Random walk on simplicial complexes

Zhihan Zhang

université
PARIS-SACLAY

Random walk on simplicial complexes

Thèse de doctorat de l'université Paris-Saclay

Ecole Doctorale de Mathématique Hadamard (EDMH) n° 574
Spécialité de doctorat : Mathématiques appliquées
Unité de recherche : Laboratoire Traitement et Communication de l'Information (Télécom Paris)
Référent : Faculté des sciences d'Orsay

Thèse présentée et soutenue en visioconférence totale, le 10 décembre 2020, par

Zhihan ZHANG

Thèse de doctorat

NNT: 2020UPASM010

Abstract

The notion of Laplacian of a graph can be generalized to simplicial complexes and hypergraphs. This notion contains information on the topology of these structures. Even for a graph, the consideration of associated simplicial complexes can provide information on its shape. Whereas the Laplacian of a graph has a simple probabilistic interpretation as the generator of a continuous time Markov chain on the graph, things are not so direct when considering simplicial complexes. In the first part of this thesis, we define a new Markov chain on simplicial complexes. For a given degree k of simplices, the state space is not the k-simplices as in previous papers about this subject but rather the set of k-chains or k-cochains. This new framework is the natural generalization on the canonical Markov chains on graphs. We show that the generator of our Markov chain is related to the upper Laplacian defined in the context of algebraic topology for discrete structure. We establish several key properties of this new process. We show that when the simplicial complexes under scrutiny are a sequence of ever refining triangulation of the flat torus, the Markov chains tend to a differential form valued continuous process.

In the second part of this thesis, we explore some applications of the random walk, *i.e.,* random walk based hole detection and simplicial complexes kernels. For the random walk based hole detection, we introduce an algorithm to make simulations carried for the cycle-valued random walk ($k = 1$) on a simplicial complex with holes. Since the state space of the cycle-valued random walk consists of all the states in the same homology group, the paths with minimal lengths are supposed to be the holes of the simplicial complex, which is the idea of the algorithm. In the case where we do not know the boundary of the simplicial complex, we need to find the initial state which is in the same homology group as the holes and has an integer value. Thus we calculate the initial states and propose another algorithm to make simulations of integer weighted random walk. Simulation results show that we can always detect the holes of simplicial complexes with boundary (initial state) known, and approximately with initial state unknown. For the simplicial complexes kernels, we extend the definition of random walk based graph kernels in order to measure the similarity between two simplicial complexes. The definition is based on this idea: given a pair of simplicial complexes, we perform our random walks on both, and count the number of matching walks. The more matching walks, the more similar these two simplicial complexes are. In order to count the number of matching walks, we perform a random walk on the direct product simplicial complex of these two simplicial complexes. We compute the probability of the random walk whose length is k, and sum them for all k. The result is the kernel of these two simplicial complexes. In order to reduce computational complexity, we rewrite the sum formula to the inverse of a matrix, and use conjugate gradient methods instead of direct computation. Simulation results show that graph kernels are related to the number of vertices and their connectivity, while simplicial complexes kernels put emphasis on homology properties.

Résumé

La notion de laplacien d'un graphe peut être généralisée aux complexes simpliciaux et aux hypergraphes. Cette notion contient des informations sur la topologie de ces structures. Même pour un graphe, la prise en compte des complexes simpliciaux associés peut fournir des informations sur sa forme. Alors que le laplacien d'un graphe a une interprétation probabiliste simple comme générateur d'un processus de Markov sur le graphe, les choses ne sont pas si directes lorsqu'on considère les complexes simpliciaux. Dans la première partie de cette thèse, nous définissons une nouvelle chaîne de Markov sur les complexes simpliciaux. Pour un degré donné k de simplexes, l'espace d'états n'est pas les k-simplexes comme dans les articles précédents sur ce sujet mais plutôt l'ensemble des k-chaines ou k-co-chaines. Ce nouveau cadre est la généralisation naturelle sur les chaînes de Markov canoniques sur des graphes. Nous montrons que le générateur de notre chaîne de Markov est lié au Laplacien supérieur défini dans le contexte de la topologie algébrique pour structure discrète. Nous établissons plusieurs propriétés clés de ce nouveau procédé. Nous montrons que lorsque les complexes simpliciaux examinés sont une séquence de triangulation du tore plat, les chaînes de Markov tendent vers une forme différentielle valorisée processus continu.

Dans la deuxième partie de cette thèse, nous explorons quelques applications de la marche aléatoire, *i.e.* la détection de trous basée sur la marche aléatoire et les noyaux complexes simpliciaux. Pour la détection de trous basée sur la marche aléatoire, nous introduisons un algorithme pour faire des simulations effectuées pour la marche aléatoire à valeur cyclique ($k = 1$) sur un complexe simplicial avec trous. Puisque l'espace d'états de la marche aléatoire à valeurs cycliques se compose de tous les états d'un même groupe d'homologie, les chemins de longueurs minimales sont supposés être les trous du complexe simplicial, ce qui est l'idée de l'algorithme. Dans le cas où nous ne connaissons pas la limite du complexe simplicial, nous devons trouver l'état initial qui est dans le même groupe d'homologie que les trous et a une valeur entière. Nous calculons ainsi les états initiaux et proposons un autre algorithme pour faire des simulations de marche aléatoire pondérée par des entiers. Les résultats de la simulation montrent que nous pouvons détecter les trous de complexes simpliciaux dont la frontière (état initial) est connue, et approximativement avec l'état initial inconnu. Pour les noyaux de complexes simpliciaux, nous étendons la définition des noyaux de graphes basés sur la marche aléatoire afin de mesurer la similitude entre deux complexes simpliciaux. La définition est basée sur cette idée : étant donné une paire de complexes simpliciaux, nous effectuons nos marches aléatoires sur les deux et comptons le nombre de marches identiques. Plus les marches sont concordantes, plus ces deux complexes simpliciaux sont similaires. Afin de compter le nombre de marches identiques, nous effectuons une marche aléatoire sur le complexe simplicial produit direct de ces deux complexes simpliciaux. Nous calculons la probabilité de la marche aléatoire de longueur k avec les probabilités initiale et d'arrêt, et les additionnons pour tout k. Le résultat est le noyau de ces deux complexes simpliciaux. Afin de réduire la complexité du calcul, nous réécrivons la formule de somme comme l'inverse d'une matrice et en utilisant des méthodes de gradient conjugué au lieu du calcul direct. Les résultats de la simulation montrent que les noyaux de graphes sont liés au nombre de sommets et à leur connectivité, tandis que les noyaux de complexes simpliciaux mettent l'accent sur les propriétés d'homologie.

List of Figures

List of Tables

Contents

Chapter 0

Résumé long en français

0.1 Introduction

Explorer et comprendre des structures complexes telles que les graphes aléatoires est un problème difficile et riche qui a motivé une littérature abondante dans ces dernières années. La première étape naturelle face à de grands graphes est de rechercher un cluster ou des structures communautaires [22, 42], c'est-à-dire une partition où la connectivité à l'intérieur d'une classe est supérieure à la connectivité entre les classes. Il y a plusieurs façons de faire cela, comme le clustering de modularité [38, 11] ou le clustering spectral [44, 48]. La théorie sous-jacente à cette dernière méthode est particulièrement populaire car elle établit un lien entre la topologie du graphe et les marches aléatoires du plus proche voisin sur celui-ci (voir [20, 32] pour une introduction). Ces marches aléatoires sont des processus de Markov qui visitent les sommets du graphe en sautant en temps continu de leur position courante v à un sommet choisi uniformément parmi les sommets voisins. Considérons un graphe fini non orienté $G = (V, E)$ constitué des ensembles (finis) de sommets V et d'arêtes E déterminant les paires de sommets qui sont connectés. La matrice d'adjacence A de G est définie comme la matrice dont l'élément à la ligne u et à la colonne v est 1 si et seulement si (u, v) est une arête de G, ce que nous désignerons $u \sim v$. Pour tout $u \in V$ et pour toute fonction f de V à $\mathbf{R}$, le générateur de la marche aléatoire est

$$Lf(u) = \sum_{v \sim u} \big(f(v) - f(u)\big) = -\big(D - A\big) f(u), \tag{0.1}$$

où D est la matrice diagonale contenant les degrés des sommets. Donc, le générateur de la marche aléatoire est l'opposé du laplacien du graphe, $D - A$. Afin de trouver des clusters dans les graphes, les composants presque déconnectés des graphes sont toujours importants. Il est bien connu que la dimension du noyau de $D - A$ est égale au nombre de composantes connexes du graphe, et le clustering spectral vise à considérer de petites valeurs propres en valeurs absolues, qui correspondent à des composants presque déconnectés.

Bien que les algorithmes existants sont bien adaptés pour étudier la connectivité d'un graphe, ils ne suffisent pas pour étudier la topologie complète du graphe. Pour résoudre ce problème, les complexes simpliciaux sont les structures correctes. Un complexe simplicial naturel associé à un graphe est obtenu en ajoutant au couple (V, E) l'ensemble $\mathcal{S}_2$ de tous les triangles dont les arêtes appartiennent à E, l'ensemble de tous les tétraèdres $\mathcal{S}_3$ dont les triangles appartiennent à $\mathcal{S}_2$ etc. Dans le complexe simplicial, le premier et deuxième nombre de Betti représentent le nombre de trous de 2 et 3 dimensions respectivement. Comme le nombre de Betti d'ordre zéro représente le nombre de composants connectés, chercher une généralisation des liens entre les nombres de Betti, les Laplaciens des graphes et les marches aléatoires est une idée naturelle.

Donc le but de cette thèse est de donner une fondation probabiliste des graphes Laplaciens et des nombres de Betti par les marches aléatoires. En sachant que les composantes connectées d'un graphe ne donnent que des informations limitées sur la connectivité globale du graphe, nous espérons trouver plus d'information que les nombres de Betti d'un complexe simplicial. Nous espérons également étendre les algorithmes d'analyse de graphes aux complexes simpliciaux. Comme la plupart des algorithmes d'analyse de graphes reposent sur les propriétés du spectre du graphe Laplacien ou d'une marche aléatoire sur un graphe, l'adaptation de ces objets au cas de complexes simpliciaux devrait donner des équivalents aux nombreux algorithmes d'analyse de graphe pour les complexes simpliciaux.

0.2 Contexte mathématique

0.2.1 Complexes Simpliciaux

Étant donné un ensemble fini ou dénombrable de sommets V, un k-simplex est un sous-ensemble non ordonné $\{v_0, v_1, \ldots, v_k\}$ où $v_i \in V$ et $v_i \neq v_j$ pour tout $i \neq j$. Les faces du k-simplex $\{v_0, v_1, \ldots, v_k\}$ sont définies comme touts les $(k-1)$-simplexes de la forme $\{v_0, \ldots, v_{j-1}, v_{j+1}, \ldots, v_k\}$ avec $0 \leq j \leq k$. Les co-faces d'un k-simplex τ sont touts les $(k+1)$-simplexes dont τ est une face. Un complexe simplicial $\mathcal{C}$ est une collection de simplexes qui est fermée par rapport à l'inclusion de faces. On note $\mathcal{S}_k(\mathcal{C})$ l'ensemble des k-simplexes de $\mathcal{C}$. On peut définir une orientation sur les simplexes en définissant un ordre sur les sommets et avec la convention que :

$$[v_0, \ldots, v_i, \ldots, v_j, \ldots, v_k] = -[v_0, \ldots, v_j, \ldots, v_i, \ldots, v_k],$$

pour $0 \leq i, j \leq k$. Chaque simplexe peut apparaître de deux manières : orienté positivement ou négativement. On note $\mathcal{S}_k^+$ (respectivement $\mathcal{S}_k^-$) l'ensemble des k-simplexes orientés positivement (respectivement négativement). Pour une arête orientée $[u, v]$ (allant de u à v), quand $[u, v] \in \mathcal{S}_1$, nous écrirons $v \sim u$.

0.2.2 Chaines et co-chaines

Pour chaque entier k, $\mathcal{C}_k$ est l'espace vectoriel couvert par l'ensemble $\mathcal{S}_k^+$ de k-simplexes de V : un élément $\tau \in \mathcal{C}_k$ est appelé une *chaine* ou *k-chaine* et peut être écrit uniquement comme

$$\tau = \sum_{s \in \mathcal{S}_k^+} \lambda_s(\tau)\, s, \tag{0.2}$$

où tous sauf un nombre fini de $\{\lambda_s(\tau) \in \mathbf{R},\ s \in \mathcal{S}_k^+\}$ sont non nuls.

Soit $\mathcal{C}^k$ le dual de $\mathcal{C}_k$:

$$\mathcal{C}^k = \Big\{ f \ :\ \mathcal{C}_k \to \mathbf{R},\ \text{ linéaire et continue } \Big\}.$$

Notez que la dimension de $\mathcal{C}_k$ est finie, donc toute forme linéaire sur $\mathcal{C}_k$ est continue. Puisque $\mathcal{C}_k$ est un espace de Hilbert, il en est de même pour $\mathcal{C}^k$.

0.2.3 La frontière et co-frontière

Pour tout entier k, la frontière ∂_k est la transformation linéaire $\partial_k : \mathcal{C}_k \to \mathcal{C}_{k-1}$ qui agit sur les éléments de base $[v_0, \ldots, v_k]$ comme

$$\partial_k[v_0,\, \ldots,\, v_k] = \sum_{i=0}^{k}(-1)^i[v_0,\, \ldots,\, v_{i-1},\, v_{i+1},\, \ldots,\, v_k], \tag{0.3}$$

et ∂_0 est la fonction nulle. Si l'on définit

$$Z_k = \ker \partial_k \text{ et } B_k = \operatorname{im} \partial_{k+1}, \tag{0.4}$$

alors, le k-ième espace vectoriel d'homologie H_k est défini comme l'espace vectoriel de quotient,

$$H_k = Z_k/B_k \tag{0.5}$$

et le k-th Betti nombre de $\mathcal{C}$ est défini comme sa dimension :

$$\beta_k = \dim H_k = \dim Z_k - \dim B_k. \tag{0.6}$$

Comme $\mathcal{C}_k$ et $\mathcal{C}^k$ sont des espaces de Hilbert, nous pouvons définir la frontière $\partial_k^* : \mathcal{C}^{k-1} \longrightarrow \mathcal{C}^k$ comme l'adjoint de ∂_k : à savoir pour $f \in \mathcal{C}^{k-1}$, $\partial_k^* f \in \mathcal{C}^k$ est définie par son action sur une chaîne de k par

$$\begin{aligned}(\partial_k^* f)[v_0, \cdots, v_k] &= \sum_{i=0}^{k}(-1)^i\langle f,\ [v_0, \cdots, v_{i-1}, v_{i+1}, \cdots, v_k]\rangle_{\mathcal{C}^{k-1},\mathcal{C}_{k-1}} \\ &= f\big(\partial_k[v_0, \cdots, v_k]\big).\end{aligned} \tag{0.7}$$

Nous définirons par convention $\partial_0^* \equiv 0$.

0.2.4 Laplacien combinatoire

Une notion cruciale pour ce qui suit est celle de laplacien combinatoire (voir [21, 34] pour les détails). Nous savons que

$$\begin{tikzcd}
{} \arrow[r, dashed, "\partial_{k+2}"] & \mathcal{C}_{k+1} \arrow[r, "\partial_{k+1}"] \arrow[d, leftrightarrow] & \mathcal{C}_k \arrow[r, "\partial_k"] \arrow[d, leftrightarrow] & \mathcal{C}_{k-1} \arrow[r, dashed, "\partial_{k-1}"] \arrow[d, leftrightarrow] & {} \\
{} & \mathcal{C}^{k+1} \arrow[l, dashed, "\partial^*_{k+2}"] & \mathcal{C}^k \arrow[l, "\partial^*_{k+1}"] & \mathcal{C}^{k-1} \arrow[l, "\partial^*_k"] & {} \arrow[l, dashed, "\partial^*_{k-1}"]
\end{tikzcd}$$

où les doubles flèches signifient que nous avons un isomorphisme isométrique entre les deux espaces Hilbert concernés. Puisque nous avons identifié les espaces $\mathcal{C}_k$ et $\mathcal{C}^k$ pour tout $k \in \mathbf{N}$, nous pouvons considérer

$$L_k^{\uparrow} := \partial_{k+1}\partial_{k+1}^* \,:\, \mathcal{C}_k \xrightarrow{\partial_{k+1}^*} \mathcal{C}_{k+1} \xrightarrow{\partial_{k+1}} \mathcal{C}_k \tag{0.8}$$

et

$$L_k^{\downarrow} := \partial_k^*\partial_k \,:\, \mathcal{C}_k \xrightarrow{\partial_k} \mathcal{C}_{k-1} \xrightarrow{\partial_k^*} \mathcal{C}_k. \tag{0.9}$$

Ces derniers opérateurs sont appelés respectivement les Laplaciens supérieur et inférieur (de l'ordre de k). Le laplacien combinatoire d'ordre k est défini comme

$$\begin{aligned} L_k \,:\, \mathcal{C}_k &\longrightarrow \mathcal{C}_k \\ \tau &\longmapsto \left(\partial_{k+1}\partial_{k+1}^* + \partial_k^*\partial_k\right)(\tau) = L_k^{\uparrow}(\tau) + L_k^{\downarrow}(\tau). \end{aligned} \tag{0.10}$$

Le théorème combinatoire de Hodge dit que

Théorème 1. *Pour tout $k \in \mathbf{N}$, nous avons*

$$\mathcal{C}_k = \operatorname{im}\partial_{k+1} \oplus \operatorname{im}\partial_k^* \oplus \ker L_k. \tag{0.11}$$

Il s'ensuit que

$$\ker L_k \simeq H_k. \tag{0.12}$$

En particulier, le k-ième nombre de Betti est la dimension de l'espace nul de L_k :

$$\beta_k = \dim\ker(L_k). \tag{0.13}$$

0.3 La marche aléatoire et ses limites diffusives continues

0.3.1 Générateur de la marche aléatoire à valeur de chaîne

Dans ce qui suit, $k \geq 1$ est fixé.

Espace des fonctions de test Considérer $\mathcal{D}$ l'espace des fonctions de $\mathcal{C}_k$ à $\mathbf{R}$ de la forme

$$F(\tau) = f\left(\langle \eta_1, \tau\rangle_{\mathcal{C}^k,\mathcal{C}_k}, \cdots, \langle \eta_m, \tau\rangle_{\mathcal{C}^k,\mathcal{C}_k}\right) \tag{0.14}$$

pour certains $m \geq 1$, $(\eta_1, \cdots, \eta_m)$ certains éléments de $\mathcal{C}^k$ et f mesurable et borné de $\mathbf{R}^m$ à $\mathbf{R}$.

Noyau de transition Pour une k-chaîne $\tau \neq 0$ et un $(k+1)$-simplex orienté $\eta \in \mathcal{S}_{k+1}$, on définit le nombre de faces communes entre τ et $\partial_{k+1}\eta$ par :

$$w\left(\tau,\ \partial_{k+1}\eta\right) = \langle (\partial_{k+1}\eta)^*,\ \tau\rangle^+_{\mathcal{C}^k,\mathcal{C}_k}$$

où $x^+ = \max(x, 0)$ for $x \in \mathbf{R}$. Pour une autre chaîne τ', on dit que τ et τ' sont adjacents (dans le sens où la marche aléatoire peut atteindre τ' à partir de l'état τ) et on écrit

$$\tau \sim \tau' \iff \exists \eta \in \mathcal{S}_{k+1}, w\left(\tau,\ \partial_{k+1}\eta\right) > 0 \text{ et } \tau' = \tau - \partial_{k+1}\eta. \tag{0.15}$$

Enfin, définissons le poids de la transition de τ à τ' :

$$K(\tau,\,\tau') = \begin{cases} 1 & \text{si } \tau = \tau' = 0 \\ w\left(\tau,\ \tau - \tau'\right) & \text{si } \tau \sim \tau' \\ 0 & \text{autrement.} \end{cases} \tag{0.16}$$

Générateur de la marche aléatoire Définissons par $(A, D(A))$ le générateur de la marche aléatoire en temps continu.

Définition 1. *Soit $D(A)$ l'ensemble des fonctions F telles que $|\sum_{\tau'\sim\tau}\Big(F(\tau') - F(\tau)\Big)\,K(\tau,\,\tau')| < +\infty$. Pour $F \in D(A)$, on peut définir*

$$\begin{aligned} AF\,:\,\mathcal{C}_k &\longrightarrow \mathbf{R} \\ \tau &\longmapsto \sum_{\tau'\sim\tau}\Big(F(\tau') - F(\tau)\Big)\,K(\tau,\,\tau'). \end{aligned}$$

Théorème 2. *Pour F une fonction linéaire de $\mathcal{A}$, nous avons pour chaque $\tau \in \ker \partial_k$,*

$$AF(\tau) = -L_k^{\uparrow}F(\tau) = -L_kF(\tau). \tag{0.17}$$

0.3.2 Récurrence de la chaîne sur les graphes finis

Pour tout k-simplex τ, laissez $d^-(\tau)$ être le degré de contiguïté inférieur de τ donné par

$$d(\tau)^- = \sum_{\tau' \in \mathcal{S}_k, \tau' \neq \tau} |\langle \big(\partial_k(\tau)\big)^*, \partial_k(\tau') \rangle_{\mathcal{C}^{k-1}, \mathcal{C}_{k-1}}|.$$

Cette quantité correspond au nombre de faces que τ et τ' ont en commun.

Théorème 3. *Soit $k \in \mathbf{N}$. Soit $\sigma_1, \dots, \sigma_{\beta_k} \in \mathcal{C}_k$ une base de H_k et soit $(\tau_1, \dots, \tau_n) \in \mathcal{S}_{k+1}$ les $(k+1)$-simplexes de notre complexe simplicial. Supposons qu'il existe $\mu_1, \dots, \mu_{\beta_k} \in \mathbf{Z}$ et $(\lambda_\tau, \tau \in \mathcal{S}_{k+1}) \in \{-1, 0, 1\}^{\mathcal{S}_{k+1}}$ tel que*

$$X(0) = \sum_{i=1}^{\beta_k} \mu_i \sigma_i + \sum_{\tau \in \mathcal{S}_{k+1}} \lambda_\tau \partial_{k+1} \tau.$$

Si nous avons pour tout $\tau \in \mathcal{S}_{k+1}$ que :

$$d(\tau)^- \leq k + 2 - |\sum_{i=1}^{\beta_k} \mu_i \langle \big(\partial_{k+1}\tau\big)^*, \sigma_i \rangle|, \tag{0.18}$$

alors X a un espace d'état fini et pour tout $t > 0$, il existe $(\lambda_\tau(t), \tau \in \mathcal{S}_{k+1}) \in \{-1, 0, 1\}^{\mathcal{S}_{k+1}}$ tel que

$$X_t = \sum_{i=1}^{\beta_k} \mu_i \sigma_i + \sum_{\tau \in \mathcal{S}_{k+1}} \lambda_\tau(t) \partial_{k+1} \tau.$$

0.3.3 Convergence de la marche aléatoire

Nous désignons par $\mathbb{T}_2 := \mathbf{R}^2/\mathbf{Z}^2$ le tore plat, que nous emboîtons dans $\mathbf{R}^2$ comme le carré $[0, 1] \times [0, 1]$ où se trouvent les arêtes opposés identifiés. Soit $\epsilon_n = 1/2n$ et considérons

$$V_n = \{(2k\epsilon_n, 2l\epsilon_n), 0 \leq k, l \leq n\} \bigcup \{((2k+1)\epsilon_n, (2l+1)\epsilon_n), 0 \leq k, l \leq n-1\},$$

l'ensemble des sommets de la triangulation régulière de la maille $2\epsilon_n$.

Soit Φ l'espace de la 1-forme différentielle continue sur le tore. Nous désignons par $\Phi^{(k)}$, l'ensemble de 1-formes k-fois différentiable.

Définition 2. *Pour $\phi = \phi^1\, dx_1 + \phi^2\, dx_2 \in \Phi^{(1)}$, sa dérivée extérieure désignée par $d\phi$ est la fonction (ou 0-forme) :*

$$d\phi(x) = \frac{\partial \phi^2}{\partial x_1}(x) - \frac{\partial \phi^1}{\partial x_2}(x).$$

La transformation Hodge des formes est la transformation linéaire définie par son l'action sur la base de formes différentielles :

$$*1 = \, dx_1 \wedge \, dx_2, \ * \, dx_1 = \, dx_2, \ * \, dx_2 = - \, dx_1, \ *(\, dx_1 \wedge \, dx_2) = 1.$$

Pour $\phi \in \Phi^{(2)}$, l'opérateur Laplace-Beltrami est alors défini par

$$\mathfrak{L} = \mathfrak{L}^{\uparrow} + \mathfrak{L}^{\downarrow} \text{ où } \mathfrak{L}^{\uparrow} = *d*d \text{ et } \mathfrak{L}^{\downarrow} = d*d*.$$

Définition 3. *Soit*

$$\operatorname{dom}\Big((\mathfrak{L}^{\uparrow})^*\Big) = \big\{p \in \mathfrak{C}_1, \exists c_p, |\langle p, \mathfrak{L}^{\uparrow}\phi\rangle_{\mathfrak{C}_1,\Phi}| \le c_p \|\phi\|_{\Phi}, \forall \phi \in \operatorname{dom}(\mathfrak{L}^{\uparrow})\big\}.$$

Notez que $\mathfrak{P} \subset \operatorname{dom}\Big((\mathfrak{L}^{\uparrow})^*\Big)$. Ensuite, $(\mathfrak{L}^{\uparrow})^*$ est défini par la relation :

$$\langle p, \, \mathfrak{L}^{\uparrow}\phi\rangle_{\mathfrak{C}_1,\Phi} = \langle (\mathfrak{L}^{\uparrow})^* p, \, \phi\rangle_{\mathfrak{C}_1,\Phi}.$$

.

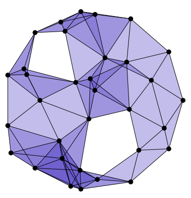
Corollaire 4. *La séquence de générateurs $(\epsilon_n^{-2} A_n, \, n \ge 1)$ tend à $(\mathfrak{L}^{\uparrow})^*$, tout comme les les semi-groupes correspondants.*

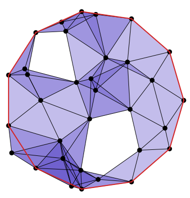
0.4 Détection de trous par la marche aléatoire

Nous introduisons un algorithme de recuit simulé pour faire des simulations effectuées pour la marche aléatoire évaluée par cycle $(k = 1)$ sur un complexe simplicial avec des trous, et voir où se situent les trous en minimisant la longueur de notre chaîne.

Le complexe de Rips avec 25 points où $\beta_0 = 1$ et $\beta_1 = 2$ est montré dans la Figure 0.1a. L'état initial comprenant les deux trous est indiqué dans la Figure 0.1b, et l'état final est indiqué dans la Figure 0.1c.

Nous pouvons voir que notre algorithme de recuit simulé localise précisément ces deux trous.


(a) *Complexe de Rips*


(b) *État initial*

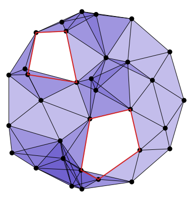
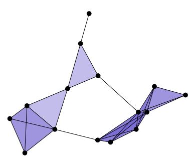
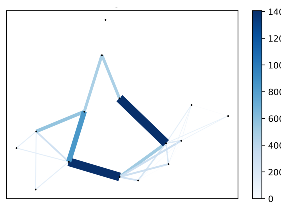

(c) *État final*

Figure 0.1: *Marche aléatoire sur un complexe de Rips dont l'état initial est connu, l'état initial et l'état final sont représentés en ligne rouge*

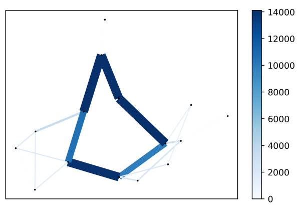
Si nous ne connaissons pas l'état initial à l'avance, nous devons générer un état initial et nous assurer qu'il se trouve dans le même groupe d'homologie que les trous de notre complexe simplicial. Dans ce cas, nous calculons le vecteur propre correspondant à la valeur propre zéro, et nous le fixons comme notre état initial.

Le complexe de Rips avec 15 points où $\beta_0 = 1$ et $\beta_1 = 1$ est montré dans la Figure 0.2a. La Figure 0.2b représente notre état initial, où l'épaisseur de chaque arête indique le poids de cette arête. L'état final est illustré par la Figure 0.2c.


(a) *Complexe de Rips*



(b) *État initial*



(c) *État final*

Figure 0.2: *Marche aléatoire sur un complexe de Rips dont l'état initial est inconnu*

Sans connaître l'état initial, il semble que nous ne puissions obtenir les limites des trous que de manière approximative.

0.5 Noyaux basés sur la marche aléatoire

Afin de comparer deux structures de données, nous générons d'abord des graphes ou des complexes simpliciaux pour les structures de données, puis nous générons des noyaux pour mesurer les similarités. Nous pouvons étendre la définition des noyaux des graphes ([47]) aux noyaux des complexes simpliciaux comme suit. Tout d'abord, nous introduisons quelques définitions de base afin de définir les noyaux.

Définition 4. *(Produits directs des complexes simpliciaux) Soient $\mathcal{C}$ et $\mathcal{C}'$ sont des complexes simpliciaux abstraits dont on ordonne l'ensemble des sommets. Nous définissons le produit direct de $\mathcal{C}$ et $\mathcal{C}'$ comme étant le complexe simplicial $\mathcal{C}_\times$ avec les propriétés suivantes :*

1. *Son ensemble de sommets*

$$V_\times = \{(v_i, v'_r) : \ v_i \in V, v'_r \in V'\},$$

où V est l'ensemble des sommets de $\mathcal{C}$ et V' est l'ensemble des sommets de $\mathcal{C}'$.

2. *Son ensemble de k-simplex*

$$(\mathcal{S}_k)_\times = \{[(v_{i_0}, v'_{r_0}), \ldots, (v_{i_k}, v'_{r_k})] : \ [v_{i_0}, \ldots, v_{i_k}] \in \mathcal{S}_k, [v'_{r_0}, \ldots, v'_{r_k}] \in \mathcal{S}'_k\}$$

où $\mathcal{S}_k$ est l'ensemble des k-simplexes de $\mathcal{C}$ et $\mathcal{S}'_k$ est l'ensemble des k-simplexes de $\mathcal{C}'$.

En suivant la définition de $K(\tau, \tau')$ dans (0.16), on peut donner la définition de la matrice de transition $\mathcal{K}$.

Définition 5 (Matrice de transition)**.** *Étant donné un complexe simplicial $\mathcal{C}$ et un état initial τ_{ini}, nous avons la classe de récurrence $\mathcal{R}(\tau_{ini})$ de τ_{ini} dont la dimension est N. Pour tout $\tau_i, \tau_j \in \mathcal{R}(\tau_{ini})$, si τ_i et τ_j sont étiquetés comme i et j dans $\mathcal{R}(\tau_{ini})$, la matrice de transition $\mathcal{K}$ est définie par*

$$\mathcal{K}_{ij} = \frac{K(\tau_i, \tau_j)}{\sum_{k=1}^{N} K(\tau_i, \tau_k)}.$$

Pour les complexes simpliciaux $\mathcal{C}$ et $\mathcal{C}'$, les dimensions de la matrice de transition correspondante $\mathcal{K}$ et $\mathcal{K}'$ sont $N \times N$ et $N' \times N'$, on associe une matrice de poids $W_\times \in \mathbf{R}^{NN' \times NN'}$ à $\mathcal{C}_\times$ par $W_\times = \mathcal{K}_\times$.

Définition 6. *Pour deux complexes simpliciaux $\mathcal{C}$ et $\mathcal{C}'$, le complexe simplicial de produit direct de $\mathcal{C}$ et $\mathcal{C}'$ est $\mathcal{C}_\times$, et la matrice de poids de $\mathcal{C}_\times$ est $W_\times$. Nous effectuons notre marche aléatoire en chaîne sur $\mathcal{C}_\times$ avec une probabilité initiale de $p_\times$ et une probabilité d'arrêt de $q_\times$. Le noyau entre $\mathcal{C}$ et $\mathcal{C}'$ est défini par*

$$k(\mathcal{C}, \mathcal{C}') = \sum_{k=0}^{\infty} \mu(k) q_\times^\top W_\times^k p_\times, \tag{0.19}$$

où $\mu(k)$ est un coefficient non négatif pour s'assurer que $k(\mathcal{C}, \mathcal{C}')$ converge.

Pour les résultats des calculs, nous définissons nos trois structures de données, désignées par $(DS)_0$, $(DS)_1$ et $(DS)_2$ dans la Figure 0.3. $(DS)_1$ et $(DS)_2$ ont tous les points que $(DS)_0$ a, et ils ont leurs propres points de perturbation, qui sont représentés en rouge dans la Figure 0.3b et la Figure 0.3c. La différence entre deux points de perturbation réside dans le fait que le point de perturbation dans $(DS)_1$ ne change pas la propriété d'homologie, mais que le point de perturbation dans $(DS)_2$ le fait.

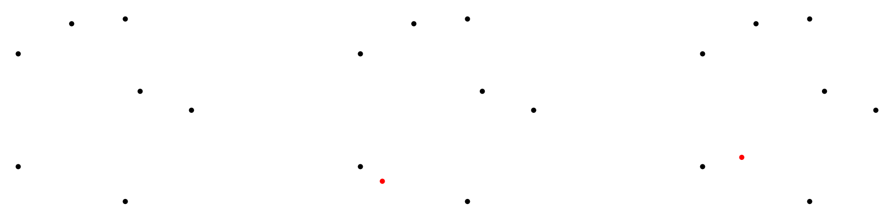
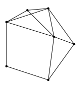
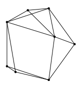
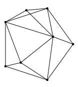
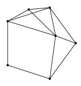
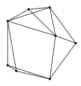
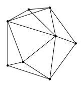

(a) $(DS)_0$ (b) $(DS)_1$ (c) $(DS)_2$

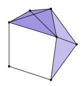
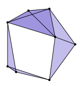
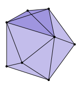
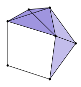
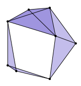
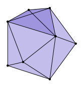
Figure 0.3: *Structures de données $(DS)_0$ avec 7 points, $(DS)_1$ avec 8 points et $(DS)_2$ avec 8 points, avec différents points entre lui et $(DS)_0$ représenté en rouge*

Les résultats de calcul des noyaux des graphes et des noyaux des complexes simpliciaux sont présentés dans le Tableau 0.1a et le Tableau 0.1b. Par noyaux des graphes, les graphes les plus similaires sont G_0 et G_2, mais nous pouvons dire que la version complexe simplicial de G_0 a un trou alors que la version complexe simplicial de G_1 n'a pas de trou. D'autre part, par les noyaux des complexes simpliciaux, elle trie parfaitement par les propriétés d'homologie.

Nous avons découvert que les noyaux des graphes sont liés au nombre de sommets et à leur connectivité, tandis que les noyaux des complexes simpliciaux mettent l'accent sur les propriétés d'homologie.

$\lambda = 0.01$	 G_0	 G_1	 G_2
 G_0	-	0.01979	0.01981
 G_1	0.01979	-	0.01737
 G_2	0.01981	0.01737	-

(a) *Les noyaux de graphes avec* $\lambda = 0.01$

$\lambda = 0.001$	 $\mathcal{C}_0$	 $\mathcal{C}_1$	 $\mathcal{C}_2$
 $\mathcal{C}_0$	-	0.00032	0.00009
 $\mathcal{C}_1$	0.00032	-	0.00002
 $\mathcal{C}_2$	0.00009	0.00002	-

(b) *Les noyaux complexes simpliciaux avec* $\lambda = 0.001$

Table 0.1: *Les noyaux de graphes et de complexes simpliciaux entre les graphes et les complexes simpliciaux correspondants*

Chapter 1

Introduction

1.1 Motivations

Exploring and understanding complex structures such as random graphs is a difficult and rich problem that has motivated an abundant literature in the last years. A first natural step when facing large graphs is to look for a cluster or community structures [22, 42], that is a partition where the connectivity inside a class is higher than the connectivity between classes (this is illustrated in Figure 1.1). There are several ways of doing this, such as for instance modularity clustering [38, 11] or spectral clustering [44, 48]. The theory underlying this latter method is in particular popular because it establishes a link between the topology of the graph and nearest neighbor random walks on it (see [20, 32] for an introduction). These random walks are Markov processes that visit the vertices of the graph by jumping in continuous time from their current position v to a vertex chosen uniformly among the neighboring vertices. Consider a finite non-oriented graph $G = (V, E)$ consisting of the (finite) sets of vertices V and edges E determining the pairs of vertices that are connected. The adjacency matrix A of G is defined as the matrix whose entry at line u and column v is 1 if and only if (u, v) is an edge of G, which we will denote by $u \sim v$. For any $u \in V$ and for any function f from V to $\mathbf{R}$, the generator of the random walk is

$$Lf(u) = \sum_{v \sim u} \big(f(v) - f(u)\big) = -\big(D - A\big) f(u), \tag{1.1}$$

where D is the diagonal matrix containing the degrees of the vertices. Thus, the generator of the random walk is the opposite of the Laplacian of the graph, $D - A$. In order to find clusters in graphs, almost disconnected components of the graphs are always of great importance. It is well-known that the dimension of the kernel of $D - A$ is equal to the number of connected components of the graph, and spectral clustering aims at considering small eigenvalues in absolute values, that correspond to almost disconnected components.

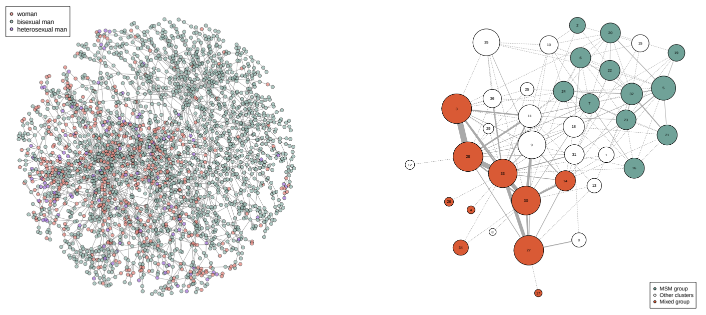


Figure 1.1: *Clustering of the graph of sexual connections among seropositive HIV individuals in Cuba from [11]*

While existing algorithms are well-suited to study the connectivity of a graph, there does not exist many tools to study the full topology of the graph. For instance, let us consider a circular-like graph presented in Figure 1.2. As soon as the cycle in the graph is slightly perturbed, for instance by addition of a point forming a new cycle as in Figure 1.2, it is not clear how to highlight the circular structure of the graph automatically. This is not surprising as graphs are not the correct tool to deal with higher order topology in the data. The correct structures one should use to deal with topology are simplicial complexes. The definition of simplicial complexes is recalled later, but a natural simplicial complex associated with a graph is obtained by adding to the pair (V, E) the set $\mathcal{S}_2$ of all triangles whose edges belong to E, the set of all tetrahedrons $\mathcal{S}_3$ whose triangles belong to $\mathcal{S}_2$ etc. For instance, if we add a triangle in the previous example, we recover a circular structure (see Figure 1.2(c)). This notion of circular structure is formalized

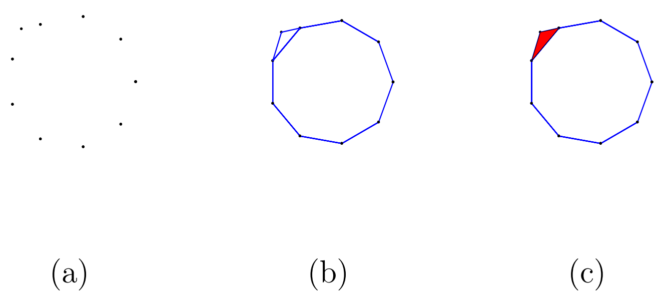


Figure 1.2: *Example where simplicial complexes are the correct structure to capture the topology of the data, and in particular detect a circular structure. (a): Data drawn from a circular structure. (b): Neighborhood graph structure, that reveals two different circular structures: a triangle and a circle. (c): A simplicial complex recovers the topology of the data.*

by the concepts of homology classes and Betti numbers which we present in Section 2.2.3. The first and second Betti numbers represent the number of circular and spherical holes in the complex. Notice that the Betti number of order zero corresponds to the number of connected components, so it is a natural idea to look for a generalization of the links between Betti numbers, graph Laplacians and random walks.

This is the purpose of the paper to give a probabilistic fundation, via random walks, of graph Laplacians and Betti numbers. Then, just as knowing the connected components of a graph only give limited information on the overall connectivity of the graph, one can expect to be able to go further than simply knowing the Betti numbers of a simplicial complex. To this end, we want to extend graph analysis algorithm to simplicial complexes. As most graph analysis algorithms rely on properties of the spectrum of the graph Laplacian or on properties of a random walk on a graph, adapting these objects to the case of simplicial complexes should yield simplicial complex counterparts of many graph analysis algorithms.

1.2 Contributions and outline

The thesis is organized as followed. First, we introduce the related works in the first part of Chapter 2. There are a lot of works on simplicial complexes in a topological way. We briefly introduce them and propose to explore in a probabilistic way. In fact, for graphs, the graph Laplacian $D - A$ is equal to the opposite of the generator of the random walk on the graph. However, for simplicial complexes, the combinatorial Laplacian, which is the generalization of graph Laplacian, cannot be transformed into the generator of any random walk. The representations of the works on graph analysis algorithms and simplicial complexes give us an insight that we can define a random walk on a simplicial complex in a different way. In the latter part of Chapter 2, we introduce the mathematical background. Indeed, we introduce simplicial complexes, chains and co-chains, boundary maps and co-boundary maps, the combinatorial Laplacian, and the relationship between the combinatorial Laplacian of order 0 and the opposite of graph Laplacian. For combinatorial Laplacian of order k, we divide it into two parts which are related to two matrices of upper and lower adjacency matrices of the simplicial complex respectively. Examples and remarks are also given. In the following, our contributions will be presented in the consecutive chapters.

As we have discussed before, it is difficult to transform the combinatorial Laplacian into the generator of any random walk. Therefore, we interpret the generator of random walk in a different way. For random walk from vertex to vertex, the random walk jumps from the current vertex u to a uniformly chosen adjacent vertex v. We can interpret the adjacent vertex v as $u + (v - u)$, and $(v - u)$ is the boundary of edge $[uv]$. For random walk from edge to edge, we add the boundary of a triangle to an edge, which gives a combination of edges, *i.e.* a 2-chain. Thus, we consider a random walk that takes its values in chains $\mathcal{C}_2$ or more generally $\mathcal{C}_k$. The exact definitions of random walk and details are described in Chapter 3. We define the space of test functions and transition kernels in order to define the generator of our random walk. We have shown that the generator of our random walk A, $-L_k$ and $-L_k^{\uparrow}$ coincide when restricted to $\ker \partial_k$. Under the assumption that the initial state of random walk $X(0)$ is simple and orientable, the random walk X is necessarily recurrent if any k-simplex set of our simplicial complex is finite. Furthermore, we can give an upper bound of the lower adjacency degree of any k-simplex of our simplicial complex under the same assumption. We also give some examples of random walk on special simplicial complexes, such as a triangulation in a torus, a simple triangulation of the sphere. In addition, we focus on the situation where we have a very regular structure, and we classify the paths by their lengths. We find that if the simplicial complex generated by the recurrence class of $X(0)$ is a "regular structure", any two chains of the same length have the same stationary probability. In the last part of Chapter 3, we investigate the continuous diffusive limits of our random walk. We consider the random walk on V_n, the set of vertices of the regular triangulation of a torus, and we show that the generator of this random walk converges to the Laplacian on the torus. Furthermore, we prove the analogous theorem for the random walk on $\mathcal{C}^1(V_n)$, where we find that the generator of the random walk converges to the upper Laplace-Beltrami operator on differential forms.

Once the random walk has been well defined, we can consider some of its applications on simplicial complexes. In Chapter 4, we introduce the first application, that is to detect the location of holes of a certain simplicial complex using our random walk. In this chapter, we consider the random walk on a simplicial complex with holes. Since the state space of the random walk consists of all the states in the same homology group, the paths with minimal lengths are supposed to be the holes of the simplicial complex. Thus, we manage to minimize the length of the paths of our random walk. In order to make simulations of our random walk, we need to determine the initial state of random walk. In this chapter, we consider two cases: random walk with known simple initial state and random walk with unknown initial state. In the first case, with known initial state, $X(0)$ is simple and the state space is in fact $\{-1, 0, 1\}^{\mathcal{S}_k}$. In the second case, it becomes more complicated. Since for any $t > 0$, $X(t)$ and $X(0)$ are always in the same homology group, we need to find the initial state $X(0)$ which is a cycle and in the same homology group as the holes. By Hodge decomposition, the eigenvectors of L_k corresponding to all the zero eigenvalues are the representative cycles of all the homology classes. Therefore, we generate our initial state by computing the eigenvectors corresponding to zero eigenvalues. Since the initial state is an integer larger than 1 in practice, the state space is no longer $\{-1, 0, 1\}^{\mathcal{S}_k}$. Therefore, we introduce an integer weighted random walk, which is an extension of our random walk defined in Chapter 3. Whether we know the initial state or not, we compute the length of the path at each step, and we look for the path with minimal length. Simulated annealing is a heuristic method to find an approximation of the global optimum. We propose a simulated annealing algorithm to minimize the lengths of paths, and we visualize the path with minimal length in the simplicial complex and see if it detects all the holes correctly. The language we use is Python 3.7 and we use the GUDHI package to compute a simplicial complex as a simplex tree, and the complexity of the algorithm is computed. We also give the convergence rate of the simulated annealing algorithm by constructing the cycle decomposition in the simulated annealing framework. At last, we randomly generate a Rips complex with 2 holes and make some simulations on it with and without known initial states in order to analyze the performances of the algorithm.

For another application of the random walk, we consider the random walk based kernels in Chapter 5. For higher order topology data, we can construct simplicial complex to store high order structures, such as triangles, tetrahedron and so on. In order to compare high order data structures, we intend to measure the similarity between simplicial complexes by defining a kernel between them. As we know, graphs are natural data structures with nodes representing objects and edges the relations between them, and Vishwanathan [47] has defined the graph kernel based on the random walk jumping from vertex to vertex. We would like to extend the definition of graph kernels to simplicial complexes kernels based on the chain-valued random walk defined in Chapter 3. The basic idea is that given a pair of data structures, we generate a pair of simplicial complexes respectively, then we perform our random walks on both and count the number of matching walks. The more matching walks, the more similar these two data structures are. Instead of counting the matching walks separately, we generate a direct product simplicial complex, perform our chain-valued random walk on it, and compute the sum of probabilities of our paths for all lengths. We propose an algorithm to find all the states in the state space of our random walk on the direct product simplicial complex, with which we can compute the transition matrix and the probabilities of our paths at any length. We also discuss the computational complexity when we compute the sum of probabilities, and we develop iterative methods such as conjugate gradient method to decrease the computational complexity. We compute graph kernels and simplicial complexes kernels for a pair of data structures. The performance results of these computation are presented in the last part of this chapter.

Finally, the Chapter 6 draws the conclusion. Also in this chapter, notable contributions are reminded, and some possible directions for future works are discussed.

Chapter 2

Related works and mathematical background

2.1 Related works

Simplicial complexes generalize the notion of triangulation of a surface and are constructed by gluing together simplices: points, edges, triangles and their higher dimensional counterparts. Simplicial complexes can be considered, at the same time, as continuous objects carrying topological and geometric information and as combinatorial data structures that can be efficiently implemented. There is a large literature on the subject of simplicial complexes.

Delaunay complexes are fundamental data structures that have been extensively studied in computational geometry and used in many application areas. Boissonnat *et al.* [6] introduce simplicial complexes which have strong ties with Delaunay complexes. Edelsbrunner, Kirkpatrick and Seidel [19, 18] first define the alpha-complex, or α-complex, of a finite set of points, which is a subcomplex of the Delaunay complex. Alpha-shapes are also widely used to represent union of balls [16] and to study the structure of macro molecules and various related problems like the docking of two molecules, see *e.g.* [17]. Alpha shapes are constructed from the Delaunay complex and are therefore of high complexity with high dimensions, De Silva [12] obtains smaller complexes by choosing a set of "landmark" points from our data set, and then constructing a “witness complex” on this set using ideas motivated by the usual Delaunay complex in Euclidean space. The identity of witness and Delaunay complexes when the number of witnesses is finite is taken from [7]. In order to compute homology, Kaczyński *et al.* [27] use reduction of chain complexes. For witness complexes, De Silva *et al.* [12] use witness complexes reduction algorithm, which is a reduction to a chosen number of points, to compute topological invariant. Apart from the reduction algorithms, Kahle [30] studies the expected topological properties of Čech and Vietoris–Rips complexes built on random points in $\mathbf{R}^d$. The author identifies thresholds for nonvanishing and vanishing of homology groups and also derive asymptotic formulas and bounds on expectations of the Betti numbers.

However, the homology invariants, such as Betti numbers, carry very limited information of the simplicial complexes. For example, even though we can infer the important features of shapes such as their homology (*e.g.* the number of holes in a simplicial complex), we are not able to locate them. Instead of exploring the topological and geometric information directly from the simplicial complexes, we are more interested in exploring them in a probabilistic way. It is already known that the graph Laplacian $D - A$ is a specific instance of the more general combinatorial Laplacian, introduced by Eckmann [15] and that we define in the following chapters. In a similar way that the graph Laplacian contains information regarding the connectivity of the graph, these combinatorial Laplacians describe the structure of the homology groups of the simplicial complex and are related to higher order Betti numbers.

In fact, there is no clear way of how to define random walks on simplicial complexes. For graphs, as has been recalled in (1.1), it is known that the graph Laplacian is equal to the opposite of the generator of the random walk on the graph. Since, graph Laplacians are generalized by combinatorial Laplacians, it was proposed in [40, 41] to define random walks on simplicial complexes as random walks with generator equal to the opposite of the combinatorial Laplacian. However, it is not easy to transform combinatorial Laplacians into generators of a random walks. In fact, a combinatorial Laplacian can be decomposed into a sum of two operators called up-Laplacian and down-Laplacian which each corresponds to a different random walk. Since we require two different random walks to characterize a single combinatorial Laplacian, it is not clear how to generalize graph algorithms based on random walks.

In this paper, we propose to define a random walk on a simplicial complex in a totally different way. More precisely, we consider a random walk on the space of chains of the simplicial complex whose transitions are given by the very definition of homology groups. In particular, similarly to how a random walk on a graph cannot leave a given connected component, which is a homology class of dimension 0, our random walk is bound to stay in the homology class of its initial state. Moreover, the generator of this random walk can be related to the up-Laplacian of the simplicial complex; we can thus use our random walk to provide some intuition regarding the relationship between the spectrum of the combinatorial Laplacians and the topology of the corresponding simplicial complex. For instance, since our random walk always stays in the homology class of its initial state, one can find the minimal length of the paths of random walk by finding the global optimum of a cost function using simulated annealing algorithm [31]. Besides, in machine learning, we are always interested in the relationships between structured objects for clustering or other purposes. Kernel methods [45] offer a natural framework to study these questions. The idea of constructing kernels between graphs was first proposed by Gärtner *et al.* [23], and extended by Borgwardt *et al.* [9]. Vishwanathan *et al.* [47] present a unified framework to study random walk graph kernel and present new algorithms for efficiently computing such kernels. Still, we can extend the graph kernel to simplicial complexes using our random walk.

Apart from the applications mentioned before, having probabilistic interpretation of topological properties of simplicial complexes is important since there is a growing literature on the subject in recent years. While the relations between graphs and simplicial complexes are considered in [14], papers more focused on the simplicial complexes themselves include for example studies of the Betti numbers and volume-like computation for random clique complexes built over the Erdös-Rényi graphs [28, 29, 39] or Čech and Vietoris–Rips complexes built over stationary point processes [1, 13, 50], or computation consideration of convex hulls of simplicial complexes [24].

2.2 Mathematical background

2.2.1 Simplicial complexes

As explained in the introduction, a natural generalization of graphs requires to consider simplicial complexes and adopt considerations from the field of homological and algebraic topology. For further reading on algebraic topology, see [2, 25, 37]. Graphs can be generalized to more generic combinatorial objects known as simplicial complexes. While graphs model binary relations, simplicial complexes represent higher order relations.
Given a finite or denumerable set of vertices V, a k-simplex is an unordered subset $\{v_0, v_1, \ldots, v_k\}$ where $v_i \in V$ and $v_i \neq v_j$ for all $i \neq j$. The faces of the k-simplex $\{v_0, v_1, \ldots, v_k\}$ are defined as all the $(k-1)$-simplices of the form $\{v_0, \ldots, v_{j-1}, v_{j+1}, \ldots, v_k\}$ with $0 \leq j \leq k$. The cofaces of a k-simplex τ are all the $(k+1)$-simplices of which τ is a face. A simplicial complex $\mathcal{C}$ is a collection of simplices which is closed with respect to the inclusion of faces, i.e. if $\{v_0, v_1, \ldots, v_k\}$ is a k-simplex of $\mathcal{C}$, then all its faces are in the set of $(k-1)$-simplices of $\mathcal{C}$. We denote by $\mathcal{S}_k(\mathcal{C})$ the set of k-simplices of $\mathcal{C}$. In the sequel, when there is no ambiguity, we will drop the dependency on $\mathcal{C}$ and simply write $\mathcal{S}_k$. By convention, $\mathcal{S}_0 = V$ consists of all the vertices. $\mathcal{S}_1$ of all the edges $\{v_0, v_1\}$ of $\mathcal{C}$ linking two vertices v_0 and $v_1 \in V$, $v_0 \neq v_1$. $\mathcal{S}_2$, $\mathcal{S}_3$ are the set of all triangles and tetrahedra of $\mathcal{C}$ etc. Then,

$$\mathcal{C} = \bigcup_{k \geq 0} \mathcal{S}_k.$$

One can define an orientation on simplices by defining an order on vertices and with the convention that:

$$[v_0, \ldots, v_i, \ldots, v_j, \ldots, v_k] = -[v_0, \ldots, v_j, \ldots, v_i, \ldots, v_k],$$

for $0 \leq i, j \leq k$. Each simplex may thus appear in two ways: positively or negatively oriented. Let us consider an orienting edge $[v_0, v_1]$. We denote by $\mathcal{S}_k^+$ (respectively $\mathcal{S}_k^-$) the set of positively (respectively negatively) oriented k-simplices, i.e. which have the orientation deduced from $[v_0, v_1]$ (resp. the inverse orientation). Then, $\mathcal{S}_k$ is the disjoint union of $\mathcal{S}_k^+$ and $\mathcal{S}_k^-$ and that $\mathcal{S}_k$ can be viewed as a subset of V^k which itself can be embedded in $\subset \mathbf{N}^k$.

For an oriented edge $[u, v]$ (going from u to v), when $[u, v] \in \mathcal{S}_1$, we will write $v \sim u$.

Example 1 (Čech complex)**.** *For $V = \{v_i, i = 1, \cdots, n\}$ n points in $\mathbf{R}^d$ (or in a metric space), and $R > 0$, the Čech complex Čech(V, R) of radius R is defined as follows: $\mathcal{S}_0 = \{v_i, i = 1, \cdots, n\}$ and $[v_{i_0}, v_{i_1}, \cdots, v_{i_k}]$ belongs to $\mathcal{S}_k$ whenever*

$$\bigcap_{m=0}^{k} B(v_{i_m}, R) \neq \emptyset.$$

This complex has the property that its topological features, as defined below, reflect that of the geometric set $\bigcup_i B(v_i, R)$.

Example 2 (Rips-Vietoris complex)**.** *Unfortunately, the construction of the Čech complex is exponentially hard so it is very common to work with the Rips-Vietoris complex. The simplicial complex Rips(V, R) has the same vertices V and edges as the Čech complex, but for $k \geq 3$, $\{v_{i_0}, \cdots, v_{i_k}\}$ belongs to $\mathcal{S}_k$ whenever all the possible pairs made by choosing two points among $\{v_{i_0}, \cdots, v_{i_k}\}$ belong to the set $\mathcal{S}_1$ of edges of Rips(V, R). Otherwise stated, the Rips-Vietoris is fully determined by the vertices and edges of the Čech complex. This graph is called the skeleton of the Rips-Vietoris. Though a priori coarser than the Čech complex, the Rips-Vietoris is not that far since (see [5])*

$$\textit{Rips}(V, R) \subset \textit{Čech}(V, R) \subset \textit{Rips}(V, 2R).$$

2.2.2 Chains and co-chains

For each integer k, $\mathcal{C}_k$ is the $\mathbf{R}$-vector space spanned by the set $\mathcal{S}_k^+$ of k-simplices of V: an element $\tau \in \mathcal{C}_k$ is called a *chain* or k-*chain* and can be uniquely written as

$$\tau = \sum_{s \in \mathcal{S}_k^+} \lambda_s(\tau)\, s, \tag{2.1}$$

where all but a finite number of $\{\lambda_s(\tau) \in \mathbf{R},\ s \in \mathcal{S}_k^+\}$ are non null. We define the support of τ to be:

$$\operatorname{supp} \tau = \{s \in \mathcal{S}_k^+,\ \lambda_s(\tau) > 0\}. \tag{2.2}$$

The set $\mathcal{C}_k$ of k-chains is equipped with the topology of $l^2(\mathbf{N}^k)$ by defining the norm as

$$\|\tau\|^2_{\mathcal{C}_k} = \| \sum_{s\in\mathcal{S}_k^+} \lambda_s(\tau)\, s\|^2_{\mathcal{C}_k} = \sum_{s\in\mathcal{S}_k^+} |\lambda_s(\tau)|^2.$$

Equations (2.1) and (2.2) amounts to define a scalar product that makes $\mathcal{C}_k$ a Hilbert space and $\{\tau,\ \tau \in \mathcal{S}_k^+\}$ an orthonormal basis of $\mathcal{C}_k$.

Let $\mathcal{C}^k$ be the topological and algebraic dual of $\mathcal{C}_k$:

$$\mathcal{C}^k = \Big\{ f\ :\ \mathcal{C}_k \to \mathbf{R},\ \ \text{linear and continuous} \Big\}.$$

Notice that the dimension of $\mathcal{C}_k$ is finite, so every linear form on $\mathcal{C}_k$ is continuous. Because $\mathcal{C}_k$ is a Hilbert space, so is $\mathcal{C}^k$. Note that $\mathcal{C}_k$ and $\mathcal{C}^k$ are isomorphic and that any element $\tau \in \mathcal{S}_k$ can be viewed either as an element of $\mathcal{C}_k$ or as an element of $\mathcal{C}^k$ by identification by the canonical isometries between an Hilbert space and its dual (see Example 3 below). When it will be convenient, we will indifferently manipulate chains or cochains in what follows as it is the most intuitive depending on the situation.
Also, any function from $\mathcal{S}_k$ to $\mathbf{R}$ can be associated canonically with a function from $\mathcal{C}^k$ to $\mathbf{R}$.

Example 3. *To illustrate the two assertion above, let us consider the case $k = 0$. We can identify the vertex $v \in \mathcal{S}_0 = V \subset \mathcal{C}_0$ to the function*

$$\begin{array}{cccc} v^* & : & \mathcal{C}_0 & \to & \mathbf{R} \\ & & v & \mapsto & 1 \\ & & u \neq v & \mapsto & 0. \end{array}$$

Then, we can extend any function $\varphi\ :\ V \to \mathbf{R}$ to a function $\varphi \in \mathcal{C}^0$ by

$$\varphi\big(\sum_{v\in V} \lambda_v v\big) := \sum_{v\in V} \lambda_v \varphi(v).$$

2.2.3 Boundary and coboundary maps

For any integer k, the boundary map ∂_k is the linear transformation ∂_k : $\mathcal{C}_k \to \mathcal{C}_{k-1}$ which acts on basis elements $[v_0, \dots, v_k]$ as

$$\partial_k[v_0,\, \dots,\, v_k] = \sum_{i=0}^{k} (-1)^i [v_0,\, \dots,\, v_{i-1},\, v_{i+1},\, \dots,\, v_k], \tag{2.3}$$

and ∂_0 is the null function. Examples of such operations are given in Table 2.1.

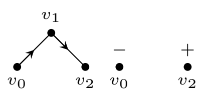
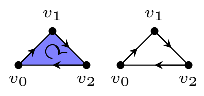
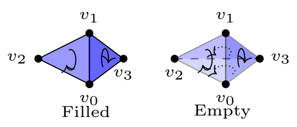
 $[v_0,v_1]+[v_1,v_2] \xrightarrow{\partial_2} [v_2]-[v_0]$	 $[v_0,v_1,v_2] \xrightarrow{\partial_3} [v_1,v_2]-[v_0,v_2]+[v_0,v_1]$	 v_1 v_2 v_3 v_0 Filled v_1 v_2 v_3 v_0 Empty $[v_0,v_1,v_2,v_3] \xrightarrow{\partial_4} +[v_1,v_2,v_3] -[v_0,v_2,v_3] +[v_0,v_1,v_3] -[v_0,v_1,v_2]$
a)	b)	c)

Table 2.1: *Examples of boundary maps. From left to right. An application over 1-simplices. Over a 2-simplex. Over a 3-simplex, turning a filled tetrahedron to an empty one.*

The maps (∂_k, $k \geq 1$) link the spaces $\mathcal{C}_k$'s as follows:

$$\cdots \xrightarrow{\partial_{k+2}} \mathcal{C}_{k+1} \xrightarrow{\partial_{k+1}} \mathcal{C}_k \xrightarrow{\partial_k} \cdots \xrightarrow{\partial_2} \mathcal{C}_1 \xrightarrow{\partial_1} \mathcal{C}_0. \tag{2.4}$$

It can then easily be checked that for any integer k,

$$\partial_k \circ \partial_{k+1} = 0. \tag{2.5}$$

In topology, a sequence of vector spaces and linear transformations satisfying (2.4) and (2.5) is called a chain complex, for which one can define the k-th homology vector space H_k as follows. If one defines

$$Z_k = \ker \partial_k \text{ and } B_k = \operatorname{im} \partial_{k+1}, \tag{2.6}$$

(2.5) induces that $B_k \subset Z_k$. Then, H_k is defined as the quotient vector space,

$$H_k = Z_k / B_k \tag{2.7}$$

and the k-th Betti number of $\mathcal{C}$ is defined as its dimension:

$$\beta_k = \dim H_k = \dim Z_k - \dim B_k. \tag{2.8}$$

Notice that an element of $\mathcal{C}_1$ is a sum of edges. It belongs to $\ker \partial_1$ whenever these oriented edges form a cycle, in the sense of graph theory. So

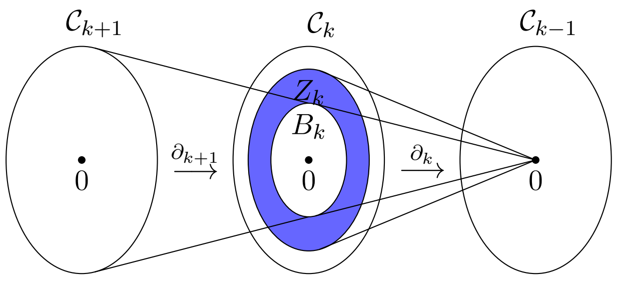


Figure 2.1: *A chain complex showing the sets $\mathcal{C}_k$, Z_k and B_k.*

elements of $Z_k = \ker \partial_k$ are called k-cycles.

Figure 2.1 illustrates the chain complex described above.

Example 4. *Let us consider the case of $k = 0$ again to illustrate (2.8). Since $\partial_0 \equiv 0$, we have $Z_0 = V$. Also, we have $B_0 = \operatorname{im} \partial_1 = \{u - v,\ [u, v] \in \mathcal{S}_1\}$. Hence, $H_0 = span(V)/\{u - v,\ [u, v] \in \mathcal{S}_1\}$ consists of equivalence classes of vertices which can be linked by a path in the graph. We thus recover that β_0 is the number of connected components of the graph. Recall that the latter number also corresponds to the number of zeros in the spectrum of the graph Laplacian (1.1). A natural question is then to wonder whether there is also a possible probabilistic connection between the other Betti number β_k and random walks on simplicial complexes.*

Remark 1. *An element of Z_1 which is not in B_1 is a cycle which cannot be written as a sum of triangles in $\mathcal{S}_2$.*
For instance, if

$$\mathcal{S}_1 = \{[ab], [bc], [cd], [ad]\} \text{ and } \mathcal{S}_2 = \emptyset,$$

then the cycle

$$[ab] + [bc] + [cd] - [ad]$$

which corresponds to the edges of a 4-gone, cannot be written as a sum of triangles since $\mathcal{S}_2$ is empty. Thus $\beta_1 = 1$ in this case.

As $\mathcal{C}_k$ and $\mathcal{C}^k$ are Hilbert spaces, we can define the coboundary map $\partial_k^* : \mathcal{C}^{k-1} \longrightarrow \mathcal{C}^k$ as the adjoint of ∂_k: namely for $f \in \mathcal{C}^{k-1}$, $\partial_k^* f \in \mathcal{C}^k$ is defined by its action over a k-chain by

$$\begin{aligned}(\partial_k^* f)[v_0,\cdots,v_k] &= \sum_{i=0}^{k}(-1)^i \langle f,\ [v_0,\cdots,v_{i-1},v_{i+1},\cdots,v_k]\rangle_{\mathcal{C}^{k-1},\mathcal{C}_{k-1}} \\ &= f\big(\partial_k[v_0,\cdots,v_k]\big). \qquad (2.9)\end{aligned}$$

We will set by convention $\partial_0^* \equiv 0$.

Remark 2 (Interpretation of the coboundary map in the case $k=1$)**.** *Recall that $\mathcal{C}_0$ is generated as a* $\mathbf{R}$*-vector space by the points $v \in V$. Let us denote by $\{v^*,\ v \in V\}$ the corresponding dual basis of $\mathcal{C}^0$: $v^*(v)=1$ and $v^*(w)=0$ for $w \neq v$. Hence, following (2.9), we have for any function $f \in \mathcal{C}^0$,*

$$\partial_1^* f[v_0,v_1] = f(v_1) - f(v_0).$$

In particular, if $f = w^*$ for $w \in V$,

$$(\partial_1^* w^*)([v_0,v_1]) = -\langle w^*, v_0\rangle_{\mathcal{C}^0,\mathcal{C}_0} + \langle w^*, v_1\rangle_{\mathcal{C}^0,\mathcal{C}_0} = \begin{cases} 0 & \textit{if } w \neq v_0, v_1 \\ -1 & \textit{if } w = v_0 \\ 1 & \textit{if } w = v_1. \end{cases}$$

The above computation shows that the coboundary of a vertex w gives a weight 1 (resp. -1) to oriented edges arriving at (resp. departing from) w. The coboundary map can then be interpreted in terms of fluxes.

Example 5. *Let us consider an example with four vertices: a, b, c and d. Consider that the edge $[a,b]$ belongs to the two triangles $[a,b,c]$ and $[a,b,d]$. Locally, the matrix representation of ∂_2 looks like*

$$\begin{array}{c} \\ {[a,b]} \\ {[a,c]} \\ {[a,d]} \\ {[b,c]} \\ {[b,d]} \end{array} \begin{array}{c} \begin{array}{cc} [a,b,c] & [a,b,d] \end{array} \\ \begin{pmatrix} 1 & 1 \\ -1 & 0 \\ 0 & -1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \end{array}.$$

The matrix representation of ∂_2^* is of course the transposed of the matrix representing ∂_2 and recalling that we have identified $\mathcal{C}_1$ (resp. $\mathcal{C}_2$) to its dual

$\mathcal{C}^1$ (resp $\mathcal{C}^2$),

$$\partial_2^*[a,b] = [a,b,c] + [a,b,d] = \sum_{\eta \in \mathcal{S}_2^+} \langle [a,b]^*,\, \partial_2 \eta \rangle^+_{\mathcal{C}^1,\mathcal{C}_1}\ \eta. \tag{2.10}$$

The bracket on the right hand side counts the occurrence of the edge $[a,b]$ among the faces of η. Otherwise stated, the coboundary of an edge is the sum of the triangles which contain it, respecting its orientation.

As before, the k-th cohomology vector space, denoted by H^k, is defined as

$$H^k = \ker \partial_k^* / \operatorname{im} \partial_{k-1}^* \tag{2.11}$$

and is the dual of H_k.

2.2.4 Combinatorial Laplacian

A crucial notion for the following is that of combinatorial Laplacian (see [21, 34] for details). We know that

$$\begin{tikzcd}
{} \arrow[r, dashed, "\partial_{k+2}"] & \mathcal{C}_{k+1} \arrow[r, "\partial_{k+1}"] \arrow[d, leftrightarrow] & \mathcal{C}_k \arrow[r, "\partial_k"] \arrow[d, leftrightarrow] & \mathcal{C}_{k-1} \arrow[r, dashed, "\partial_{k-1}"] \arrow[d, leftrightarrow] & {} \\
{} & \mathcal{C}^{k+1} \arrow[l, dashed, "\partial_{k+2}^*"'] & \mathcal{C}^k \arrow[l, "\partial_{k+1}^*"'] & \mathcal{C}^{k-1} \arrow[l, "\partial_k^*"'] & {} \arrow[l, dashed, "\partial_{k-1}^*"']
\end{tikzcd}$$

where the two tips arrows mean that we have an isometric isomorphism between the two concerned Hilbert spaces. Since we have identified the spaces $\mathcal{C}_k$ and $\mathcal{C}^k$ for any $k \in \mathbf{N}$, we may consider

$$L_k^{\uparrow} := \partial_{k+1}\partial_{k+1}^* \,:\, \mathcal{C}_k \xrightarrow{\partial_{k+1}^*} \mathcal{C}_{k+1} \xrightarrow{\partial_{k+1}} \mathcal{C}_k \tag{2.12}$$

and

$$L_k^{\downarrow} := \partial_k^*\partial_k \,:\, \mathcal{C}_k \xrightarrow{\partial_k} \mathcal{C}_{k-1} \xrightarrow{\partial_k^*} \mathcal{C}_k. \tag{2.13}$$

These latter operators are called respectively the upper and lower Laplacians (of order k). The combinatorial Laplacian of order k is defined as

$$\begin{aligned} L_k \,:\, \mathcal{C}_k &\longrightarrow \mathcal{C}_k \\ \tau &\longmapsto \left(\partial_{k+1}\partial_{k+1}^* + \partial_k^*\partial_k\right)(\tau) = L_k^{\uparrow}(\tau) + L_k^{\downarrow}(\tau). \end{aligned} \tag{2.14}$$

The combinatorial Hodge theorem says that

Theorem 3. *For any $k \in \mathbf{N}$, we have*

$$\mathcal{C}_k = \operatorname{im} \partial_{k+1} \oplus \operatorname{im} \partial_k^* \oplus \ker L_k. \tag{2.15}$$

It follows that

$$\ker L_k \simeq H_k. \tag{2.16}$$

In particular, the k-th Betti number is the dimension of the null space of L_k:

$$\beta_k = \dim \ker(L_k). \tag{2.17}$$

Proof. Assume (2.15), implying that $\dim(\ker L_k) = \dim(\mathcal{C}_k) - \dim(\operatorname{im} \partial_{k+1}) - \dim(\operatorname{im} \partial_k^*)$. Because ∂_k^* and ∂_k are adjoint, $\dim(\operatorname{im} \partial_k^*) = \dim(\operatorname{im} \partial_k) = \dim(\mathcal{C}_k) - \dim(\ker \partial_k)$ by the rank-nullity theorem. Thus,

$$\dim(\ker L_k) = \dim(\ker \partial_k) - \dim(\operatorname{im} \partial_{k+1}) = \dim(H_k) = \beta_k.$$

This proves (2.17). $\square$

Remark 4. *The combinatorial Laplacian of order 0 corresponds to the opposite of the graph Laplacian (1.1). Since there are no simplex of negative order, we set $\mathcal{C}_{-1} = 0$ and ∂_0 to be the null map, hence $L_0 = L_0^{\uparrow} = \partial_1 \partial_1^*$. The map ∂_1 maps edges to vertices and its matrix representation is exactly the so-called incidence matrix B of the graph $(V, \mathcal{S}_1)$: for $V = \{v_0, \cdots, v_k\}$ and for (e_j) an enumeration of the set of oriented edges $\mathcal{S}_1^+$,*

$$B_{ij} = \begin{cases} 1 & \textit{if } v_i \textit{ is the ego of } e_j \\ -1 & \textit{if } v_i \textit{ is the alter of } e_j \\ 0 & \textit{otherwise.} \end{cases}$$

Thus, $L_0 = BB^t$ is such that

$$(L_0)_{ij} = \begin{cases} \deg(v_i) & \textit{if } i = j, \\ -1 & \textit{if } v_i \textit{ is adjacent to } v_j, \\ 0 & \textit{otherwise.} \end{cases}$$

The map L_0 from $\mathcal{C}_0$ into itself is characterized for $v \in V$ by:

$$L_0 v^* = - \sum_{w \in V : [vw] \in \mathcal{S}_1^+} (w^* - v^*) = \sum_{w \in V : [vw] \in \mathcal{S}_1^+} \partial_1 [vw].$$

Thus, for a function $f = \sum_{v \in V} \lambda_v v^* \in \mathcal{C}^0$,

$$\begin{aligned}
-L_0 f(u) &= \sum_{v \in V} \lambda_v L_0 v^*(u) \\
&= \sum_{v \in V} \lambda_v \sum_{w \in V : [vw] \in \mathcal{S}_1^+} (w^* - v^*)(u) \\
&= \sum_{v \sim u} \lambda_v - \lambda_u \mathrm{Card}(w \sim u) \\
&= \sum_{v \sim u} \big(f(v) - f(u)\big) = Lf(u). \qquad (2.18)
\end{aligned}$$

Thus as mentioned above, $-L_0$ appears as the generator L of the continuous time random walk on the graph $(V, \mathcal{S}_1)$ as defined in (1.1).

To describe L_k, we need to introduce the notion of lower and upper adjacency for k-simplices. Two k-simplices are said to be upper-adjacent whenever they are two faces of a common $k+1$ simplex. Two k-simplices are said to be lower-adjacent whenever they are cofaces of a common $k-1$ simplex. For a simplex τ, its upper degree, $\deg_\uparrow(\tau)$, is the number of simplices which are upper adjacent to it. Two upper adjacent simplices are said to be similarly oriented if the orientation they would inherit from their common higher order simplex coincides with their orientation. They are said to be dissimilarly oriented otherwise. Two lower adjacent simplices are similarly oriented whenever they induce the same orientation to their intersection.

Example 6. *For instance, two edges are upper adjacent if they are part of a common triangle and they are lower adjacent if they share a vertex.*

The lower and upper adjacency matrix are defined as it can be expected. For τ and $\tau' \in \mathcal{S}_k$,

$$A_k^{\uparrow/\downarrow}(\tau, \tau') = \begin{cases} 1 & \text{if } \tau \text{ and } \tau' \text{ are upper/lower adjacent and similarly oriented,} \\ -1 & \text{if } \tau \text{ and } \tau' \text{ are upper/lower adjacent and dissimilarly oriented,} \\ 0 & \text{otherwise.} \end{cases}$$

Consider also D_k the diagonal matrix whose entries are the upper degrees of each k-simplex.

We can compute the matrix of the combinatorial Laplacian of order k (2.14). Then (see [34]),

$$L_k = L_k^{\uparrow} + L_k^{\downarrow} = \left(D_k - A_k^{\uparrow}\right) + \left((k+1)\,\mathrm{Id} + A_k^{\downarrow}\right).$$

The map $L_k^{\uparrow}$ has the features of a generator of a Markov process. Indeed, considering back the computation in Remark 4, we see that the upper Laplacian can be explained in terms of upper-adjacency. For a vertex u ($k = 0$), we consider all the edges that are upper-adjacent to u, choose one with u as ego and jump to the alter of this edge. This can be generalized for larger orders k: we consider all the $k+1$ simplices that are adjacent to a given k simplex τ, choose one at random that will determine the next movement (this shall be precised in the sequel, see (2.14)). Hence, $L_k^{\uparrow}$ has the structure of the generator of a Markov process. Associating to $L_k^{\uparrow}$ a Markov chain provides a probabilistic interpretation to this operator which can help understand it better. In Parzanchevski and Rosenthal [40], a connection between this random walk Y and homology is made by considering the 'expectation process' defined for an oriented edge $e \in \mathcal{S}_1$ by $\mathcal{E}_t(e) = \mathbf{P}(Y_t = e) - \mathbf{P}(Y_t = -e)$. They show that the latter process converges, when correctly renormalized and under good conditions, to $\ker L_k^{\uparrow}$.

As to the map $L_k^{\downarrow}$, Mukherjee and Steenbergen [36] proposed a similar random walk exploiting the lower-adjacency and whose generator is related to $L_k^{\downarrow}$. The generator $L_k^{\downarrow}$ does not correspond to the generator of a Markov process, hence the complexity to define a Markov process whose generator would be $L_k^{\downarrow}$ and these authors introduce killings to deal with this problem.

It is not clear how to deal simultaneously with information coming from several random walks to obtain results on the combinatorial Laplacian L_k. However, the following result says that as long as we are concerned with spectral properties, we can retrieve the information about $L_k^{\downarrow}$ by looking at $L_{k-1}^{\uparrow}$ (see [51]):

Theorem 5. *Let $\lambda > 0$ be an eigenvalue and f be an eigenvector of $L_k^{\uparrow}$. Then, $\partial_k^* f$ is a λ-eigenvector of $L_k^{\downarrow}$. Conversely, if g is a λ-eigenvector of $L_k^{\downarrow}$, then $\partial_k g$ is a λ-eigenvector of $L_k^{\uparrow}$.*

Proof. If f satisfies $L_k^{\uparrow} f = \lambda f$, then

$$L_k^{\downarrow}(\partial_k^* f) = \partial_k^* \partial_k \partial_k^* f = \partial_k^* L_k^{\uparrow} f = \lambda \, \partial_k^* f.$$

The proof is similar for the converse. □

In the next chapter, we propose a new Markov chain that contains information on the homology of the simplicial complex and exploit the links between Betti numbers and homology spaces (2.7) to provide probabilistic interpretation of these quantities.

Chapter 3

Random walk and its continuous diffusive limits

In this chapter, we will define our chain-valued random walk on simplicial complexes and explore some properties of our random walk. First of all, we will introduce the idea and the motivation behind the dynamics of our random walk, and then we will give the definition of generator of our random walk and find the link between our generator and the combinatorial Laplacian introduced in Section 2.2.4. In this way, we can simulate the random walk corresponding to the combinatorial Laplacian using our random walk. At last, we study the continuous diffusive limits of the random walk, and prove the convergence of the generator of the random walk.

3.1 Introduction

The idea behind the dynamics of our random walk is the following. The usual random walk on a graph goes from vertex to vertex. The generator of the continuous time random walk on graphs can be written as

$$Lf(u) = \sum_{v \sim u} f(v) - f(u) = \sum_{v \in V : [uv] \in \mathcal{S}_1} f(u + \partial_1[uv]) - f(u). \tag{3.1}$$

In the next dimension, we have $k = 1$, and points are replaced by edges and edges by triangles. If we follow Parzanchevski and Rosenthal [40, 41], a natural edge-valued random walk consists in jumping from the current edge

e to a uniformly chosen upper-adjacent edge. Mukherjee and Steenbergen proposed a similar random walks exploiting the lower-adjacency. But if we look for an analogue of (3.1), another way is to add the boundary of a triangle to an edge, which gives a combination of edges, i.e. a 2-chain. It is thus natural not to restrict to edges, but to consider a random walk that takes its values in $\mathcal{C}_2$ or more generally $\mathcal{C}_k$.

Recall that in (2.7), a natural way to explore the homology classes of $H_k = \ker \partial_k / \operatorname{im} \partial_{k+1}$ is to start with an element of $\ker \partial_k$ and then have transitions in $\operatorname{im} \partial_{k+1}$. Proceeding so, the random walk will remain in the homology class of its initial element, in the same way as the usual random walk remained in the connected component of its initial state. We propose a random walk taking its values in $\ker \partial_k$. For $k = 1$, this corresponds to a cycle-valued random walk. Let us now describe the transitions and generator of this random walk.

The rest of this chapter is organized as follows. Section 3.2 describes the definition of random walk, including the generator and orientability of it, recurrence of the chain on finite graphs, and examples of random walk on some particular simplicial complexes. In Section 3.3, we discuss the convergence of random walk, and we give continuous diffusive limits of it.

3.2 Random walk

3.2.1 Generator of the chain-valued random walk

In what follows, $k \geq 1$ is fixed.

Space of test functions To define the generator of the random walk, we first introduce a space of functions on which it will operate.

Consider $\mathcal{D}$ the space of functions from $\mathcal{C}_k$ to $\mathbf{R}$ of the form

$$F(\tau) = f\left(\langle \eta_1, \tau \rangle_{\mathcal{C}^k, \mathcal{C}_k}, \cdots, \langle \eta_m, \tau \rangle_{\mathcal{C}^k, \mathcal{C}_k}\right) \tag{3.2}$$

for some $m \geq 1$, $(\eta_1, \cdots, \eta_m)$ some elements of $\mathcal{C}^k$ and f measurable and bounded from $\mathbf{R}^m$ into $\mathbf{R}$.

We define the support of F as:

$$\operatorname{supp} F = \bigcup_{i=1}^{m} \operatorname{supp} \eta_i,$$

where we recall that $\operatorname{supp} \eta_i$ is defined in (2.2).

Lemma 6. *The space $\mathcal{D}$ is separating in $B(\mathcal{C}_k)$, the Banach space of bounded measurable functions from $\mathcal{C}_k$ to $\mathbf{R}$, equipped with the sup-norm.*

Proof. Since $\mathcal{C}^k$ can be embedded as a closed subset of $l^2(\mathbf{N}^k)$, it is a separable Hilbert space, we can consider a dense sequence $(\eta_n,\ n \geq 1)$ and

$$\mathcal{F}_m = \sigma\{\langle \eta_i, .\rangle_{\mathcal{C}^k,\mathcal{C}_k}, i = 1, \cdots, m\}.$$

Since $\mathcal{C}_k$ is an Hilbert space, its Borel σ-field is equal to $\bigvee_m \mathcal{F}_m$, hence the result. □

Transition kernel Let us explain the transition of our chain-valued random walk. For the sake of simplicity, imagine here that $k = 1$. Assume that we are in a state $\tau \in \mathcal{C}_1$ (a cycle for $k = 1$). Because transitions are in $\operatorname{im} \partial_2$, let us consider an element of this space, say $\partial_2\eta$ for $\eta \in \mathcal{S}_2$ (a triangle). $\partial_2\eta$ defines a possible transition if η is upper-adjacent to τ, i.e. if η and τ share at least one edge. All the possible transitions from τ are obtained by letting η vary in $\mathcal{S}_2$. The more η and τ have edges in common and the more η will be likely to define the next step of the random walk and we thus need to define a weight to account for this.

For a k-chain $\tau \neq 0$ and an oriented $(k+1)$-simplex $\eta \in \mathcal{S}_{k+1}$, define the number of common faces between τ and $\partial_{k+1}\eta$ by:

$$w\left(\tau,\ \partial_{k+1}\eta\right) = \langle (\partial_{k+1}\eta)^*,\ \tau\rangle^+_{\mathcal{C}^k,\mathcal{C}_k}$$

where $x^+ = \max(x, 0)$ for $x \in \mathbf{R}$. For another chain τ', we say that τ and τ' are adjacent (in the sense that the random walk can reach τ' from the state τ) and write

$$\tau \sim \tau' \iff \exists \eta \in \mathcal{S}_{k+1}, w\left(\tau,\ \partial_{k+1}\eta\right) > 0 \text{ and } \tau' = \tau - \partial_{k+1}\eta. \tag{3.3}$$

Finally, let us define the weight of the transition from τ to τ':

$$K(\tau,\,\tau') = \begin{cases} 1 & \text{if } \tau = \tau' = 0 \\ w\left(\tau,\ \tau - \tau'\right) & \text{if } \tau \sim \tau' \\ 0 & \text{otherwise.} \end{cases} \tag{3.4}$$

Example 7. *In Figure 3.1, we see how a difference of orientations is simply reflected in the value of the scalar product. Note also that $w(\tau, \tau - \tau')$ can be viewed as a scalar product that counts the number of edges of the triangle which are adjacent to the chain with the good orientations.*

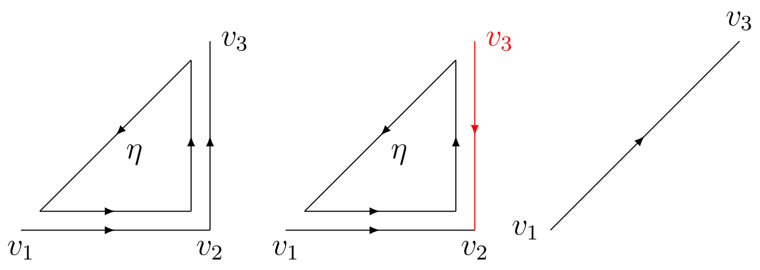


Figure 3.1: *Different cases of orientations: $\eta = [v_1v_2v_3]$. Here τ is a 1-chain, not necessarily a cycle. (a) In this case, $\tau = [v_1v_2] + [v_2v_3]$ and $w(\tau, \partial_2\eta) = 2$, which is the number of edges in common between τ and η. (b) Here, $\tau = [v_1v_2] - [v_2v_3]$ and $w(\tau, \partial_2\eta) = 0$. So η is never chosen for defining the transition to the next step here. (c) In the case (a), the next step is $\tau' = [v_1v_2] + [v_2v_3] - \partial_2\eta = [v_1v_3]$.*

Generator of the random walk Let us define by $(A, D(A))$ the generator of the continuous-time random walk.

Definition 1. *Let $D(A)$ be the set of functions F such that $|\sum_{\tau'\sim\tau}\Big(F(\tau') - F(\tau)\Big)\,K(\tau,\,\tau')| < +\infty$. For $F \in D(A)$, we can define*

$$\begin{aligned} AF\ :\ \mathcal{C}_k &\longrightarrow \mathbf{R} \\ \tau &\longmapsto \sum_{\tau'\sim\tau}\Big(F(\tau') - F(\tau)\Big)\,K(\tau,\,\tau'). \end{aligned}$$

Let $\mathcal{L}$ be the space of Lipschitz continuous functions F from $\mathcal{C}_k$ to $\mathbf{R}$, i.e. such that there exists $c_F > 0$ such that for any $\tau, \tau' \in \mathcal{C}_k$

$$|F(\tau) - F(\tau')| \le c_F \left(\sum_{\eta \in \mathcal{S}_{k+1}^+} |\lambda_\eta(\tau) - \lambda_\eta(\tau')|^2 \right)^{1/2} .$$

We remark that if τ and $\tau' \in \mathcal{C}_k$,

$$\tau \sim \tau' \Longrightarrow K(\tau, \tau') \le k+1,$$

and thus, K is a bounded kernel. Hence, for $F \in \mathcal{L}$,

$$\begin{aligned} \left| \sum_{\tau' \sim \tau} \Big(F(\tau') - F(\tau) \Big) K(\tau,\, \tau') \right| &\le c_F(k+1) \left(\sum_{\eta \in \mathcal{S}_{k+1}^+} |\lambda_\eta(\tau) - \lambda_\eta(\tau')|^2 \right)^{1/2} \\ &\le 2c_F(k+1) \left(\|\tau\|_{\mathcal{C}^k} + \|\tau'\|_{\mathcal{C}^k} \right) < +\infty, \end{aligned}$$

from which we deduce that $\mathcal{L} \subset D(A)$.

Since K is a bounded kernel, it is immediate that we have the following theorems (see Ethier and Kurtz [20, Chapter 4, Section 2 and Chapter 8, Section 3]).

Theorem 7. *The map A of domain $D(A)$ generates a strong Feller continuous Markov process $X = (X(t))_{t \ge 0}$ on $C_b(\mathcal{C}_k, \mathbf{R})$, the set of continuous bounded functions on $\mathcal{C}_k$. The set $\mathcal{A} = \mathcal{D} \cap C_b(\mathcal{C}_k, \mathbf{R})$ is a core for X.*

Remark that the process X is a continuous-time pure jump process and admits a representation with a discrete-time Markov chain and exponentially distributed clocks attached with each possible transitions (see e.g. [20, Chapter 4, Section 2]).

Theorem 8. *For any $t > 0$, $X(t)$ remains in the same homology class as $X(0)$. Moreover, if $X(0) \in \ker \partial_k$, then for any $t \ge 0$, $X(t)$ belongs to $\ker \partial_k$.*

Proof. At each change of state, we add to X an element of $\partial_{k+1}\mathcal{S}_{k+1} \subset B_k$ defined in (2.6). Since $\partial_k \circ \partial_{k+1} = 0$, we add only elements of $\ker \partial_k$ to $X(0)$, hence $X(t)$ always belongs to $\ker \partial_k$ and the homology class does not change along the dynamics of X. □

We can precise the link between A and $L_k^{\uparrow}$. Note that they cannot be equal since A operates on functions of chains whereas $L_k^{\uparrow}$ operates on cochains $\mathcal{C}^k$, i.e. linear functions of chains. However we will see that A, $-L_k$ and $-L_k^{\uparrow}$ coincide when restricted to $\ker \partial_k$.

Theorem 9. *For F a linear function of $\mathcal{A}$, we have for every $\tau \in \ker \partial_k$,*

$$AF(\tau) = -L_k^{\uparrow} F(\tau) = -L_k F(\tau). \tag{3.5}$$

Proof. Let us consider a chain $\tau \in \mathcal{C}_k$ and a function $\zeta \in \mathcal{C}^k$. Then,

$$A\zeta(\tau) = \sum_{\tau' \sim \tau} \big(\zeta(\tau') - \zeta(\tau)\big) K(\tau, \tau'). \tag{3.6}$$

By the definitions of the $\sim$ relation (see (3.3)) and of K (see (3.4)):

$$\begin{aligned}\sum_{\tau' \sim \tau} \big(\zeta(\tau') - \zeta(\tau)\big) K(\tau, \tau') &= - \Big\langle \zeta, \sum_{\eta \in \mathcal{S}_{k+1}} \partial_{k+1}\eta \langle (\partial_{k+1}\eta)^*, \tau \rangle^+_{\mathcal{C}^k, \mathcal{C}_k} \Big\rangle_{\mathcal{C}^k, \mathcal{C}_k} \\ &= - \Big\langle \partial^*_{k+1}\zeta, \sum_{\eta \in \mathcal{S}_{k+1}} \eta \, \langle (\partial_{k+1}\eta)^*, \tau \rangle^+_{\mathcal{C}^k, \mathcal{C}_k} \Big\rangle_{\mathcal{C}^{k+1}, \mathcal{C}_{k+1}} \\ &= - \langle \partial^*_{k+1}\zeta, \; \partial^*_{k+1}\tau \rangle_{\mathcal{C}^{k+1}, \mathcal{C}_{k+1}} \\ &= - \langle \partial_{k+1} \circ \partial^*_{k+1}\zeta, \; \tau \rangle_{\mathcal{C}^{k+1}, \mathcal{C}_{k+1}} = -L_k^{\uparrow}\zeta(\tau).\end{aligned}$$

When $\tau \in \ker \partial_k$, $L_k^{\downarrow}(\tau) = 0$. This concludes the proof. □

3.2.2 Orientability of the random walk

It is natural to wonder about the nature – transience or recurrence – of the continuous-time Markov chain that we have just introduced. If we start from an initial condition whose constituting simplices have integer weights, the state space of the process is *a priori* the space of chains with integer weights. Even if the number of vertices is finite, the state space of the process might be infinite. To study this, let us first introduce some considerations on orientation of the chains.

We say that a chain is *orientable* if we cannot find two elements τ and τ' of the chain which are lower adjacent with dissimilar orientation. In the following, we always assume that $X(0)$ is orientable. Note that a cycle, an element of $\ker \partial_k$ and a k-simplex are always orientable.

A chain τ is said to be simple if the weights of the k-simplices in its support are -1 or 1. We denote by $\mathcal{C}_k^s$ the set of such chains.

Lemma 10. *If $X(0)$ is a simple and orientable chain of $\mathcal{C}_k^s$, then so is $X(t)$ for any $t > 0$.*

Proof. Because X is a pure-jump process, it is sufficient to prove that after each jump, the new state of the process is still a simple chain. Assume that we are currently at a state τ that is a simple orientable k-chain. Recall that for the transition, we choose a $(k+1)$-simplex η which shares faces of same orientation with the current k-chain, τ (see (3.3)). Then, the transition consists in adding $-\partial_{k+1}\eta$, whose faces have an orientation contrary to that of τ. As a result, we always choose for the transition a chain whose orientation is contrary to that of the current state, which will give the announced result.

More precisely, let us detail the first jump after time 0. Proceeding by recursion will give the result after any arbitrary number of jumps. Using (2.1), we can write $X(0) = \sum_{s\in\mathcal{S}_k^+} \lambda_s(X(0))\ s$, where for all $s \in \mathcal{S}_k^+$, $\lambda_s(X(0)) \in \{-1, 0, 1\}$. Let $\eta \in \mathcal{S}_{k+1}$. Then $w(X(0), \partial_{k+1}\eta) > 0$ if and only if

$$0 < \big\langle(\partial_{k+1}\eta)^*, X(0)\big\rangle_{\mathcal{C}^k,\mathcal{C}_k} = \sum_{s\in\mathcal{S}_k^+} \lambda_s(X(0))\langle(\partial_{k+1}\eta)^*, s\rangle_{\mathcal{C}^k,\mathcal{C}_k}.$$

Because $X(0)$ and η are orientable, all the terms $\lambda_s(X(0))\langle(\partial_{k+1}\eta)^*, s\rangle$, for $s \in \mathcal{S}_k^+$, have the same sign. So the transition from $X(0)$ to $X(0) - \partial_{k+1}\eta$ is possible if and only if the previous terms are nonnegative and some of them are positive. Then, the state after the jump is:

$$\begin{aligned} X(0) - \partial_{k+1}\eta &= \sum_{s\in\mathcal{S}_k^+} \lambda_s(X(0))\ s - \sum_{s\in\mathcal{S}_k^+} \langle(\partial_{k+1}\eta)^*, s\rangle_{\mathcal{C}^k,\mathcal{C}_k}\ s \\ &= \sum_{s\in\mathcal{S}_k^+:\langle(\partial_{k+1}\eta)^*,s\rangle=0} \lambda_s(X(0))\ s + \sum_{s\in\mathcal{S}_k^+:\langle(\partial_{k+1}\eta)^*,s\rangle\neq0} \big(\lambda_s(X(0)) - \langle(\partial_{k+1}\eta)^*, s\rangle_{\mathcal{C}^k,\mathcal{C}_k}\big)s. \end{aligned} \tag{3.7}$$

For an admissible transition $-\partial_{k+1}\eta$, we have that $\lambda_s(X(0))\langle(\partial_{k+1}\eta)^*, s\rangle \geq 0$. So, for the second sum in the right hand side of (3.7):

- either $\lambda_s(X(0)) - \langle (\partial_{k+1}\eta)^*, s\rangle_{\mathcal{C}^k,\mathcal{C}_k} = 0$ whenever $\lambda_s(X(0)) \neq 0$, meaning that the face s does not belong to the simplicial complex $X(0) - \partial_{k+1}\eta$ any more after the jump,
- or, $\lambda_s(X(0)) = 0$ and after the jump:

$$\begin{aligned}\lambda_s(X(0) - \partial_{k+1}\eta) =& \lambda_s(X(0)) - \langle (\partial_{k+1}\eta)^*, s\rangle_{\mathcal{C}^k,\mathcal{C}_k} \\ =& - \langle (\partial_{k+1}\eta)^*, s\rangle_{\mathcal{C}^k,\mathcal{C}_k} \in \{-1, 1\},\end{aligned} \tag{3.8}$$

  so that at the next step, the move $-\partial_{k+1}\eta$ is no longer possible.

Thus, the states of the process after the first jump are increased or decreased by 1 but remain in $\{-1, 0, 1\}$.
Also, we can check that the chain $X(0) - \partial_{k+1}\eta$ remains orientable.

$\square$

Hence, there is no loss of generality to assume for the rest of this section that:

(H) : $X(0)$ is simple and orientable, implying that the state space is in fact $\{-1, 0, 1\}^{\mathcal{S}_k^+}$.

This remark has two consequences that we develop in the next subsections.

3.2.3 Recurrence of the chain on finite graphs

The first consequence of Lemma 10 deals with the nature (recurrent or transcient) of the random walk X. In the general case, there is no reason why its state space should be infinite, even when $\mathcal{S}_k$ is, since the weights λ_s in (2.1) can be unbounded. However, Assumption (H) and Lemma 10 ensure this, implying that the Markov chain is necessarily recurrent:

Corollary 11. *Under Assumption (H) and if $\mathcal{S}_k$ is finite, the state space of X is finite then the Markov chain is recurrent for any initial value.*

We can go a little further. For any k-simplex τ, let $d^-(\tau)$ be the lower adjacency degree of τ given by

$$d(\tau)^- = \sum_{\tau' \in \mathcal{S}_k, \tau' \neq \tau} |\langle \big(\partial_k(\tau)\big)^*, \partial_k(\tau')\rangle_{\mathcal{C}^{k-1},\mathcal{C}_{k-1}}|.$$

This quantity corresponds to the number of faces that τ and τ' have in common.

Theorem 12. *Let $k \in \mathbf{N}$. Let $\sigma_1, \dots, \sigma_{\beta_k} \in \mathcal{C}_k$ be a basis of H_k and let $(\tau_1, \dots, \tau_n) \in \mathcal{S}_{k+1}$ be the $(k+1)$-simplices of our simplicial complex. Suppose that there exists $\mu_1, \dots, \mu_{\beta_k} \in \mathbf{Z}$ and $(\lambda_\tau,\ \tau \in \mathcal{S}_{k+1}) \in \{-1, 0, 1\}^{\mathcal{S}_{k+1}}$ such that*

$$X(0) = \sum_{i=1}^{\beta_k} \mu_i \sigma_i + \sum_{\tau \in \mathcal{S}_{k+1}} \lambda_\tau \partial_{k+1} \tau.$$

If we have for all $\tau \in \mathcal{S}_{k+1}$ that:

$$d(\tau)^- \leq k + 2 - \Big| \sum_{i=1}^{\beta_k} \mu_i \langle (\partial_{k+1}\tau)^*, \sigma_i \rangle \Big|, \tag{3.9}$$

then X has a finite state space and for any $t > 0$, there exists $(\lambda_\tau(t), \tau \in \mathcal{S}_{k+1}) \in \{-1, 0, 1\}^{\mathcal{S}_{k+1}}$ such that

$$X_t = \sum_{i=1}^{\beta_k} \mu_i \sigma_i + \sum_{\tau \in \mathcal{S}_{k+1}} \lambda_\tau(t) \partial_{k+1} \tau.$$

Proof. First, let us consider $\eta \in \mathcal{S}_{k+1}$ such that $\lambda_\eta = 1$. We have

$$\begin{aligned}
\langle (\partial_{k+1}\eta)^*, X(0) \rangle_{\mathcal{C}^k, \mathcal{C}_k} &= \sum_{i=1}^{\beta_k} \mu_i \langle (\partial_{k+1}\eta)^*, \sigma_i \rangle + \sum_{\tau \in \mathcal{S}_{k+1}} \lambda_\tau \langle (\partial_{k+1}\eta)^*, \partial_{k+1}\tau \rangle \\
&\geq - \Big| \sum_{i=1}^{\beta_k} \mu_i \langle (\partial_{k+1}\eta)^*, \sigma_i \rangle \Big| + (k+2) \\
&\quad - \sum_{\tau \neq \eta} |\langle \partial_{k+1}(\eta), \partial_{k+1}(\tau) \rangle| \\
&\geq 0,
\end{aligned}$$

by our assumption (3.9).
Now, let T_1 the time of the first jump of X. Hence, by the definition of X,

$$\begin{aligned}
&\mathbf{P}\big(X(T_1) = X(0) + \partial_{k+1}\eta\big) \\
&\qquad = w(X_0, -\partial_{k+1}\eta) = \max\big(0, -\langle (\partial_{k+1}\eta)^*, X_0 \rangle_{\mathcal{C}^k, \mathcal{C}_k}\big) = 0.
\end{aligned} \tag{3.10}$$

Thus, $\lambda_\eta(T_1)$ must be equal to 0 or to 1.

Proceeding similarly for the simplices η for which $\lambda_\eta = -1$, we obtain that $\lambda_\tau(T_1) \in \{-1, 0, 1\}$ for any $\tau \in \mathcal{S}_k$. The proof is then concluded by induction with respect to the steps of the embedded discrete time Markov chain. □

Remark 13. *Let us comment on the assumption* (3.9)*. If the simplicial complex is a triangulation on a torus or on* $\mathbf{R}^d$*, one can pick* $\sigma_1, \dots, \sigma_{\beta_k}$ *such that for all* $i \in \{1, \dots, \beta_k\}$ *and all* $\tau \in \mathcal{S}_{k+1}$*,*

$$\Big|\sum_{i=1}^{\beta_k} \langle (\partial_{k+1}\tau)^*, \sigma_i \rangle_{\mathcal{C}^k, \mathcal{C}_k}\Big| + d(\tau)^- \le k+2.$$

This comes from the fact that any face of a simplex must either belong to exactly one other simplex or be the boundary of a hole. This is not the case if the simplicial complex is the triangulation of a torus.

Example 8. *The chain is not necessarily irreducible.*

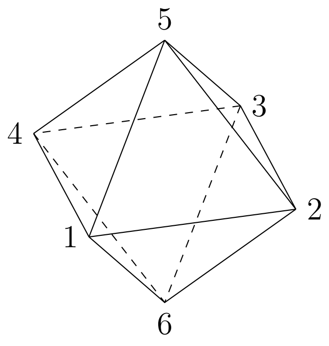


Figure 3.2: *A double tetrahedron or a simple triangulation of the sphere.*

Consider the octaedron of Figure 3.2. There are 6 *vertices,* 12 *edges and* 8 *triangles* $(T_j, j = 1, \cdots, 8)$*. Let us consider* $k = 1$*, i.e.* X *is the random walk on cycles. Getting rid of the orientation, we have at time* t *that*

$$X(t) = X(0) + \sum_{j=1}^{8} \lambda_j(t)\, T_j$$

where $\lambda_j(t) \in \{0, 1\}$*. This means that there are at most* 2^8 *chains attainable from* $X(0)$ *but for the whole tetrahedron, there are* 2^{12} *such chains.*

Definition 2. *For a k-chain τ, we denote $\mathcal{R}(\tau)$, the recurrence class of τ, consisting of all the k-chains which can be attained by X starting from τ.*

Remark 14. *Under (H), if the chain X start from a state τ belonging to a finite transient class and leading to the null chain, then the chain is absorbed by this state in a finite time.*

Remark 15. *It is well known that for random walks on undirected graphs, the stationary distribution gives to each vertex a weight proportional to its degree, i.e. to the number of edges it is adjacent to.*

We could guess that the same still holds in the present situation. The most basic situation of the simplest simplex made by one triangle abc shows that this does not hold.

If $X(0) = [ab]$, then X oscillates between $[ab]$ and $[ac] + [cb]$. The transition rate from $[ab]$ to $[ac] + [cb]$ is 1 and the transition rate from $[ac] + [cb]$ to $[ab]$ is 2, hence

$$\pi([ab]) = \frac{2}{3} \text{ and } \pi([ac] + [cb]) = \frac{1}{3}\cdot$$

This means that the stationary probability of a path is not proportional to its degree which counts the number of adjacent triangles with multiplicity.

However, when we have a very regular structure, one may classify the paths by their lengths.

Definition 3. *A group $\mathfrak{G}$ acts regularly on a simplicial complex $\mathcal{S}$ if for any ζ, ζ' two k-simplices of $\mathcal{S}$, there exists one and only one $g \in \mathfrak{G}$, which maps ζ to ζ'.*

Theorem 16. *Assume that there exists a group $\mathfrak{G}$ which acts regularly on the simplicial complex generated by $\mathcal{R}(X(0))$, the recurrence class of $X(0)$. Then, any two k-chains of the same length which belong to $\mathcal{R}(X(0))$, have the same stationary probability.*

Proof. Let ζ, ζ' two such chains. There exists a unique $g \in \mathfrak{G}$ such that $\zeta' = g\zeta$. The set of $(k+1)$-simplices adjacent to ζ' is the image by g of the set of $(k+1)$-simplices adjacent to ζ. If $X(0)=\zeta$, then $(gX(t),\, t\ge 0)$ is a Markov process starting from ζ' and for any $T>0$,

$$\int_0^T \mathbf{1}_{X(s)=\zeta}\, \mathrm{d}s = \int_0^T \mathbf{1}_{gX(s)=\zeta'}\, \mathrm{d}s,$$

hence the result by the ergodic theorem. □

Example 9. *Take $X(0)=\{1,2,3,4\}$ in the octeadron of Figure 3.2. The stationary probability is clearly stratified according to the nature of the co-chains.*

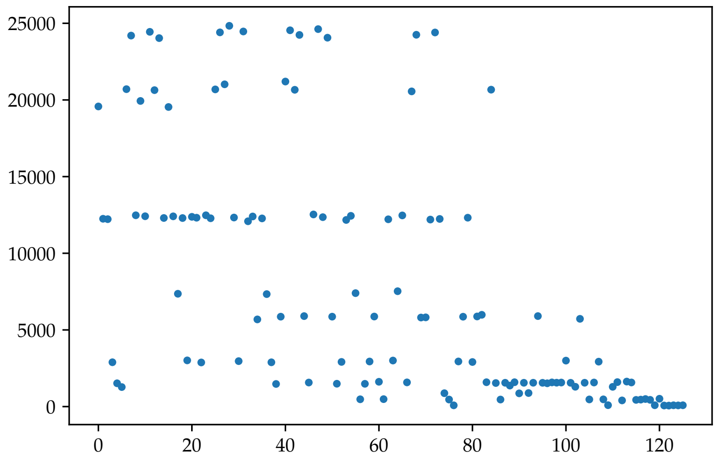


Figure 3.3: *The number of passages in each paths for the random walk on the triangulation of the sphere. The paths are numbered in order of appearance. The y-axis contains the number of passages to each path during the first 10^6 steps.*

The twelve most visited co-chains are the co-chains of length (i.e. cardinality of their support) 4 which belong to the recurrence class of the initial state, $\mathcal{R}(\{1,2,3,4\})$. They are adjacent to all triangles. Then come eight co-chains of length 3, which are adjacent to three triangles, and so on.

Example 10. *To get some insights on how the situation can be complex when the state space is infinite, let us have a look at the random walk on the triangulation of the plane from which we have removed one triangle. After a trillion of iterations, we get a graph similar to that of Figure 3.4.*

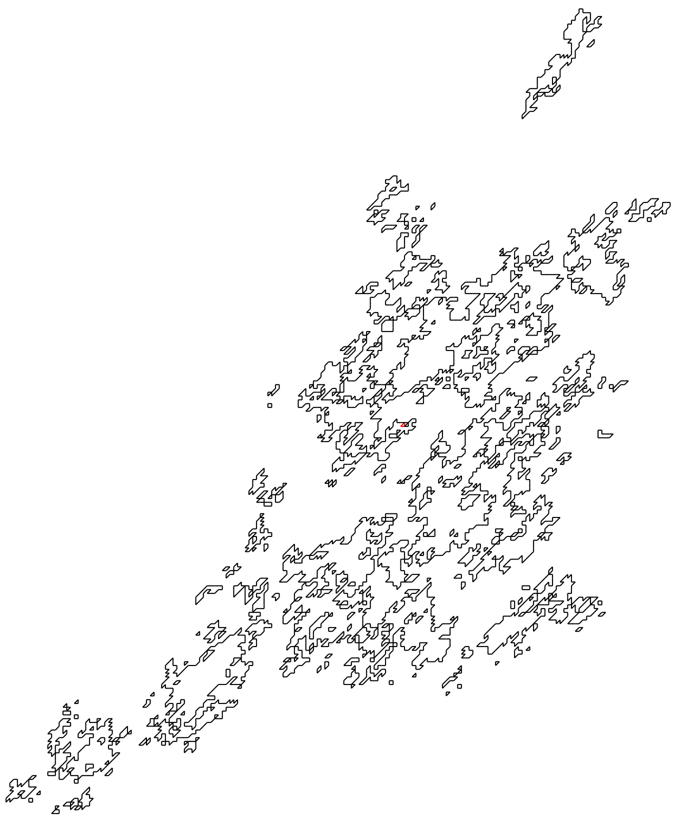


Figure 3.4: *A realization of X after 2 trillions steps. The removed triangle is in red (center of the image). Image by M. Glisse.*

The support of the process X is composed of several disconnected components, each of them may contain some holes. The isolated components are going to either die or merge with the component which contains the removed triangle. Provided that this is meaningful, if we look at the number of triangles which are inside *the chain, it can increase or decrease by* 1 *with equal probability at each step. This means that it follows the law of a symmetric random walk and is thus null recurrent. However, when the chain touches the boundary of the removed triangle, there is a drift only in the positive sense which ruins this reasoning. The simulation represented on Figure 3.5 shows that X touches the triangle very often even when it is itself large.*

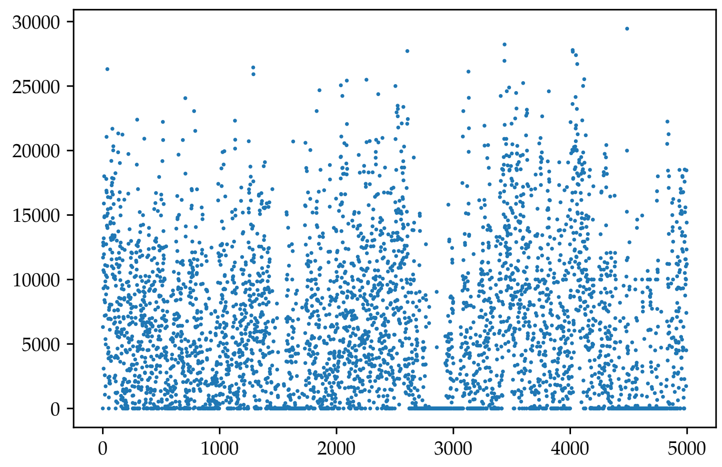


Figure 3.5: *The number of times X touches the triangle each packets of ten thousands steps. Simulation by M. Glisse.*

3.3 Convergence of random walk

When dealing with random walks, it is natural to investigate their continuous diffusive limits. For this, we need to embed our graph into another space and consider geometric random graphs. [50]

We denote by $\mathbb{T}_2 := \mathbf{R}^2/\mathbf{Z}^2$ the flat torus, which we embed into $\mathbf{R}^2$ as the square $[0,1] \times [0,1]$ where the opposite edges are identified. Let $\epsilon_n = 1/2n$ and consider

$$V_n = \{(2k\epsilon_n, 2l\epsilon_n), 0 \le k, l \le n\} \bigcup \{((2k+1)\epsilon_n, (2l+1)\epsilon_n), 0 \le k, l \le n-1\},$$

the set of vertices of the regular triangulation of mesh $2\epsilon_n$, see Figure 3.6.

First consider the random walk on V_n. We want to show that the generator of this random walk converges to the Laplacian on the torus in the sense of [20, Theorem 6.1] which says the following. For $n \ge 1$, B_n, in addition to B, is a Banach space and π_n is a bounded linear transformation from L to L_n. We suppose that $\sup_n \|\pi_n\| < \infty$ and we write $f_n \to f$ if $f \in L_n$ and

$$\lim_{n\to\infty} \|f_n - \pi_n f\|_{B_n} = 0. \tag{3.11}$$

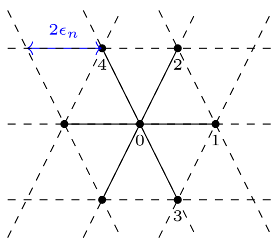


Figure 3.6: *Regular triangulation of the flat torus.* $0 := (0,0)$, $1 := (2\epsilon_n, 0)$, $2 := (\epsilon_n, \sqrt{3}\epsilon_n)$, $3 := (\epsilon_n, -\sqrt{3}\epsilon_n)$, $4 := (-\epsilon_n, \sqrt{3}\,\epsilon_n)$

Theorem 17 (Ethier-Kurtz)**.** *For $n \geq 1$, let T_n and T be strongly continuous contraction semigroups on B_n and B with generators $\hat{A}_n$ and A. Let D be a core for A. Then the following two properties are equivalent:*

1. *For each $f \in B$,*
$$T_n(t)\pi_n f \xrightarrow{n\to\infty} T(t)f,$$
uniformly on bounded intervals,

2. *For each $f \in D$, there exists $f \in \mathcal{D}(A_n)$ for each $n \geq 1$ such that*
$$f_n \xrightarrow{n\to\infty} f \text{ and } A_n f_n \xrightarrow{n\to\infty} Af.$$

In the present situation, we set B_n to be set of (bounded) functions from V_n into $\mathbf{R}$ and B to be the set of continuous functions from $\mathbb{T}_2$ into $\mathbf{R}$, all equipped with the sup-norm. The map π_n is the restriction to V_n. By construction,

$$\hat{A}_n f_n(x) = \sum_{a\in\{-1,1\}} \Big(f_n(x + (2a\epsilon_n, 0)) - f_n(x)\Big) + \sum_{a,b\in\{-1,1\}} \Big(f_n(x + (a\,\epsilon_n, b\sqrt{3}\,\epsilon_n)) - f_n(x)\Big). \quad (3.12)$$

Let D be the set of twice continuously differentiable functions from $\mathbb{T}_2$ into $\mathbf{R}$. Since V_n progressively fills in $\mathbb{T}_2$, for any $f \in D$, $f_n = \pi_n f = f_{|V_n}$ belongs to L_n and Eqn. (3.11) is trivially satisfied.

Theorem 18. *With notations as above, set $A_n = \epsilon_n^{-2}\hat{A}_n$. We have*

$$A_n f_n \xrightarrow{n\to\infty} Af := 6\,\frac{\partial^2 f}{\partial x_1^2} + 6\,\frac{\partial^2 f}{\partial x_2^2}.$$

This means that the normalized random walk behaves asymptotically as the process $\sqrt{6}(W_1,\ W_2)$ where W_1 and W_2 are two independent Brownian motion on the torus.

Proof. By symmetry, the first order terms of the Taylor expansion of the right-hand-side of (3.12) vanish. Moreover, for the same reason, the crossed derivatives of the second order term also disappear. The computations in detail are presented in Appendix 1.1.

Finally, we get,

$$A_n f_n(x) = 6\epsilon_n^2 \frac{\partial^2 f_n}{\partial x_1^2}(x) + 6\epsilon_n^2 \frac{\partial^2 f_n}{\partial x_2^2}(x) + o(\epsilon_n^2).$$

Hence the result. □

We now turn to the analogous theorem for the random walk on $\mathcal{C}^1(V_n)$.

Definition 4. *Let Φ the space of continuous 1-differential form on the torus. Such a differential form ϕ can be written as*

$$\phi = \phi^1\ dx_1 + \phi^2\ dx_2,$$

where ϕ^1 and ϕ^2 are twice differentiable functions on the torus, i.e. they can be viewed as the restriction over $[0,1]^2$ of twice differentiable, $(1,1)$-periodic functions:

$$\phi(x_1 + l_1, x_2 + l_2) = \phi(x_1, x_2)$$

for any pair of integers (l_1, l_2). We set

$$\|\phi\|_\Phi = \|\phi^1\|_\infty + \|\phi^2\|_\infty.$$

Its topological dual is the set of currents, denoted by $\mathfrak{C}_1$. It inherits the Banach norm:

$$\|p\|_{\mathfrak{C}_1} = \sup_{\phi\in\Phi} \frac{|\langle p, \phi\rangle_{\Psi,\Phi}|}{\|\phi\|_\Phi}.$$

We denote by $\mathfrak{P}$, the set of paths; i.e. the piecewise differentiable maps from $[0,1]$ into $\mathbb{T}_2$. For $\phi \in \Phi$, the curvilinear integral of ϕ along an element $p \in \mathfrak{P}$ is a linear map and

$$\left| \int_p \phi \right| \le \|\phi\|_\Phi \ \mathit{length}(p),$$

hence $\mathfrak{P}$ can be viewed as a subspace of $\mathfrak{C}_1$.

We denote by $\Phi^{(k)}$, the set of k-times differentiable 1-forms on $\mathbb{T}_2$. Remark that for any $k \geq 1$, $\Phi^{(k)}$ is dense in Φ.

Definition 5. *For $\phi = \phi^1\, dx_1 + \phi^2\, dx_2 \in \Phi^{(1)}$, its exterior derivative denoted by $d\phi$ is the function (or 0-form):*

$$d\phi(x) = \frac{\partial \phi^2}{\partial x_1}(x) - \frac{\partial \phi^1}{\partial x_2}(x).$$

The Hodge transform of forms is the linear transformation defined by its action on a basis of differential forms:

$$*1 = \ dx_1 \wedge \ dx_2, \ * \ dx_1 = \ dx_2, \ * \ dx_2 = - \ dx_1, \ *(\ dx_1 \wedge \ dx_2) = 1.$$

For $\phi \in \Phi^{(2)}$, the Laplace-Beltrami operator is then defined by

$$\mathfrak{L} = \mathfrak{L}^{\uparrow} + \mathfrak{L}^{\downarrow} \text{ where } \mathfrak{L}^{\uparrow} = *d*d \text{ and } \mathfrak{L}^{\downarrow} = d*d*.$$

A classical computation shows that

$$\mathfrak{L}^{\uparrow}\Big(\phi^1\, dx_1 + \phi^2\, dx_2\Big) = \Big(\phi^1_{22} - \phi^2_{12}\Big)\, dx_1 + \Big(\phi^2_{11} - \phi^1_{12}\Big)\, dx_2 \tag{3.13}$$

where f_i is a shortcut for the partial derivative of f with respect to the variable x_i.

Definition 6. *On Φ, we define the scalar product:*

$$\langle \phi^1 \ dx_1 + \phi^2 \ dx_2, \ \psi^1 \ dx_1 + \psi^2 \ dx_2 \rangle = \sum_{j=1,2} \int_{\mathbb{T}_2} \phi^j(x_1, x_2)\psi^j(x_1, x_2) \ dx_1 \ dx_2.$$

Lemma 19. *The map $\mathfrak{L}^{\uparrow}$, whose domain is $\Phi^{(2)}$, generates a strongly continuous contraction semi-group on Φ.*

Proof. According to [20], we have to prove that $\mathfrak{L}^{\uparrow}$ is closable, dissipative and that there exists $\lambda > 0$ such that $\lambda \operatorname{Id} - \mathfrak{L}^{\uparrow}$ is one-to-one.

According to (3.13),

$$\langle \mathfrak{L}^{\uparrow}\phi, \phi \rangle = \int_{\mathbb{T}_2} \frac{\partial}{\partial x_2}\Big(\phi^1_2 - \phi^2_1\Big)\ \phi^1 \ \mathrm{d}x_1 \ \mathrm{d}x_2 - \int_{\mathbb{T}_2} \frac{\partial}{\partial x_1}\Big(\phi^1_2 - \phi^2_1\Big)\ \phi^2 \ \mathrm{d}x_1 \ \mathrm{d}x_2.$$

By integration by parts, taking into account the periodicity of ϕ^1 and ϕ^2, we get

$$\langle \mathfrak{L}^\uparrow \phi,\, \phi\rangle = -\int_{\mathbb{T}_2} \left(\phi_2^1 - \phi_1^2\right)\phi_2^1 \,\mathrm{d}x_1\,\mathrm{d}x_2 + \int_{\mathbb{T}_2} \left(\phi_2^1 - \phi_1^2\right)\,\phi_1^2 \,\mathrm{d}x_1\,\mathrm{d}x_2$$
$$= -\int_{\mathbb{T}_2} \left(\phi_2^1 - \phi_1^2\right)^2 \,\mathrm{d}x_1\,\mathrm{d}x_2.$$

This means that $\mathfrak{L}^\uparrow$ is symmetric and negative (hence dissipative). Integrating by parts a second time yields, for any $\psi \in \Phi^{(2)}$,

$$\langle \mathfrak{L}^\uparrow \phi,\, \psi\rangle = \langle \phi,\, \mathfrak{L}^\uparrow \psi\rangle.$$

Hence, if $\phi_n \to 0$ and $\mathfrak{L}^\uparrow \phi_n \to \eta$, we get $\langle \eta,\, \psi\rangle = 0$ for any $\psi \in \Phi^{(2)}$. By density, this entails $\eta = 0$. Hence, $\mathfrak{L}^\uparrow$ is closable. We still denote by $\mathfrak{L}^\uparrow$ its extension, whose domain $\mathrm{dom}(\mathfrak{L}^\uparrow)$ contains at least $\Phi^{(2)}$.

Consider the basis of $L^2(\mathbb{T}_2, \mathbf{C})$ given by

$$e_{n,m}(x_1, x_2) = e^{2i\pi n x_1} e^{2i\pi m x_2},\ n, m \in \mathbf{Z}.$$

For $i = 1, 2$, we have in $L^2(\mathbb{T}_2, \mathbf{C})$,

$$\phi^i = \sum_{n,m\in\mathbf{Z}} c^i_{n,m}\, e_{n,m}.$$

Furthermore,

$$\phi^1_{22} - \phi^2_{12} = -4\pi^2 \sum_{n,m\in\mathbf{Z}} (c^1_{n,m} m^2 - c^2_{n,m} mn)\, e_{n,m},$$
$$\phi^2_{11} - \phi^1_{12} = -4\pi^2 \sum_{n,m\in\mathbf{Z}} (c^2_{n,m} n^2 - c^1_{n,m} mn)\, e_{n,m}.$$

Thus solving $\mathfrak{L}^\uparrow \phi = -4\pi^2 \lambda \phi$ amounts to find the $c^i_{n,m}$'s such that

$$c^1_{n,m}(m^2 - \lambda) - c^2_{n,m} mn = 0,$$
$$-c^1_{n,m} mn + c^2_{n,m}(n^2 - \lambda) = 0.$$

For λ negative irrational, this system admits the null form as unique solution, hence for such a λ, $\mathfrak{L}^\uparrow - 4\pi^2\lambda\,\mathrm{Id}$ is one-to-one and the third condition is satisfied. □

See Figure 3.6 for the notations in the following theorems.

Lemma 20. *Let $H : [0,1]^2 \to \mathbf{R}$ be C^2. Then, as ϵ_n goes to 0,*

$$\int_{[021]\circlearrowleft} H(x_1,x_2)\ dx_1\ dx_2 + \int_{[031]\circlearrowleft} H(x_1,x_2)\ dx_1\ dx_2 = -\epsilon_n^2 \int_0^{2\epsilon_n} H_2(x_1,0)\ dx_1 + O(\epsilon_n^4). \quad (3.14)$$

Proof. Using Fubini's theorem and taking into account the orientations,

$$M_1 := \int_{[021]\circlearrowleft} H(x_1,x_2)\ \mathrm{d}x_1\ \mathrm{d}x_2 = -\int_0^{\epsilon_n}\int_0^{x_1\sqrt{3}} H(x_1,x_2)\ \mathrm{d}x_1\ \mathrm{d}x_2 - \int_{\epsilon_n}^{2\epsilon_n}\int_0^{\sqrt{3}(2\epsilon_n - x_1)} H(x_1,x_2)\ \mathrm{d}x_1\ \mathrm{d}x_2,$$

and

$$M_2 := \int_{[031]\circlearrowleft} H(x_1,x_2)\ \mathrm{d}x_1\ \mathrm{d}x_2 = \int_0^{\epsilon_n}\int_{-x_1\sqrt{3}}^{0} H(x_1,x_2)\ \mathrm{d}x_1\ \mathrm{d}x_2 + \int_{\epsilon_n}^{2\epsilon_n}\int_{-\sqrt{3}(2\epsilon_n - x_1)}^{0} H(x_1,x_2)\ \mathrm{d}x_1\ \mathrm{d}x_2.$$

A Taylor expansion gives

$$H(x_1,x_2) = H(x_1,0) + x_2\, H_2(x_1,0) + {x_2}^2\, r(x_1,x_2)$$

with $\sup_{x_1,x_2\in[0,1]^2} r(x_1,x_2) < \infty$. By symmetry, the term with $H(x_1,0)$ disappears and we get

$$M_1+M_2 = -3\int_0^{\epsilon_n} H_2(x_1,0){x_1}^2\ \mathrm{d}x_1 - 3\int_{\epsilon_n}^{2\epsilon_n} H_2(x_1,0)(2\epsilon_n - x_1)^2\ \mathrm{d}x_1 + R_n,$$

where the remainder R_n is bounded (up to an irrelevant constant) by the integral of ${x_2}^2$ over the union of the two triangles [021] and [031]. The detailed computations are presented in Appendix 1.2.

This yields:

$$R_n = O(\epsilon_n^4).$$

With another Taylor expansion, we get

$$M_1 + M_2 = -2H_2(0,0)\epsilon_n^3 + O(\epsilon_n^4).$$

Moreover,

$$\int_0^{2\epsilon_n} H_2(x_1,0)\ \mathrm{d}x_1 = 2H_2(0,0)\epsilon_n + O(\epsilon_n^2).$$

The proof is thus complete. □

Lemma 21. *For a twice differentiable* 1*-form* ϕ*,*

$$\sup_n \sup_{v\in\mathcal{S}_1(V_n)} \epsilon_n^{-2} \left|\epsilon_n^{-2} A_n(\int_v \phi) - \int_v \mathfrak{L}^{\uparrow}\phi\right| < \infty. \tag{3.15}$$

Proof. Let v be an element of $\mathcal{S}_1(V_n)$ and ϕ a $\mathcal{C}^2$ 1-form. The edge v is adjacent to two triangles, which we denote by τ_v^+ and τ_v^-. Both of them are oriented so that they contain $-v$. Thus,

$$\begin{aligned}
A_n\Big(\int_v \phi\Big) &= \Big(\int_{\tau_v^+ + v}\phi - \int_v \phi\Big) + \Big(\int_{\tau_v^- + v}\phi - \int_v\phi\Big)\\
&= \Big(\int_{\tau_v^+ + v}\phi + \int_{-v}\phi\Big) + \Big(\int_{\tau_v^- + v}\phi + \int_{-v}\phi\Big)\\
&= \int_{\tau_v^+}\phi + \int_{\tau_v^-}\phi\\
&= \int_{\partial_2^* v} d\phi, \qquad (3.16)
\end{aligned}$$

according to the Stokes formula. First, we consider the case of the horizontal edge [01]. According to (3.16), we have

$$A_n(\int_{[01]}\phi) = \int_{[021]\circlearrowleft} d\phi + \int_{[031]\circlearrowleft} d\phi.$$

Apply Lemma 20 to $\partial\phi_j/\partial x_i$ for $i,j=1,2$ to get

$$\epsilon_n^{-2}A_n(\int_{[01]}\phi) - \Big(\int_0^{2\epsilon_n} \frac{\partial^2\phi_1}{\partial {x_2}^2}(x_1,0)\ \mathrm{d}x_1 - \int_0^{2\epsilon_n}\frac{\partial^2\phi_2}{\partial x_1\partial x_2}(x_1,0)\ \mathrm{d}x_1\Big) = O(\epsilon_n^2).$$

In view of (3.13), this means that

$$\epsilon_n^{-2}A_n(\int_{[01]}\phi) = \int_{[01]}\mathfrak{L}^{\uparrow}\phi + O(\epsilon_n^2). \tag{3.17}$$

The same procedure can be applied to any horizontal edge. For an oblique edge, we also benefit from the symmetries present in the triangulation. Consider that $v=[02]$. According to (3.16) (see Figure 3.6 for the location of point 4),

$$A_n\Big(\int_v\phi\Big) = \int_{[042]\circlearrowleft} d\phi + \int_{[012]\circlearrowleft} d\phi.$$

For any function H,

$$\int_{[042]\circlearrowleft} H(x_1, x_2)\ \mathrm{d}x_1\ \mathrm{d}x_2 = \int_{[021]\circlearrowleft} H \circ \Theta_{2\pi/3}(x_1, x_2)\ \mathrm{d}x_1\ \mathrm{d}x_2$$

where $\Theta_{2\pi/3}$ is the affine rotation of angle $2\pi/3$ and center 0. Hence,

$$\frac{\partial(H \circ \Theta_{2\pi/3})}{\partial x_2}(x_1, 0) = -\frac{\sqrt{3}}{2} H_1(\frac{x_1}{2}, \frac{\sqrt{3}x_1}{2}) + \frac{1}{2} H_2(\frac{x_1}{2}, \frac{\sqrt{3}x_1}{2}).$$

The path $x_1 \mapsto (x_1/2, x_1\sqrt{3}/2)$ is a parametrization of [02]. It follows that for $v = [02]$

$$\epsilon_n^{-2} A_n(\int_v \phi) = \int_v \mathfrak{L}^{\uparrow}\phi + O(\epsilon_n^2), \tag{3.18}$$

where $O(\epsilon_n^2)$ is the same function as in (3.17).

Since every element of $\mathcal{S}_1(V_n)$ can be attained by a combination of translation and rotations of [01], we see that (3.18) holds for any $v \in \mathcal{S}_1(V_n)$ with the same error function, so that (3.15) holds true. □

Since $\mathfrak{C}_1$ is the dual of Φ, we can define the adjoint of $\mathfrak{L}^{\uparrow}$ as follows.

Definition 7. *Let*

$$\mathrm{dom}\Big((\mathfrak{L}^{\uparrow})^*\Big) = \left\{p \in \mathfrak{C}_1, \exists c_p, |\langle p, \mathfrak{L}^{\uparrow}\phi\rangle_{\mathfrak{C}_1,\Phi}| \le c_p\|\phi\|_{\Phi}, \forall\phi \in \mathrm{dom}(\mathfrak{L}^{\uparrow})\right\}.$$

Note that $\mathfrak{P} \subset \mathrm{dom}\Big((\mathfrak{L}^{\uparrow})^*\Big)$. *Then,* $(\mathfrak{L}^{\uparrow})^*$ *is defined by the relation:*

$$\langle p,\ \mathfrak{L}^{\uparrow}\phi\rangle_{\mathfrak{C}_1,\Phi} = \langle(\mathfrak{L}^{\uparrow})^* p,\ \phi\rangle_{\mathfrak{C}_1,\Phi}.$$

It also generates a strongly continuous semi-group of contractions on $\mathfrak{C}_1$ and Lemma 20 means

Corollary 22. *The sequence of generators* $(\epsilon_n^{-2}A_n,\ n \ge 1)$ *tends to* $(\mathfrak{L}^{\uparrow})^*$ *in the sense of Theorem 17, and so do the corresponding semi-groups.*

Chapter 4

Random walk based hole detection

It is natural to wonder whether some k-simplices are more likely to belong to the Markov process X than others. In this chapter, we introduce an algorithm to make simulations carried for the cycle-valued random walk ($k = 1$) on a simplicial complex with holes, and it tends to show that the random walk is more likely to go through edges on the boundaries of the holes than other places. Therefore, we can apply our algorithm on simplicial complexes to detect the location of holes.

4.1 Introduction

As we have discussed before, a combinatorial Laplacian can be decomposed into a sum of two operators called up-Laplacian and down-Laplacian which each correspond to a different random walk. We can compute the matrix of the combinatorial Laplacian of order k as

$$L_k = L_k^{\uparrow} + L_k^{\downarrow} = \left(D_k - A_k^{\uparrow}\right) + \left((k+1)\,\mathrm{Id} + A_k^{\downarrow}\right).$$

While $L_k^{\uparrow}$ has the structure of the generator of a Markov process, $L_k^{\downarrow}$ does not correspond to the generator of a Markov process, hence it is of great complexity to define a Markov process whose generator would be $L_k^{\downarrow}$.

If $X(0)$ is simple and a cycle, we have seen that $L_k^{\uparrow} = L_k$, which makes it possible to make simulations of the random walk whose generator is the combinatorial Laplacian L_k. Therefore, we focus on the random walk corresponding to $L_k^{\uparrow}$ on cycles.

In this chapter, we propose a heuristic algorithm – a simulated annealing algorithm – to see for a given simplicial complex, whether the random walk tends to visit some specific simplices more often than others. For a given Rips complex with two holes, our algorithm finds the minimal cycles around these two holes and it appears that the random walk is more likely to visit the edges on the boundaries of the holes.

The rest of this chapter is organized as follows. Section 4.2 introduces the data structure to represent the simplicial complexes and the definition of random walk. Section 4.3 describes our algorithm. In Section 4.4, we discuss the complexity of our algorithm. In Section 4.5, we focus on the convergence rate of the algorithm. The simulations and results are described in Section 4.6. Finally, we make a conclusion in Section 4.7.

4.2 Definitions and model

4.2.1 Simplex tree

For simplicial complexes, for example the Čech complex and the Rips complex, their grows very rapidly with the dimension of the data set, and their use in real applications has been quite limited. Boissonnat and Maria [8] have introduced a data structure to represent the simplicial complexes and make it easier to implement some basic operations. The data structure is the so-called simplex tree.

For a simplicial complex $\mathcal{C}$ of dimension k, V its vertex set, the vertices are labeled from 1 to $|V|$ and ordered accordingly. We can associate each simplex of $\mathcal{C}$ to a word on the alphabet $1 \cdots |V|$. Let a simplex $\sigma = \{v_{l_0}, \cdots, v_{l_j}\}$, where $v_{l_i} \in V$, $l_i \in \{1, \dots, |V|\}$ and $l_0 < \cdots < l_j$. The simplex σ is represented by the word $[\sigma] = [l_0, \cdots, l_j]$. The last label of the word representation of a simplex σ will be called the last label of σ and denoted by $last(\sigma)$.

Definition 8 (Simplex tree). *The simplicial complex $\mathcal{C}$ can be defined as a* *collection of words on an alphabet of size $|V|$. To compactly represent the set of simplices of $\mathcal{C}$, we store the corresponding words in a tree $\mathcal{T}$ satisfying the following properties:*

1. *The nodes of the simplex tree $\mathcal{T}$ are in bijection with the simplices (of all dimensions) of the complex. The root is associated to the empty face.*

2. *Each node of the tree $\mathcal{T}$, except the root, stores the label of a vertex. Specifically, a node N associated to simplex $\sigma \neq \emptyset$ stores the label of vertex $last(\sigma)$.*

3. *The vertices whose labels are encountered along a path from the root to a node N, associated to a simplex σ, are the vertices of σ. The labels are sorted by increasing order along such a path, and each label appears exactly once.*

We call this data structure the Simplex Tree of $\mathcal{C}$. It may be seen as a digital tree [3] on the words representing the simplices of the complex. The depth of the root is 0 and the depth of a node is equal to the dimension of the simplex it represents plus one.

We take an example to clarify the construction of simplex tree. For a Rips complex $\mathcal{C}$ with dimension 2 whose vertex set is denoted by $\{1, 2, 3, 4, 5, 6\}$ (see Figure 4.1), first, we start from an empty tree and insert all the vertices as the children of the root. Then, for each vertex, we start from the root and find all the edges containing successively the corresponding vertex. That is, for vertex 1, the edges are $[1, 2]$ and $[1, 6]$, and for vertex 2, the edge is $[2, 3]$. At last, for an edge $[l_0, l_1]$, $l_0 < l_1$, we append l_1 to the node of l_0 and repeat this procedure for all vertices. That is, for node of vertex 1, we add nodes of vertex 2 and 6. Similarly, for the edge $[3, 4]$, we have the triangle $[3, 4, 6]$ so we append node of vertex 6 to the path $[3, 4]$, which makes 6 be a leaf node of our tree.

Nodes which share the same parent will be called sibling nodes. We attach to each set of sibling nodes a pointer to their parent so that we can access a parent in constant time. Thus, it is an efficient way for us to store the

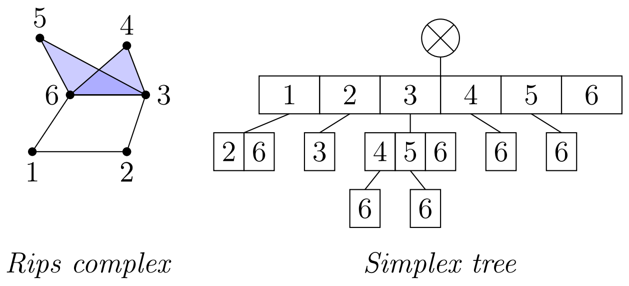


Figure 4.1: *Rips complex and its corresponding simplex tree*

simplicial complexes as simplex trees and to access the cofaces of a given simplicial complex in every step of our random walk.

In this paper, we use GUDHI library in which the general simplicial complexes are represented by their simplex trees.

4.2.2 Determination of initial state of random walk

For a given simplicial complex with dimension k, first, we need to find the initial state of our random walk if we do not know in advance. As we have discussed, for any $t > 0$, X the Markov process, $X(t)$ and $X(0)$ are always in the same homology group. To detect all the holes by random walk, we need to find a cycle which is in the homology group of these holes. Moreover, if $X(0)$ is simple and a cycle, we have seen that $L_k^{\uparrow} = L_k$ so that

$$\frac{d}{dt}w(t) = -L_k^{\uparrow}w(t)$$

is exactly the differential equation investigated by Muhammad and Egerstedt in [35]. Thus, if $X(0)$ is simple and a cycle, by Hodge decomposition we have

$$\ker L_k \simeq H_k,$$

in order to compute the homology groups of a simplicial complex, it is enough to study the null space of the matrix L_k. The eigenvectors of L_k corresponding to the zero eigenvalues are the representative cycles of a particular homology class. Therefore, we can look at each eigenvector of L_k corresponding to the zero eigenvalue at a time and set it to be our initial state.

Due to the procedure in how we determine the initial state, we know that weights of the initial state are not in $\{-1, 0, 1\}$ any more, that is to say, $X(t)$ is not simple. When adding the boundary of a triangle to the chain-valued random walk, we should be concerned about the weight of the boundary. Therefore, it is crucial to adjust the weight of transition kernel of our simple chain-valued random walk to the so-called integer weighted random walk.

Definition 9 (Transition kernel of integer weighted random walk)**.** *For a k-chain $\tau = \sum_{s \in \mathcal{S}_k^+} \lambda_s(\tau)s$, define the minimal weight of τ by:*

$$\lambda_{min}(\tau) = \min_{s \in \mathcal{S}_k^+} \lambda_s(\tau).$$

For a k-chain $\tau \neq 0$ an oriented $(k+1)$-simplex $\eta \in \mathcal{S}_{k+1}$, we define the number of common faces between τ and $\partial_{k+1}\eta$ by:

$$w\,(\tau,\ \partial_{k+1}\eta) = \langle (\partial_{k+1}\eta)^*,\ \tau\rangle^+_{\mathcal{C}^k, \mathcal{C}_k}$$

where $x^+ = \max(x, 0)$ for $x \in \mathbf{R}$, so the weight of common faces between τ and $\partial_{k+1}\eta$ is $\langle (\lambda_{min}(\tau)\partial_{k+1}\eta)^*,\ \tau\rangle^+_{\mathcal{C}^k, \mathcal{C}_k}$.

For another chain τ', we say that

$$\tau \sim \tau' \iff \exists \eta \in \mathcal{S}_{k+1}, w\,(\tau,\ \partial_{k+1}\eta) > 0 \text{ and } \tau' = \tau - \lambda_{min}(\tau)\partial_{k+1}\eta. \quad (4.1)$$

Finally, let

$$K(\tau,\, \tau') = \begin{cases} 1 & \text{if } \tau = \tau' = 0 \\ w\,(\tau,\ \tau - \tau') & \text{if } \tau \sim \tau' \\ 0 & \text{otherwise.} \end{cases} \quad (4.2)$$

Example 11. *In Figure 4.2, we see how the boundary of triangle is added to a 1-chain $\tau = 10[v_1v_2] + 13[v_2v_3]$.*

4.3 Hole detection algorithm

Consider a Rips complex, we apply our integer weighted random walk with an initial state (which is a cycle) and minimize the length of $X(t)$. Since the initial state is in the same homology group with the holes and it will not

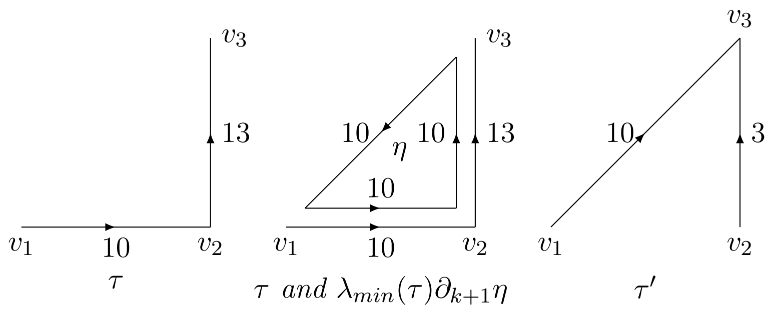


Figure 4.2: *One step of integer weighted random walk from τ to τ'*

leave the homology group, it means that our chain will always include the holes and always be cycles. In order to see where the holes locate, we can minimize the length of our chain.

We propose the simulated annealing algorithm to minimize the length of our chain. Simulated annealing (SA) is a heuristic method for approximating the global optimum of a given function U, which is called energy function. The name and inspiration come from annealing process in metal work, a technique involving heating and controlled cooling of a material to increase the size of its crystals and reduce their defects. See [4] for more details.

With the initial state computed from the null space of L_k, our SA algorithm finds the global minimal length of our chain by following the cooling schedule $T_m = T_0\alpha^m$, where T_0 is the initial temperature and α is the cooling factor subject to $0 < \alpha < 1$. At each temperature, we apply our random walk whose generator is $L_k^{\uparrow}$ on τ and we obtain τ_1, then we compute the length of τ and τ_1 as $U(\tau)$ and $U(\tau_1)$. The difference of power consumption is calculated by $\Delta U = U(\tau_1) - U(\tau)$. If $\Delta U < 0$, the length of our chain decreases, then we accept this step. If the length increases, we call it an uphill move. If $\Delta U > 0$, the uphill move is accepted with probability $\exp(-\Delta U/T)$. Thanks to uphill moves, the process can jump out from a local minimum to search for the global minimum. At each temperature T, this process is repeated L times to make the temperature decrease slowly. To set the initial parameters, we need to make sure that the initial temperature T_0 is large enough to make the probability of an uphill move at initial state be close to one. The number of schedule steps M is chosen large enough to make the probability of accepting an uphill move to be near zero at the final temperature $T_0\alpha^M$. The final configuration of chain τ is our optimal chain with minimal length. Our algorithm is written as Algorithm 1.

Algorithm 1 Simulated annealing algorithm

```
    Input: $\tau = \sum_{s \in \mathcal{S}_k^+} \lambda_s(\tau)s$ the initial state
    Output: $\tau_{opt}$ the chain with minimal length
  procedure SA($\tau$)
      for $m = 1 \to M$ do
          $T = T_0\alpha^m$
          for $l = 1 \to L$ do
              compute $\tau_1 = RW(\tau)$
              compute lengths of $\tau$ and $\tau_1$ as $U(\tau)$ and $U(\tau_1)$
              compute $\Delta U = U(\tau_1) - U(\tau)$
              if $\Delta U < 0$ then
                  $P = 1$
              else
                  $P = \exp\left(-\frac{\Delta U}{T}\right)$
              end if
              $\tau = \tau_1$ with probability $P$
          end for
      end for
      return $\tau$
  end procedure
```

For the random walk part, the algorithm is summarized in Algorithm 2. We generate a simplicial complex and build the simplex tree $\mathcal{T}$, and we initiate with a chain τ and the simplex tree $\mathcal{T}$. Firstly, we find all the cofaces of s in chain $\tau = \sum_{s \in \mathcal{S}_k^+} \lambda_s(\tau)s$, and then we choose one of the cofaces uniformly, denoted by η. Then, we compute the boundaries of η, and by (3.3), we add our chain τ with $\min_{s \in \mathcal{S}_k^+} \lambda_s(\tau)\partial_{k+1}\eta$. To avoid the case where our chain disappears after our random walk, we repeat this procedure if the chain becomes null. Therefore, we get τ_1 after one step of our integer weighted random walk.

Algorithm 2 Integer weighted random walk

```
    Input: $\tau = \sum_{s \in \mathcal{S}_k^+} \lambda_s(\tau) s$ the chain, $\mathcal{T}$ the simplex tree
    Output: $\tau_1$ the chain after one step
  function RW($\tau$, $\mathcal{T}$)
      while $\tau_1$ is null do
          find all the cofaces of $s$ in $\mathcal{T}$
          choose one of the cofaces $\eta$ and compute the boundaries of $\eta$ as $\partial_{k+1}\eta$
          $\tau_1 = \tau + \min_{s \in \mathcal{S}_k^+} \lambda_s(\tau) \partial_{k+1}\eta$
      end while
  end function
```

4.4 Complexity

The random walk based hole detection algorithm requires two main computations: random walk on simplicial complexes and simulated annealing.

For the random walk on simplicial complexes, if the Rips Complex is built on dimension $d = 2$, what we want is to search and locate all cofaces, say triangles, of a given edge. For a given edge e represented by the word $[l_0 l_1]$, the cofaces of e are the simplices of $\mathcal{C}$ which are represented by words of the form $[\star l_0 \star l_1 \star]$, where $\star$ represents an arbitrary word, possibly empty. To locate all the words of the form $[\star l_0 \star l_1 \star]$ in the simplex tree, we first find all the words of the form $[\star l_0 \star l_1]$. Using the lists $L_i(l_1)(i > 1)$, we find all the nodes at depth at least 2 which contain label l_1. For each such node N_{l_1}, we traverse the tree upwards from N_{l_1}, looking for a word of the form $[\star l_0 \star l_1]$. After that, we look for the nodes in the subtree rooted at N_{l_1}, which are represented by words of the form $[\star l_0 \star l_1 \star]$.

We define $\mathcal{T}_l^{>j}$ to be the number of nodes of $\mathcal{T}$ at depth strictly greater than j that store label l. The complexity of searching and locating the cofaces of an edge e depends on $\mathcal{T}_{last(e)}^{>1}$ of nodes and the dimension d of $\mathcal{C}$. To traverse the tree upwards, for each node in the number of $\mathcal{T}_{last(e)}^{>1}$ nodes it takes $\mathcal{O}(d)$ time. For each e, $\mathcal{T}_{last(e)}^{>1}$ is at most $(d-1)|V|$. Therefore, the complexity of searching and locating cofaces is $\mathcal{O}(d^2|V|)$.

The length of our path is at most the number of edges of the simplicial complex $\mathcal{C}$, which is at most $|V|(|V|-1)/2$. Since we do not allow neither our random walk selects an edge which has no neighboring triangles nor our random walk turns to be null after any step, the complexity of random walk on simplicial complexes is $\mathcal{O}(d^2|V|^5)$.

For the simulated annealing part, the optimization process is repeated during the length of each temperature in the cooling schedule. The length of each temperature is L loops. The temperature is updated by the cooling schedule, whose size is K steps. So, there are totally KL loops in this algorithm. In each loop, the random walk is implemented on our path on simplicial complexes. In this algorithm, the Rips complex is still built on dimension $d = 2$. Thus, the complexity of the random walk based hole detection algorithm is $\mathcal{O}(KL|V|^5)$.

4.5 Convergence rate

In this section, for simulated annealing algorithms, we introduce the definition of hierarchical decomposition of the configuration spaces, so as to study the behavior of these algorithms. Trouvé [46] gave the construction of the cycle decomposition in the simulated annealing framework, but the definition of cycle in that paper is different from us. To avoid ambiguity, we refer as "circulation" in our paper.

As we have known, for a continuous time Markov process X, E its configuration space and q its irreducible Markov kernel, if $q(i,j) > 0$, we can say that j can be joined from i. Here we introduce a specific situation where we can say that j can be joined from i at level h.

Definition 10. *Given a simplicial complex $\mathcal{C}$, let $h \in \mathbf{R}$ and $i, j \in E$, we say that j can be joined from i at level h if there exists a finite family $(i_l)_{0\leq l\leq p}$ of elements of $\mathcal{S}_k$ such that*

1. *$i = i_0$ and $j = i_p$,*
2. *$q(i_l, i_{l+1}) > 0$ for all $0 \leq l \leq p$,*
3. *and $U(i_l) \leq h$ for all $0 \leq l \leq p$.*

Definition 11 (Circulation). *Let $h \in \mathbf{R}$ and $i, j \in E$, we define the equivalence relation $\mathcal{R}_h$ by $i\mathcal{R}_h j$, if either $i = j$ or i can be joined from j at level h and j can be joined from i at level h. We denote $\mathcal{C}_h(E)$ the set of all the equivalence classes of $\mathcal{R}_h$. $\mathcal{C}_h(E)$ is called the circulation at level h.*

We define the set $\mathcal{C}(E)$ of the circulations in E by

$$\mathcal{C}(E) = \bigcup_{h \in \mathbf{R}} \mathcal{C}_h(E).$$

We categorize the circulations by a level h. By Definition 11, it is immediate to show that two circulations in E are either disjoint sets or have a full intersection. Hence, they can be placed on a tree whose leaves are the singletons and whose root is the whole space.

We take an example of our random walk corresponding to $L_2^{\uparrow}$ on a Rips complex, see Figure 4.3. In the figure, $a,\ b,\ c,\ d,\ e,\ f,$ and g are states of our random walk. We can see that $\{a,\ b,\ c,\ d,\ e,\ f,\ g\} \subset \mathcal{C}(E)$. At level 5, we have $a\mathcal{R}_5 b$, then $\{a,\ b\} \subset \mathcal{C}_5(E)$. At level 6, we have $\{a,\ b,\ c,\ g\} \subset \mathcal{C}_6(E)$. At level 7, we have $\{a,\ b,\ c,\ d,\ f,\ g\} \subset \mathcal{C}_7(E)$, and all the states are in $\mathcal{C}_8(E)$. Therefore, we can have the decomposition of circulations by level and the decomposition tree shown in Figure 4.4.

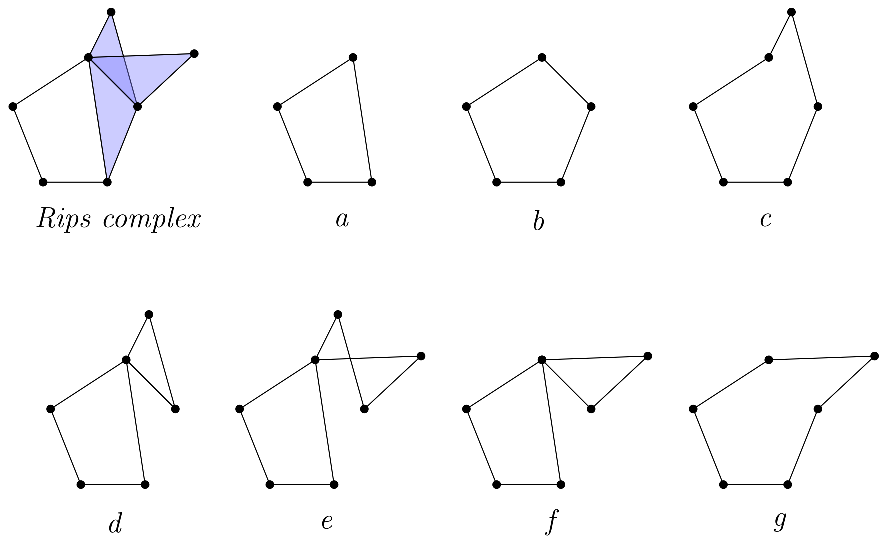


Figure 4.3: *Example of Rips complex and the states of random walk*

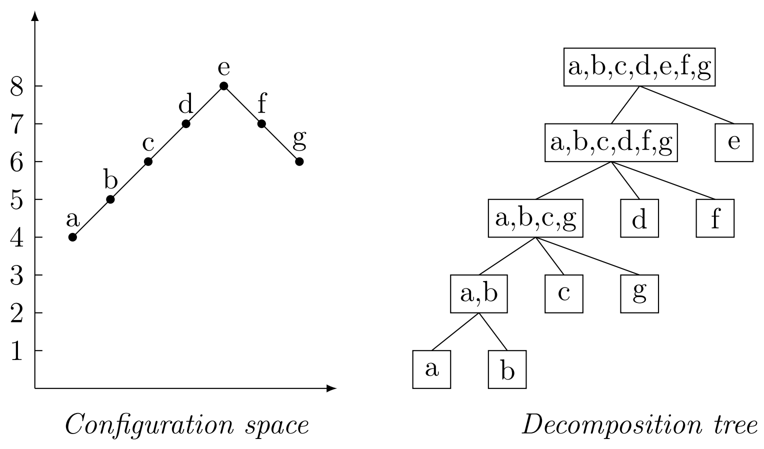


Figure 4.4: *Example of decomposition tree*

From the decomposition tree, we can see that some of the states are more "close" than others. To describe the relationship between circulations, we follow the definitions by Trouvé [46].

Definition 12. *Let* Π *be a circulation in* $\mathcal{C}(E)$*, we define*

1. *the bottom* $F(\Pi)$ *of* Π *by*

$$F(\Pi) = \{i \in \Pi \mid U(i) = \inf_{j \in \Pi} U(j)\},$$

2. *the boundary* $B(\Pi)$ *of* Π *by*

$$B(\Pi) = \{i \in E \setminus \Pi \mid \exists i \in \Pi,\ q(i,j) > 0\},$$

3. *the mixing height* $H_m(\Pi)$ *of* Π *by*

$$H_m(\Pi) = \sup_{i,j \in \Pi} (U(j) - U(i)),$$

4. *the exit height* $H_e(\Pi)$ *of* Π *by*

$$H_e(\Pi) = \inf_{j \in B(\Pi)} \sup_{i \in \Pi} (U(j) - U(i))^+,$$

5. *the potential* $U(\Pi)$ *of* Π *by*

$$U(\Pi) = \inf_{i \in \Pi} U(i),$$

6. the altitude $A_c(\Pi)$ of Π by

$$A_c(\Pi) = \max_{i \in \Pi} U(i).$$

For the previous example, let $\Pi = \{a, b, c, g\}$, by Definition 12 we have the bottom $F(\Pi) = \{a\}$, the boundary $B(\Pi) = \{d,\ f\}$, the mixing height $H_m(\Pi) = U(g) - U(a) = 2$, the exit height $H_e(\Pi) = (U(d) - U(a))^+ = 3$, and the potential $U(\Pi) = U(a) = 3$.

If the temperature is low, the probability of accepting an uphill move is close to zero. In this situation, our random walk goes from circulation to circulation and tries to find the leaf node of our decomposition tree with minimal energy. If we fix our temperature as T, for a circulation $\Pi \in \mathcal{C}(E)$, the exit time from Π is of order of $e^{H_e(\Pi)/T}$.

We look again at our previous example and we can describe the hierarchical behavior of the random walk when temperature is low in Figure 4.5. For the configuration space $E = \{a, b, c, d, e, f, g\}$, we have $H_m(E) = 4$, which means that the random walk visits all the configurations of E in a time of order of $e^{4/T}$. Now, if we consider $\Pi = \{a, b, c, d, f, g\}$ as the next circulation our random walk enters, our random walk will visit all the configurations of Π in a time of order of $e^{H_m(\Pi)/T} = e^{3/T}$, and the random walk will spend a time of order of $e^{H_e(\Pi)/T} = e^{4/T}$ before jumping to the point e and coming back to Π. The behavior within Π can be described clearly if we introduce the partition of Π in the sub-circulation including $\{a, b, c, g\}$, $\{d\}$ and $\{f\}$. Assume that our random walk enters $\{a, b, c, g\}$, it will spend a time of order of $e^{3/T}$ before jumping to either d or f and then returning to $\{a, b, c, g\}$. The behavior in $\{a, b, c, g\}$ will be described in the next sub-circulation and so on so forth. In the end, the behavior in $\{a, b\}$ will be described by introducing the partition $\{\{a\}, \{b\}\}$ at the bottom of the figure. Therefore, we have a hierarchical description of the trajectory inside E of our random walk organized by the circulation decomposition and the exit heights of the circulations.

If we consider a great number of circulations, in order to obtain minimal energy, we should empty successively all the circulations not containing the global minimum. By [46], we have the optimal exponent for the convergence

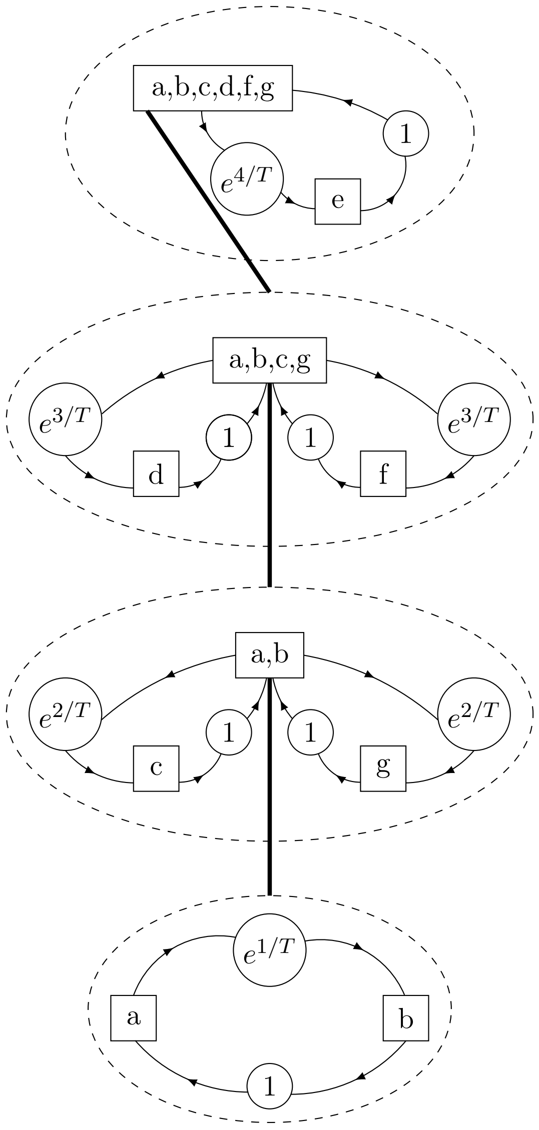


Figure 4.5: *Example of hierarchical behavior at low temperature*

rate towards the global minimal set:

$$\alpha_{opt} = \inf_{\Pi \in \mathcal{C}(E),\ \Pi \cap F(E) = \emptyset} \frac{U(\Pi) - U(E)}{H_e(\Pi)}, \tag{4.3}$$

where E is the configuration space with all the states.

Theorem 23 (Catoni [10]). *We assume that $\alpha_{opt} > 0$, there exists a constant Ksuch that for all integer $n \geq 1$, there exists a cooling schedule for which we have*

$$\sup_{i \in E} P(U(X_n) > \min U \mid X_0 = i) \leq \frac{K}{n^{\alpha_{opt}}}.$$

Corollary 24. *For a chain-valued integer-weighted random walk X and a simplicial complex $\mathcal{C}$ on dimension 2 with vertex set V, $X(0)$ is a cycle and is in the homology group including all holes. Energy function $U : \mathcal{S}_2 \to \mathbf{N}$ is the length of chain. The maximal length of chains in this homology group is L. For all $n \geq 1$, there exists a constant K and a cooling schedule for which we have*

$$\sup_{i \in E} P(U(X_n) > \min U \mid X_0 = i) \leq \frac{K}{n^{1/L}}.$$

Proof. We will focus on the value of α_{opt}. For all $\Pi \in \mathcal{C}(E)$, as we know that $U(\Pi) - \min U \geq 1$ under condition that $U(\Pi) > \min U$, it is crucial to have an upper bound of $H_e(\Pi)$.

Since after one step of our random walk, the energy of the chain will enlarge at most 2, which means that the chosen coface and the previous chain have two edges in common, we have

$$H_e(\Pi) = \sup_{i \in \Pi} (U(j) - U(i))^+ \leq 2 + H_m(\Pi).$$

For all $\Pi \in \mathcal{C}(E)$ and $\Pi \cap F(E) = \emptyset$, indicating that $U(\Pi) > U(E)$, we need to find the biggest Π which does not include the elements of $F(E)$ so as to have an upper bound of $H_m(\Pi)$. Since L is the maximal length of chains in the homology group, it is immediate to have $H_m(\Pi) < L$ for all $\Pi \in \mathcal{C}(E)$ and $\Pi \cap F(E) = \emptyset$.

By (4.3), we have

$$\begin{aligned} \alpha_{opt} &\geq \inf_{\Pi \in \mathcal{C}(E),\ U(\Pi) > \min U} \frac{1}{2 + H_m(\Pi)} \\ &\geq \frac{1}{L+2}. \end{aligned}$$

Therefore, there exists a constant K' so that $K/n^{\alpha_{opt}} \leq K'/n^{\frac{1}{L}}$, which completes the proof.

□

Remark 25. *For the upper bound of $H_m(\Pi)$, it is immediate to have $H_m(\Pi) < L$. However, for $\Pi \cap F(E) = \emptyset$, we know that the elements of Π and $F(E)$ are in the same circulation of level h where $h > A_c(\Pi)$. In order to have the biggest Π, we have $A_c(\Pi) \leq L - 1$. Therefore, to be more precise, we have $H_m(\Pi) \leq L - 1 - U(E)$.*

It is not easy to compute neither the maximal length of chains L nor the minimal length of chains $U(E)$, but in practice, we can use our random walk to find the longest path by tracking down the lengths of chains and taking the maximize of them.

4.6 Simulation results

In this section, we discuss two situations in general. In the first situation, we assume that we know a cycle which includes all the holes in advance, and we set it as our initial state. In this case, we do not need to apply the integer weighted random walk introduced in Section 4.2.2 since the weights are always in $\{-1, 0, 1\}$. In the second situation, we assume that we do not know any cycle of our Rips complex, so we calculate the initial state by looking at the eigenvectors of $\ker L_k$ corresponding to zero eigenvalues and apply the integer weighted random walk on it.

4.6.1 Random walk with known initial state

First, we draw 25 points randomly in the cube $[0, 1] \times [0, 1]$ and choose the max edge length of Rips complex as 0.3 so that $\beta_0 = 1$ and $\beta_1 = 2$. The Rips complex is shown in Figure 4.6.

In this case, we apply our simulated annealing algorithm with random walk on this Rips complex. For the parameters, we set $\alpha = 0.93$, $M = 100$, and $T_0 = 100$ so that in the end $T = T_0\alpha^M = 100 \times 0.93^{100}$ and the acceptance probability $P = \exp(-\Delta U/T) = \exp(-1/(100 \times 0.93^{100})) =$

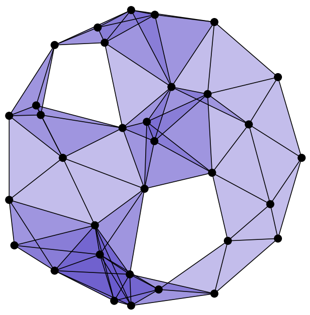
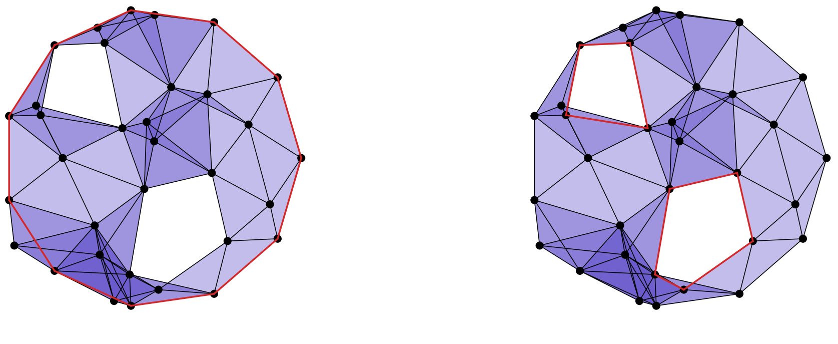


Figure 4.6: *Rips complex with 25 points and the max edge length* $\epsilon = 0.3$

6.94×10^{-7} is close to zero. At each T, we set $L = 1000$ to make the temperature decrease slowly.

With known initial state which is depicted as the red line in the left hand side of Figure 4.7, we know that the initial state includes both holes, and our random walk will not leave the same homology group as the initial state. The final state after our simulated annealing algorithm is shown in the right hand side of the figure.



(a) *Initial state* (b) *Final state*

Figure 4.7: *Rips complex with 25 points and the max edge length* $\epsilon = 0.3$*, initial state and final state is depicted in red line*

We can see that our simulated annealing algorithm locates all these two holes precisely.

4.6.2 Random walk with unknown initial state

If we do not know the initial state in advance, we should generate an initial state and make sure that it is in the same homology group with the holes in our simplicial complex. As we discussed in Section 4.2.2, we can study the null space of the matrix L_k, which is L_2 in our case. The number of zero eigenvalues equals with the dimension of H_2, which is the number of holes in our simplicial complex. For each zero eigenvalue, we calculate the corresponding eigenvector and set it to be our initial state. Therefore, for a simplicial complex with many holes, we can locate them one at a time.

However, in practice we encounter a computational accuracy problem. Since Python only prints a decimal approximation to the true decimal value of the binary approximation stored by the machine, the computational error will become super large after 10^5 steps of random walk. Therefore, we replace some of the codes in Python with SAGE since in SAGE, the exact value of fraction is stored by the machine, and we store all the fractions as integers by multiplying the least common multiple of the denominators of the fractions.

Therefore, we draw a new Rips complex where we do not know any cycle containing the hole in advance, see Figure 4.8. In this figure, we have one hole in the center, and we do not know the initial state in advance. In order to locate the hole, we should generate our initial state which is in the same homology group with the hole.

In this case, we calculate the eigenvector corresponding to the zero eigenvalue, and set it as our initial state. Figure 4.9a depicts our initial state, where the thickness of each edge indicates the weight of this edge.

For the parameters, we set $\alpha = 0.9$, $M = 100$, $T_0 = 100$ and $L_0 = 100$. The result is shown in Figure 4.9. In this figure, after 10^4 steps of random walk, the temperature T is close to zero and our state is in Figure 4.9b. We can see that our random walk goes through the boundaries of hole very often since the weights of them increase.

In Figure 4.9, we can see that the result is not as good as the random

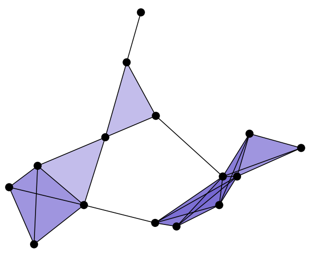
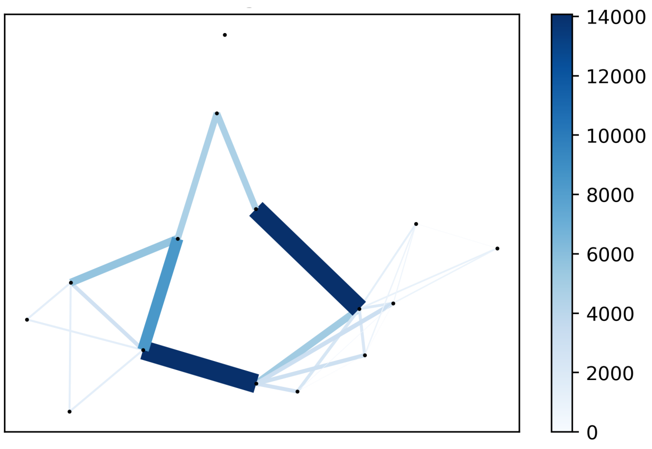


Figure 4.8: *Rips complex with 15 points and the max edge length* $\epsilon = 0.265$

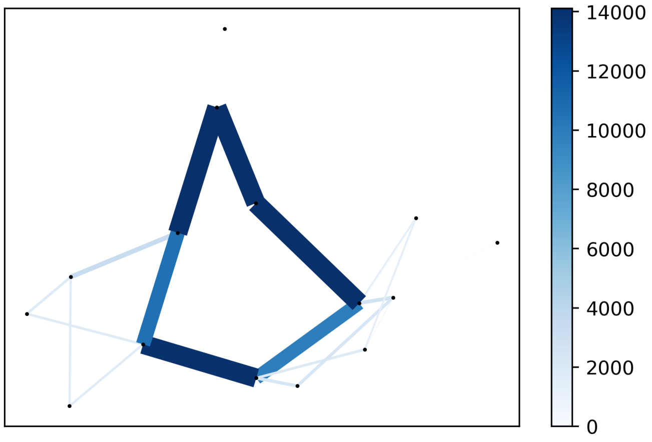
walk with known initial state, where we got exactly the boundaries of the holes. Without knowing the initial state, it seems that we can only obtain the boundaries of holes approximately. This is mainly due to the computational accuracy problem, since the weights are of order 10^4, and in order to minimize the weights, we cannot simply minimize them to a pretty small number. However, if the simplicial complex is pretty large, we will still have the boundaries of holes approximately, and in this situation, the boundaries will be very close to the theoretical result.

4.7 Summary and conclusion

In this chapter, we have introduced the random walk based hole detection algorithm on simplicial complexes. For a simplicial complex with holes, since we are not sure whether we can know the cycle which is in the same homology group with the holes or not, we divide our case into two situations: the initial state known or unknown. If we know the initial state, we can apply the random walk on $\mathbf{Z}_2$, which we have defined in Chapter 3. If we do not know the initial state in advance, we should calculate the initial state and redefine our random walk as the integer weighted random walk. Then, we introduce our algorithm, which is a simulated annealing algorithm, and we study the complexity and convergence rate of our algorithm. We find out that the convergence rate of our simulated annealing algorithm depends on the length of the longest cycle in our state space. For the simulation part, we apply our random walk on a Rips complex with both known initial state and unknown initial state, and finally we locate where the holes are. With known initial state, we can locate exactly the boundaries of holes, while with unknown initial state, we can locate the holes approximately.



(a) *Initial state*



(b) *Final state*

Figure 4.9: *Random walk on a Rips complex with unknown initial state*

Chapter 5

Random walk based kernels

Machine learning involves the study of relationships between structured objects in many domains such as bioinformatics, chemoinformatics, drug discovery, web data mining, and social networks. Graphs are natural data structures to model such structures, with nodes representing objects and edges the relations between them. We can measure the similarity between graphs by defining a kernel between them. However, with higher order topology data, simplicial complexes are better tools than graphs, since they can store high order structures, such as triangles, tetrahedron and so on. In this context, one often encounters two questions: "Can we measure the similarity between two simplicial complexes by defining a simplicial complexes kernel?" and "Does the kernel on simplicial complexes include more useful information than kernel on graphs?". In this chapter, we introduce the definition of random walk based kernels and compute the graph and simplicial complexes kernels on an example of three data structures. It tends to show that the simplicial complexes kernels are quite advantageous in detecting the similarities in homology.

5.1 Introduction

Roughly speaking, a kernel $k(x, x')$ is a measure of similarity between objects x and x'. It must satisfy two mathematical requirements: it must be symmetric, that is, $k(x, x') = k(x', x)$, and positive semi-definite (p.s.d.). Comparing graphs involves constructing a kernel between graphs. Vishwanathan [47] defined a kernel that captures the semantics inherent in the graph structure, and we would like to extend the definition of graph kernel to kernel on simplicial complexes so as to compare higher order topology data structures.

In order to compare two data structures, we first generate graphs or simplicial complexes for the data structures, and then we generate kernels based on this idea: given a pair of graphs or simplicial complexes, we perform our random walks on both, and count the number of matching walks. The more matching walks, the more similar these two data structures are.

The rest of this chapter is organized as follows. Section 5.2 introduces the definitions of kernels on both graphs and simplicial complexes, including the definitions of direct product graph and direct product simplicial complexes. Section 5.3 describes the algorithm to simulate the state space of chain-valued random walk, which determines the transition matrix and then the kernels on simplicial complexes. In Section 5.4, we discuss the computational complexity and we use conjugate gradient methods to speed up the computation. The computation settings and results are described in Section 5.5, where we compare the values of kernels on graphs and simplicial complexes on an example of three data structures. Finally, we make a summary and conclusion in Section 5.6.

5.2 Definitions and model

In this section, we introduce some basic definitions in order to define the kernel for graphs and simplicial complexes.

5.2.1 Graph kernels

Graphs are a special form of simplicial complexes, where graphs only have vertices and edges, and simplicial complexes have higher-dimensional topology structures. We introduce the direct product of a pair of graphs to link them as one large graph, and then introduce the definition of kernels over the large graph, which is the so-called graph kernels defined by S.V.N. Vishwanathan and others in [47]. Inspired by the definition of graph kernels, we will define the simplicial complexes kernels in Section 5.2.2.

First, we introduce some basic definitions in order to define the graph kernels.

Definition 13 (Kronecker product)**.** *Given real matrices $A \in \mathbf{R}^{n\times m}$ and $B \in \mathbf{R}^{p\times q}$, the Kronecker product $A \otimes B \in \mathbf{R}^{np\times mq}$ and column-stacking operator* $\operatorname{vec}(A) \in \mathbf{R}^{nm}$ *are defined as*

$$A \otimes B = \begin{bmatrix} A_{11}B & A_{12}B & \cdots & A_{1m}B \\ A_{21}B & A_{22}B & \cdots & A_{2m}B \\ \vdots & \vdots & \ddots & \vdots \\ A_{n1}B & A_{n2}B & \cdots & A_{nm}B \end{bmatrix}, \quad \operatorname{vec}(A) = \begin{bmatrix} A_{*1} \\ A_{*2} \\ \vdots \\ A_{*m} \end{bmatrix},$$

where A_{*j} denotes the j^{th} column of A.

Definition 14 (Direct product graphs [47])**.** *Given two graphs $G(V, E)$ and $G'(V', E')$, their direct product $G_\times$ is a graph with vertex set*

$$V_\times = \{(v_i, v'_r) : \; v_i \in V, v'_r \in V'\},$$

and edge set

$$E_\times = \{[(v_i, v'_r), (v_j, v'_s)] : \; [v_i, v_j] \in E \text{ and } [v'_r, v'_s] \in E'\}.$$

When G is unweighted with n vertices, we define its adjacency matrix as the $n \times n$ matrix $\tilde{A}$ with $\tilde{A}_{ij} = 1$ if $[v_i, v_j] \in E$ and 0 otherwise. For weighted graphs, $\tilde{A}_{ij} = w_{ij}$. The adjacency matrix has a normalized cousin, defined by $A := \tilde{A}D^{-1}$, which has the property that each of its columns sums to one, and it can therefore serve as the transition matrix for a stochastic process. Here, D is a diagonal matrix of node degrees, that is, $D_{ii} = d_i = \sum_j \tilde{A}_{ij}$. A random walk on G is the traditional random walk jumping from vertex to vertex subject to $\mathbf{P}(i_{k+1}|i_1, \dots i_k) = A_{i_k,i_{k+1}}$.

If A and A' are the respective adjacency matrices of G and G' , then the adjacency matrix of $G_\times$ is $\tilde{A}_\times = \tilde{A} \otimes \tilde{A}'$. Similarly, $A_\times = A \otimes A'$.

By Imrich and Klavžar [26], performing a random walk on the direct product graph $G_\times$ is equivalent to performing a simultaneous random walk on G and G'. If p and p' denote initial probability distributions over the vertices of G and G', then the corresponding initial probability distribution on the direct product graph $G_\times$ is $p_\times := p \otimes p'$. Likewise, if q and q' are stopping probabilities (that is, the probability that a random walk ends at a given vertex), then the stopping probability on the direct product graph $G_\times$ is $q_\times := q \otimes q'$.

It is well known that any continuous, symmetric, positive definite kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbf{R}$ has a corresponding Hilbert space $\mathcal{H}$, called the Reproducing Kernel Hilbert Space, which induces a feature map $\phi : \mathcal{X} \to \mathcal{H}$ satisfying $k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}}$. The natural extension of this so-called feature map to matrices is $\Phi : \mathcal{X}^{n \times n'} \to \mathcal{H}^{n \times n'}$ defined $[\Phi(A)]_{ij} := \phi(A_{ij})$.

Let $|V| =: n$ and $|V'| =: n'$. If G and G' are edge-labeled, we can associate a weight matrix $W_\times \in \mathbf{R}^{nn' \times nn'}$ with $G_\times$ by

$$W_\times = \Phi(X) \otimes \Phi(X'). \tag{5.1}$$

In our case, we simply let $\Phi(X) = A$, and then $W_\times = A_\times$.

We perform the random walk jumping from vertex to vertex on $G_\times$. In fact, the $((i-1)n' + r, (j-1)n' + s)^{\text{th}}$ entry of $A_\times^k$ represents the probability of simultaneous length k random walks on G (starting from vertex v_j and ending at vertex v_i) and G' (starting from vertex v'_s and ending at vertex v'_r), and so does $W_\times$. Given initial and stopping probability distributions $p_\times$ and $q_\times$ one can compute $q_\times^\top W_\times^k p_\times$, which is the expected similarity between simultaneous length k random walks on G and G'. Therefore, following by [26], we have the graph kernel definition to compute the similarity between G and G'.

Definition 15 (Graph kernel)**.** *For any two graphs G and G', the direct product graph of G and G' is $G_\times$, and the weight matrix of $G_\times$ is $W_\times$. We perform a random walk on $G_\times$ with initial probability $p_\times$ and stopping probability $q_\times$. The kernel between G and G' is defined by*

$$k(G, G') = \sum_{k=0}^{\infty} \mu(k) q_\times^\top W_\times^k p_\times, \tag{5.2}$$

where $\mu(k)$ is a non-negative coefficient to make sure that $k(G, G')$ converges.

5.2.2 Simplicial complexes kernels

Since the definition of simplicial complexes kernels is based on the idea of graph kernels where we perform a random walk on a pair of graphs, we extend the definition of graph kernels and perform our chain-valued random walk on a pair of simplicial complexes. We introduce direct product of simplicial complexes to link them as one large simplicial complex. Then, we define the random walk based kernel over the product of simplicial complexes.

Extending the definition of direct product graphs in Definition 14, we give the definition of direct product simplicial complexes.

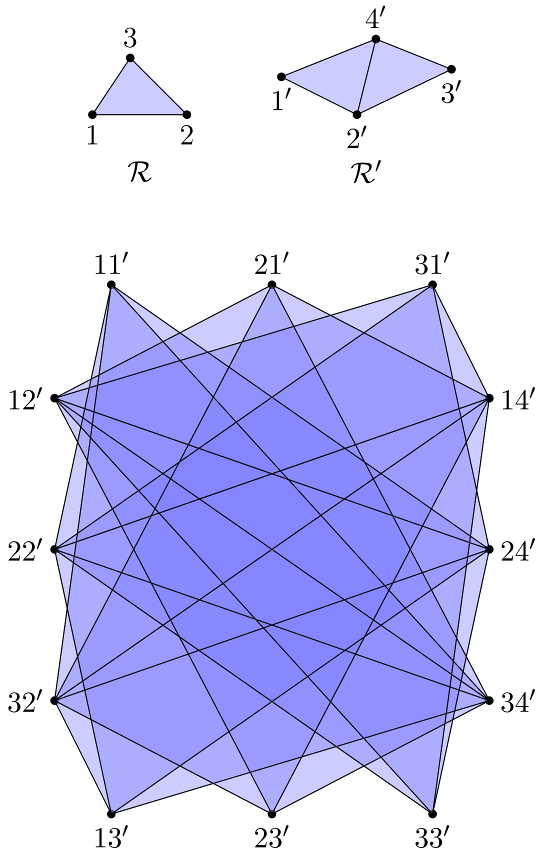
Definition 16. *(Direct product simplicial complexes) Let $\mathcal{C}$ and $\mathcal{C}'$ be abstract simplicial complexes and choose a linear ordering of the vertices. We define the direct product of $\mathcal{C}$ and $\mathcal{C}'$ to be the simplicial complex $\mathcal{C}_\times$ with the following properties:*

1. *Its vertex set*
$$V_\times = \{(v_i, v'_r) : \ v_i \in V, v'_r \in V'\},$$
where V is the vertex set of $\mathcal{C}$ and V' is the vertex set of $\mathcal{C}'$.

2. *Its k-simplex set*
$$(\mathcal{S}_k)_\times = \{[(v_{i_0}, v'_{r_0}), \ldots, (v_{i_k}, v'_{r_k})] : \ [v_{i_0}, \ldots, v_{i_k}] \in \mathcal{S}_k, [v'_{r_0}, \ldots, v'_{r_k}] \in \mathcal{S}'_k\}$$
where $\mathcal{S}_k$ is the k-simplex set of $\mathcal{C}$ and $\mathcal{S}'_k$ is the k-simplex set of $\mathcal{C}'$.

As the direct product graph is a graph over pairs of vertices from the original two graphs, $\mathcal{C}_\times$ is a simplicial complex over pairs of vertices and simplices from $\mathcal{C}$ and $\mathcal{C}'$. Two vertices are neighbors if and only if the corresponding vertices in $\mathcal{C}$ and $\mathcal{C}'$ are both neighbors, and two edges are upper (lower) adjacent if and only if the corresponding edges in $\mathcal{C}$ and $\mathcal{C}'$ are both upper (lower) adjacent.

Here, we give an example to explain our definition. For two Rips complexes $\mathcal{R}$ and $\mathcal{R}'$, we have their 0-simplex sets (vertex sets) $\mathcal{S}_0 = \{1, 2, 3\}$ and $\mathcal{S}'_0 = \{1', 2', 3', 4'\}$. The 1-simplex sets are $\mathcal{S}_1 = \{[1, 2], [1, 3], [2, 3]\}$ and $\mathcal{S}'_1 = \{[1', 2'], [1', 3'], [2', 3'], [2', 4'], [3', 4']\}$. The 2-simplex sets are $\mathcal{S}_2 = \{[1, 2, 3]\}$



The direct product simplicial complex $\mathcal{R}_\times$ of $\mathcal{R}$ and $\mathcal{R}'$

Figure 5.1: *Example of two simplicial complexes $\mathcal{R}$ and $\mathcal{R}'$ and their direct product simplicial complexes $\mathcal{R}_\times$ Each node of the direct product simplicial complex is labeled with a pair of nodes, such as $11', 21'$; an edge exists in the direct product if and only if the corresponding nodes are adjacent in both original simplicial complexes; a 2-simplex exists in the direct product if and only if the corresponding nodes are elements of a 2-simplex in both original simplicial complexes. For instance, nodes $12'$ and $31'$ are adjacent because there is an edge between nodes 1 and 3 in $\mathcal{R}$, and $2'$ and $1'$ in $\mathcal{R}'$. Similarly, $[11', 22', 34']$ is a 2-simplex in $\mathcal{R}_\times$ because $[1, 2, 3]$ is a 2-simplex in $\mathcal{R}$ and $[1', 2', 4']$ is a 2-simplex in $\mathcal{R}'$.*

and $\mathcal{S}'_2 = \{[1', 2', 4'], [2', 3', 4']\}$. For $k > 2$, we have $\mathcal{S}_k = \mathcal{S}'_k = \emptyset$. By Definition 16, we have $(\mathcal{S}_0)_\times$, $(\mathcal{S}_1)_\times$ and $(\mathcal{S}_2)_\times$ shown in Figure 5.1.

Since the random walk performed on direct product graphs is the random walk jumping from vertex to vertex, in order to extend the random walk to simplicial complexes, we need our random walk jumping from chains to chains. Thus, we introduce some basic chain-valued random walk concepts on simplicial complexes so as to introduce the definition of simplicial complexes kernels later.

As defined in (3.4), for any k-chain τ and τ', we have the transition of our chain-valued random walk $K(\tau, \tau')$ as

$$K(\tau,\, \tau') = \begin{cases} 1 & \text{if } \tau = \tau' = 0 \\ w\left(\tau,\ \tau - \tau'\right) & \text{if } \tau \sim \tau' \\ 0 & \text{otherwise.} \end{cases}$$

Following the definition of $K(\tau,\, \tau')$, we can give the definition of transition matrix $\mathcal{K}$.

Definition 17 (Transition matrix)**.** *Given a simplicial complex $\mathcal{C}$ and an initial state τ_{ini}, we have the recurrence class $\mathcal{R}(\tau_{ini})$ of τ_{ini} whose dimension is N. For any $\tau_i, \tau_j \in \mathcal{R}(\tau_{ini})$, if τ_i and τ_j are labeled as i and j in $\mathcal{R}(\tau_{ini})$, the transition matrix $\mathcal{K}$ is defined by*

$$\mathcal{K}_{ij} = \frac{K(\tau_i,\, \tau_j)}{\sum_{k=1}^{N} K(\tau_i,\, \tau_k)}.$$

In order to avoid ambiguity, we denote K by the unnormalized matrix whose entries are $K(\tau_i,\, \tau_j)$, and we denote $\mathcal{K}$ by the transition matrix after renormalizing by the sum of each rows of K.

A random walk on $\mathcal{C}$ is a process generating sequences of chains $\tau_{i_1},\, \tau_{i_2},\, \tau_{i_3},\, \ldots$ according to $\mathbf{P}(i_{k+1}|i_1, \ldots, i_k) = \mathcal{K}_{i_k, i_{k+1}}$, that is, the probability at τ_{i_k} of picking $\tau_{i_{k+1}}$ next is proportional to the number of common edges τ_{i_k} and $\tau_{i_{k+1}}$ share. The t^{th} power of $\mathcal{K}$ thus describes t-length walks, that is, $(\mathcal{K}^t)_{ij}$ is the probability of a transition from chain τ_i to τ_j via a walk of length t. If p_0 is an initial probability distribution over chains in the same recurrence class, then the probability distribution p_t describing the state of our random walker at time t is $p_t = \mathcal{K}^t p_0$. The j^{th} component of p_t denotes the probability of finishing a t-length walk at state τ_j.

In our case, the random walk does not continue infinitely. So, we associate every state τ_{i_k} with a stopping probability q_{i_k}. Our generalized random walk kernels then use the overall probability of stopping after t steps, given by $q^{\top}p_t$. As mentioned in [47], like p_0, the vector q of stopping probabilities is a place to embed prior knowledge into the kernel design. Since p_t as a probability distribution sums to one, a uniform vector q (as might be chosen in the absence of prior knowledge) would yield the same overall stopping probability for all p_t, thus leading to a kernel that is invariant with respect to the structure it is meant to measure. In this case, the unnormalized transition K should be used instead of $\mathcal{K}$.

If $\mathcal{K}$ and $\mathcal{K}'$ are the respective transition matrices of $\mathcal{C}$ and $\mathcal{C}'$, then the transition matrix of $\mathcal{C}_\times$ is $\mathcal{K}_\times = \mathcal{K} \otimes \mathcal{K}'$. Similarly, $K_\times = K \otimes K'$.

Performing a random walk on the direct product graph is equivalent to performing a simultaneous random walk on those two graphs [26], and so does simplicial complexes. If p and p' denote initial probability distributions over the chains on $\mathcal{C}$ and $\mathcal{C}'$, then the corresponding initial probability distribution on the direct product simplicial complex is $p_\times := p \otimes p'$. Likewise, if q and q' are stopping probabilities (that is, the probability that a random walk ends at a given chain), then the stopping probability on the direct product simplicial complex is $q_\times := q \otimes q'$.

Inspired by Equation (5.1), for simplicial complexes $\mathcal{C}$ and $\mathcal{C}'$, the dimensions of the corresponding transition matrix $\mathcal{K}$ and $\mathcal{K}'$ are $N \times N$ and $N' \times N'$, we associate a weight matrix $W_\times \in \mathbf{R}^{NN' \times NN'}$ with $\mathcal{C}_\times$ by $W_\times = \mathcal{K}_\times$.

Therefore, we can extend the definition of graph kernels in Definition 15 to simplicial complexes kernels as follows.

Definition 18. *For any two simplicial complexes $\mathcal{C}$ and $\mathcal{C}'$, the direct product simplicial complex of $\mathcal{C}$ and $\mathcal{C}'$ is $\mathcal{C}_\times$, and the weight matrix of $\mathcal{C}_\times$ is $W_\times$. We perform our chain-valued random walk on $\mathcal{C}_\times$ with initial probability $p_\times$ and stopping probability $q_\times$. The kernel between $\mathcal{C}$ and $\mathcal{C}'$ is defined by*

$$k(\mathcal{C}, \mathcal{C}') = \sum_{k=0}^{\infty} \mu(k) q_\times^{\top} W_\times^k p_\times, \tag{5.3}$$

where $\mu(k)$ is a non-negative coefficient to make sure that $k(\mathcal{C}, \mathcal{C}')$ converges.

5.3 Algorithm

In order to compute the simplicial complexes kernels between two simplicial complexes $\mathcal{C}$ and $\mathcal{C}'$, we need to determine the coefficient $\mu(k)$ and the weight matrix $W_\times$.

For the coefficient $\mu(k)$, since it is chosen to make sure the kernel converges, we can emphasize or de-emphasize walks of different lengths by choosing it appropriately. Without loss of generality, we set $\mu(k) = \lambda^k$ where λ is small enough for the sum in (5.3) to converge. By Taylor expansion, the kernel between $\mathcal{C}$ and $\mathcal{C}'$ can be rewritten as

$$k(\mathcal{C}, \mathcal{C}') = \sum_{k=0}^{\infty} \lambda^k q_\times^\top W_\times^k p_\times = q_\times^\top \left(I - \lambda W_\times\right)^{-1} p_\times. \tag{5.4}$$

For the weight matrix $W_\times$, if we focus on graphs, we need to generate the adjacency matrices A and A' of graph G and G', and then apply Kronecker product on them. However, if we focus on simplicial complexes, the situation becomes a little complex. Given two simplicial complexes $\mathcal{C}$ and $\mathcal{C}'$we need to determine $\mathcal{K}_\times$. Since $\mathcal{K}_\times = \mathcal{K} \otimes \mathcal{K}'$, where $\mathcal{K}$ and $\mathcal{K}'$ are the transition matrices of the random walk starting from τ_{ini} and τ'_{ini} on $\mathcal{C}$ and $\mathcal{C}'$ respectively, we need to determine $\mathcal{K}$ and $\mathcal{K}'$.

In order to compute the transition matrix $\mathcal{K}$ of simplicial complex $\mathcal{C}$ given the initial state τ_{ini}, first, we need to find all the states in the state space of our random walk. In Section 3.2.3, we explained the recurrence of the chain on simplicial complexes. Due to the nature of our random walk, it will never be trapped at any state. Therefore, we can find all the states by simulating our random walk after a large number of steps on the given simplicial complex. At first, we set our recurrence class $\mathcal{R}(\tau_{ini})$ to be a null space. Starting from τ_{ini}, we let our random walk run M steps where M is large. If the chain we get is not in $\mathcal{R}(\tau_{ini})$, we add the chain to $\mathcal{R}(\tau_{ini})$; if the chain is already in $\mathcal{R}(\tau_{ini})$, we continue our random walk. After M steps, $\mathcal{R}(\tau_{ini})$ will not enlarge any more and we refer it as our state space. This procedure is summarized in the Algorithm 3.

Knowing the state space of our random walk initiated with τ_{ini}, we label them with the numeric position in $\mathcal{R}(\tau_{ini})$. For example, the first element added to $\mathcal{R}(\tau_{ini})$ should be τ_{ini}, which is labeled with 1, and the second

Algorithm 3 State space of random walk

```
    Input: $\tau_{ini}$ the initial state, $\mathcal{C}$ the simplicial complex
    Output: $\mathcal{R}(\tau_{ini})$ the recurrence class of $\tau_{ini}$ on $\mathcal{C}$, i.e., state space of random walk initiated with $\tau_{ini}$
  procedure STATE SPACE($\tau_{ini}$)
      Let $\mathcal{R}(\tau_{ini})$ be a null space
      Add $\tau_{ini}$ to $\mathcal{R}(\tau_{ini})$
      $\tau_1 = \tau_{ini}$
      for $i = 1 \to M$ do
          $\tau_{i+1} = RW(\tau_i)$ one step of random walk
          if $\tau_{i+1} \notin \mathcal{R}(\tau_{ini})$ then
              Add $\tau_{i+1}$ to $\mathcal{R}(\tau_{ini})$
          end if
      end for
      return $\mathcal{R}(\tau_{ini})$
  end procedure
```

element added to $\mathcal{R}(\tau_{ini})$ is denoted by τ_2, which is labeled with 2. For the transition matrix $\mathcal{K}$, the element in the second column of the first row is the probability at τ_{ini} of picking τ_1 next. If the number of elements in $\mathcal{R}(\tau_{ini})$ is N, $\mathcal{K}$ is a $N \times N$ matrix. For any $1 \leq i, j \leq N$, we can compute $\mathcal{K}_{ij}$ by

$$\mathcal{K}_{ij} = \frac{K(\tau_i,\, \tau_j)}{\sum_{k=1}^{N} K(\tau_i,\, \tau_k)}.$$

Thus, we can obtain the transition matrices $\mathcal{K}$ and $\mathcal{K}'$ of random walks on $\mathcal{C}$ and $\mathcal{C}'$ initiated with τ_{ini} and τ'_{ini} respectively. Applying Kronecker product on $\mathcal{K}$ and $\mathcal{K}'$, we have $\mathcal{K}_\times$, which equals to $W_\times$, and then we can compute the simplicial complex kernel $k(\mathcal{C}, \mathcal{C}')$.

5.4 Computational complexity

Computing a geometric random walk kernel with $\mu(k) = \lambda^k$ amounts to computing the inverse matrix of $(I - \lambda W_\times)$. For simplicial complex kernel, $(I - \lambda W_\times)$ is an $NN' \times NN'$ matrix if $\mathcal{R}(\tau_{ini})$ and $\mathcal{R}(\tau'_{ini})$ have N and N' elements respectively. Since the complexity of inverting a matrix is essentially cubic in its dimensions, direct computation of (5.4) would require $\mathcal{O}((NN')^3)$ time. Since exact computation of the full kernel matrix is generally cubic in their size, we develop iterative methods to decrease the computation complexity.

Given a symmetric and positive-definite matrix M and a vector b, conjugate gradient (CG) methods solve the system of equations $Mx = b$ efficiently. The conjugate gradient method is often implemented as an iterative algorithm, applicable to sparse systems that are too large to be handled by a direct implementation or other direct methods such as the Cholesky decomposition. For more details, see [33]. Since our transition matrix is naturaly symmetric, $I - \lambda W_\times$ is symmetric and positive-definite. Therefore, we can use conjugate gradient methods to compute our kernel.

The random walk based kernel (5.2 and 5.4) can be computed by a two-step procedure:

1. we solve the linear system
$$(I - \lambda W_\times)x = p_\times,$$
2. and we compute $q_\times^\top x$.

In the first step, we apply conjugate gradient methods to solve the linear system. The conjugate gradient method can be used as an iterative method as it provides monotonically improving approximations x_k to the exact solution, which may reach the required tolerance after a relatively small (compared to the problem size) number of iterations. The improvement is typically linear and its speed is determined by the condition number $\kappa(A)$ of the system matrix A: the larger $\kappa(A)$, the slower the improvement. By [43], we know that conjugate gradient method has a time complexity of $\mathcal{O}(m\sqrt{\kappa})$, where m is the number of non-zero entries in A. Finite difference and finite element approximations of second-order elliptic boundary value problems posed on d-dimensional domains often have $\kappa \in \mathcal{O}(n^{2/d})$ if A is an $n \times n$ matrix.

Recall that $\mathcal{R}(\tau_{ini})$ and $\mathcal{R}(\tau'_{ini})$ have N and N' elements respectively, then $W_\times$ is an $NN' \times NN'$ matrix. Therefore, $\kappa(W_\times) \in \mathcal{O}((NN'))$. Moreover, due to the definition of the transition matrix, we know that the number of non-zero entries in each row of $\mathcal{K}$ is at most the number of neighboring triangles of the corresponding state. Thus, the number of non-zeros in $W_\times$ is at most $\mathcal{O}(NN')$. We can conclude that computing the simplicial complex kernel using conjugate gradient methods has a complexity of $\mathcal{O}((NN')^{3/2})$, which speeds up a lot comparing to the direct computation.

5.5 Computation results

In this section, we intend to study the similarity among three data structures using both graph kernels and simplicial complex kernels. We are quite interested in whether simplicial complex kernels could include more information than graph kernels.

5.5.1 Data structures

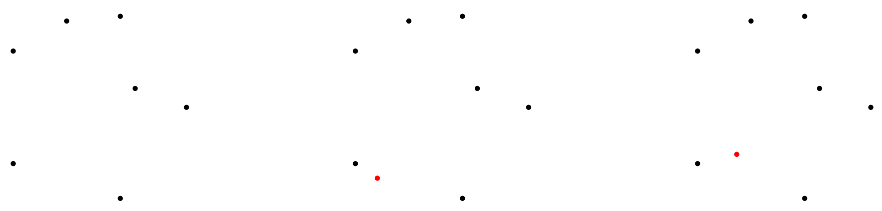
First, we define our three data structures, denoted by $(DS)_0$, $(DS)_1$, and $(DS)_2$ in Figure 5.2. $(DS)_1$ and $(DS)_2$ have all the points $(DS)_0$ has, and they have their own perturbation points, which are depicted in red in Figure 5.2b and Figure 5.2c. The difference between two perturbation points lies in the case that the perturbation point in $(DS)_1$ does not change the homology property but the perturbation point in $(DS)_2$ does. In other words, we reckon that the perturbation point in $(DS)_1$ is very close to one of the original points in $(DS)_0$, and it does not make a big difference on the original structure. However, the perturbation point in $(DS)_2$ is close to more than one of the original points, and it changes the original data structure.

If we generate the graphs and Rips complexes of these three data structures, the situation becomes clearer, see Figure 5.3 and Figure 5.4.

5.5.2 Graph kernels and simplicial complexes kernels

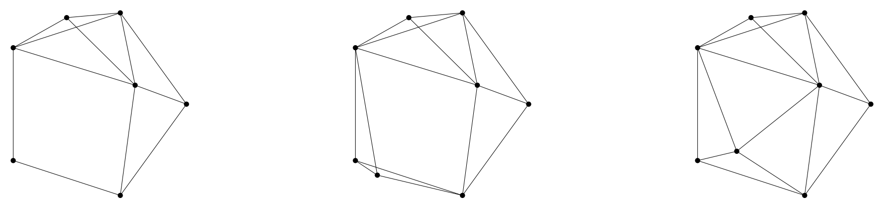
We compute the graph kernels $k(G_0, G_1)$, $k(G_0, G_2)$ and $k(G_1, G_2)$ respectively. In the same time, we compute the simplicial complex kernels $k(\mathcal{C}_0, \mathcal{C}_1)$, $k(\mathcal{C}_0, \mathcal{C}_2)$ and $k(\mathcal{C}_1, \mathcal{C}_2)$.

For graph kernels, we label all the vertices in numerical order and generate



(a) $(DS)_0$ (b) $(DS)_1$ (c) $(DS)_2$

Figure 5.2: *Data structures $(DS)_0$ with 7 points, $(DS)_1$ with 8 points and $(DS)_2$ with 8 points, with different points between itself and $(DS)_0$ depicted in red*



(a) G_0 (b) G_1 (c) G_2

Figure 5.3: *Graphs G_0 with 7 points, G_1 with 8 points and G_2 with 8 points, with max edge length $\epsilon = 0.7$*

the adjacency matrices of G_0, G_1 and G_2. We take G_0 as an example. The graph is shown in Figure 5.5. The adjacency matrix A_0 of G_0 is as followed:

$$A_0 = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 0 \end{bmatrix}.$$

Following the same procedure, we have the adjacency matrices A_1 of G_1 and A_2 of G_2. Then we compute the Kronecker products $W_\times^{01} = A_0 \otimes A_1$, $W_\times^{02} = A_0 \otimes A_2$ and $W_\times^{12} = A_1 \otimes A_2$. We define the initial and stopping

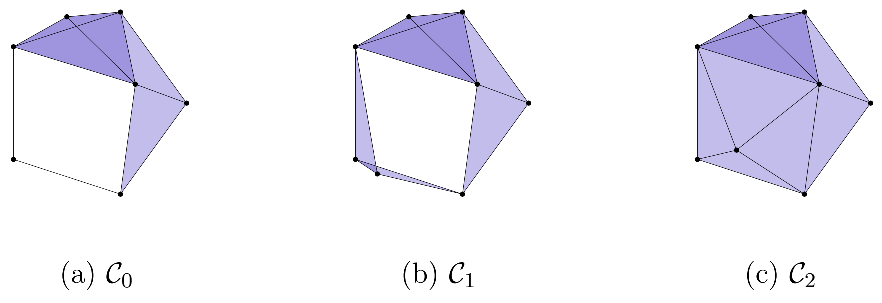

(a) $\mathcal{C}_0$ (b) $\mathcal{C}_1$ (c) $\mathcal{C}_2$

Figure 5.4: *Rips complex $\mathcal{C}_0$ with 7 points, $\mathcal{C}_1$ with 8 points and $\mathcal{C}_2$ with 8 points, with max edge length $\epsilon = 0.7$*

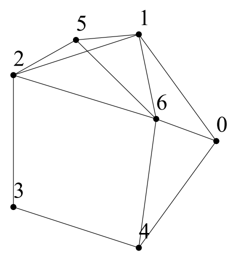


Figure 5.5: *G_0 with labeled vertices*

probabilities of G_0, G_1, G_2 as p_0, q_0, p_1, q_1, p_2 and q_2, which are uniformly distributed due to the fact that every vertex is equivalent with each other. Then we compute the Kronecker products $p_\times^{01} = p_0 \otimes p_1$, $q_\times^{01} = q_0 \otimes q_1$ and so on so forth. By definition of graph kernels and the two-step procedure in Section 5.4, we let $\lambda = 0.01$ and use conjugate gradient solver to solve the linear system

$$(I - \lambda W_\times^{01})x = p_\times^{01},$$

then we get $k(G_0, G_1) = (q_\times^{01})^\top x$. We compute $k(G_0, G_2)$ and $k(G_1, G_2)$ respectively. The results are shown in Table 5.1a.

For simplicial complexes, first, we label all the vertices in numerical order and we generate the transition matrices of the random walk on $\mathcal{C}_0$, $\mathcal{C}_1$ and $\mathcal{C}_2$ starting with its corresponding initial state. In our case, the initial states of the random walk on $\mathcal{C}_1$ and $\mathcal{C}_2$ are the same with which on $\mathcal{C}_0$. The initial state τ_{ini} is depicted in red in Figure 5.6.

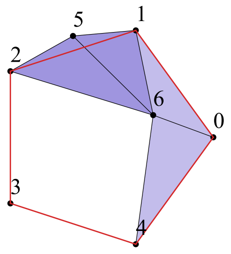
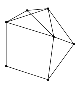
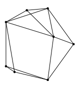
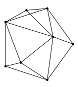
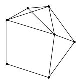
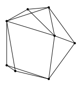
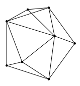


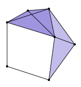
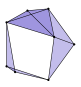
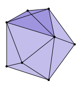
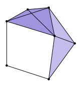
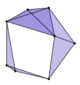
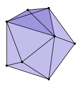
Figure 5.6: $\mathcal{C}_0$ *with labeled vertices and initial state in red*

In order to generate the transition matrices of the random walk on $\mathcal{C}_0$, $\mathcal{C}_1$ and $\mathcal{C}_2$, we simulate the state spaces $\mathcal{R}(\tau_{ini})_0$, $\mathcal{R}(\tau_{ini})_1$ and $\mathcal{R}(\tau_{ini})_2$ by Algorithm 3, and then generate the transition matrices $\mathcal{K}_0$, $\mathcal{K}_1$ and $\mathcal{K}_2$ by computing $K(\tau, \tau')$ following (3.4), where τ and τ' are the states in the corresponding state space. As a result, we have $\mathcal{K}_0$ a 32×32 sparse matrix with 192 stored elements, $\mathcal{K}_1$ an 128×128 sparse matrix with 1024 stored elements, and $\mathcal{K}_2$ a 511×511 sparse matrix with 5100 stored elements. Then we compute the Kronecker products $W_\times^{01} = \mathcal{K}_0 \otimes \mathcal{K}_1$, $W_\times^{02} = \mathcal{K}_0 \otimes \mathcal{K}_2$ and $W_\times^{12} = \mathcal{K}_1 \otimes \mathcal{K}_2$, where $W_\times^{01}$ is a 4096×4096 sparse matrix with 196608 stored elements, $W_\times^{02}$ is a 16352×16352 sparse matrix with 979200 stored elements, and $W_\times^{12}$ is a 65408×65408 sparse matrix with 5222400 stored elements. For the initial probabilities, we search for the index i of state $\{(0,1),(0,4),(1,2),(2,3),(3,4)\}$ (depicted in red in Figure 5.6) in each state space of random walk respectively, and we set the i^{th} element to be 1 and other elements to be 0. The stopping probabilities are uniformly distributed since we do not know any information on the stopping states. Therefore, we have p_0, p_1, p_2, q_0, q_1, and q_2. Then we compute the Kronecker products $p_\times^{01}$, $p_\times^{02}$, $p_\times^{12}$, $q_\times^{01}$, $q_\times^{02}$ and $q_\times^{12}$ and solve the linear system as graph kernels using conjugate gradient solvers in order to compute $k(\mathcal{C}_0, \mathcal{C}_1)$, $k(\mathcal{C}_0, \mathcal{C}_2)$ and $k(\mathcal{C}_1, \mathcal{C}_2)$. Due to the size of our sparse matrix, we set λ to be 0.001 so that (5.4) converges. The results are shown in Table 5.1b.

We can see that by graph kernels, the similarity between graphs subjects to $k(G_1, G_2) < k(G_0, G_1) < k(G_0, G_2)$, and the values of $k(G_1, G_2)$ and $k(G_0, G_1)$ are very close. However, by simplicial complexes kernels, the sim-

$\lambda = 0.01$	 G_0	 G_1	 G_2
 G_0	-	0.01979	0.01981
 G_1	0.01979	-	0.01737
 G_2	0.01981	0.01737	-

(a) *Graph kernels with* $\lambda = 0.01$

$\lambda = 0.001$	 $\mathcal{C}_0$	 $\mathcal{C}_1$	 $\mathcal{C}_2$
 $\mathcal{C}_0$	-	0.00032	0.00009
 $\mathcal{C}_1$	0.00032	-	0.00002
 $\mathcal{C}_2$	0.00009	0.00002	-

(b) *Simplicial complexes kernels with* $\lambda = 0.001$

Table 5.1: *Graph and simplicial complexes kernels between the corresponding graphs and simplicial complexes*

ilarity subjects to $k(\mathcal{C}_1, \mathcal{C}_2) < k(\mathcal{C}_0, \mathcal{C}_2) < k(\mathcal{C}_0, \mathcal{C}_1)$. Note that the value of kernel is a relative concept, which means that we can only compare $k(G_1, G_2)$ with $k(G_0, G_1)$ or $k(G_0, G_2)$ and it makes no sense of focusing on the value of $k(G_1, G_2)$ itself. In fact, $k(G_0, G_1)$, $k(G_0, G_2)$ and $k(G_1, G_2)$ are in the same magnitude order, so we conclude that the graph kernels do not tell any big difference between G_0, G_1 and G_2. Moreover, since graphs only consist of vertices and edges, the graph kernels do not take account of the homology properties of the data structure.

By graph kernels, the most similar graphs are G_0 and G_2, but we can tell from Figure 5.4 that the simplicial complex version of G_0 has a hole ($\beta_1 = 1$) while the simplicial complex version of G_1 does not have any hole ($\beta_1 = 0$). On the other hand, by simplicial complexes kernels, it sorts perfectly by the homology properties. See Table 5.1b, the most similar simplicial complexes are $\mathcal{C}_0$ and $\mathcal{C}_1$ by value of $k(\mathcal{C}_0, \mathcal{C}_1)$. Algebraically speaking, $\mathcal{C}_0$ and $\mathcal{C}_1$ has the same homology properties, whose β_0 is 1 and β_1 is 1. At the same time, the value of $k(\mathcal{C}_0, \mathcal{C}_1)$ is an order of magnitude higher than the value of $k(\mathcal{C}_0, \mathcal{C}_2)$ while $k(\mathcal{C}_0, \mathcal{C}_2)$ and $k(\mathcal{C}_1, \mathcal{C}_2)$ are in the same magnitude order, which agrees with the fact that the similarity between $\mathcal{C}_0$ ($\beta_0 = 1,\ \beta_1 = 1$) and $\mathcal{C}_1$ ($\beta_0 = 1,\ \beta_1 = 1$) is far more than the similarity between $\mathcal{C}_0$ ($\beta_0 = 1,\ \beta_1 = 1$) and $\mathcal{C}_2$ ($\beta_0 = 1,\ \beta_1 = 0$). Apart from the similarity in homology properties, the value order of simplicial complexes kernels is same with graph kernels, and they both measure the similarities in the number of vertices and properties of connectivity.

5.6 Summary and conclusion

In this chapter, we have introduced the random walk based kernels on graphs and simplicial complexes in order to measure the similarity between two data structures. For a pair of graphs given in advance, we introduce the Kronecker product to link them as one single graph called the direct product graph. Then, we apply a random walk from vertex to vertex on the direct product graph and sum up the probability of length k random walk with a given initial and stopping probability distribution for all k. The sum represents the sum of probability of simultaneous k random walk on the pair of graphs for all k, and we let it measure the similarity between the pair of graphs. For a pair of simplicial complexes, we extend the definition of direct product graph to direct product simplicial complex, and we apply the chain-valued random walk defined in Chapter 3 on it. For graphs, since the random walk is from vertex to vertex, the transition matrix of random walk is the adjacency matrix. While for simplicial complexes, since the random walk defined in Chapter 3 is chain-valued, we need to generate the state space first and then generate the transition matrix. We introduce an algorithm to generate the state space by simulation. After this, we study the computational complexity. We rewrite the sum formula to the inverse of a matrix, and use conjugate gradient methods instead of direct computation. Last but not least, we generate three data structures and compute the graph kernels and simplicial complexes kernels in practice. We figure out that graph kernels is related to the number of vertices and their connectivity, while simplicial complexes kernels put emphasis on homology properties. Therefore, if one is interested in the similarities between higher order topology data structures, the chain-valued based simplicial complexes kernels are highly recommended.

Chapter 6

Conclusions and future work

In this chapter, we summarize our main contributions and discuss future research directions.

6.1 Main contributions

This work aims at defining the random walk on simplicial complexes and studying some of its applications in homology-based hole detection and simplicial complexes kernels. The main contributions can be summarized as follows.

- **Definition of random walk on simplicial complexes**
  Different from the random walk jumping from edge to edge on simplicial complexes defined by Rosenthal [41], we define our random walk valued on chains. Giving the definition of generator of our random walk, we manage to find the link between the defined generator A and the combinatorial Laplacian L_k, that is, A, $-L_k$ and $-L_k^{\uparrow}$ coincide when restricted to space $\ker \partial_k$. Therefore, we can simulate the random walk corresponding to the combinatorial Laplacian by simulating our random walk whose generator is A when restricted on a particular space.

  Furthermore, we choose the initial state as a simple and orientable chain, and we precise each step after the initial state. We find out that the states after the initial state are always simple and orientable, implying that the Markov chain is necessarily recurrent. In addition, we prove that any two k-chains of the same length which belong to the same recurrence group, have the same stationary probability. Therefore, it is natural to expect some properties of the random walk concerning lengths of chains.

- **Random walk based hole detection**
  Given a simplicial complex with holes, we develop an algorithm to detect the locations of holes using our random walk. If we let our random walk start as a cycle which is in the same homology group with all the holes, all the states will stay in the same homology group. Therefore, given an initial state, we can locate all the holes by minimizing the length of the paths of our random walk. We propose a simulated annealing algorithm to minimize the lengths of paths, and we visualize the path with minimal length in the simplicial complex and see if it detects all the holes correctly. If we do not know the initial state in advance, we generate the initial state by computing the eigenvectors corresponding to zero eigenvalues, and we introduce the integer weighted random walk instead. Based on our simulation, we generate a Rips complex randomly, and we locate exactly the boundaries of holes with known initial state. For the case where we do not know the initial state in advance, we locate the holes approximately.

- **Random walk based simplicial complexes kernels**
  For higher order topology data, we can construct simplicial complexes to store high order structures. In extension of the definition of graph kernels, we define random walk based simplicial complexes kernels to measure the similarity between simplicial complexes in order to compare high order data structures. Given a pair of data structures, we generate a pair of simplicial complexes respectively, then we generate the direct product of them and perform the random walk on it, which presents the simultaneous random walks on these two simplicial complexes. We calculate the probability of k-length random walk on the direct product under the condition of certain initial and stopping prob-

abilities, and we sum them for all k, which is the expected similarity between simultaneous random walks at all lengths. We propose an algorithm to generate the state space of random walk, in order to calculate the probability of random walk jumping from chains to chains. To sum all the probabilities for all k, we compute the geometric random walk kernel and it amounts to computing the inverse matrix of a sparse matrix. We apply conjugate gradient method to speed up the computation. For the computation part, we define our three data structures and we compare the performances of graph kernels and simplicial complexes kernels on them. We figure out that graph kernels is related to the number of vertices and their connectivity, while simplicial complexes kernels put emphasis on information on homology properties.

6.2 Future research directions

This work mainly focuses on the definition of chain-valued random walk and some of its applications. In the future, this work can be continued with applications of the random walk as follows.

- **Commute distance for cycles**

  In machine learning, a popular tool to analyze the structure of graphs is the commute distance.

  Let $G = (V, E)$ be a graph, we wish to define a distance d between elements of V. One simple way to do so is to rely on the shortest path distance between vertices. However, such a distance does not reflect the whole graph structure. Instead, one can use the *commute distance* ([49]) between two points where for two vertices $x, y \in V$, the commute distance between x and y is equal to

  $$d(x, y) = \tau_{x,y} + \tau_{y,x},$$

  where $\tau_{x,y}$ is the expected number of jumps required for the classical random walk on the graph to move from the vertex x to the vertex y.

  The commute distance has many nice properties. It is a Euclidean distance function and can be computed in closed form. As opposed to the shortest path distance, it takes into account all paths between x and y, not just the shortest one. As a rule of thumb, the more paths connect x with y, the smaller their commute distance becomes. Hence it supposedly satisfies the following: Vertices in the same "cluster" of the graph have a small commute distance, whereas vertices in different clusters of the graph have a large commute distance to each other. Consequently, the commute distance is considered a convenient tool to encode the cluster structure of the graph.

One can generalize this distance to distance between cycles, that is elements of $\mathcal{C}_k$ for $k > 0$ by taking

$$d(\sigma_1, \sigma_2) = \tau_{\sigma_1,\sigma_2} + \tau_{\sigma_2,\sigma_1},$$

where τ_{σ_1,σ_2} is the expected number of jumps required for the random walk on the cycles defined in Chapter 3 to move from the cycle σ_1 to the cycle σ_2. We can expect that the commute distances for cycles provide more information in higher order clustering in networks.

Previously, we compute the simplicial complexes kernels by generating the transition matrix of random walk. Although the transition matrix is sparse, the dimension of the matrix is $N \times N$ where N is the number of paths in the homology class, which can be very large when the simplicial complexes are large. Instead of generating the transition matrix and solving the linear system

$$(I - \lambda W_\times)x = p_\times,$$

we can compute the simplicial complexes kernel by making a simulation of random walk. By (5.4), we have

$$k(\mathcal{C}, \mathcal{C}') = \sum_{k=0}^{\infty} \lambda^k q_\times^\top W_\times^k p_\times.$$

In absence of prior knowledge, we let $p_\times$ and $q_\times$ be uniformly distributed. We can interpret the formula of simplicial complexes kernels as the sum of probabilities of k-length random walks from a certain state to a certain state for all k. Therefore, we can simulate a random walk initiated with c_0 so that

$$c_{t+1} = \begin{cases} rw(c_t) & \text{with probability of } \frac{1}{N} \\ c_0 & \text{with probability of } 1 - \frac{1}{N}. \end{cases}$$

We simulate L random walks which run M steps and make sure L and M are large enough, and the probability of the final states of these L random walks should be the simplicial complexes kernels.

What is more, in this work we generate a random Rips complex at first and let the random walk run on it to get information on the Rips complex. However, in the real world, sometimes we do not know the whole Rips complex in the first place. Instead, we know more and more points with time passing by. Therefore, it is more suitable to model the Rips complex "dynamically". We can continue our work in modeling the random walk on a dynamic-generated simplicial complex in the future.

Appendix A

Computation of convergence of random walk

1.1 Computation of convergence of random walk on torus

Here we give the detailed computation of convergence of random walk on torus in Theorem 18.

For a flat torus $\mathbb{T}_2 := \mathbf{R}^2/\mathbf{Z}^2$, we follow the notations in Figure 1.1.

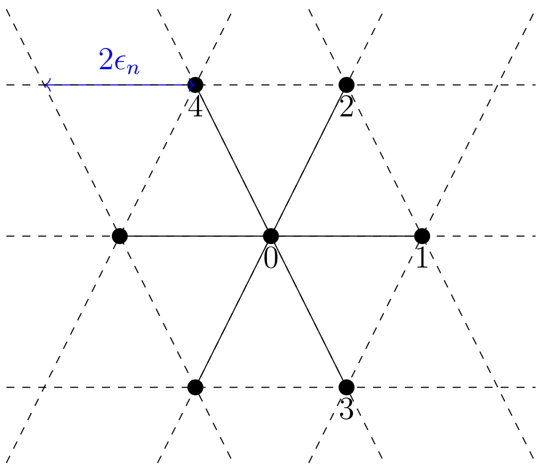


Figure 1.1: *Regular triangulation of the flat torus.* $0 := (0,0),\ 1 := (2\epsilon_n, 0),\ 2 := (\epsilon_n, \sqrt{3}\epsilon_n),\ 3 := (\epsilon_n, -\sqrt{3}\epsilon_n), 4 := (-\epsilon_n, \sqrt{3}\,\epsilon_n)$

By Eqn. (3.12), we have

$$\begin{aligned}\hat{A}_n f_n(x) &= \sum_{a\in\{-1,1\}} \Big(f_n(x+(2a\epsilon_n,0)) - f_n(x)\Big)\\ &\quad + \sum_{a,b\in\{-1,1\}} \Big(f_n(x+(a\,\epsilon_n, b\sqrt{3}\,\epsilon_n)) - f_n(x)\Big)\\ &=: \ X+Y.\end{aligned}$$

For the first part X, by Taylor expansion of f_n at x, we have

$$\begin{aligned}X &= \sum_{a\in\{-1,1\}} \Big(2a\epsilon_n \frac{\partial f_n}{\partial x_1}(x) + 2a^2\epsilon_n^2 \frac{\partial^2 f_n}{\partial x_1^2}(x) + o(\epsilon_n^2)\Big)\\ &= 4\epsilon_n^2 \frac{\partial^2 f_n}{\partial x_1^2}(x) + o(\epsilon_n^2).\end{aligned}$$

For the second part Y, we have

$$\begin{aligned}Y = &\sum_{a,b\in\{-1,1\}} \Big(a\epsilon_n \frac{\partial f_n}{\partial x_1}(x) + b\sqrt{3}\epsilon_n \frac{\partial f_n}{\partial x_2}(x) + \frac{1}{2}a^2\epsilon_n^2 \frac{\partial^2 f_n}{\partial x_1^2}(x)\\ &\quad + \frac{3}{2}b^2\epsilon_n^2 \frac{\partial^2 f_n}{\partial x_2^2}(x) + \sqrt{3}ab\epsilon_n^2 \frac{\partial^2 f_n}{\partial x_1 \partial x_2}(x) + o(\epsilon_n^2)\Big)\\ =&2\epsilon_n^2 \frac{\partial^2 f_n}{\partial x_1^2}(x) + 6\epsilon_n^2 \frac{\partial^2 f_n}{\partial x_2^2}(x) + o(\epsilon_n^2).\end{aligned}$$

Since $A_n = \epsilon_n^{-2}\hat{A}_n$, we can get

$$A_n f_n(x) = 6\ \frac{\partial^2 f_n}{\partial x_1^2}(x) + 6\ \frac{\partial^2 f_n}{\partial x_2^2}(x) + o(\epsilon_n^2).$$

As a consequence, we have

$$A_n f_n \xrightarrow{n\to\infty} Af := 6\ \frac{\partial^2 f}{\partial x_1^2} + 6\ \frac{\partial^2 f}{\partial x_2^2}.$$

1.2 Computation of convergence of random walk in Lemma 20

In order to get the limit of

$$\int_{[021]\circlearrowright} H(x_1,x_2)\ \mathrm{d}x_1\ \mathrm{d}x_2 + \int_{[031]\circlearrowleft} H(x_1,x_2)\ \mathrm{d}x_1\ \mathrm{d}x_2$$

as ϵ_n goes to 0, we have

$$\begin{aligned}M_1 &:= \int_{[021]\circlearrowleft} H(x_1,x_2)\ \mathrm{d}x_1\ \mathrm{d}x_2 \\ &= -\int_0^{\epsilon_n}\int_0^{x_1\sqrt{3}} H(x_1,x_2)\ \mathrm{d}x_1\ \mathrm{d}x_2 - \int_{\epsilon_n}^{2\epsilon_n}\int_0^{\sqrt{3}(2\epsilon_n - x_1)} H(x_1,x_2)\ \mathrm{d}x_1\ \mathrm{d}x_2,\end{aligned}$$

and

$$\begin{aligned}M_2 &:= \int_{[031]\circlearrowleft} H(x_1,x_2)\ \mathrm{d}x_1\ \mathrm{d}x_2 \\ &= \int_0^{\epsilon_n}\int_{-x_1\sqrt{3}}^{0} H(x_1,x_2)\ \mathrm{d}x_1\ \mathrm{d}x_2 + \int_{\epsilon_n}^{2\epsilon_n}\int_{-\sqrt{3}(2\epsilon_n - x_1)}^{0} H(x_1,x_2)\ \mathrm{d}x_1\ \mathrm{d}x_2.\end{aligned} \tag{A.1}$$

By Taylor expansion of H at $(x_1, 0)$, we have

$$H(x_1,x_2) = H(x_1,0) + x_2\, H_2(x_1,0) + {x_2}^2\, r(x_1,x_2) \tag{A.2}$$

with $\sup_{x_1,x_2\in[0,1]^2} r(x_1,x_2) < \infty$. We denote by $H_2(x_1,0)$ the first order derivative with respect to the second variable x_2 at $(x_1,0)$.

Substituting A.2 into A.1, we have

$$\begin{aligned}M_1 + M_2 = &\int_0^{\epsilon_n} H_2(x_1,0)\left(-\int_0^{x_1\sqrt{3}} x_2\ \mathrm{d}x_2 + \int_{-x_1\sqrt{3}}^{0} x_2\ \mathrm{d}x_2\right)\ \mathrm{d}x_1 \\ &+ \int_{\epsilon_n}^{2\epsilon_n} H_2(x_1,0)\left(-\int_0^{(2\epsilon_n - x_1)\sqrt{3}} x_2\ \mathrm{d}x_2 + \int_{-(2\epsilon_n - x_1)\sqrt{3}}^{0} x_2\ \mathrm{d}x_2\right)\ \mathrm{d}x_1 + R_n, \\ = &-3\int_0^{\epsilon_n} H_2(x_1,0){x_1}^2\ \mathrm{d}x_1 - 3\int_{\epsilon_n}^{2\epsilon_n} H_2(x_1,0)(2\epsilon_n - x_1)^2\ \mathrm{d}x_1 + R_n.\end{aligned} \tag{A.3}$$

By Taylor expansion of $H_2(x_1, 0)$ at $(0, 0)$, we have

$$H_2(x_1,0) = H_2(0,0) + x_2 r(x_1,x_2) \tag{A.4}$$

with $\sup_{x_1,x_2\in[0,1]^2} r(x_1,x_2) < \infty$.

Substituting A.4 into A.3, we have

$$\begin{aligned}M_1 + M_2 &= -3H_2(0,0)\int_0^{\epsilon_n} x_1^2\ \mathrm{d}x_1 - 3H_2(0,0)\int_{\epsilon_n}^{2\epsilon_n}(2\epsilon_n - x_1)^2\ \mathrm{d}x_1 \\ &= -2H_2(0,0)\epsilon_n^3 + O(\epsilon_n^4) \\ &= -\epsilon_n^2\int_0^{2\epsilon_n} H_2(x_1,0)\ \mathrm{d}x_1 + O(\epsilon_n^4),\end{aligned}$$

which gives

$$\int_{[021]\circlearrowright} H(x_1, x_2)\ \mathrm{d}x_1\ \mathrm{d}x_2 + \int_{[031]\circlearrowleft} H(x_1, x_2)\ \mathrm{d}x_1\ \mathrm{d}x_2$$
$$= -\epsilon_n^2 \int_0^{2\epsilon_n} H_2(x_1, 0)\ \mathrm{d}x_1 + O(\epsilon_n^4).$$

Bibliography

[1] G. Akinwande and M. Reitzner. Multivariate central limit theorems for random simplicial complexes. arXiv:1912.00975, 2019.

[2] M.A. Armstrong. *Basic topology.* Springer, New-York, 1983.

[3] Jon L Bentley and Robert Sedgewick. Fast algorithms for sorting and searching strings. In *Proceedings of the eighth annual ACM-SIAM symposium on Discrete algorithms*, pages 360–369, 1997.

[4] Dimitris Bertsimas, John Tsitsiklis, et al. Simulated annealing. *Statistical science*, 8(1):10–15, 1993.

[5] J.-D. Boissonnat, F. Chazal, and M. Yvinec. *Geometric and Topological Inference.* Cambridge University Press, 2018.

[6] Jean-Daniel Boissonnat, Frédéric Chazal, and Mariette Yvinec. *Geometric and topological inference*, volume 57. Cambridge University Press, 2018.

[7] Jean-Daniel Boissonnat, Ramsay Dyer, and Arijit Ghosh. A probabilistic approach to reducing algebraic complexity of delaunay triangulations. In *Algorithms-ESA 2015*, pages 595–606. Springer, 2015.

[8] Jean-Daniel Boissonnat and Clément Maria. The simplex tree: An efficient data structure for general simplicial complexes. *Algorithmica*, 70(3):406–427, 2014.

[9] Karsten M Borgwardt, Cheng Soon Ong, Stefan Schönauer, SVN Vishwanathan, Alex J Smola, and Hans-Peter Kriegel. Protein function prediction via graph kernels. *Bioinformatics*, 21(suppl_1):i47–i56, 2005.

[10] Olivier Catoni. Rough large deviation estimates for simulated annealing: Application to exponential schedules. *The Annals of Probability*, pages 1109–1146, 1992.

[11] S. Clémençon, H. De Arazoza, F. Rossi, and V.C. Tran. A statistical network analysis of the hiv/aids epidemics in cuba. *Social Network Analysis and Mining*, 5:Art.58, 2015.

[12] Vin De Silva and Gunnar E Carlsson. Topological estimation using witness complexes. *SPBG*, 4:157–166, 2004.

[13] L. Decreusefond, E. Ferraz, H. Randriam, and A. Vergne. Simplicial homology of random configurations. *Adv. in Appl. Probab.*, 46:1–20, 2014.

[14] K. Devriendt and P. Van Mieghem. The simplex geometry of graphs. *Journal of Complex Networks*, 7(4):469–490, 2019.

[15] B. Eckmann. Harmonische funktionen und randwertaufgaben in einem komplex. *Comm. Math. Helv.*, 17(1):240–255, 1944.

[16] Herbert Edelsbrunner. The union of balls and its dual shape. In *Proceedings of the ninth annual symposium on Computational geometry*, pages 218–231, 1993.

[17] Herbert Edelsbrunner, Michael Facello, and Jie Liang. On the definition and the construction of pockets in macromolecules. *Discrete Applied Mathematics*, 88(1):83–102, 1998.

[18] Herbert Edelsbrunner, David Kirkpatrick, and Raimund Seidel. On the shape of a set of points in the plane. *IEEE Transactions on information theory*, 29(4):551–559, 1983.

[19] Herbert Edelsbrunner and Ernst P Mücke. Three-dimensional alpha shapes. *ACM Transactions on Graphics (TOG)*, 13(1):43–72, 1994.

[20] S.N. Ethier and T.G. Kurtz. *Markov Processus, Characterization and Convergence*. John Wiley & Sons, New York, 1986.

[21] R. Forman. Combinatorial differential topology and geometry. In *New perspectives in algebraic combinatorics (Berkeley, CA, 1996–97)*, volume 38 of *Math. Sci. Res. Inst. Publ.*, pages 177–206. Cambridge Univ. Press, Cambridge, 1999.

[22] S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3-5):75–174, 2010.

[23] Thomas Gärtner, Peter Flach, and Stefan Wrobel. On graph kernels: Hardness results and efficient alternatives. In *Learning theory and kernel machines*, pages 129–143. Springer, 2003.

[24] J. Grote, Z. Kabluchko, and C. Thäle. Limit theorems for random simplices in high dimensions. *ALEA, Lat. Am. J. Probab. Math. Stat.*, 16:141–177, 2019.

[25] A. Hatcher. *Algebraic topology.* Cambridge University Press, 2002.

[26] W Imrich and Sandi Klavzar. *Product graphs: structure and recognition.* Discrete mathematics and optimization. Wiley, New York, NY, 2000.

[27] Tomasz Kaczyński, Marian Mrozek, and Maciej Ślusarek. Homology computation by reduction of chain complexes. *Computers & Mathematics with Applications*, 35(4):59–70, 1998.

[28] M. Kahle. Topology of random clique complexes. *Discrete Mathematics*, 309:1658–1671, 2009.

[29] M. Kahle and E. Meckes. Limit theorems for betti numbers of random simplicial complexes. *Homology, Homotopy and Applications*, 15:343–374, 2013. Erratum: 18 (2016), 129–142.

[30] Matthew Kahle. Random geometric complexes. *Discrete & Computational Geometry*, 45(3):553–573, 2011.

[31] Ngoc Khuyen Le. *Simplicial homology: applied to wireless networks.* PhD thesis, 2016.

[32] L. Lovàsz. Random walks on graphs: a survey. *Combinatorics*, 2:353–398, 1996. in: Paul Erdös is Eighty.

[33] Martin F Møller. *A scaled conjugate gradient algorithm for fast supervised learning.* Aarhus University, Computer Science Department, 1990.

[34] A. Muhammad and M. Egerstedt. Control Using Higher Order Laplacians in Network Topologies. In *Proceedings of the 17th International Symposium on Mathematical Theory of Networks and Systems, Kyoto, Japan*, July 2006.

[35] Abubakr Muhammad and Magnus Egerstedt. Control using higher order laplacians in network topologies. In *Proc. of 17th International Symposium on Mathematical Theory of Networks and Systems*, pages 1024–1038. Citeseer, 2006.

[36] S. Mukherjee and J. Steenbergen. Random walks on simplicial complexes and harmonics. *Random Structures & Algorithms*, 48(2):379–405, 2016.

[37] J.R. Munkres. *Elements of algebraic topology.* Addison-Wesley Publishing Company, Menlo Park, CA, 1984.

[38] A. Noack. Modularity clustering is force-directed layout. *Physical Review E*, 79:026102, 2009.

[39] T. Owada, G. Samorodnitsky, and G. Thoppe. Limit theorems for topological invariants of the dynamic multi-parameter simplicial complex. arXiv:2001.06860v1, 2020.

[40] O. Parzanchevski and R. Rosenthal. Simplicial complexes: spectrum, homology and random walks. *Random Structures & Algorithms*, 50(2):225–261, 2017.

[41] R. Rosenthal. Simplicial branching random walks and their applications. `arXiv:1412.5406`, 2014.

[42] S.E. Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27–64, 2007.

[43] Jonathan Richard Shewchuk et al. An introduction to the conjugate gradient method without the agonizing pain, 1994.

[44] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905, August 2000.

[45] Alex J Smola and Bernhard Schölkopf. *Learning with kernels*, volume 4. Citeseer, 1998.

[46] Alain Trouve. Cycle decompositions and simulated annealing. *SIAM Journal on Control and Optimization*, 34(3):966–986, 1996.

[47] S Vichy N Vishwanathan, Nicol N Schraudolph, Risi Kondor, and Karsten M Borgwardt. Graph kernels. *Journal of Machine Learning Research*, 11(Apr):1201–1242, 2010.

[48] U. von Luxburg, M. Belkin, and O. Bousquet. Consistency of spectral clustering. *Annals of Statistics*, 36(2):555–586, April 2008.

[49] Ulrike Von Luxburg, Agnes Radl, and Matthias Hein. Hitting and commute times in large random neighborhood graphs. *The Journal of Machine Learning Research*, 15(1):1751–1798, 2014.

[50] D. Yogeshwaran and R.J. Adler. On the topology of random complexes built over stationary point processes. *The Annals of Applied Probability*, 25(6):3338–3380, 2015.

[51] V. Zobel. Spectral analysis of the hodge laplacian on discrete manifolds. Master's thesis, Technische Universität Berlin (J. M. Sullivan) and ZIB (I. Hotz), 2010.

ÉCOLE DOCTORALE
de mathématiques
Hadamard (EDMH)

Titre: Marche aléatoire sur des complexes simpliciaux

Mots clés: Complexes simpliciaux, marche aléatoire, homologie simpliciale, Laplacien combinatoire

Résumé: La notion de laplacien d'un graphe peut être généralisée aux complexes simpliciaux et aux hypergraphes. Cette notion contient des informations sur la topologie de ces structures. Dans la première partie de cette thèse, nous définissons une nouvelle chaîne de Markov sur les complexes simpliciaux. Pour un degré donné k de simplexes, l'espace d'états n'est pas les k-simplexes comme dans les articles précédents sur ce sujet mais plutôt l'ensemble des k-chaines ou k-co-chaines. Ce nouveau cadre est la généralisation naturelle sur les chaînes de Markov canoniques sur des graphes. Nous montrons que le générateur de notre chaîne de Markov est lié au Laplacien supérieur défini dans le contexte de la topologie algébrique pour structure discrète. Nous établissons plusieurs propriétés clés de ce nouveau procédé. Nous montrons que lorsque les complexes simpliciaux examinés sont une séquence de triangulation du tore plat, les chaînes de Markov tendent vers une forme différentielle valorisée processus continu.

Dans la deuxième partie de cette thèse, nous explorons quelques applications de la marche aléatoire, i.e. la détection de trous basée sur la marche aléatoire et les noyaux complexes simpliciaux. Pour la détection de trous basée sur la marche aléatoire, nous introduisons un algorithme pour faire des simulations effectuées pour la marche aléatoire à valeur cyclique ($k = 1$) sur un complexe simplicial avec trous. Pour les noyaux de complexes simpliciaux, nous étendons la définition des noyaux de graphes basés sur la marche aléatoire afin de mesurer la similitude entre deux complexes simpliciaux.

Title: Random walk on simplicial complexes

Keywords: Simplicial complexes, random walk, simplicial homology, combinatorial Laplacian

Abstract: The notion of Laplacian of a graph can be generalized to simplicial complexes and hypergraphs. This notion contains information on the topology of these structures. In the first part of this thesis, we define a new Markov chain on simplicial complexes. For a given degree k of simplices, the state space is not the k-simplices as in previous papers about this subject but rather the set of k-chains or k-cochains. This new framework is the natural generalization on the canonical Markov chains on graphs. We show that the generator of our Markov chain is related to the upper Laplacian defined in the context of algebraic topology for discrete structure. We establish several key properties of this new process. We show that when the simplicial complexes under scrutiny are a sequence of ever refining triangulation of the flat torus, the Markov chains tend to a differential form valued continuous process.

In the second part of this thesis, we explore some applications of the random walk, i.e., random walk based hole detection and simplicial complexes kernels. For the random walk based hole detection, we introduce an algorithm to make simulations carried for the cycle-valued random walk ($k = 1$) on a simplicial complex with holes. For the simplicial complexes kernels, we extend the definition of random walk based graph kernels in order to measure the similarity between two simplicial complexes.

Université Paris-Saclay
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France