Internship report M2 – 2023

A probabilistic approach to diffusion-mediated surface phenomena

Yilin YE

SUPERVISOR: Denis GREBENKOV - *Laboratoire de Physique de la Matière Condensée (UMR 7643), CNRS—École Polytechnique, IP Paris, 91128 Palaiseau, France*

Abstract
We consider the particles' diffusion inside a complicated environment. As simplified models of bioactive surface with the geometric complexity, we select 2D convex Koch snowflakes as boundaries. A general procedure to generate Koch snowflakes with an arbitrary angle parameter has been firstly implemented, and then one Koch snowflake created on equilateral triangle is studied as a boundary for its relevant properties of the harmonic measure.
For numerical practices, we employ the "Geometry-adapted fast random walk" algorithm for diffusion process, to increase simulations' efficiency. Also, we consider the perfect or partial reactivity on the boundary surface. For the first case, particles are trapped after their first arrival; while for the second one, we model the reactive surface with "boundary local time", i.e. the random number of encounters prior to a successful reaction. When a particle approaches the boundary, we exploite the "reflection" method, which is verified well on the circular domain for correct probability distributions.

Keywords
Diffusion, Complex geometry, Harmonic measure, Boundary local time, Reactivity

Introduction

The importance of diffusion in chemical reactions had been first recognized by von Smoluchowski in 1918 [1] and then diffusion-controlled reactions or, more generally, diffusion-mediated surface phenomena, became a broad field of intensive research, with examples ranging from oxygen capture by lung alveolar surface to heterogeneous catalysis, gene regulation, membrane permeation, and filtration processes (Fig. 1). The involved species (atoms, ions, molecules, and even whole organisms such as bacteria) have first to encounter each other, or to find a specific target or a substrate, to initiate a reaction. Most former works focused on the role of this first-passage step and its dependence on the geometric structure of the environment, on the size and shape of the target, and on the type of diffusive process. In turn, the role of surface reactions, occurring after the first arrival of a particle onto the target remained underestimated. In practice, one successful reaction event is generally preceded by a series of failed reaction attempts on the target, and thus involves multiple diffusive excursions in the bulk, whose statistical description is still missing.

Herein, a novel probabilistic approach based on the concept of boundary local time has been recently proposed to investigate the intricate dynamics of diffusing particles near a reactive target [2]. This general paradigm allows us to describe sophisticated surface reactions controlled by random encounters between particles and the specific target, such as passivation of catalysts or activation processes in microbiology such as signaling in neurons, synaptic plasticity, cell differentiation and division. The disentanglement of the geometric structure of the medium from its surface reactivity opens far-reaching perspectives for modeling, optimization, and control of diffusion-mediated surface phenomena in nature and industry.

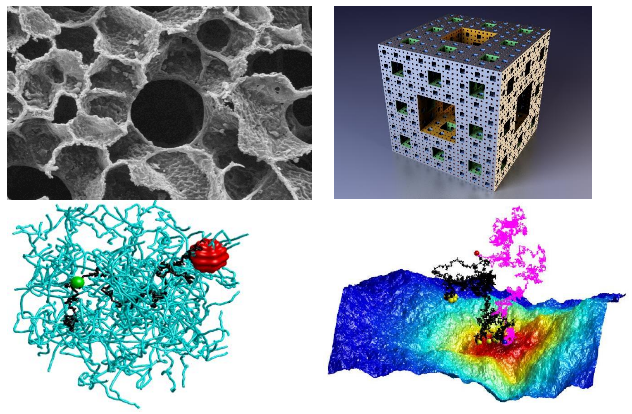


Figure 1. Top left: a cut of the pulmonary acinus exhibits a complex microstructure that determines the oxygen capture by blood and thus controls human respiration (credits to E. Weibel). Top right: the Menger sponge, a geometric model of a multiscale porous medium. Bottom left: a random trajectory of a bioactive molecule (in green) towards a protein (in red) through a network of actin filaments (in light blue). Bottom right: a particle diffusing near a reactive surface exhibits numerous encounters with that surface.

Known examples of the studied environments are limited to simple shapes such as a circle or a sphere, for which exact solutions are available; while more complicated confining media in real cases should therefore be explored by numerical methods to reveal the impact of their geometric structures onto diffusion-controlled reactions. This step also opens a possibility to address optimization problems of constructing efficient structures under specific surface reaction constraints. This basic problem can find applications in pharmaceutical industry, like programmable drug release.

For this internship, we consider diffusion-mediated surface phenomena by means of the encounter-based approach. To be exact, we developed an efficient numerical method based on "Geometry-adapted fast random walk" algorithm, to incorporate diffusion of particles inside a complex boundary environment, with the perfect or partial reactivity on its boundary. Also, the concept of the boundary local time related to reactivity plays a paramount role for partially reactive surface, where as the core strategy, the encounter-based approach is applied to several specific cases of interest, apart from the the validation on simple geometry. e.g. a circular domain.

Theoretical Background

Diffusion process

In order to simulate the diffusion process inside a cell, we suppose that the internal environment keeps homogenous, and ignore the possible movement of cell membranes like exocytosis (Fig. 2). Indeed, near a soft surface, we should take its deformation into account [3, 4], for the lift force on particle's motion. For the sake of simplification, we regard all activities inside cell as four parts [5]: bulk diffusion, bulk reaction, surface diffusion, and surface reaction (Fig. 3), neglecting all interactions within cell organelles.

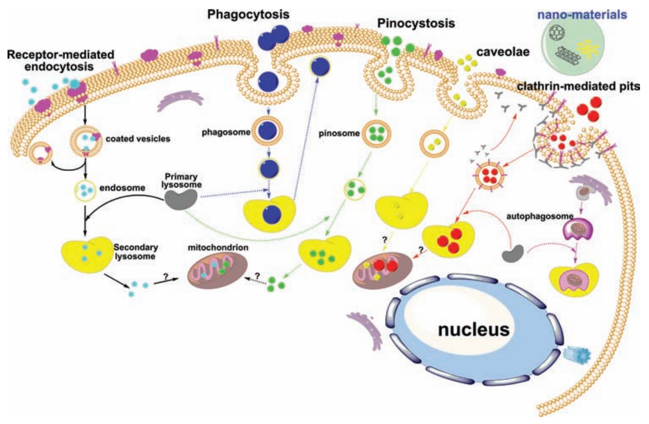


Figure 2. Known pathways for intracellular uptake of nanoparticles. [6]

Therefore, we aim to to exploit the pre-fractal boundaries as simplified models of the geometric complexity of cell membranes, since it is easier to simulate due to symmetry. In these circumstances, biological active molecules are regarded as point-like particles, while cell membranes are modeled by fractal boundaries, e.g. Koch snowflakes.

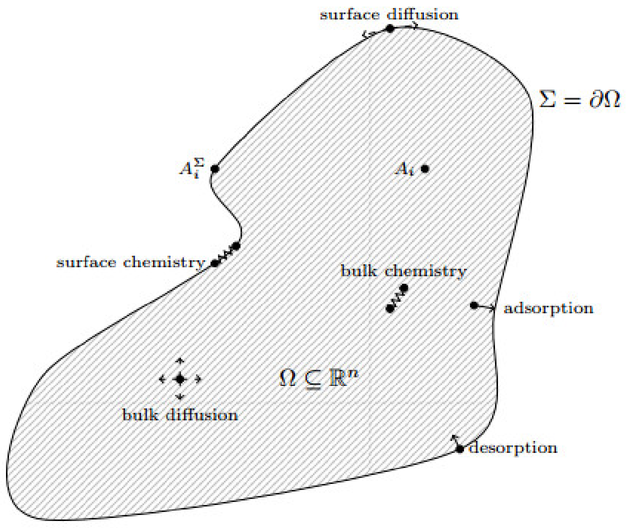


Figure 3. Physical and chemical mechanisms in a bulk-surface reaction diffusion system. [5]

Inside an homogenous environment, this motion could be modeled by Langevin equation [7]. We consider the random trajectory X_t of a particle with external force $\mathbf{F}$ at temperature T:

$$\mathrm{d}\mathbf{X}_t = \frac{D}{k_\mathrm{B}T}\mathbf{F}(\mathbf{X}_t)\mathrm{d}t + \sqrt{2D}\mathrm{d}\mathbf{W}_t \tag{1}$$

where $\mathbf{W}_t$ is a standard Brownian motion, D is the diffusion coefficient, and $k_\mathrm{B}T$ represents thermal energy of the bath. To implement reflections of the particle on the boundary, $\mathbf{F}(\mathbf{x})$ should be non-zero only near the boundary, forcing the particle to move away from the boundary back to the interior, like

$$\frac{\mathbf{F}(\mathbf{x})}{k_\mathrm{B}T} = \frac{1}{\varepsilon}\mathbf{n}(\mathbf{x})\mathbb{I}_{\partial\Omega_\varepsilon}(\mathbf{x}) \tag{2}$$

where $\mathbf{n}(\mathbf{x})$ is the normal vector to the boundary $\partial\Omega$, and $\partial\Omega_\varepsilon$ refers to the boundary layer of width ε, inside which the force $\mathbf{F}$ acts.

Koch snowflakes

We consider a general approach to create Koch snowflakes. Given a segment with length L, we insert three additional points based on two endpoints with an arbitrary angle $\alpha \in (0,\pi)$ (Fig. 4), replacing the original segment by four new segments. If these four segments have the same length l, we

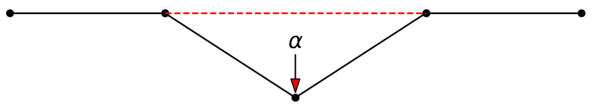


Figure 4. Portion of Koch snowflake with arbitrary angle. The length of four black segments is l, while the length of red dashed line is d.

could obviously obtain eq. (3):

$$\begin{cases} d^2 = l^2 + l^2 - 2l^2\cos\alpha \\ d + 2l = L \end{cases} \tag{3}$$

and thus the solution reads:

$$\begin{cases} l = \dfrac{L}{\sqrt{2(1-\cos\alpha)}+2} \\ d = \dfrac{\sqrt{2(1-\cos\alpha)}L}{\sqrt{2(1-\cos\alpha)}+2} \end{cases} \tag{4}$$

With these lengths known, we could generate Koch snowflakes by: initial polygon shape, angle α, direction (concave or convex), and level of generation g. We do not care the size or the initial segment length L for perfect reactivity. Then we nominate a Koch snowflake such that "6_$\pi/3+4$" refers to a fractal configuration generated on hexagon, with angle $\alpha = \pi/3$, concave ($-$ for convex), up to the fourth level.

For instance, starting from an equilateral triangle (Fig. 5, Top left), we choose $\alpha = \pi/3$ and add new points on each side based on eq. (4) and necessary rotations, to form a convex configuration. Thus, a series of configurations would be generated with names 3_$\pi/3-g$, where g refers to the level of generation. Then, 3 long segments are substituted by 12 new segments (Fig. 5, Top right). Here, we say that this new shape is generation $g = 1$, that points and segments were inserted **once**. Still, we repeat this process, such that from $g = 1$ to $g = 2$, the second operation results in 3_$\pi/3-2$ (Fig. 5, Bottom left), while from $g = 2$ to $g = 3$, we see this operations three times for 3_$\pi/3-3$ (Fig. 5, Bottom right); higher generations could be got by more repetitions. Moreover, we could also start from other polygons, such as square, pentagon, etc. Below are some example in Fig. 6.

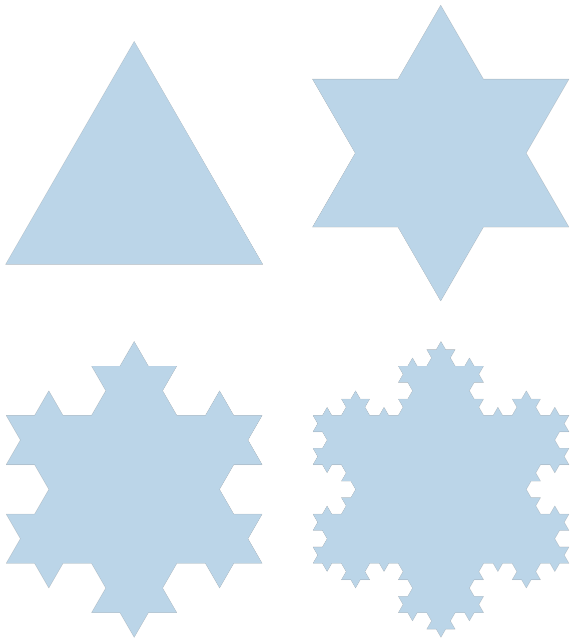


Figure 5. Procedure to generate convex Koch snowflakes. Top left: An equilateral triangle with a given length L. Top right: Level $g = 1$, $3\_\pi/3 - 1$. Bottom left: Level $g = 2$, $3\_\pi/3 - 2$. Bottom right: Level $g = 3$, $3\_\pi/3 - 3$.

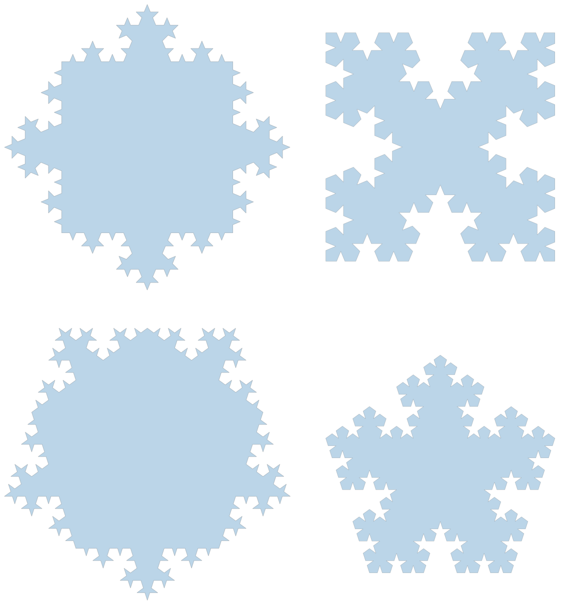


Figure 6. Examples of Koch snowflakes. Top left: $4\_\pi/4 - 3$. Top right: $4\_\pi/4 + 3$. Bottom left: $5\_\pi/5 - 3$. Bottom right: $5\_\pi/5 + 3$.

Note that, particles could be inside or outside of the cell. Then considering the direction of our complex boundary such as convex or concave, there would be 4 possible cases for diffusion near a fractal domain (Fig. 7). During this internship, we only consider the first case where the particle is located inside the convex boundary, which is the classical Koch snowflake based on equilateral triangle with angle $\alpha = \pi/3$, i.e. $3\_\pi/3 - g$ (Fig. 7, Top left).

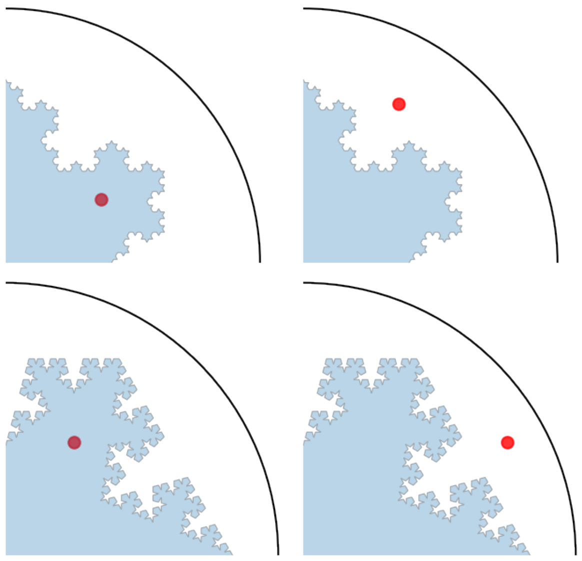


Figure 7. Possible cases for diffusion of a particle (in red) near Koch snowflake boundary (in blue) and circular domain (in black). Top left: Particle inside a convex boundary. Top right: Particle outside a convex boundary, but inside a large circle. Bottom left: Particle inside a concave boundary. Bottom right: Particle outside a concave boundary, but inside a large circle.

Harmonic measure

The harmonic measure of a subset of the boundary of a bounded domain in n-dimensional Euclidean space $\mathbb{R}^n, n \geq 2$, is the probability that a Brownian motion starting inside a domain hits that subset of the boundary, before hitting the remaining part of the boundary. In the complex plane, harmonic measure can be used to estimate the modulus of an analytic function inside a domain $\mathscr{D}$ given bounds on the modulus on the boundary of the domain.

Apart from few particular cases, the harmonic measure could not be derived analytically. Therefore, we have to seek it numerically, namely the distribution of p_k as the hitting probability on the k-th segment after particles' arrivals for their diffusion in Koch snowflake boundaries. As for how to label segment indices, see Fig. 10.

Reactivity and boundary local time

Generally, there would be many different acceptors situated on the cell membranes as the targets for surface diffusion or surface reactions. With the help of homogenous reactivity, multiple targets can be treated approximately as the same one, distributed uniformly on the surface in high enough density. However, particles near such a surface could not always find out the target at their first arrivals, while a longer trajectory including several arrivals on the surface but not reaching the target is also frequently observed in fact. Thus, we need a variable to describe this process, for instance, how much time a particle spent in seeking a target near the boundary, or how many times a particle hits the boundary. For each particle, it would enter the boundary layer from different positions, and thus leading to distinct time for reaching a target. Qualitatively, highly reactive surfaces would be teemed with all kinds of targets, which requires few or just only one contact for related reactions; while rather passive surface owns a limited number of targets for much longer time or more contacts to realize the reaction.

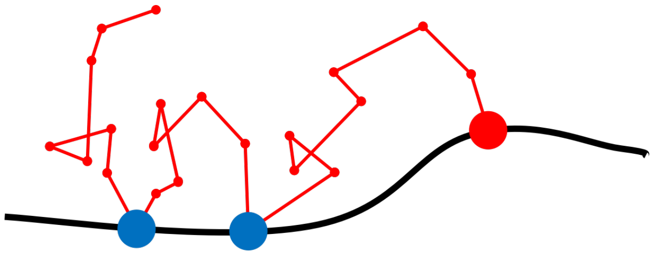


Figure 8. A possible Brownian motion trajectory of a particle (in red) near surface towards the target (in red), with two unsuccessful contacts (in blue) on the surface.

Following the paradigm [2], we introduce the successful reaction event with the help of "boundary local time". We define

$$\ell_t^\varepsilon = \frac{D}{\varepsilon}\int_0^t \mathrm{d}t' \mathbb{I}_{\partial\Omega_\varepsilon}(\mathbf{X}_{t'}) \tag{5}$$

and Langevin equation (1) turns to

$$\mathrm{d}\mathbf{X}_t = \mathbf{n}(\mathbf{X}_t)\mathrm{d}\ell_t^\varepsilon + \sqrt{2D}\mathrm{d}\mathbf{W}_t \tag{6}$$

So $\frac{\varepsilon}{D}\ell_t^\varepsilon$ refers to the residence time of $\mathbf{X}_t$ inside the boundary layer $\partial\Omega_\varepsilon$ up to time t. For thin layer limit, we denote the boundary local time ℓ_t as:

$$\ell_t = \lim_{\varepsilon\to 0} \ell_t^\varepsilon \tag{7}$$

Besides, the boundary local time ℓ_t can also be related to the number $\mathcal{N}_t^\varepsilon$ of crossings of the boundary layer $\partial\Omega_\varepsilon$:

$$\ell_t = \lim_{\varepsilon\to 0} \varepsilon\mathcal{N}_t^\varepsilon \tag{8}$$

Furthermore, due to the complex shape of the fractal boundary, we could hardly perform integration like eq. (5), or count number like eq. (8). Hence we use a local approach to calculate the boundary local time ℓ_t stepwise after the particle enters the boundary layer and hits the surface. See eqs. (12, 13, 14) in subsection Encounter-based reflection.

On the partially reactive region $\partial\Omega$, we employ the Robin boundary condition

$$-D\partial_n c(\mathbf{x},t) = \kappa_0 c(\mathbf{x},t) \tag{9}$$

with κ_0 is the reactivity of the region. If the particle attempts to react independently on each encounter with probability $p \simeq \varepsilon\kappa_0/D \ll 1$, so the probability of no surface reaction up to the n-th encounter reads:

$$\mathbb{P} = 1 - \sum_{k=1}^{n} p(1-p)^{k-1} = (1-p)^n \simeq e^{-pn} \simeq e^{-q\ell} \tag{10}$$

where $q = \kappa_0/D$ refers to reactivity parameter, and $\ell = n\varepsilon$. So we pose the assumption $p \ll 1$, and take an exponential probability density later.

Numerical Simulations

A convex Koch snowflake with angle $\pi/3$ has been chosen as the complex environment, i.e. $3\_\pi/3-g$ for level g. We start particles at the center of this fractal domain, and simulate their diffusions towards boundary segments. There is no interaction among particles, so their motions are independent with each other. Given a constant diffusion coefficient D, the internal medium is regarded as homogenous without external potentials or forces, and particles cannot leave the boundary. As we could hardly figure out the analytical solutions for hitting probability distribution (harmonic measure or spread harmonic measure), due to diffusion inside such a complex domain, and thus numerical simulations are strongly needed. To be exact, we consider two cases below:

- Diffusion until the particle is attached on the boundary, such that the distance is less than a given threshold. See subsection: Perfectly reactive boundary.
- Diffusion until the particle has passed enough time near the boundary, such that the boundary local time ℓ_t exceeds a given random threshold $\hat{\ell}$. See subsection: Partially reactive boundary.

Perfectly reactive boundary

Geometry-adapted fast random walk

Conventional Markov-chain Monte Carlo method would not be efficient enough, since small diffusion distance is needed for precise results, which leads to too long simulation time. For more details, see Appendix: Markov-chain Monte Carlo (MCMC) method.

We really need a method efficient enough, which could furnish a proper diffusion distance r in each step for precise results without particles leaving the domain. In fact, this diffusion distance would be limited by the minimal segment length, which would cause more steps far away from the fractal boundary. Therefore, to accelerate our numerical simulations by decreasing numerical steps spent in the middle due to little value of r, we are inclined to vary this distance according to the nearby surroundings, i.e. smaller value of r while the particle approaches the boundary.

This corresponds to the idea of "Geometry-adapted fast random walk" algorithm [8, 9]. We step further based on MCMC with a varying radius r_i for each step, such that $\Delta x_i = r_i \cos\theta_i$ and $\Delta y_i = r_i \sin\theta_i$ as $\theta_i = \mathrm{ran}(0,2\pi)$, a uniformly distributed variable in the interval $(0,2\pi)$. We calculate r_i at each step according to one particle's current location, and then jump it to one position located on this circle uniformly. As an instance, we illustrate the GAFRW inside a Koch snowflake $3\_\pi/3-4$ (Fig. 9). Shown in the order of red, cyan, green, purple, and black, a series of positions are drawn in solid points, and a series of circles are drawn in solid lines with dashed color lines as the radius. Except the first jump from center (in red) for the sake of numerical efficiency, i.e. $r_1 = L/4$, we always take the distance between particle and boundary for the radius. Once the radius r_i is less than a given value, say $\varepsilon = 10^{-3}L$ for L is the segment length, we assume that the particle is attached on the perfectly reactive boundary, and then stop the simulation.

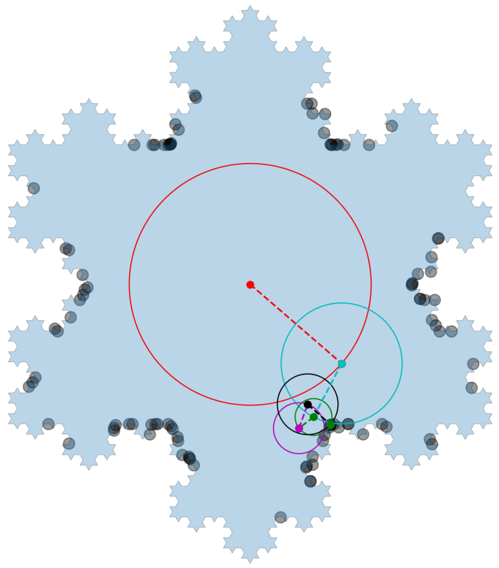


Figure 9. Brownian motion trajectory of a particle (in red) inside Koch snowflake $3\_\pi/3-4$ (in blue) by GAFRW algorithm. Final distribution after the first arrival on the boundary (in grey, 100 particles shown).

This approach is valid since Brownian motion is continuous on time. Also, with a fixed diffusion coefficient D, we compute the mean exit time $\langle\tau\rangle$ from a center of a circle of radius r to its boundary, by $\langle\tau\rangle = \frac{r^2}{4D}$, and then use it as the duration at one jump. By rotational symmetry, there should be a uniform distribution after the jump on this circle. Simply, we can replace a Brownian motion trajectory inside a circle by a random jump from center to a uniformly distributed point on the boundary. We repeat this process until the particle arrives on the boundary.

For this GAFRW algorithm, the most difficult part is to compute the distance between a given particle position and the whole complex boundary. Certainly, we could always compare all distances between the particle current position and all boundary segments or endpoints; however, it is not necessary to search among all of them. In this case, we employ a much faster method, only searching among relevant intervals. Briefly speaking, we record two nearest points in the level $g-1$, and only search segments inside this region in the level g for a proper radius r_i. After jumping uniformly on circle, we check whether the particle goes to another region, i.e. whether two nearest endpoints are not the previous ones. We go to the level $g+1$ if two nearest points in the level g are endpoints of a segment in level g, i.e. there is no other point between them. For more details, see Appendix: GAFRW algorithm.

Hitting probability distribution

With GAFRW algorithm in hand, we could simulate diffusion inside a complex domain efficiently. We do simulations for $N = 10^6$ particles to obtain an accurate empirical estimate of hitting probability distribution. We denote by p_k the hitting probability on the k-th segment. For simplicity of expression, we always label segments by numbers starting from bottom left (Fig. 10), i.e. the red one with index 1, the green one with index 2, the black one with index 3, etc. Other Koch snowflakes in higher levels are also labels in such a way.

Here we present results on Koch snowflakes $3\_\pi/3-2$ (Fig. 11), $3\_\pi/3-3$ (Fig. 12), and

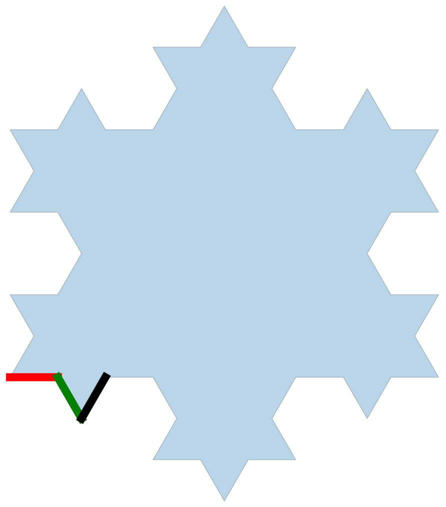


Figure 10. Koch snowflake $3\_\pi/3-2$. We label segment indices anti-clockwise starting from the bottom left corner, i.e. the red one is the segment 1, the green one is the segment 2, and the black one is the segment 3, etc.

$3\_\pi/3-4$ (Fig. 13). Obviously, there would be higher probabilities for segments closer to the center. Due to the domain axial symmetry, we see peaks with almost the same probability values. Also, due to the domain rotational symmetry, we see periodic peaks. Even larger number of particles would be needed for accurate results inside the domain of higher generation.

Note that, from $3\_\pi/3-2$ to $3\_\pi/3-3$, 48 old segments are split into 192 new ones, while there are lower probability values on each segment; similar for the case from $3\_\pi/3-3$ to $3\_\pi/3-4$. Even though it seems to exist some relations between two adjacent generations, there is no simple splitting law to infer probabilities in g from p_k in $g-1$, or construct probabilities in g by p_k in $g+1$. For more details, see Appendix: GAFRW algorithm.

Spread harmonic measure

In this subsection, we are interested in an extension of the harmonic measure, when many encounters occur prior to the reaction event. So we should continue numerical simulations even after the first arrival. There are three relevant choices to stop our numerical simulations:

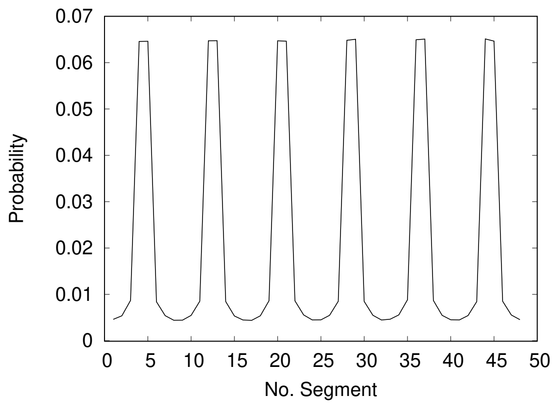


Figure 11. Hitting probability distribution as a function of segment index for Koch snowflake 3_$\pi/3-2$, $N=10^6$

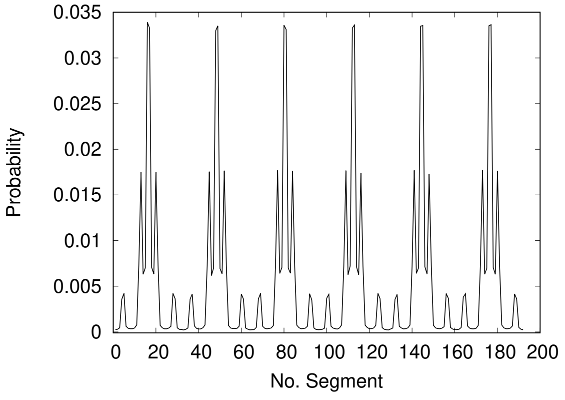


Figure 12. Hitting probability distribution as a function of segment index for Koch snowflake 3_$\pi/3-3$, $N=10^6$

- Stop the simulation after a given time T.
- Stop the simulation after a random time δ, such that $\mathbb{P}(\delta > t) = e^{-pt}$, where t is time passed for diffusion.
- Stop the simulation when the boundary local time ℓ_t exceeds a random threshold $\hat{\ell}$

We focus on the third choice and take $\hat{\ell}$ with the exponential probability: $\mathbb{P}\{\ell < \hat{\ell}\} = e^{-q\ell}$. Thus local time threshold is distributed with the density

$$\psi(\hat{\ell}) = qe^{-q\hat{\ell}} \tag{11}$$

where q is the reactivity parameter; in practice, we can generate the threshold as $\hat{\ell} = -\ln[\text{ran}(0,1)]/q$. See calculations in Appendix: Random threshold $\hat{\ell}$.

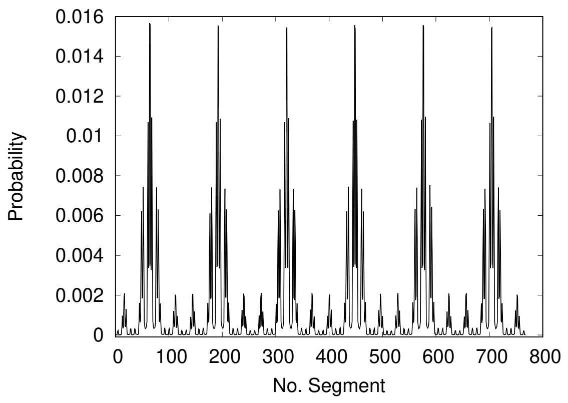


Figure 13. Hitting probability distribution as a function of segment index for Koch snowflake 3_$\pi/3-4$, $N=10^6$

Encounter-based reflection

Still, we employ GAFRW algorithm to treat the diffusion process as a random walk (Fig. 9), such as $\vec{x}_{k+1} = \vec{x}_k + \rho_k(\cos\theta_k, \sin\theta_k)$, where ρ_k refers to the radius, and θ_k is the uniformly distributed random angle. Here, $\vec{x}_k$ refers to the position after k steps, say $\vec{x}_0 = (0,0)$ at the center. Still, we compute the mean exit time of the k-th step τ_k, from a center of a circle of radius ρ_k to its boundary with a constant D by eq. (12)

$$\langle \tau_k \rangle = \frac{\rho_k^2}{4D} \tag{12}$$

In addition, we denote the (average) intermediate times t_k as the sum of average time after k-step diffusion by GAFRW:

$$t_{k+1} = t_k + \langle \tau_k \rangle \tag{13}$$

We would keep computing this variable. Besides, after entering the boundary layer $\partial\Omega_\varepsilon$, we do not stop simulation and consider the boundary local time ℓ_{t_k} computed by eq. (14) only after the particle hits the boundary at least once.

$$\ell_{t_{k+1}} = \ell_{t_k} + \sqrt{\frac{\pi}{2} \cdot D \cdot \langle \tau_k \rangle} \cdot \mathbb{I}_{\partial\Omega_\varepsilon} \cdot \mathbb{I}_{N_{\text{touch}}>0} \tag{14}$$

Here, ℓ_{t_k} is a non-decreasing variable depending on time t_k. After the first contact, we accumulate this variable for each following step until the particle leaves the boundary layer. Then we stop to accumulate it, until the next time the particle returns the boundary layer and reaches the boundary.

It is almost impossible for one particle to reach the boundary exactly with zero distance. Therefore, we employ the "reflection" method (Fig. 14) to treat this case for particles inside the boundary layer, in order to obtain an effective hit by following steps:

- Fix a radius ρ_k to be a prescribed constant ρ, e.g. a value proportional to the boundary layer thickness ε.
- Execute a uniform jump on the circle with this given radius ρ towards a new position.
- Once the new position is out of the domain, we **reflect** this position with boundary $\partial\Omega$ for another location inside the domain.
- Repeat this procedure until the particle exit the boundary layer $\partial\Omega$, then we take standard GAFRW.

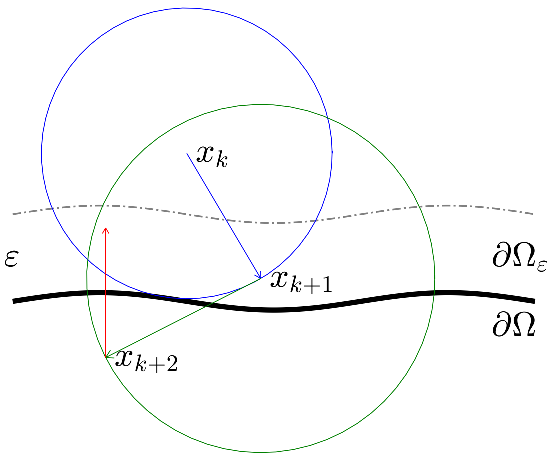


Figure 14. Illustration of several step of the walk-on-spheres algorithm [10], with real boundary $\partial\Omega$ and virtual layer $\partial\Omega_\varepsilon$. At x_k, we take one circle tangential to surface; while at x_{k+1}, we take $\rho \propto \varepsilon$. Once the particle leaves $\partial\Omega$ (at x_{k+2} on green circle), we exploit reflection backwards (in red arrow).

Since particles are independent with each other, each particle owns a unique threshold $\hat{\ell}$ with density in eq. (11). After a large enough boundary local time, such that $\ell_t \geq \hat{\ell}$, we halt the simulation for this particle and restart a new one for the next particle.

Verification on circle

Before the fractal surface, we should verify the reflection method. Here, we take a simple domain, a circle with radius $R = 1$, and we put each particle at $(r_0, 0)$ initially. For a disk, the spread harmonic measure density is known explicitly, and one can compute the hitting probability of any arc. Splitting the circle into N equal arcs, we get [11]

$$p_k = \frac{1}{N} + \sum_{j=1}^{\infty} \frac{2qr_0^j}{\pi j(j+q)} \sin\left(\frac{\pi j}{N}\right) \cos\left(\frac{2\pi j}{N}(k-\frac{1}{2})\right) \tag{15}$$

where q is reactivity mentioned previously.

To simplify the reflection without complicated calculations on tangential lines, we employ the following method as shown in Fig. 15 to realize the reflection near a circular boundary. Here, once the particle exits the domain, we do reflection just along the line linking this location and the center of disk, such that the distance of this outer position is just the one of the new position inside the boundary layer. Then we repeat until the particle exit $\partial\Omega$.

A priori, we need a reasonable radius ρ. One natural choice is just considering the relation as $\rho \propto \varepsilon$. Since the centering initial condition results in uniform p_k values according to eq. (15), we prefer to do simulations with $r_0 \neq 0$; larger r_0 results in larger difference among p_ks. Below we test several choices on a disk with $\varepsilon = 10^{-3}$ (Fig. 16), demonstrating that we should take $\rho = 2\varepsilon$. Once we take other ratios, there would be errors.

After our verification on a disk that ρ should be equal to 2ε, we step further towards different cases. In Fig. 17, a typical trajectory is shown with initial position at $(r_0 = 0.9, \theta = 0)$ and reactivity $q = 1$.

Furthermore, we test this method for different cases, with constant reactivity q, but varying initial positions (Fig. 18), or with the same initial position but varying reactivity (Fig. 19). All cases denote an excellent agreement respectively.

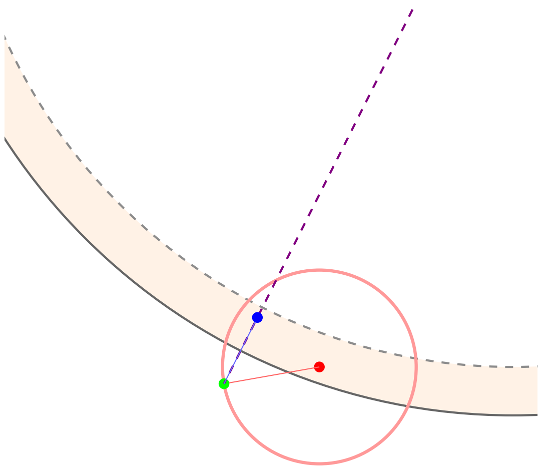


Figure 15. Illustration of reflection method on a circular boundary. If a particle (in red dot) is inside the boundary layer (in light orange) with the thickness ε, we draw a disk with radius $\rho = 2\varepsilon$. Once the new position (in green dot) jumps out the domain, we do reflection along the radius (in violet dashed line), towards the final position (in blue dot).

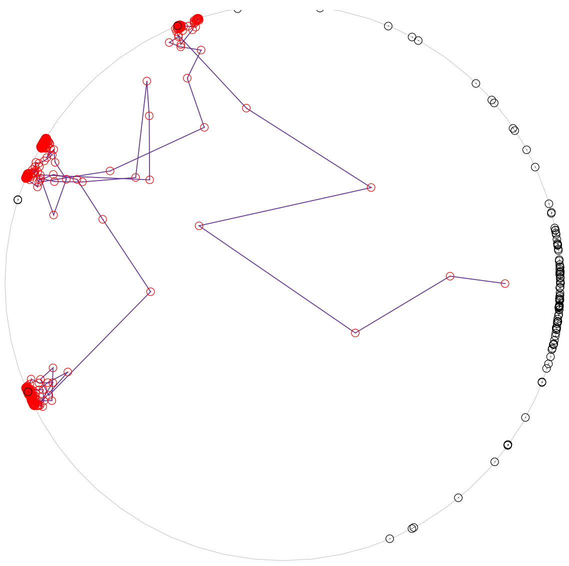


Figure 17. One random trajectory of a particle (in violet, red points for a series positions) inside a disk of radius $r = 1$ with reflection method with initial position $r_0 = 0.9, \theta = 0$. Hitting distribution for $q = 1$ on the circle (in black, 100 particles shown).

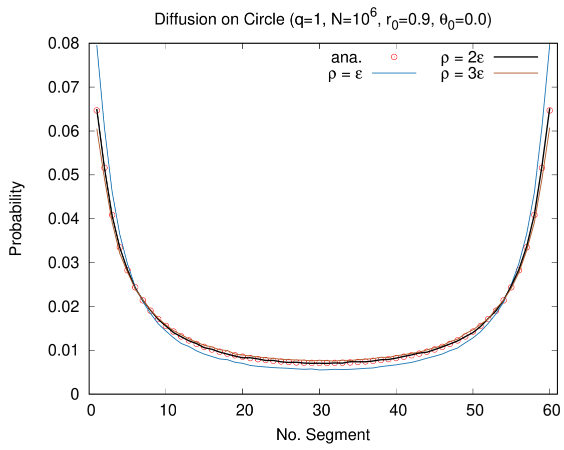


Figure 16. Comparison of the spread harmonic measures on a disk between the analytical expression (eq. (15), shown in red points) and numerical one (in color lines). Only for $\rho = 2\varepsilon$, we see an excellent agreement.

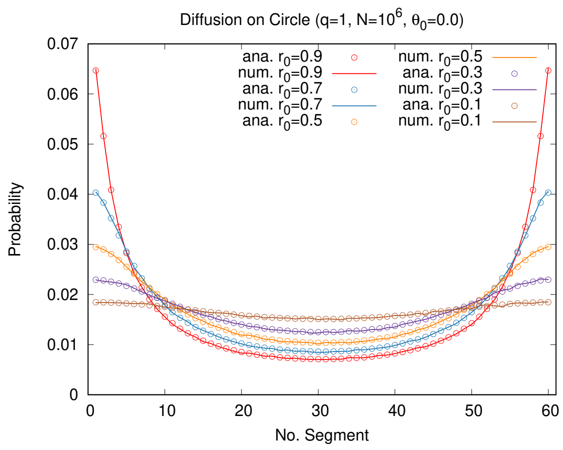


Figure 18. Spread harmonic measure in a disk of radius $R = 1$ under different initial position and a constant reactivity $q = 1$. Analytical expressions by eq. (15) are shown in dots, while numerical results are drawn in lines.

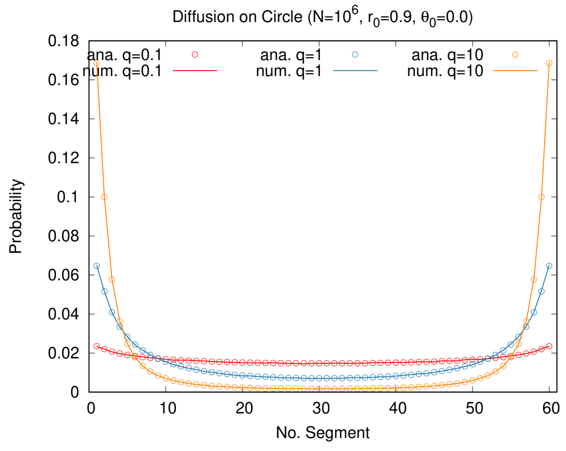


Figure 19. Harmonic measure in a circle domain with radius $R = 1$ under a constant initial position ($r_0 = 0.9$) and different reactivities. Analytical expressions by eq. (15) are shown in dots, while numerical results are drawn in lines.

Practice on Koch snowflake

We employ the reflection method directly on Koch snowflake $3\_\pi/3-2$. With a large enough domain, for example the length of equilateral triangular $L = 10^3$, we exam that once the boundary layer thickness ε is much less that boundary segment length L_g in g-th level, i.e. $\varepsilon \ll L_g = L/3^g$, there would be no significant difference for harmonic measure (Fig. 20).

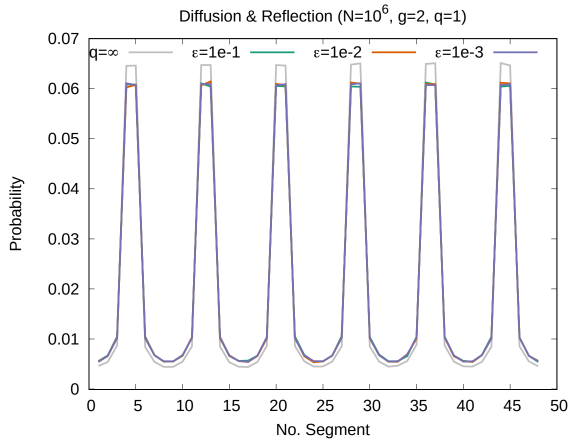


Figure 20. Comparison of harmonic measures inside the Koch snowflakes $3\_\pi/3-2$ with $L = 10^3$. The first-arrival result is shown in gray, while others cases are derived by reactivity $q = 1$.

Since $q = \kappa_0/D$, we should take qL together into account as a dimensionless parameter, as in the density eq. (11) qL is shown as exponent. Note that the domain size would affect the spread harmonic measure under a constant q. Indeed, the direct reflection method would meet numerical problems on domains with smaller length, such as $L = 2$.

Based on the original way, we always consider the radius as the distance between particle's current location and all boundary segments. For little qL, namely a ssmaller domain size, or a larger boundary local time threshold, it would be highly probable that the particle would be trapped in a corner. Therefore, the most difficult part is to define well the reflection rule near two types of corner: one for $\pi/3$, the other for $4\pi/3$. In practice, we modify the reflection method as shown in Fig. 21. Once one particle is deeply trapped in the corner, e.g. the distance is less than $10^{-9}\varepsilon$, we exploit a uniform jump with only partial range shown in Fig. 21.

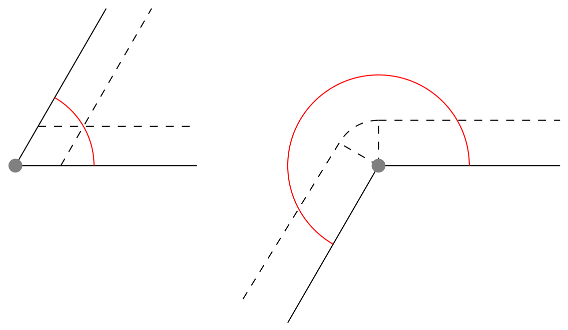


Figure 21. Possible reflections with $\rho = 2\varepsilon$ while particles are trapped at cornes. Black solid lines: boundaries; Black dashed lines: boundary layers with thickness ε; Grey points: corner vertices; Red arcs: diffusion range with radius $\rho = 2\varepsilon$.

Thanks to that modification, then we could continue to explore spread harmonic measures with small qL, e.g. $q = 1, L = 2$. One typical result derived inside $3\_\pi/3-2$ is shown in Fig. 22.

Conclusion and Perspectives

During this internship from April to July, an extensive work has been realized at PMC, on "A probabilistic approach to diffusion-mediated surface phenomena". Through various numerical simulations

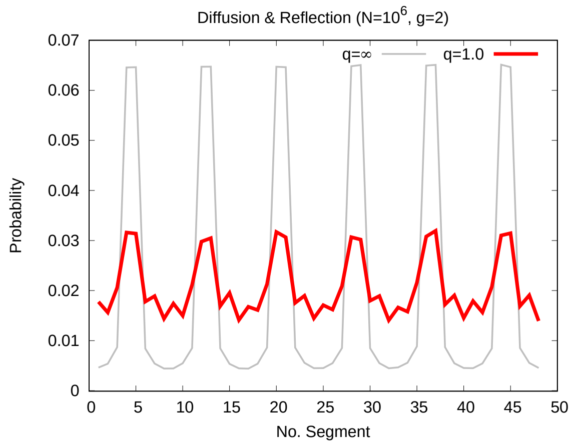


Figure 22. Comparison of harmonic measures inside the Koch snowflakes $3\_\pi/3-2$ with $L=2$. The result for $q=\infty$ is shown in gray, while the other is derived for $q=1$.

by "geometry-adapted fast random walk" technique, we have figured out the behavior of the (spread) harmonic measure inside a complex environment. Based on a Koch snowflake fractal boundary, we consider diffusion stopped after the first arrival, and then introduce the boundary local time for multiple arrivals, to mimic "target-finding" process, which plays a significant role for biochemical reactions. As the function of several parameters, probability distributions have been computed for both cases. Further research will be continued, focusing on different boundary conditions and reversible chemical reactions, and even surfaces in 3D.

While in the preliminary stage, we focus on developing theoretical and numerical aspects of the aforementioned phenomena, the ultimate goal consists in discovering applications of this framework in chemistry and biophysics. In this light, tremendous progress in optical microscopy over the last two decades opens unprecedented opportunities to follow the random trajectory of a single molecule and therefore to access experimentally the statistics of its encounters and interactions with the surface [**?**, **?**]. The increasing number of single-particle experimental data can thus help to reveal limitations of conventional approaches and to propose more accurate models of surface reactions. This ambitious goal requires elaborated statistical tools to characterize binding and unbinding events on the surface and to infer their statistics from a limited number of available noisy trajectories. While the development and applications of such tools to experimental data is a long-term project, the progress in understanding sophisticated surface reactions achieved later will prepare the theoretical ground and synthetic datasets of single-particle trajectories for future research.

Personally, I have practiced well my coding skill in such a short time range. Numerical simulations are driven by *Fortran* and *Python*, while figures are drawn by *Python* and *gnuplot*. To step further, the subsequent research will perhaps be concentrated on a boundary in 3D rather than a 2D one, or even that with irregular deformations. As for the soft surface, there exist numerous examples in the nature of such biological membranes.

Acknowledgments

Teemed with appreciation and gratitude, I would like to thank my supervisor Denis GREBENKOV at first, for his careful pedagogy, precise advice, and patient instruction during my whole internship. The prompt discussion could always give me effective feedback, not only the profile about the whole project, but also technical details in codes. Also, I thank Adrien CHAIGNEAU, the first-year PhD in our group, for his great insight on mathematical details, inspiring discussions, as well as his work on harmonic measure for necessary comparisons.

From April to present, I have been quite immersed in PMC, especially the warm group lunch. Furthermore, thanks to Anne-Marie DUJARDIN of PMC, École Polytechnique; as well as Médina MAHREZ of ENS for their generous help on administrative issues. In addition, we acknowledge the financial supports from CNRS.

References

[1] M. von Smoluchowski, *Ann. Phys.* **1915**, *48*, 1103–1112, **Über Brownsche Molekularbewegung unter Einwirkung äusserer Kräfte und deren Zusammenhang mit der verallgemeinerten Diffusionsgleichung**.

[2] D. S. Grebenkov, *Phys. Rev. Lett.* **2020**, *125(7)*, 078102, **Paradigm shift in diffusion-mediated surface phenomena**.

[3] T. Salez, L. Mahadevan, *J. Fluid Mech.* **2015**, *779*, 181-196, **Elastohydrodynamics of a sliding, spinning and sedimenting cylinder near a soft wall**.

[4] V. Bertin, Y. Amarouchene, E. Raphael, and T. Salez, *J. Fluid Mech.* **2022**, *933*, A23, **Soft-lubrication interactions between a rigid sphere and an elastic wall**.

[5] B. Augner, and D. Bothe, *arXiv:1911.13030*. **2019**, **The fast-sorption–fast-surface-reaction limit of a heterogeneous catalysis model**.

[6] F. Zhao, Y. Zhao, Y. Liu, X. Chang, C. Chen, and Y. Zhao, *Small*. **2011**, *7(10)*, 1322-1337, **Cellular uptake, intracellular trafficking, and cytotoxicity of nanomaterials**.

[7] P. Langevin, *Compt. Rendus*. **1908**, *146*, 530-533, **Sur la théorie du mouvement brownien**.

[8] M. E. Muller, *Ann. Math. Statist.* **1956**, *27*, 569-589, **Some continuous Monte Carlo methods for the Dirichlet problem**.

[9] D. S. Grebenkov, A. A. Lebedev, M. Filoche, and B. Sapoval, *Phys. Rev. E*. **2005**, *71(5)*, 056121, **Multifractal properties of the harmonic measure on Koch boundaries in two and three dimensions**.

[10] Y. Zhou, W. Cai, and E. Hsu, *Commun. Math. Sci.* **2017**, *15(1)*, 237-259, **Computation of the local time of reflecting brownian motion and the probabilistic representation of the Neumann problem**.

[11] D. S. Grebenkov, *Phys. Rev. E*. **2015**, *91(5)*, 052108, **Analytical representations of the spread harmonic measure density**.

Appendix

First arrival problem

Markov-chain Monte Carlo (MCMC) method

In this overdamped case, the 2D Brownian motion could be expressed by two independent uniformly distributed random displacement variables, such that $\Delta x_i = \text{ran}(-\delta, +\delta)$ and $\Delta y_i = \text{ran}(-\delta, +\delta)$ with a constant δ. Indeed, this method would furnish varying radius for each diffusion step (Fig. 23, Left).

However, there would be some problem in practice. If the particle gets close to the fractal boundary in a given step, it would be highly probable that the particle jumps out of the boundary directly. Even though we could adjust Metropolis algorithm to draw back the outlier, the fractal boundary would make the corresponding verification much complex than a smooth surface. Moreover, for the sake of more precise result, a rather little value of δ would lead to much longer time for simulations.

To avoid this problem, we could modify the MCMC method such that $\Delta x_i = r\cos\theta_i$ and $\Delta y_i = r\sin\theta_i$, with only one random variable uniformly distributed as $\theta_i = \text{ran}(0, 2\pi)$, and a constant radius r (Fig. 23, Right). Once we select r less than the segment length, we still meet the same problem such that a precise result is quite time-consuming.

Therefore, we employ the "Geometry-adapted fast random walk" algorithm for numerical simulations.

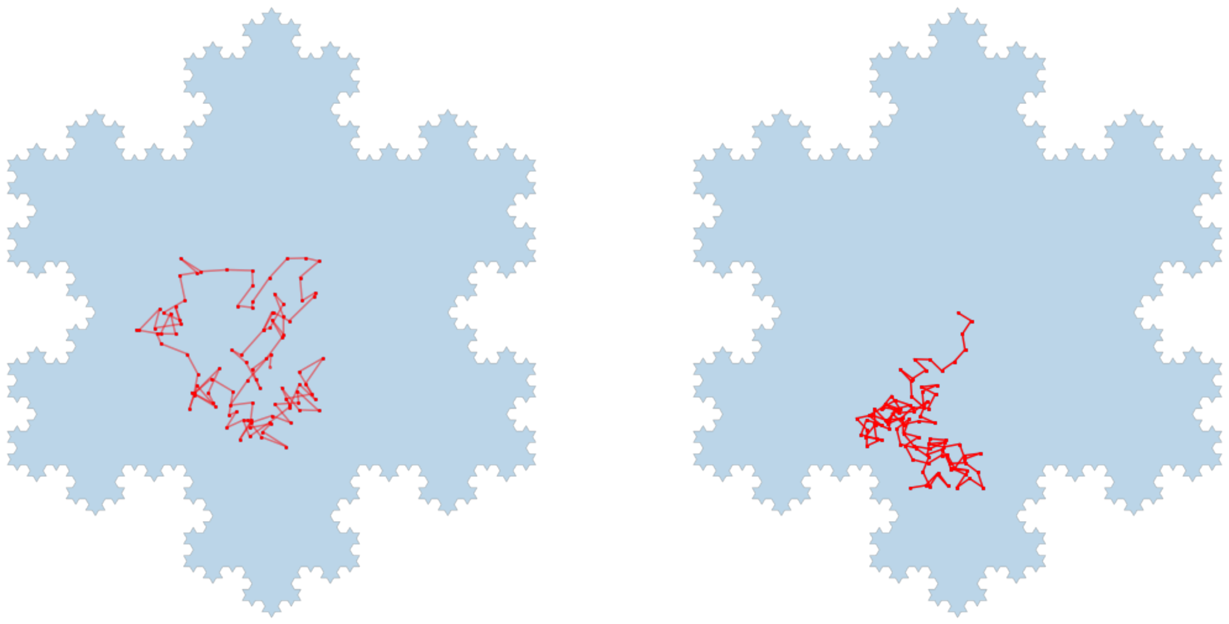


Figure 23. Left: Brownian motion trajectory of a particle (in red) inside Koch snowflake domain (in blue) by MCMC method before reaching the boundary. Right: Brownian motion trajectory of a particle (in red) inside Koch snowflake domain (in blue) by modified MCMC method before reaching the boundary.

GAFRW algorithm

To realize GAFRW algorithm, we exploit several methods in practice. The core strategy is finding the distance to the boundary in each step, such that the particle would not exit the domain. Therefore, it is necessary to search for some boundary points and figure out the radius for each step. Below are typical ideas:

- **Alg 1**. Whole space searching.
  (1) Calculate distances between the particle and all boundary segments and all boundary points at each step.
  (2) Determine the radius as the minimum one of all distances.
  (3) Repeat this process until the particle is attached on the boundary.
  This method could assure that we select the **correct** radius value for GAFRW; however, it is not efficient since the searching time is proportional to 4^g for the first arrival problem, and even much longer for other cases. Thus we only used this algorithm for verification.

- **Alg 3**. Partial space searching.
  (1) Determine the searching range in g according to endpoints in $g-1$, while the zero order refers to the equilateral triangle's three edges.
  (2) Calculate distances between the particle and all boundary segments inside a given range in generation g determined previously.
  (3) Determine the radius as the minimum one of all distances obtained above.
  (4) If the particle quits the known range after the diffusion, update endpoints in each level.
  (5) If the two nearest points form the unit segment in g, we explore search inside this region for next generation $g+1$.
  (6) Repeat the whole process until the desired maximal generation $g_{\max}$.
  (7) Stop the simulation if the particle is attached on the boundary, or the local time exceeds the threshold.
  This method is much more **efficient**, with searching time almost linear to g. Thus we used this algorithm for almost all simulations, both the perfect and partial reactivity.

- **Alg 6**. Generation-wise diffusion.
  (1) Search all boundary points in the first generation and find the diffusion radius.
  (2) Repeat the previous step until the particle is attached on the boundary.
  (3) Consider the next generation of boundary for following diffusions.
  (4) Particle in $g+1$ could not exit the old virtual boundary in g.
  (5) Stop the simulation if the particle is attached on the boundary in generation $g_{\max}$.
  This method is efficient; however, it would **NOT** furnish the correct results as Alg 1. (See Fig. 24) Indeed, there would be little probability for one particle exit the virtual boundary and towards other segments.

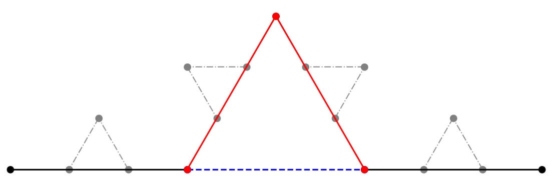
Indeed, there is no analytical solution of harmonic measure for fractal boundary like Koch snowflake, even though we sum up all probabilities on segments in sub-leading generation $g+1$ as the value of the segment in leading generation g. See Fig. 24, we did numerical simulations for the perfect reactivity inside the Koch snowflake $3\_\pi/3-2$ with all three algorithms mentioned above. Alg 6 (in light blue) could not reach the peak as Alg 1 (in black) and Alg 3 (in red). Therefore, we realized following simulations mainly based on "Alg 3" for efficiency.

Spread harmonic measure

Random threshold $\hat{\ell}$

For the continuous distribution $\pi(x)$, we have the cumulative distribution as:

$$\Pi(x) = \Pi(x-\mathrm{d}x) + \pi(x)\mathrm{d}x = \int_{-\infty}^{x} \pi(x)\mathrm{d}x \tag{A1}$$



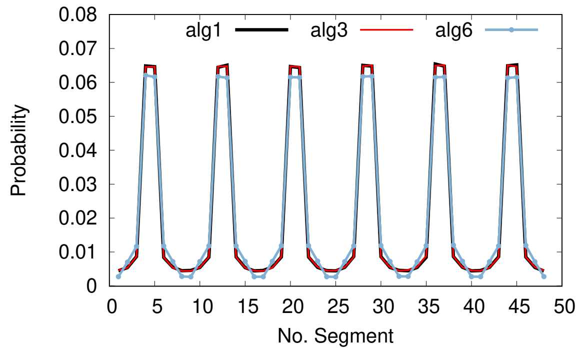


Figure 24. Left: Part of Koch snowflake. The hitting probability on blue dashed segment is not equal to sum of them on two red ones. Right: Comparison of hitting probability distribution from different algorithms (Alg. 1, Alg. 3, and Alg. 6).

Working with normalized $\pi(x)$, the possible value of Π, which we call Υ, is a uniform distribution on (0,1). Let Π^{-1} be the inverse function of Π, then the random number $x=\Pi^{-1}(\Upsilon)$ is distributed as $\pi(x)$. The tricky step is usually to find Π^{-1}.

In our case, the local time threshold is a function of reactivity q, with the distribution as $\psi(\ell)=qe^{-q\ell}$, then we compute the cumulative distribution

$$\Pi(\ell)=\int_0^{\ell}\psi(x)\mathrm{d}x=1-e^{-q\ell}=\Upsilon=\mathrm{ran}(0,1) \tag{A2}$$

Then $\ell=\Pi^{-1}(\Upsilon)=-\ln(1-\Upsilon)/q$. Since both Υ and $1-\Upsilon$ are ran(0,1), we have the random threshold as

$$\hat{\ell}=-\frac{1}{q}\ln\left[\mathrm{ran}(0,1)\right] \tag{A3}$$

Spread harmonic measure on circle

Inside a disk of radius R and center at origin, we consider a particle with initial position (r_0,θ_0) and its diffusion. Define the probability density ω_q such that a particle is attached on the circle (R,θ) finally: [11]

$$\omega_q(\theta|r_0,\theta_0)=\frac{1}{2\pi R}\times\left\{1+2\sum_{j=1}^{\infty}\left(\frac{r_0}{R}\right)^j\frac{\cos\left[j(\theta-\theta_0)\right]}{1+\frac{j}{qR}}\right\} \tag{A4}$$

There is no segment for continuous θ, so we divide the circle into N arcs in numerical practice, such that $\theta\in[k-1,k]\times\frac{2\pi}{N}$, with $k=1,2,3,\ldots,N$. The hitting probability p_k on k-th segment is thus expressed as the integration of density ω_q:

$$p_k=R\times\int_{\frac{2\pi}{N}(k-1)}^{\frac{2\pi}{N}k}\omega_q(\theta|r_0,\theta_0)\mathrm{d}\theta=\frac{1}{N}+\sum_{j=1}^{\infty}\frac{2qR}{\pi j(j+qR)}\left(\frac{r_0}{R}\right)^j\sin\left(\frac{\pi j}{N}\right)\cos\left[j\left(\frac{2\pi}{N}(k-\frac{1}{2})-\theta_0\right)\right] \tag{A5}$$

Codes

In this subsection, we list all *Fortran* codes used during the internship.

- Listing 1: “test_circle.f90”. Code to validate “Reflection” method on a circle domain.
- Listing 2: “para.h”. Common variables used by “PSD_rfl.f90”.
- Listing 3: “PSD_rfl.f90”. Code with “Reflection” method for the fractal domain. The first arrival problem could be realized by setting $q = \infty$.

Listing 1. test_circle.f90

```
!!!!! Diffuion inside circle for validation.
!!!!! Yilin YE @ 06/2023

Program main
    use MATHS
    implicit none
    integer :: i,j,k, date(8)
    integer :: num_points, num_particle, traj_max, final_max, inside, touch, n_iter
    real*8 :: angle_ini, tta, taille, radius_min, u, eps, ell, ell_hat, time_diff
    real*8 :: rho_ratio, q_rect, diff_coef, d, delta, rint, xini, yini, theta0, r0
    real*8,allocatable :: position(:,:), coord_x(:), coord_y(:), account(:), pbb_ana(:)
    real*8,external :: span,pente !! Function to calculate distance between two points.
    real*8 :: temps_debut, temps_fin, temps_provisoire
    character*128 :: rue_total

    call cpu_time(temps_debut)
    continue !! formats
        62 format(10X,'X␣coord',10X,'Y␣coord',10X,'Radius',10X,'Time',10X,'Local␣Time')
        63 format(12X,'X␣coord',12X,'Y␣coord',12X,'Time',12X,'Local␣Time',12X,'Threshold')
        64 format('#',3X,'Segment␣Index',3X,'Hitting␣Probability')
        71 format(2(3X,ES16.6))
        72 format(2(3X,f16.8),3(2X,f16.8))
        73 format(5(4X,f16.8))
        74 format(2X,I6,10X,ES16.8)
        91 format('#␣Polygon',I10,',␣Circle␣Radius',f8.2,',␣Thickness␣',ES8.2)
        92 format('#␣No.␣Particle',I9,',␣Rho␣ratio',f6.2,',␣Reactivity␣',ES8.2,',␣Diff␣coef␣',ES8.2)
        94 format('Circle␣Test␣@␣YYE.␣Version␣1.1.2..␣Simulation␣on␣',I4,2('-',I2),'␣␣␣',I2,'h',I2)
        95 format('Done␣for␣particle',I9,',␣Time␣used␣(s)',ES11.3)
        99 format('It␣spent',f10.4,'␣seconds␣for␣the␣whole␣program.')
    continue

    num_points = 61 !! Define boundary parameters
    num_particle = 1000000; traj_max = 5; final_max = 100
    allocate(position(num_particle,2)); allocate(coord_x(num_points))
    allocate(coord_y(num_points)); allocate(account(num_points-1))
    coord_x = 0.0d0; coord_y = 0.0d0; account = 0.0d0

    rho_ratio = 1.5d0; diff_coef = 1.0d0 !! Determine diffusion/reflection parameters
    taille = 1.0d0; eps = 1.0d-3; q_rect = 1.0d0

    rue_total = "cc-" !! Generate data files.
    call date_and_time(values=date)
    open(unit=31,file=TRIM(rue_total)//"coord.txt"); write(31,94) date(1), date(2), date(3), date(5), date(6)
    open(unit=32,file=TRIM(rue_total)//"particle_traj.txt"); write(32,94) date(1), date(2), date(3), date(5), date(6
        write(32,62)
    open(unit=33,file=TRIM(rue_total)//"particle_final.txt"); write(33,94) date(1), date(2), date(3), date(5), date(
        write(33,63); write(33,*)
    open(unit=34,file=TRIM(rue_total)//"pbnum.txt"); write(34,94) date(1), date(2), date(3), date(5), date(6)
        write(34,91) num_points-1, taille, eps
    write(34,92) num_particle, rho_ratio, q_rect, diff_coef; write(34,*); write(34,64); write(34,*)
    open(unit=36,file=TRIM(rue_total)//"donnees.txt")

    angle_ini = twopi/(num_points - 1) !! Generate boundary points
    do i = 1, num_points
        j = i - 1; tta = angle_ini * j
        coord_x(i) = taille * cos(tta); coord_y(i) = taille * sin(tta)
        write(31,71) coord_x(i), coord_y(i)
    end do

    xini = 0.9d0; yini = 0.0d0 !! Compute analytical probability.
    theta0 = pente(xini,yini); r0 = span(0.0d0,0.0d0,xini,yini)
    open(unit=35,file=TRIM(rue_total)//"pbana.txt")
        write(35,94) date(1), date(2), date(3), date(5), date(6); write(35,92) num_particle, rho_ratio, q_rect, diff
        write(35,*) "#␣r0,␣ 0 ␣=␣", r0, theta0; write(35,*); write(34,*) "#␣r0,␣ 0 ␣=␣", r0, theta0; write(34,*)
    allocate(pbb_ana(num_points-1)); n_iter = 50
    do k = 1, num_points-1
        pbb_ana(k) = 1.0d0 / (taille * (num_points-1))
        do j = 1, n_iter
            pbb_ana(k) = pbb_ana(k) + (2.0d0 * q_rect)/(pi*j*(j+q_rect*taille)) * (r0/taille)**j * &
            & sin(pi*j/(num_points-1)) * cos(j*(twopi/(num_points-1)*(k-0.5d0)-theta0))
        end do
        write(35,74) k, pbb_ana(k)
    end do
    deallocate(pbb_ana); close(35)

    do i = 1, num_particle !! Diffusion for all particles
                !! Initialization
        call random_number(u); ell_hat = -log(u) / q_rect !! PDF needed. psi(l) = q*exp(-q*l);
        ell = 0.0d0; time_diff = 0.0d0; inside = 0; touch = 0
        position(i,1) = xini; position(i,2) = yini
        if (i <= traj_max) then
            write(32,*); write(32,*) "#␣␣␣␣",i
            write(32,72) position(i,1), position(i,2), radius_min, time_diff, ell
        end if

        do while (1 > 0) !! Non-stop diffusion
            radius_min = taille - span(0.0d0,0.0d0,position(i,1),position(i,2))
            if (radius_min > eps) then !! Not enter diffusion layer
                call diffusion(position(i,1), position(i,2), radius_min)
                if (radius_min > taille) then
                    write(32,*) "Leave␣the␣circle␣!!"
                    goto 1101
                end if
            else
                inside = 1; radius_min = eps * rho_ratio
                call diffusion(position(i,1), position(i,2), radius_min)

                d = span(0.0d0,0.0d0,position(i,1),position(i,2))
                if (d < taille - eps) then !! Out of diffusion layer
                    inside = 2
                    !touch = 0
                else if (d > taille) then !! Need reflection!
                    touch = touch + 1
                    if (position(i,1) == 0.0d0) then
                        if (position(i,2) > 0.0d0) then
                            position(i,2) = +2.0d0 * taille - position(i,2)
                        else
                            position(i,2) = -2.0d0 * taille - position(i,2)
                        end if
                    else !!                                  x
                        tta = pente(position(i,1), position(i,2))
                        rint = 2.0d0 * taille - d !span(0.0d0,0.0d0,position(i,1), position(i,2))
                        position(i,1) = rint * cos(tta); position(i,2) = rint * sin(tta)
                    end if
                end if

                if (touch > 0) then
                    delta = radius_min**2 / (4.0d0 * diff_coef); time_diff = time_diff + delta
                    ell = ell + sqrt(pi * diff_coef * delta / 2.0d0)
                    if (ell >= ell_hat) then
                        if (i <= traj_max) write(32,72) position(i,1), position(i,2), radius_min, time_diff, ell
                        if (i == 1) write(36,72) position(i,1), position(i,2), radius_min, time_diff, ell
                        goto 1101
                    end if
                end if

                if (inside == 2) then !! If leave diffusion layer, then clear "touch" and reset "inside"
                    touch = 0; inside = 0
                end if
            end if
            if (i <= traj_max) write(32,72) position(i,1), position(i,2), radius_min, time_diff, ell
            if (i == 1) write(36,72) position(i,1), position(i,2), radius_min, time_diff, ell
        end do
        1101 continue

        tta = pente(position(i,1),position(i,2)); u = (num_points-1) * tta / twopi
        k = ceiling(u) !!                              0                                ++
        account(k) = account(k) + 1

        if (i <= traj_max) write(32,*) "Local_Time_Threshold_=", ell_hat
        if (i <= final_max) write(33,73) position(i,1), position(i,2), time_diff, ell, ell_hat
        if (MOD(i,ceiling(num_particle/5.0d0)).eq.0) then
            call cpu_time(temps_provisoire); write(*,95) i,temps_provisoire-temps_debut
        end if
      end do

      do i = 1, num_points - 1 !!! Normalize account
        account(i) = account(i) / num_particle
        write(34,74) i, account(i)
      end do
      deallocate(position); deallocate(coord_x); deallocate(coord_y); deallocate(account)
      close(31); close(32); close(33); close(34); close(36)
      call cpu_time(temps_fin)
      write(*,99) temps_fin-temps_debut !! Write the total time consumed to the screen.
    end Program main

    MODULE MATHS
      implicit none
      real(kind=8),parameter :: pi = 4.0d0*atan(1.0d0), twopi = 2.0d0*pi
    END MODULE MATHS

    real*8 function span(x1,y1,x2,y2)
      implicit none
      real*8 :: x1,y1,x2,y2
      span = sqrt((x1-x2)**2 + (y1-y2)**2)
      return
    end function span

    subroutine diffusion (x0,y0,rayon)
      use MATHS
      implicit none
      real*8,intent(in) :: rayon
      real*8 :: x0,y0, u, random_angle, move_x, move_y
      call random_number(u); random_angle = u * twopi
      move_x = rayon * cos(random_angle); move_y = rayon * sin(random_angle)
      x0 = x0 + move_x; y0 = y0 + move_y
    end subroutine

    real*8 function pente(x0,y0)
      use MATHS
      implicit none
      real*8 :: x0, y0, theta
      if (x0 == 0.0d0) then
        if (y0 > 0.0d0) then
            theta = pi / 2.0d0
        else
            theta = pi / 2.0d0 * 3.0d0
        end if
      else
        theta = atan(y0/x0)
        if (theta < 0.0d0) then
            if (x0 < 0.0d0) then
```

```
                theta = theta + pi
            else
                theta = theta + twopi
            end if
        else if (theta == 0.0d0) then
            if (x0 < 0.0d0) theta = pi
        else
            if (x0 < 0.0d0) theta = theta + pi
        end if
    end if
    pente = theta
    return
end function
```

Listing 2. para.h

```
!!!!! We define parameters used for equations

integer :: polygon_shape, Generation_max, num_particle, alpha_angle_ratio, direct, num_points
common/paraint/ polygon_shape, Generation_max, num_particle, alpha_angle_ratio, direct, num_points
real*8 :: Length, seg_l, thickness, rho_ratio, q_rect, diff_coef
common/parareal/ Length, seg_l, thickness, rho_ratio, q_rect, diff_coef
```