheless, to reduce computational complexity in MD simulations a radial cutoff (r_{cutoff}) is applied to limit the number of Lennard-Jones interactions between atoms. Normally, $r_{cutoff} = 12$ Å , so that from this value $U_{LJ} = 0$. Moreover, to maintain smooth energy transitions near the cutoff, a switching or shifting function is often employed, this ensures that the potential energy gradually goes to zero as r_{cutoff} is reached, preventing abrupt discontinuities in simulations [127]. It is important to note that when r_{cutoff} is applied to the Lennard-Jones potential, since long-range interactions are neglected, this can lead to an underestimation of pressure in the system. To correct for this, a pressure correction term needs to be applied.

The last term of Equation 2.12, is the final non-bonded potential contribution to the U_{total} and it is the Coulomb potential which describes the electrostatic intermolecular pair-wise interaction of atoms. These interactions arise from the electrostatic forces between charged particles. A schematic representation of two atoms with opposite signs interacting via the Coulomb potential is depicted in Figure 2.4 (b). The Coulomb potential describes the long-range nature of these electrostatic interactions. In Equation 2.12, q_i and q_j represent the fixed-point partial atomic charges of atoms i and j. These partial charges reflect the asymmetric distribution of electron density across covalent bonds, so the partial charge of an atom depends on its local chemical environment and the conformation of the molecule [125]. Furthermore, the Coulomb potential captures attractive forces between opposite charges (negative potential energy) and repulsive forces among charges of the same sign (positive potential energy). In MD simulations, a radial cutoff of normally 12 Å is also applied to the Coulomb potential, to avoid high computational costs. The theory and methods behind calculating long-range interactions for the non-bonded interactions will be described in more detail in Section 2.1.7.

The last point to note is that keeping track of all the atoms that are interacting with each other can be computationally heavy. For this, MD simulations use a Verlet neighbour list, which contains all of the atoms that are within a distance from each other smaller than the cutoff distance such that $r_{ij} < r_{cutoff} + \delta$, where δ is a small distance. This list is updated over the course of the simulation [133], such that the non-bonded interactions are only calculated on those pairs of atoms that are on each other's neighbour list. Note that this list reduces the computational cost of the simulation because it does not need to be updated at every time step, since in most simulated system, atoms do not move significantly between consecutive time steps. Therefore, the list remains constant over short intervals. The frequency at which this list is updated will depend on the simulation parameters, size of the system and desired balance between accuracy and efficiency.

2.1.6.3 TIP3P and Martini Polarizable Water Models

In simulations it is important to capture the behavior of water, including polarization. In the case of the CHARMM force field, the TIP3P water model. This is a three-site water model with partial charges on each water molecule atom [134], which allows this model to simulate dipoles. This model represents water molecules with three-sites, two hydrogens and one oxygen. The oxygen has a particle charge of $-0.834q_e$ and the hydrogens $0.417q_e$ where q_e is the fundamental charge. This charge difference within the water molecule accounts for the oxygen greater electronegativity of oxygen [134]. On the other hand, the polarizable water model in MARTINI is the MARTINI polarizable water model[135]. This model uses three-beads to represent four water molecules, and it has been shown to account for the polarizability of water, reproducing its dielectric nature. The three beads consist of a central bead called W which is neutral (just this bead would be the standard water model in MARTINI), and then two particles named WP and WM, bounded to W that have a positive and negative charge of $+q$ and $-q$ respectively. The central atom W interacts with the rest of particles in the system via Lennard-Jones interactions, and WM and WP interact only with the Coulomb function, and of course WM and WP belonging to the same particle do not interact with each other. This, combined with the fact that the bonds W-WP and W-WM are constraint to a certain length, make WP and WM rotate around the W particle, changing its dipole momentum [135]. Note that for simulations using the polarizable MARTINI model, the background dielectric constant is set to 2.5, instead of the non-polarizable MARTINI dielectric constant which is normally set to 15 [136]. Both, the TIP3P and MARTINI model are used in the simulations showed in this thesis.

2.1.7 Calculating Long-Range interactions

As mentioned in the previous section, not all possible pair-wise non-bonded interactions are calculated, only those where the atoms are within the cutoff distance

r_{cutoff}. The interactions outside of this range are called 'long-range interactions' and they are essential to understand the behavior, structure and properties of the system. An important point when calculating long-range interactions is to smooth the potential when approaching the cutoff value to avoid artifacts that lead to unphysical behavior during the simulation. Since the Lennard-Jones and electrostatic potential decay at different rates (they are different functions), the methods applied to smooth their potential are different.

The Lennard-Jones potential can be smoothed with a shift or switch function [125], which modifies the force as derivative of the potential, and preferably the second derivative of the potential go to zero when approaching the cutoff value. This shifting (instead of an abrupt truncation) ensures a smooth energy landscape. However, the same method cannot be applied to Coulomb interactions. One of the reasons why the shifting method works for the Lennard-Jones but not for the Coulomb potential is that the Lennard-Jones values are closer to 0 at the cutoff distance than the values of the Coulomb potential. Therefore, shifting the potential to 0 is easier for the Lennard-Jones because its values are naturally closer to the truncated value, but for the electrostatic interactions they are normally much larger than 0. This can also can be deduced since the Coulomb potential decays $\propto r_{ij}^{-1}$ while the Lennard-Jones potential decays much faster. The are different methods that address this issue of accurately calculating long-range electrostatic interactions, but the most used in MD simulations is the Particle Mesh Ewald (PME) method [137, 125]. PME is based on the Ewald summation method, which separated the long-range interactions into two components: real space (particles in the vecinity) and reciprocal space (covering longer-range interactions). In the real space, interactions are calculated directly between particles within a cutoff distance, and beyond the cutoff the interactions are smoothly truncated to zero. But, to compensate this truncation, a complementary term is introduced in the reciprocal space. This term is a Fast Fourier Transform of the charge distribution. Therefore, the real and reciprocal space contributions can be summed, eliminating the truncation artifacts [137]. The PME method is an interpolation of the reciprocal-space Ewald sum [125]. It

calculates explicitly the interactions that are local and then the long-range interactions are approximated by a discrete convolution on an interpolating grid using a 3D fast Fourier transform (FFT) [137] to perform the convolution. This means that the charge distribution is mapped onto a 3D grid, and then the FFT is used to transform charge densities between the real and reciprocal space, reducing the computational burden, since when the grid is converted into the reciprocal space, the values can be evaluated in a single sum, instead of multiple sums as occur in the Ewald summation method. Therefore this method is not pair-wise and it does not cut off the potential but converts the system into a periodic system.
Note that electrostatics are treated differently in MARTINI, in this case, long range interactions are calculated using the reaction field method. This method assumes that the solvent has a dielectric constant different from that of the solute (the molecules of interest). The choice of dielectric constants for the solute and solvent is a parameter in the MARTINI force field that can be adjusted to tune the strength of electrostatic interactions in the simulations, depending on the specific system being studied. The reaction field method simplifies the calculations by treating long-range electrostatics in a continuum approximation, which is computationally more efficient than explicitly calculating pairwise interactions between all particles in the system over long distances. In MARTINI simulations, the dielectric constant is typically set to infinity for distances greater than the radial cutoff. This choice effectively means that long-range electrostatic interactions are treated as if they are in a vacuum, and there is no damping of the electrostatic forces beyond the cutoff distance.

2.1.8 Constraint algorithms

MD simulations model particles with multiple degrees of freedom, since they possess translational, rotational and vibrational motion. However there are certain degrees of freedom that are not as important to simulate the behavior of molecules. These degrees of freedom are mainly those related to bond vibrations, such as bonds involving hydrogen atoms, which have a very high frequency and can therefore change very rapidly compared to other atom motion. This raises the issue that to capture these hydrogen vibrations, a much shorter time step would be needed, increasing the computational cost and introducing time step limitations. However, since these vibrations are normally not of interest, to avoid calculating them, which allows us to increase the time step, constraint algorithms are applied to eliminate these bond vibrations computations [125, 138]. Eliminating these faster degrees of freedom allows the time step to be increased by a factor of four, really improving the computational efficiency [138].

Constraint algorithms work by restraining specific geometries of molecules, reducing their degrees of freedom. There are many algorithms, but the most common ones are SHAKE [139] and LINCS [138]. The SHAKE algorithm iteratively adjusts the atomic positions according to constraint forces by applying Lagrange multipliers, while LINCS enforces bond length constraints by iteratively adjusting atomic positions while preserving the linearity of bond vectors. Both, LINCS and SHAKE allow the simulation time step to increase to 2 fs, moreover it has been shown that LINCS is not only more stable than SHAKE, but it also performs three to four times faster [125, 138]. The simulations in this thesis use the LINCS constraint algorithm.

2.1.9 Energy Minimisation/gradient descent

In the context of MD simulations, energy minimization aims at finding a stable and low-energy configuration for a system. As mentioned in Section 2.1.3, it is important to avoid overlapping atoms or abnormal molecular conformations in the system, as this leads to unphysical behavior. Fixing a system with these artifacts can be done via energy minimisation. Energy minimisation iteratively adjust the positions of atoms to reach a configuration where the potential energy is minimised while satisfying all the imposed constraints (such as bond length or angles). An energy minimsed structure is synonymous of a stable state, which will not have steric clashes or high energy molecular conformations, allowing MD to accurately capture the system's dynamics and interactions.

The most common minimisation technique is the 'gradient descent algorithm'.

In the GROMACS [140] package, this algorithm is applied on every atom and it implemented in the following form:

$$\vec{r}_i(t+\Delta t) = \vec{r}_i(t) - \frac{\vec{F_i(t)}}{\max(|F(t)|)} h(t) \tag{2.15}$$

where $\vec{r}_i(t)$ is the atomic position of atom i at the current step, $\vec{r}_i(t+\Delta t)$ is the atomic position of i at the next step, $\vec{F_i(t)}$ is the force acting on atom i at the current step, which is given by the gradient of the potential energy $\vec{F(t)} = \nabla U(\vec{r(t)})$. $\max(|F(t)|$ is the modulus of the largest force acting on any atom of the system at step t. $h_i(t)$ is the maximum atomic displacement from step t to step $t+\Delta t$. At each next step, if the potential energy is smaller than the one from the previous step, the new coordinates are accepted, and if not they are rejected. Therefore, in a nutshell, the gradient descent algorithm iteratively minimizes the potential energy of a system by adjusting its atomic position along the direction of the steepest energy decrease. This process is continued until a desired $\max(|F(t)|)$ or number of iterations is reached. This way, a minimum in the potential energy is found, removing high energy interactions or configurations.

2.1.10 Thermostat and Barostat

The methods described above allow the simulations of systems in the microcanonical ensemble (NVE), this means constant number of particles (N), volume (V) and energy (E). However a NVE ensemble is not a realistic biological environment, since it corresponds to an isolated system. The systems that we can study in the lab are normally under constant pressure (P) and temperature (T). Therefore, MD needs to be able to simulate systems that keep these two state variables constant, but not V or E. Thermostat and Barostats are tools in MD simulations for controlling temperature and pressure respectively, which creates realistic thermodynamic environments. These more realistic thermodynamic settings that MD simulations normally work on are the canonical ensembles (NVT) with constant V and T and isothermal-isobaric (NPT) ensembles, where T and P are constant. Both ensembles have been used in all simulations of this thesis.

It is common to start MD simulations utilising the NVT ensemble, since this allows the temperature of the system to equilibrate [122]. In the NVT ensemble, the thermostat regulates the temperature of the system while keeping V and N constant. It achieves this by rescaling atomic velocities or applying stochastic forces, such that it simulates the interactions between the system and a heat bath, making the system remain at the desired temperature throughout the simulations and allowing the system to explore its energy landscape. A very common thermostat, used in this thesis is the Nosé-Hoover thermostat [141]. The Nosé-Hoover thermostat introduces an additional damping dynamic variable into the equations of motion, which adjusts the velocities of atoms as if the system is exchanging energies with the heat bath. The modification of the equations in the Nosé-Hoover thermostat are:

$$m_i\vec{a}_i(t) = \vec{F}_i(t) - \xi m_i\vec{v}_i(t) \tag{2.16}$$

$$\frac{d\xi(t)}{dt} = \frac{1}{Q}\sum_{i=1}^{N}\left[\frac{m_i\vec{v}_i(t)^2}{2} - \frac{(3N+1)k_BT}{2}\right] \tag{2.17}$$

Where ξ is the damping parameter, Q is the heat bath's mass which controls the damping parameters and T is the target temperature. In Equation 2.17 the +1 in the second term of the parenthesis accounts for the extra degree of freedom introduced by the Nosé-Hoover thermostat [141]. Also, from Equation 2.17 is is clear that if the system is at equilibrium, so when $\partial_t\xi_i = 0$, the kinetic energy is given by the equipartition theorem. Into the velocity-Verlet algorithm the Nosé-Hoover thermostat is introduced in the following way:

$$\vec{r}(t+\delta t) = \vec{r}(t) + \vec{v}(t)\cdot\delta t + \frac{\delta t^2}{2}\left[\vec{a}(t) - \xi(t)v(t)\right] \tag{2.18}$$

$$\vec{v}(t+\frac{\delta t}{2}) = \vec{v}(t) + \frac{\delta t}{2}\left[\vec{a}(t) - \xi(t)v(t)\right] \tag{2.19}$$

$$\vec{F}(t+\delta t) = \vec{F}(\vec{r}(t+\delta t)) \tag{2.20}$$

$$\xi(t+\frac{\delta t}{2}) = \frac{1}{2Q}\sum_{i=1}^{N}\left[\frac{m_i\vec{v}_i(t)^2}{2} - \frac{(3N+1)k_BT}{2}\right] \tag{2.21}$$

In this thesis, NVT ensembles were used in most of the equilibration steps of the simulation studies.

All production steps in this thesis were performed in a NPT ensemble, so constant P and T, since it represents lab conditions better. In MD, barostats adjust the simulation box dimensions to maintain a desired pressure. It mimics the effect of an external pressure on the box, and it allows the simulation box to expand or contract in response to this pressure. Therefore, by controlling the box dimensions, the barostat regulates the system's density and pressure, also enabling simulations of condensed-phase systems. A common barostat, also used in the simulations presented in this thesis is the Parrinello-Rahman barostat [142, 143]. This barostat employs a fictitious mass-spring system to adjust the box dimensions, letting it respond to the external pressure changes and it also allows anisotropic and isotropic pressure coupling [143]. This barostat enhances the accuracy of simulations that study phase transitions, structural changes dependent on pressure changes and many other pressure-dependent phenomena.

2.1.11 Periodic Boundary conditions

Systems cannot be simulated within a 'solid' box, since this would restrain atom movement, and would lead to box-size-dependent or even unphysical behavior. The most common approach to solve this problem is the use of periodic boundary conditions (PBC). Figure 2.5 illustrates the PBC of the simulation box highlighted in black. As illustrated in this Figure, the PBC repeat the image of the simulation box infinitely in every dimension. The central box, delimited by the black lines, is the actual simulation box which is being replicated, and all periodic images, simulation boxes in grey, will move like the central box. Of course, due to the periodicity of this set up, when a particle leaves the central box, the opposite image of this particle will enter into the box (through the opposite side), and this will happen in every box of the system. Therefore, it is only necessary to keep track of the molecules in the central box [125]. Also, the most common PBC geometries used are rectangular

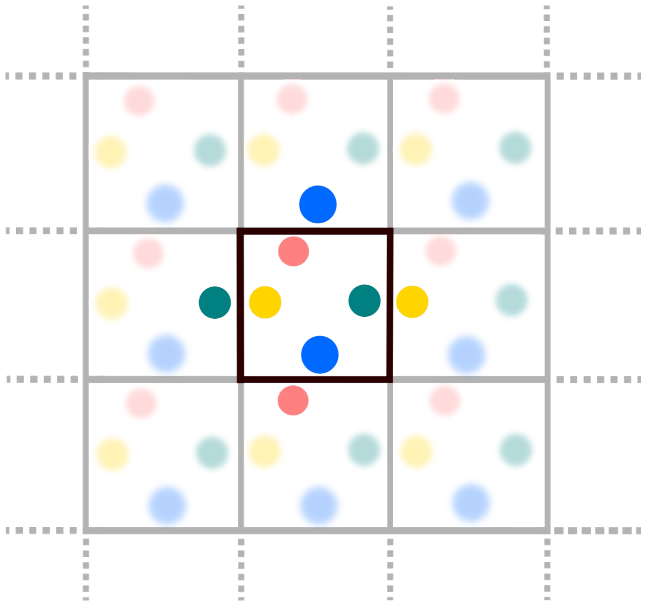


Figure 2.5: Schematic representation of PBC and minimum image convention. The main simulation box is in black and the periodic images in grey.

and cubic boxes, since their geometry is simple, which facilitates calculations.

Furthermore, to calculate short-ranged interactions in PBC conditions, the technique of the 'minimum image convention' also known as 'nearest image convention' is used, which is also illustrated in Figure 2.5. The 'minimum image convention' technique calculates interactions by considering the shortest distance between particles and also their periodic image [125, 144], so that only the 'nearest' interactions are taken into account. This prevents artifacts that could arise from particle-image interactions that could be farther apart than particle-particle interactions. For example, in Figure 2.5, the green particle is closer to the periodic image than to the real image of the yellow particle. Therefore, the distance used for the computation of their interaction will be the distance through the periodic boundary. This is also true for the interactions of the red and blue particles in Figure 2.5. However, the interactions between the red and yellow particles and green and blue particles will be computed with the distance in the real image, since it is shorter than the periodic distance. On the other hand, long-range interactions are harder to calculate with PBC conditions. For this, methods such as PME (described in Section 2.1.7) need to be employed, since these techniques account for the infinite replicas of the simulations box. PBC has limitations, such that sometimes molecules can interact with themselves, or the fact that angular momentum is not preserved by PBC. However, PBC has very little or no effect on the equilibrium properties of the system [125], so it is widely used in all MD simulations.

2.1.12 Parallelisation

The advancement of MD simulations comes hand by hand with the advances in computational power. One of the most fundamental strategies to run considerably sized systems with MD simulations is parallelization, which allows the harnessing of high-performance computing (HPC) to expedite computations. In a nutshell, HPC parallelization decomposes task into smaller ones and distributes them across multiple processors or nodes, accelerating simulations and enabling the study of larger and more complex systems. Furthermore, techniques like data parallelism, domain decomposition, and the use of the Message Passing Interface (MPI) facilitate efficient communication and synchronization among processors. This results in significant speedup, allowing simulations that would otherwise take months to complete on a single processor to be finished in a fraction of the time. Moreover, parallelization also enhances sampling methods, supports larger parameter explorations, and advances our understanding of complex biological and material phenomena. However, it is important to choose the adequate communication strategies and optimization algorithms to achieve optimal performance and scalability in parallelized MD simulations.

2.1.13 Calculating observable quantities

MD simulations produce at each time step the positions, velocities, forces and other quantities of all atoms of the system. From this time-dependent data, thermodynamic observable quantities can be obtained thanks to the link between ensemble averaging and time averaging [145]. An ensemble average consists in averaging over the configurations samples (microstates) from the trajectory, since each configuration is a point in phase space, this average captures the diversity in conformations and interactions. Ensemble averaging is useful because according to the ergodic hypothesis, systems explore all accessible states in phase space with equal probability throughout a sufficient amount of time. The average over time of a quantity is the same as that quantity averaged over the statistical ensemble[145]. This concept, allows us to measure macroscopic observables from MD simulations, because while MD simulations do not cover all possible microstates, the thermodynamic variables can be obtained from the time-average of the corresponding microscopic quantity. Also, the ergodic hypothesis assumes that given a sufficiently long time, the results from ensemble and time averages are consistent. In summary, MD simulations allow the calculation of time-averages of microscopic quantities, from which the macroscopic variables can be obtained.

2.2 Applications of Graph Theory to Molecular Simulations

MD simulations generate large amounts of data that need to be analysed in order to understand the biophysics of the system. There are many different techniques and software that extract the time evolution of specific structural properties, like hydrogen-bond distances, radial distribution functions of atoms or root mean square displacement of global structures (RMSD). However, there are not that many computational analysis methods that provide a direct knowledge of the changes in the global 3-d structures over time [146]. For this purpose, graph theory can be of use, since this is a mathematical framework that analyzes relationships between entities, often represented as nodes, and their interactions, represented as edges, which can be updated across time steps. Furthermore, graph theory is computationally very efficient, so it can be used for large system with small computational cost [146].

As its core, a graph $G(V,E)$ is made up of vertices V (also known as nodes), that are connected by edges E. These nodes can represent different entities, either single atoms or the center of mass of a molecule. The edges are the relationships, interactions or connections that exist between these nodes, for example covalent

(a)

$$A = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

(b)

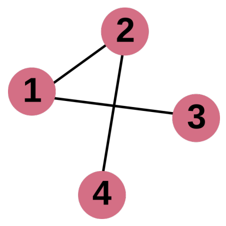


Figure 2.6: Conversion from adjacency matrix A to Graph $G(V,E)$. (a) Adjacency matrix A and (b) its corresponding graph representation.

bonds, Hydrogen bonds or a simple distance criteria. Therefore, it is easy to use graph theory to track global structural changes, since the high dimensionality of the simulation trajectory can be reduced by examining the network connectivity between the nodes and edges changes throughout the simulation. The information to create the graph is obtained from the adjacency matrix A. The adjacency matrix is a symmetric matrix that summarizes the network connectivity information of a graph. Figure 2.6 shows an example of how the connectivity of a graph can be obtained from the adjacency matrix.

In Figure 2.6 (a) each element of the matrix is an element of the graph. If the element A_{ij} of the matrix is 1, it means that nodes i and j are connected with an edge, while if it is 0 it means they are not connected. For example, the first rwo of the matrix in Figure 2.6 (a) has 1s in the second and third element, meaning that element 1 is connected to element 2 and 3, as depicted in Figure 2.6 (b). This type of matrix, where the edges are just 0s and 1s, so that the edges have no associated weights to them, are called unweighted graphs. These are the type of graphs that will be used in this thesis. The use of graph theory to analyse molecular simulations is not new, it has been used to track hydrogen bonding [147], studying the solvation of lipid headgroups [148] or to study molecular isomorphism [146] among many other physical and chemical properties. However, there is currently no openly available software that uses graph theory to study the self-assembly of soft NP. All graph-based analysis done in Python in this thesis has been performed with Networkx [149].

2.2.1 Graph theoretical cluster algorithm

Studying the structure of the self-assembly of micelles is not trivial. Polymers do not tend to form perfect NP structures, where all polymers are clearly packed in the same way during the whole simulation. Actually, it is quite common for polymers to not only form one micelle, but to form various clusters within the simulation box and for these clusters to fluctuate over time. This makes the analysis of the simulation more difficult, since always selecting the largest or most stable micelle in each time step separately is not trivial, and there is currently no openly available software to do this.

In this thesis, a novel graph theory-based clustering algorithm is developed, and has been introduced into the software package PySoftK so that it is openly available to analyse any simulations where there are aggregates (it can be used for aggregates formed by lipids, polymers, proteins, etc.) More details on the implementation of this code into PySoftK and case studies will be discussed in Chapter 4.

This method uses an unweighted undirected graph $G(V,E)$, where V are individual polymers and E are the interactions between them. These interactions are defined as a distance criteria, in such a way that if the distance between two molecules i and j, is less than the cut off distance (that is defined by the user), they are connected and form part of the same graph. The combination of connected subgraphs of G form the specific cluster at a given time. The nodes are also defined by the user. The user can define as many nodes to represent a molecule as they wish. For example, in the case of an amphiphilic molecule, each molecule can be defined with two different atoms, one in the hydrophobic part and one in the hydrophilic part, and and edge is made between this molecule i and another molecule j is the distance between any of these two atoms of molecules i and any of the two atoms of molecule j is less than the cutoff. Examples of the applications of this code can be seen in Figure 2.7, here (a) - (c) are snapshots of polymer simulations, where

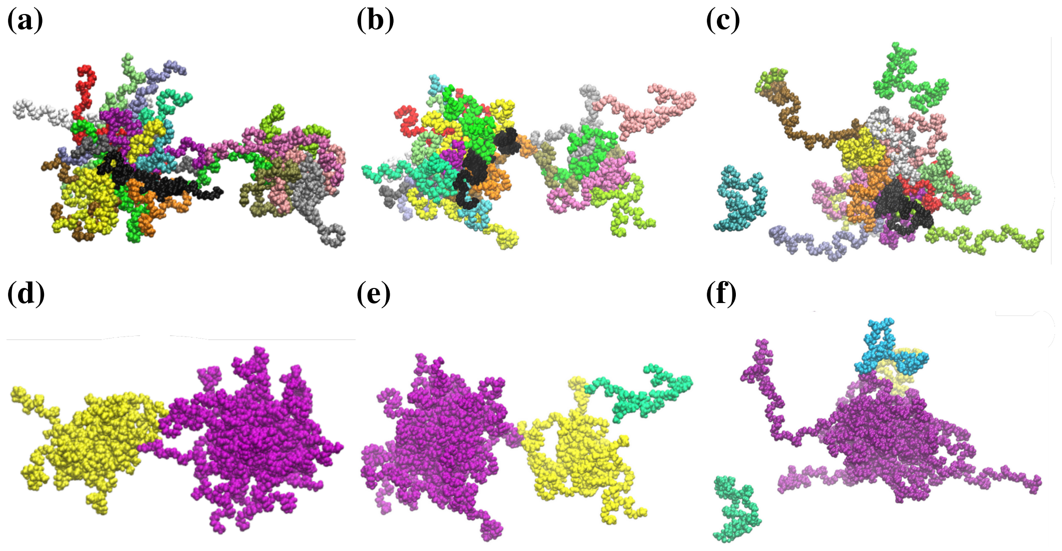


Figure 2.7: Novel graph-theory based clustering algorithm applied on different polymer micelles. (a) - (c) Plots of non-clustered micelles, where each color represents a different polymer. (d) - (f) Result of applying the clustering algorithm to each systems respectively. Polymers belonging to the same micelle have the same color. From here it is clear that the aggregates within the same system are distinctly separated.

they do not form a clear defined single micelle. The results of the application of this algorithm to them are shown in (d) - (f), from here it is clear that the algorithm successfully identifies the clusters and groups them into the different aggregates and micelles present in the simulation, which facilitates the analysis of each micelle individually.

2.3 Applications of machine learning to molecular simulations

Due to the vast amount of data that MD simulations produce, sometimes it may be hard to extract meaningful insights from simulations manually. Machine learning (ML) has emerged as a powerful tool for analysing, interpreting and finding patterns within the vast amount of data produced by MD simulations. Broadly, ML is a type of artificial intelligence that allow computers to improve their performance at a task over time by identifying patterns in the data. There are two types of ML, supervised and unsupervised learning. Supervised learning, consists of training a model to make predictions or classifications based on labeled data. These algorithms normally involve feeding the model with input features and the corresponding target labels, such that the model learns to generalize from the labeled data and can predict properties for new input. In the case of MD simulations, the input parameters can be obtained from simulations, such as atomic coordinates or energy values, and their labels could be physical characteristics, such that the model could predict new properties from unseen atomic configurations. On the other hand, there is unsupervised learning, which deals with extracting patterns, structures and relationships from unlabeled data, so that the output is completely unknown. This approach is particularly useful if the underlying data is not well understood. In the context of molecular simulations, unsupervised learning can be used to understand complex behaviours or to cluster molecular conformations throughout a simulation. Typical unsupervised learning techniques are clustering and dimensionality reduction. In this thesis, unsupervised learning techniques (clustering and dimensionality reduction) are applied in Chapters 3 and 5 to better understand site specific polymer conformations within a NP and their role in drug encapsulation and micelle formation.

2.3.1 Dimensionality reduction

As mentioned in the above section, dimensionality reduction is a typical unsupervised learning technique. MD simulations often output high-dimensional data spaces. This high-dimensionality makes this data hard to visualize, interpret and analyse effectively. Dimensionality reduction is a powerful technique that can convert high-dimensional data into a lower-dimensional representation while preserving as much of the relevant information as possible. Therefore, dimensionality reduction can be applied to MD simulation data to better understand and analyse the system, since the reduction of dimensionality allows the discovery of underlying patterns, relationships and structures within the data. There are many different techniques for dimensionality reduction, such as principal component analysis (PCA), autoencoders, or diffusion maps. The dimensionality reduction technique used in this thesis is UMAP [150], which is a non-linear dimensionality reduction technique.

UMAP stands for Uniform Manifold Approximation and Projection for Dimension reduction, and it is used in this thesis to obtain 2-d embedded spaces representation of high dimensional input data (polymer distances). UMAP is inspired by concepts from topological data analysis and manifold learning, and it aims to capture both, local and global relationships of the data [150]. Unlike other dimensionality reduction methods, UMAP does not only rely on preserving pairwise distances or variables of nearby data, but it preserves the topological structure of the data. This allows UMAP to maintain the intrinsic relationship between data points on a manifold, capturing features of the data that would be impossible with linear dimensionality reduction techniques such as PCA. In summary, UMAP is able to capture non-linear relationships within the data points, and also can preserve local and global information in the low-dimensional embedding. UMAP works by creating a topological description of the original data (which is a weighted graph), and then it uses a gradient descent technique to optimise the low-dimensional representation (low-dimensional graph) to be as closed to the high-dimensional one as possible.

In this thesis, UMAP is used to reveal if polymers adopt location-specific conformation within NP and the influence of these conformations in NP stability and drug-loading. Probably the most important step when applying UMAP to MD data is to select input data that, in this case, better captures the possible conformations polymers can take. In the work described in Chapter 3 all input data are intra-polymer distances, but the selection of these distances change depending on the topology. In this Chapter, the distances that better capture polymer conformation would be those that are able to describe the extension of the hydrophobic and hydrophilic blocks of the individual polymers. Note that molecular symmetry had to be taken into account when preparing the input data, since some of this topologies were chemically symmetric. More details on distance selection and symmetry considerations for the UMAP input data are discussed in the respective Chapter. In Chapter 5 the location-specific polymer conformation was also studied, but in this case the polymers had a CG representation and they were much bigger (around 200 monomers). However, since they had a diblock topology, it was only necessary to use two distances (per polymer) to capture their possible conformations: the distance between the hydrophobic and hydrophilic block. This system was particularly big, around 200 polymers, which shows that UMAP works really well to reduce the complexity of high-dimensional space with a relative low dimensional input features.

2.3.2 Clustering

Once the embedded space of the low-dimensional data is obtained with UMAP, it is necessary to group the data into clusters so that, in this case, the polymer conformations can be identified. The clustering technique used in this thesis is HDBSCAN (Hierarchical Density-Based Spatial Clustering of Applications with Noise) [151], which is an advanced clustering algorithm that can identify clusters and outliers in complex data. It builds upon the foundation of traditional density-based clustering methods, but extends their capabilities by introducing a hierarchical approach that captures the clusters of varying densities and shapes. HDBSCAN works by dividing up the input space into regions of high density, and identifies them as potential cluster cores. It also identifies regions of smaller density, which HDBSCAN classifies them as noise. Then, it forms a distance weighted graph to represent the non-noise data, and it then finds its minimum spanning tree (MST). The MST of a weighted graph is the set of vertices that generate a fully-connected acyclic graph while minimising the total weight of its vertices. The MST of a graph is depicted in Figure 2.8. Figure 2.8 (a) depicts the original graph and Figure 2.8 (b) is MST representation. From here, it is clear that the graph in Figure 2.8 (b) contains the same number of nodes, but the difference is in the edges. The MST method creates the minimum number of edges to main the full connectivity of the original graph. For example, the graph in Figure 2.8 (b) does not need an edge between node 2

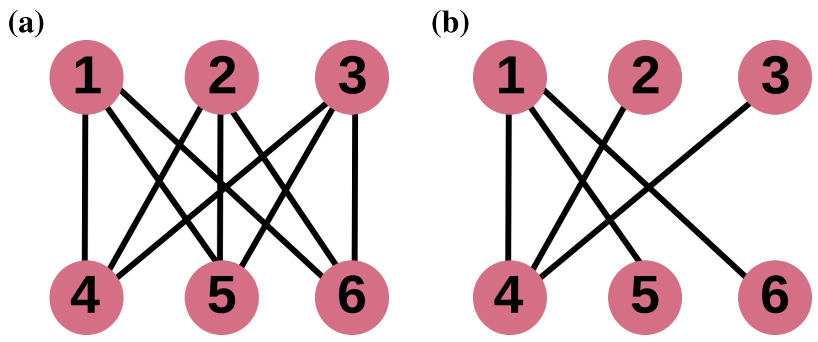


Figure 2.8: Visual representation of the MST of an unweighted graph. (a) Unweighted graph and (b) its MST representation. The MST graph contains the same number of nodes but minimises the number of edges.

and 5 because they are both already connected through node 4. Therefore, the MST creates a fully connected graph with vertices with the minimal weight. On the MST graph that HDBSCAN generates, the clustering hierarchy is applied and different sets of clusters are found, trying to make all clusters similar. HDBSCAN uses two hyperparameters to identify these clusters, the minimum number of points to form a cluster and a cluster cutoff distance (points at a distance shorter than the cutoff will be merged onto the same cluster), these hyperparameters are defined by the user. Correctly using HDBSCAN is an iterative process, which consists in trying different hyperparameters until the clusters obtained have physical meaning.

2.4 Other novel analysis tools

The work presented in this thesis also involves the creation of other novel tools for the analysis of simulations. All analysis tools developed in this thesis that can be transferable and applied to other systems or simulations, have been added to PySoftK so that the wider MD community can benefit from them. This section highlights two analysis tools developed for the analysis of simulations and that solve two long-standing problems that the MD community faces.

2.4.1 Solving the unwrapping of complex molecules

As previously described in Section 2.1.11, molecules can be broken across the PBC. During the simulation this is not an issue, since the image of the simulation box is repeated in every dimension, so molecules keep being whole in the simulation thanks to their periodic images. However, molecules breaking across the PBC can raise issues during the analysis, since sometimes analysis tools do not take PBC properly into account and this ends up giving very large or unphysical values of the system's characteristics. For example, in the case of polymer-based NP, if a NP is broken across the PBC, this can lead to very large values for the radius of gyration (does not take into account the periodic distance) or in the case of interactions between molecules, since the molecule is broken across the PBC, it is not able to capture all its interactions with the surrounding atoms properly (including the surrounding atoms of the periodic image). GROMACS [152] and also the python-based library MDAnalysis [153] have developed methods to account for this that are able to make molecules or structures whole. Also MDAnalysis, includes an optional parameter called 'PBC' in some of its functions, such as in the distance analysis functions [153], so that it takes into account the periodic boundary conditions of the system. However, when the structure of interest has a size larger than half the box size, MDAnalysis and GROMACS are not able to make this structure whole across the PBC, nor are they able to perform the analysis correctly.

In this thesis a function named `make_cluster_whole` has been developed to account for this issue. This code is able to reconstruct molecules and structures even when they are bigger than half the box size, and the coordinates of the whole structure can be used to perform the correct analysis of the simulations. This function takes as the input the positions of the atoms of the structure that needs to be made whole. Then, it bins the positions within a range of bins defined by the user, which is recommended to be at least the size of the simulation box in one of its directions. Once the positions are binned, and the number of atoms within each bin are counted, it is outputted as an array. Therefore, when the structure is broken across the PBC, this array will contain specific numbers, followed by zeros, and then numbers again, as for example $[1,2,3,4,5,6,0,0,0,0,6,5,4]$ so that the bins corresponding to 0 are the space through which the structure is broken, since a whole molecule should not have its atoms split into two regions of the binned space. Then, the atoms that are after the broken binned space are then moved towards the filled binned space, taking into account the dimensions of the box, in such a way that the previous array would now be: $[6,5,4,1,2,3,4,5,6,0,0,0,0]$ so that the atoms have been moved across the broken space, making the structure whole and independently of the box length. This is why `make_cluster_whole` can make structures whole when they are bigger than half the box length, when GROMACS or MDAnalysis fail. Furthermore, this function has been developed so that it can also wrap the solvent around the whole structure. Examples of how to call this function and more practical examples will be discussed in Chapter 4.

2.4.2 Ring-ring stacking

The ring-ring stacking calculation tool developed in this thesis is able to determine if molecules within a simulation have inter molecular stacking of their ring structures. It is not only able to detect pair-wise stacking, but it obtains any series of molecules with rings stacked. Measuring ring stacking is of interest, particularly for polymeric materials, since polymeric materials with ring stacking exhibit interesting optical and/or electrical properties as a consequence of collective phenomena such as building block agglomeration or crystallization processes [154] Figure 2.9 shows two polymers with ring stacking identified by this algorithm. Currently, PySoftK is the only code that offers an open-source implementation enabling this analysis.

Briefly, this code works by first using RDKit [155] to find the atoms belonging to the ring structure of each polymer in the system. Then, the distance between all polymers of the system are calculated, and for those polymers which distance is below a certain cutoff, that is to say, for those polymers that are close enough that could be interacting, the distance between their rings is calculated. Afterwards, if the distance between the rings is below the stacking distance cutoff, the angle

(a)

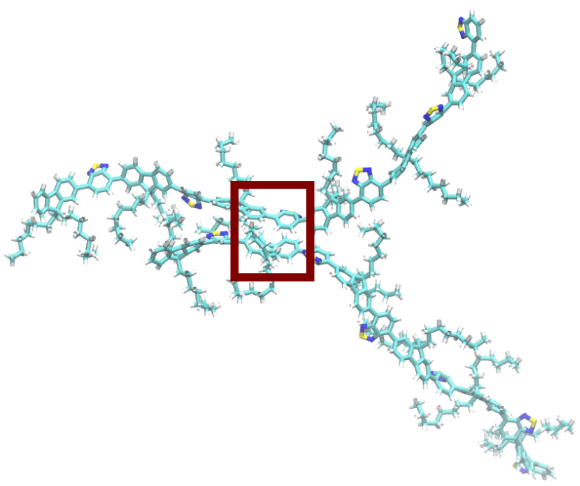


Figure 2.9: Conjugated polymer ring stacking identified by ring-stacking calculation tool (a) Snapshot of two conjugated polymers with ring structures. The rings in the red square shows the stacking of the rings of these two polymers. This stacking is the output of the ring-stacking calculation tool developed in this thesis.

between the planes formed by their rings is calculated. If this angle is within the user-defined angle range, then they are considered to be stacked. The angle calculation between the rings is not trivial, it is necessary to perform singular value decomposition (SVD) to find the right unitary singular vector that is normal to the plane of the ring. Then the angle is calculated with the following formula:

$$\theta = \left| \arccos \left(\frac{\vec{v} \cdot \vec{k}}{|\vec{v}|} |\vec{k}| \right) \right| \tag{2.22}$$

Where $\vec{v}$ and $\vec{k}$ are the unitary singular vectors orthogonal to the ring belonging to polymer 1 and 2 respectively, that are found via SVD, and θ is the angle between the two planes formed by the ring atoms. As an optional output of this algorithm the pdb files of pairs of polymers with their rings stacked are printed. Then, to further explore the ring stacking structure, a tool to obtain the polymer network connected by ring stacking was developed. This method works similar to the graph-based clustering tool, if two polymers have their rings stacked they are added as vertices of the same subgraph. This outputs the network of polymers connected via ring stacking within the simulation box. Example of the usage and output of this algorithm are displayed in Chapter 4.

Chapter 3

Effect of polymer topology on the self-assembly of micelles

This Chapter investigates the effect of polymer topology on the self-assembly of micelles using MD simulations. For this purpose, the self-assembly of poly(ethylene oxide)-poly(methyl acrylate) (PMA) block copolymers were studied with three different topologies: diblock (PEO-PMA), triblock (PMA-PEO-PMA and PEO-PMA-PEO) and ring (PEO-PMA-). All polymers had the same number of monomer species and all simulations had the same number of polymers for an appropriate comparison. The simulations reported in this Chapter show that polymer topology influences the size of the micelle and number of polymers per micelle. The ring topology leads to the smallest micelle in size and polymer number, followed by the EO-terminated micelle and afterwards the MA-terminated one. Diblock polymers result in the largest micelle in terms of size and polymer count. Interestingly, all micelles have a defined hydrophobic core primarily composed by MA monomers with a minimal amount of water, except for the MA-terminated triblock. This micelle does not have a distinct hydrophobic core, which leads to a high amount of water molecules being encapsulated within the micelle. Furthermore, in this Chapter, unsupervised machine learning (UMAP) was applied to show that polymers adopt specific spatial conformations. Moreover, these specific conformations are location-specific within the micelle, if the micelle has a clear defined hydrophobic core-hydrophilic shell structure. If the micelle does not have a clear structure, as for the MA-terminated polymers, then these polymer conformations are not confined to specific environments of the micelle. Overall, this work quantifies the changes in physical and structural properties of micelles depending on their topology, showing that polymer topology is a key parameter for the rational design of polymer nanostructures.

This Chapter consists of an article that was published on *Nanoscale* in 2023 with DOI:10.1039/D3NR01204B. The supporting information of the article is provided in this thesis in Appendix A. I, the first author of the paper, performed all the molecular dynamics simulations and data analysis, produced all figures and visualizations and wrote the manuscript.

Nanoscale

PAPER

View Article Online
View Journal

Cite this: DOI: 10.1039/d3nr01204b

Topology-controlled self-assembly of amphiphilic block copolymers†

Raquel López-Ríos de Castro,[a,b] Robert M. Ziolek[a] and Christian D. Lorenz *[a]

Received 15th March 2023,
Accepted 16th August 2023

DOI: 10.1039/d3nr01204b

rsc.li/nanoscale

Contemporary synthetic chemistry approaches can be used to yield a range of distinct polymer topologies with precise control. The topology of a polymer strongly influences its self-assembly into complex nanostructures however a clear mechanistic understanding of the relationship between polymer topology and self-assembly has not yet been developed. In this work, we use atomistic molecular dynamics simulations to provide a nanoscale picture of the self-assembly of three poly(ethylene oxide)-poly(methyl acrylate) block copolymers with different topologies into micelles. We find that the topology affects the ability of the micelle to form a compact hydrophobic core, which directly affects its stability. Also, we apply unsupervised machine learning techniques to show that the topology of a polymer affects its ability to take a conformation in response to the local environment within the micelles. This work provides foundations for the rational design of polymer nanostructures based on their underlying topology.

Introduction

The ability of amphiphilic polymers to self-assemble into specific morphologies in solution has driven interest in their deployment for a diverse range of applications.[1–4] The topology of block copolymers exerts great influence over their properties and therefore their potential applications. Ring polymers are one synthetically accessible topology that have drawn considerable attention as a result of the unique properties that they exhibit in comparison to their linear counterparts.[5–13] Functional polymer nanostructures have been typically fabricated using linear polymers but significant synthetic advances in the past two decades have made ring copolymer synthesis possible. Ring polymers demonstrate distinct self-assembly behavior,[9,12,13] which leads to their resultant micelles possessing markedly different properties,[9] including the size and shape,[14] morphology,[15,16] temperature, salt tolerance,[17,18] and degradation[14] with respect to micelles formed from analogous linear polymers.

[a]Biological Physics and Soft Matter Group, Department of Physics, King's College London, London, WC2R 2LS, UK. E-mail: chris.lorenz@kcl.ac.uk; Tel: +44 207 848 2639

[b]Department of Chemistry, King's College London, London, SE1 1DB, UK

† Electronic supplementary information (ESI) available: (i) A detailed description of the analysis carried out for the various simulations, (ii) plots of the R_G and eccentricity of the micelles as a function of time (iii) the contacts between the EO monomers and the hydration of the EO monomers on the polymers within each micelle (iv) the intrinsic density of the various components within the simulations for each micelle and (v) the outputs of the dimensionality reduction and clustering of the molecules within each micelle. See DOI: https://doi.org/10.1039/d3nr01204b

In drug delivery applications, the ability to control the size and stability of micellar aggregates is particularly important. The size of such micelles is one of the most critical features in determining biodistribution and the stability can be tuned to prevent premature release or to enable a controlled release of therapeutics. Ring polymers have shown great promise as potential drug and gene delivery vehicles because they often show improved drug loading and releasing capacity,[19,20] greater efficacy,[21–24] longer *in vivo* circulation times,[25,26] and high cancer cell uptake[25–28] as the same polymers with a linear topology.

While interest in the application of self-assembling ring polymers in drug-delivery applications is building, there is a relative lack of detailed understanding of the molecular-scale mechanisms that drive the emergence of their desirable properties. Molecular-scale simulations present the unique opportunity to build this level of understanding. Simulations have recently been used to develop understanding of the unique properties of ring polymers within polymeric melts,[29,30] extensional flows[31,32] and thin films.[33,34] However, relatively few simulation studies have investigated the underlying mechanisms that lead to the properties of ring polymers in aqueous environments observed experimentally. Studies that have been performed have primarily utilized coarse-grain polymer models to gain insight into how polymer topology affects the morphology of the micelles that form.[23,35–39]

In this manuscript, we employ all-atom molecular dynamics simulations to gain a detailed understanding of the atomistic interactions and molecular mechanisms that drive the self-assembly of a ring polymer consisting of poly(methyl acrylate) and poly(ethylene oxide) blocks (-($MA_{12}EO_{31}$-)) in comparison to

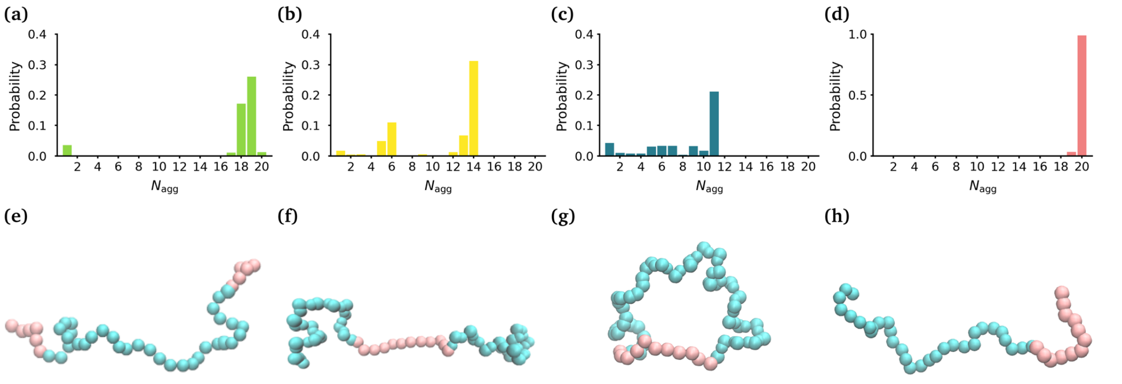


Fig. 1 Size and shape of micelles. Probability distribution of N_{agg} for the (a) MA-terminated polymers, (b) EO-terminated polymers, (c) ring polymers and (d) diblock polymers. Snapshots of the (e) MA-terminated (f) EO-terminated, (g) ring polymers and (h) diblock polymers. MA is shown in pink and EO in blue.

its analogous linear diblock topology ($MA_{12}EO_{30}$) and triblock topologies (MA-terminated ($MA_6EO_{31}MA_6$) & EO-terminated ($EO_{15}MA_{12}EO_{15}$)) (see Fig. 1(e)–(h)). We provide a detailed description of the internal structure of the micelles that each polymer forms, which plays a key role in drug solubilization, as well as the stability of micelles as drug delivery vehicles.

Results

Effect of polymer topology on the size & shape of micelles

In order to determine the size, shape and compactness of the micelles formed by the different polymers, we measured the number of polymer molecules in each micelle within our simulations at stationarity. We also measured the radius of gyration (R_G) and the eccentricity of the largest micelle in each system (Fig. S2†).

Fig. 1(a)–(d) shows the probability distribution of the aggregation number for the different topologies. The MA-terminated linear polymers form one micelle which contains approximately 19 (of the 20) polymers (Fig. 1(a)). The EO-terminated linear polymers self-assemble into two micelles, one with approximately 14 polymers and the other with 6 polymers (Fig. 1(b)). The ring polymers form multiple micelles with the largest one containing approximately 11 polymers (Fig. 1(c)). Finally, the diblock polymers predominantly form one micelle with all 20 polymers (Fig. 1(d)). The values of R_G correlate directly with the aggregation numbers, such that the diblock polymer micelle has the largest R_G, followed closely by the MA-terminated linear polymer one and then, in decreasing order, the EO-terminated linear polymer and the ring polymer (Table 1). Despite the difference in size of the micelles for the four different polymers, all of the micelles are approximately spherical (eccentricities ~0.1) (Table 1).

We have also carried out simulations of each of the different polymer topologies that contain 30 polymers at the

Table 1 Effect of polymer topology on the size and shape of micelles. The average and standard deviation for the R_G, the eccentricity ε and the average aggregation number N_{agg}

Topology	R_G (Å)	ε	N_{agg}
MA-terminated linear	28.2 ± 1.4	0.10 ± 0.06	19 ± 1
EO-terminated linear	23.3 ± 0.5	0.09 ± 0.06	14 ± 1
Ring	19.5 ± 0.4	0.07 ± 0.04	11 ± 1
Diblock	29.3 ± 0.9	0.10 ± 0.07	20 ± 1

same concentration as in the 20 polymer simulations. We found the very similar aggregation numbers in these larger systems for each of the topologies, except for the diblock (see Fig. S3†). In the diblock system, we once again see that nearly all of the polymers self-assemble into a single micelle.

Effect of polymer topology on the internal structure of micelles

We calculated the radial density (Fig. 2) of the micelles, as well as the corresponding intrinsic density using the intrinsic core–shell interface (ICSI) method[40] (Fig. S6†), in order to understand how the internal structure of each micelle is affected by the topology of each polymer. For all topologies, the corona of the micelle is constituted primarily of the EO blocks. In the case of the MA-terminated linear polymer, we observe that approximately 20% of the polymers have at least one MA-terminated end in the corona of the micelle. Therefore, the micelle core formed by these polymers has significantly more EO monomers and as a result, more water, present in its core than either of the other micelles (Table 2). Regarding the other topologies, the diblock, EO-terminated linear polymer and the ring polymer have a small amount of EO monomers in the core (Fig. 2(b)–(d) & Table 2). However the ring polymer has a slight increase in the density of EO monomers (also seen in the intrinsic densities as shown in Fig. S8†) in the core as there are no free ends of the polymer, instead both ends of the

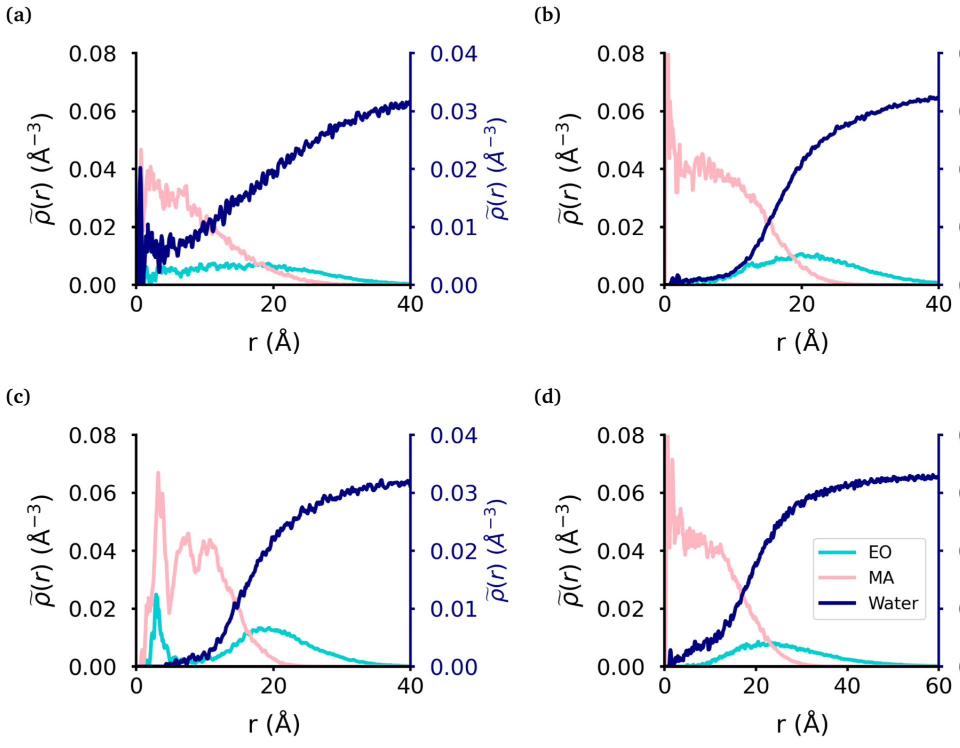


Fig. 2 Radial density of micelle components. The radial density of micelles formed from the (a) MA-terminated polymer, (b) EO-terminated polymer, (c) ring polymer and (d) diblock polymer. MA monomers are displayed in pink, EO in blue and water in dark blue.

Table 2 Hydration of the micelle core

Topology	MA_{H_2O}	EO_{H_2O}	EO_{core}
MA-terminated linear	3.0 ± 1.0	10.6 ± 3.8	51.0 ± 15.8
EO-terminated linear	0.4 ± 0.2	4.6 ± 3.1	7.9 ± 3.2
Ring	0.5 ± 0.1	2.3 ± 0.9	11.7 ± 3.0
Diblock	0.5 ± 0.1	8.8 ± 6.5	11.7 ± 6.4

The core was defined by the intrinsic surface created by the MA monomers. The two first columns are the average and standard deviation for the number of water molecules per monomer in the core with respect to the monomer units inside the core. The last column is the average and standard deviation of the number of EO monomers in the core over the trajectory.

EO block are attached to MA blocks. As there is a peak in the MA density which corresponds to the peak in EO density in the core of the micelle, it is clear that the peak in the EO density is a result of its connectivity to the MA monomers. It should also be noted that while the peaks in each curve look significant, as the volume measured that close to the core of the micelle is quite small and so the actual amount of EO is quite small.

The core of each micelle consists primarily of MA blocks. Fig. 3(a)–(d) show the normalized intermolecular contacts of the (chemically equivalent, except for the case of the diblock, where there are no chemically equivalent atoms) MA monomers in the MA-terminated linear, EO-terminated linear, ring and diblock polymer micelles. In all topologies except the diblock polymer, the number of contacts increases the further a MA monomer is from the EO blocks in each polymer with MA6, the monomer furthest away from the EO blocks, undertaking the highest number of contacts. In the diblock polymer micelles, the MA monomers that are closest to the EO blocks, are also the monomers with the lowest number of contacts. But in this case, the MA monomers found in the middle of the PMA block are the ones that have the highest contacts.

While all of the MA monomers contribute to the micelle's core, the MA monomers furthest away from the EO blocks are the monomers that play the most significant role in the formation and stabilization of the micelle core. The MA-terminated linear polymers have the lowest number of contacts between their MA monomers as a result of the MA monomers being divided into two blocks which are separated by the block of EO monomers and the number of MA monomers outside the core. Fig. 3(e)–(h) show the normalized number of water molecules within the first hydration shell of the carbonyl oxygens in the different chemically equivalent MA monomers within each polymer. The MA-terminated linear polymers have

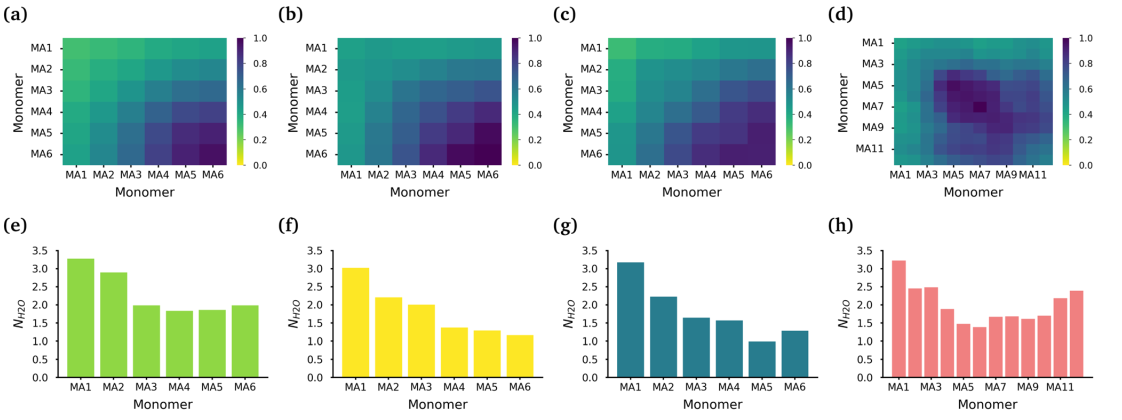


Fig. 3 Interactions within the MA core of the polymer micelles. The normalized intermolecular MA contacts within the core of the (a) MA-terminated polymer, (b) EO-terminated polymer, (c) ring polymer and (d) diblock polymer micelles. Average hydration of the carbonyl oxygen atoms in the PMA backbone of the (e) MA-terminated polymers, (f) EO-terminated polymers, (g) ring polymers and (h) diblock polymers.

the largest coordination number values, which is consistent with the measured water densities that demonstrate that more water is found within the core of this micelle. In all micelles, the most hydrated monomer is MA1 which is directly bonded to an EO monomer, and generally the hydration decreases as the monomer is further from the EO monomers.

Effect of polymer topology on polymer conformations within micelles

While the MA monomers are key in the formation and stability of the micelles, the conformations that each topology of the polymers take within the micelle is significantly different. To investigate the specific conformations that different polymers adopt within a micelle, we applied a two step machine learning protocol:[41] dimensionality reduction using the Uniform Manifold Approximation and Projection (UMAP) algorithm,[42] followed by clustering in the resulting embedded space using Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN)[43] (see the ESI section:† 'Dimensionality reduction and clustering' for the full methodology and results of this protocol). In each embedding, three clusters were identified representing the different groupings of similar conformations taken by each polymer (see Fig. S7†). In each case, there is less than 8% of the data that is not clustered by HDBSCAN, which is shown in the bar charts in gray. Fig. 4 shows the probability distribution of each cluster in the various micelles as well as representative structures of each cluster of conformations for each polymer. The representative structures show that the conformations are clearly differentiable by the relative extension of the EO and MA blocks.

We then use the ICSI method to measure the location of the various polymer conformations within each micelle (Fig. 4). Snapshots of each micelle with its constituent polymers colored by the corresponding cluster number are also shown in Fig. 4. In the MA-terminated linear polymers, the intrinsic densities of the various clusters are less than found in the other micelles, which is indicative of more water present in the core as shown in Fig. 2(a). Also there is not a significant difference in the distributions of the three conformations within the micelle. The most extended conformation (cluster 2) is representative of the previously mentioned polymers that have at most one MA block in the core of the micelle, and is also slightly more commonly found in the core. In the EO-terminated linear and cyclic micelles that have a more stable core, the polymers take specific conformations depending on their position within the core of the micelle. For example, the most extended conformation of the EO-terminated linear polymers (cluster 2) is more likely to be found in the core of the micelle with the MA block spanning the micelle and the two EO blocks extended into solution. Closer to the core–shell interface, there is an increased density of the other two conformations which have more collapsed MA blocks resulting in the MA monomers shielding the core of the micelle from the surrounding water.

For the ring polymer micelle, the adopted conformations that are most elongated (clusters 1 and 3) are found to be enriched in the core of the micelle. In these conformations, the EO block is more extended so that it can reach the micelle corona and interact with the surrounding aqueous environment. The polymers at the interface of the core of the ring polymer micelle take on a more conventional ring shape (cluster 2), allowing the EO block to expand to maximize its contact with the surrounding water and the MA block to embed into the core to minimize its interaction with water.

For the diblock polymer micelle, the pattern is similar as for the cyclic one. The most extended conformations (cluster 1 and 2) are predominate in the core of the micelle. In these conformations, the MA is more extended, allowing it to maximize its contacts with the rest of the MA present in the core. Finally, cluster 3 is more likely to be found at the core–shell interface.

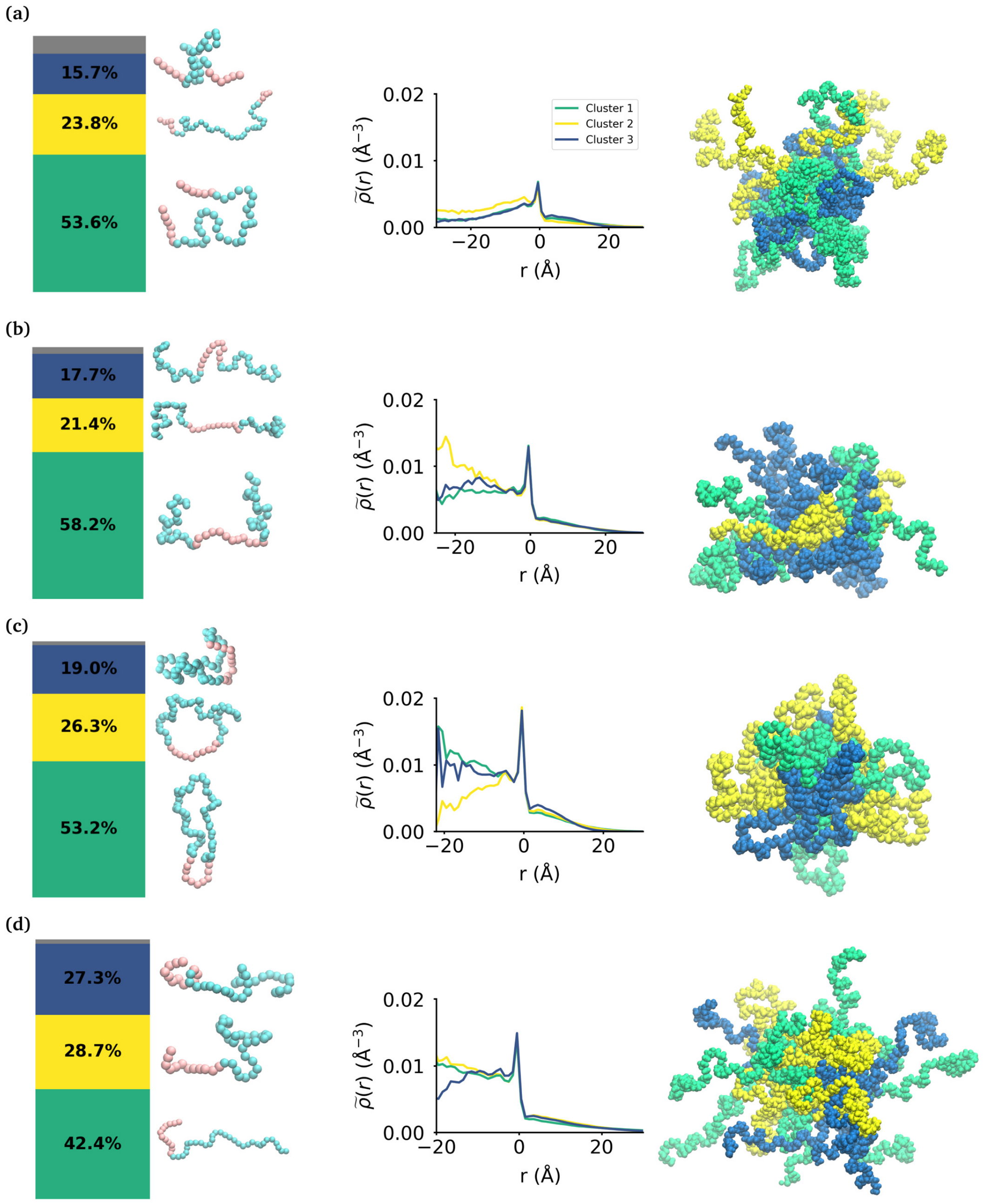


Fig. 4 Effect of topology on the internal structure of the polymer micelles. From left to right, a bar chart shows the percentage of each cluster of conformations within the micelle, then there are representative snapshots of the polymers within each cluster, and then plots of the intrinsic density of the various clusters within the micelle and finally a snapshot of the micelle with each polymer color-coded for the cluster it belongs to. These are shown for the (a) MA-terminated polymer micelle, (b) EO-terminated polymer micelle (c) ring polymer micelle and (d) diblock polymer micelle. Sizes are not to scale.

This cluster presents a collapsed MA and extended EO, which allows the EO to maximise its contacts with the water, while the MA minimizes its contacts with this solvent by collapsing within itself.

Discussion

The results of our simulations show excellent agreement with previous experimental work studying the effect of topology on the self-assembly of block copolymers. In this work, we show that the linear polymer with the hydrophobic monomers on either end of the polymer (MA-terminated linear polymer) forms larger aggregates that are less stable than those formed from the cyclic or diblock polymer. Honda *et al.* have studied MA-EO-MA linear and MA-EO ring block copolymers and found that the linear polymers form micelles that have larger hydrodynamic diameters and aggregation numbers, while also being less thermally and salt stable than the corresponding ring polymer.[18] The same authors also studied butyl acrylate (BA)-ethylene oxide linear and cyclic block copolymers and found that the size of the micelles from the two polymers were similar but the ring polymer showed greater thermal stability.[17] Our simulations show that there are more MA–MA contacts within the core of ring polymer as compared to the MA-terminated linear polymer which results in a more compact (ring: ~119 Å^2 per polymer; MA-terminated linear: ~124 Å^2 per polymer) and more stable micelle (ring has smaller fluctuations in R_G than MA-terminated; Table 1). We also find that the MA-terminated linear polymers form micelles which have a significant number of EO monomers internalized into the core of the micelle which results in there being a significant amount of water within the core (Table 2). This increased amount of the water in the core reduces the stability of the micelle (Table 1).

Our ability to identify three distinct conformations of each of the polymers allows us to provide a detailed picture of the internal structure of the micelles. In doing so, we show that for the linear polymer with the hydrophobic monomers on either end (MA-terminated linear) there are two conformations where the MA blocks are near to one another and one conformation in which the polymer is fully extended with the MA blocks separated from another. This is consistent with the general picture suggested for the MA-EO and BA-EO polymers studied by Honda *et al.*[17,18] as well as for Pluronics which contain blocks of propylene oxide (PO) and ethylene oxide.[44] In each case, the authors suggest that these polymers with the hydrophobic monomers on the terminal ends form flower-like micelles where a majority of the polymers have both terminal ends within the core of the micelle, and some of the polymers have a hydrophobic terminal end in solution. The results of our simulations for the MA-terminated linear polymers show that ~20% of the polymers take conformations which result in at least one of the MA-blocks being in the corona of the micelle. Interestingly, with the larger aggregation number for the MA-terminated linear polymers than for the micelles formed from the EO-terminated linear polymers, we find that both micelles have roughly the same number of MA monomers (~360) in the core of their micelles.

We found that in the micelles formed by the EO-terminated triblock, the diblock and the ring polymers, which have a well defined core and corona, the polymers take different conformations depending on their location within the micelle. In the case of the EO-terminated linear polymer we find that the polymers in the core of the micelle have a propensity to have an elongated MA block which maximizes the hydrophobic contact between MA monomers and more compact EO blocks which lie on the surface of the micelle. The polymers at the core/shell interface of the micelle have more compact MA blocks which allows the polymers to more effectively shield their hydrophobic blocks and the EO blocks are more extended in order to maximize their hydration. While in the ring polymer micelle, we find two more elongated conformations which are most prominent in the core of the micelle, whereas the other more ring-like conformation sits at the core–corona interface. These conformations taken by the ring polymers in the different parts of the micelles allow the polymers to maximize the hydrophobic contact of the MA blocks while also allowing the EO monomers to maximize their interaction with the surrounding water. In the case of the diblock polymer micelle, we find that the conformations where the MA blocks are the most extended are located closer to the core, while the conformation with a collapsed MA block is found close to the core–shell interface. Then, it is clear that these conformations are the result of the MA monomers maximising their hydrophobic interactions and minimising their contact with the aqueous environment. Therefore our findings show that polymers that can take location specific conformations will form stable micelles that have hydrophobic cores which are shielded by the hydrophilic monomers, and those that cannot, the MA-terminated polymer in this case, will not.

Conclusions

Our simulations provide a mechanistic picture of what leads to the difference in size and stability of micelles formed by block copolymers that differ in topology but not in the chemical composition of their constituent monomers. Additionally, we have been able to demonstrate the range of conformations that are taken by four different topologies of polymers within the micelle and how they determine the stability of the micelles. We have also shown how the conformations of the polymers change as their position within the micelle changes, which is particularly interesting when considering loading these micelles with small molecule therapeutics, as the location and the hydration of the drug within the micelle will be driven largely by the conformations of the polymers in its local environment. This understanding allows polymer topology to become another parameter that can be used to perform rational design of polymer nanoparticles for the use in a variety of applications including drug delivery.[45–47]

Methods

Each simulation reported consists of 20 polymers placed in a simulation box with initial dimensions of 147 Å × 147 Å × 147 Å containing approximately 105 000 water molecules, resulting in 3 wt% solutions of each polymer. We used the OPLS forcefield parameters as prescribed by the PolyParGen webserver[48] to describe the interactions of the polymers and the TIP3P water model.[49] All of the simulations were performed using GROMACS[50] versions 2019.2 and 2020.4. The same simulation protocol was followed for each of three simulations, which begins with energy minimization by steepest descent, followed by a 125 ps simulation in the NVT ensemble using the Nosè–Hoover thermostat to control the temperature (target temperature 300 K) with a timestep of 1 fs. Subsequently we ran 1 μs production simulations in the NPT ensemble using the Nosè–Hoover thermostat and the Parrinello–Rahman barostat to control the temperature (target temperature 300 K) and pressure of 1 atm, respectively with a 2 fs timestep while all hydrogen-containing bonds were constrained using the LINCS algorithm.[51] In all simulations, the non-bonded interactions were cut off at 12 Å while the particle-mesh Ewald (PME) algorithm was used to calculate long-range electrostatic interactions. Appropriate burn-in times were calculated, with only the stationary portion of the production simulations used for analysis. A description of all of the analyses conducted on these simulations is described in the ESI.†

Author contributions

Raquel López-Ríos de Castro: data curation, formal analysis, investigation, methodology, software, validation, visualisation, writing – original draft. Robert M. Ziolek: conceptualization, formal analysis, methodology, software, supervision, writing – review & editing. Christian D. Lorenz: conceptualization, funding acquisition, project administration, resources, supervision, writing – review & editing.

Conflicts of interest

There are no conflicts to declare.

Acknowledgements

We are grateful to the UK Materials and Molecular Modelling Hub, which is partially funded by EPSRC (EP/P020194/1 and EP/T022213/1), and the UK HPC Materials Chemistry Consortium, which is also funded by EPSRC (EP/R029431), for providing us access to computational resources. This work also benefited from access to the King's Computational Research, Engineering and Technology Environment (CREATE) at King's College London.[52] R. L.-R. D. C. acknowledges the support by the Biotechnology and Biological Sciences Research Council (BB/T008709/1) *via* the London Interdisciplinary Doctoral Programme (LIDo). R. M. Z. and C. D. L. acknowledge the Engineering and Physical Sciences Research Council (EPSRC) for funding (EP/V049771/1). For the purpose of open access, the author has applied a Creative Commons Attribution (CC BY) licence (where permitted by UKRI, 'Open Government Licence' or 'Creative Commons Attribution No-derivatives (CC BY-ND) public copyright licence' may be stated instead) to any Author Accepted Manuscript version arising.

References

1 S. A. Jenekhe and X. L. Chen, *Science*, 1998, **279**, 1903–1907.
2 D. E. Discher and A. Eisenberg, *Science*, 2002, **297**, 967–973.
3 F. Ahmed and D. E. Discher, *J. Controlled Release*, 2004, **96**, 37–53.
4 P. P. Ghoroghchian, G. Li, D. H. Levine, K. P. Davis, F. S. Bates, D. A. Hammer and M. J. Therien, *Macromolecules*, 2006, **39**, 1673–1675.
5 B. A. Laurent and S. M. Grayson, *Chem. Soc. Rev.*, 2009, **38**, 2202–2213.
6 H. R. Kricheldorf, *J. Polym. Sci., Part A: Polym. Chem.*, 2010, **48**, 251–284.
7 T. Yamamoto and Y. Tezuka, *Polym. Chem.*, 2011, **2**, 1930–1941.
8 Z. Jia and M. J. Monteiro, *J. Polym. Sci., Part A: Polym. Chem.*, 2012, **50**, 2085–2097.
9 R. J. Williams, A. P. Dove and R. K. O'Reilly, *Polym. Chem.*, 2015, **6**, 2998–3008.
10 T. Yamamoto and Y. Tezuka, *Soft Matter*, 2015, **11**, 7458–7468.
11 X.-Y. Tu, M.-Z. Liu and H. Wei, *J. Polym. Sci., Part A: Polym. Chem.*, 2016, **54**, 1447–1458.
12 F. M. Haque and S. M. Grayson, *Nat. Chem.*, 2020, **12**, 433–444.
13 C. Chen and T. Weil, *Nanoscale Horiz.*, 2022, **7**, 1121–1135.
14 B. Zhang, H. Zhang, Y. Li, J. N. Hoskins and S. M. Grayson, *ACS Macro Lett.*, 2013, **2**, 845–848.
15 E. Minatti, R. Borsali, M. Schappacher, A. Deffieux, V. Soldi, T. Narayanan and J.-L. Putaux, *Macromol. Rapid Commun.*, 2002, **23**, 978–982.
16 E. Minatti, P. Viville, R. Borsali, M. Schappacher, A. Deffieux and R. Lazzaroni, *Macromolecules*, 2003, **36**, 4125–4133.
17 S. Honda, T. Yamamoto and Y. Tezuka, *J. Am. Chem. Soc.*, 2010, **132**, 10251–10253.
18 S. Honda, T. Yamamoto and Y. Tezuka, *Nat. Commun.*, 2013, **4**, 1574.
19 X. Wan, T. Liu and S. Liu, *Biomacromolecules*, 2011, **12**, 1146–1154.
20 G. Kang, L. Sun, Y. Liu, C. Meng, W. Ma, B. Wang, L. Ma, C. Yu and H. Wei, *Langmuir*, 2019, **35**, 12509–12517.
21 H. Wei, S. H. Chu, J. Zhao, J. A. Pahang and S. H. Pun, *ACS Macro Lett.*, 2013, **2**, 1047–1050.

22 M. A. Cortez, W. T. Godbey, Y. Fang, M. E. Payne, B. J. Cafferty, K. A. Kosakowska and S. M. Grayson, *J. Am. Chem. Soc.*, 2015, **137**, 6541–6549.
23 G. Kang, L. Sun, Y. Liu, C. Meng, W. Ma, B. Wang, L. Ma, C. Yu and H. Wei, *Small*, 2016, **12**, 2750–2758.
24 C. Wang, X. Huang, L. Sun, Q. Li, Z. Li, H. Yong, D. Che, C. Yan, S. Geng, W. Wang and D. Zhou, *Chem. Commun.*, 2022, **58**, 2136–2139.
25 N. Nasongkla, B. Chen, N. Macaraeg, M. E. Fox, J. M. J. Frèchet and F. C. Szoka, *J. Am. Chem. Soc.*, 2009, **131**, 3842–3843.
26 B. Chen, K. Jerger, J. M. J. Frèchet, J. Francis and C. Szoka, *J. Controlled Release*, 2009, **140**, 203–209.
27 Y. Wang, R. Zhang, F.-S. Du, Y.-L. Wang, Y.-X. Tan, S.-P. Ji, D.-H. Liang and Z.-C. Li, *Biomacromolecules*, 2011, **12**, 66–74.
28 N. Nasongkla, B. Chen, N. Macaraeg, M. E. Fox, J. M. J. Frèchet and F. C. Szoka, *Biomacromolecules*, 2016, **17**, 69–75.
29 K. Hagita and T. Murashima, *Macromolecules*, 2021, **54**, 8043–8051.
30 T. C. Oćonnor, T. Ge and G. S. Grest, *J. Rheol.*, 2022, **66**, 49.
31 A. Borger, W. Wang, T. C. Oćonnor, T. Ge, G. S. Grest, G. V. Jensen, J. Ahn, T. Chang, O. Hassager, K. Mortensen, D. Vlassopoulos and Q. Huang, *ACS Macro Lett.*, 2020, **9**, 1452–1457.
32 T. C. O'Connor, T. Ge, M. Rubinstein and G. S. Grest, *Phys. Rev. Lett.*, 2020, **124**, 027801.
33 F. M. Gaitho, M. Tsige, G. T. Mola and G. Pellicane, *Polymers*, 2018, **10**, 324.
34 M. Megnidio-Tchoukouegno, F. M. Gaitho, G. T. Mola, M. Tsige and G. Pellicane, *Fluid Phase Equilib.*, 2017, **441**, 33–42.
35 L. Liu, S. Parameswaran, A. Sharma, S. M. Grayson, H. S. Ashbaugh and S. W. Rick, *J. Phys. Chem. B*, 2014, **118**, 6491–6497.
36 Y. Song, R. Jiang, Z. Wang, L. Wang, Y. Yin, B. Li and A.-C. Shi, *Macromol. Theory Simul.*, 2016, **25**, 559–570.
37 Y. Song, T. Xie, R. Jiang, Z. Wang, Y. Yin, B. Li and A.-C. Shi, *Langmuir*, 2018, **34**, 4013–4023.
38 T. E. Gartner III, F. M. Haque, A. M. Gomi, S. M. Grayson, M. J. A. Hore and A. Jayaraman, *Macromolecules*, 2019, **52**, 4579–4589.
39 Y. Song, R. Jiang, Z. Wang, Y. Yin, B. Li and A.-C. Shi, *ACS Omega*, 2020, **5**, 9366–9376.
40 R. M. Ziolek, P. Smith, D. L. Pink, C. A. Dreiss and C. D. Lorenz, *Macromolecules*, 2021, **54**, 3755–3768.
41 R. M. Ziolek, A. Santana-Bonilla, R. López-Ríos de Castro, R. Kühn, M. Green and C. D. Lorenz, *ACS Nano*, 2022, **16**(9), 14432–14442.
42 L. McInnes, J. Healy and J. Melville, *arXiv preprint arXiv*, 2018, DOI: 10.48550/arXiv.1802.03426.
43 L. McInnes, J. Healy and S. Astels, *J. Open Source Softw.*, 2017, **2**, 205.
44 T. Watanabe, Y. Wang, T. Ono, S. Chimura, T. Isono, K. Tajima, T. Satoh, S.-I. Sato, D. Ida and T. Yamamoto, *Polymers*, 2022, **14**, 1823.
45 M. C. Arno, R. J. Williams, P. Bexis, A. Pitto-Barry, N. Kirby, A. P. Dove and R. K. O'Reilly, *Biomaterials*, 2018, **180**, 184–192.
46 G. Kang, Y. Liu, L. Li, L. Sun, W. Ma, C. Meng, L. Ma, G. Zheng, C. Chang and H. Wei, *Chem. Commun.*, 2020, **56**, 3003–3006.
47 M. Zhang, P. Yu, J. Xie and J. Li, *J. Mater. Chem. B*, 2022, **10**, 2338–2356.
48 M. Yabe, K. Mori, K. Ueda and M. Takeda, *J. Comput. Chem., Jpn.*, 2019, **5**, 2018–0034.
49 W. L. Jorgensen, J. Chandrasekhar, J. D. Madura, R. W. Impey and M. L. Klein, *J. Chem. Phys.*, 1983, **79**, 926–935.
50 H. J. Berendsen, D. van der Spoel and R. van Drunen, *Comput. Phys. Commun.*, 1995, **91**, 43–56.
51 B. Hess, H. Bekker, H. J. C. Berendsen and J. G. E. M. Fraaije, *J. Comput. Chem.*, 1997, **18**, 1463–1472.
52 King's College London (2022). King's Computational Research, Engineering and Technology Environment (CREATE),DOI: 10.18742/rnvf-m076.

Chapter 4

PySoftK v2.0: tools for the analysis of polymer simulations

Molecular dynamics (MD) simulations are widely used for the study of polymer systems. However, the vast amount of data outputted by these simulations is sometimes complex to analyse, as this data is high dimensional and polymers are normally very dynamic. This causes the analysis of polymer simulation to be complicated, needing the implementation of specific algorithms to analyse the properties of the system. Therefore, there is a need for automated computational analysis workflow and tools to analyse polymer simulations with minimal user input. Such platform would also standardize polymer simulation analysis to facilitate reproducibility and replicability within the field. In this Chapter, PySoftK v2.0 is introduced as a platform with tools for the analysis of polymer simulations. The module `pol_analysis` has been added to the already published PySoftK [95] in a modular fashion. Key features of this software include a fast clustering algorithm to account for changes in the number of polymers conforming a micelle over time, a ring-stacking analysis tool, to study the ring-stacking networks in a polymer system, and an analysis tool to make molecular structures whole, even when their size is greater than half the box length, that can be applied very efficiently in every step of the simulation to properly account for the periodic boundary conditions in the analysis. Note that there is currently no openly available software that contains the tools highlighted here. These analysis tools, and the other that comprise PySoftK

v2.0, are designed to require minimal user input, and are easy to use so that very complex polymer analysis can be performed by any user, no matter their coding experience. Also, it is important to note that, while the primary focus of this software is on polymers, it has been coded with a general approach, allowing its application to other molecules like proteins or lipids. PySoftK v2.0 aims to provide a platform for automating and standardizing complex polymer analysis.
It is worth noting that I have developed all analysis tools presented in this Chapter, except for the intrinsic density analysis tool, which is based on the python package pySoftWhere. In this case, I implemented it into PySoftk and made it more generalisable.

4.1 Introduction

Polymers play a key role in materials science, spanning applications from efficient thermoelectric materials [156] to drug delivery systems [2]. Furthermore, advances in synthetic chemistry now enable the creation of increasingly complex polymers [157, 158], which means that the possibilities of creating polymeric materials with unique characteristics are endless. At the heart of understanding the behavior and properties of these polymeric materials lies the complex interplay of the molecular structure, conformational dynamics and intermolecular interactions of the constituent polymer molecules. Establishing general structure-property relationships for materials constituted by polymers imply a precise understanding in how individual properties can dynamically change by the surrounding environment of the moiety [159].

Effectively, molecular dynamics (MD) simulations can provide a realistic framework to statistically determine mathematical correlations between individual components (polymers) and the interactions among these units from a molecular description, which currently no experimental technique can resolve. For example, MD simulations have been used to study the self-assembly of polymers [160], and also how polymer-specific characteristics affect the overall material properties [69]. Additionally, MD simulations can be used to calculate physical properties of polymeric materials, such as the thermal conductivity of carbon nanotubes[161] and polymer nanofibers [162], as well as polymer chain diffusivity [163] and chiral optical properties in polymer films [14], among other characteristics [164]. It is evident that MD simulations are a powerful tool for gaining deeper insights into polymer-based materials. Nevertheless, MD simulations of polymers are often complex to set up, especially for long or complex polymer structures, such as branched topologies. This complexity arises because many computational platforms lack these options [95]. To address this gap, our group developed PySoftK [95], a modular software for generating and modeling polymers structures with different topologies.

Independent of how a MD simulation of polymers is set up, the simulation will generate a significant amount of data, often making it challenging to extract relevant information. This difficulty arises from interpreting multidimensional data, the dynamic nature of polymers, characterizing dynamic equilibria, or deriving accurate quantitative metrics for molecular-scale interactions that drive self-assembly or contribute to the function of polymeric material. Interpreting and quantifying the result from polymer MD simulations is not trivial, and it requires advanced computational tools to quantify such complex behavior. As a result, it is often not possible to replicate experimental findings or to reproduce quantifiable results [165].

While the scientific community has invested significant effort in simplifying the creation of input for polymer simulations, as exemplified by tools like PySoftK, Polymer Structure Predictor [102] or Radonpy [100], a comprehensive package for analyzing specific polymer material properties has been notably absent. To address this issue, PySoftK v2.0 introduces a toolkit designed for advanced analysis of polymer (and other soft matter) simulations, seamlessly incorporated into the existing PySoftK framework. The main advantage of PySoftK v2.0 is its provision of an unified computational framework in which modelling and analysis can be streamlined under modern software development standards. This feature ensures both provenance of data and results reproducibility. In line with the commitment of PySoftK to minimal user input and highly efficient code, PySoftK v2.0 retains these principles, enabling the analysis of large-scale systems with ease.

In this Chapter, the analysis tools will be described, providing illustrated case studies and example results, along with links to Jupyter Notebook tutorials for demonstration. The tools are divided into two analysis groups, properties of aggregates and molecular-sale interactions analysis. Furthermore, this software can be employed to investigate different kind of soft matter systems, since the implemented algorithms are agnostic in their design. This new module of PySoftK will allow the computational polymer community to use the analysis tools and also build upon on it to understand complex polymer structures and interactions. Furthermore, the existence of an open-access analysis tool will contribute to the better reproducibility and replicability of polymer computational experiments and measurements, accelerating the understanding of polymer science.

4.2 Software development

PySoftK v2.0 can be easily installed using the `pip` command strategy: `pip install pysoftk`, just like PySoftK v1.0. PySoftK v2.0 has been thoroughly tested and is compatible with Linux and macOS operating systems. It requires Python 3.9 or a higher version. Furthermore, PySoftK utilizes three python libraries: MDAnalysis v2.5 [153], pySoftWhere and Networkx [149]. In order to enhance computational efficiency of this module, parallel strategies have been implemented. These strategies utilize the `concurrent.futures` library or `MDAnalysis.lib.distances` function for parallel distance calculations.

PySoftK v2.0 has been built using modern code development practices. To ensure the preservation of functional software, Continuous Integration (CI) strategies have been implemented in PySoftK v2.0. Firstly, a peer-reviewed process through constant code committing has been carried out ensuring the detection of inconsistencies between versions of PySoftK. Secondly, a new set of tests has been developed and the new version of PySoftK has been used within each test ensuring its compatibility. Finally, code-coverage has been employed for ensuring the maximum compatibility between versions. The tests designed for PySoftK v2.0

rigorously check the functionality of all the new analysis tools. Each function is tested with various polymer types to ensure broad coverage of the analysis tools. Successful tests lead to continuous deployment, allowing for seamless updates of PySoftK. Additionally, jupyter notebook tutorials showing working examples (based on our tests) have been developed for all the analysis functions of PySoftK v2.0.

4.3 Software Overview

PySoftK v2.0 is a modular Python package that has the same functionality as PySoftK v1.0 [95] with an additional module named `pol_analysis`. This new module focuses on the analysis of polymer simulations to extract time dependent properties of polymer systems with minimal user input and high efficiency to tackle large systems. The analysis tools of this module are divided into two blocks, properties of aggregates and molecular-scale interactions analysis.

4.3.1 Properties of aggregates

The functions encompassed in this section are for the analysis of overall physical properties of soft-matter aggregates, such as polymer micelles. The functions that belong to this category are: `SCP`, `make_micelle_whole`, `rgyr`, `ecc`, `spherical_density` and `intrinsic_density`.

`SCP`: spatial clustering of polymers. Polymers are highly dynamic, which makes tracking their self-assembly process quite challenging. This complexity arises because polymers will form different sized aggregates during their self-assembly, and a given polymer can move from one aggregate to another [69]. Consequently, to properly analyse polymer aggregates, it is essential to isolate the relevant polymers belonging to the clusters while excluding others from the calculation. Additionally, this polymer clustering must be performed at each time step, so it should have minimal computational cost. For this purpose, the `SCP` class clusters polymers using a graph theory approach described in Section 2.2.1. In essence, each

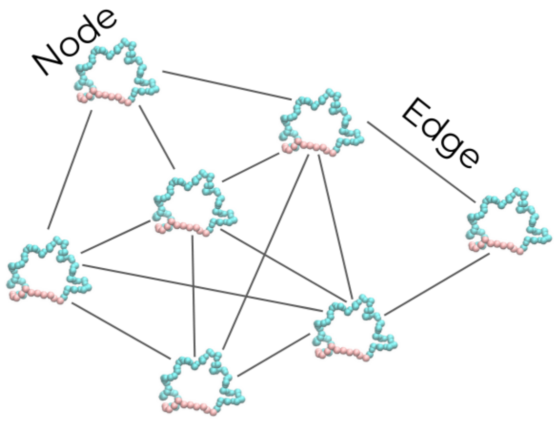


Figure 4.1: Schematic representation of how `SCP` work. The selected polymer atoms are assigned as nodes of a graph. If two atoms belonging to different polymers are at a distance lower than a specific cutoff, then an edge is created between those two nodes, becoming part of the same subgraph. This way the connection between all polymers belonging to the same cluster is tracked.

polymer is a node V of a graph G. If the distance between any specific atoms (specified by the user) of two polymers is below an user-defined cutoff, an edge is drawn between these two nodes (polymers) in $G(V,E)$. This is repeated for each polymer in the simulation. Therefore, when two polymers are within the cutoff distance, they are added to the same subgraph if one of them already belongs to an existing subgraph, or they form a new subgraph. If a molecule (node) in one subgraph is found to be connected (share an edge) with another subgraph, the two subgraphs are combined. Any polymer that is not found to be part of any aggregate is stored as a unimer. A schematic representation of the polymer graph is depicted in Figure 4.1. New graphs are generated at each time step, and the time steps selected for the calculations are set by the user.

The atom selection for determining which polymers are in contact varies depending on the polymer. The user can select as many atoms as it deems necessary to describe its system. The `SCP` tool will calculate the distances between all the selected atoms of both polymers. If any of these distances is below the cutoff it will then add them to the same subgraph. It is important to note that a lower number of selected atoms leads to faster calculation. To efficiently select the atoms for the clustering calculation, it is key to take into account polymer topology, since the atoms chosen need to be those that better represent the polymer interactions that form the polymer clusters. For instance, in the case of a linear diblock amphiphilic polymer in solution, the main forces driving its self-assembly will be the hydrophobic effect, so it should be sufficient to pick only one atom in the hydrophobic block of the polymer. However, in the case of an amphiphilic triblock ABA polymer, where A is hydrophobic, it will be necessary to pick one atom for each of the hydrophobic blocks (two atoms in total), since at least one of these needs to be close to the hydrophobic block of another polymer for both of them to be part of the same micelle. Figure 4.2 shows how the atom selection affects the clustering. Figure 4.2 (a) shows the system to cluster, two micelles formed by ABA triblock polymers, with B representing the hydrophobic monomers and A the hydrophilic monomers. Figure 4.2 (b) displays the correct clustering of Figure 4.2 (a). This result is achieved by selecting the backbone C atom of the middle monomer of the hydrophobic block. Since the hydrophobic block tightly interacts with those of other polymers in the micelle, picking atoms within this domain is a good choice. On the other hand, 4.2 (c) shows the clustering done for the same system but picking atoms at the end of the hydrophilic blocks. In this case, the clustering is not accurate because the hydrophilic atoms at the end of the polymer do not significantly contribute to micelle formation. Therefore, these atoms are not a good selection for clustering.

Furthermore, the other user input required to run the clustering tool is the cutoff distance. This distance will determine if two polymers belong to the same cluster or not. The distance can be obtained from the radial distribution function of the selected atoms, but it is recommended to use distances between 10 Å and 13 Å , since this range outputted correct results for the different polymer simulations in Chapter 3. The reason of this range is that polymers may not be interacting directly (having the smallest distance) between the atoms selected for the clustering calculation. Therefore, it is necessary to give some distance margin between these atoms, since if two atoms of two distinct polymer hydrophobic blocks are at a distance smaller than 13 Å the polymers are likely to be part of the same aggregate. Also, picking the right distance may vary per simulation and may be an iterative process: picking

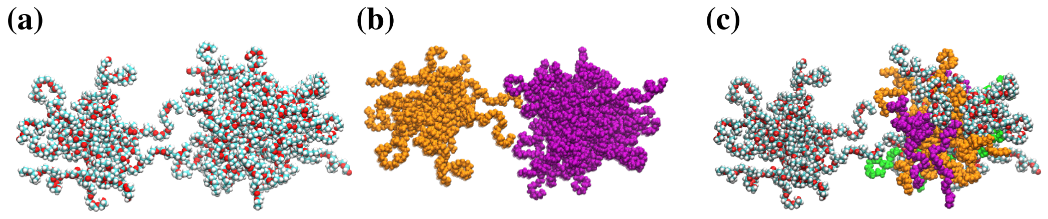


Figure 4.2: SCP atoms selection affects polymer clustering. (a) Triblock hydrophilic terminated polymer system to cluster. (b) SCP algorithm applied on the system in (a) picking the middle hydrophobic monomer C atoms for the clustering. Polymers with the same color belong to the same cluster outputted by the algorithm. Clearly, the clustering here is done correctly. (c)SCP algorithm applied on the system in (a) picking the ending hydrophilic atoms. Polymers with the same color belong to the same cluster outputted by the algorithm. It is evident that the clustering is done badly when these atoms are used. The ending hydrophilic monomers do not play a key role in the intermolecular interactions with other polymers during the self-assembly of the micelle, so they are not good clustering candidates.

distances within the 8-13 Å range, and checking by visual inspection if they output the right clustering.

The code in Figure 4.3 shows an example of how to use the SCP algorithm. From this code snippet, it is clear that to obtain the clustering of a simulation, apart from the atom names and the cutoff distance, all that is needed are the topology and trajectory of the simulation, the start and stop frames to define the time range for running the clustering algorithm, and the step parameter, which determines the number of frames to skip. Finally, `results_name` is the name of the output of the SCP function, which is a parquet file that contains a pandas data frame with the resids of the polymers grouped by the micelle they belong to per time step.

Aditionally, The code snippet in Figure 4.4 illustrates the output for the SCP clustering. This Figure displays a pandas data frame containing three columns, the first one is the time in ps, the second one is the resids of the polymers that belong to the same cluster. The polymers belonging to the same cluster are grouped by square brackets (a pythonic list). And the last column is the micelle size (number of polymers) of each of the clusters in the simulation. Note that the SCP_tutorial can be found on github, with a step by step guide on how to use the SCP tool in two different case examples.

```python
from pysoftk.pol_analysis.tools.utils_mda import MDA_input
from pysoftk.pol_analysis.tools.utils_tools import *
from pysoftk.pol_analysis.clustering import SCP

#Select trajectory
topology='topology.tpr'
trajectory='trajectory.xtc'

#Select output name
results_name='results_clustering'

#Select cluster cutoff distance
cluster_cutoff=12

#Select atom names
atom_names=['C02B', 'C01K', 'C02N']

#Select frames to run the clustering on
start=0
stop=10001
step=1

#Run clustering
clustering = SCP(topology, trajectory).spatial_clustering_run(
    start, stop, step, atom_names, cluster_cutoff, results_name)
```

Figure 4.3: Code snippet showing how to run the `SCP` function.

```
time     micelle_resids                          micelle_size
0.0     [[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]]         [10]
100.0   [[1, 2, 3, 4, 5, 6, 7], [8, 9], [10]]]    [7, 2, 1,]
```

Figure 4.4: Code snippet showing the output of the `SCP` function.

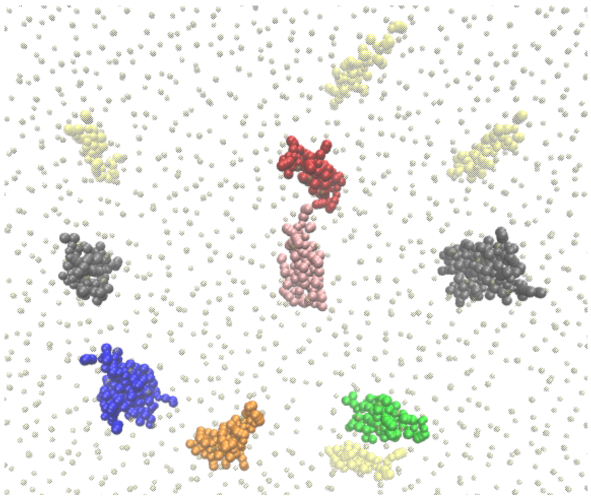


Figure 4.5: `SCP` applied to a CG protein simulation to measure protein aggregation. Since the input needed to run this tool is not exclusive to polymers, it can be applied to any other type of system to measure molecular clustering. In this case, this is the result of the `SCP` being applied on CG transmembrane proteins inserted into a membrane. This is a top-view of the protein-membrane system. Proteins colored in the same way belong to the same cluster, and non-clustered proteins are in yellow. Also the phosphate group of the lipids are represented in dark green.

Finally, the implemented agnostic algorithm can be employed to further analyse different system such as protein or lipids, and not being limited by the applications presented in this chapter. For instance, Figure 4.5 shows the result of applying the `SCP` tool to a CG protein simulation to measure protein aggregation.

`make_micelle_whole`: making a structure whole across the PBC. An important feature that is needed to perform reliable analysis in soft-matter systems is the correct connectivity (i.e. all atoms are connected through bonds) of the moieties. To enable this, different algorithms have been implemented in codes such as GROMACS or MDAnalysis, that make structures whole across the PBC. However, it has been highlighted that when molecules or aggregates occupy spaces larger than half of the simulation box size, these algorithms are not reliable to reconstruct the connectivity across the PBC [166]. In contrast, PySoftK `make_micelle_whole` tool is able to obtain the coordinates of molecular structures broken across the PBC, even if their size is greater than half the box size. Theoretical details on this algo-

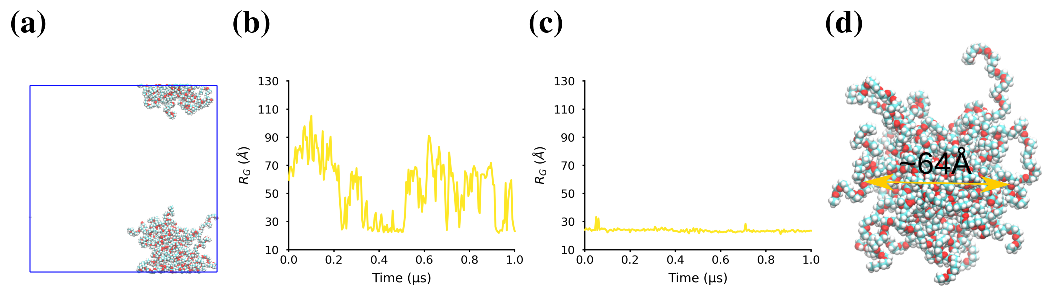


Figure 4.6: Effect of `make_micelle_whole` function on the calculation of the radius of gyration. Result from running MDAnalysis `radius_of_gyration()` on the system broken across the PBC as depicted in (a) with the (b) `pbc=True` MDAnalysis option and (c) on the whole coordinates obtained with `make_micelle_whole`. It is clear that `radius_of_gyration(pbc=True)` does not take PBC effects properly into account during the radius of gyration calculation. (d) Snapshot of the micelle studied made whole with the radius measured with VMD.

rithm are described in Section 2.4.1. Therefore, `make_micelle_whole` not only provides a clearer representation of the system, but also ensures the accurate computation of physical properties across the PBC. For example, certain functions of MDAnalysis, such as `radius_of_gyration()` or `moment_of_inertia()`, may produce erroneous results when applied to broken molecules, leading to artefacts in the simulation analysis. For instance, Figure 4.6 shows the difference between using solely the MDanalysis `radius_of_gyration(pbc=True)` function (Figure 4.6 (b)) and the MDanalysis `radius_of_gyration()` function applied on the whole structure created by `make_micelle_whole` (Figure 4.6 (c)) of a broken micelle depicted in Figure 4.6 (a). From Figure 4.6 (b), it is clear that MDAnalysis with the `pbc=True` parameter is not able to calculate the correct radius of gyration of a broken micelle. However, Figure 4.6 (c) shows that when this same function `radius_of_gyration()` is applied on the whole coordinates obtained with `make_micelle_whole` the radius of gyration is properly calculated. Therefore, it is evident that ensuring the correct reconstruction of the connectivity of a system across the PBC is essential for the accurate analysis of polymer simulations.

Furthermore, Figure 4.7 illustrates that the `make_micelle_whole` tool offers a superior algorithm for producing connected moiety. Figure 4.7 (a) displays a micelle that is broken across the PBC. Figure 4.7 (b) shows the same micelle but

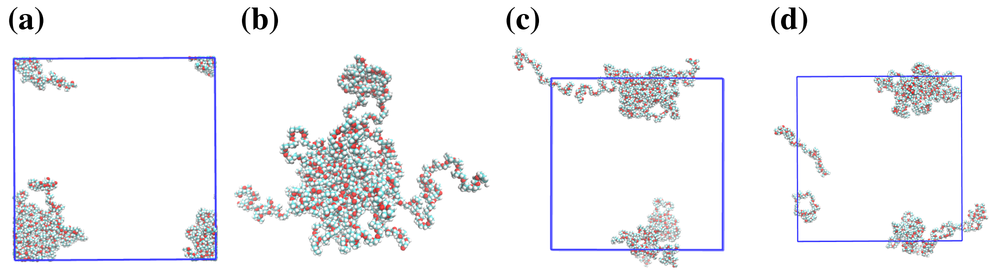


Figure 4.7: Effect of `make_micelle_whole` on structure that is greater than half the box length. Trying to make the micelle in (a) which is broken across the pbc and that is greater than half the box length in one dimension using (b) `make_micelle_whole` (c) MDAnalysis transformation (d) GROMACS `gmx trjconv -pbc mol` command. It is clear that only the PySoftK analysis tool `make_micelle_whole` is able to properly make the micelle whole across the PBC even if one of its dimensions is greater than half the box length.

successfully reconstructed with PySoftK's `make_micelle_whole`. On the other hand, Figure 4.7 (c) and (d) demonstrate that MDAnalysis v2.5 and GROMACS, respectively, are not able to reconstruct the connected bond structure of the micelle using the same input as files as `make_micelle_whole`. A jupyter notebook tutorial titled `example_mdanalysis_vs_make_micelle_whole` illustrates an example of a polymer micelle, broken across multiple sites due to the PBC, and of large size. This tutorial shows that MDAnalysis transformation tools fail to make the micelle whole, but PySoftK's `make_micelle_whole` outputs the micelle displaying the expected spatial bond-connected distribution. Thereby, using this function to obtain the coordinates of a soft-matter system is the most reliable method to perform analysis of simulations.

The `make_micelle_whole` class is composed of three functions. The first one, `obtain_largest_micelle_resids`, can be used to extract the largest micelle polymer residues from the pandas dataframe obtained with `SCP`. This function was implemented because, in many cases, the structure of interest is the largest aggregate within the simulation. The code snippet in Figure 4.8 demonstrates how to use this tool. This Figure shows that the `obtain_largest_micelle_resids` function only requires the result from `SCP` as input. It then produces a list, with as many entries as time steps evaluated, containing the resids of the molecules that belong to the largest cluster at each time

```python
from pysoftk.pol_analysis.tools.utils_mda import MDA_input
from pysoftk.pol_analysis.tools.utils_tools import *
from pysoftk.pol_analysis.clustering import SCP
from pysoftk.pol_analysis.make_micelle_whole import micelle_whole

#Select trajectory
topology='topology.tpr'
trajectory='trajectory.xtc'

#Load clustering resids
resids_total='results_clustering.parquet'

#Run function
result = micelle_whole(topology, trajectory).
    obtain_largest_micelle_resids(resids_total)
```

Figure 4.8: Code snippet showing how to use `obtain_largest_micelle_resids`.

step.

Furthermore, another pivotal function in the `make_micelle_whole` class is `make_cluster_whole`, which is its core function. This function performs the calculations to obtain the coordinates of the connected bond structure. The output of this function is a numpy array with the atom coordinates. An example of how to run this function is illustrated in Figure 4.9. From here, it is clear that the only inputs required are: the resids on which the calculation will be performed, the time steps for the calculation and the resname of the molecules forming the structure. This function can accept multiple resnames as needed.

Finally, the last function of this class, `obtain_snapshot`, is designed to output an user-defined `pdb` file for the whole structure at the selected frame. This can be convenient if the user wants to input the structure into a different simulation box, for example, if the user wants to study the interaction of a micelle with a membrane, it can obtain the micelle pdb file with this function and then insert it into the membrane box. Figure 4.10 displays a code snippet showing how to use the function. The code in Figure 4.10 will write the pdb file into the directory and with the name specified in `results_name`.

```python
from pysoftk.pol_analysis.tools.utils_mda import MDA_input
from pysoftk.pol_analysis.tools.utils_tools import *
from pysoftk.pol_analysis.clustering import SCP
from pysoftk.pol_analysis.make_micelle_whole import micelle_whole

#Select trajectory
topology='topology.tpr'
trajectory='trajectory.xtc'

#Select times to run micelle_whole on
start=0
stop=10001
step=1

#Load clustering resids
resids_total='results_clustering.parquet'

#Obtain the resids of the structure of interest
resids = micelle_whole(topology, trajectory).
    obtain_largest_micelle_resids(resids_total)

#Run make_cluster_whole function
atom_pos = micelle_whole(topology, trajectory).
    running_make_cluster_whole(['LIG'], resids, start, stop, step)
```

Figure 4.9: Code snippet showing how to use `running_make_cluster_whole`.

Thus, PySoftK's implementation of `make_micelle_whole` provides a more reliable algorithm to reconstruct structures broken by the PBC while enabling the user to produce a single and streamlined workflow. Furthermore, this is done with minimal input from the user. An illustrated and step-by-step guide on how to use all the functions in the `make_micelle_whole` class is provided in the jupyter notebook tutorial `micelle_whole_tutorial`. Furthermore, in this tutorial, it is shown how different polymer aggregates can be selected from the simulation, not just the largest micelle. In this Chapter, all the examples displayed are with the largest micelle for simplicity. Nevertheless, the `SCP` tool outputs a pandas dataframe with the resids of polymers belonging to all aggregates in the system.

```
from pysoftk.pol_analysis.tools.utils_mda import MDA_input
from pysoftk.pol_analysis.tools.utils_tools import *
from pysoftk.pol_analysis.clustering import SCP
from pysoftk.pol_analysis.make_micelle_whole import micelle_whole

#Select trajectory
topology='topology.tpr'
trajectory='trajectory.xtc'

#Select times to run micelle_whole on
start=0
stop=10001
step=1

#Name of results file
results_name='Desktop/system_frame_2.pdb'

#Load clustering resids
resids_total='results_clustering.parquet'


#Obtain the resids of the structure of interest
resids = micelle_whole(topology, trajectory).
    obtain_largest_micelle_resids(resids_total)

#Run make_cluster_whole function
atom_pos = micelle_whole(topology, trajectory).
    running_make_cluster_whole(['LIG'], resids, start, stop, step)

#Obtaining pdb of the system at frame 2
snapshot = micelle_whole(topology, trajectory).obtain_snapshot(
    results_name, atom_pos[2], resids[2], ['LIG'], 2)
```

Figure 4.10: Code snippet showing how to use `obtain_snapshot`.

Therefore, choosing other aggregates to analyse, such as the two smallest aggregates per time step (as illustrated in the tutorial), it is as easy as manipulating the pandas data frame to select the desired data.

`rgyr`: radius of gyration. This function allows the user to easily calculate the radius of gyration of a structure that is not always formed by the same molecules throughout the simulation. It utilises the MDAnalysis function `radius_of_gyration()`, but allows users to specify the atom positions and resids on which to perform this calculation at each time step. The radius of gyration calculation is given by the following formula:

$$R_g = \sqrt{\frac{\sum_i m_i \vec{r}_i}{\sum_i m_i}} \tag{4.1}$$

Where R_g is the radius of gyration, i is a specific particle of the system, m_i is the mass of particle i and $\vec{r}_i$ is the distance of particle i from the center of mass of the selection. Figure 4.6 shows the comparison between using the MDAnalysis radius of gyration function alone compared to the PySoftK `rgyr`, which captures the right radius of gyration of the micelle when computed on the whole coordinates from `make_micelle_whole`. An illustrative example is provided in the Jupyter notebook tutorial `rgyr_mdanalysis_vs_pysoftk`, where the radius of gyration of a micelle broken across the PBC is compared using MDAnalysis and PySoftK, demonstrating the effectiveness of PySoftK in computing this property with minimal user input. The code snippet in Figure 4.11 shows how easily the radius of gyration can be calculated for a micelle with varying number of polymers over the simulation.

As illustrated in Figure 4.11, the inputs for the `rgyr` tool are: the topology and trajectory of the simulation, `resids`, `atom_pos` and the frames (`start, stop, step`) to analyse. `resids` is a numpy array containing the resids of the molecules that belong to the structure of interest at each time step. In this way, it is very easy for the user to account for the varying conformation of the micelle. Also,

```python
from pysoftk.pol_analysis.tools.utils_mda import MDA_input
from pysoftk.pol_analysis.tools.utils_tools import *
from pysoftk.pol_analysis.make_micelle_whole import micelle_whole
from pysoftk.pol_analysis.rgyr_micelle import rgyr

#Select trajectory
topology='topology.tpr'
trajectory='trajectory.xtc'

#Load clustering resids
resids_total='results_clustering.parquet'

#Select times to run micelle_whole on
start=0
stop=10001
step=1

#Run micelle_whole
resids = micelle_whole(topology, trajectory).
    obtain_largest_micelle_resids(resids_total)
atom_pos = micelle_whole(topology, trajectory).
    running_make_cluster_whole(['LIG'], resids, start, stop, step)

#Run rgyr calculation
rgyr_micelle_whole = rgyr(topology, trajectory).running_rgyr(
    resids, atom_pos, start, stop, step)
```

Figure 4.11: Code snippet showing how to use `rgyr`.

`atom_pos` is the numpy array containing the whole atom positions. It is important to keep in mind, that `atom_pos` must include the coordinates of all the atoms contained in `resids`. The output `rgyr_micelle_whole` from Figure 4.11 is a numpy array with the radius of gyration of the micelle over the selected time frames. The jupyter notebook tutorial named `rgyr_tutorial` provides a step by step example on how to run the radius of gyration calculation using PySoftK's `rgyr` tool.

`ecc`: eccentricity calculation. The eccentricity is a measure of how much a structure deviates from a sphere, so it can be used to measure the shape of spherical-like soft-matter aggregates. The `ecc` class calculates the eccentricity of a molecular structure. The eccentricity computation employes the MDAnalysis function `moment_of_inertia()`. MDAnalysis calculates the moment of inertia tensor I for a group of N atoms, where each atom i has mass m_i with coordinates r_i relative to the center of mass of the selection:

$$I = \sum_{i=1}^{N} m_i \vec{r_i}^2 \tag{4.2}$$

Given the moment of inertia, the eccentricity is calculated using the following formula:

$$\varepsilon = 1 - \frac{I_{min}}{I_{mean}} \tag{4.3}$$

Where ε is the eccentricity value, I_{min} is the minimum moment of inertia across all axis, and I_{mean} is the mean moment of inertia over all axis. A perfect sphere corresponds to $\varepsilon = 0$, while increasing values indicate more oblong structures. Similar to the `rgyr` calculation, the `ecc` tool can account for varying number of molecules within the structure and ensures accurate calculations without artefacts from PBC by considering whole atom coordinates. Figure 4.12 illustrates how `ecc` accurately computes the eccentricity of a micelle over time compared to the MDAnalysis strategy. The input parameters of `ecc` are the same as `rgyr`, and the output is a numpy array that contains the eccentricity values of the desired structure at each time step. The tutorial `ecc_tutorial` shows how to use this function.

`spherical_density`: spherical density calculation. Understanding the internal distribution of components within molecular ensembles is essential for characterizing their structure. For molecular structures that are approximately spherical, this can be studied with the spherical density. There are numerous MD studies that measure the density of components of polymer micelles [167, 54, 160], but there are limited open-sourced codes that can compute the spherical density.

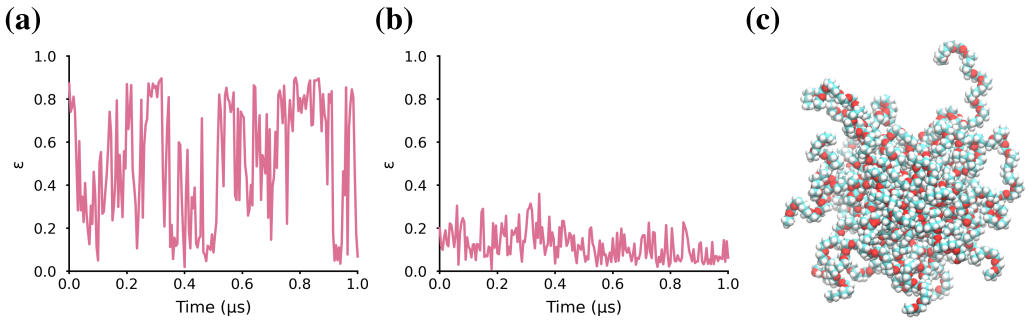


Figure 4.12: Eccentricity calculation of polymer micelle. Calculation of the eccentricity of a micelle solving Equation 6.2 over time with (a) only MDAnalysis `moment_of_inertia()` and (b) PySoftK `ecc` function. It is clear that `ecc` can easily use the correct polymers belonging to the cluster at each time step and the correct atom positions across the PBC to compute the eccentricity values, while the MDAnlysis function on its own, even with `pbc=True` is not able to compute them properly. (c) Snapshot of the micelle on which the eccentricity is being calculated, clearly it is slightly spherical, so the values from (b) are the correct ones. The micelle representation is not to scale.

PySoftK `spherical_density` tool allows users to easily calculate the spherical density over time, even for structures with varying molecule numbers throughout the simulation. It computes the average density (over time) with respect to the distance from the center of mass of the molecular structure. This is achieved by dividing up the molecular structure in spherical bins whose origin is placed at the center of mass. For each of these bins, the number of particles in them is counted and divided by the volume of the bin. A 2-d representation of this calculation is depicted in Figure 4.13.

The formula for the spherical density calculation is defined as:

$$\rho_{bin} = \frac{N_{particles}}{\frac{4}{3}\pi(R^3 - r^3)} \tag{4.4}$$

Where ρ_{bin} is the density per bin, $N_{particles}$ is the number of particles in that bin, R is the outer bin radius and r the inner bin radius. The outer and inner radius are depicted in Figure 4.13. The user can define the number of bins and size of the bins to divide the space. It is recommended to pick a range of bins large enough to account for the whole extension of the molecular structure. It is important to keep in mind that optimizing the number of bins and their width is an iterative process. This

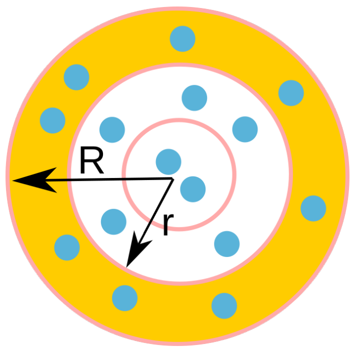


Figure 4.13: 2d-radial density calculation Representation of 2D radial density calculation, in this case the density being computed is of the yellow shell. Therefore the number of blue particles in the yellow shell will be counted and divided by the area (2-d representation) of the yellow bin. For a 3-d calculation, the number of particles would be divided by the volume instead of the surface. *R* is the outer radius and *r* the inner radius of the surface/volume calculation.

function is very intuitive, as shown in Figure 4.14 where an example is provided.

Figure 4.14 shows that the inputs needed to run `spherical_density` are: the trajectory, the topology, the type of string selection for all the molecules of the molecular ensemble, the selection of all the molecules of the ensemble, the whole atom coordinates of all the molecules of the ensemble and the names of the component of the density calculation. For the string selection type, in this example the option `'resids '` is selected. However, note that all different keywords available in MDAnalysis (i.e. 'resname', 'name', among others) can be employed as an input for this parameter. The `resids` parameter in Figure 4.14 represents the selection of all molecules belonging to the same molecular structure (per time step) on which the density calculation will be performed. Here, `resids` is a list containing the resids of all the molecules that belong to the micelle. If the string selection parameter had been a different one, such as the atom name, instead of a list of resids, this parameter would have been a list of names. `atom_pos` is a numpy array with the whole atom positions of molecules listed in `resids`. Finally, `names_total` are the atom names of the component for the density calculation, which needs to be a list of `str`. The output `spherical_density_whole` is a numpy array with the average density over time per bin. The bins are stored in the array `binned_space`.

Additionally, the `spherical_density_water` class provides a cus-

```
from pysoftk.pol_analysis.tools.utils_mda import MDA_input
from pysoftk.pol_analysis.tools.utils_tools import *
from pysoftk.pol_analysis.make_micelle_whole import micelle_whole
from pysoftk.pol_analysis.spherical_density import
    spherical_density

#Select trajectory
topology='topology.tpr'
trajectory='trajectory.xtc'

#Load clustering resids
resids_total='results_clustering.parquet'

#Select times to run micelle_whole on
start=0
stop=10001
step=1

#Run micelle_whole
resids = micelle_whole(topology, trajectory).
    obtain_largest_micelle_resids(resids_total)
atom_pos = micelle_whole(topology, trajectory).
    running_make_cluster_whole(['LIG'], resids, start, stop, step)

#Selecting molecule atoms for density calculation
names_total =['C00A', 'C009', 'C008', 'C007', 'C006', 'C005', '
    C004', 'C003', 'C002', 'C001', 'C000', 'C00L', 'C00O', 'C00P',
    'C00Q', 'C00R', 'C00S', 'C00T']

#Run density calculation
spherical_density_whole, binned_space = spherical_density(topology
    , trajectory).run_density_calc('resid ', resids, atom_pos,
    names_total)
```

Figure 4.14: Code snippet showing how to run the spherical density function.

tomized version of this algorithm to investigate only the water density. It is a separate function because to properly calculate the distances of water with respect to the center of mass of the molecular structure, the water coordinates need to be wrapped around the whole coordinates of the structure. Since this process is computationally more expensive, it was created in a different class. It works in exactly the same way as the `spherical_density`, but the atom names of the solvent need to be inputted by the user. In the case of water, it is enough to only select the water oxygens for the density calculation. Examples of both spherical density calculations, for the micelle components and water, are illustrated in the tutorial `spherical_density`. This tutorial shows how to calculate and plot the density in both cases.

`intrinsic_density`: intrinsic density calculation. In cases where molecular structures have irregular internal and interfacial structures, spherical density approaches do not work. For these cases, intrinsic interface techniques have been previously employed to analyse the distribution of components within such irregular structures [168]. A computational method that has been used to study the intrinsic density of polymer micelles with a distinct core-shell structure, is the intrinsic core-shell interface method (ICSI) developed by Ziolek *et al* [160]. This approach divides the molecular structure of interest into the core and shell region with an intermediate region in between these. The masses of the core and shell are determined, and the volumes of the core, shell, and interface are calculated. The total density of the molecular ensemble is then computed by dividing the total mass by the sum of the volumes of the core, shell, and interface, which accounts for the varying properties of the core and shell. PySoftK's `intrinsic_density` class harnesses the existing ICSI method from the Python package PySoftWhere to perform intrinsic density calculations. However, in PySoftK, this method is implemented modularly and can easily operate on the whole reconstructed coordinates of the system (provided by `make_micelle_whole`). Also, it can handle varying numbers of molecules constituting the structure of interest at each time step. The

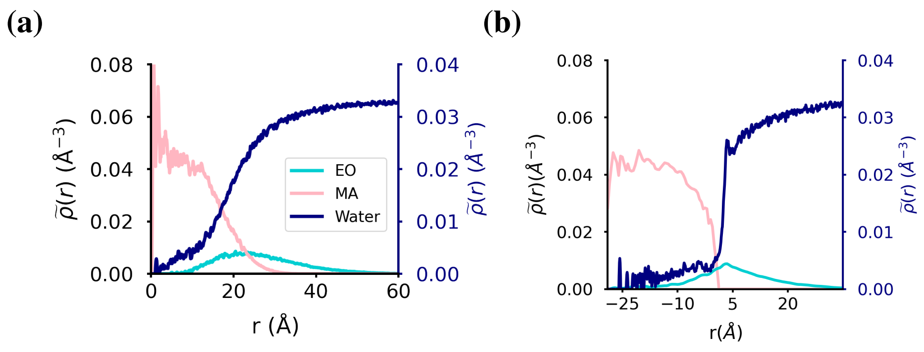


Figure 4.15: Density calculation comparison. Density calculation of a spherical micelle formed by PEO-PMA polymers using: (a) `spherical_density` (b) `intrinsic_density`. The results are very similar, since the micelle is spherical and has a clear core-shell interface. Therefore, in this case both methods can be used to determine the distribution of components within the micelle.

usage of this function closely resembles that of the spherical density function. Additionally, there is a `intrinsic_density_water` class for the computation of the intrinsic density of water.

The `intrinsic_density` class outputs a a numpy array with the average densities (over time) with respect to the distance to the core-shell interface, being 0 the location of this interface. It also outputs another numpy array with the values of the bins used in the density calculation. Figure 4.15 shows density calculations of the same system using the spherical density tool and intrinsic density tool respectively. It can be seen that PySoftK spherical density tools output the density as a function of the distance from the center of mass of the micelle (Figure 4.15 (a)), while the intrinsic density outputs the density as a function of the distance from the core-shell interface (Figure 4.15 (b)). Therefore, negative distance values in the intrinsic density represent atoms within the core. In this case, since the micelle was spherical, both density methods produce similar results.

The tutorial titled `intrinsic_density_tutorial` covers how to run the intrinsic density of the hydrophobic component of a polymer micelle and the water of the system with PySoftK's intrinsic density tools.

4.3.2 Molecular-scale interactions analysis

The functions encompassed in this section, are the tools for the analysis of different aspects of intermolecular interactions of molecules within a system, such as ring-stacking, hydration or contacts between the system's components. This analysis is useful to understand the driving forces behind the molecular structure formation. Furthermore, it is important to note that, as all the other functions, these are not specific to polymers, and they can all be used in other soft-matter systems.

`contacts`: quantification of intermolecular interactions. Understanding which and how molecules interact within a molecular ensemble is essential for comprehending the forces driving the formation of a molecular structure. This can be achieved by computing the intermolecular contacts between molecules of a system. The `contacts` class calculates the contacts between polymers by measuring the distance between selected polymer atoms. If the intermolecular distance between two selected polymer atoms falls below a user-defined cutoff, it is considered a contact. This calculation is done per timestep, and the atoms and distance cutoff are defined by the user. The values of the distance cutoff can oscillate depending on the aim of the analysis. Nevertheless, for the quantification of standard atomic-interactions, distance cutoff values between 4 Å and 7 Å have been used previously in the literature[69, 169, 170]. A visual representation of intermolecular contacts between two polymers in the same micelle is depicted in Figure 4.17 (a). The distinctive feature of the `contacts` class is that it uses the output from the `make_micelle_whole` tool, ensuring that, at each timestep, the molecules considered for contact analysis are made whole, thus enabling accurate distance calculations.

Figure 4.16 presents a code snippet illustrating the usage of the `contacts` tool. It requires several input files: the topology and trajectory of the simulation, the resids of the molecules on which the analysis will be performed on (`resids`), the whole atomic coordinates of the molecular ensemble (`atom_pos`), the atom names for both groups involved in the contact calculation (`MA_names`, `MA_names`)

```python
from pysoftk.pol_analysis.tools.utils_mda import MDA_input
from pysoftk.pol_analysis.tools.utils_tools import *
from pysoftk.pol_analysis.make_micelle_whole import micelle_whole
from pysoftk.pol_analysis.contact_analysis import contacts

#Select trajectory
topology='topology.tpr'
trajectory='trajectory.xtc'

#Load clustering resids
resids_total='results_clustering.parquet'

#Select cutoff contact distance
cutoff=10

#Select times to run micelle_whole on
start=0
stop=10001
step=1

#Select atom contact names
MA_names = ['C027', 'C023', 'C021', 'C02H', 'C02L', 'C02P',
    'C02T', 'C02X', 'C00U', 'C00R', 'C00P', 'C00L']

#Run micelle_whole
resids = micelle_whole(topology, trajectory).
    obtain_largest_micelle_resids(resids_total)
atom_pos = micelle_whole(topology, trajectory).
    running_make_cluster_whole(['LIG'], resids, start, stop, step)

#Run contacts calculation
contacts_matrix = contacts(topology, trajectory).run_contacts_calc
    (resids, atom_pos, MA_names, MA_names, cutoff)
```

Figure 4.16: Code snippet showing how to run the `contacts` function.

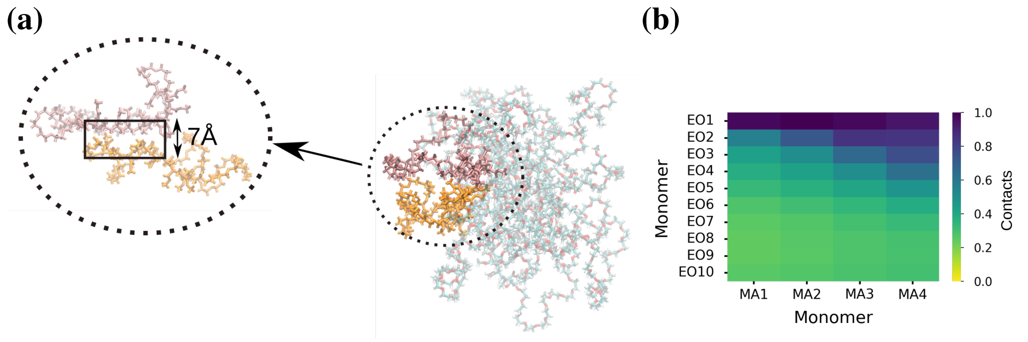


Figure 4.17: Normalized `contacts` output. (a) Snapshot of two cyclic polymer within a micelle that are in contact. The distance between some of their atoms is below 7 Å. This is represented in the figure by the atoms inside the rectangle with a vertical size of 7 Å. These inter-molecular interactions would be picked up by the `contacts` algorithm as contacts. (b) Output of the calculation of the intermolecular contacts of PEO-PMA polymers forming a micelle. The output matrix is represented as a heatmap. The color bar represented the normalized number of contacts throughout the analysis, with 1.0 being the maximum number of contacts. The rows and columns represent the contact groups of the calculation.

and the cutoff distance (`cutoff`). In Figure 4.16, intermolecular contacts are calculated between atoms of the same type, which is why `MA_names` is selected for both contact groups. However, users have the flexibility to calculate contacts between different atom groups, as demonstrated in Figure 4.17 (b), which illustrates normalized contacts between PMA and PEO atoms. The output of the `contacts` class, in this case `contacts_matrix`, is a contact matrix containing the total number of intermolecular contacts between atoms of both contact groups. Rows correspond to atoms belonging to the first contact group, and columns correspond to the second contact group. Figure 4.17 (b) shows a heatmap representation of the normalized matrix generated by `contacts` illustrating intermolecular EO-MA interactions of a PEO-PMA polymer. It is important to note that to ensure consistency in performing this analysis, the frames from the user-provided trajectory employed in `SCP` and `make_micelle_whole` need to be the same, as depicted in Figure 4.16. For a step-by-step guide on using the `contacts` tool and recommendations for result plotting, refer to the tutorial named `contacts_tutorial`.

`RSA`: Ring Stacking Analysis. Non-covalent forces are the main driving mechanism in which individual moieties can interact with each other in real space.

Among them, an important interaction is the one produced between aromatic units adjacently stacked. Thus, the so-called ring-ring stacking interaction has been found to be the source of many collective phenomena ranging from DNA base pairing, protein-drug binding, and "through-space" charge transfer in conjugated polymers [171]. Computer simulations have played a crucial role providing a detailed insight to peruse relevant structure-property relationships at atomistic resolutions. Specifically, MD simulations can supply dynamical information to investigate the organisation of molecular building blocks in a non-covalent manner. In the context of ring stacking, it has been widely studied experimentally, however there is only one study that has provided a network-level picture of this phenomenon from MD simulations [172]. Having a molecular picture from MD simulations of ring interactions will be key to control the optical properties of these materials, but this analysis can be quite complex.

To address this issue, a new class to establish ring-ring interactions based on user-provided geometrical parameters has been developed and included in PySoftK v2.0. The implemented algorithm consists of three stages. Firstly, all atoms belonging to aromatic (conjugated) rings are detected for a single selected moiety. Secondly, pairs of molecular complexes are screened employing a user-provided cutoff to define molecules in close contact. Finally, moieties which fulfill the previous condition are selected for computing the distances between aromatic units. Since this interaction is highly directional, the algorithm has been designed to compute the best fitting plane for the atoms computed in step one, providing the normal vectors and whose projection (dot product) defines the angle between aromatic units, as explained in detail in Section 2.4.2. Therefore, the `RSA` class allows users to identify ring stacking patterns within a simulation and obtain the stacking polymer network and its evolution over time. This tool implemented in PySoftK enables the user to perform this analysis based on minimal input parameters. This algorithm has been tested and can be used on any type of conjugated polymer simulation, such as a polymer melt, as illustrated in Figure 4.18.

The code snippet in Figure 4.19 shows how to use this function. This code

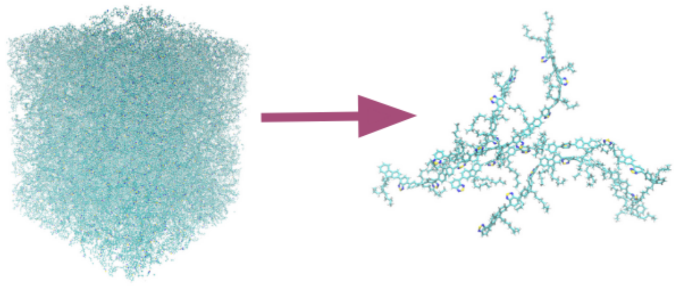


Figure 4.18: Example of ring-ring stacking calculation using the `RSA` class on a polymer melt. `RSA` is able to obtain ring stacking in complicated and large systems.

illustrates that the required input parameters are the topology and trajectory of the simulation, a ring-ring distance cutoff (`dist_cut`), a maximum angle between ring units (`angle_cut`), and the name of the output file name (`output`). The cutoff distance represents the maximum spatial separation allowed between aromatic units for the calculation of their stacking angle. The value has been set to a default of 4 Å . Based on previous research, a maximum of 7 Å is recommended [172] for the cutoff distance. Similarly, the angle cutoff value will depend on the system and the properties of interest. The angle cutoff defines the angle range between two rings that is counted as stacking (angle between rings either smaller than the cutoff angle or larger than $\pi - cutoff$). The default recommended value is 30 °[172], but for very tight ring stacking, smaller values can be used. The output of the `RSA` tool is a pandas dataframe with all the resids of the polymers that are stacked and also the pdb files of pairs of stacked polymers. Furthermore, the `RSA` class has a another function `find_several_rings_stacked` that uses as input the output from the rings stacking calculation and outputs the network of polymers that are interacting via ring stacking. It creates a graph object ($G(V,E)$) in the same manner that was defined in `SCP` class to keep track of the polymers that are connected. The tutorial titled `RSA_tutorial` shows how to run both RSA functions.

Thus, PySoftK v2.0 allows users to conduct ring-stacking analysis, including the possibility to investigate the formation of dynamical networks in MD simulations. Therefore, this class makes ring stacking calculations available to any user, contributing to the progress of computational research on conjugated polymers.

```python
from pysoftk.pol_analysis.tools.utils_mda import MDA_input
from pysoftk.pol_analysis.tools.utils_tools import *
from pysoftk.pol_analysis.make_micelle_whole import micelle_whole
from pysoftk.pol_analysis.ring_ring import RSA

#Select trajectory
topology='topology.tpr'
trajectory='trajectory.xtc'

#Define the angle and distance cutoff
dist_cut=5
angle_cut=30

#Define the output name
output='results.parquet'

#Run RSA calculation
rsa=RSA(topology, trajectory).stacking_analysis(dist_cut,
    angle_cut, output)
```

Figure 4.19: Code snippet showing how to run the `RSA` tool.

`hydration`: solvation calculation. Solvation analysis plays a crucial role in understanding the hydrophobic characteristics of PNP. It helps in elucidating the solvation shell around these NP and predicting the encapsulation behavior of hydrophobic and hydrophilic drugs within them [69, 93]. Unfortunately, there is currently no openly available software that facilitates easy solvation calculations in this context. MDAnalysis has the class `MDAnalysis.analysis.waterdynamics`, but it focuses on the dynamics of water and the interactions of water with other molecules via hydrogen bonds, which may not be applicable to study the solvation of polymers that cannot form hydrogen bonds. PySoftK's `hydration` class fills this gap by providing a straightforward method for quantifying solvation. It determines the number of solvent molecules within the first solvation shell of the specified molecules.

The `hydration` class operates similarly to the `contacts` class, where distances between selected atoms of the molecules and specific solvent atoms are computed. If the distance is shorter than a user-defined cutoff, the atom is considered to be solvated. Both atom groups are inputted by the user. Users input both atom groups and define the cutoff distance, which is recommended to be no greater than the second peak in the radial distribution function between the solvent and the polymer of interest. This is because this distance corresponds to the first hydration shell. When using water as the solvent, it is recommended to select just the oxygen water atoms to speed up calculations. Like all other tools in PySoftK v2.0, the solvation code can easily take into account the varying number of molecules conforming the structure and the whole atomic coordinates at each time step. The code snippet in Figure 4.20 shows how to use this function.

Thus, as it can be seen in Figure 4.20 the required inputs for this function are: the topology and trajectory files (`water_topology, water_trajectory`), the frames for the calculation (`start, stop, step`), the resids of the molecules for the solvation calculation (`resids`), the whole atomic position of all molecules in `resids` (`atom_pos`), the name of the solvent molecules (`water_name`), the selected polymer atoms for the solvent calculation (`molecule_names`) and the cutoff distance to compute the solvation shell (`cut`). The output, in this case `hydration_number`, is a list where each entry represents the solvation of the selected atoms across all molecules in the system. There are as many entries as the frames in the calculation. Figure 4.21 shows the average solvation number calculated for all hydrophobic monomers of a micelle. The tutorial named `hydration_tutorial` shows step by step how to use the `hydration` function to calculate the solvation of specific atoms of polymers belonging to the same micelle, and how to further process and plot the output data.

4.3.3 tools

All PySoftK analysis classes use the `pysoftk.pol_analysis.tools` module, which has two classes, `MDA_input` and `utils_tools`. The `MDA_input` class is used to load the trajectory as an MDA universe for all calculations. On

```python
from pysoftk.pol_analysis.tools.utils_mda import MDA_input
from pysoftk.pol_analysis.tools.utils_tools import *
from pysoftk.pol_analysis.make_micelle_whole import micelle_whole
from pysoftk.pol_analysis.hydration import hydration

#Select trajectory
topology='topology.tpr'
trajectory='trajectory.xtc'
#Load clustering resids
resids_total='results_clustering.parquet'
#Select times to run micelle_whole on
start=0
stop=10001
step=1
#Run micelle_whole
resids = micelle_whole(topology, trajectory).
    obtain_largest_micelle_resids(resids_total)
atom_pos = micelle_whole(topology, trajectory).
    running_make_cluster_whole(['LIG'], resids, start, stop, step)
#Selecting molecule atoms to study hydration
molecule_names=['O00A', 'O00D', 'O00C', 'O01G', 'O01F']
#Oxygen solvent name
water_name=['OW']
#Cutoff
cut=4.5
#Run hydration
hydration_number=hydration(topology, trajectory).
    hydration_calc_run(start, stop, step, resids, atom_pos,
    water_name, molecule_names, cut)
```

Figure 4.20: Code snippet showing how to run the `hydration` class.

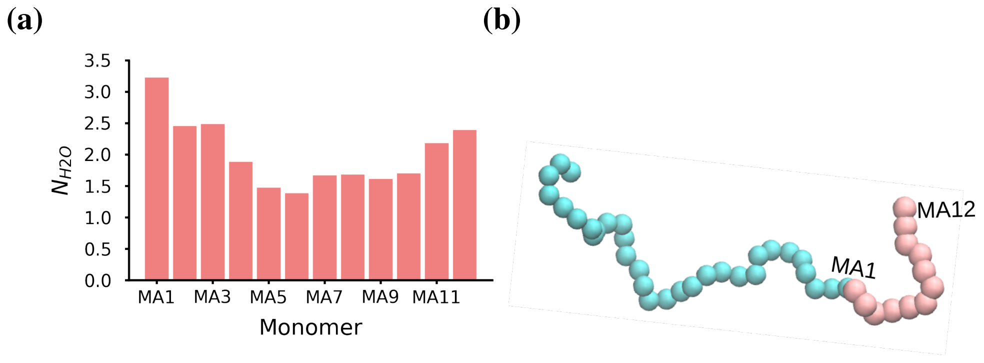


Figure 4.21: Hydration calculation of polymer micelle. Average solvation of a diblock BCP micelle. (a) Average over time of water coordination numbers for all monomers of hydrophobic block. (b) Snapshot of the diblock polymer being studied. PMA in pink and PEO in blue. Polymer representation is not to scale.

the other hand, `utils_tools` contains all recurrent functions that the analysis classes use in their computations. Therefore, both `tools.utils_mda` and `tools.utils_tools` need to be imported whenever a PySoftK analysis tool is used.

4.4 Conclusion

Pysoftk v2.0 adds a complete module for polymer simulation analysis on top of PySoftK. This module is divided into two types of analysis, properties of aggregates and molecular-scale interactions analysis. These are designed to offer a wide variety of analysis possibilities, from density calculations to ring-ring conjugated polymer interactions. One of the key features of PySoftK v2.0 is that it is able to reconstruct broken molecules across the pbc when other software, such as GROMACS or MDAnalysis fail. This is particularly convenient if the final structure wants to be inputted in a different simulation, and also to calculate all types of physico-chemical properties without artefacts, since MDAnalysis and GROMACS, do not account for the PBC properly in some of their analysis functions, as explained previously. A wide set of testing scripts have been created aiming to cover all the code to ensure its correct functionality. Furthermore, PySoftK v2.0 is designed to provide maximum flexibility to the user, so most functions output the data per frame, so that the user can decide how to represent or further process the data. Another important feature, is that although PySoftK has a special focus on polymers, the analysis module can be used for any type of molecule, since everything that is needed are just resids, resnames or atom names of interest. The goal of this module, is to create an open-source platform that allows users to analyse complex characteristics of their simulations, such as intrinsic density or ring-ring interactions, with minimal user input. PySoftK v2.0 contributes to the standardisation of polymer simulation analysis which will allow accurate comparisons across different types of polymer simulations, speeding up the *in silico* rational design of polymeric materials.

4.5 Future work

PySoftK is an evolving project driven by the collaborative efforts of its contributors, with the aim of providing a robust computational tool for the polymer research community. Therefore, it is constantly being developed. One immediate focus is the adaptation of the `make_micelle_whole` code so that it is able to make whole any molecular structure. Currently, it fails when the molecular structure is bigger than the box size, leaving a low amount of atoms broken across the PBC, as illustrated in Figure 4.22.

This is not concerning because in this cases, the number of affected atoms is normally relatively low, so analysis tools are able to compute average quantities. Additionally, simulations where the structure is bigger than the box length in at least one dimensions are not good practice, since there is a risk of the structure interacting with itself, which leads to unphysical results. Nevertheless, a function that is able to make a structure whole and place it in a new box with larger dimensions is being currently developed.

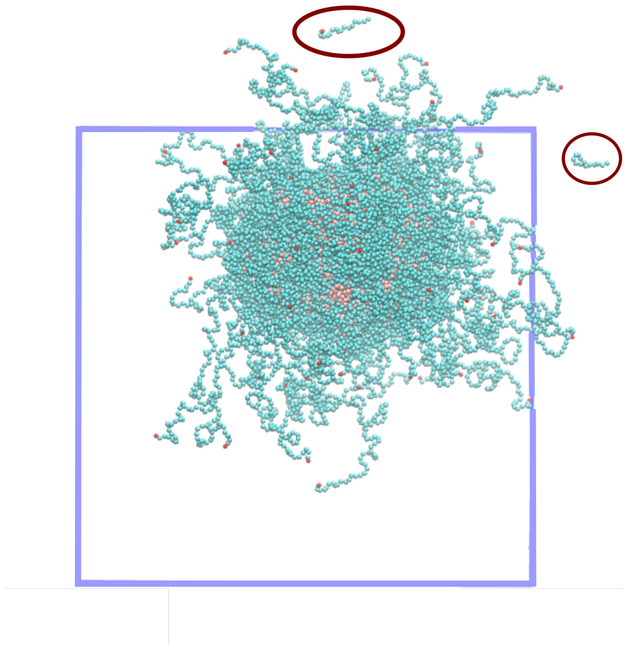


Figure 4.22: Limit of `make_micelle_whole`. This CG NP is bigger in size than the box length in one of the dimensions, so `make_micelle_whole` cannot make the micelle completely whole. Atoms inside the red circle are the ones that are not clustered, PBC box size is in dark blue.

Chapter 5

Therapeutic peptides are preferentially solubilized in specific microenvironments within PEG-PLGA polymer nanoparticles

Polymeric nanoparticles have demonstrated promising results *in vitro*. However, the limited understanding of the molecular mechanisms underlying their drug solubilization and controlled release capabilities has hindered efficient clinical translation of such technologies. Polyethylene glycol-poly(lactic-co-glycolic) acid (PEG-PLGA) nanoparticles have been widely studied as cancer drug delivery vehicles. In this Chapter, unbiased coarse-grained molecular dynamics simulations are employed to model the self-assembly of a PEG-PLGA nanoparticle and its solubulization of the anticancer peptide, EEK. This nanoformulation has been shown to be efficacious against triple negative breast cancer cells *in vivo*, with the simulations in this Chapter in agreement with the physical characteristics previously reported experimentally. Following this validation process, unsupervised machine learning techniques are utilised to quantify the conformations that polymers adopt at various locations within the nanoparticle. Findings reveal that the local microenvironments formed by the various polymer conformations promote preferential EEK solubilization within specific regions of the NP. This demonstrates that these microenvironments are key in controlling drug storage locations within nanoparticles, implying that the individual polymer conformations within such nanoparticles are potentially tunable parameters for the rational design of new polymer nanoparticles for therapeutic applications.

The work described in this chapter was submitted for publication as a letter entitled Therapeutic peptides are preferentially solubilized in specific microenvironments within PEG-PLGA polymer nanoparticles in October 2023.

5.1 Introduction

In the last decades, nanomedicine, and in particular drug-loaded polymeric nanoparticles (NPs) [19, 27], have attracted significant attention as potential candidates for improving therapeutic delivery, including in tackling cancer [19]. Polymer-based NPs have several characteristics that make them ideal delivery vehicles for cancer therapeutics, such as the biodegradability of their polymeric components [26], increased circulation time of the encapsulated drug [25, 173] and high NP drug-loading capacity [17, 174]. In particular, PEG-PLGA NPs have been the subject of numerous studies, as they have been shown to successfully deliver anti-cancer drugs to tumorous tissues *in vitro* [17, 175] and in animal models [33, 176]. Both, PEG and PLGA, are FDA approved polymers [177].

PEG-PLGA block copolymers are amphiphilic, self-assembling into core-shell nanoparticles with significant drug encapsulation potential [29, 93, 178]. PEG, which principally forms the hydrophilic corona of these NPs, increases the water solubility of the NPs, leading to an increased circulation lifetime and reduced toxicity [179, 180]. Furthermore, PEG-coated NPs have a significantly reduced systemic clearance time compared to non-PEG NPs [33]. On the other hand, the PLGA blocks form the hydrophobic core of these PEG-PLGA NPs. PLGA plays an important role in the controlled drug release [33], reduces the cellular uptake of the NP by healthy cells via the endocytic route, and increases the drug circulation time *in vivo* [181]. Also, due to their amphiphilic nature, PEG-PLGA NPs can encapsulate drugs with low water-solubility inside the PLGA core [180], and hydrophilic drugs within the PEG corona [54].

While PEG-PLGA NPs have been shown to be successful in encapsulating a range of small molecule therapeutics and delivering them to cancer cells *in vitro* and in mouse models, the lack of a clear understanding of the molecular mechanisms that govern the structure of these NPs, their ability to encapsulate small molecules and their interactions with cells has prohibited them from having similar success in clinical applications [45, 48]. These processes are highly dynamic and challenging to study experimentally, however, a deep understanding at the molecular level is possible with molecular dynamics (MD) simulations. For example, Stipa *et al.* [29] have used all-atom MD simulations to study how PLGA and PLA NPs interact with paracetamol, prednisolone and isoniazid. However, in this work the drugs were randomly added to the NPs, so the encapsulation process of the drugs was not captured. Most literature only focuses on small PLGA NPs [54, 182] or simplified models consisting of only one drug molecule and very small polymer concentrations [93, 183]. It is important to note that to the author's best knowledge, there is no MD simulation study of the formation of polymer-based nanoparticles with the same polymer species, polymer rations and drugs as that of a experimentally validated NPs. Since the polymer species, length and concentration will affect the self-assembly, structure and physico-chemical characteristics of the nanoparticle, a molecular understanding of how experimentally validated PEG-PLGA NPs self-assemble and encapsulate their cargo is needed. This will not only contribute to further understand the current characteristics of this delivery vehicle, but it will also facilitate the tuning of these NPs to obtain enhanced or other desired properties.

Regarding the NP drug cargo, a type of anti-cancer therapeutic that can be encapsulated into NPs are antimicrobial peptides (AMP). AMPs are potentially selective cancer therapeutics. Their physicochemical characteristics (small, cationic and amphiphilic) [184] enable them to selectively target cancer cells, mainly due to cancer cells having more negatively charged lipids than healthy cells [185]. AMPs have been encapsulated in NP experimentally [186, 187]. NP encapsulation increases the circulation time of AMPs, protects them from degradation and also delivers a large therapeutic concentration in localized areas, increasing its selectivity and lowering the dose needed. There are very few peptide-NP simulations and none with realistic drug-polymer systems. For example: Jafari *et al.* [93] simulated the encapsulation of the AMP magainin with different concentration of PEG-PLGA, PEG and PLGA. This study only simulated one peptide per simulation with very small polymer concentrations, which lead to very small polymer aggregates. Therefore, computational MD studies studying the mechanisms of encapsulation of AMPs into experimentally validated polymer-based nanoparticles are needed.

In this Chapter coarse-grained (CG) MD simulations are used to investigate the self-assembly of a PEG-PLGA NP and the simultaneous solubilization of the ACP named EEK. This formulation has been tested against triple negative breast cancer cells *in vivo* [2]. From this simulation, the internal structure of the NP and correspondingly the local environment of EEK within the NP can be studied. This Chapter provides a molecular-level understanding of PEG-PLGA NP cargo loading by finding specific polymer conformations that drive the solubilization of EEK in different drug storage locations. This knowledge is key for the tuning of NP cargo storage location and the *in silico* rational design of NP to carry specific therapeutics.

5.2 Methods

5.2.1 Simulation details

CG MD simulation of the self-assembly of the PEG-PLGA NP along with EEK were performed using GROMACS 2020.3, [188] and the MARTINI (martini22p) force field [189], which includes polar amino acids and polarizable water. The polarizable water model more accurately represents aqueous solution [135] and the martini22p forcefield has been previously shown to accurately model PEG [190]. Also, since PEG is a non-standard MARTINI topology, an additional PEG topology [191] was used. The atomistic structure of the antimicrobial peptide, EEK, was converted into a CG representation using the CHARMM-GUI Martini Solution Maker [192, 119]. Previously reported experiments [2] used a molar ratio of EEK to PEG-PLGA (PEG average $M_n = 5000$, PLGA $M_n = 7000$) of 1:14 (weight ratio is

1:100) [2]. As well as reproducing the experimental synthesis procedure within the simulation protocol, the simulation had the same molecular weight polymer, polymer concentration and the PEG:PLGA ratio as were used in the experiments. 200 PEG-PLGA copolymers and 15 EEK peptides were randomly placed within a box of water with dimensions of $20 \times 20 \times 20$ nm^3 (the system consists of 600 417 CG beads). Subsequently, this system is inserted into a larger box with size $58 \times 58 \times 58$ nm^3 so that the various components have space to assemble into the NP. Polarizable MARTINI water is added to the system and sodium (Na^+) and chlorine (Cl^-) ions are added such that the salt concentration is 0.15 M. Periodic boundary conditions were used. The long-range electrostatic interactions were computed using the Reaction Field algorithm [152] with the cut-off distance set to 11 Å. Lennard-Jones interactions also have a cut-off distance of 11 Å. Two consecutive steepest descent minimizations were performed to remove any undesirable steric artefacts. These energy minimizations were followed by a temperature equilibration simulation in the NVT ensemble (4 ns) at 303.15 K, using the leap-frog integrator with a 20 fs time step. The velocity-rescale thermostat was used, [193] with separate thermostat groups assigned to the EEK molecules, PEG-PLGA polymers, ions and water molecules. The production simulation was run in the NPT ensemble at 350 K and at a pressure of 1 atm for 935 ns using the Parrinello-Rahman barostat [194] (all other simulation details are the same as for the NVT run).

5.2.2 Analysis methods

The analysis of the simulation was performed with Python 3.7 codes using the MDAnalysis [153] python module. Plots were produced with Matplotlib [195] and simulation visualizations with VMD [196]. All reported analysis was performed on the stationary part of the simulation.

NP structural equilibration analysis. In order to determine when the NP structure had reached stationarity, the time-evolution of the fraction of PLGA monomers found instantaneously within different regions of the NP core was calculated. This methodology has previously been used in the study of Pluronic and Tetronic micelle simulations. [147]. Structural equilibrium is considered reached when the fraction of monomers plateaus. Generally, it is important to consider the internal structure of equilibrating micelles, rather than just their bulk size, in order to accurately quantify structural equilibration in MD simulations. All analysis in this Chapter was performed on only the stationary portion of the micelle trajectory. In this case, a burn-in time of 0.6 μs was used, which means that only the final 300 ns of the simulation trajectory were considered in the analysis.

Radius of gyration.The radius of gyration of the NP, R_{gyr}, is described by Equation 5.1

$$R_{gyr} = \sqrt{\frac{1}{M}\sum_{i=1}^{N} m_i(r_i - R)^2} \tag{5.1}$$

where M is the total mass of the body, m_i is the mass of atom i and R is the mean position of all atoms. Both, the radius of gyration of the core of the NP and of the whole NP were calculated with the MDAnalysis function `radius_of_gyration()`.

Spherical density calculation. The spherical density can be used to study the distribution of components that are approximately spherical. Here, the spherical densities of the components found within the NP were calculated as a function of the distance to the center of mass of the NP. That is to say, the distances between all CG beads of all molecules and the center of mass of the NP (calculated with the MDAnalysis function `center_of_mass()`) were calculated for every time step. Additionally, for each time step, the distances were binned intp spherical bins (bins of 0.9nm width), and the number of distances in each bin was counted. This is equivalent to counting how many particles there are in each spherical bin. Then the average of the counts in each bin over all time steps was computed to obtain the time average. This final ouuput contains the average number of particles found at different distances (bins) from the center of mass of the NP. Then, the density is calculated with the following equation:

$$\rho_b = \frac{N_b}{\frac{4}{3}\pi(R^3 - r^3)} \tag{5.2}$$

Where ρ_b is the density of bin b, N_b is the number of particles in bin b, R is the outer radius of the spherical bin and r is the inner radius of the bin.

Intrinsic Core-Shell Interface (ICSI) Method. For micelles with irregular interfacial structure, intrinsic interface techniques can be used to investigate their internal and interfacial structure [147]. For this algorithm, you need to define which atoms in your system form part of the core of your micelle, and which atoms form part of the shell. First, the ICSI method performs a change of coordinates, from Cartesian coordinates $[x,y,z]$ to spherical polar coordinates $[r,\theta,\phi]$. Afterwards, the ICSI is constructed using a $n \times n$ discrete grid centered on the center of mass of the micelle and that covers the angular domains: $\cos(\theta) \in [-1,1]$ and $\theta \in [-\pi,\pi]$. Then, for each bin of the grid, the r-positions of the ICSI is defined as the position of the heavy atoms in the core found furthest from the center of mass of the micelle. If no atoms are found in a specific bin, which is possible since the micelle can have an irregular shape, the average r-value of the 8 adjacent grid bins is used. This process is repeated until the whole ICSI is covered. Therefore, the ICSI equation is given by:

$$\tilde{\rho}(r) \equiv \left\langle \sum_i \frac{\delta[r-(r_i-\xi(\theta,\phi))]}{\bar{S}_i(r)} \right\rangle \tag{5.3}$$

where r_i is the r-position of atom i and $\xi(\theta,\phi)$ is the r-position of the ICSI, and $\bar{S}_i(r)$ is the average volume of the shell. This last parameter, $\bar{S}_i(r)$, cannot be calculated analytically [147]. Therefore, Monte Carlo integration is used to calculate the shell volumes as:

$$\bar{S}_i(r) = \frac{n_i \bar{V}_{\text{box}}}{N} \tag{5.4}$$

where n_i is the number of points found in the same shell as atom i is found over all the frames analyzed, $\bar{V}_{\text{box}}$ is the average volume of the simulation box, and N is the total number of random coordinates used in the normalization procedure. Normally, using a number of random points that is an order of magnitude greater than the number of water molecules in the system, mitigates uncertainty in the intrinsic density profiles arising from the stochastic normalization procedure. Detailed information on the working of this algorithm can be found in Ziolek *et al.*[147]. Specifically in this Chapter, the MA heavy atoms were selected to form the core of the micelle, as they are the principal component of the core (this information could be inferred by the contacts maps, the hydration data and the spherical density of components). The grid selected was 21×21.

Autocorrelation function. The autocorrelation function is a statistical tool used to measure the degree of similarity between a dataset and a time-shifted version of itself. It quantifies how a signal or dataset correlates with itself at different time lags, providing insights into temporal patterns, periodicities, and the persistence of correlations within the data. In essence, it reveals how data points at one time relate to data points at other times within the same dataset, helping to uncover underlying trends and dynamics. The calculation of the autocorrelation function was done with the MDAnalysis module `MDAnalysis.lib.correlations`, which calculates the autocorrelation function of binary variables, meaning that it is either true or false at each frame for a given atom or molecule. The formula used to calculate the autocorrelations of a property x (such as potential energy, positions, velocity, etc...) at time t_0 to time $t_0 + \tau$ is:

$$C(\tau) = \langle \frac{x(t_0)x(t_{0+\tau})}{x(t_0)x(t_0)} \rangle \tag{5.5}$$

Contacts calculation. Contacts are used in this Chapter to study the interactions between the peptide, polymer and water. The contacts reported in Figure 5.4 are calculated by computing the distance between the peptide center of mass and all polymer monomer positions. If this distance was below 6.5 Å, it is counted as a contact. This is repeated for all time steps, and the final values used for the difference calculation in Figure 5.4 are the time averages. This parameter to measure hydration-contacts between molecule and water atoms, is called water coordination number. The coordination number of a central atom is the number of specific types of atoms that are within a close distance from it. This value has also been used to study hydration in Chapter 3. The cutoff distance of 6.5 Å was obtained from the minimum which directly follows the first peak of the radial distribution functions.

Water-peptide residence time. The residence time of an interaction refers to the duration or period for which two molecules or atoms remain in close proximity or are engaged in a specific interaction. This provides insight into the stability of interactions, with longer residence times suggesting more stable interactions and shorter residence times showing transient interactions. This is calculated by computing the contacts between two molecules in each frame, and counting how many frames a specific contact exists. In this Chapter the residence time of two different interactions was calculated. First the residence time of the water-peptide interactions of any peptide atom with any water molecule of interest. This provides information about whether the peptide is in contact with at least one of its atom with the water at each time step. The second residence time was of specific peptide amino acids with the water molecules, this provides information regarding how transient the interactions between a specific peptide residue and the water are. In both cases, the average over the whole trajectory of the time length of the contacts is the average residence time. The average residence time is presented as the percentage of frames the two molecules are interacting out of all analysis frames.

Contact enrichment between peptides and polymer species. Contact enrichment is a quantitative measure used to assess the preferential interaction between distinct molecular species within a given system, as revealed by molecular dynamics simulations. It quantifies the extent to which one molecule exhibits a higher tendency to interact with another molecule, relative to their respective abundances. This metric is computed by comparing the observed interaction frequency involving a specific molecule to the expected average interaction frequency, considering the relative proportions of the molecules involved. The contact enrichment $N_{\text{enrichment}}$ is calculated with the following formula:

$$N_{\text{enrichment}} = \frac{N_{\text{pol\_species}}/N_{\text{tot\_pol}}}{P_{pol\_species}/P_{\text{tot\_pol}}} \tag{5.6}$$

where $N_{\text{pol\_species}}$ is the number of contacts between a peptide and a specific polymer species, $N_{\text{tot\_pol}}$ are the total contacts between a peptide and all polymers. $P_{pol\_species}$ are the number of monomers belonging to that specific polymer species and $P_{\text{tot\_pol}}$ are the total number of all polymers. A contact between the peptide and a polymer monomer was considered if any peptide atom was within 6.5 Å of any polymer monomer. This was calculated at every time step and the values reported in Table 5.3 are the average over time for the two separate storage locations.

Dimensionality reduction and clustering. The distances chosen as the input space to generate the two dimensional UMAP embedded data are topology specific. The goal was to find the minimum number of distances that could represent the conformational complexity adopted by the polymers. Two distances were selected, the distance between the terminal monomers of the PLGA block and the distance from the terminal monomers of the PEO block. These distances were computed for all frames of the equilibrated system. These two distances were sufficient to capture the complexity of the polymer conformations. The UMAP embedded output was later clustered with HDBSCAN, Table 5.1 shows the UMAP and HDBSCAN parameters chosen.

The intrinsic density of the UMAP clusters was calculated in the same way as for the overall micelle intrinsic density using the ICSI method [147]. But instead of using all polymers, only the ones belonging to the specific cluster density being calculated were used.

	Parameters
n_neighbours	25
min_cluster_size	45
cluster_selection_epsilon	0.85

Table 5.1: UMAP and HDBSCAN parameters. UMAP (*n_neighbours*) and HDBSCAN (*min_cluster_size* and *cluster_selection_epsilon*) parameters.

Cargo interactions with polymer clusters. After obtaining the polymer cluster conformations, the investigation focused on how peptides interacted with the different clusters. To analyse this, the enrichment of the various polymer conformational clusters near the two distinct peptide storage locations was calculated. To characterise the local environment of the storage location located in the core of the NP, all polymers that had their last PLGA monomer within a radius of 25 Å from the center of mass of the NP were considered. For the core/corona interface storage location environment, the focus was on polymers whose last PLGA monomer was found between 65 Å and 90 Å from the center of mass of the micelle. Subsequently, the enrichment of the clusters, $C_{\text{enrichment}}$, was determined using the following formula:

$$C_{\text{enrichment}} = \frac{N_{i,\text{local}}/N_{\text{tot,local}}}{N_i/N_{\text{total}}} \tag{5.7}$$

Here, $N_{i,\text{local}}$ is the number of polymers from cluster i in the local environment, $N_{\text{tot,local}}$ is the total number of polymers in the local environment, N_i is the number of polymers from cluster i in the entire NP and N_{total} is the total number of polymers in the entire NP. Figures 5.8 (a) and5.8 (b) show the values of $C_{\text{enrichment},i}$ for the core/corona interface and core environments, respectively. These calculations provides insights into how the polymer conformation clusters changed in the peptide storage locations compared to the average abundance throughout the whole NP. Then, the contacts between the peptides and the polymers in these environments were calculated. A contact was considered if the center of mass of a peptide was within 6.5 Å of any polymer bead. This was calculated at every time step. Subsequently, the cluster labels of each polymer in contact with the peptide were identified, and the number of contacts with each cluster was counted and averaged over the peptides in the different environments and over time. Finally, the percentage of contacts with each cluster (with respect to all clusters in each location) was calculated, and this was divided by the cluster presence percentage at each location, to determine whether contacts between the peptides and clusters were due to preferential interactions, or caused by the relative abundance of each polymer conformation.

Amino acid-wise contacts between cargo and polymer clusters. The contacts between the peptide residues and the polymers belonging to a specific cluster (depicted in Figure 5.9) were calculated in a similar way to other contact calculations in this Chapter. The center of mass of each amino acid was calculated, and if the distance between itself and any monomer of a polymer belonging to the selected cluster was below 6.5 Å, it was counted as a contact. This was repeated for all time steps, and the average was computed over peptides belonging to the same storage location and over time.

Confidence interval calculation. Error bars in the histograms of this chapter represent confidence intervals calculated at a 90% confidence level. They were calculated with the following formula:

$$CI = \bar{x} \pm z \frac{\sigma}{\sqrt{n}} \tag{5.8}$$

where $\bar{x}$ is the mean of the sample, z is set to 1.645 for a 90% CI, σ is the standard deviation and n is the sample size.

5.3 Results and Discussion

5.3.1 Structural equilibration of PEG-PLGA NP

To determine if the NP had reached equilibrium, the time evolution of the fraction of PLGA monomers within different regions of the NP core was calculated, as described in Section 5.2.2. Figure 5.1 (a) shows that these quantities become stationary at approximately 0.6 μs, implying that at that time, the internal structure of the NP has reached equilibrium. The time frames on which the analysis is calculated throughout this Chapter is shaded in pink in Figure 5.1 (a), since these are the stationary frames of the simulation. Figure 5.1 (b) shows the radius of of the NP (and its core alone) over time; these values plateau earlier (after approximately 0.3 μs) than the fraction of PLGA monomers. However, as Figure 5.1 (a) suggests, structural equilibrium is not achieved by 0.3 μs. This shows that the radius of gyration is not a sufficient parameter to determine structural equilibrium, as it stabilises much faster. Instead, the fraction of monomers within the core is a better indicator. This finding agrees with previous studies conducted in our group [147].

The average radius of gyration of the NP at equilibrium (9.93 ± 0.06 nm) is in close correspondence to the experimentally measured radius (10 nm) for this formulation [2]. Since all the simulation components were the same as the experimental ones, and the size and peptide loading capacity obtained from the simulation were the same as in experiments, it can be assumed that this NP simulation is an accurate computational representation of the experimental formulation.

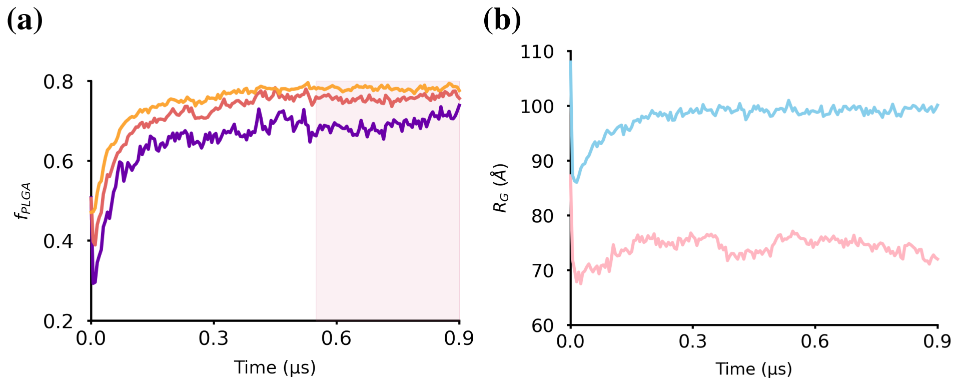


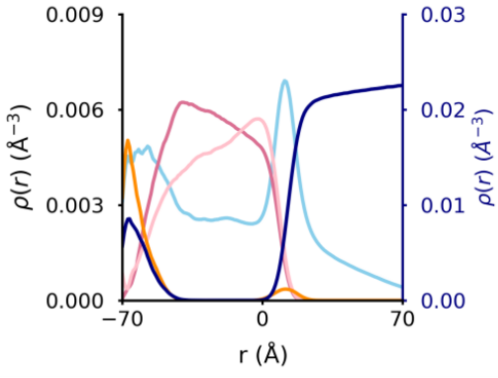
Figure 5.1: Micelle structural equilibration analysis. (a) Fraction of PLGA monomers in the core as a function of time. Purple is a distance of $r < 30$ Å, red $r < 40$ Å, and orange $r < 50$ Å from the micelle center of mass. The area shaded in pink denotes the time from which the NP is determined to have reached structural equilibration. (b) Radius of gyration over time of the whole NP (blue) and core of the NP (pink).

5.3.2 Distribution of NP components

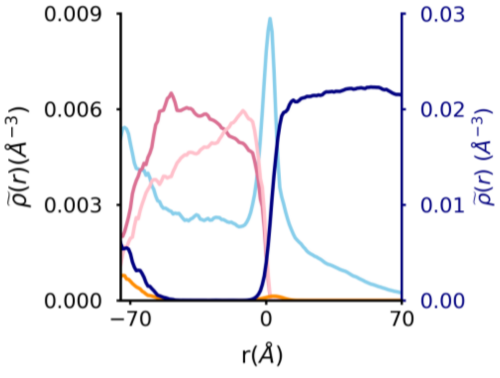
To understand the distribution of the various monomers and peptides within the NP, the average radial density of the NP components was calculated, as well as the analogous intrinsic core-shell interface (ICSI) method for comparison [147]. Figure 5.2 (a) shows the average radial density of components, while Figure 5.2 (b) illustrates the average intrinsic density. Given the NP is approximately spherical, both methods yield very similar results. From Figure 5.2 (c), which displays the percentage of polymer species within the core, it becomes evident that the NP core primarily consists of LA and GA blocks.

Surprisingly, a small nucleus of water forms near the center of the NP core during the self-assembly process in this unbiased CG MD simulation, as indicated in Figure 5.2 (a) and (b) by a peak in the water density curve (dark blue) centered at $r \approx -70$ Å. In this same region, there are also peaks in the density of both PEG and EEK. Therefore, there are EEK peptides encapsulated deeper into the NP core, and they are found at the interface of the small nucleus of water, as highlighted in the snapshot in Figure 5.4 (d). To quantify the amount of each molecule type in the core of the NP, Figure 5.2 (c) shows the percentage of CG beads of the polymer species, peptides and water within the core. PLGA blocks account for 70% of the total number of beads within the hydrophobic core while PEO is only around 20%.

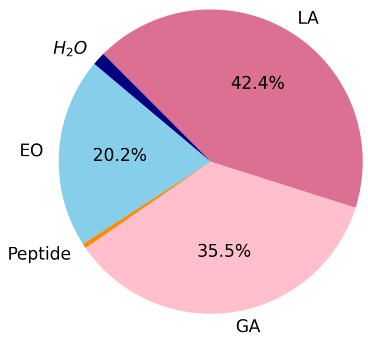
(a)



(b)



(c)



(d)

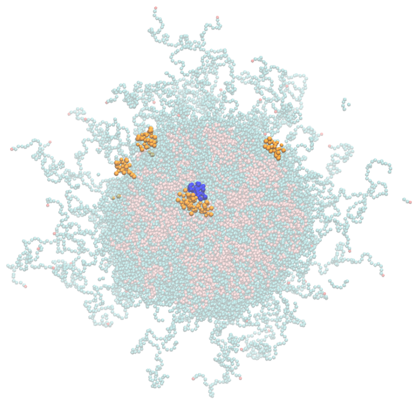


Figure 5.2: Internal composition of the PLGA-PEG nanoformulation. (a) Spherical and (b) intrinsic densities of various components of the polymeric NP and its aqueous environment, where glycolic acid (GA) is shown in light pink, lactic acid (LA) in fucsia, EO in light blue, EEK in orange and water in navy blue. (c) Percentage of the NP core made up by each polymer block, EEK ('peptide') and water. EEK peptides account for 0.5% of the beads in the core. (d) Snapshot of the cross section of the polymeric NP loaded with EEK peptides, where PLGA is shown in pink, PEG in light blue, peptide in orange and water in navy blue. The two peptides located in the inner core can be clearly seen close to the centre of the NP core, next to the water nucleus.

The water beads in the core only make up to 1.4% of the total beads and the peptides just 0.5%, as only two peptides are encapsulated deeper into the core as is shown later. Furthermore, the EO density peak at $r \approx 5$ Å (and sudden decrease in the density of the LA and GA monomers) denotes the boundary between the NP core and its hydrophilic PEG-based corona. There is also a peak in the EEK density at the core-shell interface at $r \approx 0$ Å), showing that peptides are located at the core-corona interface (as well as in the NP core).

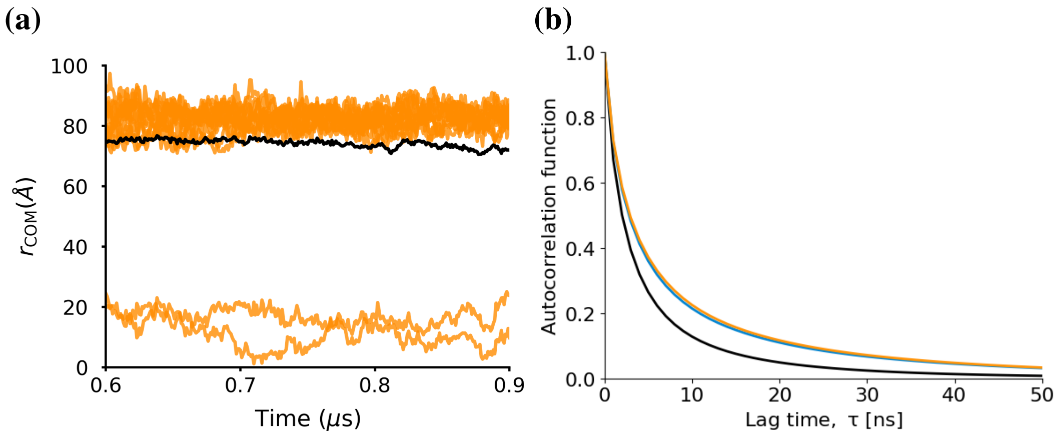


Figure 5.3: Peptide local environment. (a) Distance from all peptides to the COM of the nanoparticle over time. Orange lines denote each of the peptides and the black line is the R_g of the core. It is clear that there are two storage locations, one very close to the center, populated by two peptides, and another one at the core-shell interface with the remaining peptides. (b) The autocorrelation functions show the change in local environment of the peptides (average of peptides at the core-corona interface are shown in black and the two peptides found in the micelle core are shown in blue and orange). The peptide local environment is defined as all polymer beads found within 7.5 Å of any bead that is part of a specific peptide.

Additionally, to gain a deeper understanding of the encapsulation of the EEK peptides within the NP, the next step was to study the storage locations of the peptides. For this purpose, Figure 5.3 (a) shows the distance between the center of mass (COM) of each peptide and the COM of the NP over time. This analysis clearly reveals that two EEK molecules are encapsulated within the core of the NP, while the remaining EEK molecules are found at the core-shell interface. No significant transport of EEK within the NP is observed after the NP self-assembly and structural equilibration phases are complete. Furthermore, in order to understand if the peptides always stayed bounded to the same polymer beads, or if they moved dynamically, independently of the polymers, while maintaining an approximately constant radial distance from the COM of the NP, the autocorrelation function (ACF) of the peptides was calculated. This function allows the study of the time-evolution of each peptide's local environment (defining the instantaneous local environment as a list of polymer beads found within a cutoff distance of the peptide). The ACF results are presented in Figure 5.3 (b). Here, the black lines are the peptides at the

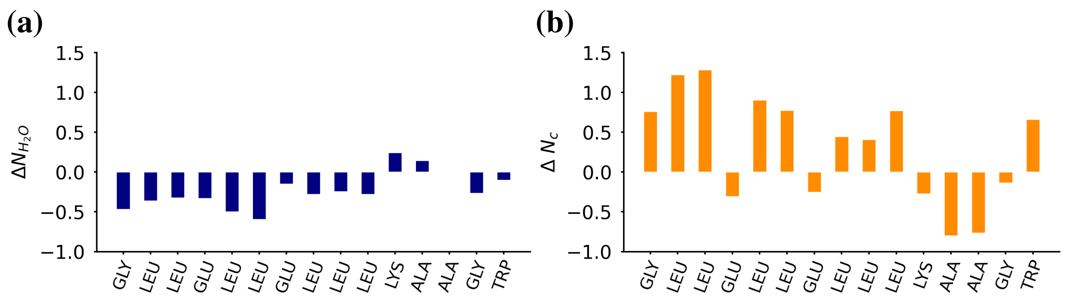


Figure 5.4: Cargo contacts and hydration difference between peptide storage locations. (a) Difference between the average water-peptide contacts (hydration) per amino acid between the peptides within the core and at the core-corona interface. (b) Difference between the average polymer-peptide contacts per amino acid between the peptides within the core and at the core-corona interface. A positive value corresponds to more contacts between a specific EEK residue and either a polymer bead or water at the center of the core than at the core-corona interface, and vice versa for a negative value.

core-shell interface and orange and green are the two peptides encapsulated into the core. It is clear that peptides in both locations readily exchange between different polymer beads, since the correlation rapidly decays. This indicates that the peptides are not static within the NP. Interestingly, the local environment of the two peptides in the core of the NP changes more slowly than those peptides at the core-corona interface, showing they are less dynamic. This is expected, as they are in contact with the water, limiting their movement to where the small amount of water trapped in the core is located. Also, the peptides located in the core are in a more compact region of the NP than the core-shell peptides, further constraining their movement.

5.3.3 Peptide polymer interactions

To determine the mechanisms of peptide encapsulation in the two different locations within the NP, the interactions of the EEK molecules with the polymer blocks and water were investigated. To compare between the two storage locations, the peptide-polymer and peptide-water contacts were computed for both locations. The difference in these contact values between both storage locations was computed and is displayed in Figure 5.4. Figure 5.4(a) shows that EEK peptides inside the core are generally less hydrated than the peptides on the surface. This is expected since they have less water available to interact with, as they can only interact with the small amount of water close to the NP core center. These EEK-water interactions occur because the polymer blocks do not completely shield EEK from water, in agreement with previous computational studies of peptide solubilization by polymers [93]. On the other hand, Figure 5.4(b) shows the difference in contacts between EEK and all polymer beads at the core center and the core-corona interface. In the core center, most peptide residues exhibit a greater number of contacts with polymers than those peptides at the core-shell interface. This is expected, since peptides in the core are completely surrounded by polymers, having a greater polymer surface to interact with. ALA and LYS residues of core residing peptides have the fewest polymer contacts but are the most heavily hydrated. So for peptides in the core center, these residues are most commonly found to be in contact with the small water nucleus in the NP core.

To quantify the frequency with which peptides in the core, and more specifically certain amino acids within them, come in contact with the small water nucleus, the residence time of interactions between water beads and both the entire peptide and individual amino acids (LYS and ALAs) were calculated for both peptides situated in the core center. The results are displayed in Table 5.2. This table shows that for the whole of the equilibrated simulation, both peptides located at the core have at least one bead always in contact with the water trapped inside the core. Therefore, these peptides are always located near the small water nucleus. Additionally, the LYS amino acid has a higher residence time than the ALAs, meaning that the LYS residue will be found more often in contact with the water molecules trapped in the core.

Knowledge of the storage location allows to study the differences in the interactions between the peptides and polymers in both locations. Table 5.3 displays the time-averaged enrichment of contacts between the peptides in the two different locations and each of the polymer species. The enrichment of contacts is used to measure the tendency of molecules to interact with each other, taking into account the molecules abundances. From Table 5.3, it is clear that peptides have more interactions with PEO in both locations. Interestingly, the peptides located in the

Peptide	Whole	LYS	ALA	ALA
1	100% CI[100%, 100%]	24.63% CI[0.04%, 54.35%]	2.15% CI[1.6%, 2.67%]	2.08% CI[1.48%, 2.68%]
2	100% CI[100%, 100%]	16.12% CI[0.89%, 31.65%]	2.39% CI[1.50%, 3.27%]	2.57% CI[1.84%, 3.31%]

Table 5.2: Residence time of interactions between peptides and water encapsulated in the core. Values reported are the average residence time over the trajectory and its corresponding 90% CI in square brackets. Further information about this method can be found in the Section 5.2.2. The residence times are reported as percentages of the total time of the simulation analysis. The residence times of the peptides are 100% which means that the peptides always have at least one atom in contact with the water, proving that the peptides are always in contact with the small water nucleus in the core. The amino acids selected for the residence time calculation with the water are the ones that are most hydrated with respect to the amino acids of the core-shell peptides counterparts. In this case, the confidence intervals are larger, because these interactions normally consist of either very long interactions or very transient interactions, which increases the standard deviation of the data set.

Location	LA	GA	EO
Core center	0.33 CI[0.32, 0.34]	0.37 CI[0.36, 0.39]	2.19 CI[2.17, 2.21]
Core-shell interface	0.48 CI[0.47, 0.48]	0.30 CI[0.29, 0.30]	1.59 CI[1.58, 1.59]

Table 5.3: Contact enrichment between peptides and polymer species. Contact enrichment quantifies the extent to which one molecule exhibits a higher tendency to interact with another molecule, relative to their respective abundances. This table shows the contact enrichment between polymer species and peptides, differentiating between the two storage location: core and core-shell interface. In this case, the enrichment takes into account the polymer species and peptide population in the areas of cargo encapsulation. A value greater than 1 means a tendency to interact with that species, 1 indicates no preference in interactions, and a value less than 1 is a tendency to not interact with that polymer species. Values reported are the contact enrichment average over time with their corresponding 90% CI in square brackets.

core exhibit a much higher tendency to interact with EO monomers than the peptides located at the core-shell interface. This is attributed to the amphiphilicity of EEK, resulting in its close proximity to the trapped water in the NP core. EO is hydrophilic, and there is a peak in EO density surrounding the water in the core (Figure 5.2 (a)); EO is found around the water trapped in the core, where the peptides are also located. This results in a higher number of interactions between EEK and EO within the NP core.

5.3.4 Unsupervised learning reveals location-specific polymer conformations

To investigate the specific conformations that the polymers adopt within the NP, a two-step unsupervised machine learning protocol consisting of dimensionality reduction with UMAP and clustering in the subsequent embedding using HDBSCAN was applied [147, 150, 151]. The goal of this protocol was to determine whether polymers within the NP adopt conformations with common features, similar enough for them to be clustered by UMAP, or if their conformations are very random and distinct from each other. Since the polymers under investigation are amphiphilic diblock polymers, their spatial conformation is defined by the end-to-end distance of the PLGA and PEO block. Therefore, these distances per polymer and over all equilibrated time steps were the input for UMAP. These two distances were enough to understand how these polymers fold and to cluster the conformations, since they capture the behavior of the hydrophobic and hydrophilic blocks of the polymers respectively. Figure 5.5 (a) shows the outputted UMAP embedded space clustered by HDBSCAN with only 0.41% of molecular conformations not clustered. There are four distinct clusters, each grouping polymers with similar spatial conformations. Moreover, Figure 5.5 (b) displays the prevalence of each of the cluster throughout the equilibrated trajectory and a snapshot of a polymer belonging to each cluster. Cluster 1 is the most common conformation polymers adopt and cluster 4 the least common, but it still represents around 15% of the total conformations. Furthermore, Figures 5.5 (c)-(d) illustrate the average block distances of each cluster. This also shows that the conformations are properly clustered, since all histograms (and therefore clusters) are sufficiently distinct from each other. From these histograms, it is clear that Cluster 1 (Figure 5.5 (c)) has a collapsed PEO block and an extended PLGA block. Conformations within cluster 2 (Figure 5.5 (d)) possess an extended PEO block and collapsed PLGA block, while in cluster 3 (Figure 5.5 (e)) both blocks are collapsed. Finally, conformations within cluster 4 ((Figure 5.5 (f))) has a particularly extended PEO block, as well as an extended PLGA block. Each cluster therefore represents different polymer conformations, which can be readily

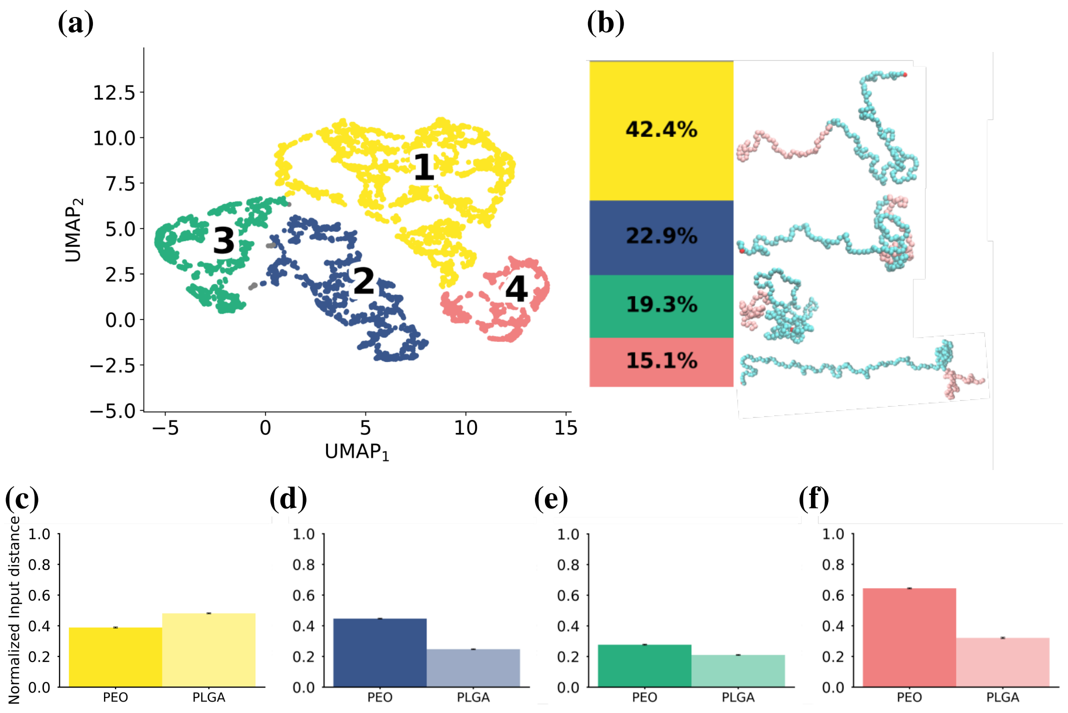
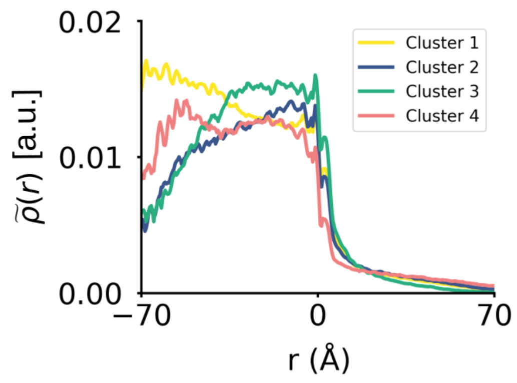


Figure 5.5: Unsupervised learning reveals specific polymer conformations. (a) UMAP embedded space clustered by HDBSCAN of polymers. (b) Bar chart with the percentage of each conformational cluster within the NP. Next to each percentage there is a snapshot of a random polymer within each cluster. In the snapshots, PLGA is shown in pink and PEG in cyan. Snapshots of the polymers are not to scale. Histograms of the average distances of each cluster: (c) cluster 1, (d) cluster 2, (e) cluster 3 and (f) cluster 4. Note that the error bars show the 90% CI.

understood in a physical sense.

The ICS density calculation method was used to calculate the intrinsic density of each conformational cluster, to study how these conformations were distributed throughout the NP. Figure 5.6 (a) reveals the spatial distribution of each conformation within the NP, with respect to the core-corona interface, which is set as 0 Å in the x axis of Figure 5.6 (a). Figure 5.6 (a) has been normalized to account for cluster population, as the density values of more populated clusters are higher due to the larger number of particles. This makes the comparison between density profiles on the same plot harder (cluster density values would be very far apart). This normalization is applicable, since the density values themselves are not of interest, but the profile of the density, which provides information of where the polymer cluster is more likely to be found within the NP. Note that cluster 1, with its extended

(a)



(b)

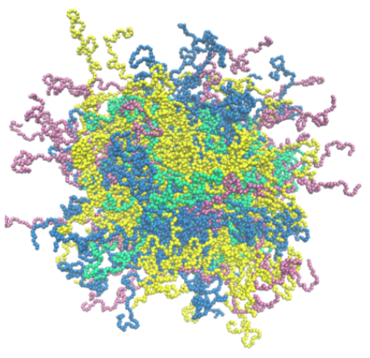


Figure 5.6: Polymer conformations are location specific. (a) Normalised intrinsic density profile of the various clusters within the NP. The normalization takes into account the cluster population. Normalized per polymer cluster. (b) Snapshot of NP with the polymers colored in correspondence to their cluster. The colors applied in the snapshot are the same as those used for the different clusters in (a). NP snapshot is not to scale.

PLGA block, exhibits high density in the center of the NP core. Cluster 4, which conversely has a particularly extended PEG block is surprisingly also found at high density in the NP core center: the small nucleus of water at the NP promotes the unexpected location of this conformational state. The PEO block extends to cover as much of the water surface as possible. Conformational clusters with collapsed PLGA block (cluster 2 and 3) are more commonly located closer to the core-shell interface. This is expected, as PLGA will be trying to minimise its contact with the aqueous environment. Therefore, polymers take specific conformations as a result of their location within the NP, similarly to previously reported atomistic-level studies by our group, where we have shown that block copolymers adopt location-specific conformations within micelles of various polymer chemistries and topologies [69, 147], like in Chapter 3.

5.3.5 NP microenvironments allow for the encapsulation of EEK in different locations

Having established this link between location and polymer conformation, the investigation focused on how EEK interacts with the different conformational states of the polymers in both the core center and at the core-corona interface. The goal was to understand if polymer conformations play an important role in cargo encapsulation. That is to say, whether specific polymer clusters are the ones that allow for cargo encapsulation, or if EEK peptides interact randomly with all polymer conformations. For this purpose, the number of contacts between each EEK peptide and each of the different polymer conformations were calculated, taking into account the relative abundance of polymer conformations in the two solubilization locations. This is essential in order to determine whether any preferential interactions exist between specific polymer conformations and EEK, rather than just local enrichment of a specific conformation in certain local environments of the NP.

Therefore, the first step was to calculate the relative polymer cluster abundance in both peptide storage locations. The theoretical details on how the cluster presence percentage in each storage locations was calculated can be found in Section 5.2.2. Figures 5.7 (a) and (c) show the abundance percentage of each cluster in the core and core-shell interface cargo storage locations respectively. To understand how these values change with respect to the polymer cluster abundance averaged throughout the whole NP (showed in Figure 5.5 (b)), the cluster presence enrichment of the polymer clusters at these locations was calculated with respect to the overall cluster presence. Figures 5.7 (b) and (d) show the polymer cluster enrichment at the core and core-shell interface respectively. From Figures 5.7 (a)-(b) it is clear that cluster 1 is the most dominant cluster in the inner storage location with a high enrichment, meaning that its abundance there is higher than its average cluster abundance. On the other hand, Figures 5.7 (c)-(d) display that at the core-shell interface location, it is actually cluster 2 and 3 which have a larger enrichment, meaning that these clusters are present at a higher percentage in this region than their average value throughout the NP. These results agree with the intrinsic density of the clusters depicted in Figure 5.6 (a). It is interesting to note that this quantification of the polymer clusters shows that cluster 1, the most abundant cluster in the NP, really decreases its presence, by almost half, at the core-shell interface location. Furthermore, polymer cluster percentages are significantly different (comparing Figures 5.7 (a) and (c)) at the storage locations. Therefore, this NP clearly posses different polymer micro-environments (at the core and core-shell interface) that change with respect to the distance from the center.

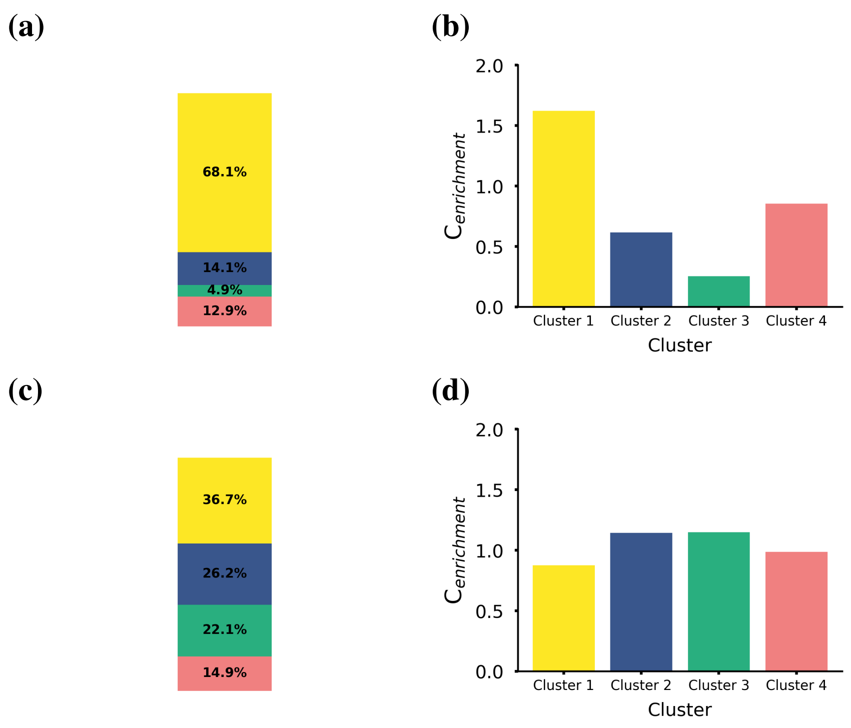


Figure 5.7: Polymer micro-environment of peptide storage locations. Cluster presence percentage of storage location at the (a) core and (c) core-shell interface. Cluster presence enrichment of polymer clusters at the (b) core and (d) core-shell location of peptide storage with respect to overall cluster presence. A value greater than 1 means that the cluster is enriched in that region of the NP, and a value smaller than 1 means the cluster is underrepresented in that area.

The contacts between the peptides and the polymer clusters were computed at both storage locations, taking into account their relative abundances, to determine if there were preferential interactions between the peptides and the polymer clusters at the different storage locations. For this purpose the fractional enrichment of the contacts (ε_i) was calculated. The fractional enrichment of contacts between the polymer conformations and EEK quantifies the tendency of EEK to interact with a specific polymer cluster, taking into account the abundance of that cluster in the region of the NP where the EEK molecule is located. ε_i was calculated as follows:

$$\varepsilon_i = \frac{n_i/n_{\text{total}}}{N_i/N_{\text{environment}}} \tag{5.9}$$

where n_i is the number of contacts between the peptides and the polymers belonging to conformational cluster i in its local environment; n_{total} is the total number of contacts between the peptides and all polymers in the local environment. N_i is the number of polymers in conformational cluster i that are found in the local environment and $N_{\text{environment}}$ is the total number of polymers in the local environment. Figure 5.8 shows the fractional contact enrichment for peptides located in both storage locations. From Figure 5.8 (a) it is evident that EEKs at the core-shell interface do not preferentially interact with a specific polymer conformation, as the differences among the clusters are relatively small. Conversely, Figure 5.8 (b) reveals that peptides located inside the NP core exhibit a clear preference to interact with a specific polymer environment within the NP hydrophobic core (cluster 4). This mechanism of encapsulation may be relevant for other amphiphilic drugs that are also encapsulated into the core of amphiphilic polymer-based NPs [197]. Polymer conformations featuring a collapsed hydrophobic block and extended hydrophilic segment that are found in the core of the NP, such as cluster 4, will facilitate the encapsulation of drugs near water molecules trapped in the core. This is achieved because the extended hydrophilic block maximizes its contact with the water nucleus and drug, while the collapsed hydrophobic block minimises contacts with the aqueous environment.

The next step to understand the encapsulation process was to analyse the monomers with which the peptides tend to interact. For this purpose, Figure 5.9 shows the normalized contact maps for peptides in the core and at the core-shell of the NP with their preferential polymer cluster, respectively. These contact maps display the contacts per amino acid of the peptide with all polymer monomers. It is important to note that EEK found in both locations reside at a polymer-water interface (Figure 5.2 (a)). Figure 5.9 (a) shows that peptides within the NP core primarily interact with the final monomers of the GA block and the first few monomers of the EO block of cluster 4 polymers. While interacting with EO, EEK is also in close proximity to the hydrophobic PLGA block (reflecting EEK's amphiphilicity). The polymers which make up cluster 4 are present near the trapped water inter-

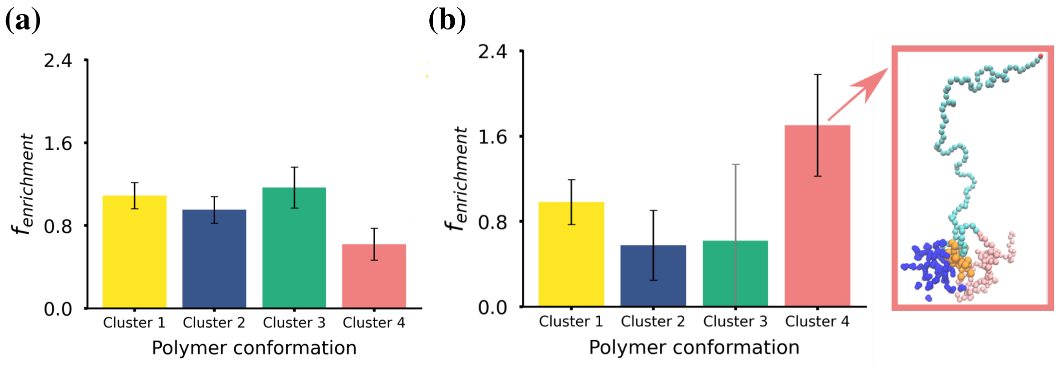
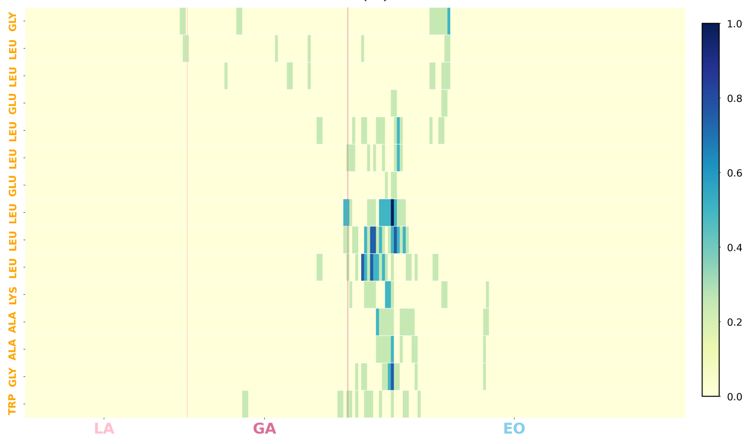


Figure 5.8: Cargo interactions with polymer conformational clusters. Average enrichment fraction of contacts between polymers in each of the different polymer clusters within the NP and (a) peptides located at the core-shell interface and (b) peptides captured inside the core. Note that all error bars show the 90% CI. The enrichment takes into account the relative cluster population, such that an enrichment value greater than 1 means the peptides have a tendency to interact with that polymer conformation, an enrichment value less than 1 suggests that there is no preferential interaction and a value of 1 means that the peptide and the polymers from that cluster interact randomly. In (b), there is a snapshot showing a peptide interacting with the most preferential polymer cluster, cluster 4. In this snapshot, PLGA is coloured pink, PEG is light blue, the peptide is orange and water is navy blue. Snapshot is not to scale. The error bar of Cluster 3 in (b) is colored in gray due to poor statistics for this cluster, as there are not many interaction between the peptides captured inside the core and this cluster.

face: its extended EO block shields the hydrophobic PLGA polymer blocks from the water encapsulated in the core. Conversely, EEKs at the core-corona interface generally interacts mainly with all EO monomers without showing a clear preference with a particular region of the hydrophilic block and have very few contacts with the hydrophobic monomers, as shown in Figure 5.9 (b). This is probably due to the fact that there is not a clear preferential interaction for peptides located at the core-shell interface with a specific polymer cluster. Therefore, these peptides just interact randomly with all EO monomers available, benefiting from interacting with polymer clusters 1 and 3 (the clusters with the highest preference), since they have a collapsed PEG block, increasing the available EO monomers to interact with.

(a)



(b)

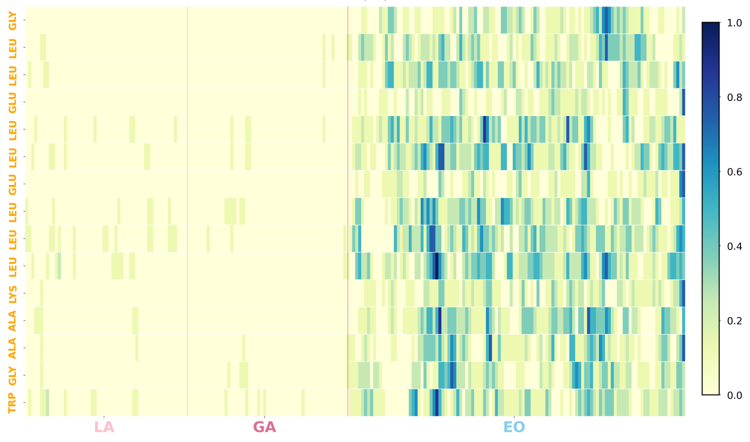


Figure 5.9: Normalized contacts between peptides and preferential polymer cluster. Average normalized contacts over time between peptides at (a) core storage locations and polymers belonging to cluster 4 and (b) peptides located at the core-shell interface and cluster 3 polymers.

5.4 Conclusion

The results presented here provide molecular-scale mechanistic insights into the formation of experimentally validated PEG-PLGA NPs and the encapsulation of their cargo. The CG MD simulation in this Chapter, results in a NP with the same loading capacity of EEK peptides and R_G as observed in experiments [2]. The peptides are primarily located at the interface of the NP hydrophobic core (formed by the PLGA blocks of the polymers) and NP corona (comprised of PEG blocks). Interestingly, two peptides were encapsulated within the core of the NP, and they are residing at the interface of a small nucleus of water trapped during the NP formation. This agrees with experimental studies where they have found drugs encapsulated into the core and at surface of PLGA based NPs [33].

Having identified peptides in two distinct locations within the NP, the conformational changes that the polymers undergo at different locations within the NP were studied. Four distinct conformational clusters were identified, with two of them being primarily found within the core of the NP and the other two at the interface of the NP core and its corona. Within the core of the NP, one of the most prevalent conformations (cluster 1) has an extended PLGA block that extends through the core of the NP such that its semi-collapsed PEG block can sit at the core-corona interface. The other conformation (cluster 4) found primarily in the core of the NP is generally found near the nucleus of water that has formed within the core. This conformation has an extended PEG block that extends through the water and a collapsed PLGA block that sits at the interface of the water and polymer. In the other storage location, at the core-shell interface, both of the most prevalent conformations have a collapsed PLGA block which sits at the interface of the core and corona. Regarding the PEO block, one of the conformations has an extended PEG block (cluster 2) and the other one (cluster 3) a collapsed PEG block. Previous studies carried out by our group investigated the dynamics of PEG blocks in amphiphilic block polymers at the interface of polymeric NPs, as they transition through states where the PEG block is contracted, extended and in an intermediate [147] state. Thus the two conformations of these PEG-PLGA block copolymers identified at the core-corona interface, which are present in almost equal measures, are consistent with the contracted and extended states of the PEG blocks.

Finally, it was investigated if the peptides in the different locations within the NP interact preferentially with the polymers found in those locations, or if these interactions are random. Peptides located in the NP core, preferentially interact with the polymers in cluster 4, which are predominantly found near the small water nucleus. This is attributed to the amphiphilic nature of EEK; being encapsulated close to the water nucleus allows the most hydrophilic peptide residues to remain hydrated. Therefore, polymers in cluster 4 are the only ones that enable EEK to be in contact with the water, while at the same time allowing EEK non-polar amino acids to be in contact with the (collapsed) hydrophobic block of the polymers. Therefore, polymer clusters with a collapsed hydrophobic region and extended hydrophilic block found within the hydrophobic core of NPs, promote the deeper encapsulation of amphiphilic drugs into the NP core. On the other hand, at the core-corona interface, peptides interact slightly more with polymers in clusters 1 and 3. These clusters feature a semi-extended PEG block, and thus can potentially shield some of the hydrophobic parts of the peptide which may otherwise be hydrated by the surrounding water. Also, the semi-extended PEG block increases the available EO monomers for the peptides to interact with. Given the preference of the peptides to interact with EO monomers compared to the other polymer species, semi-extended PEG conformations play an important role for the encapsulation of peptides at the core-shell interface. However, it is important to note, that the peptides at this location do not have a particularly strong preferential interaction with any of the clusters.

In conclusion, this Chapter shows that the cargo loading within polymeric NPs is dependent not only on the NP internal structure, but also on the conformations that polymers adopt at different regions within the NP. These distinct conformations create different chemical environments throughout the NP, affecting the ability of drug encapsulation within these environments. These new results, combined with the results from Chapter 3, which revealed that polymer topology affects the ability of polymers to take location-specific conformations within polymeric NPs [69], demonstrate that controlling the local environments within polymeric NPs is a key aspect to consider in the rational design of drug loaded NPs. Controlling these environments will enable precise adjustments to the locations of encapsulated drugs.

5.5 Future work

The methodology presented in this chapter for studying drug encapsulation by polymeric NP holds promise for broader applications in drug-NP systems. Future investigations could leverage the unsupervised machine learning technique introduced here to explore whether other drugs also exhibit specific encapsulation preferences within distinct polymer microenvironments. This research could unveil trends and correlations, such as the relationship between the physicochemical characteristics (e.g. hydrophobicity) of a drug and its propensity for encapsulation within particular polymer microenvironments.

Furthermore, for the purpose of NP rational design, it would be of interest to examine whether the polymer clusters identified in this specific PEG-PLGA drug-loaded NP are dependent on the number of polymers present. Investigating whether these polymer conformations occur consistently in NPs made of this polymer, regardless of size, or if there is a minimum polymer count required to achieve these specific polymer conformational clusters could provide valuable insights. This research could be conducted in conjunction with an analysis of NP storage locations to optimize the balance between NP size and drug loading capacity. Striking the right balance is essential for overcoming biological barriers that require small NP sizes while ensuring the successful delivery of an effective drug dose.

Chapter 6

PEG-PLGA active-compound loaded nanoparticles selectively target cancer cells due to specific polymer-lipid interactions

Drug-loaded polyethylene glycol poly(lactic-co-glycolic) acid (PEG-PLGA) nanoparticles (NPs) have been extensively studied as cancer therapeutics. One of the reasons is that they have demonstrated selective targeting of cancer cells *in vitro* and in animal models [2]. However, the mechanisms driving this targeting remain unclear. Conventionally, it has been believed that NPs target cancer cells through the Enhanced Permeation and Retention effect (EPR), which postulates that NPs accumulate more in tumorous tissue due to easier access via enlarged tight junctions between endothelial cells. Nevertheless, recent research suggests that the EPR effect may not be the primary cause of NP selective targeting [35].

In this Chapter, coarse-grain (CG) molecular dynamics (MD) simulations were employed to study the interactions between an experimentally validated drug-loaded PEG-PLGA NP [2] and two model membranes with complex lipid compositions, representing cancer (glioma) and healthy (oligodendroglial) membranes. These simulations aimed to identify molecular behavior differences in the NP-membrane interactions that may underlie NP selectivity. To achieve this, the changes in the physicochemical characteristics of the NPs and membranes induced by the NP-membrane interactions were quantified and compared across simulations. The simulations revealed that the NP interacting with the cancer membrane undergoes more significant changes in size and shape than when it interacts with the healthy membrane. Interestingly, these changes made the NP interacting with the cancer membrane adopt a size and shape that favour NP transcytosis through membranes. Furthermore, the NP disrupted the cancer membrane to a greater extend than the healthy membrane, as evidenced by changes in membrane thickness. This shows that NP selectivity can be studied *in silico* with membranes containing different lipid composition. Additionally, the primary factor contributing to the greater changes to the NP and the membrane in the cancer simulation is the preferential interaction of PEO polymers with a specific lipid species, which is present in higher percentages in cancer membranes. Therefore, preferential polymer-lipid interactions emerge as a key parameter in understanding NP selectivity towards cancer cells. Furthermore, this chapter introduces parameters for *in silico* assessment of NP selectivity that can be applied to the study of other NP-membrane systems.

6.1 Introduction

Drug-loaded polyethylene glycol poly(lactic-co-glycolic) acid (PEG-PLGA) nanoparticles (NPs) show promise as cancer therapeutics [198]. These NPs possess several key characteristics that make them potential anti-cancer drugs, including controlled drug delivery, biodegradability of their components, enhanced drug solubility, prolonged circulation times, reduced systemic toxicity, and the ability to precisely target cancer cells while sparing healthy cells [17, 19, 25, 26, 32, 198]. Furthermore, PEG-PLGA NPs have been shown to successfully deliver anti-cancer drugs *in vitro* [2, 175] and in animal models [33, 176]. Despite these characteristics and promising results *in vitro*, there is still no FDA approved PEG-PLGA NP-based treatment for cancer (or other diseases). One of the main reasons is that the mechanisms of action behind the cancer cell selectivity of these NPs are not well understood [45, 48].

For an extended period, the prevailing belief held that NPs target cancer cells through the passive targeting strategy of the Enhanced Permeation and Retention (EPR) effect [34], also known as the ’leaky vasculature’ of tumours. The EPR effect centers around the presence of abnormally large gaps between endothelial cells in cancer tissue. These enlarged gaps result in an increased vascular permeability, enabling NPs to access and accumulate within tumorous tissues more readily than in healthy tissues [199]. Additionally, the EPR effect suggests that the increased accumulation of NPs in cancer tissues stems from the faulty lympathic system within the tumour, preventing NPs from exiting the tumorous tissue [200]. However, recent research challenges the notion that the EPR effect is the primary mechanism through which NPs effectively target cancer cells [34, 35]. For example, Sindhwani *et al.* [35] demonstrated that 97% of NPs enter tumorous tissue via active targeting of endothelial cells, such as binding of NPs to endothelial cells or transcellular transport [35]. In this study, researchers are not able to pinpoint which active targeting mechanisms are more critical, but they highlight the importance on understanding NP-tumour endothelial cells interactions. Furthermore, Nguyen *et al.* [201] showed that NPs can actually exit solid tumors, so they do not stay trapped within the tumour matrix as hypothesized by the EPR effect. According to Nguyen’s study, NPs are able to exit through the lympathics and return to the blood system, allowing them to recirculate and interact with the cancer cells again [201]. This allows NP to deliver the therapeutics that they may not have been able to deliver previously, which increases the efficacy of the treatment and lowers the required therapeutic dose. Overall, it is evident that there is an urgent need to further study the interactions between cancer cells and NPs to gain a deeper understanding of why and how NPs target cancer cells over healthy cells.

The distinction in lipid composition between cancer and healthy cells underscores the power of lipidomics, the quantification of lipids in cells, tissues, fluids, or organisms, as a robust cancer biomarker [39, 202, 203, 204]. Numerous studies have focused on understanding the several mechanisms behind NP-loaded drug delivery to cancer cells, including the forces that drive NP-cancer membrane interactions [35, 45, 49, 50, 51, 52], changes in PEG-PLGA NPs' properties [49, 53] affecting cancer cell specificity [37], or the mechanisms governing cargo storage [54] and delivery into cells [55]. However, there is a notable gap in research on interactions between the polymeric components of experimentally-validated NPs and lipid species of cancer and healthy cells. These studies would be key to understand if specific polymers have preferential interactions with lipids that are present in higher percentages in cancer cells. Since the EPR effect may not be the main cause why NPs target cancer cells, the NP components will influence its selectivity, and lipid-polymer interactions may play a key role in this [205]. These interactions are hard to study experimentally, as they are highly dynamic. Nevertheless, a deep understanding of polymer-lipid interactions is possible at the molecular level with molecular dynamics (MD) simulations [49, 50, 56]. For example, previous studies have shown the preference for PEG to interact with certain lipid types such as 1-palmitoyl-2-oleoyl-sn-glycero-3-phospho-(1'-rac-glycerol) (POPG) which are present in higher percentages in some types of cancer membranes [44] than in healthy cell membranes. Also, there have been MD simulation studies of anti-cancer NP and membrane interactions, but most of these studies use very simple membrane models, with only one or two lipids species [206, 207], so the effect of cancer specific lipid composition is lost in these simulations.

In this Chapter, CG MD were employed to study the mechanisms of action underlying the cancer selectivity of an experimentally validated PEG-PLGA NP loaded with an anticancer peptide, named EEK [2]. This NP, introduced in Chapter 5, has been experimentally confirmed to exhibit high selectivity against triple negative breast cancer cells. In these simulations, the NP had the same polymer number and ratio, diameter and cargo loading capacity as in experiments. Two processes were simulated, the first one; the interactions of the NP with a model glioma (cancer) membrane and the second one, the interactions of the NP with a model oligodendroglia (healthy) membrane. Both membranes consist of the 7 major lipid species of each cell line, which are of course, different between them. To the author's best knowledge, this is the most detailed computational study with complex membranes of experimentally validated PEG-PLGA NPs. From these simulations, molecular-level detail of the differences in the interaction of this NP and EEK cargo with cancer and healthy cells were obtained. Furthermore, these MD simulations enable the quantification of several physicochemical changes that NPs undergo when interacting with membranes, facilitating comparison across simulations that reveal how NPs display distinct behaviour at the molecular level depending on the membrane with which they interact. Finally, these simulations also demonstrate that specific polymer-lipid interactions may be a key factor in NP selectivity towards cancer cells. It is important to note that the simulation approach proposed in this Chapter can be used to study the same processes for different polymeric NPs and cell lines. The molecular study of the proposed polymer-lipid interaction parameter is key for the rational design of NPs, enabling the tuning and synthesis of NPs with enhanced cancer specificity.

6.2 Methods

6.2.1 Simulation details

All simulations were performed using GROMACS [188] versions 2019.2 and 2020.3 *(www.gromacs.org)*. Coarse-grained (CG) molecular dynamics (MD) simulations were employed to investigate the interactions of a NP formed by block copolymers containing polyethylene glycol (PEG) and poly(lactic-co-glycolic acid) (PLGA) loaded with the antimicrobial peptide named EEK with two different lipid membranes: a model glioma lipid bilayer and a model bovine oligodendroglia lipid bilayer.

The MARTINI CG force field [189] was used. In particular, the martini22p forcefield, which includes: polar amino acids, Martini 2.0 lipids and polarizable water, as this model represents more accurately the polarized nature of water [135]. This forcefield has been shown to adequately simulate polymers, especially PEG [190]. Also, the non-standard MARTINI topology which includes PEO [191] was used, as this is one of the NP polymeric components.

The NP used for the simulations is the same as in Chapter 5. Therefore, the methods describing the formation of this NP can be seen in Section 5.2.1. The NP was simulated with two different model membranes: (i) a healthy cell membrane, which was modelled using the lipid composition of Bovine oligodendroglia cells [208] and (ii) a cancer cell membrane, for which the composition of a glioma cell was used [208, 209]. All bilayers were created using CHARMM-GUI Martini Bilayer Maker [119, 192, 210] with the martini22p forcefield. All membranes have approximately 5000 lipids in total. The lipid compositions of the different bilayers are illustrated in Table 6.1.

	Cancer model		**Healthy model**	
Lipid type	UL	LL	UL	LL
POPC	44%	34%	50%	40%
POPE	6%	17%	20%	25%
DPSM	39%	0%	12%	0%
POPS	0%	11%	0%	12%
POP2	0%	6%	0%	2%
POPI	0%	21%	0%	3%
CHOL	11%	11%	18%	18%

Table 6.1: Membrane lipid composition. Table showing the lipid composition in the upper (UL) and lower leaflets (LL) of the cancer and healthy model membranes respectively.

In both simulations, the center of mass of the NP was initially placed approximately 20 nm above the phospholipid headgroups of the upper leaflet of the membrane bilayer. The solvant in every simulation was water, and Na^+ and Cl^- ions were added so that the system had neutral charge and a 0.15 M salt concentration. In order to equilibrate all systems, the first step was a steepest descent minimization, followed by six equilibration steps. First, a 1 ns NVT equilibration with a timestep of 2 fs with the whole system at 303.15 K. Then a 1 ns NVT simulation with a 5 fs time step with the water and ions at 363K and the rest of the system at 303.15 K, to let the water equilibrate faster around the system. Then, three NPT simulations of 1 ns each with timesteps of 1, 15 and 20 fs respectively. All NPT simulations used a velocity-rescale thermostat, with the temperature groups of the protein, polymers, membrane (which includes all lipids forming the bilayer) and ions and water as one group. All groups were at 303.15 K. The electrostatic interactions were computed using the Reaction-Field algorithm [152] with a coulomb cut-off of 11 Å and the van der Waals interactions had a cut-off of 10 Å. In the minimization and equilibration steps, the NP was subject to position restraints, to let the solvent and membrane equilibrate in the new system, as the NP had already run for a considerably long equilibration.

For the production run, the same parameters as in the NPT equilibration were used except for the barostat which is is Parrinello-Rahman [194] at 1 atm. These simulations ran for 5 μs, but they were capped at 2 μs, so all analysis showed in this Chapter is for the first 2 μs of the simulations. The simulation analysis was only performed until 2 μs due to two reasons, which are illustrated in Figure 6.1. First, from 2 μs, the membrane area started becoming very saturated with PEO polymer arms, since these were crossing the PBC. This also caused the NP to interact with itself. Figure 6.1 (a) shows the fraction of the x-y membrane area covered by the NP over time. It is clear that from 2 μs, the values start approaching 1, meaning that the number of PEO arms crossing the PBC and interacting with itself is significant. This phenomenon can be visually observed in Figure 6.1 (c). This Figure shows a top-view of the NP interacting with the cancer membrane. The PEO monomers inside the red circles, have just crossed the PBC. The second reason is that after 2 μs, probably due to the membrane saturation, the membranes started bending very quickly, in a non-physical way, particularly the healthy membrane. This membrane bending led to a significant reduction in size of the simulation box in the x-y plane, creating a non-physical situation. This is captured in Figure 6.1 (b) which shows the 2-d simulation area over time. From here, it is clear that the area decreases significantly, specially for the healthy membrane. Therefore, the analysis of the simulations were capped at 2 μs, just before nonphysical bending and reduction in simulation x-y area, and self-interactions of the NP appeared. Also, note that the different membranes had been previously equilibrated for 500 ns.

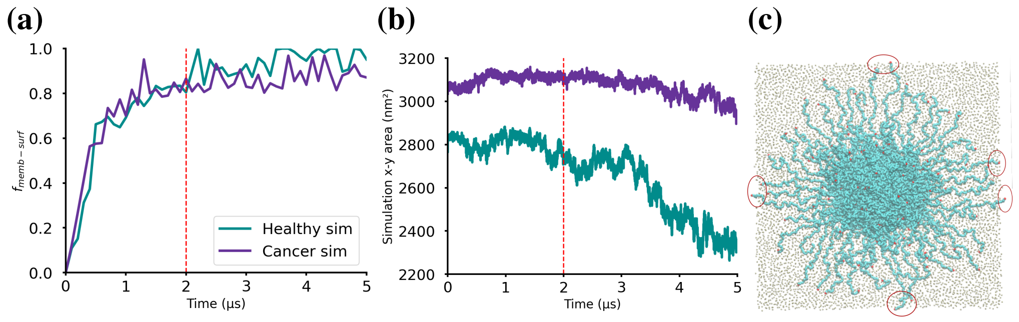


Figure 6.1: Simulations were capped at 2 µs. (a) Fraction of membrane surface covered by the NP in the cancer (purple) and healthy (green) simulations. (b) Evolution of simulation box x-y surface over time for the cancer (purple) and healthy (green) simulations. (c) Top-view representation of the NP interacting with the cancer membrane at 2.2 µs. PEO beads are colored in cyan and phosphate groups of the membrane in dark green. The red circles encapsulate the PEO beads that have crossed the PBC. Representation is not to scale.

6.2.2 Analysis methods

The analysis of these simulations focused on: i) the physichochemical properties of the NP during its interaction with the different membranes, ii) membrane perturbation due to the NP, and iii) drug delivery efficiency. All analysis were performed using Python 3.7, with the Python packages: MDAnalysis [153, 211] and LiPyphilic [166], as well as the membrane thickness analysis software FATSLiM [212] and GROMACS. Plots were produced with Matplotlib [195] and simulation visualizations with VMD [196].

NP-membrane distance. The distance between the NP and the membrane is computed to discern which membrane the NP interacts with more rapidly and within which membrane it embeds itself more deeply, indicative of preferential interactions. The distance from the NP center of mass (COM) to the membrane surface was measured using MDAnalysis. The distance was taken from the z coordinate of the center of mass of the NP, determined by the MDAnalysis function `center_of_mass()`, to the membrane surface. The membrane surface was defined by the average *z*-coordinate of the PO4 (phosphate headgroup) beads of the upper leaflet of the lipid bilayer. As stated above, to calculate the NP-membrane distance, the NP COM was picked instead of the center of geometry. The COM of an object is defined as the point at which the entire mass of the object can be assumed to be concentrated. Mathematically, the COM is calculated as the weighted average of the positions of all the individual mass elements in the system, with the weights being the masses of those elements. On the other hand, the center of geometry (COG) is the point that represents the geometrical center of an object. For the calculation of the NP-membrane distance, the COM was picked instead of the COG because of the conformational changes the NP undergoes throughout its interactions with the membrane. The PEO arms of the NP expand and contract regularly, which may change significantly the shape of the NP, resulting in the COG being quite variable during the course of its interaction with the membrane. On the other hand, the COM of the NP is more stable, since it takes longer for the mass distribution of the NP to change, and thus is a better reference point for this analysis.

NP radius of gyration (R_{gyr}). The radius of gyration of the NP is a measure of the size of the NP. The radius of gyration of the NP, R_{gyr}, is described by Equation (6.1)

$$R_{gyr} = \sqrt{\frac{1}{M}\sum_{i=1}^{N} m_i(r_i - R)^2} \tag{6.1}$$

where M is the total mass of the body, m_i is the mass of atom i and R is the mean position of all atoms. Both, the radius of gyration of the core of the NP and of the whole NP were calculated with the MDAnalysis function `radius_of_gyration()`. The core of the NP was defined as the PLGA block of the polymers and EEK peptides, and the whole NP was defined as the PEO-PGA polymer and EEK peptides. This definition of NP whole and core is the same for all other analysis.

NP eccentricity (ε). The eccentricity of the NP was used in this Chapter to study the shape of the NP. The eccentricity computation was calculated with the MDAnalysis function `moment_of_inertia()`. From the moment of inertia, the eccentricity ε was calculated with the following formula:

$$\varepsilon = 1 - \frac{I_{min}}{I_{mean}} \tag{6.2}$$

where ε is the eccentricity value, I_{min} is the minimum moment of inertia across all axis, and I_{mean} is the mean moment of inertia over all axis. If $\varepsilon = 0$, then the structure is considered to be a perfect sphere, and as values go further way from 0, the structure is increasingly oblong.

Radial distance of EEK peptides with respect to the NP COM. To determine if the peptides changed their storage location within the NP throughout the simulation, the radial distance between the peptides and the COM of the NP was calculated. At each time step, the distance between the peptides COM and the NP COM was calculated using the MDAnalysis function, `mda.distances.distance_array`.

Fractions of EO monomers in contact. The fraction of EO monomers interacting with at least one lipid was calculated by analysing how many EO monomers had at least one contact with any lipid atom. A contact is counted if an EO monomer is at a maximum distance of 6 Å from any lipid atom. The distance of 6 Å was obtained from the second peak of the radial distribution function plot between the EO monomers and lipids, calculated with the MDAnalysis radial distribution function `rdf_calc.InterRDF`. This contact calculation was computed over the course of the trajectory. Per time step, the number of contacts was divided by the total number of EO monomers to obtain the fraction of EO monomers in contact with a lipid out of all EO monomers.

Contacts lipids-EO. The contacts between the lipids and the EO is used in this Chapter to study the interactions between the polymer and the membrane. The number of contacts per time step was calculated by counting how many lipid atoms were at a distance smaller than 6 Å from any EO monomer. This was calculated per lipid species and for all lipids.

Fractional enrichment of EO-lipid interactions. The fractional enrichment of EO-lipid interactions was calculated to discern whether these interactions between EO monomers and a specific lipid species were primarily due to the lipid's high presence in the membrane or indicative of a genuine preferential interaction with PEO polymers. Therefore, contact enrichment is a quantitative measure used to assess the preferential interaction between distinct molecular species within a given system. It quantifies the extent to which one molecule exhibits a higher tendency to interact with another molecule, relative to their respective abundances. This metric is computed by comparing the observed interaction frequency involving a specific molecule to the expected average interaction frequency, considering the relative proportions of the molecules involved. This contact enrichment $N_{\text{enrichment}}$ is calculated with the following formula:

$$N_{\text{enrichment}} = \frac{N_{\text{lipid\_species}}/N_{\text{tot\_lipid}}}{P_{\text{lipid\_species}}/P_{\text{tot\_lipid}}} \tag{6.3}$$

where $N_{\text{lipid\_species}}$ is the number of contacts between EO monomers and a specific lipid species, $N_{\text{tot\_lipid}}$ are the total contacts between EO monomers and all lipids. $P_{\text{lipid\_species}}$ are the number of lipids belonging to that specific lipid species and $P_{\text{tot\_lipid}}$ are the total number lipids. A contact between the lipid and a EO monomer was considered if any lipid atom was within 6 Å of any EO monomer. The enrichment values reported in this Chapter are the time averages.

EEK distance to membrane. To study if the peptides also have selectivity towards cancer cells, the distance in z between the peptides center of mass and the upper leaflet of the membrane was calculated. This was computed as the norm between the z position of the center of mass of each peptide and the average z position of the membrane upper leaflet phosphate groups.

Probability of PEO polymer extension on membrane. To quantify the extension of the PEO polymers once in contact with the membrane, the probability of PEO arm extension on the membrane was computed. This calculation was performed at different time steps to provide a time-evolution of the PEO extension. This was calculated by obtaining the two dimensional distance (using only the x and y coordinates) from the COM of the NP of those EO monomers that were in contact with at least one lipid atom. Then these distances were divided by the length of the membrane in x (the membrane has the same size in x and y) so that the distance value provided is a percentage of the membrane extension. This is useful to know how much of the membrane the polymers are able to cover. Subsequently, these distances are binned and the number of distance values in each bin were counted. Afterwards, the number of values in each bin was divided by the number of total values binned, to calculated the probability. This calculation was performed over 100 frames and averaged over those frames. The times of the analysis were 500ns, 1 µs and 2 µs to see how the probability distribution of the PEO arm extension varies over time.

Lipid coordination number. The lipid coordination number of EO monomers serves as a metric for assessing the number of lipid atoms in close proximity to a specific EO monomer, essentially indicating how closely packed lipid atoms are around it. This measure aids in determining the extent to which polymers are embedded within the membrane, as deeper embedding correlates with a more densely packed lipid environment surrounding the EO monomers. This is calculated per time step, by counting how many lipid atoms are in contact with a specific monomer type. Again, two atoms are defined to be in contact if the distance between them is less than 6 Å. Then, the total number of contacts per time step is divided by the amount of monomers of that specific type to normalize the results.

Membrane thickness calculation. Membrane thickness was used to quantify the degree of NP-induced perturbation in both membranes. In order to perform this analysis, the membrane was centered in the box using the LiPyphilic tool `lipyphilic.transformations.center_membrane` [166]. Afterwards, due to high membrane curvature, the membrane thickness per time step was calculated with FATSLim [212] and it was plotted with Lipyphilic [166]. FATSlim divides the membrane into a three dimensional grid, where each cube of the grid contains a few lipids. This allows the calculation of the thickness per cube of the grid, obtaining the thickness of different points of the membrane. This also allows FATSlim to easily account for membrane curvature in the thickness calculation. The membrane thickness was plotted with the LiPyphilic tool, `lipyphilic.projection_plot()` which allows a 2-d top view representation of the membrane, displaying a heatmap of the thickness throughout the membrane. Also, in these plots the 2-d representation of the NP on top of the membrane is shown by two circles centered at the 2-d COM of the NP, where the radius of the inner circle is the R_G of the NP core and the outer circle is the R_G of the whole NP.

This representation shows the correlation between the NP position on the membrane and the decrease in membrane thickness under that region.

Confidence interval calculation. Error intervals reported in this Chapter are the confidence intervals calculated at a 90% confidence. They were calculated with the following formula:

$$CI = \bar{x} \pm z\frac{\sigma}{\sqrt{n}} \tag{6.4}$$

where $\bar{x}$ is the mean of the sample, z is set to 1.645 for a 90% CI, σ is the standard deviation and n is the sample size.

6.3 Results and Discussion

6.3.1 NP approach to membrane

The first parameter studied to ascertain whether the NP displays preferential selectivity with either of the membranes is the distance between the center of mass (COM) of the NP and the upper leaflet of the membrane. This distance is depicted for both simulations in Figure 6.2 (d). Notably, the NP approached each of the membranes at different speeds, reaching the cancer membrane significantly faster than the healthy one. Both simulations began with the NP positioned at the same initial distance from the membrane, as depicted in Figure 6.2 (a). Figure 6.2 (d) shows that the NP initially approached both membranes similarly until approximately 700 ns. From this point, the NP starts advancing toward the cancer membrane much more rapidly than the healthy membrane. It's worth noting that the NP has a core radius of approximately 7 nm, represented in Figure 6.2 (d) by the dashed line. Therefore, when the distance between the NP COM and the membrane upper leaflet falls below the NP core radius, it signifies that the NP is embedding itself within the membrane. In the case of the cancer simulation, this occurs just before 1 µs, while in the healthy simulation this happens towards the end of the simulation, just before 2 µs. This observation suggests preferential interactions between the polymer components of the NP and the lipids of the cancer membrane. Specifically, interactions between the PEO polymers and the membrane play a crucial role in pulling the NP toward the membrane, as it can be seen in 6.2 (b). Consequently, Figure 6.2 (d) leads to the hypothesis that there are more interactions between the cancer lipids and the NP, enabling them to exert a stronger and faster pull on the NP toward the membrane compared to the healthy counterpart. Furthermore, not only the NP approached the cancer membrane faster, but it was also able to insert deeper into the membrane. Figure 6.2 (c) shows the NP embedded into the cancer membrane.

6.3.2 Quantification of NP physical changes during its interaction with the membrane

The NP physicochemical characteristics impact its cancer selectivity and drug delivery capabilities. In particular, the shape of the NP can influence its ability to insert into the membrane [51, 213]. Among the NP shapes that have shown a greater efficacy in breaching lipid membranes are ellipsoidal or rice-shaped NPs [214]. These elongated geometries facilitate insertion into lipid bilayers, thereby enhancing their membrane-penetrating properties. The mechanistic underpinnings of this phenomenon involve reduced steric hindrance and an increased contact area between the NP surface and the lipid membrane. Furthermore, Guo *et al.* [215] demonstrated using Monte Carlo simulations that NP can change their shape in response to interactions with the membrane. In the simulations presented in this Chapter, the NP also undergoes changes in its shape and size during interactions with both membranes. These changes were quantified by measuring the eccentricity and radius of gyration of the NP core and whole NP during the interactions with the membranes.

Figure 6.3 shows the changes in the radius of gyration and eccentricity of the NP in both simulations as a function of the NP proximity to the membrane. From here, it is clear that the NP undergoes size changes during the interaction with the membranes, as evidenced by variations in the radius of gyration and eccentricity over the course of the simulation. These results align with the simulations finding in Guo *et al.* Figures 6.3 (a) and (d) depict the changes in the R_G for the cancer and healthy simulation respectively. From these figures, it is clear that the general trend in both simulations is that the radius of gyration of the whole NP increases as the NP is closer the membrane. This is due to the interaction of the PEO arms

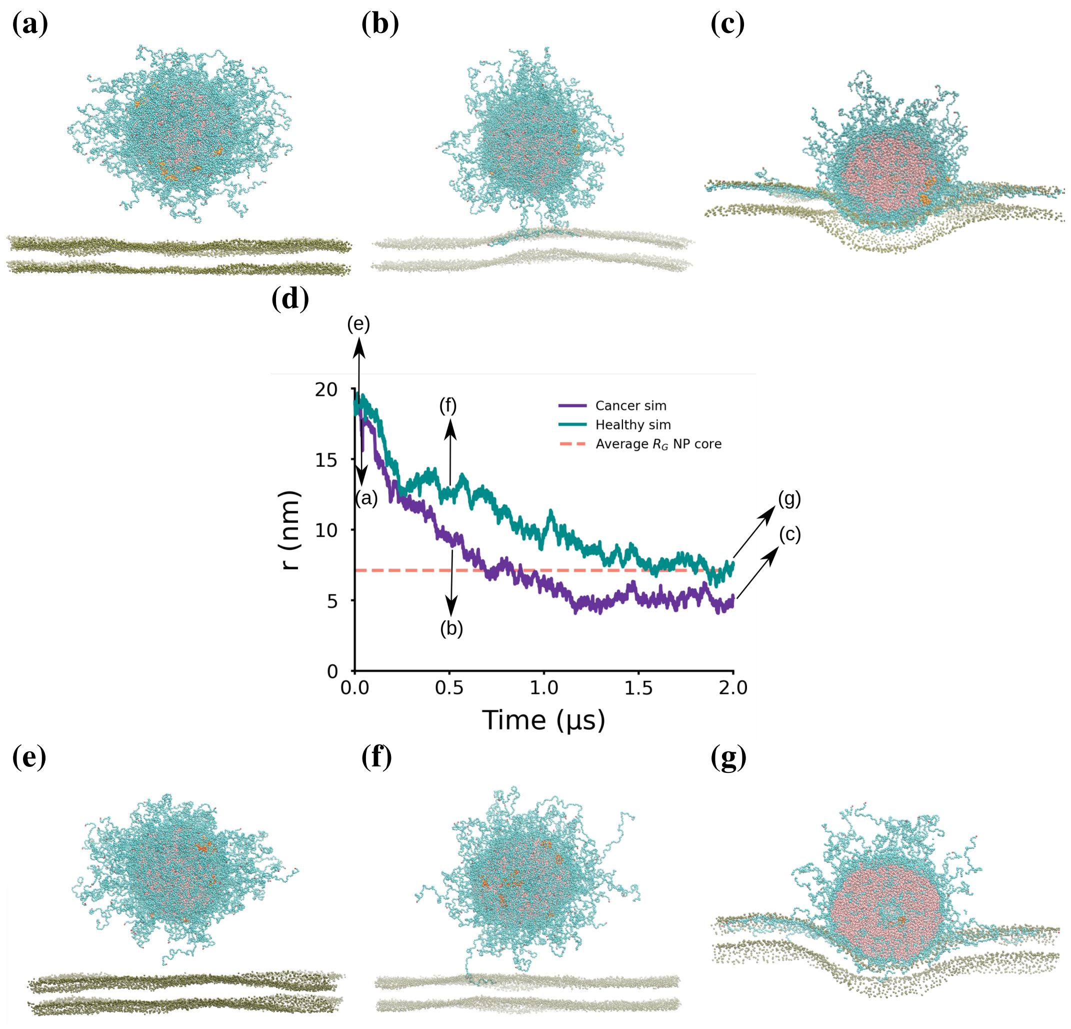


Figure 6.2: PEG-PLGA NP approaches cancer membrane faster. (a) Snapshot of NP and cancer membrane (a) at the start of the simulation, (b) at 500ns and (c) cross-section view at approximately 2 μs. Snapshot of NP and healthy membrane (e) at the start of the simulation, (f) at 500ns and (g) at 2 μs. In all snapshots, PEG is in blue, PLGA in pink, EEK in orange and phosphate groups of the membrane in green. Snapshots in (b) and (f) have the phosphate groups colored transparently so the extension of the PEO arms on the membrane can be appreciated. Moreover, in snapshots (c) and (g) the membrane curvature induced by the NP is clearly visible, being more pronounced in (c) than (g). Snapshot are not to scale. (d) Distance in the z dimension between the NP center of mass (COM) and the average phosphate group positions in the upper leaflet. In green, the distance obtained from the healthy simulation and purple from the cancer one. The R_G of the NP core is represented by the dotted line in salmon. Therefore, if the distance between the membrane and NP is below the dotted line it means the NP is embedded into the membrane.

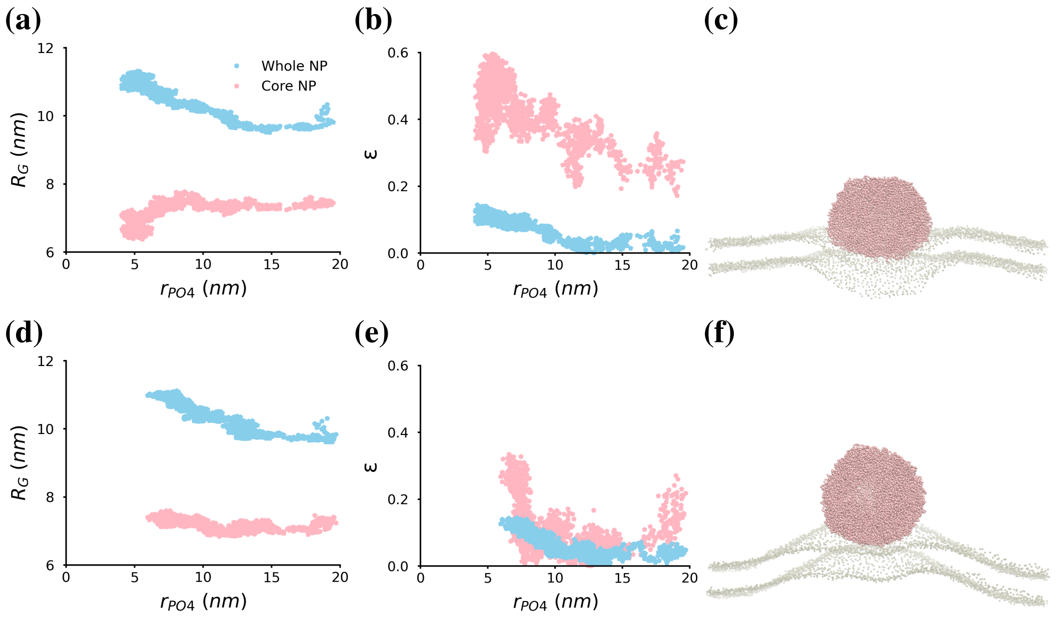


Figure 6.3: Differences in the shape of NP during its interactions with the membranes. Changes in the value of the R_G of the core and whole NP as a function of a distance to the phosphate groups of the upper leaflet for (a) cancer and (d) healthy simulation. Changes in the value of the eccentricity (ε) of the core and whole NP as a function of a distance to the phosphate groups of the upper leaflet for (b) cancer and (e) healthy simulation. Snapshot of the PLGA core (to illustrate its shape) interacting with (c) cancer and (f) healthy membranes. The PLGA core is displayed in pink and the phosphate groups of the membrane in transparent green. Snapshots are not to scale.

with the lipids within the membrane. Throughout the simulations, the PEO arms extend over the surface of the membranes, and they also extend into (and in some cases across) the hydrophobic core of the membrane. These PEO arm extensions are the responsible for the overall increase in the R_G of the NP. Meanwhile, when considering the radius of gyration of the core of the NP, it is harder to observe a clear trend in the NP-healthy membrane simulation, as seen in Figure 6.3 (d), where the R_G of the core takes approximately constant values. In contrast, the NP-cancer membrane simulation exhibits a pronounced trend. Figure 6.3 (a) shows that the R_G of the NP core decreases once the distance between the NP and membrane falls below 10 Å, and continues to decrease as the NP embeds itself deeper into the membrane. This observation implies that the NP core becomes denser and more rigid as it inserts deeper into the membrane.

The R_G measures the size of NPs, but to quantify the shape of pseudo-spherical NPs, the eccentricity (ε) is normally used. The eccentricity is a measure of how spherical an object is. An ε of 0 means the body is a perfect sphere, and as values increase approaching 1, the body is increasingly oblong. Similar to the R_G, the eccentricity of the entire NP experiences an increase as the NP approaches both membranes, as illustrated in Figures 6.3 (b) and (e) for the cancer and healthy membranes, respectively. This rise in eccentricity for the entire NP is attributed to the extension of the PEO arms, which reduces the overall sphericity of the NP. Again, determining a distinct trend for the eccentricity of the NP core during the interaction with the healthy membrane is challenging. As observed in Figure 6.3 (e), these eccentricity values exhibit no specific pattern in relation to the distance between the NP and the membrane, with all ε values contained within the range [0 and 0.32], meaning that the NP core is approximately spherical during its interaction with the healthy membrane. A snapshot of the NP core at 1.5 µs is depicted in Figure 6.3 (f). From this snapshot, it is evident that the NP core is mostly spherical. This, coupled with the minimal variations in the R_G of the NP core during the interaction with the healthy membrane, indicates that the NP core undergoes few physical alterations in the healthy simulation.

Conversely, regarding the eccentricity of the NP interacting with the cancer membrane, the trend is again very clear, the NP PLGA core adopts a more oblong shape the more it interacts with the cancer membrane, eventually reaching ε values of 0.6, as depicted in Figure 6.3 (b). A snapshot of the NP core interacting with the cancer membrane at 1.2 µs is shown in Figure 6.3 (c). This snapshot clearly showcases this transformation of the NP core as it interacts with the cancer membrane, showing that the NP core has adopted a more oblong or ‘rice-like’ shape. The reduction in the R_G of the NP PLGA core, coupled with the increase in eccentricity, indicates that the NP core undergoes size and shape alterations during its interaction with the cancer membrane, becoming more compact and stiff (an overall reduction in volume) and taking on an ellipsoidal-like form. This shape results in a larger contact area between the PLGA core and the membrane compared to the spherical shape. Furthermore, recent studies [213, 214] show that NPs that are more stiff or have rice-like shape are able to transcytose faster than spherical soft NPs. Hence, according to existing literature, the physical modifications undergone by the NP core during the interaction with the cancer membrane promote rapid transcytosis. Moreover, the reduction in size and increase in NP surface area in contact with the membrane, also favours the delivery of the peptides and their diffusion into the membrane, because this proximity allows the EEK peptides to approach the membrane more closely and increase their contact area with the lipids.

These changes in size and shape of the PLGA core probably occur due to the rapid interactions of the PEO polymers with the cancer lipids. The extension of the PEO arms over the membrane causes the PLGA core to be less shielded from water (since a high amount of PEO monomers are now interacting with the membrane instead of covering the PLGA). This forces the PLGA core to collapse to minimise its contacts with the water molecules, reducing its R_G, and at the same time, to increase its surface in contact with the membrane (reducing the surface exposed to the water molecules), thereby elevating its eccentricity. It is also possible that the PEO arms reaching the membrane and extending over it, may exert pressure on the PLGA core, forcing it to adopt a more oblong-like shape.

Overall, the rapid interactions of the PEO polymers with the cancer lipids, demonstrated by the fast approach of the NP to this membrane, induce changes in the PLGA core of the NP. These changes not only facilitate the approach of the peptides to the membrane, but also make the PLGA core adopt a size and shape that favour NP transcytosis through the membrane.

6.3.3 Behaviour of nano-encapsulated EEK peptides

EEK peptides are encapsulated into this PEG-PLGA NP at two different storage locations. The first location is deep into the PLGA core, and there are two peptides, and the second storage location is at the core-shell interface, where the remaining 13 peptides can be found. These two storage locations should be understood as specific ranges of radial distances from the COM of the NP, so that the peptide can move within that radial distance and would still be considered to be in the same storage

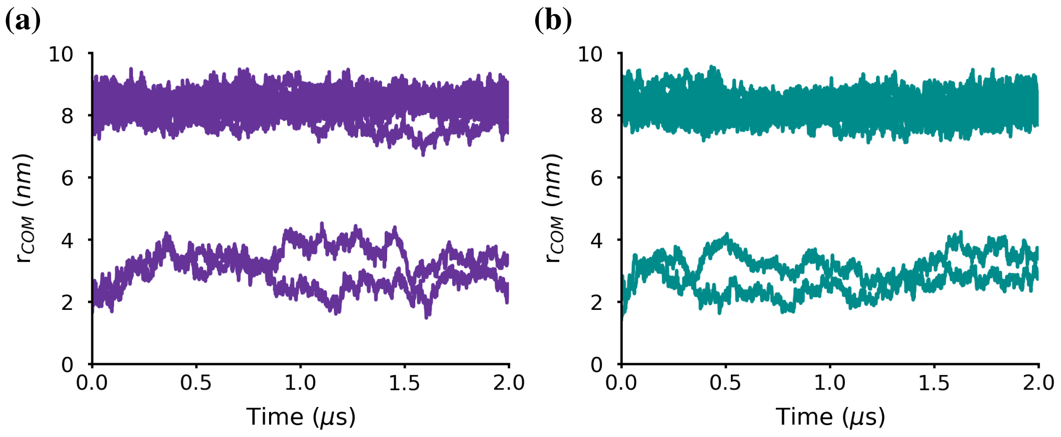


Figure 6.4: Peptides storage location remains stable during interactions with the membrane. Radial distance with respect to the NP COM of each of the peptides for (a) cancer and (b) healthy simulation. Clearly, there are two storage locations in the NP, one deeper into the NP core, where two peptides reside, and another one at the core-shell interface with the remaining 13 peptides.

location. Details on both of these storage locations can be found in Chapter 5, which discusses how EEK peptides are encapsulated into these two distinct locations. In order to determine if the storage location of the peptides change as the NP interacts with the membrane, the radial distance between each of the EEK peptides COM and the NP COM was calculated per time frame. Figure 6.4 shows the distance for all peptides throughout the simulations. In both membranes, cancer (Figure 6.4 (a)) and healthy (Figure 6.4 (b)), peptides do not change significantly their storage location, since they are at approximately the same distance from the NP COM throughout the simulations. This is probably due to the fact that there are still enough PEO polymers surrounding the NP so that it is still more preferential for peptides at the core-shell interface to stay there than to diffuse into the membrane. The same explanation can be applied to the peptides in the core storage location.

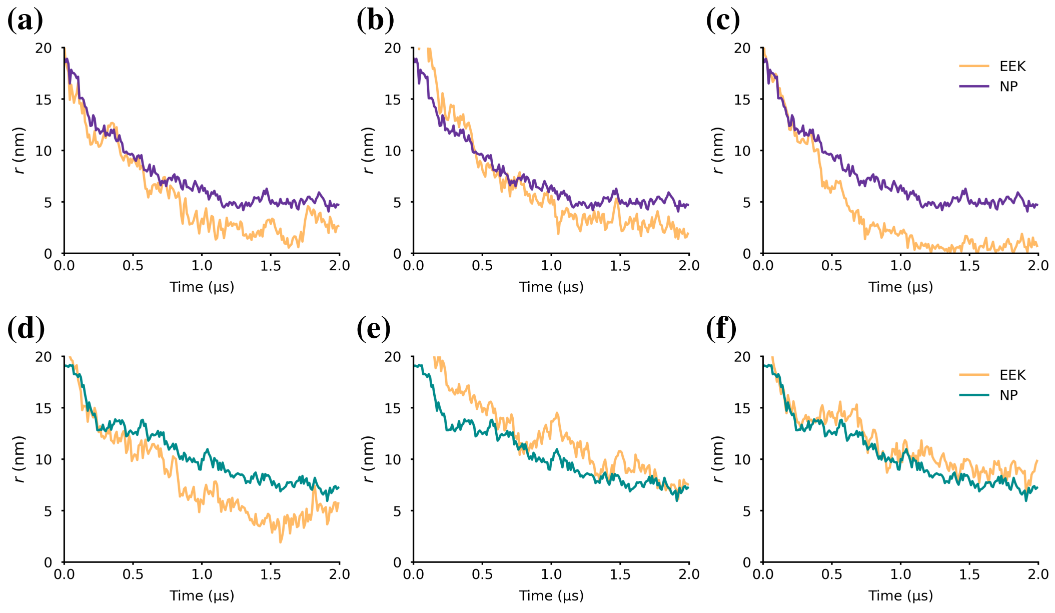
However, the fact that peptides do not change their radial position does not mean that they are not moving towards the membrane, since they could be moving towards to the membrane within their radial storage space. To quantify this, the distances between the z coordinate of each EEK COM and the average z position of the phosphate groups of the upper leaflet of both membranes were calculated. The average over the last 50 frames of the simulation, so when both NP are in contact

Simulation	Distance (Å)
Cancer	29.47 ± 2.43
Healthy	56.72 ± 3.25

Table 6.2: Average distance EEK and membrane Average over the last 50 frames of the distance between all EEK peptides and the average upper leaflet phosphate group of the membrane.

with their respective membranes, is shown in Table 6.2.

Table 6.2 reveals that in the cancer simulation, EEK peptides approach the membrane more closely compared to the healthy simulation. This means that peptides in the cancer simulation, keeping their radial distance to the NP COM approximately the same, move towards the lower part of the NP, which is the one closer to the membrane. Furthermore, Figures 6.5 (a)-(f) show the z distance between some of the peptides COM and the phosphate group of the upper leaflet lipids for the healthy and cancer simulation. Additionally, to show that the peptides are not static within the NP, the z distances between the COM of the NP and the membranes are also plotted in Figures 6.5 (a)-(f). Since the peptide-membrane distance and NP-membrane distance evolve differently, it is clear that the peptides move independently of the NP. Figures 6.5 (a)-(c) depict the movement of three peptides in the cancer simulation and Figures 6.5 (d)-(f) display the same peptides but for the healthy simulation. It is important to note that peptides have the same starting positions at the beginning of both simulations. From theses Figure, it is evident that the EEK peptides in the cancer simulation consistently move towards the membrane, regardless of the NP movement. These peptides approach the membrane more rapidly and closely than their healthy counterparts. In contrast, peptides in the healthy simulation do not exhibit a clear trend towards the membrane. Their movement is more aligned with the NP movement, as shown in Figures 6.5 (e)-(f). Another difference across simulations is that EEK peptides in the cancer simulation continuously move towards the membrane, while in the healthy simulation, peptides oscillate directions more, which can be particularly appreciated in Figure 6.5 (e), where the peptide does not consistently reduce its distance from the membrane. Figure 6.5 (g) illustrates a snapshot of the PLGA core (pink) with the peptides (orange) interacting



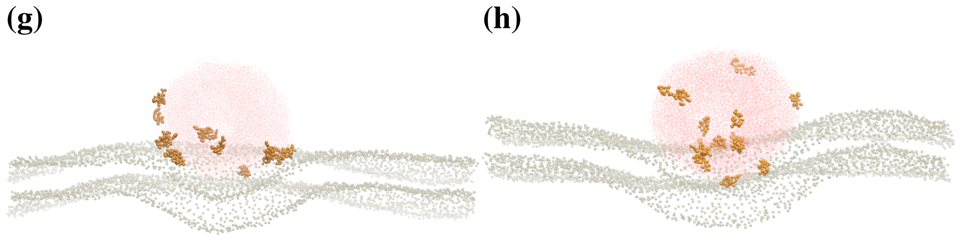


Figure 6.5: EEK peptides are selective towards cancer membranes. z-distance between specific EEK peptides and membrane upper leaflet for (a), (b) and (c) cancer simulation and (d), (e) and (f) for the healthy simulation. Snapshots of the PLGA core (pink), peptides (orange) and phosphate groups of the membrane lipids (dark green) in the (g) cancer simulations and (h) healthy simulations. Snapshots are not to scale.

with the cancer membrane (green). Here, most peptides have migrated to the lower part of the NP, which is closer to the membrane. On the other hand, Figure 6.5 (h) depicts the same scenario but for the healthy simulation. This figure illustrates that the peptides are more evenly distributed throughout the NP. Note that both snapshots, Figures 6.5 (g) and (h) were taken at approximately 1.5 μs. Consequently, these results indicate that not only the NP exhibits selectivity towards cancer cells, but the peptides also preferentially interact with cancer cell membranes.

6.3.4 PEO selective interactions with cancer lipids

The PEO polymers are the ones in charge of dragging the NP towards the membrane, as seen in Figure 6.2 (b). Once in contact, they begin to spread over the

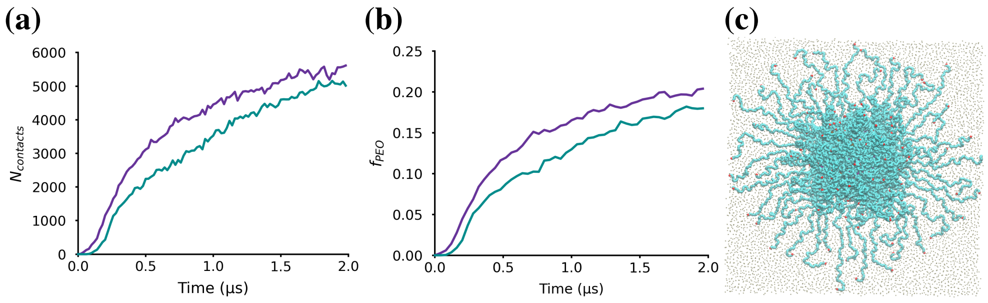


Figure 6.6: PEO polymers have more interactions with the cancer membrane lipids over the course of the simulation. (a) Total number of contacts between EO monomers and any membrane lipid. Results for cancer simulation in purple and healthy simulation in green. (b) Fraction of EO monomers interacting with at least one lipid with respect to time for the cancer (purple) and healthy (green) simulations. (c) Snapshot of the top-view of the NP (in blue) interacting with the cancer membrane (phosphate groups in brown). Here, the spreading of the PEO arms is visible. This snapshot is not to scale.

membrane. PEO polymers are widely used in cancer nanotechnology, since they increase the biocompatibility of the drug, as well as reduce the systemic clearance, cytotoxicity and interactions with blood components among many other advantages [216, 217]. Furthermore, several studies show that NPs conjugated with PEO polymer, increase their cancer selectivity [2, 217]. Despite the studies on the advantages of PEO-based delivery systems, there is not a lot of literature addressing the mechanisms behind the cancer selectivity characteristics of PEO polymers. From the simulations in this Chapter, it is clear that PEO plays a key role in the interaction with the cell membrane. To gain insights into the potential selectivity characteristics of PEO polymers, the differences between the interactions of PEO polymers with both membranes were quantified.

First, the total number of contacts or interactions between the EO monomers and the membrane lipids were quantified, which is depicted in Figure 6.6 (a). Notably, the cancer simulation exhibit a higher number of contacts compared to the healthy simulation. At the beginning of the simulations, the values are more similar, but just before 500 ns, the difference in the number of contacts between the healthy and cancer simulations increases. This increased can be attributed to the preference of PEO polymers to interact more with the cancer membrane, as evidenced by the rapid approach of the NP to this membrane (Figure 6.2 (d)). Towards the end of the simulation, the difference between the healthy and cancer values decreases, mainly due to the fact that there is a maximum number of possible contacts. As the PEO arms expand through the membrane and begin to cross the periodic boundary condition, there is limited space available for new PEO arms to interact with the membrane. However, the results up to the saturation of the system show a greater number of interactions between the PEO polymers and cancer membrane lipids that with the healthy membrane.

Furthermore, Figure 6.6 (b) shows the fraction of EO monomers in contact with the membrane lipids. This Figure shows that a higher percentage of EO monomers are interacting with the lipids in the cancer cell membrane than in the healthy membrane, further underscoring preferential interactions. The saturation of the system can also be observed here, since the percentages start plateauing towards the end of the simulation. If the system was bigger, the fraction of EO monomers interacting would keep increasing. However, the plateauing only occurs towards the end, meaning this phenomena have minimal impact on the results reported here. For visual purposes, Figure 6.6 (c) shows the top view of the NP cancer simulation at approximately 1.5 µs. As it can be seen, the PEO arms spread over the membrane as they increase their interactions with the lipid components. Additionally, it is noticeable that the PEO arms are on the verge of crossing the periodic boundary at this stage.

It is clear that PEO polymers exhibit more interactions with the cancer membrane than with the healthy membrane. In the simulations presented in this Chapter, the difference between the membranes is the lipid composition. Therefore, in order to understand why PEO polymers preferentially interact with the cancer membrane, the contacts per lipid species were analysed. Figures 6.7 (a) and (b) depict the total number of contacts between EO monomers and each lipid species within the cancer and healthy membranes. It is clear that the highest number of contacts in boths simulations is with POPC lipids, however the second highest is with DPSM. Interestingly, DPSM lipids are present in higher percentages in cancer membranes [208, 209]. In order to analyse if the interactions of EO monomers with the different lipid species were preferential or were merely a result of the high presence

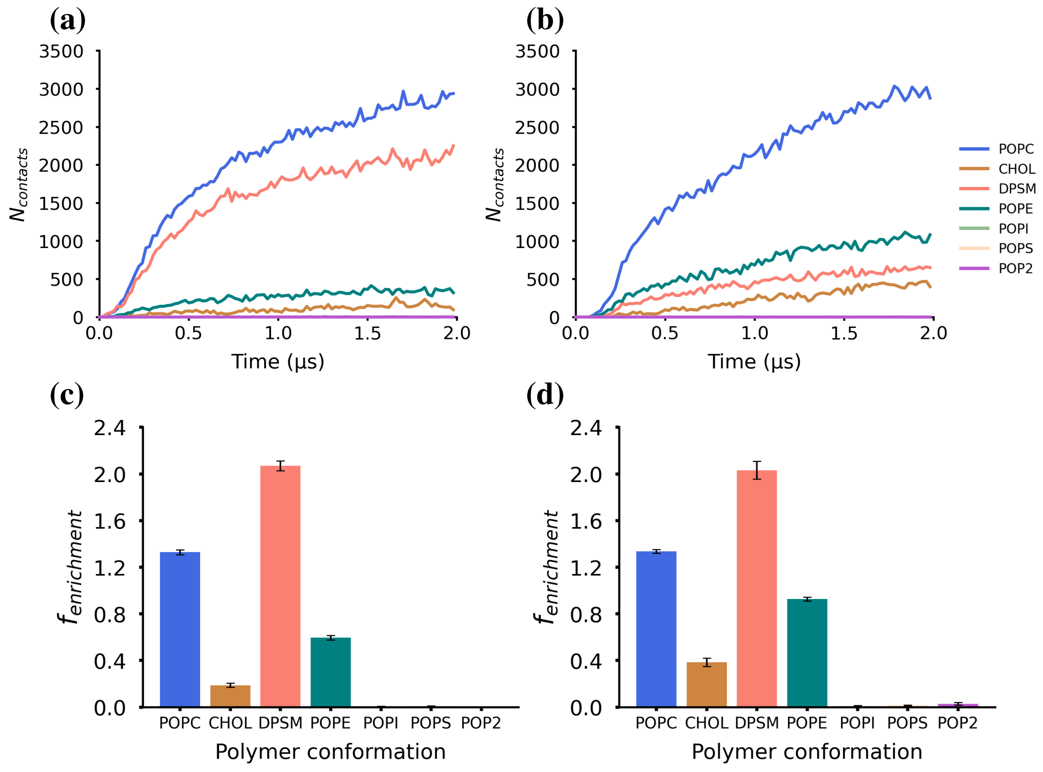


Figure 6.7: PEO polymers preferentially interact with DPSM lipid species. (a) Fraction of EO monomers interacting with at least one lipid with respect to time for the cancer (purple) and healthy (green) simulations. (b) Snapshot of the top-view of the NP (in blue) interacting with the cancer membrane (phosphate groups in brown). This snapshot is not to scale.

of these lipid types in the membrane, the fractional enrichment of the contacts was calculated. The fractional enrichment quantifies the tendency of an interaction between two molecules, taking into account their respective abundances. Figures 6.7 (c) and (d) show the enrichment of the interactions between the EO monomers and the lipid types for the cancer and healthy simulation, respectively. An enrichment of 1, means the interaction is random, values above 1 mean there is a preferential interaction, and values below 1 mean the interaction is not favourable. Clearly, in both systems, interactions with DPSM lipids are the most preferential (enrichment value above 2), while interactions with POPC are slightly preferential. Interactions with all other lipids are not favourable. Therefore, this shows that PEO polymers have a distinct tendency to interact with DPSM lipids over other lipids studied in these simulations. Clearly, one of the reasons for the selectivity of PEO-based NP towards cancer cells is the preference of the polymers to interact with lipids present

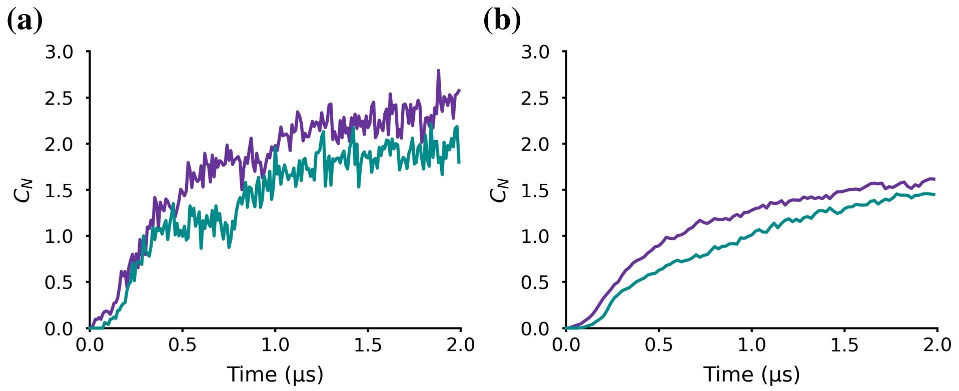


Figure 6.8: Last PEO monomer has a higher tendency to interact with the membrane lipids. Lipid coordination number of (a) last OH monomer of PEO polymers (b) non-terminal C monomers of PEO polymers. The cancer simulation is in purple and healthy in green.

in higher percentages in cancer membranes, such as DPSM.

The CG representation of a PEO polymer consists of two monomer types, the terminal monomer, which is an OH group and the second monomer is the EO (ethylene oxide) type, which makes up all the non-terminal PEO monomers. To understand the role of these two monomer types in the interactions with the membrane lipids, the lipid coordination number was calculated. This parameter quantifies the number of nearest neighbor lipid beads for each monomer type. A higher coordination number, means that the monomer is surrounded by more lipids. This means, the monomer not only has more interactions but that is probably more embedded into the lipid environment. The normalized lipid coordination number of the OH and EO monomers for both simulations are depicted in Figure 6.8. Here it is clear that in both simulations, the OH monomer type (Figure 6.8 (a)) has a higher coordination number than the EO monomer type (Figure 6.8 (b)). This means the OH monomer type has a higher tendency to interact with the membrane and it is also more deeply embedded into the lipid environment. It is likely that the last monomer of the PEO polymers can interact more readily with the membrane due to its position at the end of the chain, providing increased flexibility and the capacity for membrane insertion. Also, PEO OH beads are more polar than EO beads, favoring the formation of hydrogen bonds with the membrane. It is important to note, that for both monomer types, the monomers interacting with the cancer lipids have higher

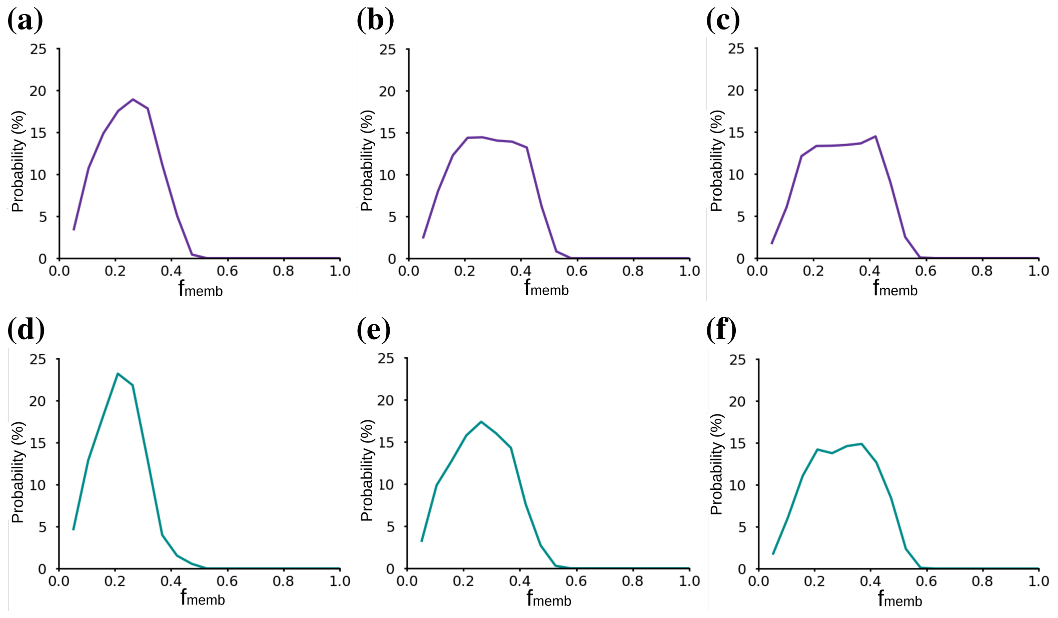


Figure 6.9: PEO polymers spread across the membrane. Radial probability of PEO arm extension with respect to the membrane x-y dimensions. The data presented is the average over 100 frames at 500ns, 1μ s and 2μ s, respectively, from left to right. The calculations on the cancer membrane are (a)-(c), and healthy membrane (d)-(f).

lipid coordination number than their healthy counterparts, in agreement with the higher number of contacts in the cancer simulation. This means that monomers are more inserted in the cancer simulation than in the healthy one. These results show that terminal monomers in PEO polymers play an important role in the interactions with the membrane lipids.

From the above results, it is clear that PEO polymers play a key role in the selectivity and preferential interactions of the NP with cancer cells. Once the PEO polymers contact the cancer membrane, they begin to spread over it, maximizing its contacts, as depicted in Figure 6.6 (c). To quantify this phenomenon, the probability distribution of the PEO arm extension with respect to the total membrane length was calculated. This distribution was calculated at different time steps to show the time evolution of the systems. The PEO extension probability distribution is shown in Figure 6.9. This Figure shows the distribution at 500 ns, 1μs and 2 μs. Figures 6.9 (a)-(c) show the results for the cancer simulation while Figures 6.9 (d)-(f) for the healthy simulation. For the calculations at 500 ns and 1μs the cancer simulation

(Figures 6.9 (a)-(b)) has a wider distribution than the healthy simulation (Figures 6.9 (d)-(e)), meaning PEO polymers reach larger extensions. This is expected, since PEO polymers interact more with the cancer membrane. For the probability distribution at 2 µs, the distribution of PEO extensions on the cancer (Figure 6.9 (c)) and healthy (Figure 6.9 (f)) membranes are similar. The difference is that the extension distribution is more evenly distributed in the cancer simulation, with slightly higher probabilities for the higher extension values compared to the healthy counterpart. However, it is important to note that at this point the membrane is already saturated, meaning that the PEO arms are crossing the periodic boundary, making the membrane very crowded and limiting the space available for PEO polymers to keep expanding over the membrane. Also, once the PEO monomers cross the periodic boundary, the distances calculated do not account for the real extension. This results in the probability distribution values of the cancer and healthy membrane being similar at this stage. Overall, the PEO arm extension is larger for the polymer interacting with the cancer membrane, as it was expected from the higher number of interactions with this membrane. Also, these results show that as the number of interactions increase, the PEO arm extension also increases, establishing a link between these two events. Therefore, PEO interactions with the membrane result in PEO maximising its contacts with specific lipid species by expanding over the membrane.

6.3.5 NP disruption of lipid membrane

An advantage of conducting MD simulations of membranes is the ability to easily quantify parameters such as membrane thickness with higher molecular detail than is achievable in experiments. Membrane thickness can be used as an indicator of membrane disturbance, as prior to membrane rupture or invagination of external particles, there is membrane thinning. From the previous results, it is clear that PEG-PLGA NP selectively interact with the cancer model membrane over the healthy model. To determine if this interaction also leads to greater membrane disturbance, the membrane thickness at different times throughout the simulation was

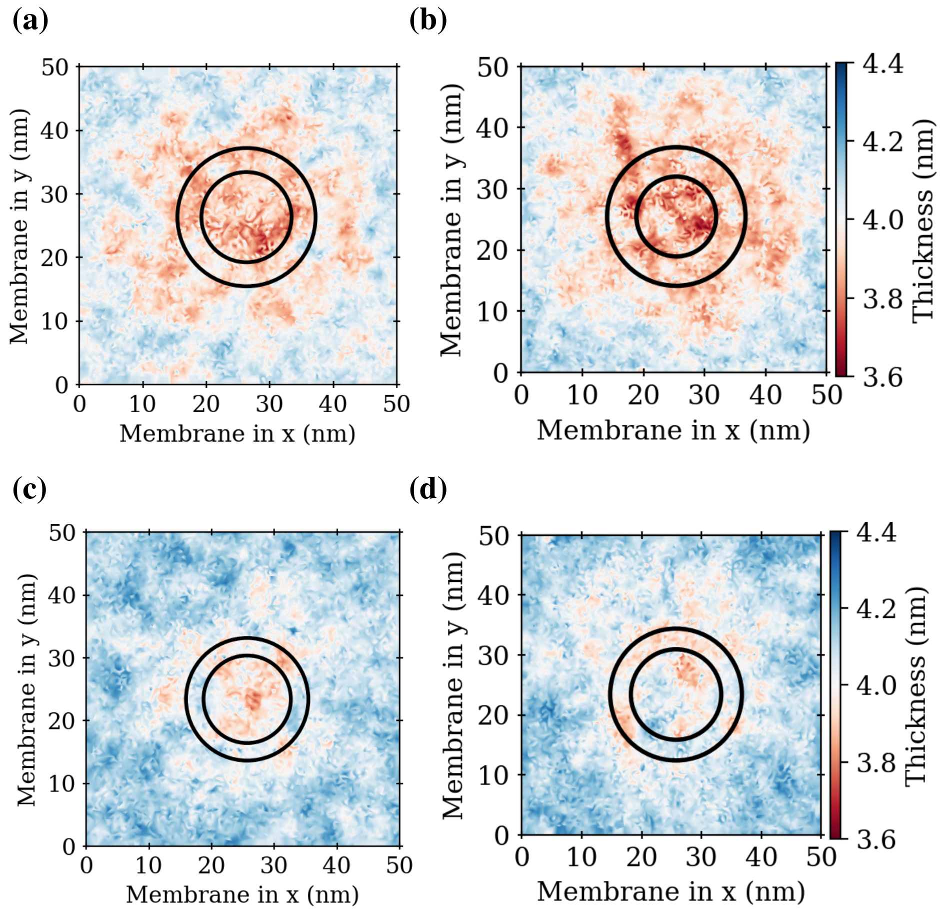


Figure 6.10: NP disrupts cancer membrane more than healthy membrane. 2-d visualization of membrane thickness at 1µ s and 2µ s for the cancer ((a)-(b)) and healthy ((c)-(d)) membrane respectively. Both circles are centered at the x-y coordinates of the radius of gyration of the NP. The radius of the inner circle is the radius of gyration of the core and the outer circle the radius of gyration of the whole NP. Note that for the cancer membrane the average initial thickness is approximately 4.2nm for both membranes.

measured.

Figure 6.10 presents a 2d top view representation of the NP-membrane systems. The membrane is the heatmap itself and the black circles are the NP. Both circles are centered on the NP COM, the radius of the inner circle is the average NP core radius, and the radius of the larger circle is the average whole NP radius over the calculation time steps. The coloring of the heatmap represents the membrane thickness. In both membranes, cancer and healthy, the NP induces membrane thinning, as the decrease in membrane thickness is greater in the areas of the membrane below or surrounding the NP. Additionally, Figures 6.10 (b)-(d) have lower thickness values than Figures 6.10 (a)-(c) respectively, indicating an increase in membrane thinning over the course of the simulations. However, it is clear that the membrane thinning is much more pronounce in the cancer membrane (Figure 6.10 (a)-(b)) compared to the healthy one (Figure 6.10 (c)-(d)). Firstly, the cancer membrane adopts significantly thinner values than the healthy one. Secondly, the membrane area affected by the thinning, is larger in the cancer membrane than in the healthy one at all time steps. Consequently, the NP induces greater membrane disturbance in the cancer membrane than in the healthy membrane. From Figures 6.10 (a)-(b) the reduction in the R_G of the NP core can also be observed, since the inner circle decreases its size from Figure 6.10 (a) to (b). The primary reason for the increased membrane thinning in the cancer membrane is the PEO preference to interact with lipids present in higher percentages in cancer membranes, such as sphingomyellin (DPSM), as shown in Figure 6.7 (c). The higher number of PEO interactions and greater extension of the PEO arms over the membrane do not induce stress on the membrane on their own. As discussed earlier, these PEO interactions with the cancer lipids force the PLGA core to collapse and maximise its contact surface with the membrane, which likely also increases the pressure exerted on the membrane. This would also explain why in the cancer simulation, the thinner values are mostly found just below the NP core (inside the inner circle), while for the healthy simulation they are more evenly distributed throughout the whole NP extension, since the NP core does not change much its shape, it does not induce much pressure on the healthy membrane. These results show that PEG-PLGA NP selectively target, and therefore, disrupt cancer cells more effectively than healthy cells. Overall, this study also demonstrates that membrane disturbance caused by NPs can be studied *in silico*. In summary, the preferential interactions of PEO polymers with cancer membrane lipids, cause the PEO to extend over the membrane and also to induce changes in the PLGA core. Both of these processes increase the pressure exerted on the membrane.

6.4 Conclusion

The results presented in this Chapter provide insights into the specific molecular mechanisms and physicochemical properties underpinning the cancer selectivity of this experimentally validated PEG-PLGA NP. Furthermore, these results contribute to the evaluation of the use of MD simulations to study NP selectivity towards cancer cells *in silico*.

CG MD simulations of the same EEK-loaded PEG-PLGA NP with a glioma (cancer) and oligodendroglia (healthy) model membrane were performed. These simulations enabled the quantitative comparison of NP behavior with the cancer and healthy membrane. Both models were more 'realistic' that other simulation studies of NP interactions with cancer membranes, since all studies carried out previously used membranes consisting of a maximum of three lipid species. Additionally, the lipid species used in the model membranes were sourced from lipidomics research, with those representing over 10% of the total lipid content in real cell membranes being included in the model membranes. These simulations facilitated the analysis of the targeting ability of this NP towards cancer cells by direct comparison of: (i) changes in the physicochemical properties of the NP when interacting with the membranes, (ii) membrane disruption and (iii) differences in the interactions of the NP with both membranes.

First, the NP approached much faster the cancer membrane than the healthy membrane, and it inserted deeper into the membrane as well. The rapid approach and interactions of the PEG-PLGA NP with the cancer membrane can be attributed to the preferential interactions of PEG polymers with lipids, such as DPSM, that are present in higher percentages in cancer membranes. The strong tendency of interaction of PEG polymers with cancer lipids, not only made EO monomers have a higher number of contacts with the cancer membrane than the healthy membrane, but it also caused the PEG polymers to expand over the membrane, reaching longer distances than in the healthy simulation. This shows that polymer-lipid interactions are a key parameter to determine and tune NP cancer selectivity.

Furthermore, these simulations showed that the PEG-PLGA NP physical characteristics are not fixed, but vary during the interaction with the membrane. Interestingly, the NP interacting with the cancer membrane, adopted a size and shape that favours its transcytosis through the membrane. Due to these interactions, PEO arms rapidly expand through the membrane, leaving the PLGA hydrophpbic core less shielded from water. This causes the PLGA core to collapse, reducing its R_G, as seen in Figure 6.3 (a), and also changing its shape into a more elliptical one, to maximise its contacts with the membrane (instead of with the water), as observed in Figure 6.3 (b). These physical changes of the NP PLGA core, plus the extension of the PEO arms over the membrane, induce membrane stress. The PEO arms interact with the lipids and insert into the membrane, while the PLGA core reduced size exerts a higher amount of pressure onto the membrane. Furthermore, NP rice-liked shapes have been shown to exhibit the highest transcytosis rates [214]. The increased membrane stressed due to these factors can be quantified by membrane thinning, which is the step prior to membrane disruption. Figure 6.10 (b) shows that towards the end of the simulation, the cancer membrane is under a large amount of stressed induced by the NP. The areas just below the core, have the greatest membrane thinning, agreeing with the higher pressure exerted by the PLGA core. Moreover, the PEO also cause stress, since there is also membrane thinning around the overall R_G of the whole NP, but it gets reduced radially as a distance from the NP core. What is particularly interesting of these results, is that they are very different from the ones obtained from the healthy simulation. In the healthy simulations, there are also interactions between the PEO polymers and membrane lipids, since PEO polymers have a slight tendency to interact with POPC lipids, as shown in Figure 6.7(d), but since this tendency is much lower, and the sphyongmyelin content in the healthy cell is lower, the number of interactions between the NP and the polymer increase much slower than in the cancer simulation. Also, since these interactions are not as preferential, PEO arms are not able to extend as much nor as fast as in the cancer simulation, as seen in Figure 6.9. This makes the PEO polymers shield the PLGA core for longer, so the PLGA core does not change its size or shape much over the course of the simulation. Since the PEO polymers have a lower number of interactions with the healthy lipids, they do not expand as much or insert as deep into the membrane as in the cancer simulation, shown by the lower lipid coordination number in the healthy simulation (Figure 6.8). This causes the NP to exert a much lower stress on the healthy membrane than on the cancer membrane, as the PLGA core does not exert much pressure with its original size and spherical shape, and the PEO arms do not insert as deeply into the membrane to disrupt the lipids. The combination of all of these results, show that polymer-lipid interactions are a vital parameter to understand NP differential selectivity towards cancer cells. Moreover, these simulations also show the mechanisms of action (PEO interactions and PLGA core changes) by which this specific PEG-PLGA NP induces stress on membranes to further deliver its cargo.

Furthermore, the activity of the ACPs was also monitored throughout the simulation to asses if the active compound is also selective towards cancer cells. The peptides in the cancer simulation advance towards the membrane more independently of the NP core movement than in the healthy membrane, and move to distances nearer to the membrane interface than the peptides in the oligodendroglia system. Therefore, as the peptides in the cancer membrane simulation show a greater tendency to move towards the membrane, it can also be assumed that this antimicrobial peptide is also selective towards glioma cells.

Overall, the work presented in this Chapter shows that it is possible to use MD simulations to study and understand the mechanisms by which polymer-based NP target cancer cells. These results show that NP selectivity may be related to polymer-lipid interactions and changes in the NP physical properties that make it exert a greater pressure on the membrane. The parameters used in this work to determine this specific NP selectivity can be used to study the selectivity of other NPs or other cancer and healthy membrane combinations. Therefore this work can be used as an *in silico* platform to study NP selectivity against cancer cells. The parameters proposed in this platform to assess NP selectivity are: specific lipid-polymer interaction, changes in physical properties of the NP, polymer extension on membrane, membrane thinning and active-compound movement. All of these parameters are either impossible or harder to study experimentally, so can only be examined computationally. The work presented in this Chapter sheds light onto the mechanisms of action by which PEG-PLGA NP are able to selectively target and disrupt cancer cells over healthy cells, and in doing so, an *in silico* platform to study NP cancer selectivity is also proposed. Unveiling these molecular mechanisms is key for the advancement of NP-based cancer treatment, as well as to tune NP to efficiently target only cancer cells, reducing the overall toxicity and therefore increasing the patient's quality of life. The parameters and mechanisms revealed in this Chapter are key for the rational design of NPs for cancer treatment.

6.5 Future work

One of the limitations in this Chapter was the membrane size, since it was not large enough for the NPs to fully extend their PEO arms, so the NPs crossed the periodic boundary and started to interact with themselves and also overcrowded the membrane, preventing other PEO arms to interact with membrane lipids. Therefore these simulations could not run for longer than 2 μs, since at this point the membrane became too saturated with PEO polymers. However, simulating a larger membrane is not possible due to the high computational cost, since the system is already at the forefront of what is computationally possible for the supercomputers I had access to during my PhD (these systems had more than 3000000 beads). To account for this problem, simulations with the same membranes but with NP containing a lower amount of PEG-PLGA polymers (but keeping the PEG to PLGA ratio constant), can be simulated. In these systems, the PEG polymers would have a greater membrane surface to interact with, since the membrane would not become as crowded. This would allow the simulations to run for longer, providing further insight into the physical changes of the membrane and the NP at longer timescales. Furthermore, it would be interesting to study how NP size affects cancer selectivity and if the same trends in the changes of the physical characteristics (decreased size and increased eccentricity) is conserved across different NP sizes. Two smaller NP have already been created, and will soon be put together with the cancer and healthy membranes. Figure 6.11 shows the smaller equilibrated NP formed by only 20 PEG-PLGA polymers.

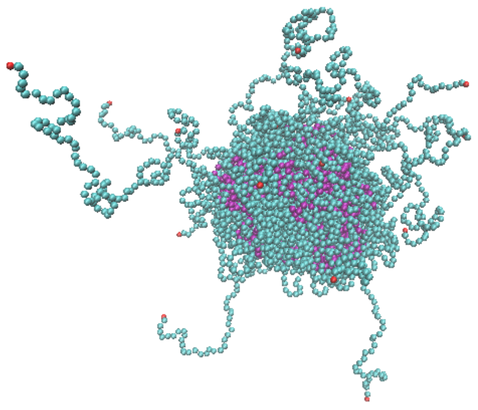


Figure 6.11: Smaller version of EEK-loaded PEG-PLGA NP PEG-PLGA NP with only 20 PEG-PLGA polymers. PEG in blue and PLGA in purple. Snapshot is not to scale.

Other future work would include to analyse more parameters to further understand the induced stress by the NP on the membranes such as lipid order parameter or membrane curvature, as well as to run replicas of all simulations.

Chapter 7

Conclusions

The objective of this thesis was to provide new molecular insights into the mechanisms by which polymer-based NPs for cancer treatment self-assemble, store their therapeutic cargo and display selectivity for targeting cancer cells. This research was motivated by the lack of knowledge on the cancer cell targeting mechanisms of NPs, as well as on the rational design of NPs, since it is not well known how to predict aspects such as cargo storage location or polymer effect on micelle characetristics. This gap is limiting the clinical translation of NP-based cancer therapy.

The findings presented in this thesis provide a comprehensive understanding of these mechanisms. As a result, key parameters have been proposed to facilitate the rational design of polymer-based NPs. Additionally, to support the advancement of the computational polymer research community, the analysis tools developed during this research for the study of polymer micelles have been rigorously tested and are now publicly available in the Python-based software package PySoftK v2.0. These tools can be applied on any soft-matter simulation at different scales, having a broad applicability in the classical simulations community. Collectively, the work presented in this thesis advances the field of NP-based cancer therapy and promotes the rational design of NPs for more effective and targeted treatments.

In Chapter 3, all-atom MD simulation were used to investigate the effect of polymer topology on the self-assembly of micelles. Simulations of amphiphilic block co-polymers with the same number and type of monomer components but organised in different architectures were performed to understand how the spatial distribution of the monomer components affects the overall physicochemical characteristics of the micelles they self-assemble into. These simulations showed that polymer topology has a large effect on the micelle properties, from size to micelle stability. Furthermore, unsupervised machine learning techniques were applied to understand if polymers take location specific conformations within the micelle. It was found that polymers in all systems adopted location specific conformations, except for the micelle without a defined core-shell structure. In this case, the conformations were randomly distributed throughout the micelle. Interestingly, this micelle was the least stable one (largest amount of water in the core). These results show that polymer topology has a large impact on the physicochemical properties of micelles. Furthermore, this Chapter also provides parameters to quantify these effects, so that the same analysis can be used to study other topologies. Overall, this research shows that polymer topology is a key parameter in the rational design of polymer NP.

The methods analysis tools developed in Chapter 3 could be used to study other systems. Therefore, these methods, combined with new ones, were generalised and re-introduced in Chapter 4 within the software package PySoftK v2.0. This software is based on the original PySoftK but with a new module for the analysis of soft-matter simulations, with a special focus on polymers. The goal of this software is to provide an automated computational analysis workflow that allows scientist to study complex properties of molecular aggregates with minimal user input. Key analysis tools include a graph theory-based clustering algorithm and a ring-ring stacking analysis tool, among others. Furthermore, this software is openly available and it contributes to the standardization and replicability of the analysis of polymer simulations, which will aid the development of computational polymer science.

The same unsupervised learning method applied in Chapter 3, to study if polymers adopt location-specific conformations in small micelles (20 polymers), was applied on a 200 polymer CG PEG-PLGA NP loaded with an anti-cancer peptide named EEK in Chapter 5. This simulation revealed that peptides can be stored in two different locations within the NP, which agreed with previous literature on this type of NP [33]. Interestingly, polymers also adopted location-specific conformations within this NP, since it had a clear core-shell structure. Therefore, this seems to be a pattern within polymeric NP with a hydrophobic core and hydrophilic corona. Furthermore, it was found that these polymer conformations formed local microenvironments within the NP that allowed the preferential EEK solubilization in the cargo storage regions. This lead to two contained storage locations. Therefore, Chapter 5 demonstrates that the location dependent microenvironments that polymers form due to their conformation are key to control the drug storage locations within NP. Thus, the conformations taken by polymers are yet another key parameter for the rational design of polymeric NP with therapeutic applications. This parameter has not been previously proposed in the literature.

The first three chapters covered the molecular mechanisms underlying polymer NP formation, drug encapsulation processes, and the development of quantification methods for studying their physicochemical characteristics. In contrast, the last Chapter in this thesis, Chapter 6, employed CG MD simulations to elucidate the selectivity mechanisms of an experimentally validated PEG-PLGA NP [2] towards cancer cells. For this purpose, the NP was simulated with a model cancer and healthy membrane. These simulations, showed that the NP preferentially interacted with the lipid species DPSM, which is present in higher percentages in cancer membranes. Because of this, there were more NP-cancer cell interactions that in the NP-healthy membrane system. These interactions caused PEG polymers to attach and expand on the membrane, leaving the PLGA hydrophobic core more exposed to the environment, forcing it to collapse to shield itself from the water. This lead to drastic changes in the size and shape of the PLGA core, making it smaller and rice-shaped. Interestingly, these are physical characteristics that favour NP penetration into membranes. Furthermore, the peptides in the cancer simulation, moved with a clear trend towards the membrane. Since DPSM is present in lower percentages in healthy membranes, the polymer-lipid interactions were not as numerous in the healthy simulation, inducing less changes in the NP. The polymer-lipid interactions also caused changes on the membrane. To measure membrane disturbance, the membrane thickness was quantified. The cancer membrane reduced its thickness significantly compared to the healthy membrane, meaning that the higher amount of PEO-lipid interactions caused more stress on the membrane. These results reveal that polymer-lipid interactions may be key for the selectivity of NP towards cancer cells. Therefore, the preference of polymers for specific lipids emerges as a crucial factor for fine-tuning NP cancer cell selectivity. Additionally, this Chapter showed that NP selectivity can be studied *in silico* by examining changes in the NP and membrane physical characteristics.

Overall, the work presented in this thesis represents a significant step forward in the field of polymer-based NPs for cancer therapy. This thesis proposes new parameters and computational workflows that contribute to a deeper understanding of NP rational design. In doing so, some of the underlying mechanisms governing NP cargo encapsulation, cancer selectivity and the influence of individual polymers on micelle characteristics have been elucidated. This was achieved by a series of AA and CG simulations, coupled with the development of novel and complex analysis tools grounded in graph theory and machine learning. These methods have been integrated into the publicly accessible PySoftK v2.0 software, fostering their application within the broader computational polymer community for studying diverse polymer and NP systems. The parameters and workflows proposed in this work will advance our understanding of the rational design of polymer NP, promising more effective and targeted cancer therapies. Moreover, this research demonstrates the feasibility of assessing various NP characteristics and processes *in silico*, contributing to the growing field of *in silico* drug design. This work bridges the gap between theoretical and experimental insights, contributing to the development of greener, faster, and more efficient methods for designing cancer drugs, poised to make a substantial impact on the future of cancer treatment.

Appendix A

Supplementary Information for "Topology-Controlled Self-assembly of Amphiphilic Block Copolymers"

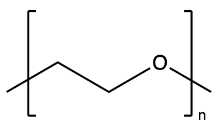
Supplementary Information for "Topology-Controlled Self-Assembly of Amphiphilic Block Copolymers"

Raquel López-Ríos de Castro,[†,‡] Robert M. Ziolek,[‡] and Christian D. Lorenz[*,‡]

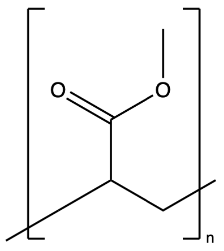
†*Department of Chemistry, King's College London, London, SE1 1DB, United Kingdom*

‡*Biological Physics and Soft Matter Group, Department of Physics, King's College London, London, WC2R 2LS, United Kingdom*

E-mail: chris.lorenz@kcl.ac.uk



(a) PEO



(b) PMA

Figure S1: (a) Molecular structure of PEO (b) Molecular structure of PMA.

Analysis of simulations All analysis was carried out with Python 3.9.5. The Python modules MDAnalysis,[1] pySoftWhere[2] and NetworkX,[3] the algorithms of UMAP[4] and HDBSCAN[5] were used to develop in-house scripts for this analysis.

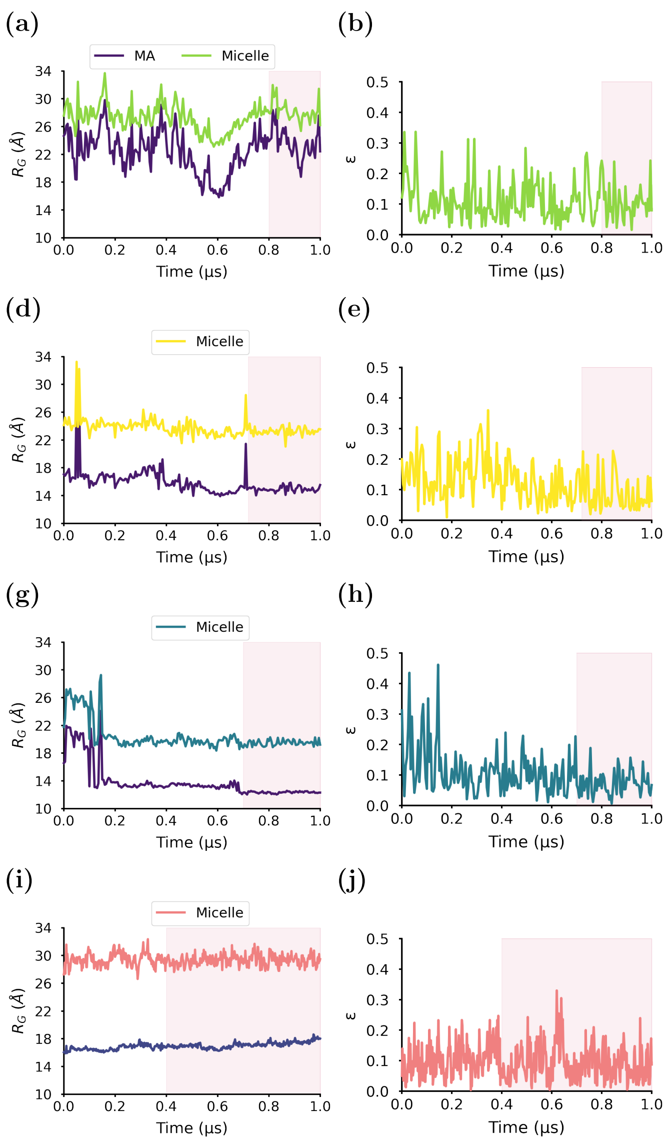


Figure S2: **Size and shape of micelles.** Plots of the R_G for the largest micelle as a function of time for the (a) MA-terminated polymer micelle, (d) EO-terminated polymer micelle, (g) ring polymer micelle and (i) diblock polymer micelle. Plots of the eccentricy of the largest micelle as a function of time for the (b) MA-terminated polymer micelle, (e) EO-terminated polymer micelle, (h) ring polymer micelle and (j) diblock polymer micelle. The time scale over which each micelle has reached equilibrium is shaded in pink.

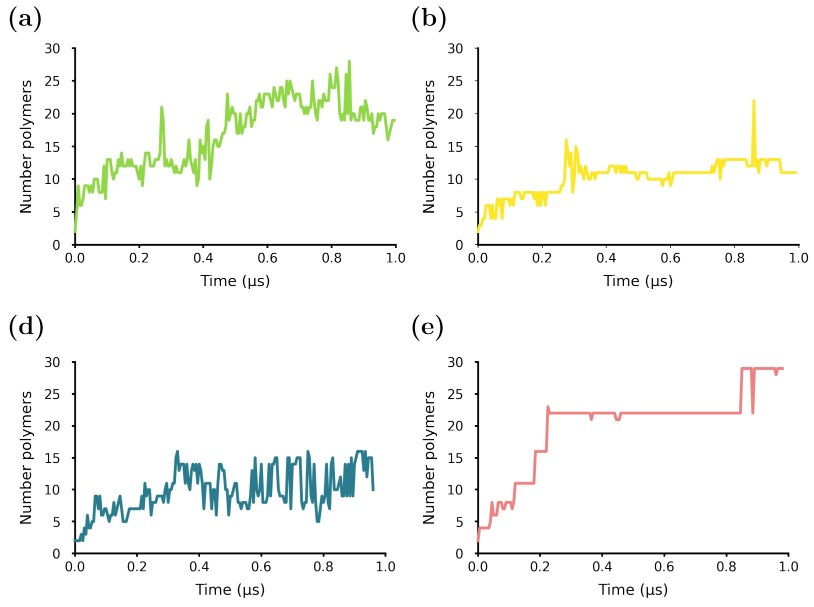


Figure S3: **Largest micelle size in 30 polymer simulation.** Plots of the number of polymers belonging to the biggest micelle over time for a) triblock ma terminated, b) triblock eo terminated, c) cyclic and d) diblock polymer topologies, for simulations with 30 polymers but same water to polymer ratio as for the simulations of 20 papers tated everywhere else in the manuscript. This proves that there is no system size effect for these systems and in fact we would expect if you had even more polymers the largest micelle would contain the same number of polymers but there would be multiple of the same size. Except for the case of the EO-MA diblock polymer (simulated in order to address a question raised by Reviewer 1), wherecase it seems that those polymers phase-separate instead of form micelles, and therefore their micelle size continues to increase

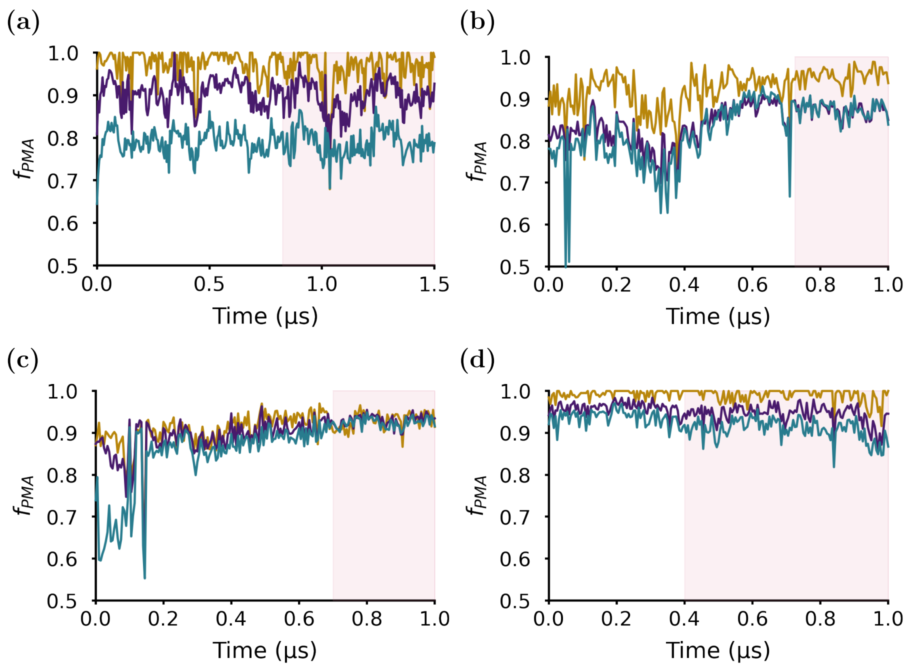


Figure S4: **Equilibration analysis** Fraction of PMA heavy atoms that are in the core with respect to all heavy atoms of the polymers species in the core (a) MA-terminated polymer micelle, (b) EO-terminated polymer micelle and (c)ring polymer micelle. The brown line represents the fraction of PMA atoms at a distance within $10\mathring{A}$ from the core, the purple line is within a distance of $15\mathring{A}$ and the blue line is a distance below the radius of gyration of the core.

The R_G of the micelle, is described by Equation (1)

$$R_\mathrm{G} = \sqrt{\frac{1}{M}\sum_{i=1}^{N} m_i (r_i - R)^2} \tag{1}$$

where M is the total mass of the body, m_i is the mass of atom i and R is the mean position of all atoms. Both, the R_G of the core of the NP and of the whole NP were calculated with the MDAnalysis function `radius_of_gyration()`.

In order to calculate the eccentricity (ϵ) of the whole and the core of the micelle, the moment of inertia around the principal axis were obtained with the MDAnalysis function `moment_of_intertia()`. The eccentricity was obtained from Equation 2

$$\epsilon = 1 - \left(\frac{I_{min}}{I_{ave}}\right) \tag{2}$$

where I_{ave} is the average across all moments of inertia and I_{min} is the moment of inertia around the x axis.

Identifying the amount of polymers clustered in each micelle was done with the Python module Networkx, which is able to study the structure of complex NetworkX. Each polymer was defined by two atoms (triblock topologies) or one atom (ring topology), if polymers were closed enough via these atoms, they were considered part of the same network, which meant they formed part of the same micelle. The atom selection chosen in all topologies were MA atoms, as these form the hydrophobic core.

The number of contacts was performed using MDAnalysis tools. First, to define the intermolecular contact distance between the main carbon backbone atom of the different monomer species and water, we used the pair radial distribution function `rdf_calc.InterRDF` between the molecules that we wanted to study. The second peak in the rdf plots was used to determined the contact distance between the specific pair, which in this case was 7 Å. Afterwards, a contact was counted if the distance between the selected monomer atoms was below the assigned contact distance. For the hydration a similar procedure was followed, but the distances was selected to be approximately 4 Å and the water atom chosen for the contacts was the water oxygen.

When the contact results were plotted, the values were normalized by dividing the number of contacts between a pair of atoms by the number of configurations used in the analysis. Then, it was divided by the mean cluster size, to account for micelles being formed by different numbers of polymer molecules. To finalize the normalization across all simulations, the number of contacts between pairs of atoms was divided by the maximum number of contacts found across all three topologies.

Intrinsic Core-Shell Interface (ICSI) Method.For micelles with an irregular internal and interfacial structures, intrinsic interface techniques can be used to investigate the interfacial structure of micelles.[6] The intrinsic density was calculated with an intrinsic core-shell interface (ICSI) method provided by the python package pySoftWhere.[2] For this method we selected the MA heavy atoms to form the ICSI as they are the principal component of the core (this information could be inferred by the contacts maps, the hydration data and the spherical density of components). The grid selected was 30 × 30. Detailed information on the working of this algorithm can be found in (Ziolek et al).[6] The ICSI equation is:

$$\tilde{\rho}(r) \equiv \left\langle \sum_i \frac{\delta[r - (r_i - \xi(\theta, \phi))]}{\bar{S}_i(r)} \right\rangle \tag{3}$$

where r_i is the r-position of atom i (of the chosen group of atoms) and $\xi(\theta, \phi)$ is the r-position of the ICSI. The average volume of the shell in which a given atom is found when using the intrinsic surface approach, $\bar{S}_i(r)$, which normalizes the intrinsic density, is given by:

$$\bar{S}_i(r) = \frac{n_i \bar{V}_{\text{box}}}{N} \tag{4}$$

where n_i is the number of points found in the shell in which atom i is found over all the clusters analyzed, $\bar{V}_{\text{box}}$ is the average volume of the simulation box, and N is the total number of random coordinates used in the normalization procedure.

Dimensionality reduction and clustering. The distances chosen as the input space to generate the two dimensional UMAP embedded data are topology specific. The goal was to find the minimum number of distances that could resume the conformational complexity adopted by the polymers. First, for the MA-terminated polymer 5 distances were selected: distance between terminal MAs, both terminal MAs to central EO, and last EO monomers to central EO. Similarly, for the EO-terminated polymer, 5 distances were also selected. Distance between terminal EOs, distance between terminal EOs and central MA, and distance between the last MA monomers and the central MA. Finally, for the ring polymer only 4 were needed. Middle MA to middle EO, terminal MA to middle EO and the distance between the last MA monomers. The UMAP embedded output was latter cluster with HDBSCAN, Table S1 shows the UMAP and HDBSCAN parameters chosen for each topology.

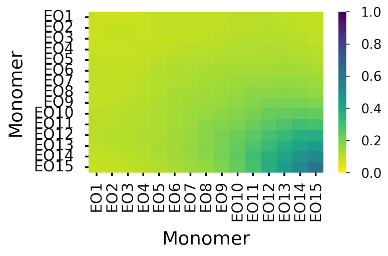
The intrinsic density of the UMAP clusters was calculated in the same as for the overall micelle intrinsic density. But instead of using all polymers, only the ones belonging to the specific cluster density being calculated were used.

Table S1: UMAP (*n_neighbours*) and HDBSCAN (*min_cluster_size* and *cluster_selection_epsilon*) parameters

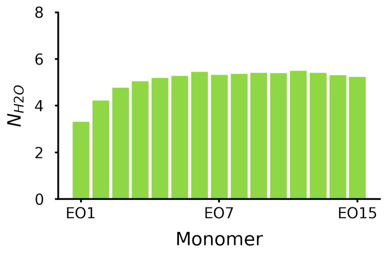
	MA-terminated	EO-terminated	Ring	Diblock
n_neighbours	5	6	8	12
min_cluster_size	25	55	34	55
cluster_selection_epsilon	0.8	1	1.1	0.85

Table S2: Micelle equilibration time and the mean micelle size (Å).

	MA-terminated	EO-terminated	Ring	Diblock
Equilibration time (μs)	0.8	0.72	0.6	0.4
Mean micelle size	18.6 ± 0.6	13.7 ± 0.8	10.9 ± 0.4	20.0 ± 0.2

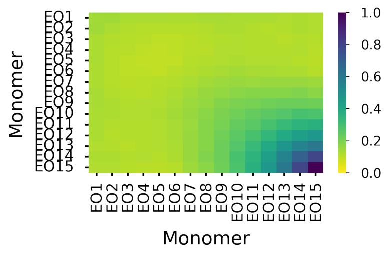


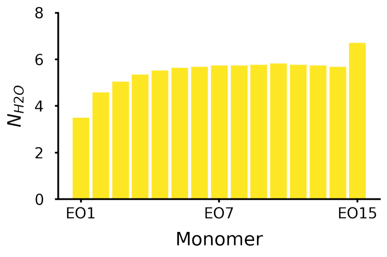
(a) EO-EO contacts



(b) EO hydration

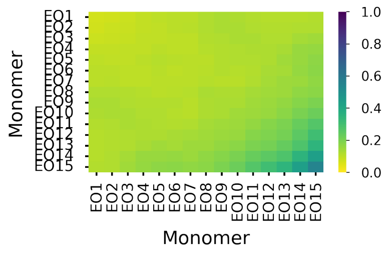
Figure S5: (a) Normalised intermolecular contact maps between the EO monomers and (b) normalised hydration of the oxygen atoms of the EO monomers for the MA-terminated polymer micelle.

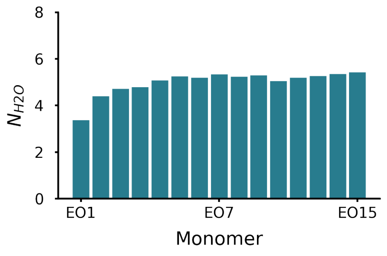




(a) EO-EO contacts (b) EO hydration

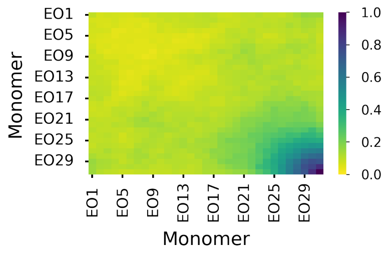
Figure S6: (a) Normalised intermolecular contact maps between the EO monomers and (b) normalised hydration of the oxygen atoms of the EO monomers for the EO-terminated polymer micelle.

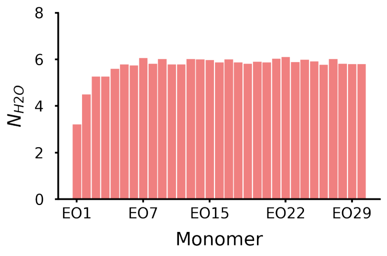




(a) EO-EO contacts (b) EO hydration

Figure S7: (a) Normalised intermolecular contact maps between the EO monomers and (b) normalised hydration of the oxygen atoms of the EO monomers for the ring polymer micelle.





(a) EO-EO contacts (b) EO hydration

Figure S8: (a) Normalised intermolecular contact maps between the EO monomers and (b) normalised hydration of the oxygen atoms of the EO monomers for the diblock polymer micelle.

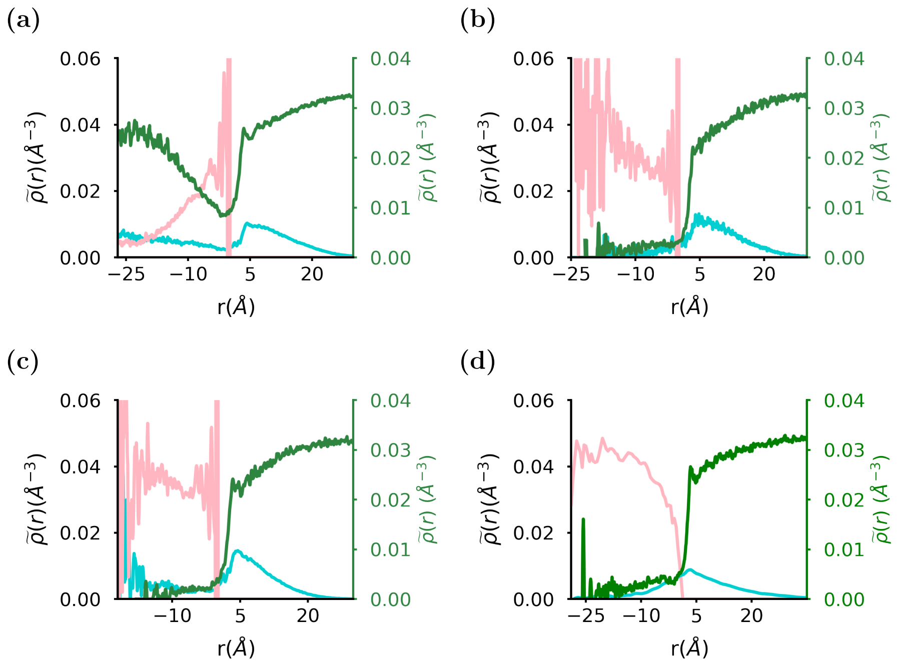


Figure S9: **Intrinsic density of micelle components.** Intrinsic density of the (a) MA-terminated polymer micelle, (b) EO-terminated polymer micelle and (c)ring polymer micelle. The intrinsic density of the MA monomers is displayed in pink, EO in blue and water in dark green.

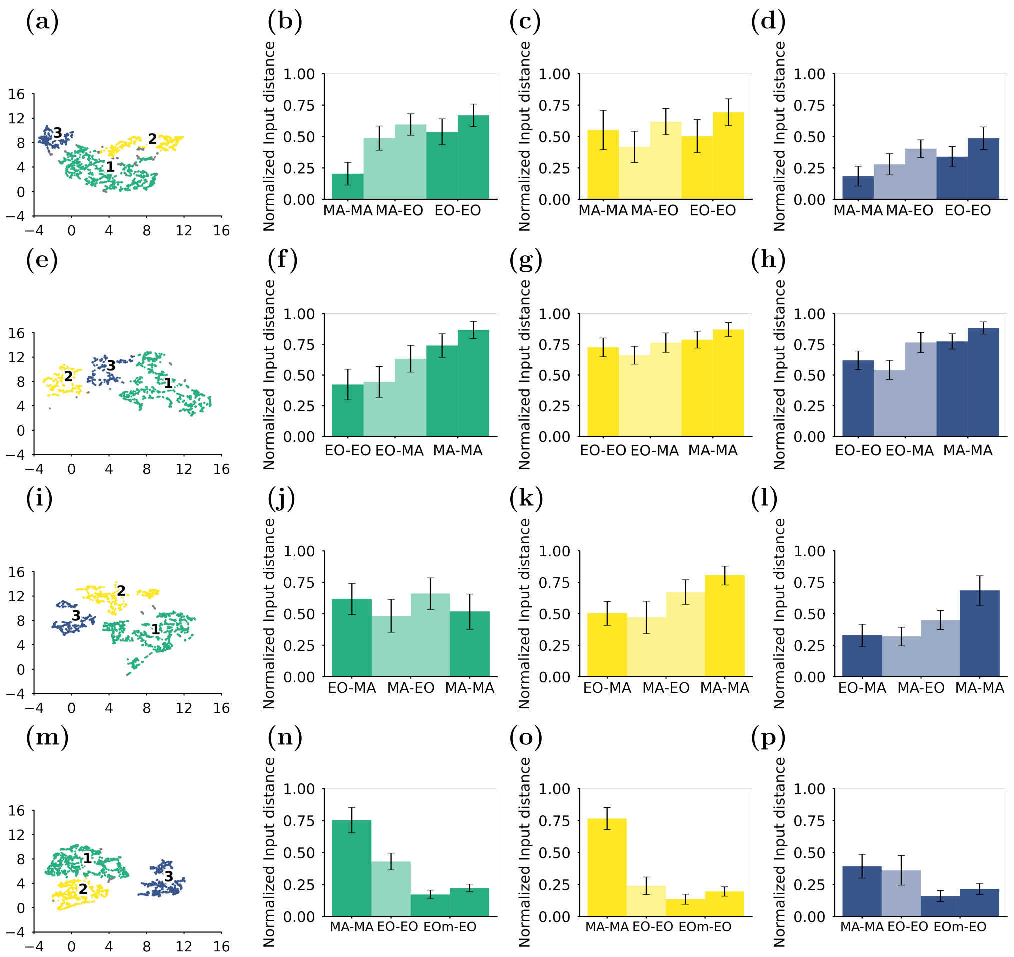


Figure S10: **UMAP embedded space and average cluster distances** UMAP embedded space clustered by HDBSCAN of (a) linear triblock MA-EO-MA polymers, (e) linear triblock EO-MA-EO polymers, (i) ring MA-EO polymers and (m) diblock polymer. Histograms of the average distances of each cluster. Cluster 1 for (b) linear triblock MA-EO-MA polymers, (f) linear triblock EO-MA-EO polymers, (j) ring MA-EO polymers and (j) diblock polymer. Cluster 2 for (c) linear triblock MA-EO-MA polymers, (g) linear triblock EO-MA-EO polymers, (k) ring MA-EO polymers and (o) diblock polymer. Cluster 3 (d) linear triblock MA-EO-MA polymers, (h) linear triblock EO-MA-EO polymers, (l) ring MA-EO polymers and (p) diblock polymer.

References

(1) Michaud-Agrawal, N.; Denning, E. J.; Woolf, T. B.; Beckstein, O. MDAnalysis: a toolkit for the analysis of molecular dynamics simulations. *J. Comput. Chem.* **2011**, *32*, 2319–2327.

(2) `https://pysoftwhere.readthedocs.io/en/latest/index.html`.

(3) Hagberg, A. A.; Schult, D. A.; Swart, P. J. Exploring Network Structure, Dynamics, and Function using NetworkX. Proceedings of the 7th Python in Science Conference. Pasadena, CA USA, 2008; pp 11 – 15.

(4) McInnes, L.; Healy, J.; Melville, J. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426* **2018**,

(5) McInnes, L.; Healy, J.; Astels, S. hdbscan: Hierarchical density based clustering. *J. Open Source Softw.* **2017**, *2*, 205.

(6) Ziolek, R. M.; Smith, P.; Pink, D. L.; Dreiss, C. A.; Lorenz, C. D. Unsupervised learning unravels the structure of four-arm and linear block copolymer micelles. *Macromolecules* **2021**, *54*, 3755–3768.

Bibliography

[1] Rebecca L Siegel, Kimberly D Miller, Nikita Sandeep Wagle, and Ahmedin Jemal. Cancer statistics, 2023. *Ca Cancer J Clin*, 73(1):17–48, 2023.

[2] Charles H Chen, Yu-Han Liu, Arvin Eskandari, Jenisha Ghimire, Leon Chien-Wei Lin, Zih-Syun Fang, William C Wimley, Jakob P Ulmschneider, Kogularamanan Suntharalingam, Che-Ming Jack Hu, et al. Integrated design of a membrane-lytic peptide-based intravenous nanotherapeutic suppresses triple-negative breast cancer. *Advanced Science*, 9(13):2105506, 2022.

[3] David W Hoskin and Ayyalusamy Ramamoorthy. Studies on anticancer activities of antimicrobial peptides. *Biochimica et Biophysica Acta (BBA)-Biomembranes*, 1778(2):357–375, 2008.

[4] John S Bertram. The molecular biology of cancer. *Molecular aspects of medicine*, 21(6):167–223, 2000.

[5] Vincent T DeVita and Edward Chu. A history of cancer chemotherapy. *Cancer research*, 68(21):8643–8653, 2008.

[6] Joseph Leiter, BJ Abbott, and Saul A Schepartz. Screening data from the cancer chemotherapy national service center screening laboratories. xxviii. *Cancer research*, 25(9 Part 2):1626–1769, 1965.

[7] World Health Organisation (WHO). Who statistical information system. http://www.who.int/whosis, November 2020.

[8] National Cancer Institute (NCI). Cancer statistics. https://www.cancer.gov/about-cancer/understanding/statistics, September 2020.

[9] Jon Zugazagoitia, Cristiano Guedes, Santiago Ponce, Irene Ferrer, Sonia Molina-Pinelo, and Luis Paz-Ares. Current challenges in cancer treatment. *Clinical therapeutics*, 38(7):1551–1566, 2016.

[10] Gaorav P Gupta and Joan Massagué. Cancer metastasis: building a framework. *Cell*, 127(4):679–695, 2006.

[11] Hyuna Sung, Jacques Ferlay, Rebecca L Siegel, Mathieu Laversanne, Isabelle Soerjomataram, Ahmedin Jemal, and Freddie Bray. Global cancer statistics 2020: Globocan estimates of incidence and mortality worldwide for 36 cancers in 185 countries. *CA: a cancer journal for clinicians*, 71(3):209–249, 2021.

[12] Manfred Schuster, Andreas Nechansky, and Ralf Kircheis. Cancer immunotherapy. *Biotechnology Journal: Healthcare Nutrition Technology*, 1(2):138–147, 2006.

[13] Rajamanickam Baskar, Kuo Ann Lee, Richard Yeo, and Kheng-Wei Yeoh. Cancer and radiation therapy: current advances and future directions. *International journal of medical sciences*, 9(3):193, 2012.

[14] Vincent Grégoire, Matthias Guckenberger, Karin Haustermans, Jan JW Lagendijk, Cynthia Ménard, Richard Pötter, Ben J Slotman, Kari Tanderup, Daniela Thorwarth, Marcel Van Herk, et al. Image guidance in radiation therapy for better cure of cancer. *Molecular oncology*, 14(7):1470–1491, 2020.

[15] Peter Nygren. What is cancer chemotherapy? *Acta Oncologica*, 40(2-3):166–174, 2001.

[16] Eskinder Ayalew Sisay, Ephrem Engidawork, Teshager Aklilu Yesuf, and Ezra Belay Ketema. Drug related problems in chemotherapy of cancer patients. *J Cancer Sci Ther*, 7(2):55–59, 2015.

[17] Hai Wang, Ying Zhao, Yan Wu, Yu-lin Hu, Kaihui Nan, Guangjun Nie, and Hao Chen. Enhanced anti-tumor efficacy by co-delivery of doxorubicin and paclitaxel with amphiphilic methoxy peg-plga copolymer nanoparticles. *Biomaterials*, 32(32):8281–8290, 2011.

[18] Sina Taefehshokr, Aram Parhizkar, Shima Hayati, Morteza Mousapour, Amin Mahmoudpour, Liliane Eleid, Dara Rahmanpour, Sahand Fattahi, Hadi Shabani, and Nima Taefehshokr. Cancer immunotherapy: Challenges and limitations. *Pathology-Research and Practice*, 229:153723, 2022.

[19] Suphiya Parveen and Sanjeeb K Sahoo. Polymeric nanoparticles for cancer therapy. *Journal of drug targeting*, 16(2):108–123, 2008.

[20] Ranjita Misra, Sarbari Acharya, and Sanjeeb K Sahoo. Cancer nanotechnology: application of nanotechnology in cancer therapy. *Drug discovery today*, 15(19-20):842–850, 2010.

[21] Jessica A Kemp and Young Jik Kwon. Cancer nanotechnology: Current status and perspectives. *Nano convergence*, 8(1):34, 2021.

[22] Shuming Nie, Yun Xing, Gloria J Kim, and Jonathan W Simons. Nanotechnology applications in cancer. *Annu. Rev. Biomed. Eng.*, 9:257–288, 2007.

[23] Mauro Ferrari. Cancer nanotechnology: opportunities and challenges. *Nature reviews cancer*, 5(3):161–171, 2005.

[24] Yechezkel Chezy Barenholz. Doxil®—the first fda-approved nano-drug: Lessons learned. *Journal of controlled release*, 160(2):117–134, 2012.

[25] Vladimir P Torchilin. Recent advances with liposomes as pharmaceutical carriers. *Nature reviews Drug discovery*, 4(2):145–160, 2005.

[26] SA Wissing, Oliver Kayser, and RH Müller. Solid lipid nanoparticles for parenteral drug delivery. *Advanced drug delivery reviews*, 56(9):1257–1272, 2004.

[27] Farha Masood. Polymeric nanoparticles for targeted drug delivery system for cancer therapy. *Materials Science and Engineering: C*, 60:569–578, 2016.

[28] Nazila Kamaly, Basit Yameen, Jun Wu, and Omid C Farokhzad. Degradable controlled-release polymers and polymeric nanoparticles: mechanisms of controlling drug release. *Chemical reviews*, 116(4):2602–2663, 2016.

[29] Pierluigi Stipa, Stefania Marano, Roberta Galeazzi, Cristina Minnelli, Giovanna Mobbili, and Emiliano Laudadio. Prediction of drug-carrier interactions of pla and plga drug-loaded nanoparticles by molecular dynamics simulations. *European Polymer Journal*, 147:110292, 2021.

[30] Sovan Lal Pal, Utpal Jana, Prabal Kumar Manna, Guru Prasad Mohanta, and R Manavalan. Nanoparticle: An overview of preparation and characterization. *Journal of applied pharmaceutical science*, (Issue):228–234, 2011.

[31] Carina IC Crucho and Maria Teresa Barros. Formulation of functionalized plga polymeric nanoparticles for targeted drug delivery. *Polymer*, 68:41–46, 2015.

[32] Hiroshi Maeda, Hideaki Nakamura, and Jun Fang. The epr effect for macromolecular drug delivery to solid tumors: Improvement of tumor uptake, lowering of systemic toxicity, and distinct tumor imaging in vivo. *Advanced drug delivery reviews*, 65(1):71–79, 2013.

[33] Jianwei Guo, Xiaoling Gao, Lina Su, Huimin Xia, Guangzhi Gu, Zhiqing Pang, Xinguo Jiang, Lei Yao, Jun Chen, and Hongzhuan Chen. Aptamer-functionalized peg–plga nanoparticles for enhanced anti-glioma drug delivery. *Biomaterials*, 32(31):8010–8020, 2011.

[34] Andre Nel, Erkki Ruoslahti, and Huan Meng. New insights into "permeability" as in the enhanced permeability and retention effect of cancer nanotherapeutics, 2017.

[35] Shrey Sindhwani, Abdullah Muhammad Syed, Jessica Ngai, Benjamin R Kingston, Laura Maiorino, Jeremy Rothschild, Presley MacMillan, Yuwei Zhang, Netra Unni Rajesh, Tran Hoang, et al. The entry of nanoparticles into solid tumours. *Nature materials*, 19(5):566–575, 2020.

[36] Benjamin R Kingston, Zachary P Lin, Ben Ouyang, Presley MacMillan, Jessica Ngai, Abdullah Muhammad Syed, Shrey Sindhwani, and Warren CW Chan. Specific endothelial cells govern nanoparticle entry into solid tumors. *ACS nano*, 2021.

[37] Haizheng Zhao and Lin Yue Lanry Yung. Selectivity of folate conjugated polymer micelles against different tumor cells. *International journal of pharmaceutics*, 349(1-2):256–268, 2008.

[38] Markus R Wenk. The emerging field of lipidomics. *Nature reviews Drug discovery*, 4(7):594–610, 2005.

[39] Furong Yan, Hong Zhao, and Yiming Zeng. Lipidomics: a promising cancer biomarker. *Clinical and translational medicine*, 7(1):1–3, 2018.

[40] Diana Gaspar, A Salomé Veiga, and Miguel ARB Castanho. From antimicrobial to anticancer peptides. a review. *Frontiers in microbiology*, 4:294, 2013.

[41] Carola Leuschner and William Hansel. Membrane disrupting lytic peptides for cancer treatments. *Current pharmaceutical design*, 10(19):2299–2310, 2004.

[42] Frank Schweizer. Cationic amphiphilic peptides with cancer-selective toxicity. *European journal of pharmacology*, 625(1-3):190–194, 2009.

[43] Siu-Chiu Chan, LING Hui, and Hueih Min Chen. Enhancement of the cytolytic effect of anti-bacterial cecropin by the microvilli of cancer cells. *Anticancer research*, 18(6A):4467–4474, 1998.

[44] Wenjia Zhang, Joseph M Metzger, Benjamin J Hackel, Frank S Bates, and Timothy P Lodge. Influence of the headgroup on the interaction of poly (ethylene oxide)-poly (propylene oxide) block copolymers with lipid bilayers. *The Journal of Physical Chemistry B*, 124(12):2417–2424, 2020.

[45] Stefan Wilhelm, Anthony J Tavares, Qin Dai, Seiichi Ohta, Julie Audet, Harold F Dvorak, and Warren CW Chan. Analysis of nanoparticle delivery to tumours. *Nature reviews materials*, 1(5):1–12, 2016.

[46] Yu Dang and Jianjun Guan. Nanoparticle-based drug delivery systems for cancer therapy. *Smart Materials in Medicine*, 1:10–19, 2020.

[47] Elvin Blanco, Haifa Shen, and Mauro Ferrari. Principles of nanoparticle design for overcoming biological barriers to drug delivery. *Nature biotechnology*, 33(9):941–951, 2015.

[48] Cheng Zhu, Xuejie Zhou, Ziteng Liu, Hongwei Chen, Hongfeng Wu, Xiao Yang, Xiangdong Zhu, Jing Ma, and Hao Dong. The morphology of hydroxyapatite nanoparticles regulates cargo recognition in clathrin-mediated endocytosis. *Frontiers in molecular biosciences*, 8, 2021.

[49] Jiaqi Lin, Lei Miao, Grace Zhong, Chih-Hsin Lin, Roozbeh Dargazangy, and Alfredo Alexander-Katz. Understanding the synergistic effect of physicochemical properties of nanoparticles and their cellular entry pathways. *Communications Biology*, 3(1):1–10, 2020.

[50] Phu K Tang, Anjela Manandhar, William Hu, Myungshim Kang, and Sharon M Loverde. The interaction of supramolecular anticancer drug amphiphiles with phospholipid membranes. *Nanoscale Advances*, 2020.

[51] Hong-ming Ding, Wen-de Tian, and Yu-qiang Ma. Designing nanoparticle translocation through membranes by computer simulations. *ACS nano*, 6(2):1230–1238, 2012.

[52] Sulin Zhang, Ju Li, George Lykotrafitis, Gang Bao, and Subra Suresh. Size-dependent endocytosis of nanoparticles. *Advanced materials*, 21(4):419–424, 2009.

[53] Wen Jiang, Betty YS Kim, James T Rutka, and Warren CW Chan. Nanoparticle-mediated cellular response is size-dependent. *Nature nanotechnology*, 3(3):145–150, 2008.

[54] Natalia Wilkosz, Grzegorz Łazarski, Lubomir Kovacik, Patrycja Gargas, Maria Nowakowska, Dorota Jamroz, and Mariusz Kepczynski. Molecular insight into drug-loading capacity of peg–plga nanoparticles for itraconazole. *The Journal of Physical Chemistry B*, 122(28):7080–7090, 2018.

[55] Murugesan Sathiya Deepika, Ramar Thangam, Shenbagamoorthy Sundarraj, Thankaraj Salammal Sheena, Srinivasan Sivasubramanian, Jeganathan Kulandaivel, and Ramasamy Thirumurugan. Co-delivery of diverse therapeutic compounds using peg–plga nanoparticle cargo against drug-resistant bacteria: an improved anti-biofilm strategy. *ACS Applied Bio Materials*, 3(1):385–399, 2019.

[56] Lu-yi Huang, You-sheng Yu, Xiang Lu, Hong-ming Ding, and Yu-qiang Ma. Designing a nanoparticle-containing polymeric substrate for detecting cancer cells by computer simulations. *Nanoscale*, 11(5):2170–2178, 2019.

[57] Lihong Liu, Kaijin Xu, Huaying Wang, PK Jeremy Tan, Weimin Fan, Subbu S Venkatraman, Lanjuan Li, and Yi-Yan Yang. Self-assembled cationic peptide nanoparticles as an efficient antimicrobial agent. *Nature nanotechnology*, 4(7):457–463, 2009.

[58] Mário R Felício, Osmar N Silva, Sônia Gonçalves, Nuno C Santos, and Octávio L Franco. Peptides with dual antimicrobial and anticancer activities. *Frontiers in chemistry*, 5:5, 2017.

[59] Maria Magana, Muthuirulan Pushpanathan, Ana L Santos, Leon Leanse, Michael Fernandez, Anastasios Ioannidis, Marc A Giulianotti, Yiorgos Apidianakis, Steven Bradfute, Andrew L Ferguson, et al. The value of antimicrobial peptides in the age of resistance. *The Lancet Infectious Diseases*, 2020.

[60] Maja Mihajlovic and Themis Lazaridis. Antimicrobial peptides in toroidal and cylindrical pores. *Biochimica et Biophysica Acta (BBA)-Biomembranes*, 1798(8):1485–1493, 2010.

[61] Hui Sun, Yuanxiu Hong, Yuejing Xi, Yijie Zou, Jingyi Gao, and Jianzhong Du. Synthesis, self-assembly, and biomedical applications of antimicrobial peptide–polymer conjugates. *Biomacromolecules*, 19(6):1701–1720, 2018.

[62] Marcin Makowski, Ítala C Silva, Constança Pais do Amaral, Sónia Gonçalves, and Nuno C Santos. Advances in lipid and metal nanoparticles for antimicrobial peptide delivery. *Pharmaceutics*, 11(11):588, 2019.

[63] Michael C Hacker, Jan Krieghoff, and Antonios G Mikos. Synthetic polymers. In *Principles of regenerative medicine*, pages 559–590. Elsevier, 2019.

[64] Rolf Mülhaupt. Hermann staudinger and the origin of macromolecular chemistry. *Angewandte Chemie International Edition*, 43(9):1054–1063, 2004.

[65] LG Griffith. Polymeric biomaterials. *Acta materialia*, 48(1):263–277, 2000.

[66] Yiyong Mai and Adi Eisenberg. Self-assembly of block copolymers. *Chemical Society Reviews*, 41(18):5969–5985, 2012.

[67] Chase B Thompson and LaShanda TJ Korley. 100th anniversary of macromolecular science viewpoint: engineering supramolecular materials for responsive applications—design and functionality. *ACS Macro Letters*, 9(9):1198–1216, 2020.

[68] Timothy P Lodge. Block copolymers: past successes and future challenges. *Macromolecular chemistry and physics*, 204(2):265–273, 2003.

[69] Raquel Lopez-Rios De Castro, Robert Ziolek, and Chris Lorenz. Topology-controlled self-assembly of amphiphilic block copolymers. 2023.

[70] Yi Wang and Scott M Grayson. Approaches for the preparation of non-linear amphiphilic polymers and their applications to drug delivery. *Advanced drug delivery reviews*, 64(9):852–865, 2012.

[71] Ludwik Leibler. Theory of microphase separation in block copolymers. *Macromolecules*, 13(6):1602–1617, 1980.

[72] Fikret Aydin, Xiaolei Chu, Geetartha Uppaladadium, David Devore, Ritu Goyal, N Sanjeeva Murthy, Zheng Zhang, Joachim Kohn, and Meenakshi Dutt. Self-assembly and critical aggregation concentration measurements of aba triblock copolymers with varying b block types: model development, prediction, and validation. *The journal of physical chemistry B*, 120(15):3666–3676, 2016.

[73] Bartosz A Grzybowski, Christopher E Wilmer, Jiwon Kim, Kevin P Browne, and Kyle JM Bishop. Self-assembly: from crystals to cells. *Soft Matter*, 5(6):1110–1128, 2009.

[74] G Kocak, CANSEL Tuncer, and VJPC Bütün. ph-responsive polymers. *Polymer Chemistry*, 8(1):144–176, 2017.

[75] Takahiro Sato and Rintaro Takahashi. Competition between the micellization and the liquid–liquid phase separation in amphiphilic block copolymer solutions. *Polymer Journal*, 49(2):273–277, 2017.

[76] Saeed Najafi, James McCarty, Kris T Delaney, Glenn H Fredrickson, and Joan-Emma Shea. Field-theoretic simulation method to study the liquid–liquid phase separation of polymers. In *Phase-Separated Biomolecular Condensates: Methods and Protocols*, pages 37–49. Springer, 2022.

[77] Ghaleb A Husseini and William G Pitt. Micelles and nanoparticles for ultrasonic drug and gene delivery. *Advanced drug delivery reviews*, 60(10):1137–1152, 2008.

[78] M Ghezzi, S Pescina, C Padula, P Santi, E Del Favero, L Cantù, and S Nicoli. Polymeric micelles in drug delivery: An insight of the techniques for their characterization and assessment in biorelevant conditions. *Journal of Controlled Release*, 332:312–336, 2021.

[79] Jiaxiao Xue, Zhou Guan, Jiaping Lin, Chunhua Cai, Wenjie Zhang, and Xinquan Jiang. Cellular internalization of rod-like nanoparticles with various surface patterns: Novel entry pathway and controllable uptake capacity. *small*, 13(24):1604214, 2017.

[80] Jacob N Israelachvili. *Intermolecular and surface forces*. Academic press, 2011.

[81] Adam Blanazs, Steven P Armes, and Anthony J Ryan. Self-assembled block copolymer aggregates: from micelles to vesicles and their biological applications. *Macromolecular rapid communications*, 30(4-5):267–277, 2009.

[82] J Andrew McCammon, Bruce R Gelin, and Martin Karplus. Dynamics of folded proteins. *Nature*, 267(5612):585–590, 1977.

[83] Martin Karplus and J Andrew McCammon. Molecular dynamics simulations of biomolecules. *Nature structural biology*, 9(9):646–652, 2002.

[84] Jacob D Durrant and J Andrew McCammon. Molecular dynamics simulations and drug discovery. *BMC biology*, 9(1):1–9, 2011.

[85] Michael L Klein and Wataru Shinoda. Large-scale molecular dynamics simulations of self-assembling systems. *Science*, 321(5890):798–800, 2008.

[86] Soumil Y Joshi and Sanket A Deshmukh. A review of advancements in coarse-grained molecular dynamics simulations. *Molecular Simulation*, 47(10-11):786–803, 2021.

[87] Margarita Valero and Cécile A Dreiss. Growth, shrinking, and breaking of pluronic micelles in the presence of drugs and/or β-cyclodextrin, a study by small-angle neutron scattering and fluorescence spectroscopy. *Langmuir*, 26(13):10561–10571, 2010.

[88] Jesús Del Barrio, Luis Oriol, Carlos Sanchez, José Luis Serrano, Aurelie Di Cicco, Patrick Keller, and Min-Hui Li. Self-assembly of linear- dendritic diblock copolymers: from nanofibers to polymersomes. *Journal of the American Chemical Society*, 132(11):3762–3769, 2010.

[89] Jiyuan Yang, Kuangshi Wu, Cestmír Konák, and Jindrich Kopecek. Dynamic light scattering study of self-assembly of hpma hybrid graft copolymers. *Biomacromolecules*, 9(2):510–517, 2008.

[90] Jiri Brus, Alexander Zhigunov, Jiri Czernek, Libor Kobera, Mariusz Uchman, and Pavel Matejicek. Control over the self-assembly and dynamics of metallacarborane nanorotors by the nature of the polymer matrix: A solid-state nmr study. *Macromolecules*, 47(18):6343–6354, 2014.

[91] Pim WJM Frederix, Ilias Patmanidis, and Siewert J Marrink. Molecular simulations of self-assembling bio-inspired supramolecular systems and their connection to experiments. *Chemical Society Reviews*, 47(10):3470–3489, 2018.

[92] Goundla Srinivas, Dennis E Discher, and Michael L Klein. Self-assembly and properties of diblock copolymers by coarse-grain molecular dynamics. *Nature materials*, 3(9):638–644, 2004.

[93] Majid Jafari, Farahnoosh Doustdar, and Faramarz Mehrnejad. Molecular self-assembly strategy for encapsulation of an amphipathic α-helical antimicrobial peptide into the different polymeric and copolymeric nanoparticles. *Journal of chemical information and modeling*, 59(1):550–563, 2018.

[94] Juhae Park, Vikram Thapar, Yeojin Choe, Luis Adrian Padilla Salas, Abelardo Ramírez-Hernández, Juan J de Pablo, and Su-Mi Hur. Coarse-

grained simulation of bottlebrush: From single-chain properties to self-assembly. *ACS Macro Letters*, 11(9):1167–1173, 2022.

[95] Alejandro Santana-Bonilla, Raquel Lopez-Rios de Castro, Peike Sun, Robert M Ziolek, and Christian D Lorenz. Modular software for generating and modeling diverse polymer databases. *Journal of Chemical Information and Modeling*, 2023.

[96] Makoto YABE, Kazuki MORI, Kazuyoshi UEDA, and Minoru TAKEDA. Development of polypargen software to facilitate the determination of molecular dynamics simulation parameters for polymers. *Journal of Computer Chemistry, Japan-International Edition*, 5:2018–0034, 2019.

[97] Xiaoli Yan and Santanu Chaudhuri. An interactive polymer building toolkit for molecular dynamics simulations: Polymaps. *arXiv preprint arXiv:2204.14218*, 2022.

[98] Peter T Cummings, Clare MCabe, Christopher R Iacovella, Akos Ledeczi, Eric Jankowski, Arthi Jayaraman, Jeremy C Palmer, Edward J Maginn, Sharon C Glotzer, Joshua A Anderson, et al. Openâ€ source molecular modeling software in chemical engineering focusing on the molecular simulation design framework. *AIChE Journal*, 67(3), 2021.

[99] Alexander G Demidov, B Lakshitha A Perera, Michael E Fortunato, Sibo Lin, and Coray M Colina. Update 1.1 to “pysimm: a python package for simulation of molecular systems”,(pii: S2352711016300395). *SoftwareX*, 15:100749, 2021.

[100] Yoshihiro Hayashi, Junichiro Shiomi, Junko Morikawa, and Ryo Yoshida. Radonpy: automated physical property calculation using all-atom classical molecular dynamics simulations for polymer informatics. *npj Computational Materials*, 8(1):222, 2022.

[101] Yeol Kyo Choi, Sang-Jun Park, Soohyung Park, Seonghoon Kim, Nathan R Kern, Jumin Lee, and Wonpil Im. Charmm-gui polymer builder for model-

ing and simulation of synthetic polymers. *Journal of chemical theory and computation*, 17(4):2431–2443, 2021.

[102] Harikrishna Sahu, Kuan-Hsuan Shen, Joseph H Montoya, Huan Tran, and Rampi Ramprasad. Polymer structure predictor (psp): a python toolkit for predicting atomic-level structural models for a range of polymer geometries. *Journal of Chemical Theory and Computation*, 18(4):2737–2748, 2022.

[103] Pik Kwan Lo, Pierre Karam, Faisal A Aldaye, Christopher K McLaughlin, Graham D Hamblin, Gonzalo Cosa, and Hanadi F Sleiman. Loading and selective release of cargo in dna nanotubes with longitudinal variation. *Nature chemistry*, 2(4):319–328, 2010.

[104] Andrew A Herzing, Masashi Watanabe, Jennifer K Edwards, Marco Conte, Zi-Rong Tang, Graham J Hutchings, and Christopher J Kiely. Energy dispersive x-ray spectroscopy of bimetallic nanoparticles in an aberration corrected scanning transmission electron microscope. *Faraday discussions*, 138:337–351, 2008.

[105] Shiao-Wen Tsai, Yi-Yun Chen, and Jiunn-Woei Liaw. Compound cellular imaging of laser scanning confocal microscopy by using gold nanoparticles and dyes. *Sensors*, 8(4):2306–2316, 2008.

[106] Georgios Pyrgiotakis, Christoph O Blattmann, and Philip Demokritou. Real-time nanoparticle–cell interactions in physiological media by atomic force microscopy. *ACS sustainable chemistry & engineering*, 2(7):1681–1690, 2014.

[107] Nafiseh Farhadian, Malihe Samadi Kazemi, Fatemeh Moosavi Baigi, and Mehdi Khalaj. Molecular dynamics simulation of drug delivery across the cell membrane by applying gold nanoparticle carrier: Flutamide as hydrophobic and glutathione as hydrophilic drugs as the case studies. *Journal of Molecular Graphics and Modelling*, 116:108271, 2022.

[108] Liuyang Zhang, Yiping Zhao, and Xianqiao Wang. Nanoparticle-mediated mechanical destruction of cell membranes: a coarse-grained molecular dynamics study. *ACS applied materials & interfaces*, 9(32):26665–26673, 2017.

[109] Yang Li, Xin Chen, and Ning Gu. Computational investigation of interaction between nanoparticles and membranes: hydrophobic/hydrophilic effect. *The Journal of Physical Chemistry B*, 112(51):16647–16653, 2008.

[110] Mitradip Das, Udaya Dahal, Oluwaseun Mesele, Dongyue Liang, and Qiang Cui. Molecular dynamics simulation of interaction between functionalized nanoparticles with lipid membranes: Analysis of coarse-grained models. *The Journal of Physical Chemistry B*, 123(49):10547–10561, 2019.

[111] Sereina Riniker. Fixed-charge atomistic force fields for molecular dynamics simulations in the condensed phase: An overview. *Journal of chemical information and modeling*, 58(3):565–578, 2018.

[112] Mark E Tuckerman. Ab initio molecular dynamics: basic concepts, current trends and novel applications. *Journal of Physics: Condensed Matter*, 14(50):R1297, 2002.

[113] Zhongyue Yang, Rimsha Mehmood, Mengyi Wang, Helena W Qi, Adam H Steeves, and Heather J Kulik. Revealing quantum mechanical effects in enzyme catalysis with large-scale electronic structure simulation. *Reaction Chemistry & Engineering*, 4(2):298–315, 2019.

[114] Hanning Chen, Mark A Ratner, and George C Schatz. Qm/mm study of photoinduced reduction of a tetrahedral ag20+ cluster by a ag atom. *The Journal of Physical Chemistry C*, 118(4):1755–1762, 2014.

[115] Sarah Lee, Alan Tran, Matthew Allsopp, Joseph B Lim, Jérôme Hénin, and Jeffery B Klauda. Charmm36 united atom chain model for lipids and surfactants. *The journal of physical chemistry B*, 118(2):547–556, 2014.

[116] Jérôme Hénin, Wataru Shinoda, and Michael L Klein. United-atom acyl chains for charmm phospholipids. *The Journal of Physical Chemistry B*, 112(23):7008–7015, 2008.

[117] Srinivas Ramachandran, Pradeep Kota, Feng Ding, and Nikolay V Dokholyan. Automated minimization of steric clashes in protein structures. *Proteins: Structure, Function, and Bioinformatics*, 79(1):261–270, 2011.

[118] Emilia L Wu, Xi Cheng, Sunhwan Jo, Huan Rui, Kevin C Song, Eder M Dávila-Contreras, Yifei Qi, Jumin Lee, Viviana Monje-Galvan, Richard M Venable, et al. Charmm-gui membrane builder toward realistic biological membrane simulations, 2014.

[119] Yifei Qi, Helgi I Ingólfsson, Xi Cheng, Jumin Lee, Siewert J Marrink, and Wonpil Im. Charmm-gui martini maker for coarse-grained simulations with the martini force field. *Journal of chemical theory and computation*, 11(9):4486–4494, 2015.

[120] Abhinandan Jain, In-Hee Park, and Nagarajan Vaidehi. Equipartition principle for internal coordinate molecular dynamics. *Journal of chemical theory and computation*, 8(8):2581–2587, 2012.

[121] JS Rowlinson*. The maxwell–boltzmann distribution. *Molecular Physics*, 103(21-23):2821–2828, 2005.

[122] Michael P Allen and Dominic J Tildesley. *Computer simulation of liquids*. Oxford university press, 2017.

[123] Loup Verlet. Computer" experiments" on classical fluids. i. thermodynamical properties of lennard-jones molecules. *Physical review*, 159(1):98, 1967.

[124] Dennis C Rapaport. *The art of molecular dynamics simulation*. Cambridge university press, 2004.

[125] Luca Monticelli and D Peter Tieleman. Force fields for classical molecular dynamics. *Biomolecular simulations: Methods and protocols*, pages 197–213, 2013.

[126] César A López, Zofie Sovova, Floris J van Eerden, Alex H de Vries, and Siewert J Marrink. Martini force field parameters for glycolipids. *Journal of chemical theory and computation*, 9(3):1694–1708, 2013.

[127] Luca Monticelli, Senthil K Kandasamy, Xavier Periole, Ronald G Larson, D Peter Tieleman, and Siewert-Jan Marrink. The martini coarse-grained force field: extension to proteins. *Journal of chemical theory and computation*, 4(5):819–834, 2008.

[128] Alexander D MacKerell Jr, Nilesh Banavali, and Nicolas Foloppe. Development and current status of the charmm force field for nucleic acids. *Biopolymers: Original Research on Biomolecules*, 56(4):257–265, 2000.

[129] William L Jorgensen, David S Maxwell, and Julian Tirado-Rives. Development and testing of the opls all-atom force field on conformational energetics and properties of organic liquids. *Journal of the American Chemical Society*, 118(45):11225–11236, 1996.

[130] Nathan Schmid, Andreas P Eichenberger, Alexandra Choutko, Sereina Riniker, Moritz Winger, Alan E Mark, and Wilfred F van Gunsteren. Definition and testing of the gromos force-field versions 54a7 and 54b7. *European biophysics journal*, 40(7):843–856, 2011.

[131] Junmei Wang, Romain M Wolf, James W Caldwell, Peter A Kollman, and David A Case. Development and testing of a general amber force field. *Journal of computational chemistry*, 25(9):1157–1174, 2004.

[132] Thomas P Senftle, Sungwook Hong, Md Mahbubul Islam, Sudhir B Kylasa, Yuanxia Zheng, Yun Kyung Shin, Chad Junkermeier, Roman Engel-Herbert, Michael J Janik, Hasan Metin Aktulga, et al. The reaxff reactive force-field:

development, applications and future directions. *npj Computational Materials*, 2(1):1–14, 2016.

[133] Ariel A Chialvo and Pablo G Debenedetti. On the use of the verlet neighbor list in molecular dynamics. *Computer physics communications*, 60(2):215–224, 1990.

[134] William L Jorgensen, Jayaraman Chandrasekhar, Jeffry D Madura, Roger W Impey, and Michael L Klein. Comparison of simple potential functions for simulating liquid water. *The Journal of chemical physics*, 79(2):926–935, 1983.

[135] Semen O Yesylevskyy, Lars V Schäfer, Durba Sengupta, and Siewert J Marrink. Polarizable water model for the coarse-grained martini force field. *PLoS Comput Biol*, 6(6):e1000810, 2010.

[136] Martin Vögele, Christian Holm, and Jens Smiatek. Properties of the polarizable martini water model: A comparative study for aqueous electrolyte solutions. *Journal of Molecular Liquids*, 212:103–110, 2015.

[137] Ulrich Essmann, Lalith Perera, Max L Berkowitz, Tom Darden, Hsing Lee, and Lee G Pedersen. A smooth particle mesh ewald method. *The Journal of chemical physics*, 103(19):8577–8593, 1995.

[138] Berk Hess, Henk Bekker, Herman JC Berendsen, and Johannes GEM Fraaije. Lincs: A linear constraint solver for molecular simulations. *Journal of computational chemistry*, 18(12):1463–1472, 1997.

[139] Jean-Paul Ryckaert, Giovanni Ciccotti, and Herman JC Berendsen. Numerical integration of the cartesian equations of motion of a system with constraints: molecular dynamics of n-alkanes. *Journal of computational physics*, 23(3):327–341, 1977.

[140] Mark James Abraham, Teemu Murtola, Roland Schulz, Szilárd Páll, Jeremy C Smith, Berk Hess, and Erik Lindahl. Gromacs: High performance

molecular simulations through multi-level parallelism from laptops to supercomputers. *SoftwareX*, 1:19–25, 2015.

[141] Denis J Evans and Brad Lee Holian. The nose–hoover thermostat. *The Journal of chemical physics*, 83(8):4069–4074, 1985.

[142] Mario Fernández-Pendás, Bruno Escribano, Tijana Radivojević, and Elena Akhmatskaya. Constant pressure hybrid monte carlo simulations in gromacs. *Journal of molecular modeling*, 20:1–10, 2014.

[143] Michele Parrinello and Aneesur Rahman. Polymorphic transitions in single crystals: A new molecular dynamics method. *Journal of Applied physics*, 52(12):7182–7190, 1981.

[144] Tomas Hansson, Chris Oostenbrink, and WilfredF van Gunsteren. Molecular dynamics simulations. *Current opinion in structural biology*, 12(2):190–196, 2002.

[145] Hans C Andersen. Molecular dynamics simulations at constant pressure and/or temperature. *The Journal of chemical physics*, 72(4):2384–2393, 1980.

[146] Sana Bougueroua, Riccardo Spezia, S Pezzotti, Sandrine Vial, Franck Quessette, D Barth, and M-P Gaigeot. Graph theory for automatic structural recognition in molecular dynamics simulations. *The Journal of chemical physics*, 149(18), 2018.

[147] Robert M Ziolek, Paul Smith, Demi L Pink, Cecile A Dreiss, and Christian D Lorenz. Unsupervised learning unravels the structure of four-arm and linear block copolymer micelles. *Macromolecules*, 54(8):3755–3768, 2021.

[148] Natasha H Rhys, Mohamed Ali Al-Badri, Robert M Ziolek, Richard J Gillams, Louise E Collins, M Jayne Lawrence, Christian D Lorenz, and Sylvia E McLain. On the solvation of the phosphocholine headgroup in

an aqueous propylene glycol solution. *The Journal of chemical physics*, 148(13), 2018.

[149] Aric Hagberg and Drew Conway. Networkx: Network analysis with python. *URL: https://networkx. github. io*, 2020.

[150] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[151] Leland McInnes, John Healy, and Steve Astels. hdbscan: Hierarchical density based clustering. *J. Open Source Softw.*, 2(11):205, 2017.

[152] Berk Hess, Carsten Kutzner, David Van Der Spoel, and Erik Lindahl. Gromacs 4: algorithms for highly efficient, load-balanced, and scalable molecular simulation. *Journal of chemical theory and computation*, 4(3):435–447, 2008.

[153] Naveen Michaud-Agrawal, Elizabeth J Denning, Thomas B Woolf, and Oliver Beckstein. Mdanalysis: a toolkit for the analysis of molecular dynamics simulations. *Journal of computational chemistry*, 32(10):2319–2327, 2011.

[154] Jiří Šponer, Jerzy Leszczynski, and Pavel Hobza. Electronic properties, hydrogen bonding, stacking, and cation binding of dna and rna bases. *Biopolymers: Original Research on Biomolecules*, 61(1):3–31, 2001.

[155] Greg Landrum. Rdkit documentation. *Release*, 1(1-79):4, 2013.

[156] Suhao Wang, Guangzheng Zuo, Jongho Kim, and Henning Sirringhaus. Progress of conjugated polymers as emerging thermoelectric materials. *Progress in Polymer Science*, 129:101548, 2022.

[157] Martin Gauthier-Jaques, Hatice Mutlu, and Patrick Theato. Cage-shaped polymers synthesis: A comprehensive state-of-the-art. *Macromolecular Rapid Communications*, 43(12):2100760, 2022.

[158] Takuya Yamamoto and Yasuyuki Tezuka. Topological polymer chemistry: a cyclic approach toward novel polymer properties and functions. *Polymer Chemistry*, 2(9):1930–1941, 2011.

[159] Josef Jancar, JF Douglas, Francis W Starr, SK Kumar, Philippe Cassagnau, AJ Lesser, Sandy Sanford Sternstein, and MJ Buehler. Current issues in research on structure–property relationships in polymer nanocomposites. *Polymer*, 51(15):3321–3343, 2010.

[160] Robert M Ziolek, Jasmin Omar, Wenjing Hu, Lionel Porcar, Gustavo Gonzalez-Gaitano, Cecile A Dreiss, and Christian D Lorenz. Understanding the ph-directed self-assembly of a four-arm block copolymer. *Macromolecules*, 53(24):11065–11076, 2020.

[161] S Shenogin, A Bodapati, L Xue, R Ozisik, and P Keblinski. Effect of chemical functionalization on thermal transport of carbon nanotube composites. *Applied Physics Letters*, 85(12):2229–2231, 2004.

[162] Sezen Curgul, Krystyn J Van Vliet, and Gregory C Rutledge. Molecular dynamics simulation of size-dependent structural and thermal properties of polymer nanofibers. *Macromolecules*, 40(23):8483–8489, 2007.

[163] Suchira Sen, James D Thomin, Sanat K Kumar, and Pawel Keblinski. Molecular underpinnings of the mechanical reinforcement in polymer nanocomposites. *Macromolecules*, 40(11):4059–4067, 2007.

[164] Jinbo Zhao, Lili Wu, Chuanxing Zhan, Qian Shao, Zhanhu Guo, and Liqun Zhang. Overview of polymer nanocomposites: Computer simulation understanding of physical properties. *Polymer*, 133:272–287, 2017.

[165] Thomas E Gartner III and Arthi Jayaraman. Modeling and simulations of polymers: a roadmap. *Macromolecules*, 52(3):755–786, 2019.

[166] Paul Smith and Christian D Lorenz. Lipyphilic: A python toolkit for the analysis of lipid membrane simulations. *bioRxiv*, 2021.

[167] Marina Kovacevic, Igor Balaz, Domenico Marson, Erik Laurini, and Branislav Jovic. Mixed-monolayer functionalized gold nanoparticles for cancer treatment: Atomistic molecular dynamics simulations study. *Biosystems*, 202:104354, 2021.

[168] Marcello Sega, Sofia S Kantorovich, Pál Jedlovszky, and Miguel Jorge. The generalized identification of truly interfacial molecules (itim) algorithm for nonplanar interfaces. *The Journal of chemical physics*, 138(4), 2013.

[169] Martin B Ulmschneider, Jacques PF Doux, J Antoinette Killian, Jeremy C Smith, and Jakob P Ulmschneider. Mechanism and kinetics of peptide partitioning into membranes from all-atom simulations of thermostable peptides. *Journal of the American Chemical Society*, 132(10):3452–3460, 2010.

[170] Xiaotian Sun, Zhiwei Feng, Tingjun Hou, and Youyong Li. Mechanism of graphene oxide as an enzyme inhibitor from molecular dynamics simulations. *ACS applied materials & interfaces*, 6(10):7153–7163, 2014.

[171] Benjamin J Schwartz. Conjugated polymers as molecular materials: How chain conformation and film morphology influence energy transfer and interchain interactions. *Annual review of physical chemistry*, 54(1):141–172, 2003.

[172] Robert M Ziolek, Alejandro Santana-Bonilla, Raquel Lopez-Rios de Castro, Reimer Kuhn, Mark Green, and Christian D Lorenz. Conformational heterogeneity and interchain percolation revealed in an amorphous conjugated polymer. *ACS nano*, 2022.

[173] Andrea Joseph, Chris W Nyambura, Danielle Bondurant, Kylie Corry, Denise Beebout, Thomas R Wood, Jim Pfaendtner, and Elizabeth Nance. Formulation and efficacy of catalase-loaded nanoparticles for the treatment of neonatal hypoxic-ischemic encephalopathy. *Pharmaceutics*, 13(8):1131, 2021.

[174] Elizabeth M Enlow, J Christopher Luft, Mary E Napier, and Joseph M DeSimone. Potent engineered plga nanoparticles by virtue of exceptionally high chemotherapeutic loadings. *Nano letters*, 11(2):808–813, 2011.

[175] Yunfei Wang, Peifeng Liu, Yourong Duan, Xia Yin, Qi Wang, Xiaofei Liu, Xinran Wang, Jinhua Zhou, Wenwen Wang, Lihua Qiu, et al. Specific cell targeting with aprpg conjugated peg–plga nanoparticles for treating ovarian cancer. *Biomaterials*, 35(3):983–992, 2014.

[176] Mona Alibolandi, Fatemeh Sadeghi, Khalil Abnous, Fatemeh Atyabi, Mohammad Ramezani, and Farzin Hadizadeh. The chemotherapeutic potential of doxorubicin-loaded peg-b-plga nanopolymersomes in mouse breast cancer model. *European Journal of Pharmaceutics and Biopharmaceutics*, 94:521–531, 2015.

[177] SM Jusu, JD Obayemi, AA Salifu, CC Nwazojie, V Uzonwanne, OS Odusanya, and WO Soboyejo. Drug-encapsulated blend of plga-peg microspheres, 2020.

[178] Chris W Nyambura, Janani Sampath, Elizabeth Nance, and Jim Pfaendtner. Exploring structure and dynamics of the polylactic-co-glycolic acid–polyethylene glycol copolymer and its homopolymer constituents in various solvents using all-atom molecular dynamics. *Journal of Applied Polymer Science*, 139(31):e52732, 2022.

[179] Hwankyu Lee. Molecular simulations of pegylated biomolecules, liposomes, and nanoparticles for drug delivery applications. *Pharmaceutics*, 12(6):533, 2020.

[180] Ya-Ping Li, Yuan-Ying Pei, Xian-Ying Zhang, Zhou-Hui Gu, Zhao-Hui Zhou, Wei-Fang Yuan, Jian-Jun Zhou, Jian-Hua Zhu, and Xiu-Jian Gao. Pegylated plga nanoparticles as protein carriers: synthesis, preparation and biodistribution in rats. *Journal of controlled release*, 71(2):203–211, 2001.

[181] Sarbari Acharya and Sanjeeb K Sahoo. Plga nanoparticles containing various anticancer agents and tumour delivery by epr effect. *Advanced drug delivery reviews*, 63(3):170–183, 2011.

[182] Homayoun Asadzadeh and Ali Moosavi. Investigation of the interactions between melittin and the plga and pla polymers: molecular dynamic simulation and binding free energy calculation. *Materials Research Express*, 6(5):055318, 2019.

[183] H Asadzadeh, A Moosavi, and JH Arghavani. The effect of chitosan and peg polymers on stabilization of gf-17 structure: a molecular dynamics study. *Carbohydrate polymers*, 237:116124, 2020.

[184] Shahin Aghamiri, Farshid Zandsalimi, Pourya Raee, Mohammad-Amin Abdollahifar, Shing Cheng Tan, Teck Yew Low, Sajad Najafi, Milad Ashrafizadeh, Ali Zarrabi, Hossein Ghanbarian, et al. Antimicrobial peptides as potential therapeutics for breast cancer. *Pharmacological research*, 171:105777, 2021.

[185] Amy A Baxter, Fung T Lay, Ivan KH Poon, Marc Kvansakul, and Mark D Hulett. Tumor cell membrane-targeting cationic antimicrobial peptides: novel insights into mechanisms of action and therapeutic prospects. *Cellular and Molecular Life Sciences*, 74(20):3809–3825, 2017.

[186] Adriano Brandelli. Nanostructures as promising tools for delivery of antimicrobial peptides. *Mini reviews in medicinal chemistry*, 12(8):731–741, 2012.

[187] Jorrit Jeroen Water, Simon Smart, Henrik Franzyk, Camilla Foged, and Hanne Mørck Nielsen. Nanoparticle-mediated delivery of the antimicrobial peptide plectasin against staphylococcus aureus in infected epithelial cells. *European Journal of Pharmaceutics and Biopharmaceutics*, 92:65–73, 2015.

[188] Herman JC Berendsen, David van der Spoel, and Rudi van Drunen. Gromacs: a message-passing parallel molecular dynamics implementation. *Computer physics communications*, 91(1-3):43–56, 1995.

[189] Siewert J Marrink, H Jelger Risselada, Serge Yefimov, D Peter Tieleman, and Alex H De Vries. The martini force field: coarse grained model for biomolecular simulations. *The journal of physical chemistry B*, 111(27):7812–7824, 2007.

[190] Kevin R Hinkle. Using coarse-grained models to examine structure-property relationships of diblock-arm star polymers. *European Polymer Journal*, page 110149, 2020.

[191] Fabian Grunewald, Giulia Rossi, Alex H de Vries, Siewert J Marrink, and Luca Monticelli. Transferable martini model of poly (ethylene oxide). *The Journal of Physical Chemistry B*, 122(29):7436–7449, 2018.

[192] Sunhwan Jo, Taehoon Kim, Vidyashankara G Iyer, and Wonpil Im. Charmm-gui: a web-based graphical user interface for charmm. *Journal of computational chemistry*, 29(11):1859–1865, 2008.

[193] Giovanni Bussi, Davide Donadio, and Michele Parrinello. Canonical sampling through velocity rescaling. *The Journal of chemical physics*, 126(1), 2007.

[194] Par Bjelkmar, Per Larsson, Michel A Cuendet, Berk Hess, and Erik Lindahl. Implementation of the charmm force field in gromacs: analysis of protein stability effects from correction maps, virtual interaction sites, and water models. *Journal of chemical theory and computation*, 6(2):459–466, 2010.

[195] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[196] William Humphrey, Andrew Dalke, and Klaus Schulten. Vmd: visual molecular dynamics. *Journal of molecular graphics*, 14(1):33–38, 1996.

[197] Zhiqiang Shen, Mu-Ping Nieh, and Ying Li. Decorating nanoparticle surface for targeted drug delivery: opportunities and challenges. *Polymers*, 8(3):83, 2016.

[198] Xinru You, Yang Kang, Geoffrey Hollett, Xing Chen, Wei Zhao, Zhipeng Gu, and Jun Wu. Polymeric nanoparticles for colon cancer therapy: overview and perspectives. *Journal of Materials Chemistry B*, 4(48):7779–7792, 2016.

[199] Vinod Ravasaheb Shinde, Neeraja Revi, Sivasubramanian Murugappan, Surya Prakash Singh, and Aravind Kumar Rengan. Enhanced permeability and retention effect: A key facilitator for solid tumor targeting by nanoparticles. *Photodiagnosis and Photodynamic Therapy*, 39:102915, 2022.

[200] Dan Peer, Jeffrey M Karp, Seungpyo Hong, Omid C Farokhzad, Rimona Margalit, and Robert Langer. Nanocarriers as an emerging platform for cancer therapy. *Nano-enabled medical applications*, pages 61–91, 2020.

[201] Luan NM Nguyen, Zachary P Lin, Shrey Sindhwani, Presley MacMillan, Stefan M Mladjenovic, Benjamin Stordy, Wayne Ngo, and Warren CW Chan. The exit of nanoparticles from solid tumours. *Nature Materials*, pages 1–12, 2023.

[202] Francesca Perrotti, Consuelo Rosa, Ilaria Cicalini, Paolo Sacchetta, Piero Del Boccio, Domenico Genovesi, and Damiana Pieragostino. Advances in lipidomics for cancer biomarkers discovery. *International journal of molecular sciences*, 17(12):1992, 2016.

[203] Linlin Zhang, Bijun Zhu, Yiming Zeng, Hui Shen, Jiaqiang Zhang, and Xiangdong Wang. Clinical lipidomics in understanding of lung cancer: opportunity and challenge. *Cancer letters*, 470:75–83, 2020.

[204] Emily G Armitage and Andrew D Southam. Monitoring cancer prognosis, diagnosis and treatment efficacy using metabolomics and lipidomics. *Metabolomics*, 12:1–15, 2016.

[205] Chiranjeevi Peetla, Sivakumar Vijayaraghavalu, and Vinod Labhasetwar. Biophysics of cell membrane lipids in cancer drug resistance: Implications for drug transport and drug delivery with nanoparticles. *Advanced drug delivery reviews*, 65(13-14):1686–1698, 2013.

[206] Maria Patitsa, Konstantina Karathanou, Zoi Kanaki, Lamprini Tzioga, Natassa Pippa, Constantinos Demetzos, Dimitris A Verganelakis, Zoe Cournia, and Apostolos Klinakis. Magnetic nanoparticles coated with polyarabic acid demonstrate enhanced drug delivery and imaging properties for cancer theranostic applications. *Scientific reports*, 7(1):775, 2017.

[207] Lingxiao Li, Yuanyuan Yang, Lin Wang, Feng Xu, Yuan Li, and Xiaocong He. The effects of serum albumin pre-adsorption of nanoparticles on protein corona and membrane interaction: A molecular simulation study. *Journal of Molecular Biology*, 435(1):167771, 2023.

[208] Shirley E Poduslo, Karen Miller, and Yong Jang. Comparison of lipids and lipid metabolism in a human glioma cell line, its clone, and oligodendroglia. *Cancer research*, 43(3):1014–1018, 1983.

[209] Helgi I Ingólfsson, Timothy S Carpenter, Harsh Bhatia, Peer-Timo Bremer, Siewert J Marrink, and Felice C Lightstone. Computational lipidomics of the neuronal plasma membrane. *Biophysical journal*, 113(10):2271–2280, 2017.

[210] Pin-Chia Hsu, Bart MH Bruininks, Damien Jefferies, Paulo Cesar Telles de Souza, Jumin Lee, Dhilon S Patel, Siewert J Marrink, Yifei Qi, Syma Khalid, and Wonpil Im. Charmm-gui martini maker for modeling and simulation of complex bacterial membranes with lipopolysaccharides. *Journal of computational chemistry*, 38(27):2354–2363, 2017.

[211] RJ Gowers, M Linke, J Barnoud, TJE Reddy, MN Melo, SL Seyler, J Domański, DL Dotson, S Buchoux, IM Kenney, et al. Proceedings of the 15th python in science conference. 2016.

[212] Sébastien Buchoux. Fatslim: a fast and robust software to analyze md simulations of membranes. *Bioinformatics*, 33(1):133–134, 2017.

[213] Yue Hui, Xin Yi, Fei Hou, David Wibowo, Fan Zhang, Dongyuan Zhao, Huajian Gao, and Chun-Xia Zhao. Role of nanoparticle mechanical properties in cancer drug delivery. *ACS nano*, 13(7):7410–7424, 2019.

[214] Shikha Nangia and Radhakrishna Sureshkumar. Effects of nanoparticle charge and shape anisotropy on translocation through cell membranes. *Langmuir*, 28(51):17666–17671, 2012.

[215] Yachong Guo, Marco Werner, Wenfei Li, Jens-Uwe Sommer, and Vladimir A Baulin. Shape-adaptive single-chain nanoparticles interacting with lipid membranes. *Macromolecules*, 52(24):9578–9584, 2019.

[216] Erem Bilensoy. Cationic nanoparticles for cancer therapy. *Expert opinion on drug delivery*, 7(7):795–809, 2010.

[217] Ke-Tao Jin, Ze-Bei Lu, Jin-Yang Chen, Yu-Yao Liu, Huan-Rong Lan, Hai-Ying Dong, Fan Yang, Yuan-Yuan Zhao, and Xiao-Yi Chen. Recent trends in nanocarrier-based targeted chemotherapy: selective delivery of anticancer drugs for effective lung, colon, cervical, and breast cancer treatment. *Journal of Nanomaterials*, 2020:1–14, 2020.