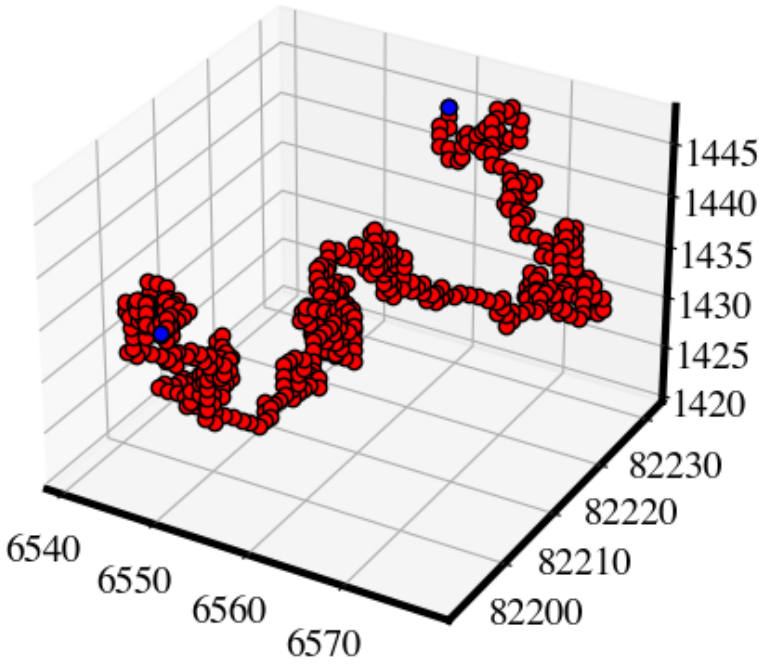
Project Report

Rosenbluth Sampling of Lattice Polymers : From Self-Avoiding Random Walks to the Coil-Globule Transition



Allaglo N., Bry C.-E., Duroy E.

MU4PY210

SORBONNE UNIVERSITÉ

Master Physique fondamentale et applications

Le 24 mars 2024

Table des matières

Table des figures

Liste des tableaux

1 Introduction

Polymers are large molecules composed of several repeating monomers. They are involved in chemical processes and in the formation of biological structures. DNA, proteins, cellulose, paint and plastics are just a few examples of polymers. Technological breakthroughs are linked to these studies, especially for the development of new materials or the study of biological processes. In the context of physics, the field is vast and complex on top of numerical simulations becoming particularly meaningful. The latter offer acute comprehension into macromolecules's configurations, behaviors and properties. For example, these predictions offer insights into the synthesis of novel materials with significant potential. Various simulation methods exist, such as **Monte Carlo** or **Molecular Dynamics**. Because different spatial and time scales are relevant in the physics of polymers, some methods might be preferred to others depending on the simulated system.
The purpose of this project is mainly computational : discover and practice numerical approaches typically encountered in the physics of polymers. The aim is to build a robust simulation program from scratch. In this document, we will introduce the relevant formalism of lattice polymers before reviewing static Monte Carlo methods. In order to benchmark our program, we first try to model a *self-avoiding random walks* and replicate some of its universal scaling laws. Finally, we attempt to obtain a *coil-globule transition* with an *interacting* self-avoiding walk.

2 Review of the physics of polymers

Etymologically, the word polymer comes from the Greek polus, meaning "many," and meros, meaning "parts." A real polymer consists of the repetition of a large number of identical or different monomer units linked by covalent bonds. The interactions between neighboring monomers depend on their relative orientation. Additionally, van der Waals forces (2 to 16 kJ/mol), hydrogen bonds (40 kJ/mol), and electrostatic forces also modify the structure of a polymer. These forces depend on the solvent used, the nature of the monomer atoms (or molecules) and the temperature. Above a certain temperature, the attractive interactions become negligible, and the repulsive potential can be estimated by a hard sphere.

The field of polymer science comprises of two main branches : polymer chemistry and polymer physics. Polymer physics, in particular, draws on statistical and numerical physics and allows for the study of their properties such as structure, dynamics, and mechanical, thermal, electrical and optical properties.

2.1 Framework

Several parameters are essential and commonly studied in polymer physics. First, the degree of polymerization (DP) is an important parameter that defines the length of a polymer chain (number of monomer units). It is directly proportional to the molecular weight of the polymer and influences its properties at the macroscopic scale. It is also related to the end-to-end distance of the chain. Moreover, we can define the spatial extent occupied by a polymer with its radius of gyration. Mathematically, it is the average distance between one end of the chain and the center of mass of the structure [1]. The conformation in the space of a polymer is related to statistical entropy. These last two parameters (radius of gyration and end-to-end distance) are widely studied in numerical polymer physics, and they will be the key components our program will measure.

Finally, our purpose is to study the coil-globule transition. The "coil-globule" transition is a change in the structural configuration of a polymer. The extended structure is referred to as "coil," while the compact and folded structure is called "globule." These structures depend on the balance between the attractive and repulsive forces among monomers, such as the environment (temperature, solvent nature, etc.). Thus, for a given polymer system, by modifying the strength of attraction between monomers, we are able to study this transition. When the forces between monomers are repulsive, the polymer assumes an extended shape. Conversely, when the forces are attractive, the structure collapses into a "globule." The "coil-globule" transition is often characterized by significant changes in the polymer's physical properties, such as

viscosity, density, and diffusion, making it an important phenomenon to study in the field of polymer physics. Practical applications include gel formation, polymer precipitation, and the structure of DNA in biological cells.

2.2 Scale and simulation methods

Some of our first bibliographic research leads us to a multitude of information that we have to classify in order to really understand the subject. In particular, the diversity of possibilities and different models caught our attention. Polymers are complex and multi-scale structures. Depending on the properties that we want to study, the space-temporal scales used will determine the accuracy of what we study and some simulation models are therefore more or less well adapted. The bigger the scale is, the more approximations there are, and the more we can observe the global properties of materials. Different models are associated with the scales, and each have its advantages and limits [2] (Table 1).

Global scale	space scale (m)	time scale (s)	algorithm
atomic	10^{-10}	10^{-15}	Monte Carlo, molecular dynamic
molecular	10^{-9}	10^{-9}	Monte Carlo, molecular dynamic
mesoscopic	10^{-6} to 10^{-3}	10^{-6} to 10^{-3}	Dissipative particle dynamics
macroscopic	10^{-2}	1	finite element method

Table 1. Computer algorithms used in polymer simulation depending on their time and space scale resolution

Furthermore, we realized that there are accessible simulation tools and complex programs available on the Internet for polymer simulations. The list below presents some examples. The study of these tools tells us what it is possible to do, the necessary approximations of certain models and their limits. Numerical simulation is vast and comparing these models can be interesting. However, this requires an effort to understand and to analyze the programs. The codes already available will probably be more efficient, but making our own code allows us to learn more about physics and numerical programming. The goal of this project is not to produce research results. In this regard, many powerful simulation tools already such as LAMMPS [3], GROMACS [4], CHARMM [5] or OxDNA [6] exist.

2.3 Our model for polymers

Theoretically, a polymer can be represented in many different ways. One of these is the Gaussian polymer model, a very simple description based on a sequence of monomers with equiprobable and random directions. In numerical physics, this type of polymer can be simulated by a 3D random walk (§2.4). In this case, the chain ends follow a Gaussian distribution. However, this model is limited [7]. For example, it does not account for interactions between monomers that can induce correlations between orientations. One trick is to create a self-avoiding random walk (§3.2.3), thus modeling the hard-sphere repulsion in a simple and approximate way.

The next section digs deeper into the choices we made for the simulation.

2.4 Lattice polymers

First of all, to simulate an object, we have to define its characteristics. In this project we want to simulate polymers that are linear, and immersed in a good solvent. Being in a good solvent means that different polymers are separated from each other and will not interact. That is why we can thus consider the physics of a single polymer alone. One particular property of a polymer is its length and its rigidity : in general polymers are rather floppy. We will not describe this from a microscopic model (such as quantum chemistry), but we will instead use a more simplistic model.
A polymer is hereby described as occupancy of a 3-dimensional lattice. With this approximation, we are able to encode the monomers positions as cartesian coordinates in a 3-dimensional space. A polymer of size N monomers is therefore characterized by a set of N triplets

$$\{\boldsymbol{r}_1, \boldsymbol{r}_2, \cdots, \boldsymbol{r}_{N-1}\}\,. \tag{1}$$

Such a system (without any physical/energetic constraint) can be modelled as a simple *random walk* where the next site is chosen uniformly between the closest-neighbors of the present site. To be more realistic, we shall consider that two monomers cannot interpenetrate. To implement this, we must replace the random walk by a *self-avoiding random walk*.

To describe a polymer quantitatively, we can study its length r_e (end-to-end distance) and/or its radius of gyration r_g which describes how "curled up" the polymer is. We now define them mathematically [1] :

$$r_e^2(N) = |\boldsymbol{r}_N - \boldsymbol{r}_0|^2 \tag{2}$$

$$r_g^2(N) = \frac{1}{N}\sum_{i=0}^{N} |r_i - r_{\mathrm{cm}}|^2, \tag{3}$$

with $r_{\mathrm{cm}} = \frac{1}{N}\sum_{i=0}^{N} \boldsymbol{r}_i$ and N being the number of monomers in the chain.
In order to simulate lattice polymers, inspired by the famous Ising model, Monte Carlo methods can become very powerful. In the next section, we will give a brief review of the latter before focusing on static schemes methods and more precisely the Prune-Enriched Rosenbluth Method (PERM).

3 Monte Carlo sampling methods applied to polymers

3.1 Review of Monte Carlo methods

3.1.1 Use in statistical physics

In statistical physics, we typically want to compute *partition functions*

$$Z_\psi = \sum_\ell \exp(-\beta\psi(\ell)) = \sum_\ell \boldsymbol{\psi}(\ell), \tag{4}$$

with $\beta = \frac{1}{k_{\mathrm{B}}T}$ defining the inverse temperature, ψ (resp. $\boldsymbol{\psi}$) being the thermodynamic potential (resp. Boltzmann weight) associated with a peculiar ensemble (depending on external constraints) and ℓ characterizing a micro-state in a configuration space Ω. A very naive way to sample (4) would be to choose a set of random states $\{\ell_1, \ell_2, \cdots, \ell_M\}$ of size M according to a probability distribution $\mathbf{P}_S(\ell)$ and to approximate Z_ψ to

$$\hat{Z}_\psi^M = \frac{1}{M}\sum_{i=1}^{M} \frac{\boldsymbol{\psi}(\ell_i)}{\mathbf{P}_S(\ell_i)} =: \frac{1}{M}\sum_{i=1}^{M} W(\ell_i), \tag{5}$$

knowing that $Z_\psi = \sum_\ell \mathbf{P}_S(\ell)\frac{\boldsymbol{\psi}(\ell)}{\mathbf{P}_S(\ell)}$. We eventually have $Z_\psi^M \underset{M\to\infty}{\longrightarrow} Z_\psi$ thanks to the law of large numbers. The computation of partition functions therefore relies on a new quantity (a weight actually) $W(\ell) = \dfrac{\boldsymbol{\psi}(\ell)}{\mathbf{P}_S(\ell)}$. In the same spirit, this approximation finally enables us to compute thermodynamic averages for any observable A :

$$\langle A\rangle_\psi = \frac{1}{Z_\psi}\sum_\ell A(\ell)\boldsymbol{\psi}(\ell) \approx \frac{\sum_{i=1}^{M} A(\ell_i)W(\ell_i)}{\sum_{i=1}^{M} W(\ell_i)} =: \langle\hat{A}\rangle_\psi, \tag{6}$$

because $M\hat{Z}_\psi^M = \sum_{i=1}^{M} W(\ell_i)$. The computation of thermodynamic observable properties finally depends on the choice of a sampling probability distribution function $\mathbf{P}_S(\ell)$. In order for a calculation to be efficient yet precise, we need to sample a dense region of the configuration space Ω, in other words all that matters is $\mathbf{P}_S(\ell)$.

3.1.2 Exact sampling

Choosing $\mathbf{P}_S(\ell) \propto \boldsymbol{\psi}(\ell)$ provides exact sampling : this can be done through the *Metropolis-Hastings algorithm*. The real probability distribution function behaving as $\mathbf{P}_R(\ell) = \dfrac{\boldsymbol{\psi}(\ell)}{Z}$, we can avoid the calculation of Z with the fraction of two microstates α and β populations [1]. These populations fractions come handy in stochastic (*Markov-Chain*) methods to evaluate transition rates. Stochastic methods yielding highly correlated trials, their principle usefulness is to converge to one relaxed macrostate. This approach however requires to construct a fully-developed physical object and to apply configuration *moves* obeying stochastic rules on it. It might not always be the best when the system at hand can be constructed sequentially (e.g. polymers) : in this case, it is sometimes better to generate collections of decorrelated physical objects. The *Rosenbluth algorithm* tries to address this issue.

3.2 Rosenbluth sampling

In this section, we shall consider a simple self-avoiding random walk (SARW) on a d-dimensional cubic lattice. More precisely we will focus on a SARW with N monomers built from scratch - the only rule being that a site cannot be passed through more than once. This instance requires $\boldsymbol{\psi}(\ell) = 1$ so that the only interesting constraint is for a walk to survive (not reach a closed-loop of bonded-neighbors) : the phenomenon is called *attrition*. Later on, we will denote Z_n^M the partition function of polymer of size n with M trials. We will introduce energetic and mechanical constraints in section §3.2.3.

3.2.1 The original Rosenbluth algorithm

Rosenbluth's main idea is that $\mathbf{P}_S(\ell)$ represents a bias which can itself be constructed sequentially. For the first trial (labelled 1), the first monomer has $m_1^1 = 2d$ free-neighbors. The second added monomer has $m_2^1 = 2d-1$ free-neighbors. At the n-th step in the growth, the last added monomers has m_n^1 free-neighbors so that the last monomer has m_N^1 free-neighbors. The number of similar growths for this trial is simply given by $m_N^1 \cdot m_{N-1}^1 \cdots m_1^1$ (for the first step, you could have taken $2d$ different directions, for the second step m_2^1, etc.). The algorithm now repeats the process to generate a collection of SARW. The statistical weight $W(i) = \prod_{n=1}^{N} m_n^i$ for the i-th walk is based on the sampling bias (cf. (5))

$$\mathbf{P}_S(\ell_i) = \left(\prod_{n=1}^{N} m_n^i \right)^{-1} \tag{7}$$

which itself is based on the sequential bias $\dfrac{1}{m_n^i}$ at the n-th step in the growth. This bias is actually very intuitive : if there are m_n free-neighbors at the last step, the walk proceeds to go in one of these m_n sites with equal probabilities !

There are two main problems with the original Rosenbluth algorithm [8] : the surviving rate decreases with N increasing and the law of $W(i)$ shows dramatic fluctuations since all the $\left\{m_n^i\right\}_n$ are weakly correlated. As a consequence, the sampling of the configuration space is fairly poor ...

3.2.2 *Prune-enriched Rosenbluth method* (PERM)

PERM has proved its robustness in a large range of fields : stretched polymers, confined geometries, protein folding and much more [8]. The main idea is that 1- we will eliminate low-weight trials by still managing to travel in the neighbourhood of low-density portions of the configuration space (pruning) and 2- we will clone high-weight walks in order to tour high-density sections of the configuration space for a longer time (without losing CPU time). The huge fluctuations of $W(i)$ are reduced thanks to PERM since it leads to a potential stabilization of the weights coming from the cloning. If at any step n, the weight $W_n(i) = \prod_{k=1}^{n} m_k^i$ would be $> W_n^+$, the polymer is cloned and stacked in a drawer.

1. Indeed, the quantity $\frac{\mathbf{P}_R(\alpha)}{\mathbf{P}_R(\beta)}$ does not depend of Z !

On the other hand, if $W_n(i) < W_n^-$, there is a 50% chance that the polymer is either discarded either saved. In the first case, because we generate two identical polymers, we must divide their weight by 2 to preserve statistical averages. In the second case, the weight is multiplied since *in average*, half the time, the polymer is killed NOT because of attrition : in order to balance this inequity one must multiply the weight by 2.
The key part of PERM lies in the choice of the upper and lower thresholds $W_n^{\pm}$. We usually want to compare how the current weight compares to the average weight computed during all previous runs : that is precisely the partition function $Z_n^M = \frac{1}{M}\sum_{i=1}^{n} W_n(i)$. The thresholds are thus chosen as

$$W_n^{\pm} = c^{\pm} Z_n^M, \tag{8}$$

$c_{\pm}$ being global constants chosen at the start of the run and M representing the current number of trials of length n polymers in our simulation. The method of course needs to start after some trials to have an estimation of the partition function. PERM is finally implemented as follows :

1. check if there exists a clone in the stack of clones and if so select the last one, if not start a new polymer from scratch
2. add a new step n in the growth of a polymer (based on some bias like Rosenbluth's)
3. update the weight $W_n(i) = W_{n-1}(i) \times m_n^i$
4. update partition function at current step Z_n
5. control the weight by applying PERM criteria
   (a) if **pruned** : call a random number $r < 1$, if $r < 1/2$ stop the growth of the current polymer, else, go back to step 2.
   (b) if **cloned** : place the current polymer properties on top of the stack of clones and go back to step 2.

3.2.3 Interacting Self-Avoiding Walks (ISAW)

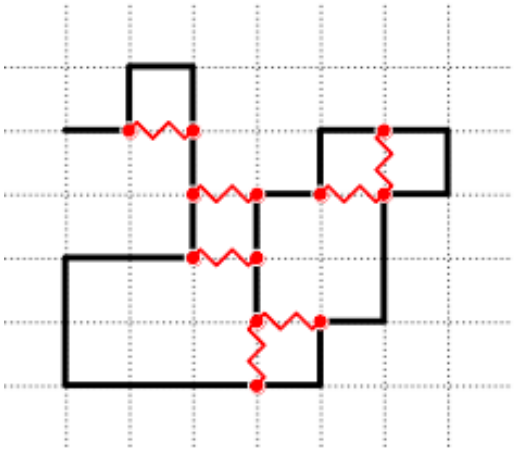


Fig. 1. Illustration of the interacting self-avoiding walk. Fictive monomer-monomer bonds are supported through a Boltzmann weight $e^{\beta\epsilon}$. Coiling is therefore thermally favoured. Figure taken from [9].

One can consider the SARW kind of boring : mononomers can only see their first neighbors and just avoid them. In fact, one can imagine that if there exists a monomer-monomer bonding energy, this energy remains present *even if* the bond has not been formed in the polymer generation. This process is understood better with some visualization. Take for example Figure 1 : in the process of constructing the polymer, all the red monomers did not bond. Nevertheless, these polymers experiment an extra fictive attractive energy ! Indeed, they are all still occupied sites set apart from a bonding length. Under an addition thermodynamic constraint characterized by the potential K, the partition function (4) is thus modified as

$$Z = \sum_{\ell} \exp(-\beta(j(\ell)\epsilon + K(\ell))) =: \sum_{\ell} q^{j(\ell)} e^{-\beta K(\ell)}, \tag{9}$$

ℓ browsing all possible walks, $j(\ell)$ representing the total number of non-bonded nearest neighbor pairs for configuration ℓ and $q = e^{-\beta\epsilon}$ the Boltzmann weight associated with the bonding-energy $\epsilon < 0$.

For $K = 0$, (5) yields the following weight :

$$W(\ell) = \frac{q^{j(\ell)}}{\mathbf{P}_S(\ell)}. \tag{10}$$

On top of that, $q^{j(\ell)}$ can be sequentially factorized : we have $j(\ell) = \sum_n^N j_n(\ell)$ with $j_n(\ell)$ denoting the number of non-bonded nearest neighbor pairs of the last monomer at the n-th step in the polymer's growth. The traditional Rosenbluth bias not changing from (7), we finally have the following sequentially factorized weight for the i-th trial/configuration :

$$W(i) = \prod_{n=1}^{N} m_n^i q^{j_n^i}. \tag{11}$$

A variant of the ISAW model is called "Biased ISAW." A bias (directional, physical structure, or bonding) is introduced into the system. These two simulation models lead to the coil-globule transition.

3.2.4 Thermo-mechanical constraints and Biased Self-Avoiding Walks (BISAW)

We only focus here on a constant external force $\boldsymbol{f}$ applied in an arbitrary direction $\mathbf{u}$. Without loss of generality, we can fix $\mathbf{u} = \mathbf{e}_x$. With a force reservoir, one can show that the potential K equals

$$K(\ell) = -\boldsymbol{f} \cdot (\boldsymbol{r}_N^\ell - \boldsymbol{r}_0^\ell), \tag{12}$$

$\boldsymbol{r}_i^\ell = x_i^\ell \mathbf{e}_x + y_i^\ell \mathbf{e}_y + z_i^\ell \mathbf{e}_z$ pointing to the position of the i-th monomer in the configuration ℓ. This associated Boltzmann weight simplifies to :

$$\exp\left(\beta f (x_N^\ell - x_0^\ell)\right) =: b^{(\Delta x)_N^\ell}, \tag{13}$$

with $b = \mathrm{e}^{\beta f}$ and $(\Delta x)_k^\ell = x_k^\ell - x_0^\ell$. (13) can also be sequentially factorized during the generation process : we have indeed $x_N^\ell - x_0^\ell = (x_N^\ell - x_{N-1}^\ell) + (x_{N-1}^\ell - x_{N-2}^\ell) + \cdots + (x_1^\ell - x_0^\ell)$ so that the whole weight (14) reads

$$W(i) = \prod_{n=1}^{N} m_n^i q^{j_n^i} b^{(\Delta x)_n^i}. \tag{14}$$

The sampling of the configuration space in this case can be accelerated by inducing a new bias at each step. Ref. [8] suggests that with a force in the x direction one can take the following random step probabilities (non-normalized) :

$$p_{\parallel} = \sqrt{b},\ p_{-\parallel} = \frac{1}{\sqrt{b}},\ p_{\perp} = 1. \tag{15}$$

Instead of selecting the m_n^i number of neighbors based on Rosenbluth bias, now we must take the inverse of these new biased probabilities.

3.3 Error estimation and benchmarking

3.3.1 Bootstrapping method

Bootstrapping methods are general techniques that we can use to estimate error relatively accurately and quickly. It is based on a sample S of polymers (usually the result of a MC simulation), from which we generate p new samples S_i of the same size by choosing random polymers and their associated weight *with replacement* (some polymers may be chosen several times, and some others not once). Then any observable obs is measured as an average on each polymers sample $\langle \mathrm{obs} \rangle_i$ and the targeted error can be estimated by the standard deviation :

$$s(\langle \mathrm{obs} \rangle) = \sqrt{\frac{1}{p}\sum_{i=1}^{p} (\langle \mathrm{obs} \rangle_i)^2 - \left(\frac{1}{p}\sum_{i=1}^{p} \langle \mathrm{obs} \rangle_i\right)^2}. \tag{16}$$

The bootstraping method has a lot of advantages but requires additional effort and computation time [10]. From these we can estimate the error as :

3.3.2 Comparison with universal laws

A natural way to evaluate the validity of our algorithm is to compare the results obtained with known universal laws. For example, for the SARW, it has been shown only numerically (there is no analytical proof of the following) that both the end-to-end and gyration lengths scale as :

$$\left\langle r_e^2 \right\rangle \sim N^{2\nu}, \tag{17}$$

$$\left\langle r_g^2 \right\rangle \sim N^{2\nu}. \tag{18}$$

with $\nu \approx 0.586$ in 3D [11]. Despite the fact that ν comes from simulations, theses scaling laws are quite robust to the point of being called *universal*. We will therefore take (17) and (18) as benchmarks for our SARW model.

4 Results

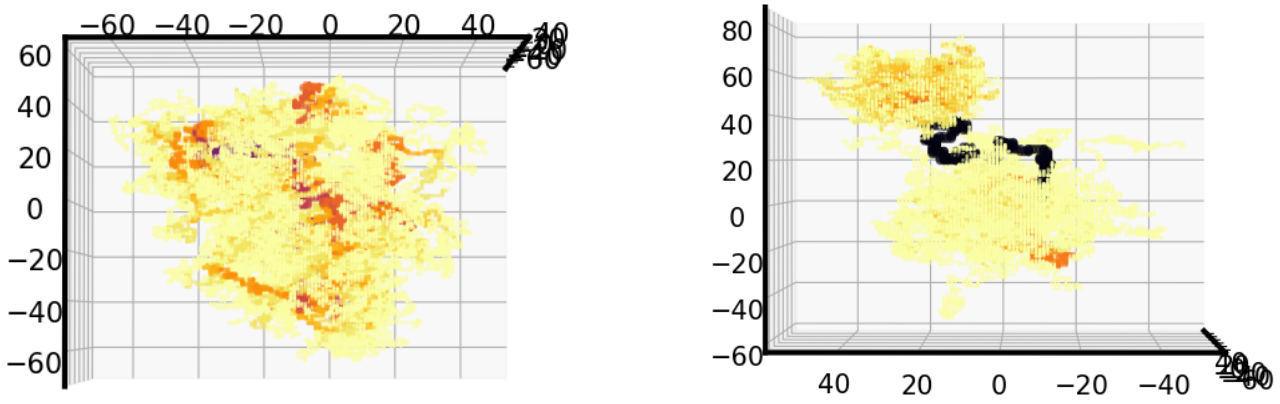
Before delving deeper into the results we obtained, we must point some difficulties regarding the convergence of our simulations. We were strongly limited regarding the computational resources at our hand : we could hardly reach $N = 2000$ monomers even if most of the exotic behaviors happen with much larger polymers. Also, our number of Monte Carlo trials (to obtain convergence) were also solidly limited by the size of the polymer as we will explain in the subsequent analysis. In Appendix §A, the reader will find some other inevitable problems with every implementation of PERM that we did manage (or not) to handle.

4.1 Heatmap and quick visualizations

First, we created a heatmap in order to visualize our polymers on a 3D graph. The heatmap gives an idea of where the polymers tend to go the most. The grid points get darker and bigger for points on which polymers pass more frequently. As a first evaluation tool for our method, it gives us a rough estimate of the end-to-end length to compare with the theoretical value. It also opens an interpretation of the PERM algorithm. It lets us see every PERM tour as a "tree" in a forest that starts at (0,0,0). We can then see the density of the "foliage" or lack thereof. We can indeed observe the difference between a simulation that explores only one part of the configurations and one that explores a wider array in Figure 2. In 2b, it is clear that most polymers took the same path that stands out in black, so that means the sampling was mediocre. In 2a, the more round shape and the numerous red paths indicate that while a lot of polymers passed through these red paths, the configurations were explored more thoroughly.

4.2 Scaling laws with SARW

The robustness of our implementation is tested with universal scaling exponents of the self-avoiding random walk (cf. 2.4). This test has already been verified by the method's creators [12] and the found behavior was as follows : logarithmically with respect to N, the ratio $\frac{1}{N^{2\nu}}\langle r_{e/g}^2 \rangle$ grows logarithmically until $N = 1000$ before stabilizing around an horizontal asymptote. In that regard, our results are plotted on Figure 3. The logarithmic growth is almost seen as long as we accept the large error bars. The discrepency at large N comes from the the ratio of pruned/cloned polymers : this ratio is strongly influenced by the thresholds as explained in §A.2.3. For large and intermediate N, the simulation does not explore the configuration space as efficiently as for the small ones, which explains the much larger error bars. This phenomenon is understandable by an over-run of end-of-chain clones. Even if not fine-tuning the relevant program's parameters, we could have still achieved partial convergence by replacing global averages with sequential averages. In the process of generating a polymer of size N, we generate all the smaller polymers. Therefore, we use a single run to compute all sizes observables. However, despite this procedure improving the computation time, it is not perfect since



(a) Parameters : poly per run=250, runs=2

(b) Parameters : poly per run=500, runs=1

Fig. 2. Heatmaps for two different simulations that contain the same amount of polymers of same length, but generated with different parameters. `poly per run` represents the number of polymers in a single run and `run` is the number of sequential runs achieving decorrelation. More information is available in §A.2.

all sizes have their own optimal PERM parameters. The correct calculation would be to compute a single run for each desired point on the scatter plot.

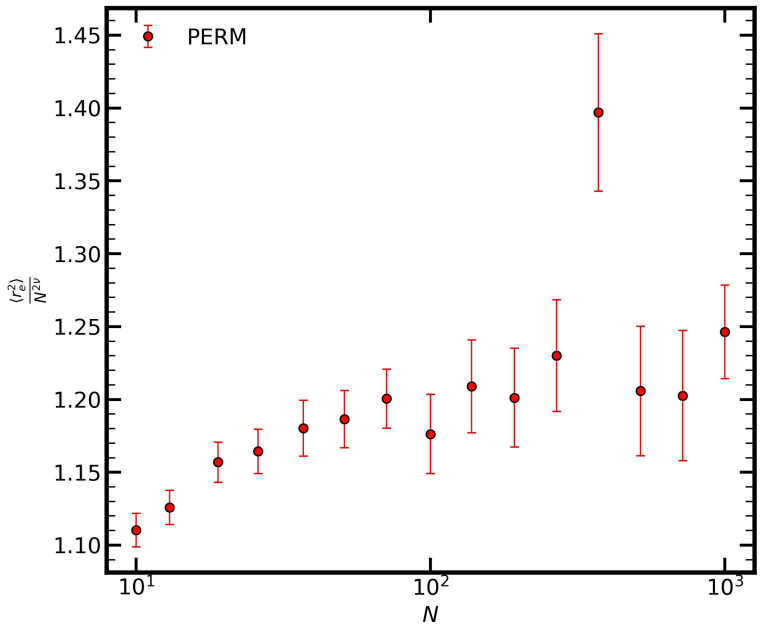


Fig. 3. Mean end-to-end distance square $\langle r_e^2 \rangle$ as a function of the number of monomers N with SARW.

4.3 Coil-globule transition (ISAW)

From the ISAW model, we represented the interaction energy q using the Boltzmann factor $e^{\beta\epsilon}$. This factor influences the interactions between monomers and therefore, the final structure of the polymers. Curled-configuration will be thermally favorized since they tend to yield a bigger number of non-bonded closest-neighbors pairs. There is still a balance because curling favorizes attrition. Figure 4 presents the mean of the end-to-end square distance as a function of N for different Boltzmann weights. It emphasizes that the more important the q parameter is, the less important the end-to-end square distance is, and therefore, the more compact the polymer structure is. Indeed, a positive value of the Boltzmann factor corresponds to attractive interactions between monomers, which favorizes the formation of globule

structures. That is why, it is possible to observe the coil-globule transition by studying the evolution of the end-to-end distance as a function of the Boltzmann weight. If we reason in terms of temperature, the polymer has a tendancy to coil at higher temperatures, that correspond to lower q's.
We must highlight the remarkable agreement with the method's creators in Ref [12]. If we managed to simulate larger sizes, this gives hints on a more pronounced transition at the critical value $q_c = 1.3087$ (the dark-blue curve on Figure 4). All curves above will tend to grow faster and faster with lower q. The tendency is drastically different above this critical value : the end-to-end distance will tend to decrease towards 0 when N increasing. This behaviour is not captured by our simulation because it appears at much bigger sizes. The tendency is still almost visible : $q = 1.3087$ is clearly the most stable line between 100 and 1000 monomers.
Each q represents approximately 4 hours of CPU time and the results seem almost converged. Simulating larger polymers would have been difficult with the growing size of the configurational space of all self-avoiding random walks. An explanation on why the convergence seems so different when compared with §4.2 can be the following : PERM, if wrongly optimized, should capture almost identical high-weight polymers and never prune them. The problem with the SARW is that all polymer chains are equiprobable and we end up sampling a really tiny portion of the configuration space. This tendency is not seen with the ISAW because high-weight polymers are curled and curling favorizes attrition ! After some step the cloning will eventually stop because there is a balance coming from the gradual regression in the number of free neighbors.

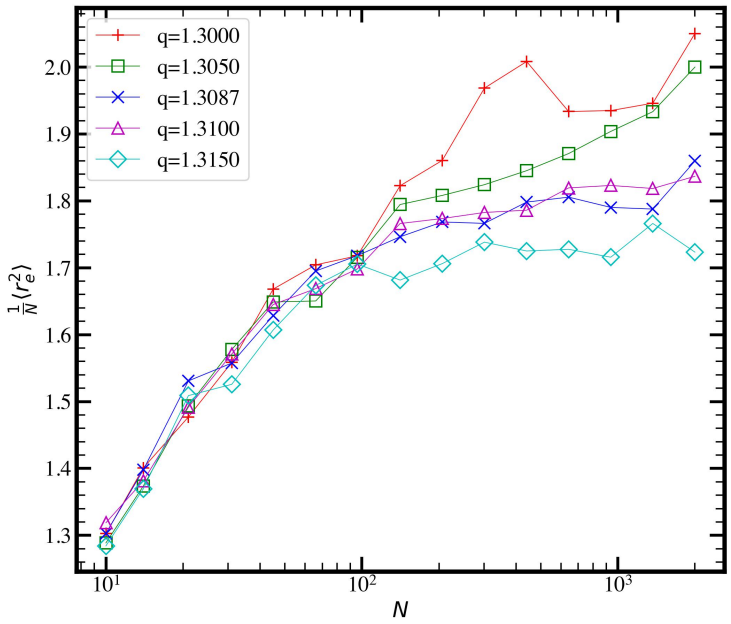


Fig. 4. Mean end-to-end squared distance as a function of N with different ISAWs. Each q relates to a different Boltzmann factor $e^{\beta\epsilon}$.

4.4 Prospective non-converged results

4.4.1 BISAW

We finally tried to observe the coil-globule transition in a poor-solvent : $q = 1.5$ gives highly collapsed polymers. Figure 5 shows the mean extension of different sizes of polymers when an external force is applied on the first and last monomer. Our simulation is clearly unconverged and the results seem unexploitable. Still, one might want to take a look at the trend followed for the computed forces. At low force, we retrieve the collapsed behavior. At high force, the extension should scale linearly with N since the force completely dominates the entropic force collapsing the molecule, and the polymer is mainly subjected to the external force extending it. As for [12], one must observe a phase transition at a

critical value of the boltzmann force $b \approx 1.6/1.65$. This phase transition is visible at $N > 6000$ monomers which is far above our computational limit. We still observe the two limit phases (max and min forces) but the intermediate force values yield so much fluctuation the curves do not show any clue of a coil-globule transition. As mentionned in §A.2, the optimization of PERM is difficult and we did not have the time at the end of our project to fine-tune PERM parameters for various values of b. We still managed to implement a BISAW which rendered a stretched coil at high force. We hope to present nicer results during our presentation.

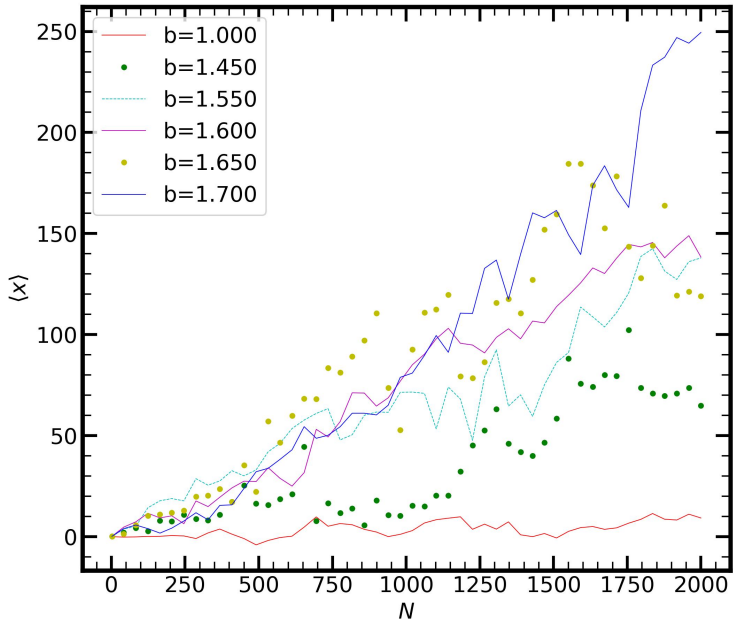


Fig. 5. Mean extension in the x direction as a function of N with different BISAWs. Each b relates to a different Boltzmann factor $e^{\beta f}$.

5 Conclusion

During this project, we had the opportunity to create our own computer program for polymer simulation. We based our code on a self-avoiding random walk, from which we implemented a Monte Carlo sampling method using the prune-enriched Rosenbluth Method. This method is very resource-intensive : previous studies have showed a 3500 hours of CPU time with 100 000 monomers [13] ; yet it effectively explores the configuration space. Our code is a versatile module able to simulate three types of polymer models on a lattice. The first model, based on SARW, was benchmarked with universal scaling laws that gave us something to compare our results with. On the one hand, this model generally exhibits a good trend, but on an other, it is quite sensitive to simulation parameters, leading to some convergence issues that we sought to optimize. Then, the second model (ISARW) suggested a coil-globule transition but required significant computational resources to confirm the phase transition. However we managed to obtain significant results validating previous findings and demonstrating the proper functioning of our program. Finally, the BISARW model, envisaged for implementing an external force, requires even more fine-tuning, and thus we lacked the time and resources to explore it in detail. We have understood that the three models do not react in the same way with similar PERM parameters. For the SARW model, fine-tuning is required in excess, whereas the ISARW model adapts well because of its peculiar complexity. The BISARW model has to be drastically improved, one approach could be to create an automatic scheme finding the optimal PERM parameters reaching convergence or to compute one simulation per desired polymer length.

Annexes

A Encountered problems

A.1 The need for smaller weights

A problem that we encountered during the implementation was that weights grow exponentially with the number of monomers N. In fact, at each step, the weights get on average approximately 4,8 [2] times bigger. This means that for N monomers, the weights are of the order of at least $4,8^N$. In the worst case scenario, the exponent multiplied to the number of neighbors bloats the weights even more.

The solution we found that worked best to solve this problem was to calculate only the logarithms of the weights. That way, the weight associated with a polymer is a sum of the logarithms calculated at each step. Instead of growing exponentially, the weight now grows linearly. We therefore define the new weights as :

$$w_i := \log(W(i)) = \sum_{n=1}^{N} \log\left(m_n^i q^{j_n^i)}\right). \tag{19}$$

Here, i is the polymer number and n the monomer number, and the logarithm is in base 10 for convenience. It should also be noted that the weight will also be defined as :

$$w_i := \sum_{n=1}^{N} w_n^i, \quad \text{and} \quad w_n^i = \log\left(m_n^i q^{j_n^i}\right). \tag{20}$$

similar to $W_i := \prod_{n=1}^{N} W_n^i$ with $W_n^i = m_n^i q^{j_n^i}$. For the same reason, we will calculate $\log(Z)$ instead of Z :

$$\log(Z_n) = \log\left(\frac{1}{M}\sum_{i=1}^{M} m_n^i q^{j_n^i}\right). \tag{21}$$

We encounter a new problem here : how can one compute the sum of the weights inside the logarithm ? As each weight will be too big for python, the sum will be even worse. The sum therefore has to be represented by a single weight. To achieve that, every weight needs to be written as a factor multiplying the maximum weight (the maximum weight was chosen for convenience and to maximise the calculation power of our method). The equation can then be rewritten as :

$$\log(Z_n) = \log\left(\frac{1}{M}\right) + \log\left(\sum_{i=1}^{M} W_n^i\right) = \log\left(\frac{1}{M}\right) + \log\left(\sum_{i=1}^{M} a_i W_n^{max}\right). \tag{22}$$

Then, since W_n^{max} does not depend on i, it can be taken out of the sum, and the logarithm can be separated, so the equation becomes :

$$\log(Z_n) = \log\left(\frac{1}{M}\right) + \log\left(\sum_{i=1}^{M} a_i\right) + w_n^{max}. \tag{23}$$

with $w_n^{max} = \log\left(W_n^{max}\right)$.
To determine the a_i coefficients, one only needs to notice that

$$W_n^i = a_i W_n^{max}$$

$$a_i = W_n^i / W_n^{max}.$$

2. This number comes from solving total_weight = X^N with a weight small enough to be calculated. The precision on the number comes from [14].

$$a_i = 1/10^{\log(W_n^{max}) - \log(W_n^i)} \tag{24}$$

This deals with numbers small enough to be calculated as long as weights are of similar order of magnitude. This should be the case for every polymer that has a length N for the scale we want to simulate at. If we wanted to go higher, it would be simple to implement a restriction on this difference of logarithms after which the coefficient a_i is set manually to 0.

Now, to calculate the observable, one needs to use these new variables. We can rewrite the latter part of equation 6 as :

$$\frac{\sum_{i=1}^{M} A(\ell_i) W(\ell_i)}{\sum_{i=1}^{M} W(\ell_i)} = \sum_{i=1}^{M} \frac{W(\ell_i)}{\sum_{i=1}^{M} W(\ell_i)} A(\ell_i) = \sum_{i=1}^{M} 10^{w_N^i - \log(M) - \log(Z_N)} A(\ell_i) = \langle \hat{A} \rangle_{\psi} \tag{25}$$

with $W(\ell_i) = 10^{w_n^i}$ and $\sum_{i=1}^{M} W(\ell_i) = MZ_n = 10^{\log(M)+\log(Z_N)}$.

A.2 The perpetual adjustment of PERM parameters

It is important to know that to get our results, we had to adapt our algorithm to improve the sampling with the five key adjustable parameters allowed by our program :

1. `N` : number of monomers in a single polymer
2. `poly_per_run` : number of polymers in a single run
3. `runs` : number of runs in a single simulation (number of times the simulation is restarted for decorrelation of overall sampling)
4. `c_m` : lower threshold factor for the PERM algorithm
5. `c_p` : higher threshold factor for the PERM algorithm

A.2.1 Choosing N

The first parameter to set was N. We had to decide the N that we wanted to simulate. Ultimately, the time we had and our optimisation didn't allow us to go past N=2000, so that is where we settled to produce the results. In some of the papers we found during our research, they pushed N up to millions[13], but they had to run simulations for a significant amount of time (thousands of hours) to achieve their results, and they probably had access to better computers as well.

A.2.2 Separating the runs

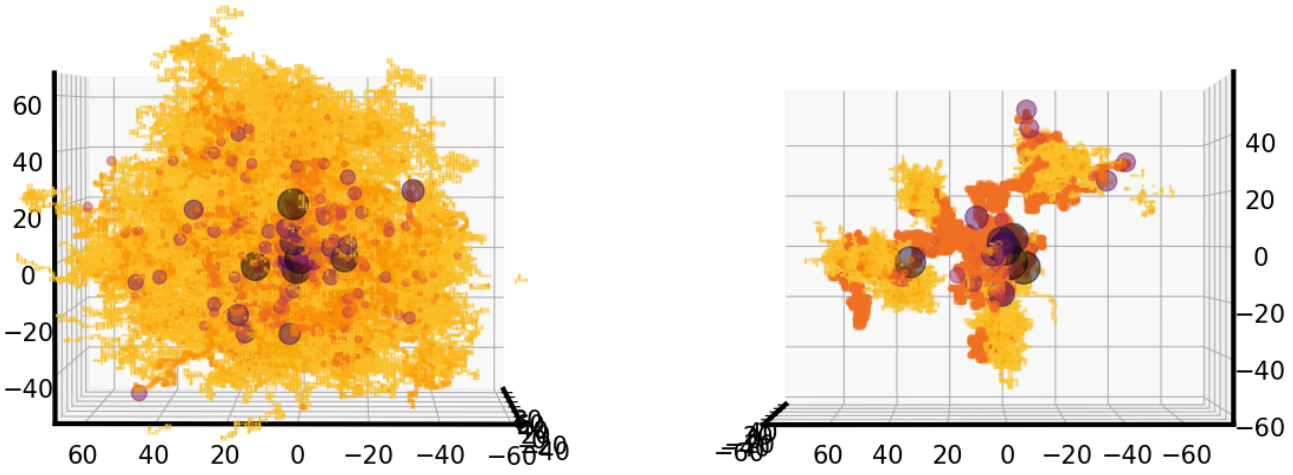
The parameters `poly_per_run` and `runs` go together to dictate how many polymers will be simulated for one simulation. The reason it is separated into runs is that it decorrelates the polymers. It allows them to go to a new tour if the active tour is taking too long. The number of runs can be seen as the minimum amount of tours in one simulation. To explore the possible configurations better, it is optimal to fill the space with clones of sizes well distributed across the length of the polymer. In other words, the foliage density has to remain constant throughout the polymer's length. This is achieved by having the clones appear uniformly inside the 2000 monomers range. This means a good balance between cloning and pruning has to be achieved.

A.2.3 Regulating cloning and pruning

To really control the PERM algorithm, `c_m` and `c_p` are the main factors, both being multiplied to the partition function Z to form the lower and higher threshold respectively. If the lower threshold is higher, the polymers will be pruned more frequently, and if the higher threshold is lowered, they will be cloned more easily. The key is to find the coefficients that

will bring balance to the cloning and pruning.

However, this is easier said than done. When regulating cloning, one has to be aware that cloning too much means only exploring the tip of the chains, because since we circle back to clones using a depth-first algorithm, if there are too many clones we will never reach the first clones that were made along the way (the ones that have more monomers left to generate) before the end of the run.



(a) Parameters : c_m=0.05, c_p=6.5

(b) Parameters : c_m=0.1, c_p=1.5

Fig. 6. Heatmaps for two different simulations of N=1000 monomers, 10 runs and 200 polymers per run

In figure 6, the lower and higher thresholds were altered. In 6a, we have a properly cloning PERM, and in 6b, there is too much cloning. To make the cloning go down, there are two options. First, we could increase the higher threshold to make cloning more difficult. But we could also increase the lower threshold, so the death rate. The second option will however make it more difficult for the generating polymer to make it to the chosen N. In fact, if N is big, and that is the case with our N, a small probability of pruning at every step will make the surviving rate plummet. The generation could get stuck, being pruned even before creating a clone. But not pruning enough might lead to too many clones surviving and the sampling would then be worse, especially for the intermediary N's.

In addition to this sensibility, the optimal parameters were not the same for every N, and neither were they for every q or every b. The fine tuning required a lot of time and had we had the computing power to do so, a grid-search would have been useful.

B Source code

The GitHub repository is accessible with this link. It contains a README in which information is given about how to run the program adequately to produce results.

Références

[1]M. Tabrizi, "Rosenbluth algorithm studies of self-avoiding walks", Physics, Materials Science (2015).

[2]G. Maurel, "Simulations multi-échelles de matériaux polymères", thèse de doct. (Blaise Pascal, Clermont-Ferrand II, 12 juin 2014).

[3]A. P. Thompson, H. M. Aktulga, R. Berger, D. S. Bolintineanu, W. M. Brown, P. S. Crozier, P. J. in 't Veld, A. Kohlmeyer, S. G. Moore, T. D. Nguyen, R. Shan, M. J. Stevens, J. Tranchida, C. Trott et S. J. Plimpton, "LAMMPS - a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales", Comp. Phys. Comm. **271**, 108171 (2022).

[4]M. Abraham, A. Alekseenko, V. Basov, C. Bergh, E. Briand, A. Brown, M. Doijade, G. Fiorin, S. Fleischmann, S. Gorelov, G. Gouaillardet, A. Grey, M. E. Irrgang, F. Jalalypour, J. Jordan, C. Kutzner, J. A. Lemkul, M. Lundborg, P. Merz, V. Miletic, D. Morozov, J. Nabet, S. Pall, A. Pasquadibisceglie, M. Pellegrino, H. Santuz, R. Schulz, T. Shugaeva, A. Shvetsov, A. Villa, S. Wingbermuehle, B. Hess et E. Lindahl, "GROMACS 2024.1 Manual", 10.5281/ZENODO.10721192 (2024).

[5]S. Jo, T. Kim, V. G. Iyer et W. Im, "CHARMM-GUI : A web-based graphical user interface for CHARMM", Journal of Computational Chemistry **29**, 1859-1865 (2008).

[6]P. Šulc, F. Romano, T. E. Ouldridge, L. Rovigatti, J. P. K. Doye et A. A. Louis, "Sequence-dependent thermodynamics of a coarse-grained DNA model", The Journal of Chemical Physics **137**, 135101, 135101 (2012).

[7]K. Nagai, "Non-gaussian character of real polymer chains", The Journal of Chemical Physics **38**, 924-933 (1963).

[8]H.-P. Hsu et P. Grassberger, "A review of monte carlo simulations of polymers with PERM", Journal of Statistical Physics **144**, 597-637 (2011).

[9]T. Prellberg, J. Krawczyk et A. Rechnitzer, "Polymer simulations with a flat histogram stochastic growth algorithm", in *Computer simulation studies in condensed-matter physics XVI*, t. 103, sous la dir. de D. P. Landau, S. P. Lewis et H.-B. Schüttler, Series Title : Springer proceedings in physics (Springer Berlin Heidelberg, 2006), p. 122-135.

[10]D. F. Gatz et L. Smith, "The standard error of a weighted mean concentration-i. boostrapping VS other methods", Atmospheric Environment **29**, 1185-1193 (1995).

[11]A. Ferguson et M. Z. Bazant, *Lecture 19 : polymer models : persistence and self-avoidance*, 14 avr. 2005.

[12]H.-P. Hsu, "Polymer simulations with pruned-enriched rosenbluth method i",

[13]P. Grassberger, "Pruned-enriched rosenbluth method : simulations of polymers of chain length up to 1 000 000", Physical Review E **56**, 3682-3693 (1997).

[14]R. Bauerschmidt, H. Duminil-Copin, J. Goodman et G. Slade, *Lectures on Self-Avoiding Walks*, 2012.