Modèles de graphes aléatoires avec séquences de degrés et de centralité fixes

Mémoire

François Thibault

Maîtrise en physique - avec mémoire
Maître ès sciences (M. Sc.)

Québec, Canada

© François Thibault, 2022

Modèles de graphes aléatoires avec séquences de degrés et de centralité fixes

Mémoire

François Thibault

Sous la direction de:

Antoine Allard, directeur de recherche

Résumé

Les réseaux complexes sont un paradigme de plus en plus utilisé pour modéliser la structure des systèmes complexes. Les connexions neuronales ou la transmission de maladies entre différents humains sont des exemples typiques de ce genre de structure. L'un des défis de la science des réseaux est de fournir des outils permettant de les analyser, notamment par le développement de modèles aléatoires de réseaux. Ces modèles permettent de supposer un hasard sous-jacent dans la construction d'un ensemble de réseaux, tout en conservant certaines propriétés importantes. En comparant un réseau mesuré dans un système réel à ceux issus d'un ensemble généré à l'aide d'un modèle aléatoire, il est possible d'identifier des propriétés ne pouvant pas s'expliquer par le hasard et ainsi mettre en évidence la présence de processus de formation cachés.

Ce projet de maîtrise vise à développer une famille de modèles aléatoires qui se base sur une propriété de centralité des noeuds nommée la décomposition en oignon. Des algorithmes sont développés afin d'échantillonner des réseaux issus de différents ensembles. On montre que ces algorithmes peuvent construire les échantillons sans aucun biais dans le cas où les contraintes sont conservées exactement, et que les échantillons construits sont représentatifs des ensembles dans le cas où les contraintes sont conservées en moyenne sur l'ensemble. Finalement, on compare les nouveaux ensembles développés avec des ensembles déjà existants afin d'obtenir une nouvelle intuition sur le rôle que joue l'organisation à moyenne échelle sur les propriétés des réseaux complexes réels.

Abstract

Complex networks are recent tools with a growing popularity that are used to study the structure of complex systems. Examples of these structures are the connections between neurons or the transmission of diseases along social ties in a population, both represented as links between nodes. A major challenge in network science is to develop tools that allow to understand these complex structures. Among these tools is the use of random graph models, which allow us to build ensembles of networks that share a common property while also having an underlying randomness. By comparing a network obtained from real data to a random graph model, it is possible to identify certain properties that cannot be explained by randomness, thereby highlighting the existence of some hidden formation process.

This project aims to develop a family of random graph models that are based on a centrality property called the Onion Decomposition. Algorithms to create representative samples of these models are proposed. We show that the algorithms build the sample with no bias in the case of exact constraints, or with the proper bias in the case where the constraints are kept on average. Finally, we compare the new ensembles to ensembles in the literature to obtain a better intuition on the role of meso-scale organization in real complex networks.

Table des matières

Résumé **ii**

Abstract **iii**

Table des matières **iv**

Liste des figures **v**

Liste des abréviations **vi**

Remerciements **vii**

Introduction **1**

1 Notions préliminaires **4**
1.1 Graphes 4
1.2 Précisions sur la décomposition en oignon 9
1.3 Ensembles de graphes aléatoires 11
1.4 Modèle des configurations 14
1.5 Chaînes de Markov 15
1.6 Échantillonnage du modèle des configurations 32
1.7 Balance détaillée et Metropolis-Hastings 37
1.8 Modèle des configurations corrélées 41

2 Modèles de graphes avec séquence de centralité fixe **45**
2.1 Modèle des configurations étagées 45
2.2 Modèles des configurations étagées et corrélées 55
2.3 Librairie C++ et résultats numériques sur la convergence 58
2.4 Application sur des réseaux réels 63

Conclusion **71**

A Test d'échange pour la décomposition en oignon - version colorée **74**

B Piste de preuve alternative pour l'apériodicité du LCM **77**

C Réductibilité du LCCM souple **79**

Bibliographie **82**

Liste des figures

1.1 Exemple de graphe orienté et non orienté 5
1.2 Exemple de pseudo-graphe, multigraphe et demi-liens 5
1.3 Exemple de période dans un graphe 6
1.4 Décomposition en oignon sur un petit graphe 11
1.5 Illustration de marche aléatoire 21
1.6 Illustration d'échanges de liens 34

2.1 Graphe de graphes non connecté pour le LCM 53
2.2 Erreur quadratique moyenne entre la distribution uniforme et la distribution des algorithmes en fonction de la taille de l'échantillon 60
2.3 Erreur quadratique moyenne entre la matrice de corrélation du graphe initial et la moyenne des matrices de corrélations sur les différents ensembles en fonction de la taille de l'échantillon 61
2.4 Matrices de couches jointes du *C. elegans* et de la moyenne de son LCCM CxC, et erreur quadratique moyenne entre les deux matrices 62
2.5 Distribution d'assortativité sur des échantillons des ensembles 65
2.6 Distribution du coefficient d'agrégation sur des échantillons des ensembles . . . 67
2.7 Nombre de noeuds infectés en fonction du temps de simulation pour une dynamique SIS sur les différents ensembles 69

B.1 Exemples de 3-cycles dans le LCM 78

C.1 Exemple de non connectivité des ensembles à contraintes souples 80

Liste des abréviations

Liste des abréviations

OD	Décomposition en oignon	***O**nion **D**ecomposition*
CM	Modèle des configurations	***C**onfiguration **M**odel*
MCMC	Chaîne de Markov Monte Carlo	***M**arkov **C**hain **M**onte **C**arlo*
CCM	Modèle des configurations corrélé	***C**orrelated **C**onfiguration **M**odel*
LCM	Modèles des configurations étagé	***L**ayered **C**onfiguration **M**odel*
LCCM	Modèle des configurations corrélé et étagé	***L**ayered and **C**orrelated **C**onfiguration **M**odel*
LCCM **DxD**	LCCM avec corrélations en degrés	
LCCM **CxC**	LCCM avec corrélations en couches	
LCCM **CDxCD**	LCCM avec corrélations en (degré,couche)	

Remerciements

Il me semble que j'ai le bonheur de côtoyer une multitude de personnes formidables qui m'ont toutes aidé d'une manière plus ou moins direct dans l'élucubration de ce mémoire, et qui méritent toutes une part dans cette section des remerciements. Comme il serait impossible de toutes les nommer, voici une liste non exhaustive de ces personnes :

Je tiens d'abord à remercier mon directeur de recherche Antoine pour son aide précieuse est ses conseils toujours judicieux. Je suis extrêmement heureux d'avoir un guide aussi passionné, compréhensif et rigoureux, et ses encouragements m'ont permis et me permettront toujours d'aller de l'avant dans mes projets.

Je tiens aussi à remercier l'entièreté du groupe Dynamica, en commençant par Patrick pour sa passion et son amour des mathématiques qui resteront toujours une inspiration majeure. Je remercie les membres senior Charles et Vincent pour nos discussions fascinantes et pour l'aide qu'ils ont toujours été heureux d'apporter lorsque j'avais des questions. J'ai bien hâte de collaborer à nouveau avec vous. Je tiens à faire un merci tout spécial à Béatrice, collègue et amie que j'admire depuis mon tout premier Bourbaki, et qui sait toujours rendre le travail plus agréable. Finalement, je tiens à remercier mon compagnon quasi-quotidien depuis le début de la maîtrise, Simon. Ta passion débordante est toujours bienvenue, et c'est un réel plaisir de pouvoir travailler ou procrastiner à tes côtés.

Au-delà de mon entourage dans la recherche, je tiens à remercier tous mes amis, physiciens ou non. Les amis que je me suis fait au bac et qui continuent d'être à mes côtés sont une source infinie de bonheur, de bons moments et d'inspiration. Je tiens donc à remercier les Shitposteux, Miche, Justine, Colin et Nicolas. Du côté des non-physiciens, je tiens particulièrement à remercier mon ami de longue date Édouard, ainsi que Maria et Roxanne, avec qui je passe toujours d'incroyables moments.

Finalement, je tiens à remercier toute ma famille pour le soutien inconditionnel que je reçois depuis que je suis tout jeune. Je remercie d'abord mes parents pour toute leur aide précieuse et pour avoir formé la personne que je suis maintenant ; je remercie également mes deux frères, mes grand-parents qui sont toujours si fiers quand je leur explique mes travaux, ainsi que mes oncles, tantes et cousines.

Introduction

La science des réseaux s'inspire principalement de deux paradigmes récents : la science de la complexité [4, 91] et la science des données [9]. Le paradigme de la complexité s'inscrit à mi-chemin entre le paradigme des systèmes *réduits* de Newton et le paradigme de la *complexité désorganisée* de la physique statistique de Boltzmann. Les études dans la science de la complexité portent sur les *systèmes complexes*, qui sont généralement compris comme étant des systèmes possédant un grand nombre de variables organisées selon une structure qui influence fortement le comportement dudit système. Les exemples de systèmes complexes sont nombreux : on y trouve notamment le cerveau, la propagation de maladies dans une population, le système financier, les interactions au sein d'un écosystème ou même encore les colonies d'insectes sociaux. De nombreuses méthodes ont été développées dans le dernier siècle pour étudier ces systèmes, notamment la théorie du chaos déterministe, l'étude des fractales, l'étude de l'instabilité ou encore la brisure de symétrie [5]. Malgré certains succès de ces approches théoriques, les systèmes complexes sont restés globalement incompris et la nécessité de créer de nouveaux outils s'est rapidement fait sentir.

C'est dans ce contexte qu'est arrivée la science des réseaux. Celle-ci se base sur les *réseaux complexes*, qui sont des outils issus de la théorie des graphes, pour modéliser toutes les interactions entre les variables d'un système complexe. Cette nouvelle approche se distingue des précédentes par le fait qu'elle se base sur l'analyse de données, et forme ainsi un outil plus cohérent avec l'idée d'une science qui se veut empirique [9]. En science des réseaux donc, on cherchera à analyser un système en créant un *réseau*, c'est-à-dire un ensemble de *noeuds* reliés par des *liens*, à partir de données réelles. Le réseau sera créé en considérant les interactions (liens) entre les différentes composantes (noeuds) du système à l'étude. Le projet présenté dans ce mémoire s'inscrit dans l'analyse à faire une fois que ce réseau est créé.

De nombreuses mesures sur réseaux existent pour les caractériser [68]. On peut penser notamment à l'assortativité, qui indique la propension des noeuds de hauts degrés à se connecter à d'autres noeuds de hauts degrés, ou encore au coefficient d'agglomération qui indique la proportion de connections en triangle. Cependant, l'interprétation d'une valeur donnée pour une mesure nécessite généralement une comparaison à d'autres valeurs mesurées sur des graphes similaires. Plus précisément, pour obtenir une conclusion valide d'un point de vue statistique

à partir de ces mesures, il faut être capable de les comparer à des hypothèses nulles [33]. C'est ici qu'interviennent les *modèles de graphes aléatoires*, outils qui seront le sujet principal de ce mémoire. Ceux-ci seront définis formellement dans le document, mais il suffit pour l'instant de savoir que les modèles de graphes aléatoires sont des ensembles de graphes basés sur des contraintes, ces dernières étant les fondations d'une hypothèse nulle à laquelle on voudrait comparer un réseau issu de données réelles. Cette comparaison statistique permettra alors de déterminer à quel point les contraintes peuvent expliquer les mesures observées sur le réseau. Cette technique est utilisée dans de nombreux domaines, notamment pour le repérage de motifs (sous-graphes) dans des réseaux métaboliques [48, 77]. L'idée dans ce cas est de comparer les motifs du réseau réel aux motifs trouvés lorsque l'on mélange aléatoirement le réseau. Des analyses statistiques similaires interviennent aussi en sociologie [27, 63], en écologie [20, 93], pour l'étude du réseau Internet [58], l'étude de réseaux trophiques [83] ou l'étude de trajectoires académiques [57] et finalement, pour l'étude de dynamiques de propagation [16, 81, 89].

Certains modèles de graphes aléatoires ont d'abord été résolus analytiquement. C'est le cas pour les premiers modèles aléatoires qui imposent des contraintes sur le nombre de liens [30, 38], ou encore pour des ensembles avec une distribution de degrés imposée [66, 69]. Une famille de modèles, les *Exponential Random Graph Models*, a été développée pour tenir compte d'une contrainte arbitraire sur un ensemble de graphes [36, 44]. Cette famille de modèles analytiques reste très utilisée, notamment dans le domaine des sciences humaines qui l'a grandement approfondie [84, 88] et généralisée pour des réseaux évolutifs [8, 74, 80].

D'autres modèles de graphes importants ont plutôt été définis par des algorithmes de croissance. On peut penser notamment au modèle « petit-monde » de Watts-Strogatz [90], au modèle « d'attachement préférentiel » de Barabási-Albert [10] ou au modèle généré par le principe de duplication [18, 72, 87], les deux derniers permettant de générer des réseaux invariants d'échelle. Plus récemment, un modèle de croissance préservant la séquence de degrés a aussi été développé [50].

La plupart de ces modèles ont fait l'objet d'études analytiques qui permettent de bien comprendre les caractéristiques principales des ensembles. Cependant, pour faire une comparaison numérique entre le réseau réel et l'hypothèse nulle, il est nécessaire d'obtenir un échantillon de réseaux qui soit représentatif de cette dernière. Les algorithmes de croissance permettent déjà d'extraire des graphes de leur modèle associé, mais ils sont assez limités dans les contraintes qu'ils peuvent conserver. Ainsi, le développement d'*algorithmes d'échantillonnages* occupe une place importante dans l'étude des modèles de graphes aléatoires [23], puisqu'ils permettent de créer des données synthétiques qui représentent les hypothèses nulles à tester. Ces algorithmes doivent cependant faire l'objet de vérifications approfondies pour s'assurer de leur bon fonctionnement [15, 35, 79].

Un modèle important pour ce mémoire est le *modèle des configurations*, qui est un modèle avec une contrainte sur la séquence de degrés. De récentes réévaluations de ce modèle [15, 22, 35] ont permis le développement d'un formalisme pour s'assurer que les algorithmes d'échantillonnage convergent vers la bonne distribution de réseaux échantillonnés. Le projet présenté dans ce mémoire s'inspire de ce formalisme pour développer une nouvelle panoplie de modèles aléatoires de graphes qui ajoutent des contraintes sur le modèle des configurations, puis pour ensuite développer les algorithmes pour échantillonner ces ensembles. Les algorithmes seront par la suite évalués pour s'assurer de leur convergence, puis seront appliqués sur des données réelles.

Le mémoire est divisé en deux chapitres. Le chapitre 1 contient une introduction sur tous les sujets abordés dans le projet. Au-delà d'un simple rappel, les notions vues dans ce chapitre seront essentielles pour bien comprendre le projet ; certaines preuves mathématiques pour le projet seront basées sur des preuves présentées dans ce chapitre. Le chapitre 2 contiendra donc l'essentiel du projet de maîtrise. Les nouveaux ensembles seront définis dans ce chapitre, de même que les algorithmes d'échantillonnages développés. Finalement, les résultats des tests de la librairie d'algorithmes seront présentés dans ce chapitre, de même que les résultats d'application des modèles sur des réseaux réels.

Chapitre 1

Notions préliminaires

Ce chapitre se veut une présentation des concepts essentiels à la compréhension du projet et de ses contributions originales, et présente des références permettant d'approfondir les sujets. Les théorèmes présentés montrent des résultats connus, mais ceux-ci ont été remaniés pour s'appliquer directement au problème à l'étude.

1.1 Graphes

Le graphe étant l'outil de base pour le projet, il est pertinent d'offrir une courte définition afin d'introduire la notation utilisée dans ce document. Un *graphe non orienté* $G(V, E)$ est composé d'un ensemble V de *noeuds* $v \in V$, ainsi que d'un ensemble E de *liens* $e \in E$. Le nombre de noeuds est noté $n := |V|$ et le nombre de liens est noté $m := |E|$. Un lien est dénoté comme étant une paire de noeuds, soit $e = (u, v)$ pour $u, v \in V$. Si un lien (u, v) existe, on dit que les noeuds u et v sont *connectés*, ou *voisins*. Il est souvent utile d'ordonner les noeuds, et donc de les classer selon une *étiquette*, par exemple en décrivant un noeud avec la notation v_i pour $i \in \{1, \ldots, n\}$.

On peut aussi donner une orientation aux liens dans un graphe. Pour les *graphes orientés*, on note les liens comme $(u \to v)$. Si un tel lien existe dans l'ensemble E, on dira que le lien pointe du noeud u vers le noeud v. Un exemple de graphe orienté et graphe non orienté est illustré à la figure 1.1.

Pour les graphes non orientés, le *degré* $d(v)$ d'un noeud correspond à son nombre de voisins. On définit la *séquence de degrés* $\mathbf{d} = \{d_1, d_2, \ldots, d_n\}$ d'un graphe comme étant une séquence contenant le degrés de tous les noeuds $v_i \in V$. On dit d'un graphe qu'il est *régulier* si tous les degrés sont les mêmes, donc si $d_1 = d_2 = \ldots = d_n$.

De la même manière, pour les graphes orientés, on définit le *degré sortant* $d_{\text{out}}(u)$ comme étant le nombre de liens qui partent du noeud u et le *degré entrant* $d_{\text{in}}(u)$ comme étant le nombre de liens qui pointent vers ce noeud. On utilisera alors la séquence de degrés sortants $\mathbf{d}_{\text{out}}$ et

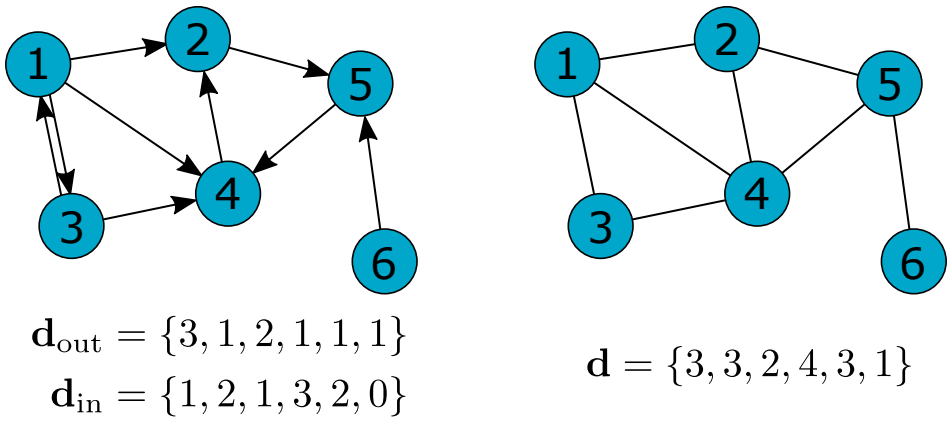

FIGURE 1.1 – Exemple de graphe orienté (gauche) et de graphe non orienté (droite). Les séquences de degrés sont indiquées en dessous de chaque graphe.

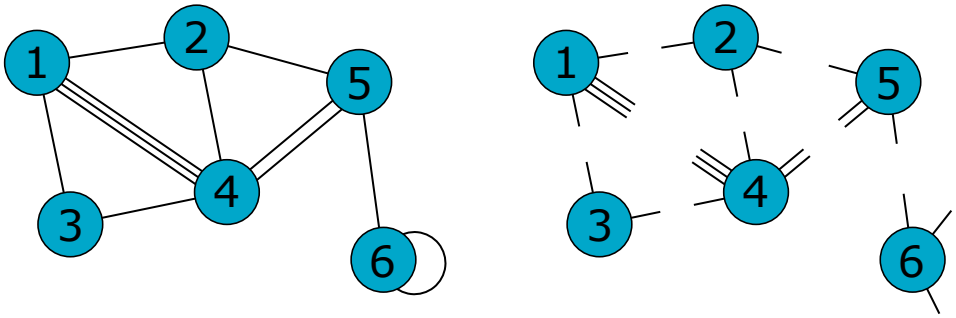

FIGURE 1.2 – (gauche) Exemple de graphe non simple contenant des multiliens et des boucles. Le multilien $(1, 4)$ a une multiplicité de 3, et le multilien $(4, 5)$ une multiplicité de 2. La boucle se trouve au lien $(6, 6)$. (droite) Les liens sur le graphe sont divisés en demi-liens qui doivent être connectés.

la séquence de degrés entrants $\mathbf{d}_{\text{in}}$.

Certains graphes possèdent des *multiliens*, c'est-à-dire que l'ensemble E contient plusieurs fois le même lien. E est alors un multiensemble. Les graphes qui contiennent des multiliens sont appelés *multigraphes*, et le nombre de fois qu'un lien apparaît est sa *multiplicité*. Certains graphes peuvent aussi contenir des *boucles*, c'est-à-dire des liens qui partent d'un noeud et pointent vers ce même noeud. Ils sont notés soit $(v \to v)$ ou (v, v). Les graphes qui contiennent des boucles sont des *pseudo-graphes*. On définit les graphes *simples* comme les graphes qui ne contiennent aucun multilien et aucune boucle.

Dans ce projet, il sera souvent question de *demi-liens*. Ces derniers correspondent à un début de lien sortant d'un noeud qui n'est pas encore connecté à un autre noeud. En abusant de la notation, on pourrait dire qu'il s'agit de la partie « $(u,$ » d'un lien, qui attend d'être connectée à un autre demi-lien « $(v,$ » pour faire un lien complet (u, v). Des exemples de multiliens, de boucle et de demi-liens sont présentés à la figure 1.2

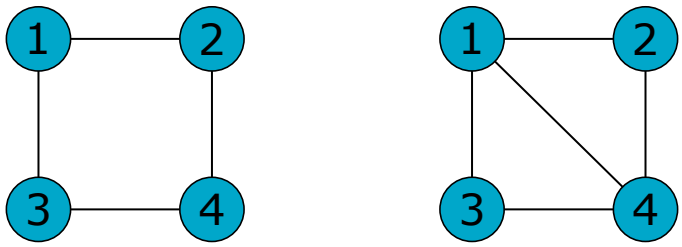

FIGURE 1.3 – (gauche) Graphe de période 2. Si on considère le noeud 1, des cycles possibles sont $(1,2);(2,1)$ de longueur 2, ou encore $(1,2);(2,4);(4,3);(3,1)$ de longueur 4. En combinant ces deux marches, on ne peut qu'obtenir des cycles de longueur paire, et donc le plus grand commun diviseur de toutes les longueurs de cycle est 2. (droite) En ajoutant le lien $(1,4)$, on permet la création d'un cycle de longueur 3, par exemple $(1,2);(2,4);(4,1)$. Le plus grand commun diviseur de 2 et 3 est 1, donc le graphe est apériodique.

Il est aussi possible de décrire les graphes avec une représentation matricielle. La *matrice d'adjacence*, notée $A(G)$ ou tout simplement A d'un graphe simple est une matrice $n \times n$ dont les entrées A_{ij} sont égales à 1 si les noeuds v_i et v_j sont connectés, ou 0 autrement. Pour les graphes non orientés, la matrice d'adjacence est toujours symétrique, c'est-à-dire que $A = A^T$. On note aussi que si les graphes ne contiennent pas de boucle, la diagonale de la matrice d'adjacence est toujours nulle. Pour un multigraphe, les entrées A_{ij} correspondent à la multiplicité des liens entre v_i et v_j. La matrice d'adjacence contient toute l'information sur un graphe, notamment le degré d'un noeud, qui est donné par

$$d(v_i) = \sum_j A_{ij} = \sum_j A_{ji}. \tag{1.1}$$

On peut maintenant définir les *chemins* sur un graphe comme étant une séquence de liens que l'on parcourt pour arriver à un noeud v à partir d'un noeud u. La longueur du chemin est le nombre de liens que l'on parcourt. Un *cycle* est un tel chemin qui part d'un noeud v et qui revient au même noeud v.

Finalement, on définit la *période* d'un graphe comme étant le plus grand commun dénominateur de tous ses cycles. On dit d'un graphe qu'il est *apériodique* si et seulement si sa période est de 1. Un exemple de graphe périodique et apériodique est présenté à la figure 1.3.

1.1.1 Corrélations de types

Comme mentionné dans l'introduction, les graphes sont souvent utilisés pour représenter des système réels. Dans ce cas, les noeuds représentent des entités réelles et les liens représentent des interactions entre ces différentes entités. Prenons l'exemple d'un réseau social humain, où les noeuds représentent des personnes et les liens représentent des relations d'amitié entre ces différentes personnes. Pour avoir des données plus complètes sur ce réseau de relations, il peut s'avérer utile de noter certaines informations pour chaque noeud présent. Des *types* sont alors

associées à chacun des noeuds du réseau, pour décrire par exemple l'âge de chaque individu, ou encore leur profession.

On peut noter la relation entre les différents types en utilisant les *matrices conjointes*, aussi appelées *matrices de corrélation* [64, 65]. Un élément $e(x, y)$ de la matrice contient la fraction des liens qui existent entre des noeuds de type x et des noeuds de type y. Par exemple, pour un réseau d'amitié, avoir $e(30, 34) = 1/3$ signifierait qu'un tiers des amitiés dans le réseau sont entre des personnes de 30 ans et 34 ans.

Les éléments des matrices de corrélation respectent les identités suivantes :

$$\sum_{x,y} e(x, y) = 1, \qquad \sum_{y} e(x, y) = a_x, \qquad \sum_{x} e(x, y) = b_y \tag{1.2}$$

où a_x est la fraction de demi-liens qui sortent des noeuds de type x (en sommant sur les colonnes de la matrice) et b_y est la fraction de demi-liens qui sortent des noeuds de type y (en sommant sur les lignes). Pour les graphes non orientés, les matrices conjointes sont toujours symétriques, soit $a_x = b_x$. Définir les quantités a_x et b_y en termes de demi-liens diffère de la littérature (où on parle plutôt de liens complets), mais permet de clarifier que la diagonale doit être doublée, c'est-à-dire qu'un lien entre deux noeuds de type x contribue $\frac{2}{m}$ à la quantité $e(x, x)$.

1.1.2 Décomposition en oignon

Outre le degré, la période et les matrices de corrélation, il existe de nombreuses autres mesures sur graphe. Parmi les plus utilisées figurent les mesures de *centralité*, qui servent à hiérarchiser les différents noeuds pour déterminer ceux qui sont les plus « importants » dans le graphe. Ce qu'on entend par importance pour un noeud peut varier significativement selon la mesure utilisée ; certaines définitions classiques permettent de calculer à quel point un noeud pourrait contribuer à une dynamique de propagation sur le réseau, ou encore à quel point un certain noeud peut contrôler l'information qui se transmet sur le réseau [68].

La *décomposition en oignon* est une telle mesure de centralité, qui sert à déterminer quels sont les noeuds les plus centraux de la structure d'un graphe [43]. Il s'agit d'une généralisation de la décomposition en *k-coeurs*, une mesure de centralité bien connue dans la littérature [42, 56, 86] qui a été introduite en 1983 [75].

Une manière simple de définir les k-coeurs d'un graphe est en utilisant l'algorithme suivant : en partant d'un graphe G, on retire les noeuds de degré inférieur à k, et on répète cette étape jusqu'à ce qu'il ne soit plus possible de retirer des noeuds. Le sous-graphe qui reste est alors le *k-coeur* de G. Un noeud v est dit avoir une *cardialité*[1] $c(v) = k$ si v fait partie du k-coeur, mais pas du $k + 1$-coeur. L'ensemble de tous les noeuds de cardialité $c = k$ est appelé la *k-coquille*.

1. Traduction libre de l'anglais *coreness*.

Comme pour la séquence de degrés, on garde l'information sur les k-coeurs dans une séquence $\mathbf{c} = \{c_1, c_2, \ldots, c_n\}$ indiquant la cardialité c_i du noeud $v_i \in V$.

L'algorithme introduit pour le calcul des k-coeurs d'un graphe [11] permet aussi de définir la décomposition en oignon [43]. L'idée est d'utiliser le même algorithme en commençant à $k = 1$, et de donner une *couche* $\ell = 1$ à chaque noeud de degré 1 ou moins qui se fait retirer [2]. On augmente ensuite la valeur de ℓ à 2, et on enlève à nouveau les noeuds de degré 1 ou moins. Cette étape est répétée jusqu'à ce que la 1-coquille soit complètement retirée du graphe, toujours en incrémentant de 1 la valeur de la couche ℓ à chaque étape. On passe ensuite à $k = 2$, et on refait les même opérations, sans oublier d'incrémenter ℓ de 1 à chaque itération où des noeuds sont retirés. On continue de la même manière, jusqu'à ce qu'il ne reste plus de noeuds dans le graphe.

Au final, en suivant cette procédure, deux valeurs seront assignées à chacun des noeuds du graphe, soit c sa cardialité et ℓ sa couche. L'algorithme 1 décrit plus en détails la procédure exacte à suivre pour extraire les k-coeurs et la décomposition en oignon d'un graphe. Comme pour la séquence de degrés, on garde l'information sur les k-coeurs et la décomposition en oignon dans des séquences $\mathbf{c} = \{c_1, c_2, \ldots, c_n\}$ et $\mathbf{OD} = \{\ell_1, \ell_2, \ldots, \ell_n\}$ indiquant respectivement la cardialité c_i et la couche ℓ_i du noeud $v_i \in V$. La figure 1.4 montre un exemple du résultat de l'algorithme 1 sur un petit graphe.

Algorithme 1 Décomposition en oignon

entrée : Graphe simple G avec ensemble de noeud V et ensemble de liens E
sortie : Séquences $\mathbf{c} = \{c_1, c_2, \ldots, c_n\}$ et $\mathbf{OD} = \{\ell_1, \ell_2, \ldots, \ell_n\}$

```
ℓ ← 0
c ← 0
tant que V est non vide faire
  pour tous les noeuds v_i ∈ V faire
    si d(v_i) ≤ c alors
      c_i ← c
      ℓ_i ← ℓ
      retirer le noeud v_i de V
    fin si
  fin pour
  si aucun noeud n'a été retiré à cette itération alors
    c ← c + 1
  sinon
    ℓ ← ℓ + 1
  fin si
```

2. On suppose ici qu'il n'y a pas de noeuds de degré 0. Si c'est le cas, on les place dans le 0-coeur et dans la couche $\ell = 0$.

fin tant que

En résumé, la décomposition en k-coeurs d'un graphe permet d'obtenir une idée générale sur les composantes denses de celui-ci. On dira alors que les noeuds faisant partie des plus hauts coeurs seront plus « centraux » au graphe, et auront une influence plus grande sur le système complexe associé au graphe que les autres noeuds « périphériques ». En indiquant le coeur le plus élevé d'un graphe, on obtient donc une idée macroscopique sur le fonctionnement de celui-ci et sur les dynamiques qui pourraient y être associées [51]. La décomposition en oignon est, pour sa part, un raffinement de cette mesure, donnant une indication sur l'organisation interne des coeurs. Malgré sa définition simple, la décomposition en oignon recèle d'information. Attardons-nous encore un peu sur cette mesure afin d'en obtenir une meilleure compréhension.

1.2 Précisions sur la décomposition en oignon

Le présent projet étant basé en grande partie sur la décomposition en oignon, une deuxième section apportant des précisions s'avère nécessaire. Cette section permettra d'abord de présenter la décomposition en oignon comme une mesure locale sur un noeud et ses voisins, puis amènera ensuite une nouvelle représentation de celle-ci, permettant d'offrir une nouvelle perspective qui sera réutilisée dans le mémoire.

1.2.1 Mesure locale

La décomposition en oignon et en k-coeurs a précédemment été présentée comme une mesure permettant d'obtenir une idée macroscopique de la composition d'un graphe. L'information contenue dans les voisins immédiats d'un seul noeud est cependant suffisante pour décrire la couche de celui-ci, ce qui permet de décrire la mesure à partir de règles dites locales ; la décomposition en oignon est donc en même temps une mesure donnant de l'information à une échelle microscopique (noeud à noeud) sur un graphe. Le théorème suivant montre les règles locales permettant de déterminer la couche d'un noeud à partir de ses voisins :

Théorème 1. *Soit un noeud v, de degré d, dans la couche ℓ et de cardialité c. Alors*

1. *Si ℓ est la première couche de la c-coquille, v doit être connecté à exactement c noeuds tel que ces noeuds sont dans une couche $\ell' \geq \ell$.*
2. *Si ℓ n'est pas la première couche de la c-coquille, v doit (a) être connecté à au moins $c+1$ liens vers des couches $\ell' \geq \ell - 1$ et (b) être connecté au plus à c liens vers des couches $\ell' \geq \ell$.*

Démonstration. Commençons par le cas où ℓ est la première couche de la c-coquille. Supposons que v possède moins de c liens avec les couches $\ell' \geq \ell$. Alors l'algorithme 1 aurait classé v dans la coquille précédente, et donc dans une couche $\ell' < \ell$. Au contraire, supposons que v

possède plus de c liens avec les couches $\ell' \geq \ell$. Alors l'algorithme ne le considérerait pas dans la première couche de c, et il serait classé dans une couche $\ell' > \ell$.[3]

Considérons maintenant le cas où ℓ n'est pas la première couche de c. Pour la règle (b), supposons que v est connecté à plus de c liens vers des couches $\ell' \geq \ell$. Alors l'algorithme le classerait dans une couche $\ell' > \ell$, et donc il faut au plus c liens vers des couches $\ell' \geq \ell$.

Pour la règle (a), on considère les liens vers les couches supérieures ou égales $\ell' \geq \ell$, de même que la couche précédente $\ell' = \ell - 1$. Supposons qu'il n'y a aucun lien vers des couches $\ell' \geq \ell$. Dans ce cas, pour que v soit classé dans la couche ℓ, il doit avoir au moins un lien de plus que la coquille le permet (autrement, il serait classé dans la première couche de sa coquille). On a donc au minimum $c + 1$ liens vers $\ell' = \ell - 1$. Maintenant, prenons un de ces $c + 1$ liens et plaçons-le vers une couche $\ell' \geq \ell$. L'algorithme classera encore v dans la couche ℓ, car son degré est trop élevé avant que l'on arrête de considérer la couche $\ell - 1$. On peut répéter cette opération jusqu'à avoir au maximum c liens vers $\ell' \geq \ell$ (pour respecter la règle (b)), et v sera toujours classé dans la couche ℓ. En suivant cette logique, ajouter des liens supplémentaires entre la couche ℓ et $\ell - 1$ ne change pas comment l'algorithme va classer v. On a donc montré qu'il faut au moins $c + 1$ liens vers des couches $\ell' \geq \ell - 1$, mais qu'il n'y a pas de limite supérieure pour ce même nombre. □

On note que, pour les noeuds qui ne sont pas dans la première couche d'un coeur, les règles (a) et (b) exigent qu'il y ait toujours au moins un lien entre la couche ℓ et la couche $\ell - 1$.

1.2.2 Couleurs de demi-liens

Observons plus en détail l'énoncé du théorème 1, en se basant sur un noeud de couche ℓ. Les demi-liens qui sont associés à ce noeud peuvent être divisés en trois catégories : les demi-liens vers des noeuds de couche $\ell' \geq \ell$, ceux vers des noeuds de couche $\ell' = \ell - 1$ et les autres, ceux vers des noeuds de couche $\ell' < \ell - 1$. Pour simplifier la notation et mieux exprimer certains concepts, il sera pertinent de donner une nouvelle représentation à ces catégories. Pour suivre la littérature existante [1], les demi-liens seront différenciés avec des couleurs, soit le rouge pour les demi-liens vers des couches $\ell' \geq \ell$, le noir pour les demi-liens vers des couches $\ell' = \ell - 1$ et le vert pour les demi-liens vers des couches $\ell' < \ell - 1$.

Pour chaque lien, on associe donc deux couleurs, une par demi-lien. Au final, il n'y a que trois combinaisons possibles pour les couleurs de liens : les liens rouge-rouge, les liens rouge-noir, et les liens rouge-vert[4]. La figure 1.4 illustre les différentes couleurs de demi-liens que l'on trouve

3. On peut aussi se convaincre en considérant le fait que si l'algorithme arrive à la première couche d'une c-coquille, il n'y a aucun noeud avec moins de c voisins, et aucun lien vers des coeurs inférieurs. Comme il enlève les noeuds qui ont exactement c liens, il enlève ceux qui ont c liens vers des couches $\ell' \geq \ell$, et donc ceux-ci sont classés dans la première couche de la c-coquille.

4. Les autres combinaisons créent des contradictions, par exemple pour les liens noir-noir on aurait $\ell = \ell' - 1$ et $\ell' = \ell - 1$

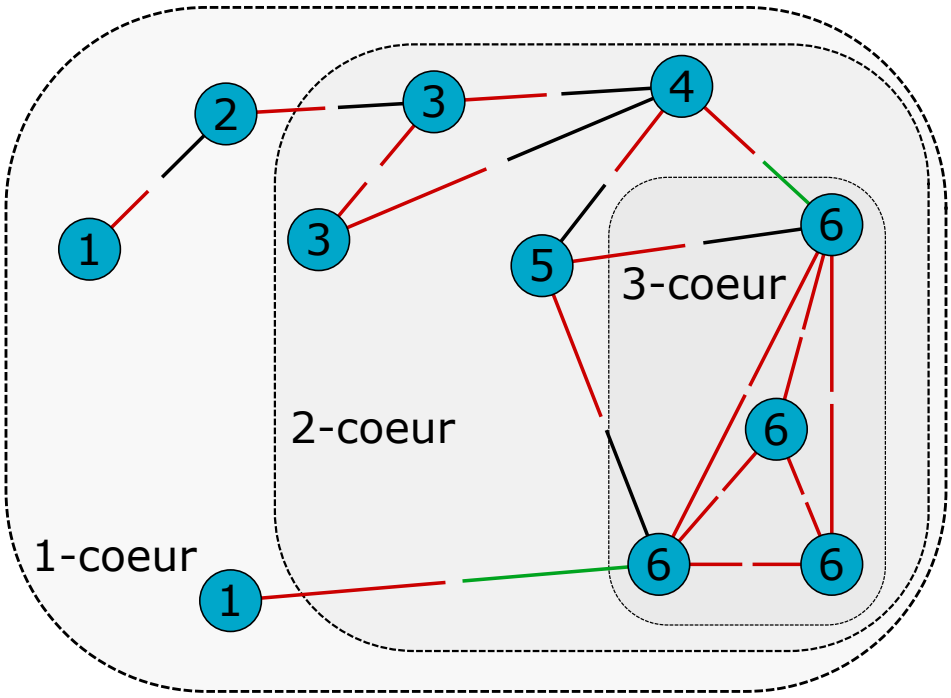

FIGURE 1.4 – Décomposition en oignon et k-coeurs sur un petit graphe. Les étiquettes dans les noeuds indiquent leur couche. La couleur des demi-liens est aussi affichée.

sur un petit graphe.

Finalement, on peut réécrire le théorème 1 en utilisant le formalisme des couleurs, ce qui donne

Théorème 2. *Soit un noeud v, de degré d, dans la couche ℓ et de cardialité c. Alors*

1. *Si ℓ est la première couche de la c-coquille, v doit avoir exactement c demi-liens rouges.*
2. *Si ℓ n'est pas la première couche de la c-coquille, v doit (a) avoir au moins $c+1$ demi-liens rouges ou noirs et (b) avoir au plus c demi-liens rouges.*

Comme mentionné précédemment, les liens qui ne sont pas dans la première couche d'un coeur possèdent toujours un demi-lien noir au minimum.

1.3 Ensembles de graphes aléatoires

La comparaison de données ou de théories à une hypothèse est indissociable de la méthode scientifique [33]. Lors d'une expérience, les données sont comparées à des modèles basés sur une hypothèse nulle. Ces modèles sont par extension appelés *modèles nuls*, et la comparaison qui est faite entre ces derniers et les résultats de l'expérience permettra éventuellement de rejeter l'hypothèse. Cette vérification statistique est essentielle et la science des réseaux n'y échappe pas : pour comparer des réseaux à des hypothèses nulles, ce sont les *ensembles de graphes aléatoires*, ou *modèles de graphes* qui seront utilisés.

Les ensembles de graphes aléatoires permettent de générer des graphes, ou du moins de représenter une hypothèse nulle sur des graphes, que l'on peut utiliser comme données « synthétiques » pour un modèle nul. Cet outil permet d'effectuer certaines comparaisons de données réelles qui ne pourraient être faites autrement dû à l'impossibilité d'obtenir des réseaux à partir de groupes de contrôles. Par exemple, on peut penser à l'analyse du réseau généré par les hyperliens sur le *World Wide Web*, qui ne permet pas nécessairement la création de groupe de contrôle puisqu'il n'existe pas nécessairement d'autres réseaux publics sur Internet d'une aussi grande envergure.

On définit un ensemble de graphes aléatoires comme un regroupement de graphes qui possèdent tous une caractéristique commune qui est dictée par l'hypothèse nulle. Outre cette caractéristique commune, la structure des graphes est entièrement aléatoire. En général, les ensembles de graphes aléatoires sont données par des contraintes que l'on applique sur l'ensemble de tous les graphes possibles.

Soit $\mathcal{G}$ l'ensemble contenant tous les graphes possibles. Un ensemble aléatoire de graphes est décrit par la distribution de probabilité sur $\mathcal{G}$, soit $P(G)$ tel que $\sum_{G\in\mathcal{G}} P(G) = 1$. On peut aussi utiliser les matrices d'adjacences pour décrire la distribution, auquel cas on écrit $P(A)$ pour $A \in \mathcal{G}$.

Un exemple bien connu en science des réseaux est le modèle de Gilbert, souvent connu sous le nom de $G(n,p)$, où l'on impose une contrainte sur le nombre de noeuds n et sur le nombre moyen de liens $\langle m\rangle$ [38]. Pour imposer ces contraintes, on prend d'abord un ensemble V de n noeuds, puis on ajoute un lien entre chaque noeud, avec une probabilité $p = 2\langle m\rangle/n(n-1)$ [5]. La distribution de probabilité pour un graphe A dans l'ensemble est alors donnée par

$$P(A) = \prod_{i=1}^{n}\prod_{j=1}^{i-1}\left[p\,\delta_{A_{ij},1} + (1-p)\,\delta_{A_{ij},0}\right]. \tag{1.3}$$

On note que cette distribution est normalisée puisque

$$\begin{aligned}\sum_{A\in\mathcal{G}} P(A) &= \sum_{A\in\mathcal{G}}\prod_{i=1}^{n}\prod_{j=1}^{i-1}\left[p\,\delta_{A_{ij},1} + (1-p)\,\delta_{A_{ij},0}\right]\\ &= \prod_{i=1}^{n}\prod_{j=1}^{i-1}\sum_{A_{ij}\in\{0,1\}}\left[p\,\delta_{A_{ij},1} + (1-p)\,\delta_{A_{ij},0}\right]\\ &= \prod_{i=1}^{n}\prod_{j=1}^{i}\left[p + (1-p)\right]\\ &= 1.\end{aligned}$$

5. De cette manière, on a $\langle m\rangle = p\frac{n(n-1)}{2}$, où $\frac{n(n-1)}{2}$ est le nombre de degrés libres dans la matrice d'adjacence d'un graphe non orienté de n noeuds (ou encore, le nombre maximal de liens).

Un autre modèle similaire est le modèle d'Erdős-Rényi, ou $G(n, m)$ [30]. Dans celui-ci, on impose encore une contrainte sur le nombre de noeud et sur le nombre de liens, mais cette fois-ci, la contrainte sur le nombre de liens est exacte. Donc, tous les graphes contenus dans l'ensemble $G(n, m)$ sont des graphes qui possèdent exactement m liens, contrairement au modèle $G(n, p)$ où ce nombre peut varier. Cet exemple permet d'introduire les deux types de contraintes que l'on peut imposer sur les ensembles de graphes aléatoires ; Le modèle de Gilbert est un modèle imposant une *contrainte souple* sur le nombre de liens, tandis que le modèle d'Erdős-Rényi est un modèle à *contraintes rigides*.

1.3.1 Contraintes rigides

Les modèles à contraintes rigides sont des modèles où l'on impose qu'une certaine mesure $\mu(A)$ sur un graphe A soit conservée exactement pour chacun des graphes dans l'ensemble. Souvent, le modèle est basé sur un premier graphe G^* ou A^*, qui est celui issu des données réelles et que l'on veut comparer au modèle. Donc, pour un ensemble avec une contrainte rigide sur la mesure $\mu(A^*)$, on a

$$\mu(A) = \mu(A^*), \qquad \forall A \in \mathcal{G} \text{ tel que } P(A) > 0. \tag{1.4}$$

Autrement dit, on demande que la probabilité $P(A)$ soit nulle si $\mu(A) \neq \mu(A^*)$.

On peut réduire l'ensemble $\mathcal{G}$ pour travailler uniquement avec les graphes qui respectent la contrainte rigide. Appelons le sous-ensemble Ω. Comme on veut utiliser les ensembles aléatoires comme modèles nuls, on cherche à avoir la distribution de probabilité qui est la plus « aléatoire » possible. Pour ce faire, on demande que la distribution des probabilités maximise l'entropie de Shannon sur Ω, qui est donnée par

$$S(P, \Omega) = -\sum_{A \in \Omega} P(A) \log P(A). \tag{1.5}$$

Pour maximiser S, on peut utiliser la méthode des multiplicateurs de Lagrange, avec λ comme paramètre. La seule contrainte que l'ensemble doit respecter est $\sum_{A \in \Omega} P(A) = 1$. Donc, le système d'équation à résoudre est

$$\frac{\partial}{\partial P(A)} \left(S(P, \Omega) - \lambda \left[1 - \sum_{A' \in \Omega} P(A') \right] \right) = 0, \tag{1.6}$$

et la solution est le système d'équation

$$\log P(A) + 1 = \lambda \quad \text{ou} \quad P(A) = e^{\lambda - 1}. \tag{1.7}$$

Les probabilités ne dépendent que du paramètre λ, et on a donc que $P(A) = P(A')$ pour tout $A, A' \in \Omega$. La condition de normalisation nous indique finalement que la solution qui maximise l'entropie est

$$P(A) = \frac{1}{|\Omega|} \tag{1.8}$$

où $|\Omega|$ est le nombre de graphes A tels que $\mu(A) = \mu(A^*)$. Donc, pour les modèles à contraintes rigides, on cherche toujours à avoir une distribution uniforme sur l'ensemble Ω.

1.3.2 Contraintes souples

Pour les modèles à contraintes souples, on assouplit la condition (1.4) et on demande que les mesures $\mu(A)$ soient conservées en moyenne dans l'ensemble. Si l'on veut contraindre la mesure $\mu(A^*)$ du graphe A^*, on aura la condition

$$\langle \mu \rangle = \sum_{A \in \Omega} P(A)\mu(A) = \mu(A^*). \tag{1.9}$$

Encore une fois, on veut maximiser l'entropie dans ce type d'ensemble, mais cette fois en considérant ces nouvelles contraintes. Le développement n'est pas présenté ici, mais le résultat donne des modèles exponentiels de graphes aléatoires (de l'anglais *Exponential random graph models*) [23].

En général, quand on considère des ensembles de graphes, on met une contrainte implicite sur le nombre de noeuds. C'est-à-dire, plutôt que d'utiliser l'ensemble $\mathcal{G}$ de tous les graphes possibles, on appliquera une distribution sur l'ensemble $\mathcal{G}_n$ de tous les graphes avec n noeuds. Avec ceci, il est possible de faire un parallèle entre les différents types de modèles de graphes et les ensembles en physique statistique : les modèles où le nombre de noeuds peut changer, c'est-à-dire ceux où l'on se base sur $\mathcal{G}$, sont équivalents au modèle grand canonique, où le nombre de particules peut varier. Si on se base plutôt sur les ensembles $\mathcal{G}_n$, on sort du grand canonique et on retrouve l'ensemble microcanonique pour les ensembles à contraintes rigides, et l'ensemble canonique pour les ensembles à contraintes souples.

Pour revenir aux exemples précédents, on peut présenter le modèle d'Erdős-Rényi comme un modèle à contrainte rigide basé sur l'ensemble $\mathcal{G}_n$, puisque tous les graphes de probabilité non nulle ont exactement n noeuds et m liens. Pour sa part, le modèle de Gilbert est un modèle à contrainte souple sur le nombre de liens, aussi basé sur $\mathcal{G}_n$. D'autres ensembles à contraintes rigides et à contraintes mixtes ont été développés pour le projet, et ceux-ci auront toujours une contrainte rigide sur le nombre de noeuds, même si celle-ci n'est pas explicitement mentionnée. On peut désormais présenter un nouveau modèle de graphes, qui forme une base pour tous les autres ensembles développés dans le projet.

1.4 Modèle des configurations

Le degré des noeuds, et donc par extension la séquence de degrés est une observation simple qui peut rapidement révéler de l'information sur le voisinage des noeuds dans un graphe. Cette dernière peut même être vue comme une mesure de centralité, en indiquant quels sont les noeuds les plus « populaires » dans un système. Son importance et sa simplicité expliquent pourquoi cette mesure est l'une des premières à avoir été considérée pour le développement

de modèles de graphes [17, 69]. À ce sujet, il existe une longue histoire de comparaison de données réelles à des modèles aléatoires basés sur des séquences de degrés fixes dans plusieurs domaines [35].

Le modèle des configurations présenté ici est un modèle avec une contrainte rigide sur la séquence de degrés : soit un graphe G avec un ensemble de noeuds $V = \{v_i\}_{i=1}^n$ et une séquence de degrés $\mathbf{d} = \{d_i\}_{i=1}^n$. Le *modèle des configurations de* G (CM [6]), noté $\Gamma_{CM}(G)$, est l'ensemble contenant tous les graphes ayant la séquence de degrés $\mathbf{d}$.

À noter que dans la littérature, le modèle des configurations peut être utilisé pour différents ensembles, tout dépendamment de la manière d'indicer les interactions. On peut parler d'ensembles à indices sur les demi-liens (*stub-labeled* en anglais), d'ensembles à indices sur les noeuds (*vertex-labeled*), ou d'ensembles sans indices. Ces trois nuances donnent des ensembles différents mais hiérarchisés, l'ensemble indicé sur les demi-liens contenant les deux autres, et l'ensemble indicé sur les noeuds contenant l'ensemble sans indice. Le présent document traite uniquement des ensembles indicés sur les noeuds, mais les résultats qui seront présentés seront valides pour tous ces ensembles.

En général, pour faire les comparaisons avec les données réelles, on cherche à obtenir un échantillon de graphes qui est tiré de notre modèle nul. Pour ce faire, il faudra développer des algorithmes qui permettent d'obtenir ces graphes. L'algorithme utilisé sera décrit plus tard dans la section 1.6, mais on peut déjà noter qu'il s'agit d'un algorithme de type *Chaînes de Markov Monte Carlo* (MCMC [7]). La prochaine section se concentrera sur la théorie derrière les MCMC, et cette théorie sera appliquée par la suite pour bien développer l'algorithme d'échantillonnage pour le modèle des configurations.

1.5 Chaînes de Markov

Comme leur nom l'indique, les algorithmes de MCMC sont basés sur les chaînes de Markov, et sur la méthode de Monte Carlo.

Pour la partie Monte Carlo, nous nous contenterons de dire qu'il s'agit d'une famille de méthodes stochastiques, qui utilise la loi des grands nombres pour donner une solution à des problèmes de nature analytique. La *loi des grands nombres* désigne ici le fait que pour une variable aléatoire X donnée, la moyenne des échantillons obtenus, soit $\frac{1}{N}\sum_{i=1}^N x_i$ tendra vers l'espérance $E[X]$ de la distribution $P(X = x)$ pour de larges échantillons. Autrement dit, nous avons

$$\frac{1}{N}\sum_{i=1}^{N} x_i \to E[X] \quad \text{lorsque} \quad N \to \infty.$$

6. Acronyme de l'anglais *Configuration Model*.
7. Acronyme en anglais pour *Markov Chain Monte Carlo*.

Il s'agit d'un résultat intuitif, mais qui revêt une grande importance. Dans des termes plus simples, on peut interpréter la loi pour dire que si l'on répète un très grand nombre de fois une expérience aléatoire, alors la moyenne arithmétique des résultats de l'expérience tendra vers l'espérance de la distribution. À la lumière de cette définition, on peut inférer que le principe des algorithmes de Monte Carlo est de répéter plusieurs fois une expérience numérique, d'estimer l'espérance de la distribution de l'expérience à partir des résultats obtenus et finalement de résoudre le problème initial grâce à cette estimation.

La partie Monte Carlo de l'algorithme consiste donc à répéter plusieurs fois une expérience, qui permettra d'évaluer l'espérance sur l'ensemble. Dans notre cas, l'expérience consiste à ajouter un graphe à notre échantillon. La partie chaîne de Markov, quant à elle, permet de déterminer quelle sera la distribution de notre expérience, c'est-à-dire quelle sera la probabilité d'obtenir un certain graphe. Il faut donc une bonne compréhension des chaînes de Markov, pour assurer que l'algorithme ait une convergence vers une distribution représentative du modèle que l'on veut échantillonner.

1.5.1 Premières définitions

Soit un espace des états Ω et une variable aléatoire dépendant du temps $X(t)$ pour $t = 1, 2, \ldots$. Dans notre cas, nous ne considérons que les espace d'états discrets, c'est-à-dire $|\Omega| < \infty$, et la variable de temps est discrète. Aux temps $t_1, t_2, \ldots$, on mesure les états $x_1, x_2, \ldots$ pour $x_i \in \Omega$ avec probabilités conjointes

$$P(x_1, t_1; x_2, t_2; x_3, t_3; \ldots). \tag{1.10}$$

Si toutes les probabilités jointes sont connues, on peut alors parler d'un *processus stochastique*[8]. En d'autres mots, un processus stochastique est un processus qui varie entre différents états dans le temps, ces états changeant selon les probabilités (1.10).

Les probabilités conjointes (1.10) peuvent être transformées en probabilités conditionnelles : la probabilité de mesurer l'état x_i au temps t_i si l'on avait mesuré les états y_i aux temps τ_i est donnée par

$$P(x_1, t_1; x_2, t_2; \ldots | y_1, \tau_1; y_2, \tau_2) = \frac{P(x_1, t_1; x_2, t_2; \ldots; y_1, \tau_1; y_2, \tau_2; \ldots)}{P(y_1, \tau_1; y_2, \tau_2; \ldots)}. \tag{1.11}$$

Il est maintenant possible de proposer la définition suivante : une *chaîne de Markov* est un processus stochastique dont les transitions vers un nouvel état sont basées uniquement sur l'état actuel de la chaîne. Donc, pour $t_1 > \tau_1 > \tau_2 > \ldots$, les probabilités conditionnelles d'une chaîne de Markov prennent la forme

$$P(x_1, t_1 | y_1, \tau_1; y_2, \tau_2; \ldots) = P(x_1, t_1 | y_1, \tau_1), \tag{1.12}$$

8. Il ne s'agit pas d'une définition complète, le lecteur intéressé est référé à [37] pour plus de détails.

ou, pour $t_n > t_{n-1} > \ldots > t_2 > t_1$,

$$\begin{aligned} &P(x_1, t_1; x_2, t_2; \ldots; x_n, t_n | y_1, \tau_1; y_2, \tau_2; \ldots) \\ &\qquad = P(x_n, t_n | x_{n-1}, t_{n-1}) \ldots P(x_2, t_2 | x_1, t_1) P(x_1, t_1 | y_1, \tau_1). \end{aligned} \tag{1.13}$$

Cette équation est la traduction mathématique de la *propriété de Markov* qui définit les chaînes du même nom. Elle illustre comment les chaînes de Markov n'ont pas de « mémoire », c'est-à-dire comment les états précédents ne sont pas considérés dans le calcul de probabilité d'une transition vers le prochain état.

Les chaînes qui seront étudiées ici sont des chaînes à temps discret. Le temps sera donc structuré à partir d'un temps t, auquel on incrémente 1 à chaque itération de la chaîne ; pour une itération, on passe du temps t au temps $t + 1$.

On peut décrire une chaîne de Markov au temps t avec un *vecteur d'état* $\pi(t)$, qui est un vecteur colonne de $|\Omega|$ éléments. Chaque élément du vecteur d'état représente la probabilité d'être dans l'état associé à cet élément, soit

$$\pi(t) = \begin{pmatrix} P(x_1, t) \\ P(x_2, t) \\ \vdots \end{pmatrix}. \tag{1.14}$$

Par exemple, si on initialise une chaîne de Markov dans un état initial où $X(t = 0) = x_1$, on aura le vecteur d'état

$$\pi(0) = \begin{pmatrix} 1 \\ 0 \\ \vdots \end{pmatrix}, \tag{1.15}$$

où la probabilité d'être dans l'état x_1 est 1, et la probabilité d'être dans tous les autres états est nulle.

Par définition, la probabilité d'être dans un état x_j au temps $t + 1$ est donnée par

$$\begin{aligned} P(x_j, t+1) &= P(x_j, t+1 | x_1, t) P(x_1, t) + P(x_j, t+1 | x_2, t) P(x_2, t) + \ldots \\ &= \sum_{x_i \in \Omega} P(x_j, t+1 | x_i, t) P(x_i, t), \end{aligned} \tag{1.16}$$

où les probabilités conditionnelles donnent la probabilité que l'on passe d'un état y à un état x. En général, on considère que ces probabilités sont invariantes dans le temps [9] : donc pour tout états x_i, x_j, on a que

$$P(x_j, t+1 | x_i, t) =: P(x_j | x_i). \tag{1.17}$$

9. Il existe des chaînes de Markov, dites *inhomogènes*, avec des probabilités de transition qui varient avec le temps. Celles-ci sont cependant peu étudiées dans la littérature puisqu'il est extrêmement ardu de faire des preuves générales sur leurs processus.

La probabilité d'être dans un état x_j au temps $t+1$ devient alors

$$P(x_j, t+1) = \sum_{x_i \in \Omega} P(x_j|x_i)P(x_i, t). \tag{1.18}$$

On appelle les probabilités $P(x_j|x_i)$ les *probabilités de transition*. On peut écrire toutes les probabilités de transitions sous forme matricielle, dans une matrice P que l'on appelle *matrice de transition*. Celle-ci est une matrice de dimension $|\Omega| \times |\Omega|$, et possède la forme suivante :

$$P := \begin{pmatrix} P(x_1|x_1) & P(x_1|x_2) & \dots \\ P(x_2|x_1) & P(x_2|x_2) & \dots \\ \vdots & \vdots & \ddots \end{pmatrix}. \tag{1.19}$$

Observons ce qui se produit si on multiplie un vecteur d'état $\pi(t)$ par une matrice de transition P :

$$\begin{aligned} P\pi(t) &= \begin{pmatrix} P(x_1|x_1) & P(x_1|x_2) & \dots \\ P(x_2|x_1) & P(x_2|x_2) & \dots \\ \vdots & \vdots & \ddots \end{pmatrix} \begin{pmatrix} P(x_1, t) \\ P(x_2, t) \\ \vdots \end{pmatrix} \\ &= \begin{pmatrix} \sum_{x_i \in \Omega} P(x_1|x_i)P(x_i, t) \\ \sum_{x_i \in \Omega} P(x_2|x_i)P(x_i, t) \\ \vdots \end{pmatrix} \\ &= \begin{pmatrix} P(x_1, t+1) \\ P(x_2, t+1) \\ \vdots \end{pmatrix} = \pi(t+1) \end{aligned}$$

Donc, en utilisant les vecteurs d'états et la matrice de transition, on peut trouver une équation simple qui permet de décrire entièrement l'évolution d'une chaîne de Markov lors d'une itération :

$$\pi(t+1) = P\pi(t). \tag{1.20}$$

Si on applique cette équation pour obtenir le temps $t+2$, on trouve

$$\pi(t+2) = P\pi(t+1) = P^2\pi(t).$$

De manière générale pour n itérations de la chaîne, on aura

$$\pi(t+n) = P^n\pi(t). \tag{1.21}$$

Finalement, l'équation pour décrire l'évolution d'une chaîne de Markov depuis son état initial $\pi(0)$ est

$$\pi(t) = P^t\pi(0). \tag{1.22}$$

1.5.2 Marches aléatoires

Illustrons les concepts définis précédemment avec un exemple simple. Les marches aléatoires sont des processus stochastiques où un objet se déplace dans un espace de manière aléatoire. Si on pense au déplacement sur une droite, on peut imaginer un objet qui, à chaque itération, fait un pas vers la droite avec probabilité p, fait un pas vers la gauche avec probabilité q, ou reste sur place avec probabilité $1-p-q$. Le processus décrit ainsi une chaîne de Markov, dont l'espace des états est l'ensemble des positions que l'objet peut prendre sur la droite.

On peut aussi élargir ce concept à des espaces autres que des droites, notamment les graphes. Soit un graphe G avec l'ensemble de noeuds V et l'ensemble de liens E. Une *marche aléatoire sur graphe* est une marche aléatoire dont l'espace des états est composé de l'ensemble des noeuds V. Les probabilités de transition sont données par les liens de E, auxquels on associe un poids $w(x_i \to x_j)$ qui dicte les probabilités de transition tel que $P(x_j|x_i) = w(x_i \to x_j)$. Comme les probabilités de transitions ne dépendent que des poids, les marches aléatoires sur graphes telles que décrites sont des chaînes de Markov.

On peut également créer une marche aléatoire sur graphe à partir d'une chaîne de Markov : prenons une chaîne de Markov avec un espace des états Ω et une matrice de transition P. Il est toujours possible de construire un ensemble de noeuds V représentant chaque état de la chaîne. On peut ensuite former un ensemble de liens, où un lien partant du noeud x_i allant vers le noeud x_j existe si $P(x_j|x_i) > 0$ dans la chaîne. Si on associe le poids $w(x_i \to x_j) = P(x_j|x_i)$ aux liens ainsi créés, on obtient un *graphe pondéré* G_w, qui contient toute l'information sur la chaîne de Markov originale. La marche aléatoire sur G_w sera alors équivalente à cette dernière.

Grâce à cette procédure, on peut représenter toutes les chaînes de Markov discrètes avec une marche aléatoire sur un graphe. Il s'agit d'une deuxième manière de présenter les chaînes de Markov qui s'avérera très utile dans le développement de preuves. En général dans ce document, les preuves sur les propriétés des chaînes seront explicités en utilisant les deux représentations : des premières preuves se concentreront sur les propriétés de la matrice de transition, et celles-ci seront traduites en une version qui considère la topologie du graphe G_w associé. Ceci aura pour but d'offrir une vision alternative à chaque problème étudié et permettra d'obtenir une meilleure intuition sur certains résultats, en plus de raccorder certains concepts venant de différentes branches des mathématiques.

Pour illustrer les concepts présentés, imaginons un espace composé de quatre cases placées sur un axe horizontal. Appelons ces cases x_1, x_2, x_3 et x_4. Un objet commence à la case x_2, et se déplace à chaque temps d'une case vers la gauche avec probabilité $\frac{1}{2}$, ou d'une case vers la droite avec probabilité $\frac{1}{2}$. On ne peut se déplacer vers la gauche à partir de x_1, ni se déplacer vers la droite à partir de x_4. Dans ce cas, on va dans l'autre direction avec probabilité 1. Ce processus est une chaîne de Markov, puisque toute les probabilités de transitions ne dépendent

que de l'état actuel. La matrice de transition associée à ce problème est

$$P = \begin{pmatrix} 0 & \frac{1}{2} & 0 & 0 \\ 1 & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & 0 & 1 \\ 0 & 0 & \frac{1}{2} & 0 \end{pmatrix}. \tag{1.23}$$

Comme mentionné précédemment, nous pouvons créer un graphe à partir de cette chaîne, et effectuer une marche aléatoire en fonction des poids sur celle-ci, pour obtenir une nouvelle représentation de la même chaîne de Markov. La chaîne de Markov et son graphe associé sont présentés à la figure 1.5a.

On peut complexifier ce premier exemple en permettant à l'objet de sauter par dessus une case. Disons que l'objet se déplace d'une case vers la gauche avec probabilité $\frac{1}{3}$, se déplace de deux cases (saut) vers la gauche avec probabilité $\frac{1}{6}$, et même chose pour la droite. Lorsque le déplacement fait sortir l'objet de l'espace (par exemple un saut vers la gauche à partir de x_2 ou un déplacement vers la droite depuis x_4), on double la probabilité de faire la même action mais dans la direction opposée. La matrice de transition obtenue serait

$$Q = \begin{pmatrix} 0 & \frac{1}{3} & \frac{1}{3} & 0 \\ \frac{2}{3} & 0 & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & 0 & \frac{2}{3} \\ 0 & \frac{1}{3} & \frac{1}{3} & 0 \end{pmatrix}. \tag{1.24}$$

Cette nouvelle chaîne de Markov et son graphe sont illustrés à la figure 1.5b.

Pour analyser les deux exemples, nous pouvons utiliser l'équation (1.22). Pour la deuxième marche présentée, si l'on commence à la position x_2, la probabilité de se trouver dans un état quelconque au temps t est contenue dans le vecteur d'état $\pi(t)$ et se calcule selon

$$\pi(t) = Q^t \pi(0) = \begin{pmatrix} 0 & \frac{1}{3} & \frac{1}{3} & 0 \\ \frac{2}{3} & 0 & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & 0 & \frac{2}{3} \\ 0 & \frac{1}{3} & \frac{1}{3} & 0 \end{pmatrix}^t \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}. \tag{1.25}$$

Une question à laquelle on s'intéresse généralement dans les processus stochastiques est la suivante : quel est le comportement à long terme de la chaîne ? C'est-à-dire, pour un temps $t \to \infty$, quelles sont les probabilités d'être dans chaque état ? Observons d'abord le comportement de la chaîne B.

En effectuant le calcul à la main (ou numériquement) pour de très grandes valeurs de t, nous remarquerons deux comportements importants : d'abord, les probabilités d'être dans un

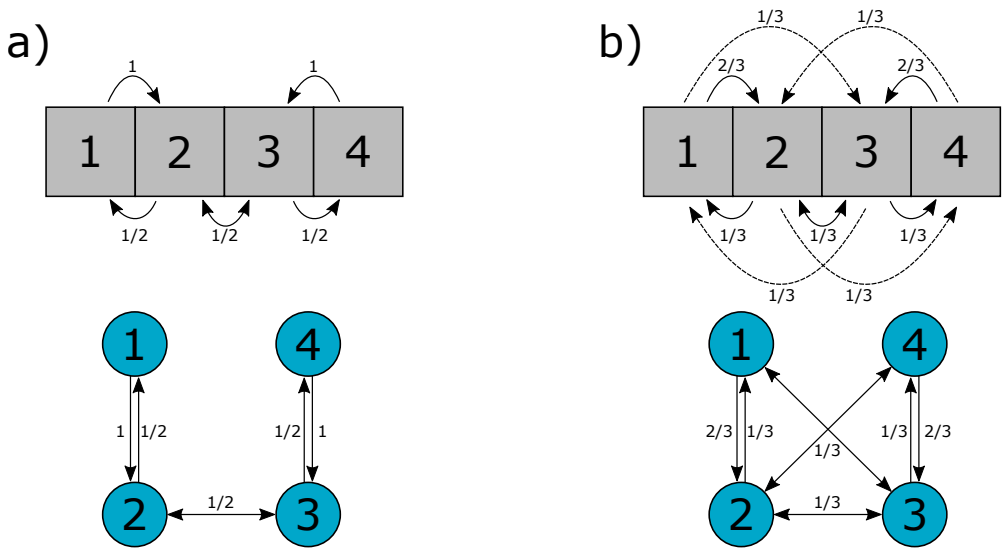

FIGURE 1.5 – Illustration des marches aléatoires sur les cases, et leurs graphes associés. Les flèches pointillées indiquent les sauts.

certain état vont se stabiliser et vont converger vers une certaine valeur. On dira alors que la chaîne a atteint son *état d'équilibre*. Ensuite, si on reproduit l'expérience mais en partant de n'importe quel autre état initial, la chaîne convergera encore, et toujours vers les mêmes valeurs d'équilibre qu'en partant de x_2.

Donc, pour cette chaîne, il existe un état d'équilibre vers lequel elle converge et dans lequel le vecteur d'état est constant. De plus, cet état d'équilibre ne dépend pas de l'état initial de la chaîne. On pourrait aussi analyser la vitesse de la convergence, et on remarquerait que, comme pour l'état d'équilibre, cette dernière ne dépend pas de l'état initial.

Si on refait le même exercice pour la marche aléatoire A, on remarquera que le résultat ne dépend toujours pas de la condition initiale, mais qu'il n'y aura pas de convergence vers un état d'équilibre. Le vecteur d'état obtenu variera plutôt entre deux valeurs à chaque itération. Les résultats présentés ici de manière informelles sont très importants : les prochaines sous-sections définiront mathématiquement ces comportements que l'on observe par l'expérience.

1.5.3 Distribution stationnaire

La *distribution stationnaire* d'une chaîne de Markov est un vecteur d'état π^* définit par

$$\pi^* = P\pi^*. \tag{1.26}$$

Donc, lorsque le vecteur d'état d'une chaîne atteint la distribution stationnaire, il n'y a plus d'évolution dans les probabilités du système. La chaîne atteint l'équilibre statistique, c'est-à-dire que les propriétés macroscopiques (le vecteur d'état) sont constantes même si ses propriétés microscopiques (les états de la chaîne) continuent de changer à chaque itération.

On remarque de l'équation (1.26) que l'on peut décrire la distribution stationnaire π^* comme un vecteur propre de P avec valeur propre $\lambda = 1$. Pour une chaîne donnée, il faut s'assurer de l'existence de ce vecteur propre. C'est ce que nous ferons ci-dessous avec une série de théorèmes. Toutefois, avant de présenter ces derniers, définissons le rayon spectral d'une matrice : le *rayon spectral* de la matrice A, noté $\rho(A)$ est la norme de sa plus grande valeur propre, soit

$$\rho(A) = \max_{\lambda \in \sigma(A)} |\lambda|, \tag{1.27}$$

où $\sigma(A)$ est le spectre de A, l'ensemble contenant ses valeurs propres.

Théorème 3. *[45] Soit A une matrice carrée $n \times n$. Alors $\sigma(A) = \sigma(A^T)$.*

Démonstration. Les valeurs propres d'une matrice sont déterminées en trouvant les racines de son polynôme caractéristique, lequel est calculé avec $\det(\lambda I - A)$ [10]. Pour A^T, on a

$$\det\big(\lambda I - A^T\big) = \det\big(\lambda I^T - A^T\big) = \det\Big([\lambda I - A]^T\Big) = \det(\lambda I - A)$$

puisque $\det(A) = \det\big(A^T\big)$ par définition du déterminant d'une matrice. Donc, A^T et A ont le même polynôme caractéristique, et donc le même spectre. □

Le premier théorème est un résultat bien connu, dont la preuve est tirée de [45]. Le deuxième théorème est présenté sans preuve, mais est un corollaire du théorème des disques de Gershgorin, qui est un résultat bien connu en algèbre linéaire [12]. L'énoncé du théorème des disques permet une application directe de 3 pour obtenir le théorème suivant :

Théorème 4. *Soit une matrice carrée A de dimension $n \times n$. Alors le rayon spectral $\rho(A)$ est borné supérieurement tel que*

$$\rho(A) \leq R_i = \sum_{j=1}^{n} |A_{ij}| \tag{1.28}$$

$$\rho(A) \leq \tilde{R}_i = \sum_{j=1}^{n} |A_{ji}| \tag{1.29}$$

pour tout i. R_i est la somme de la valeur absolue des éléments sur une ligne de A et $\tilde{R}_i$ est la somme de la valeur absolue des éléments d'une colonne.

On montre maintenant que la valeur propre $\lambda = 1$ existe toujours pour la matrice de transition d'une chaîne de Markov.

Théorème 5. *Soit la matrice P, une matrice de transition d'une chaîne de Markov. Alors $\lambda = 1$ est une valeur propre de P. De plus, le rayon spectral de P est $\rho(P) = 1$.*

10. La matrice I est la *matrice identité*, une matrice carrée dont les éléments sont nuls excepté sur la diagonale où ils sont tous égaux à 1.

Démonstration. Pour prouver le premier résultat, on commence par noter que les matrices de transition sont, par construction, des matrices stochastiques par la gauche. Les matrices stochastiques par la gauche sont des matrices dont la somme de chaque colonne est 1 ; dans le cas des matrices de transition, la somme sur une colonne est $\sum_{x_i \in \Omega} P(x_i|x_j)$, ce qui donne toujours 1.

On peut donc définir un vecteur unitaire $\mathbf{e} = \begin{pmatrix} 1 & 1 & \dots \end{pmatrix}^T$, qui est un vecteur colonne de longueur $|\Omega|$. Si on multiplie ce vecteur par P^T, on obtient

$$P^T\mathbf{e} = \begin{pmatrix} P(x_1|x_1) & P(x_1|x_2) & \dots \\ P(x_2|x_1) & P(x_2|x_2) & \dots \\ \vdots & \vdots & \ddots \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ \vdots \end{pmatrix} = \begin{pmatrix} \sum_{x_i \in \Omega} P(x_i|x_1) \\ \sum_{x_i \in \Omega} P(x_i|x_2) \\ \vdots \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ \vdots \end{pmatrix} = \mathbf{e}.$$

Ce qui montre que $\mathbf{e}$ est un vecteur propre de P^T avec valeur propre $\lambda = 1$. Par le théorème 3, on a que $\lambda = 1$ est aussi une valeur propre de P.

Le deuxième résultat est, quant à lui, une conséquence directe du théorème 4 et du fait que P est stochastique. En effet, comme P est stochastique par la gauche, nous avons que la somme sur ses colonnes est de 1, ou plus précisément $\tilde{R}_i = \sum_j |A_{ji}| = \sum_j A_{ji} = 1$ pour tout i. Donc, par ce théorème, nous avons que $\rho(P) \leq \tilde{R}_i = 1$. Comme nous savons que $\lambda = 1$ est une valeur propre de P, ceci donne $\rho(P) = 1$. □

1.5.4 Conditions pour une distribution stationnaire

Le théorème 5 qui vient d'être présenté implique qu'il existe toujours une distribution stationnaire pour les chaînes de Markov. En effet, puisque $\lambda = 1$ est toujours une valeur propre de P, il existe toujours au moins un vecteur propre qui lui est associé. Le théorème ne donne cependant aucune information sur l'unicité de cette distribution stationnaire ; il n'indique pas si π^* est un vecteur « physique », c'est-à-dire si ce dernier représente bien une distribution, avec des éléments non négatifs qui somment à 1 ; finalement, il n'indique pas non plus comment une chaîne peut converger vers cet état, comme l'expérience précédente nous a montré.

Il existe de nombreuses façons de répondre à ces questions. Ici, nous choisissons de rester dans le formalisme de l'algèbre linéaire, et utilisons le théorème de Perron-Frobenius pour obtenir plus de détails sur la distribution stationnaire. Le théorème de Perron-Frobenius est une série de résultats sur les matrices non négatives, c'est-à-dire sur les matrices dont tous les éléments sont soit nuls, soit positifs. Les résultats du théorème sont présentés sans preuves, mais le lecteur intéressé pourra se familiariser avec celles-ci dans tout ouvrage de référence d'algèbre linéaire avancée [45, 59].

Avant de présenter le théorème de Perron-Frobenius, nous présentons deux propriétés qu'une matrice peut avoir. Celles-ci formeront la base des théorèmes importants pour le projet.

Une matrice A de dimension $n \times n$ est dite *réductible* s'il existe une matrice de permutation $\mathcal{P}$ telle que

$$\mathcal{P}^T A \mathcal{P} = \begin{pmatrix} X & Y \\ 0 & Z \end{pmatrix}, \tag{1.30}$$

où X et Z sont deux matrices carrées quelconques, et Y est une matrice quelconque. Autrement dit, s'il existe une série de permutations de lignes et de colonnes sur A qui la rend triangulaire supérieure par blocs, A est réductible. S'il n'existe pas de telle permutation, A est dite *irréductible*.

Une matrice A de dimension $n \times n$ est dite *primitive* si elle est irréductible et si elle possède une unique valeur propre sur son rayon spectral.

Ces deux propriétés seront présentées à nouveau dans le contexte des chaînes de Markov à la prochaine sous-section. Pour l'instant, retournons au théorème de Perron-Frobenius :

Théorème 6. ***Théorème de Perron-Frobenius*** *[59, Chapitre 8.3]. Soit A une matrice irréductible, primitive et non négative. Alors*

- *il existe une unique valeur propre réelle et positive r qui se situe sur le rayon spectral. Cette valeur propre s'appelle la* ***racine de Perron****;*
- *il existe un vecteur propre $A\mathbf{p} = r\mathbf{p}$ avec des entrées strictement positives. $\mathbf{p}$ est le seul vecteur propre non négatif de A. On pose que $\mathbf{p}$ est normalisé. On appelle ce vecteur le* ***vecteur de Perron****;*
- *A est primitive si et seulement si $\lim_{k\to\infty} \left(\frac{A}{r}\right)^k$ existe, auquel cas*

$$\lim_{k\to\infty} \left(\frac{A}{r}\right)^k = \frac{\mathbf{p}\mathbf{q}^T}{\mathbf{q}^T\mathbf{p}} > \mathbf{0}, \tag{1.31}$$

où $\mathbf{q}$ est le vecteur de Perron de la matrice A^T et $\mathbf{0}$ est la matrice nulle.

Dans le cas des chaînes de Markov, on peut identifier la racine de Perron comme étant la valeur propre $\lambda = 1$, et le vecteur de Perron comme étant la distribution stationnaire π^*. Donc, avec le théorème de Perron-Frobenius, on peut affirmer que si la matrice de transition P d'une chaîne de Markov est primitive, il existe une unique distribution stationnaire, et celle-ci représente une vraie distribution avec des entrées strictement positives. De plus, nous pouvons donner un critère pour la convergence de la chaîne vers sa distribution stationnaire :

Théorème 7. ***Théorème de convergence****. Soit une chaîne de Markov avec la matrice de transition P et la distribution stationnaire π^*. Si P est primitive, alors le vecteur d'état $\pi(t)$ convergera vers π^*, peu importe la distribution initiale $\pi(0)$. Plus exactement,*

$$\lim_{t\to\infty} \pi(t) = \pi^* \tag{1.32}$$

si et seulement si P est primitive.

Démonstration. La racine de Perron de P est $r = 1$, et son vecteur de Perron est π^*. La preuve du théorème 3 nous montre que P^T a le vecteur de Perron $\mathbf{q} = \mathbf{e}$, ainsi que la même racine que P. On s'intéresse au comportement de $\pi(t)$ pour $t \to \infty$. En appliquant l'équation (1.22) pour l'évolution de la chaîne, on trouve

$$\lim_{t\to\infty} \pi(t) = \lim_{t\to\infty} P^t \pi(0) = \lim_{t\to\infty} \left(\frac{P}{1}\right)^t \pi(0). \tag{1.33}$$

Comme P est primitive, on peut appliquer l'équation (1.31), ce qui donne

$$\lim_{t\to\infty} \left(\frac{P}{1}\right)^t = \frac{\pi^* \mathbf{e}^T}{\mathbf{e}^T \pi^*} = \pi^* \mathbf{e} \tag{1.34}$$

puisque $\mathbf{e}^T \pi^* = \sum_i \pi_i = 1$. Finalement,

$$\lim_{t\to\infty} \pi(t) = \lim_{t\to\infty} P^t \pi(0) = \pi^* \mathbf{e} \pi(0) = \pi^*, \tag{1.35}$$

puisque $\mathbf{e}\pi(0) = 1$. Ceci conclut la preuve. □

On peut observer que le théorème de convergence est une adaptation de la loi des grands nombres pour les chaînes de Markov. C'est d'ailleurs de cette manière que le résultat a été présenté par Markov lui-même en 1906 [76].

Les résultats présentés expliquent les observations faites lors de l'exemple de la marche aléatoire avec un saut. En effet, on peut montrer que la matrice Q de (1.24) est irréductible et que son spectre est $\sigma(Q) = \{1, -\frac{2}{3}, -\frac{1}{3}, 0\}$; elle est donc primitive. Son vecteur d'état convergera donc vers le vecteur de Perron de Q, qui est

$$\pi_Q^* = \begin{pmatrix} \frac{1}{5} \\ \frac{3}{10} \\ \frac{3}{10} \\ \frac{1}{5} \end{pmatrix}. \tag{1.36}$$

Pour la marche sans le saut, le vecteur de Perron est

$$\pi_P^* = \begin{pmatrix} \frac{1}{6} \\ \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{6} \end{pmatrix}, \tag{1.37}$$

mais la matrice P n'est pas primitive, puisqu'elle possède les valeurs propres 1 et -1. On ne peut pas directement appliquer le théorème de convergence, et l'expérience montre en effet qu'il n'y a pas de convergence. Il y aura plutôt une alternance entre deux vecteurs d'état qui

ne sont pas le vecteur de Perron. Dans le cas des matrices non primitives, la convergence se fait tout de même, mais il s'agit d'une convergence au sens de Cesàro, c'est-à-dire que la moyenne des sommes partielles sur $\pi(t)$ lorsque t tend vers l'infini donnera le vecteur de Perron π_P^* [59, Chapitre 8.4].

Pour obtenir une meilleure intuition sur les résultats présentés jusqu'à présent, la prochaine sous-section reverra les mêmes concepts présentés ici, mais en insistant sur la représentation des chaînes de Markov en tant que marches aléatoires sur un graphe.

1.5.5 Conditions pour une distribution stationnaire du point de vue des marches aléatoires sur graphe

Commençons d'abord par revoir les concepts de matrice irréductible et primitive du point de vue des chaînes de Markov. Les définitions et preuves de cette section sont majoritairement inspirées de [45, 54].

On commence par définir la probabilité $P(x_j|x_i)^t$ comme étant la probabilité de passer de l'état x_i à l'état x_j après t itérations de la chaîne.

Une chaîne de Markov est dite *irréductible* si $P(x_j|x_i)^t > 0$ pour un temps t fini et pour tout i, j. Autrement dit, la chaîne est irréductible si pour chaque paire d'états (x_i, x_j), il est possible de passer de l'état x_i à l'état x_j dans un temps fini. Si une chaîne n'est pas irréductible, elle est *réductible*.

Considérons un état x_i d'une chaîne de Markov. Soit $\mathcal{T}(x_i) = \{t : P(x_i|x_i)^t > 0\}$ l'ensemble des temps t auxquels la chaîne peut retourner à l'état x_i en partant de ce même état. La *période* de x_i, notée $p(x_i)$ est définie comme étant le plus grand commun diviseur de $\mathcal{T}(x_i)$.

À l'instar des graphes (voir section 1.1), une chaîne de Markov est dite *apériodique* si la période de tous ses états est 1. Autrement, elle est *périodique*.

Ces nouvelles définitions seront maintenant utilisées pour mettre en relation la matrice de transition de la chaîne et la représentation en marche aléatoire sur graphe :

Théorème 8. *Soit une chaîne de Markov avec un espace des états discrets Ω et une matrice de transition P. Soit le graphe orienté G avec l'ensemble de noeuds $V = \{x_i\}_{x_i \in \Omega}$, où il y a un lien de x_i vers x_j si et seulement si $P_{ji} > 0$. Alors les énoncés suivants sont équivalents :*

- *La chaîne de Markov est irréductible ;*
- *P est irréductible ;*
- *G est fortement connecté.*

Démonstration. Un graphe orienté est *fortement connecté* s'il existe un chemin pour se rendre au noeud x_j depuis le noeud x_i pour tout i, j. Un graphe orienté n'est pas fortement connecté

si et seulement s'il existe une permutation des noeuds telle que la matrice d'adjacence de G est triangulaire supérieure par blocs [11]. Donc, la matrice d'adjacence de G est irréductible si et seulement si G est fortement connecté. Comme la forme de la matrice d'adjacence de G est donnée par P, on trouve que G est fortement connecté si et seulement si P est irréductible.

Si G est fortement connecté, il existe un chemin de longueur finie partant du noeud x_i et arrivant au noeud x_j pour tout i, j. Par construction de G, ceci signifie qu'il existe un t tel que $P^t(x_j|x_i) > 0$. Donc, si G est fortement connecté, la chaîne est irréductible. Dans le sens inverse, si la chaîne est irréductible, il existe un t tel que $P^t(x_j|x_i) > 0$. Ceci signifie qu'il existe un chemin entre les noeuds x_j et x_i, et donc G est fortement connecté. On a donc montré que la chaîne est irréductible si et seulement si G est fortement connecté. □

La preuve précédente montre donc que pour savoir si une chaîne de Markov est irréductible, on peut regarder le graphe associé et voir s'il est fortement connecté. On veut maintenant mettre en relation la primitivité de la matrice de transition, la période de la chaîne et la période du graphe associé. Le prochain théorème est une première étape dans cette direction :

Théorème 9. *Soit une chaîne de Markov irréductible avec un espace des états Ω. Alors tous les états ont la même période.*

Démonstration. Adaptée de [78] [12]. Soit x_i et x_j deux états de la chaîne. Comme cette dernière est irréductible, il existe forcément un m et un n tels que $P^n(x_i|x_j) > 0$ et $P^m(x_j|x_i) > 0$. Dénotons par $p(x_i)$ la période de l'état x_i, c'est-à-dire le plus grand commun diviseur de $\mathcal{T}(x_i)$. On a que $P^{n+m}(x_i|x_i) \geq P^n(x_i|x_j)P^m(x_j|x_i) > 0$, c'est-à-dire que la probabilité que la chaîne parte de x_i et retourne à x_i en $n+m$ itérations est plus grande ou égale à la probabilité que la chaîne parte de x_i, passe par x_j, et retourne à x_i en $n+m$ itérations. En effet, il y a au moins autant de chemins dans le premiers cas que dans le deuxième (autrement dit, le premier terme inclut tous les chemins intermédiaires, incluant ceux qui ne passent pas par x_j). Donc, $n+m \in \mathcal{T}(x_i)$ et $p(x_i)$ est un diviseur de $n+m$. On peut appliquer le même raisonnement pour dire que $(m+n) \in \mathcal{T}(x_j)$.

Maintenant supposons qu'il existe un t tel que $P^t(x_j|x_j) > 0$. Alors

$$P^{n+m+t}(x_i|x_i) \geq P^n(x_i|x_j)P^t(x_j|x_j)P^m(x_j|x_i) > 0$$

et $(n+m+t) \in \mathcal{T}(x_i)$ et $p(x_i)$ est un diviseur de $(n+m+t)$. Comme $p(x_i)$ est un diviseur de $(n+m)$ et de $(n+m+t)$, on peut conclure qu'il est aussi un diviseur de t.

Pour récapituler, on a $(n+m), (n+m+t), t \in \mathcal{T}(x_i)$ et $(m+n), t \in \mathcal{T}(x_j)$. Par définition des périodes en tant que plus grand commun diviseur, ceci donne que $p(x_j) \geq p(x_i)$. Comme

11. Ceci correspond à avoir une partie du graphe qui ne peut pas communiquer avec une autre partie du graphe.
12. Il s'agit d'un site internet dont voici le lien.

le résultat est vrai pour tout i, j, on peut inverser les rôles de x_i et x_j, et finalement obtenir $p(x_i) = p(x_j)$ pour tout i, j. $\square$

À la vue de ce résultat, on conclut que pour une chaîne irréductible, on peut parler par abus de langage de la période de la chaîne, plutôt que de la période de ses états. Un corollaire important découlant directement de ce théorème et de la définition de l'apériodicité d'une chaîne est le suivant :

Théorème 10. *Une chaîne de Markov irréductible est apériodique si et seulement si un de ses états a une période de* 1.

Le prochain théorème est un théorème sur les matrices primitives qui sera utilisé pour la mise en relation entre la périodicité et la primitivité :

Théorème 11. *Soit A une matrice $n \times n$ irréductible et non négative quelconque. Alors A est primitive si et seulement s'il existe un $m \geq 1$ tel que A^m est une matrice strictement positive, soit $A^m > \mathbf{0}$.*

Démonstration. On note d'abord que le théorème de Perron-Frobenius pour les matrices strictement positives indique que la racine de Perron, qui se trouve sur le rayon spectral, est toujours unique[13] ; il s'agit de la seule valeur propre sur le rayon spectral [45, 59]. De plus, on note que l'élément (i, j) d'une matrice d'adjacence élevée à la k-ième puissance, $[A^k]_{ij}$, indique le nombre de chemins par lesquels il est possible de passer du noeud v_i au noeud v_j en suivant exactement k liens. Par exemple, si $[A^k]_{ij} = 2$, ceci veut dire qu'il y a deux chemins de longueur k qui partent du noeud v_i et se rendent au noeud v_j.

En premier lieu, considérons le cas où A^m est strictement positive. Soit le graphe non pondéré G formé à partir de la matrice A. Comme A^m est strictement positive, il existe au moins un chemin de longueur m entre chaque paire de noeud de G. Donc, G est fortement connecté, ce qui revient à dire que A est irréductible par le théorème 8. Ensuite, comme A^m est strictement positive, elle possède une unique valeur propre sur son rayon spectral $\rho(A)$. Soit le spectre de A donné par $\sigma(A) = \{\lambda_1, \lambda_2, \ldots, \lambda_n\}$. Alors le spectre de A^m est $\sigma(A^m) = \{\lambda_1^m, \lambda_2^m, \ldots, \lambda_n^m\}$. Si λ_1^m est la valeur propre de module $\rho(A^m)$, alors $\lambda_i^m < \rho(A)^m = \rho(A^m)$ pour tout $i \neq 1$, ce qui implique que $\lambda_i < \rho(A)$. Donc, λ_1 est la seule valeur propre sur le rayon spectral de A, ce qui montre que A est primitive.

Par la suite, considérons le cas où A est primitive. Dans ce cas, par le théorème 6, la limite $\lim_{m\to\infty} \left(\frac{A}{\rho(A)}\right)^m = \frac{\mathbf{p}\mathbf{q}^T}{\mathbf{q}^T\mathbf{p}} > \mathbf{0}$ existe, ce qui implique qu'il existe un m tel que $A^m > \mathbf{0}$. $\square$

13. On perd l'unicité lorsque l'on considère les matrices non négatives, c'est-à-dire lorsque l'on enlève la contrainte de positivité stricte. C'est ce qui nous oblige à ajouter le concept de matrices primitives.

Les outils nécessaires ayant tous été présentés, on peut maintenant montrer la relation entre la primitivité des matrices de transition et la périodicité de la chaîne et de son graphe :

Théorème 12. *Soit une chaîne de Markov irréductible avec un espace d'états Ω et une matrice de transition P. Soit le graphe orienté G avec l'ensemble de noeuds $V = \Omega$, où il y a un lien de x_i vers x_j si et seulement si $P_{ji} > 0$. Alors les énoncés suivants sont équivalents :*

- *La chaîne de Markov est apériodique ;*
- *P est primitive ;*
- *G est apériodique.*

Démonstration. Commençons par l'équivalence entre le graphe et la chaîne : pour chaque noeud x_i, on définit l'ensemble $L_i = \{\ell_1^{(i)}, \ell_2^{(i)}, \ldots\}$ qui contient toutes les longueurs des cycles de x_i. On note que L_i est une liste infinie, puisque les multiples entiers des longueurs sont aussi inclus. On définit ensuite g_i comme étant le plus grand commun diviseur de L_i. Par construction, l'ensemble L_i pour les noeuds de G est équivalent à l'ensemble $\mathcal{T}(x_i)$ pour les états de la chaîne de Markov, et $g_i = p(x_i)$ pour tout i. Donc, G est apériodique si et seulement si la chaîne de Markov est apériodique.

Le reste de la preuve sera consacré à l'équivalence entre l'apériodicité d'un graphe et la primitivité de P. On commence en supposant que P est primitive. Comme P est irréductible, on peut appliquer le théorème 8 pour dire que G est fortement connecté. Ainsi, il existe un chemin de x_i à x_j, puis un chemin de x_j à x_i pour tout i, j et les ensembles L_i sont donc non vides pour tout i. Ensuite, le théorème 11 permet de dire que comme P est primitive, il existe un $m \geq 1$ tel que $P^m > \mathbf{0}$, et on aura que $P^{m+p} > \mathbf{0}$ pour tout $p \geq 0$. Ainsi, on a que $m + p \in L_i$ pour tout indice i (puisque que $P_{ii}^{m+p} > 0$). Le seul diviseur commun pour tous les $m + p \in L_i$ est 1, et donc $g_1 = g_2 = \ldots = g_n = 1$ et G est apériodique.

Maintenant, supposons que le graphe G est apériodique. Alors $g_1 = g_2 = \ldots = g_n = 1$, ce qui permet de dire que tous les L_i forment des *demi-groupes numériques*[14]. La condition de fermeture (sur l'addition) de ces demi-groupes implique que les L_i contiennent tous les entiers suffisamment grands. Donc, il existe un entier t_i tel que $P_{ii}^s > 0$ pour tout $s \geq t_i$. De plus, comme P est irréductible, il existe pour toute paire i, j un entier r_{ij} tel que $P_{ij}^r > 0$. On a

14. Un demi-groupe numérique est un sous-ensemble S des entiers non négatifs $\mathbb{N}_+$, muni de l'addition, et qui respecte les axiomes suivants :

1. 0 est contenu dans S ;
2. Le complément $\mathbb{N}_+ - S$ est fini ;
3. Si $x, y \in S$ alors $x + y \in S$.

On peut construire un demi-groupe numérique en partant d'un ensemble $A = \{n_1, n_2, \ldots, n_k\}$ d'entiers non négatifs en considérant tous les entiers de forme $n_1x_1 + n_2x_2 + \ldots + n_kx_k$ pour $x_i \in \mathbb{N}_+$. Un théorème important et que nous utilisons ici est le suivant : S est un demi-groupe numérique si et seulement si le plus grand commun diviseur de A est 1. Pour plus d'information, voir [73].

donc, pour $s \geq t$,

$$P_{ij}^{s+r} \geq P_{ii}^{s} P_{ij}^{r} > 0. \tag{1.38}$$

Ici, la première inégalité est justifiée par le fait que la probabilité de passer de l'état i à l'état j après $s+r$ itérations est plus élevée ou égale à la probabilité de d'abord parcourir un s-cycle de i puis se rendre à j en r itérations ; on peut se convaincre qu'il y a au moins autant de chemins dans le premier cas qu'il y a de chemins dans le deuxième. Finalement, en prenant $k' := \max_{i,j}\{t_i + r_{ij}\}$, on trouve que $k' \geq t_i + r_{ij}$ pour tout i, j et donc $P_{ij}^{k'} > 0$ pour tout i, j. Donc il existe un k' tel que $P^{k'} > \mathbf{0}$, ce qui implique par le théorème 11 que P est une matrice primitive. □

À la vue de ces résultats, on peut conclure que pour qu'une chaîne de Markov converge vers une distribution stationnaire qui est unique, elle doit être irréductible et apériodique. Du point de vue des graphes, on veut avoir un graphe fortement connecté, et apériodique. Si ces conditions sont respectées, on pourra en effet appliquer le théorème de convergence 7.

1.5.6 Distribution stationnaire uniforme

Dans le contexte du projet, on cherche un moyen d'échantillonner un ensemble de graphe. Il a été montré que pour les ensembles à contraintes rigides, on cherche à avoir un échantillonage qui respecte une distribution uniforme sur les graphes. Voyons donc comment une chaîne de Markov peut avoir une distribution de ce genre. Ceci permettra de développer un algorithme d'échantillonnage non biaisé.

On cherche une *distribution stationnaire uniforme* telle que

$$\pi^* = \frac{1}{|\Omega|}\mathbf{e}. \tag{1.39}$$

Les deux conditions pour l'existence d'une distribution stationnaire, soit l'irréductibilité et l'apériodicité de la chaîne sont déjà connues. Il ne reste donc qu'à trouver les bonnes conditions pour que cette distribution soit aussi uniforme. Le théorème suivant donne un premier indice pour construire le bon algorithme :

Théorème 13. *Soit une chaîne de Markov irréductible et apériodique, avec l'espace des états Ω et la matrice de transition P. Si P est doublement stochastique, le vecteur d'état convergera vers la distribution stationnaire uniforme.*

Démonstration. Une matrice *doublement stochastique* est une matrice dont les lignes et les colonnes somment à 1, soit

$$\sum_i A_{ij} = 1 \qquad \sum_i A_{ji} = 1 \qquad \forall j. \tag{1.40}$$

Dans le cas des matrices doublement stochastiques, le vecteur $\mathbf{e}$ est le vecteur de Perron de A et de A^T.

Comme P est primitive (par le théorème 12), on connaît la limite $\lim_{k\to\infty}\left(\frac{A}{r}\right)^k$. Pour un temps $t \to \infty$, on aura donc

$$\pi^* = \lim_{t\to\infty}\left(\frac{P}{1}\right)^t \pi(0) = \frac{\mathbf{e}\mathbf{e}^T}{\mathbf{e}^T\mathbf{e}}\pi(0) = \frac{\mathbf{e}\mathbf{e}^T}{|\Omega|}\pi(0) = \frac{1}{|\Omega|}\mathbf{e}. \tag{1.41}$$

□

Une condition suffisante pour avoir une distribution uniforme est donc d'avoir des probabilités de transition symétriques. En effet, comme P est stochastique par la gauche par construction, si elle est symétrique, on aura qu'elle sera aussi stochastique par la droite, ce qui indique qu'elle sera doublement stochastique. La symétrie est donc une condition suffisante pour le développement d'algorithme d'échantillonage, et elle a comme avantage majeur d'être plus simple à implémenter que la condition plus générale de double stochasticité. On peut finalement présenter les caractéristiques que l'on cherchera pour avoir une chaîne de Markov avec une distribution stationnaire uniforme :

Théorème 14. *Soit une chaîne de Markov avec matrice de transition P. La distribution stationnaire sera la distribution uniforme si*

1. *Les probabilités de transition sont symétriques ;*
2. *La chaîne est irréductible ;*
3. *La chaîne est apériodique.*

Démonstration. Les conditions 2 et 3 permettent l'application du théorème 7 qui assure une convergence, et la condition 1 permet l'application du théorème 13 qui assure la distribution uniforme. □

Du point de vue de la marche aléatoire sur graphes, la condition de symétrie s'impose différemment : On demande plutôt que le graphe soit régulier. Dans la littérature [35], le problème d'échantillonnage des graphes est décrit comme une marche aléatoire sur des graphes non pondérés, et c'est ce qu'on utilise ici aussi.

Théorème 15. *Soit une chaîne de Markov irréductible avec un espace des états Ω. Soit le graphe G avec l'ensemble de noeuds $V = \Omega$. Il y a un lien non pondéré depuis le noeud x_j vers le noeud x_i, noté $(x_j \to x_i)$, si $P(x_i|x_j) > 0$. Alors la chaîne de Markov est symétrique si et seulement si le graphe G est régulier.*

Démonstration. La marche aléatoire sur G est donnée par

$$P(x_i|x_j) = \begin{cases} 0 & \text{si} \quad (x_j \to x_i) \notin E \\ \frac{1}{d_{\text{out}}(x_j)} & \text{si} \quad (x_j \to x_i) \in E \end{cases} \tag{1.42}$$

où $d_{\text{out}}(x_j)$ est le degré sortant du noeud x_j.

On suppose d'abord que G est régulier. Alors $d_{\text{out}}(x_i) = d_{\text{out}}(x_j)$ pour tout i, j. Ceci implique que $P(x_i|x_j) = P(x_j|x_i)$, et donc la chaîne de Markov est symétrique.

On suppose ensuite que la chaîne de Markov est symétrique. Si x_j et x_i sont connectés, on a que $d_{\text{out}}^{-1}(x_i) = P(x_i|x_j) = P(x_j|x_i) = d_{\text{out}}^{-1}(x_j)$, et donc $d_{\text{out}}(x_i) = d_{\text{out}}(x_j)$. Cette propriété est transitive et s'applique donc pour tous les noeuds dans la même composante. Comme la chaîne est irréductible, G est fortement connecté et la transitivité s'applique à tous les noeuds, ce qui permet de conclure que G est régulier. □

Ce résultat est vrai pour les marches aléatoires sur des graphes non pondérés. Si on ajoute des poids sur les liens, on pourra retrouver la condition de double stochasticité (plutôt que d'avoir seulement la symétrie).

Pour résumer, dans la représentation en marche aléatoire sur graphe, les conditions du théorème 14 deviennent :

Théorème 16. *Soit une marche aléatoire sur le graphe G. Alors la distribution stationnaire de la marche est uniforme si*

1. *G est régulier ;*
2. *G est fortement connecté ;*
3. *G est apériodique.*

Démonstration. Les conditions 2 et 3 couplées avec les théorèmes 8 et 12 permettent l'application du théorème 7 qui assure une convergence, et la condition 1 couplée avec le théorème 15 permet l'application du théorème 13 qui assure la distribution uniforme. □

Lors du développement d'algorithmes MCMC d'échantillonnage des ensembles de graphe, il faudra toujours s'assurer que ces trois conditions soient respectées. La prochaine section présente un premier algorithme qui permet d'explorer le modèle des configurations, ainsi que les preuves que toutes ces conditions sont respectées.

1.6 Échantillonnage du modèle des configurations

Il existe deux principaux types d'algorithmes permettant de former des échantillons pour un ensemble aléatoire de graphes [23] :

Le premier type comprend les algorithmes de générations de graphes [10, 18, 28, 52, 90]. Dans ce type d'algorithme, un nouveau graphe est généré selon certaines règles à chaque itération. Ces algorithmes ne sont pas des chaînes de Markov, ils correspondent plutôt à des processus entièrement stochastiques, c'est-à-dire qu'on peut atteindre n'importe quel état (graphe) à n'importe quel moment, sans que cette probabilité ne soit basée sur l'état actuel. Il serait théoriquement possible d'utiliser des algorithmes de ce type pour le projet actuel, mais le développement de tels algorithmes est souvent ardu, et trop peu de travail à été fait jusqu'à présent dans la littérature pour déterminer quelles seraient les règles à respecter. De plus, l'idée de générer un nouveau graphe à chaque itération pourrait être coûteux d'un point de vue de calcul numérique, notamment pour de très grands graphes.

Les algorithmes qui seront privilégiés pour ce projet seront plutôt les algorithmes du deuxième type, qui sont basés sur le mélange de graphes [13, 14, 35, 65, 86]. Dans ce type d'algorithme, on part d'un graphe de base (l'état initial de la chaîne de Markov) et on procède à une série de transformations qui permettent de transformer notre graphe en un autre graphe qui fait partie de l'ensemble à explorer. Ces algorithmes sont de type MCMC, où les états de la chaîne de Markov correspondent aux différents graphes de l'ensemble, et où les transformations permettent les transitions entre ces différents états. Ainsi, l'algorithme MCMC sera un déplacement sur ce qu'on peut appeler un *graphe de graphes*.

D'un point de vue formelle, on peut définir une transformation F comme une fonction prenant en entrée un graphe G et redonnant un graphe G'. Il existe plusieurs types de transformations, notamment la suppression et la création de liens (où une transformation F_{uv} enlève ou ajoute le lien (u, v) de l'ensemble E du graphe), la reconnection d'un lien unique (une transformation F_{uvx} où un lien (u, v) est retiré et un lien (u, x) est ajouté) [15] ou encore l'échange de liens. Cette dernière transformation est celle qui sera privilégiée dans le projet.

1.6.1 Échange de liens

Un *échange de liens* est défini comme une transformation F_{uvxy} sur un graphe G prenant une paire de liens $(u, v); (x, y)$, qui la retire de l'ensemble E du graphe, et qui la remplace par une nouvelle paire $(u, x); (v, y)$. On note la transformation F_{uvxy} sur le graphe G avec $(u, v); (x, y) \rightarrow (u, x); (v, y)$. En général, lorsque l'on sélectionne une paire de liens au hasard, deux transformations sont possibles : Pour la paire de liens $(u, v); (x, y)$, on pourrait avoir l'échange $(u, v); (x, y) \rightarrow (u, x); (v, y)$ ou l'échange $(u, v); (x, y) \rightarrow (u, y); (v, x)$. Cette subtilité est très importante, et lorsque deux liens seront sélectionnés au hasard dans les algorithmes, il faudra toujours considérer les deux possibilités. Un exemple d'échange de liens est illustré à la figure 1.6

La principale propriété d'un échange de liens, et celle qui nous intéresse, est le fait que cette

15. Aussi connu sous le nom de *hinge flip* dans la littérature.

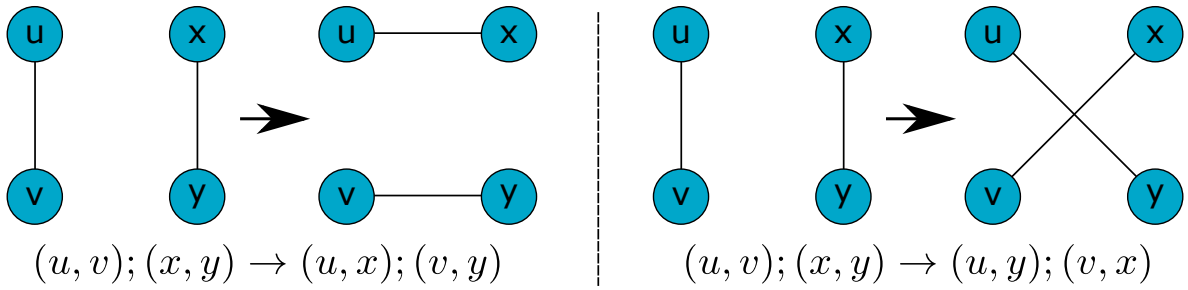

FIGURE 1.6 – Illustration des deux échanges de liens possibles pour une paire de liens $(u,v);(x,y)$. Il existe toujours les échanges F_{uvxy} et F_{uvyx}.

transformation conserve la séquence de degrés **d** d'un graphe. En effet, le nombre de voisins est conservé pour chacun des noeuds impliqués dans l'échange. Du point de vue des demi-liens, on peut tout simplement dire que le nombre de demi-liens sortant d'un noeud est conservé, et que les demi-liens dans l'échange sont connectés à de nouveaux demi-liens.

Finalement, on note que certains échanges de liens ne sont pas possible si l'on veut n'échantillonner que des graphes simples. En effet, si on essaie de faire un échange $(u,v);(u,x) \rightarrow (u,u);(v,x)$, on crée un graphe avec une boucle, ce qui nous sort de l'ensemble des graphes simples. De la même manière, on doit éviter de procéder à un échange $(u,v);(x,y) \rightarrow (u,x);(v,y)$ si le lien (u,x) ou (v,y) existe déjà, puisque l'on se retrouve dans l'ensemble des multigraphes.

1.6.2 Algorithme d'échanges de liens

L'échange de liens, du fait qu'il ne change pas la séquence de degrés d'un graphe, est la transformation par excellence pour explorer et échantillonner le modèle des configurations. On peut alors construire un algorithme de mélange de graphes qui utilisera les transformations de type F_{uvxy} pour trouver de nouvelles configurations de l'ensemble. Ce dernier est présenté à l'algorithme 2. L'algorithme d'échantillonnage proposé dans cette section est bien connu dans la littérature et a été revu à maintes reprises [15, 35].

Algorithme 2 MCMC pour le modèle des configurations

entrée : Graphe initial G_0
sortie : Échantillons de graphes $\{G_i\}$
  pour $i <$ taille de l'échantillon **faire**
    choisir deux liens (u,v) et (x,y) uniformément au hasard
    choisir au hasard l'échange $(u,v);(x,y) \rightarrow (u,x);(v,y)$ ou $(u,v);(x,y) \rightarrow (u,y);(v,x)$
    si l'échange crée une boucle ou un multilien **alors**
      $G_i \leftarrow G_{i-1}$
    sinon
      procéder à l'échange de liens pour obtenir G_i
    fin si

ajouter G_i à l'échantillon
fin pour

Comme expliqué précédemment, l'algorithme 2 part d'un graphe d'origine G_0, qui se trouve généralement être celui issu de données réelles et que l'on veut comparer au modèle. À chaque itération, deux liens seront choisis uniformément au hasard et un échange de lien sera tenté entre ceux-ci. Si l'échange ne crée pas de boucle ou de multiliens, c'est-à-dire si on reste dans l'espace des graphes simples, alors on obtient un nouveau graphe G_i que l'on ajoute à l'échantillon. Autrement, l'échange est refusé, et on échantillonne à nouveau le graphe actuel (pour l'itération i on échantillonne donc le graphe G_{i-1} de l'itération précédente). Au final, un tel algorithme permettra d'obtenir un ensemble de graphes qui ont tous la même séquence de degrés, et le nombre de fois que ces graphes apparaîtront dans l'échantillon permettra d'estimer leur probabilité d'être obtenus dans l'ensemble, c'est-à-dire $P(G) = \frac{\text{Nombre d'apparition de } G \text{ dans l'échantillon}}{\text{Taille de l'échantillon}}$.

On sait que l'échange de liens conserve la séquence de degrés et donc que tous les graphes obtenus avec l'algorithme 2 feront partie du modèle des configurations de G_0. Cependant, il reste encore à déterminer si l'ensemble exploré correspond bel et bien au modèle des configurations. Autrement dit, on veut savoir si l'algorithme explore l'entièreté du modèle des configurations et ce, avec une distribution stationnaire uniforme.

Dans la littérature [35], l'algorithme est décrit comme une marche aléatoire sur un *graphe de graphes* $\mathcal{G}$, c'est-à-dire une marche sur un graphe orienté dont l'ensemble de noeuds V correspond au modèle des configurations $\Gamma_{CM}(G_0)$ et où un lien de G_i vers G_j pour $G_i, G_j \in V$ existe si et seulement s'il existe un échange de liens permettant de passer de G_i à G_j. Comme la paire de liens à échanger est choisie uniformément au hasard, la marche est décrite par

$$P(G_j|G_i) = \begin{cases} 0 & \text{si} \quad (G_i \to G_j) \notin E \\ \frac{1}{\deg(G_i)} & \text{si} \quad (G_i \to G_j) \in E \end{cases} \tag{1.43}$$

pour E l'ensemble de liens de $\mathcal{G}$. On peut appliquer le théorème 16 sur cette marche, ce qui indique que pour avoir une distribution stationnaire uniforme, on veut montrer que $\mathcal{G}$ est régulier, fortement connecté, et apériodique. Les preuves pour les trois conditions seront présentées ici, puisque ces dernières forment la base des preuves appliquées sur les algorithmes développés durant le projet.

Théorème 17. *Le graphe de graphes $\mathcal{G}$ est fortement connecté.*

Les preuves de connectivité sont souvent celles qui posent problème dans les chaînes de Markov [54]. De fait, une preuve complète ne peut être présentée ici. Il existe cependant de nombreuses preuves pour montrer que les échanges de liens permettent d'explorer tous les graphes d'un modèle des configurations dans la littérature. Il existe notamment des preuves directes [55, 92], des preuves par induction [85] et des preuves par contradiction [94]. Une preuve plus simple

pour les ensembles étiquetés sur les demi-liens est présentée dans [35], et donne une bonne idée du genre de raisonnement que l'on peut utiliser.

Théorème 18. *Le graphe de graphes $\mathcal{G}$ est apériodique.*

Démonstration. On note d'abord que les échanges de liens sont des transformations réversibles, c'est-à-dire que si l'on effectue l'échange $(u, v); (x, y) \to (u, x); (v, y)$ pour passer de G_i à G_j, on peut tout simplement effectuer l'échange inverse $(u, x); (v, y) \to (u, v); (x, y)$ pour retourner à G_i. Ceci correspond à un cycle de longueur 2 dans $\mathcal{G}$.

On peut aussi trouver un cycle de longueur 3 pour tous les graphes si l'on considère la séquence suivante : Pour quatre noeuds distincts $u \neq v \neq x \neq y$, on procède à l'échange $(u, v); (x, y) \to (u, x); (v, y)$, à l'échange $(u, x); (v, y) \to (u, y); (v, x)$, puis on retourne à la configuration originale avec l'échange $(u, y); (v, x) \to (u, v); (x, y)$. On demande que les noeuds soient distincts pour éviter la création de boucles, qui serait refusée par l'algorithme. Ces noeuds distincts existent pour les graphes de 4 noeuds ou plus avec au moins 2 liens, donc pour tous les graphes que l'on considère. Il faut aussi considérer le cas où la création de multiliens empêcheraient tout échange dans les graphes. Dans ce cas, le graphe d'origine pour l'algorithme serait un graphe complet K_n, et il n'existe qu'une seule configuration qui lui est associée, plus précisément $\{K_n\} = \Gamma_{CM}(K_n)$.

Donc, pour tous les graphes non triviaux, il existe un cycle de longueur 2 et un cycle de longueur 3. Le plus grand commun diviseur de ces deux nombres est 1, et donc chaque composante connectée de $\mathcal{G}$ est apériodique. Comme $\mathcal{G}$ est fortement connecté (du théorème 17), $\mathcal{G}$ est apériodique. □

Une preuve alternative pour l'apériodicité sera présenté au chapitre 2, et cette dernière pourrait être appliquée ici. Nous choisissons tout de même de présenter cette preuve pour des raisons qui seront détaillées à ce moment et dans l'annexe B.

Théorème 19. *Le graphe de graphes $\mathcal{G}$ est régulier.*

Démonstration. Pour qu'un graphe orienté soit régulier, tous les noeuds doivent avoir le même degré entrant d_{in} et le même degré sortant d_{out}.

L'algorithme rééchantillonne le même graphe lorsqu'un échange crée une boucle ou un multilien. Ceci correspond à une boucle dans le graphe de graphes, c'est-à-dire un lien qui part de G_i et revient à G_i. Une telle boucle contribue 1 au degré sortant de G_i et 1 à son degré entrant. Autrement, un échange permis correspond à un chemin unique qui part du noeud G_i et qui arrive à un noeud G_j. Ceci contribue 1 au degré sortant de G_i. Comme l'échange est réversible (comme mentionné à la preuve du théorème 18), ce dernier contribue aussi 1 au degré entrant de G_i.

Donc, chaque paire de liens contribue 1 au degré sortant et au degré entrant, que l'échange soit permis ou non par l'algorithme. Pour un graphe G_i, il existe $\binom{m_i}{2}$ paires de liens, où m_i correspond au nombre de liens de G_i. Chaque paire de liens peut être échangée de 2 manières différentes, donc, $d_{in}(G_i) = d_{out}(G_i) =: d(G_i) = 2\binom{m_i}{2}$. Finalement, comme tous les graphes explorés par l'algorithme ont le même nombre de liens, $m_i = m$ pour tout i et donc $d(G_i) = 2\binom{m}{2}$ pour tous les noeuds G_i de $\mathcal{G}$. □

Donc, l'algorithme 2 correspond à une marche aléatoire sur un graphe de graphes régulier, fortement connecté et apériodique. On peut donc invoquer le théorème 16, qui nous dit que la distribution de l'échantillon obtenu tendra vers la distribution uniforme pour un grand nombre d'itérations. La vitesse de convergence, c'est-à-dire le nombre d'itérations nécessaires pour obtenir l'équilibre, est encore à ce jour inconnu. Cependant, quelques outils statistiques ont récemment été développés pour déterminer à quel moment la chaîne arrive à son état d'équilibre [29].

Jusqu'à présent, une méthode d'analyse a été présentée pour explorer les ensembles de graphes à contraintes rigides de manière uniforme. Cette méthode sera réutilisée dans le prochain chapitre, qui présentera de nouveaux ensembles à contraintes rigides qui devront être aussi explorés. Avant cela cependant, il reste encore à développer une méthode permettant d'échantillonner les ensembles à contraintes souples. La prochaine section se concentrera sur le type d'algorithme qui nous sera utile.

1.7 Balance détaillée et Metropolis-Hastings

Une propriété qui n'a pas été mentionnée jusqu'à présent, mais qui est bien connue dans les chaînes de Markov, est ce qu'on appelle leur *réversibilité*. Une chaîne de Markov est dite *réversible* si elle respecte les *équations de balance détaillée*. Soit π_i l'entrée du vecteur d'état π qui correspond à l'état x_i. Alors les équations de balance détaillée sont

$$\pi_i P(x_j|x_i) = \pi_j P(x_i|x_j). \tag{1.44}$$

pour tout i et j. Donc, si la chaîne est dans une distribution π qui respecte ces équations, elle sera réversible.

La balance détaillée est un concept qui a d'abord été développé en physique statistique. Ce dernier permet d'introduire une symétrie en temps dans un système statistique. Plus précisément, un système réversible est un système qui se comporte de la même manière dans une direction du temps que dans la direction inverse. On pourrait par exemple imaginer un échange de particules entre différentes boîtes. Si le système n'est pas à l'équilibre, on peut déterminer la direction du temps en observant un changement d'entropie dans une des boîtes : pour des boîtes communiquant entre elles à température constante, on s'attendrait à avoir un

plus grand flux de particules vers la boîte de plus grand volume. Ce changement représenterait une augmentation d'entropie si on va dans la direction normale du temps, et une diminution si on va dans la direction opposée.

Cependant, lorsque le système atteint l'équilibre et que la condition de balance détaillée est respectée, il est impossible de déterminer la direction du temps. En effet, la condition demande que le flux d'échange soit le même entre chacune des boîtes ; si on observe spécifiquement une boîte A et une boîte B, le flux d'échange de A vers B sera le même que le flux de B vers A. C'est cette égalité des deux flux, étalée à l'entièreté du système, qui empêche de déterminer la direction du temps et donc qui amène la propriété de réversibilité au système.

C'est le même phénomène qui se produit dans les chaînes de Markov réversible. Plus exactement, on parlera du « flux de probabilité », qui doit être symétrique entre chaque états de la chaîne. La balance détaillée est une condition supplémentaire sur une chaîne de Markov, qui n'est pas nécessaire pour avoir un état stationnaire. Cependant, une distribution π qui respecte les équations (1.44) est aussi une distribution stationnaire.

Pour se convaincre de ce résultat, on considère d'abord l'équation (1.26) pour une distribution stationnaire $\pi^* = P\pi^*$. En explicitant la notation matricielle, on peut obtenir $|\Omega|$ équations qui décrivent chacune un seul état de la chaîne :

$$\pi_i^* = \sum_j \pi_j^* P(x_i|x_j) \tag{1.45}$$

pour tout i. On appelle ce système d'équations les *équations de balance*, et ces dernières représentent une autre manière d'écrire la distribution stationnaire d'une chaîne de Markov ; si une distribution π respecte ces équations pour une chaîne donnée, alors π est la distribution stationnaire π^*.

Maintenant, si l'on prend les équations (1.44) et que l'on somme sur tous les états, on obtient

$$\begin{aligned}
\sum_j \pi_i P(x_j|x_i) &= \sum_j \pi_j P(x_i|x_j) \\
\pi_i \sum_j P(x_j|x_i) &= \sum_j \pi_j P(x_i|x_j) \\
\pi_i \cdot 1 &= \sum_j \pi_j P(x_i|x_j) \\
\pi_i &= \sum_j \pi_j P(x_i|x_j),
\end{aligned}$$

ce qui correspond aux équations (1.45) et montre que π est la distribution stationnaire π^*. Donc, respecter la balance détaillée est une condition suffisante pour être la distribution stationnaire. Les équations de balance détaillées sont parfois utilisées pour faciliter la recherche d'une distribution stationnaire dans une chaîne de Markov. En effet, comme elles demandent une restriction supplémentaire et qu'on se retrouve avec $\binom{|\Omega|}{2}$ équations à la place de $|\Omega|$, on

peut utiliser de nouvelles techniques pour solutionner le système d'équations et trouver la solution π^*.

1.7.1 Metropolis-Hastings

Pour échantillonner les ensembles à contraintes rigides, on cherchait à avoir une distribution stationnaire uniforme. En effet, cette solution est la solution maximalement entropique, c'est-à-dire celle qui suppose le moins de contraintes sur l'ensemble et donc qui est la plus « aléatoire ». Pour les ensembles à contraintes souples cependant, ce n'est plus le cas ; dans ces ensembles, la distribution des probabilités sera dictée par l'équation (1.9). Donc, pour ajouter une contrainte souple au modèle des configurations, il sera nécessaire d'introduire un biais dans l'algorithme que l'on a déjà développé.

On peut faire exactement cela si on introduit des *probabilités d'acceptation* dans l'algorithme 2. La probabilité $P(G'|G)$ est pour l'instant déterminée uniquement par la probabilité $g(G'|G)$ de choisir la transformation menant de G à G'. Pour introduire un biais, il suffit alors d'ajouter la probabilité $\mathcal{A}(G'|G)$ d'accepter ou non la transformation proposée. La nouvelle probabilité de transition est alors donnée par

$$P(G'|G) = g(G'|G)\mathcal{A}(G'|G) \tag{1.46}$$

ou par

$$P(x_j|x_i) = g(x_j|x_i)\mathcal{A}(x_j|x_i) \tag{1.47}$$

si on utilise la notation pour une chaîne de Markov quelconque.

Un algorithme très utilisé dans la communauté scientifique et qui utilise de telles probabilités d'acceptation est l'algorithme dit de *Metropolis-Hastings*. Ce dernier utilise les probabilités d'acceptations suivantes :

$$\mathcal{A}(x_j|x_i) = \min\left\{1, \frac{P(x_j)g(x_i|x_j)}{P(x_i)g(x_j|x_i)}\right\}. \tag{1.48}$$

Le choix de ces probabilités (1.48) se justifie de deux manières. D'abord, ce choix de probabilité permet de respecter les équations de balance détaillée (1.44). En effet, en insérant (1.48) dans (1.44), on obtient

$$\begin{aligned}
\pi_i P(x_j|x_i) &= \pi_j P(x_i|x_j) \\
\pi_i g(x_j|x_i)\mathcal{A}(x_j|x_i) &= \pi_j g(x_i|x_j)\mathcal{A}(x_i|x_j) \\
\pi_i g(x_j|x_i)\min\left\{1, \frac{P(x_j)g(x_i|x_j)}{P(x_i)g(x_j|x_i)}\right\} &= \pi_j g(x_i|x_j)\min\left\{1, \frac{P(x_i)g(x_j|x_i)}{P(x_j)g(x_i|x_j)}\right\}.
\end{aligned}$$

Ici, on peut supposer sans perte de généralité que $P(x_i)g(x_j|x_i) > P(x_j)g(x_i|x_j) > 0$, ce qui donne que

$$\begin{aligned}
\pi_i g(x_j|x_i)\frac{P(x_j)g(x_i|x_j)}{P(x_i)g(x_j|x_i)} &= \pi_j g(x_i|x_j)\cdot 1 \\
\pi_i \frac{P(x_j)}{P(x_i)} &= \pi_j \\
\pi_i P(x_j) &= \pi_j P(x_i).
\end{aligned}$$

On remarque d'abord que l'on peut voir la forme de la distribution stationnaire en sommant sur tous les états dans l'équation précédente :

$$\begin{aligned}
\sum_j \pi_i P(x_j) &= \sum_j \pi_j P(x_i) \\
\pi_i \sum_j P(x_j) &= P(x_i)\sum_j \pi_j \\
\pi_i &= P(x_i).
\end{aligned}$$

En insérant ce résultat dans le développement , on trouve finalement que

$$\begin{aligned}
\pi_i P(x_j) &= \pi_j P(x_i) \\
P(x_i)P(x_j) &= P(x_j)P(x_i) \\
1 &= 1
\end{aligned}$$

ce qui est une tautologie et montre que les équations de balance détaillée sont toujours respectées par une chaîne de Markov utilisant les probabilités d'acceptation d'un Metropolis-Hastings. Comme la balance détaillée est obtenue, on sait que l'algorithme pourra converger vers la distribution stationnaire $\pi_i^* = P(x_i)$ sans problème.

Comme dans notre cas, les probabilités $g(x_j|x_i)$ de choisir une transformation sont toujours les mêmes pour tout i, j (puisque l'on choisit les transformations uniformément au hasard), on peut simplifier la notation pour avoir

$$\mathcal{A}(x_j|x_i) = \min\left\{1, \frac{P(x_j)}{P(x_i)}\right\}. \tag{1.49}$$

Il existe d'autres probabilités d'acceptation qui respectent la balance détaillée par exemple les probabilité de Glauber

$$\mathcal{A}(x_j|x_i) = \frac{P(x_j)}{P(x_j) + P(x_i)}. \tag{1.50}$$

Cependant, elle ne sont pas aussi pratiques que les probabilités de Metropolis-Hastings car elles n'impliquent pas le ratio des probabilités $P(x_j)P(x_i)$. Il s'agit de la deuxième raison qui explique pourquoi on préfère les probabilités (1.48). En effet, il est parfois ardu de calculer exactement la distribution de probabilité pour chaque graphe de l'ensemble. Le fait que l'on

puisse obtenir la bonne distribution stationnaire en connaissant uniquement le ratio des densités de probabilité est extrêmement utile, et c'est ce qui fait la popularité des algorithmes de Metropolis-Hastings. C'est aussi le cas pour ce projet-ci, puisque le fait de calculer uniquement le ratio entre les probabilités simplifiera grandement le calcul à effectuer lors de l'ajout des contraintes souples.

1.8 Modèle des configurations corrélées

Cette section présente deux nouveaux modèles de graphes qui sont connus et utilisés dans la littérature [2, 26, 64, 65, 82]. Il s'agit des *modèles des configurations corrélées* (CCM [16]), qui sont définis de la manière suivante :

Soit un graphe G avec un ensemble de noeuds $V = \{v_i\}_{i=1}^n$, une séquence de degrés $\mathbf{d} = \{d_i\}_{i=1}^n$ et une corrélation degré à degré donné par la matrice conjointe e. Alors le *modèle des configurations corrélées rigide* de G, noté $\Gamma^r_{\text{CCM}}(G)$, est l'ensemble ayant une contrainte rigide sur la séquence de degrés $\mathbf{d}$ et la matrice conjointe e.

De la même manière, le *modèle des configurations corrélées souple* de G, noté $\Gamma^s_{\text{CCM}}(G)$, est l'ensemble ayant une contrainte rigide sur la séquence de degrés $\mathbf{d}$ et une contrainte souple sur la matrice conjointe e.

Ces deux ensembles sont évidemment des sous-ensembles du CM auxquels on a ajouté une contrainte rigide ou souple. Pour les explorer, on peut encore utiliser les échanges de liens, qui assureront toujours la conservation de la séquence de degrés à chaque itération. Il faudra cependant modifier l'algorithme 2 afin de respecter en même temps les nouvelles contraintes. Le cas des contraintes souples est traité en premier, puisque ce dernier nécessite simplement l'application du Metropolis-Hastings présenté à la section 1.7.

1.8.1 Échantillonage du CCM souple

Pour pouvoir appliquer la formule (1.49) pour les probabilités d'acceptation, il faut d'abord déterminer quelle est la distribution $P(A)$ que l'on cherche pour une matrice e tirée d'un graphe A^* donné. Pour trouver exactement cette distribution, on voudrait utiliser la méthode des multiplicateurs de Lagrange pour maximiser l'entropie. Cependant, le calcul est très complexe et il n'existe pas de solution exacte dans la littérature pour l'instant. Il sera donc préférable d'utiliser une distribution approximative, qui a été proposé par Newman en 2002 [64, 65]. La distribution est la suivante :

$$P(A) = \prod_{i=1}^{n} \prod_{j=1}^{i} e(d_i, d_j)^{A_{ij}} \tag{1.51}$$

16. Acronyme de l'anglais *Correlated Configuration Model*.

où d_i représente le degré du noeud i dans le graphe A (et non dans le graphe A^*). Comme l'élément $e(d_i, d_j)$ indique la fraction de liens de type (d_i, d_j), cette formule implique que la distribution est obtenue en multipliant toutes les probabilités d'obtenir un lien de type (d_i, d_j) en tirant un lien au hasard parmi tous les liens, et ce pour chacun des liens du graphes. L'exposant A_{ij} permet tout simplement d'ignorer les liens qui ne sont pas dans le graphe. Par construction, cette distribution correspond à un ensemble canonique duquel on peut tirer des graphes qui auront en moyenne la matrice de degré joint e. On remarque de cette distribution que la séquence de degrés n'est pas fixée exactement ; il s'agit plutôt de la distribution de degrés qui est dictée par la matrice e. Cette approximation est cependant négligeable pour de grands graphes, et pour un $n \to \infty$, on obtient exactement le bon ensemble [65].

Soient deux graphes A et A' qui sont séparés par un échange de liens $(u, v); (x, y) \to (u, x); (v, y)$. Alors le ratio des probabilités entre ces deux graphes sera

$$\frac{P(A')}{P(A)} = \frac{e(d_u, d_x)e(d_v, d_y)}{e(d_u, d_v)e(d_x, d_y)}. \tag{1.52}$$

La probabilité d'acceptation pour procéder à l'échange entre ces deux graphes sera donc

$$\mathcal{A}(A'|A) = \min\left\{1, \frac{e(d_u, d_x)e(d_v, d_y)}{e(d_u, d_v)e(d_x, d_y)}\right\}. \tag{1.53}$$

Comme montré précédemment, ces probabilités respectent les équations de balance détaillée et permettront donc une convergence vers la distribution désirée. On note qu'avec la division, il est possible de modifier la matrice e pour qu'elle contienne le nombre au lieu de la fraction de liens entre les différents types.

Finalement, l'algorithme pour échantillonner le CCM souple est présenté à l'algorithme 3. Il s'agit, comme mentionné, d'une modification de l'algorithme 2 avec les probabilités d'acceptation (1.53).

Algorithme 3 MCMC pour le CCM souple

entrée : Graphe initial G_0
sortie : Échantillons de graphes $\{G_i\}$
  calculer la matrice de degrés conjoints e
  pour $i <$ taille de l'échantillon **faire**
    choisir deux liens (u, v) et (x, y) uniformément au hasard
    choisir au hasard l'échange $(u, v); (x, y) \to (u, x); (v, y)$ ou $(u, v); (x, y) \to (u, y); (v, x)$
    si l'échange crée une boucle ou un multilien **alors**
      $G_i \leftarrow G_{i-1}$
    sinon
      tirer un nombre $0 \leq r < 1$ uniformément au hasard
      si $r \leq \min\left\{1, \frac{e(d_u,d_x)e(d_v,d_y)}{e(d_u,d_v)e(d_x,d_y)}\right\}$ [17] **alors**

17. ou si $r \leq \min\left\{1, \frac{e(d_u,d_y)e(d_v,d_x)}{e(d_u,d_v)e(d_x,d_y)}\right\}$, dépendamment de l'échange choisi.

procéder à l'échange de liens pour obtenir G_i
sinon
$G_i \leftarrow G_{i-1}$
fin si
fin si
ajouter G_i à l'échantillon
fin pour

1.8.2 Échantillonnage du CCM rigide

L'échantillonnage de la version rigide du CCM demande plus de modifications à l'algorithme 2 que pour la version souple. La stratégie adoptée est la suivante : on procède encore à des échanges de liens, mais les échanges qui sont proposés sont seulement ceux qui ne changent pas la matrice de degrés conjoints.

Il est possible de montrer qu'avec cette stratégie, on obtient un graphe de graphes qui est régulier, fortement connecté et apériodique [2, 26, 82]. Donc, un algorithme d'échange de liens qui ne propose que les échanges qui ne modifient pas la matrice de degrés conjoints convergera aussi vers la distribution uniforme, et peut être utilisé pour échantillonner uniformément le CCM rigide.

Algorithme 4 MCMC pour le CCM rigide

entrée : Graphe initial G_0
sortie : Échantillons de graphes $\{G_i\}$
créer les ensembles E_d et leur attribuer une probabilité $P(E_d)$ proportionnelle au nombre de liens dans chaque ensemble
pour $i <$ taille de l'échantillon **faire**
choisir un ensemble E_d avec probabilité $P(E_d)$.
choisir deux liens (u, v) et (x, y) uniformément au hasard parmi les liens de E_d
si $d_u = d_v$ ou $d_x = d_y$ **alors**
choisir un des deux échanges possibles au hasard
sinon
choisir le seul échange approprié
fin si
si l'échange crée une boucle ou un multilien **alors**
$G_i \leftarrow G_{i-1}$
sinon
procéder à l'échange de liens pour obtenir G_i
fin si
ajouter G_i à l'échantillon
fin pour

Maintenant, il faut déterminer comment on peut proposer uniquement les liens qui ne changent pas la corrélation. La première étape est de créer des sous-ensembles de liens qui contiennent tous les liens connectés à des noeuds de degré d, que l'on peut appeler E_d, et de compter leur nombre de liens associé. On choisit ensuite un des ensembles E_d avec une probabilité proportionnelle à leur nombre de liens, puis on sélectionne une paire de liens uniformément au hasard dans cet ensemble. Cette étape est équivalente à choisir d'abord un lien uniformément au hasard, de choisir par la suite un groupe E_d qui lui est associé, puis de choisir un deuxième lien qui fait partie de ce groupe E_d. La paire de liens ainsi formée peut alors être échangée.

En général, un seul des deux échanges de la paire peut être effectué ; l'autre échange change la corrélation de degrés. Soit des noeuds u, v, x, y où u et x ont le même degré, soit $d_u = d_x = d$. On cherche à effectuer l'échange $(u, v); (x, y) \rightarrow (u, y); (v, x)$ qui correspond en terme de degrés à un échange $(d, d_v); (d, d_y) \rightarrow (d, d_y); (d, d_v)$, et ne change donc pas la matrice e puisqu'il reste encore un lien (d, d_v) et un lien (d, d_y). L'autre échange donnerait des liens $(d_y, d_v); (d, d)$, ce qui pourrait changer le nombre de liens de chaque type et donc la matrice de corrélation. Il est possible (par exemple dans le cas où le degré de tous les noeuds est le même) que deux échanges soient possibles. Dans ce cas, on choisit un des deux uniformément au hasard.

L'algorithme qui est fourni dans le cadre du projet est décrit en détail dans l'algorithme 4.

Les deux algorithmes pour échantillonner les CCMs existent déjà dans la littérature [26, 64, 65, 82]. Ils ont été refaits dans le cadre du projet de maîtrise puisque les corrélations de types seront conservées dans les ensembles définis au prochain chapitre. Les algorithmes 3 et 4 sont en quelque sorte une base qui permettra de mieux décrire les algorithmes pour les nouveaux ensembles.

Chapitre 2

Modèles de graphes avec séquence de centralité fixe

Les notions de base ayant été présentées, il est maintenant possible de débuter une description plus détaillée du projet de recherche. Le but de ce dernier est de contrer les limitations du modèle des configurations en offrant des modèles de graphes alternatifs et plus contraints. Les limitations du modèle des configurations viennent notamment du fait que la contrainte rigide sur la séquence de degrés est une contrainte locale, c'est-à-dire qui se base uniquement sur les voisins des noeuds. Un modèle nul avec seulement des contraintes locales perd beaucoup de structure à une échelle moyenne ou globale, ce qui empêche le modèle de reproduire de nombreuses caractéristiques que l'on observe dans les réseaux réels, par exemple la connectivité ou la structure en communautés. Les modèles des configurations corrélées présentés à la section 1.8 sont des premières tentatives d'obtenir des ensembles de graphes plus réalistes, de même que les ensembles qui conservent la connectivité [46] ou le coefficient d'agrégation [61, 67]. Le présent projet s'inscrit dans cette tendance ; en développant des modèles de graphes avec une contrainte sur la décomposition en oignon, il sera possible de conserver une information non seulement à une échelle microscopique (règles locales), mais aussi à une échelle macroscopique (en conservant l'organisation des coeurs).

2.1 Modèle des configurations étagées

Soit un graphe G avec un ensemble de noeuds $V = \{v_i\}_{i=1}^n$, une séquence de degrés $\mathbf{d}$, une séquence de coeurs $\mathbf{c}$ et une séquence de couches $\mathbf{OD}$. Le *modèle des configurations étagées* de G (LCM [1]), noté $\Gamma_{LCM}(G)$, est l'ensemble généré par une contrainte rigide sur la séquence de degrés $\mathbf{d}$, la séquence de coeurs $\mathbf{c}$ et la séquence de couches $\mathbf{OD}$.

Ce modèle est un sous-ensemble du modèle des configurations dans lequel tous les graphes

1. de l'anglais *Layered Configuration Model*.

auront la même décomposition en oignon. Comme mentionné précédemment, ceci a pour but de conserver une structure plus générale que seulement la séquence de degrés du graphe, et peut donc reproduire des caractéristiques plus générales du graphe de base.

Comme la décomposition en oignon est récente, il n'existe toujours pas d'algorithme de type génératif pour échantillonner cet ensemble de graphes : on optera plutôt pour l'approche du mélange de graphes. En conséquence, comme on avait fait pour le CCM rigide, on choisit de modifier l'algorithme d'échange de liens 2 pour respecter les nouvelles contraintes.

2.1.1 Algorithme d'échantillonnage

Pour échantillonner l'ensemble de manière efficace, on veut idéalement avoir un algorithme qui propose uniquement les échanges qui ne changent pas la décomposition en oignon des graphes. Cependant, l'application directe de cette stratégie pour le LCM n'a pas encore été élucidée. Pour une configuration de graphe donné, la manière de diviser les liens entre différents ensembles (comme les ensembles E_d pour le CCM) n'est pas connue, et si ces ensembles existent, il pourrait être trop coûteux de les calculer numériquement.

Pour contourner cette difficulté, la stratégie dite de *switch and hold* sera adoptée [6]. Cette stratégie inverse tout simplement l'ordre actuel de proposer les bons échanges et de les effectuer ; dans le *switch and hold*, on commence par proposer un échange quelconque, et on le refuse si ce dernier ne respecte pas les contraintes que l'on donne. Lorsqu'un échange est refusé, on échantillonne à nouveau la configuration de l'itération précédente. C'est cette stratégie qui a déjà été adoptée dans l'algorithme 2 pour empêcher la création de boucle et de multiliens. Cette stratégie est plus facile à adopter, mais elle a l'inconvénient principal de ralentir la convergence vers une distribution stationnaire. En effet, le fait d'échantillonner à nouveau une configuration lorsque l'échange est refusé correspond à une boucle dans le graphe de graphes et fait en sorte que les déplacements entre les configurations seront moins fréquents (au contraire d'avoir un déplacement garanti à chaque itération si on ne propose que les bons échanges). Le problème se surmonte en augmentant la taille de l'échantillon, mais cette solution peut entraîner des problèmes numériques, en terme de mémoire notamment. La recherche d'un algorithme qui ne propose que les échanges valides pour explorer le modèle des configurations étagées est donc toujours d'actualité, mais nécessiterait une meilleure compréhension de la décomposition en oignon et de ses règles locales.

Pour l'exploration du LCM donc, l'idée de l'algorithme est de proposer une paire de liens uniformément au hasard parmi tous les liens du graphe, et d'effectuer seulement les échanges qui ne créent pas de boucle, ne créent pas de multilien, et qui ne changent pas la décomposition en oignon. Pour cette dernière vérification, on pourrait naïvement calculer la décomposition en oignon de la configuration actuelle, effectuer un échange, puis calculer la décomposition en oignon de la nouvelle configuration. Cependant, le calcul de la décomposition en oignon

pour un graphe de n noeuds et m liens est de l'ordre $\mathcal{O}(m \log n)$ [43] et l'application de cette méthode est prohibitive d'un point de vue numérique.

Pour échantillonner l'ensemble de manière efficace, il faudra utiliser les règles locales 1 de la décomposition en oignon. Ces règles permettent de déterminer si un échange peut être accepté ou non en effectuant quelques vérifications simples, ce qui donne un algorithme beaucoup plus efficace, avec une vérification d'ordre $\mathcal{O}(1)$. Les règles locales permettent de déterminer la couche d'un noeud à partir de celle de ses voisins. La vérification se base sur ce principe, en considérant d'abord le voisinage du lien existant, puis en considérant le nouveau voisinage si on effectuait l'échange.

Algorithme 5 Test d'échange pour la décomposition en oignon

entrée : Graphe G_{i-1} de l'itération précédente, noeud a, voisin b actuel de a et voisin c potentiel de a

sortie : **vrai** si l'échange change la décomposition en oignon, **faux** sinon

```
c ← coeur du noeud a
ℓ_a ← couche du noeud a
ℓ_b ← couche du noeud b
ℓ_c ← couche du noeud c
n_r ← nombre de voisins du noeud a qui sont dans une couche ℓ ≥ ℓ_a
n_rb ← nombre de voisins du noeud a qui sont dans une couche ℓ ≥ ℓ_a − 1
si l_a est la première couche du coeur c alors
  si ℓ_b ≥ ℓ_a alors
    si ℓ_c ≥ ℓ_a alors
      retourne faux
    sinon
      retourne vrai
    fin si
  sinon si ℓ_b < ℓ_a alors
    si ℓ_c ≥ ℓ_a alors
      retourne faux
    sinon
      retourne vrai
    fin si
  fin si
sinon
  si ℓ_b ≥ ℓ_a alors
    si n_rb = c + 1 alors
      si ℓ_c ≥ ℓ_a − 1 alors
        retourne faux
```

```
        sinon
          retourne vrai
        fin si
      sinon
        retourne faux
      fin si
    sinon si ℓ_b = ℓ_a − 1 alors
      si n_rb = c + 1 alors
        si ℓ_c ≥ ℓ_a − 1 alors
          retourne faux
        sinon
          retourne vrai
        fin si
      sinon si n_r = c alors
        si ℓ_c < ℓ_a alors
          retourne faux
        sinon
          retourne vrai
        fin si
      sinon
        retourne faux
      fin si
    sinon
      si n_r = c alors
        si ℓ_c < ℓ_a alors
          retourne faux
        sinon
          retourne vrai
        fin si
      sinon
        retourne faux
      fin si
    fin si
  fin si
```

Pour un éventuel échange $(u, v); (x, y) \rightarrow (u, x); (v, y)$, on vérifie le voisinage actuel pour chacun des noeuds u, v, x, y, et l'on compare chacun au voisinage si on effectue l'échange. Il faut donc effectuer un *test d'échange* à 4 reprises, une fois pour chaque noeud impliqué dans l'échange. Le test d'échange est décrit à l'algorithme 5. Comme mentionné, il s'inspire directement des règles 1, s'assurant que chacune d'elles soit respectée si l'échange est effectué.

Le test est ici présenté en utilisant les valeurs des couches (ceci peut faciliter l'implémentation numérique), mais il peut parfois être instructif de le revoir en terme des couleurs de la section 1.2.2, notamment pour faciliter les approches théoriques. Une version de l'algorithme qui utilise les couleurs de demi-liens est présentée à l'annexe A.

L'algorithme complet pour échantillonner le LCM est présenté à l'algorithme 6. Encore une fois, il s'agit d'une modification simple de l'algorithme 2 pour échantillonner le CM.

Algorithme 6 MCMC pour le modèle des configurations étagées

entrée : Graphe initial G_0
sortie : Échantillons de graphes $\{G_i\}$
  pour $i <$ taille de l'échantillon **faire**
    choisir deux liens (u, v) et (x, y) uniformément au hasard
    choisir au hasard l'échange $(u, v); (x, y) \to (u, x); (v, y)$ ou $(u, v); (x, y) \to (u, y); (v, x)$
    si l'échange crée une boucle ou un multilien **alors**
      $G_i \leftarrow G_{i-1}$
    sinon si l'échange change la décomposition en oignon (alg. 5) **alors**
      $G_i \leftarrow G_{i-1}$
    sinon
      procéder à l'échange de liens pour obtenir G_i
    fin si
    ajouter G_i à l'échantillon
  fin pour

2.1.2 Uniformité de l'échantillonnage

Les avantages d'utiliser la stratégie de *switch and hold* deviennent plus évidents lorsque l'on évalue la distribution stationnaire vers laquelle tend l'algorithme 6. Cette stratégie peut en effet grandement simplifier les preuves pour l'uniformité de la distribution.

D'abord, comme pour les échanges qui créent des boucles ou des multiliens, lorsqu'un échange est refusé parce qu'il change la décomposition en oignon, ce refus correspond à une boucle dans le graphe de graphes. Il a été montré dans la preuve du théorème 19 qu'une boucle correspond autant aux degrés entrant et sortant qu'un échange qui serait accepté. On peut en conclure que le degré de chacun des noeuds du graphe de graphes reste inchangé, et donc que ce dernier est encore régulier.

Ensuite, comme cette stratégie crée des boucles dans le graphe de graphes, elle peut être utilisée pour élaborer une preuve alternative pour l'apériodicité de la chaîne. Soit le graphe de graphes $\mathcal{G}$ associé au modèle des configurations étagées $\Gamma_{LCM}(G)$ d'un graphe G. On suppose que $\mathcal{G}$ est fortement connecté. Alors

Théorème 20. *Le graphe de graphes $\mathcal{G}$ est apériodique.*

Démonstration. On note d'abord que, par définition, si une boucle existe dans $\mathcal{G}$, alors $\mathcal{G}$ est apériodique. La preuve montre qu'il existe toujours au moins une boucle dans $\mathcal{G}$. Celle-ci est séparée en deux parties : on considère d'abord le cas où il existe au moins un noeud de degré 2 ou plus dans le graphe.

Comme il existe au moins un noeud v de degré 2 ou plus, il existe forcément une paire de liens $(u, v); (v, x)$. Un échange sur cette paire de liens donne soit $(u, v); (v, x) \to (u, v); (v, x)$, qui correspond à une boucle puisque le graphe obtenu est le même, ou soit $(u, v); (v, x) \to (u, x); (v, v)$, qui est refusé par l'algorithme et correspond donc aussi à une boucle. Dans les deux cas, il existe une boucle dans $\mathcal{G}$, qui est donc apériodique.

Ensuite, s'il n'existe pas de noeud de degré 2 ou plus, alors les noeuds sont tous de degré 0 ou 1. Les seuls connections possibles dans le graphe sont des paires isolées, et tous les liens sont connecté à des noeuds de la couche $\ell = 1$. Lorsque les noeuds impliqués dans les échanges sont tous de la même couche, les échanges ne sont jamais refusés par les tests 5 (on retrouve alors une équivalence avec l'algorithme 2 pour le CM). On peut alors appliquer directement la preuve de 18. □

Cette démonstration repose sur le principe du *switch and hold* dans sa formulation. Dans le cas où une nouvelle méthode d'échantillonnage serait trouvée qui ne propose que les échanges valides, il faudrait la revoir. Des pistes de réflexions pour ce cas sont proposées dans l'annexe B.

On a donc montré, avec des preuves simples, que l'algorithme 6 donne un graphe de graphes régulier et apériodique. Ceci indique qu'il est capable d'explorer l'ensemble de manière non biaisée, à condition que l'ensemble soit entièrement connecté par les échanges de liens qui soient encore acceptés. Si l'ensemble n'est pas entièrement connecté cependant, on sait au moins que l'espace qui sera exploré par l'algorithme le sera sans biais.

La question de la connectivité du modèle des configurations étagées sous les bons échanges de liens est beaucoup plus complexe que les preuves pour les autres propriétés du graphe de graphes ; la prochaine sous-section lui est entièrement consacrée.

2.1.3 Irréductibilité de la chaîne

Pour simplifier le texte, lorsque l'on parlera d'échanges de liens pour le LCM, on parlera par défaut des échanges qui ne créent pas de boucles, pas de multiliens, et qui ne changent pas la décomposition en oignon. C'est-à-dire que l'on réfère uniquement aux échanges qui seraient acceptés par l'algorithme 6.

Beaucoup de travail a été fait durant le projet de maîtrise pour savoir si les échanges de liens étaient suffisants pour explorer l'entièreté du LCM pour n'importe quel graphe G ou si d'autres

types de transformations étaient aussi nécessaires ; cette section est un résumé des résultats et des pistes de solutions qui ont été explorées.

D'abord, la solution complète pour le problème n'est pas encore connue. De par la complexité du problème, une preuve complète qui prend en compte tous les cas possibles n'a pas pu être développée. Il est toutefois possible de présenter certaines pistes de solution, à commencer par présenter des cas où il est toujours possible de créer un chemin entre différentes configurations du graphe de graphes.

Un premier cas simple est le cas où tous les noeuds sont dans la même couche. Soit un graphe G quelconque avec un ensemble de noeuds V. Soit un sous-ensemble S de V où $\ell(v) = \ell$ pour tout $v \in S$. Les liens entre les noeuds de S seront tous de couleurs rouge-rouge, et procéder à un échange entre ces noeuds ne changera jamais la couleur d'aucun demi-lien. Donc, les règles 2 ne seront jamais enfreintes, et tous les échanges entre ces liens seront acceptés comme si on était dans le modèle des configurations. Comme on sait que le CM est fortement connecté, on peut en conclure que toutes les configurations entre les noeuds de S sont aussi connectées, ce qui permet de connecter un sous-ensemble $\mathcal{S}$ de $\Gamma_{LCM}(G)$.

On peut utiliser un raisonnement similaire pour échanger deux liens de couleurs rouge-noir qui sont entre les mêmes couches. En effet, si on effectue des échanges de liens entre les noeuds de couche ℓ et $\ell - 1$, les règles 2 sont toujours respectées. On peut se convaincre que c'est aussi le cas pour les liens rouge-vert entre les mêmes couches. De plus, on peut généraliser les échanges entre deux liens rouge-vert de même couche pour montrer qu'il est toujours possible d'échanger deux liens rouge-vert à condition que la différence entre la couche du demi-lien vert le plus bas et celle du demi-lien rouge le plus haut soit d'au moins 2.

On peut aussi considérer certains échanges entre des noeuds de différentes couleurs : dans le cas où trois noeuds concernant l'échange sont dans la même couche, il est toujours possible d'effectuer l'échange, peu importe dans quelle couche se trouve le quatrième noeud concerné. Ceci inclut donc une partie des échanges entre des liens rouge-rouge et rouge-noir ou des liens rouge-rouge et rouge-vert.

Les transformations qui viennent d'être présentées peuvent être utilisées pour produire des échanges entre un très grand nombre de configurations du LCM. Ils représentent des échanges où les couleurs de demi-liens ne sont pas modifiées par l'échange, et il pourrait donc être facile de les utiliser dans d'éventuelles preuves de connectivité. Ces cas n'englobent cependant pas toutes les possibilités d'échanges entre tous les liens d'un graphe du LCM ; il reste plusieurs cas spécifiques à explorer avant de conclure à la connectivité de l'espace, notamment des cas où l'échange change la couleur des demi-liens sans contrevenir aux règles 2.

Tous les cas spécifiques n'ont pas été explorés, et encore à ce jour, il est impossible de déterminer s'il est possible de les classifier pour tous les énumérer. Ainsi, obtenir une conclusion sur

l'irréductibilité de l'algorithme 6 d'un point de vue théorique demeure une question ouverte.

En parallèle avec les pistes théoriques, une méthode numérique a été développée pour tester la connectivité de petits ensembles. Il s'agit d'une expérience numérique que l'on peut décrire de la manière suivante : on commence par utiliser l'algorithme 6 pour trouver le plus de graphes possible à partir d'un graphe G. Ceci donne un ensemble de graphes $\mathcal{G}_{LCM}$ qui est forcément connecté. Par la suite, utiliser l'algorithme du CM 2, et trouver le plus de graphes possible encore une fois. L'ensemble de graphes obtenu $\mathcal{G}_{CM}$ est aussi connecté, et $\mathcal{G}_{LCM}$ devrait en être un sous-ensemble [2]. Il est maintenant possible de parcourir tous les graphes de $\mathcal{G}_{CM}$, et conserver seulement ceux qui ont la même décomposition en oignon que G. Ceci donnera un nouveau sous-ensemble $\mathcal{S}$ de $\mathcal{G}_{CM}$, que l'on pourra comparer à $\mathcal{G}_{LCM}$. Ce test correspond à prendre tous les graphes du CM [3] et garder seulement ceux qui ont la bonne décomposition en oignon, et regarder si tous ces graphes ont pu être obtenus avec l'algorithme pour le LCM. Si ce n'est pas le cas, c'est-à-dire si $|\mathcal{S}| > |\mathcal{G}_{LCM}|$, alors cela signifie qu'il existe des graphes qui ont la même décomposition en oignon que G, mais qui n'ont pas été atteints par l'algorithme. Une fois ces graphes identifiés, on peut les analyser manuellement pour voir s'ils peuvent être obtenus par échange de liens, ou s'ils sont bel et bien isolés dans le graphe de graphes.

Cette expérience a été menée sur des réseaux issus du modèle $G(n, m)$ pour des valeurs de n allant jusqu'à 9 noeuds (pour toutes les valeurs de m possibles). Les réseaux ont été générés avec un algorithme génératif pour ce modèle. Quelques contre-exemples montrant que le modèle des configurations étagées n'est pas connecté sous échange de liens ont été trouvés. L'une des ces situations est illustrée à la figure 2.1. On peut se convaincre, en analysant tous les échanges possibles, qu'il est impossible de créer un chemin entre le graphe isolé de droite et la composante fortement connectée de gauche, et donc que cet ensemble spécifique est déconnecté.

Cet exemple est le premier contre-exemple à la preuve de connectivité qui a été découvert avec l'expérience numérique. Il indique que le LCM n'est pas connecté avec les échanges de liens pour tous les graphes. Quelques autres exemples de ce genre ont été trouvés, même si en général cette méthode semble montrer que l'algorithme est irréductible pour la plupart des graphes. Une caractéristique remarquable qui est commune à tous ces exemples est qu'ils impliquent tous un lien isolé comme le lien $(1, 2)$ dans la figure.

Ce contre-exemple permet de jeter un nouveau regard et de nouveaux questionnements à la question de la connectivité, à savoir : quelle est la proportion de graphes pour lesquels la chaîne n'est pas irréductible ? Est-ce que le lien isolé est le seul exemple de structure qui empêche la

2. Il est possible, dû à la taille des ensembles, que des graphes soient trouvés dans $\mathcal{G}_{LCM}$ mais pas dans $\mathcal{G}_{CM}$. Pour corriger cette erreur, on peut tout simplement les ajouter manuellement à $\mathcal{G}_{CM}$, ce qui permettra de dire que $\mathcal{G}_{LCM}$ est un sous-ensemble de $\mathcal{G}_{CM}$.

3. Tous les graphes de $\Gamma_C M(G)$ ne sont généralement pas obtenus en faisant le test. En théorie cependant, avec un temps de calcul et une mémoire suffisamment grands, il serait possible de tous les énumérer de cette manière.

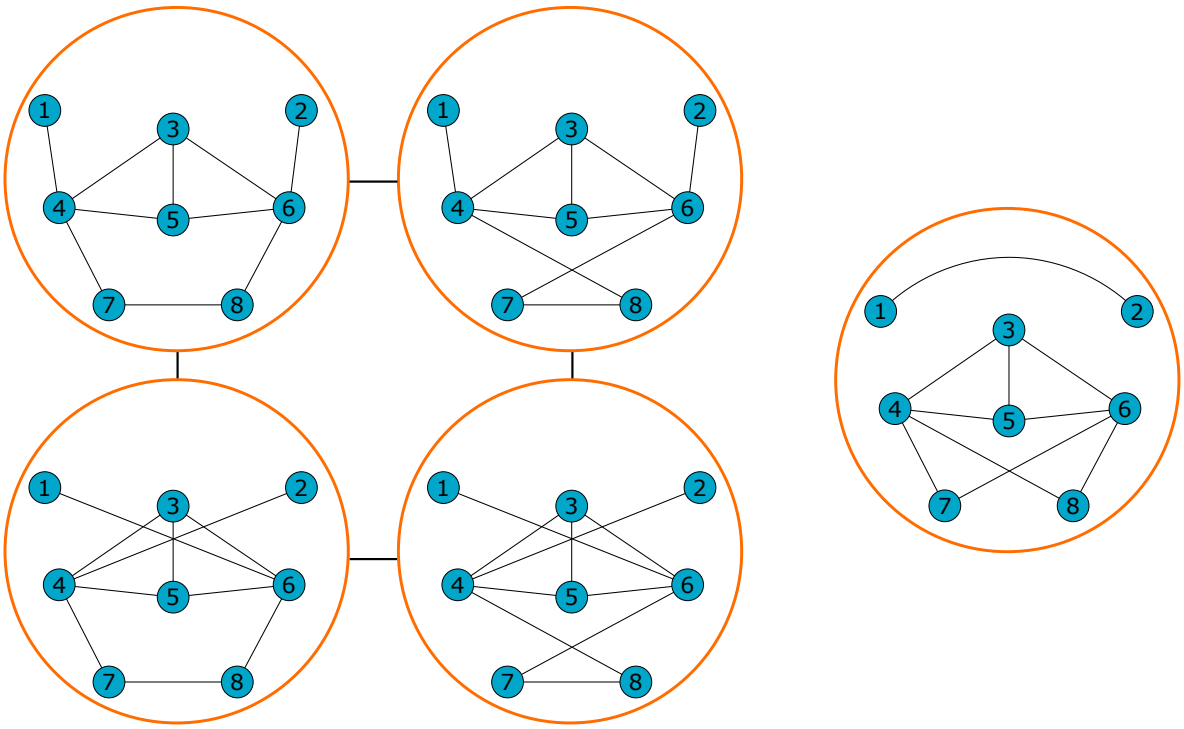

FIGURE 2.1 – Graphe de graphes pour le LCM d'un graphe de 9 noeuds et 10 liens. Les graphes affichés ont tous la même séquence de degrés et la même décomposition en oignon et font donc tous partie du même LCM. Le graphe de graphes n'est cependant pas connecté : ce dernier est formé d'une composante de 4 graphes fortement connectée, et d'une configuration isolée. On note que les boucles ne sont pas affichées dans cet image.

connectivité ? Peut-on trouver des structures dans le graphe de base qui indique immédiatement si l'ensemble est déconnecté ? Lorsqu'un ensemble est déconnecté, est-il toujours formé de deux composantes ou est-il possible d'avoir trois sous-ensembles ou plus qui sont déconnectés ? Ou encore, est-ce que la partie déconnectée du graphe peut-être négligée de manière à considérer que l'algorithme est toujours soit irréductible ou presque irréductible ?

Ces questions sont pour l'instant toutes sans réponse. On peut cependant avancer quelques hypothèses en se basant sur le travail théorique mentionné précédemment. D'abord, plus les graphes sont grands, plus on peut s'attendre à avoir des échanges « simples » qui ne changent pas la couleur des liens. Plus ce nombre est grand, plus on peut explorer des configurations, et dans ce cas on pourrait s'attendre à avoir des ensembles connectés pour de grands graphes, ou du moins, avoir une grande composante qui elle est fortement connectée, et de petits cas isolés de composantes qui sont déconnectées. Il serait très intéressant de pouvoir quantifier ces hypothèses, ce qui permettrait de mieux justifier l'utilisation des échanges de liens pour l'exploration du LCM.

Autrement, si l'algorithme proposé dans le mémoire n'est pas irréductible pour une trop grande quantité de graphes, il existe de nombreuses modifications qui permettraient de corriger cette

situation. On note d'abord que la stratégie utilisée pour l'expérience, qui consiste à échantillonner le CM pour ensuite sélectionner uniquement les graphes qui respectent les contraintes imposées, n'est pas une stratégie viable. En effet, la taille du CM, qui est seulement bornée par le nombre de liens dans le graphe de base, augmente beaucoup plus rapidement que la taille des autres ensembles [47]. De cette manière, la probabilité d'obtenir un graphe faisant partie d'un sous-ensemble du CM diminue lorsque la taille du graphe de base augmente, devenant négligeable pour de très grands graphes. Donc, selon cette stratégie, la plupart des itérations de l'algorithme donneraient des graphes devant être rejetés, ce qui demanderait un temps de calcul beaucoup trop grand pour obtenir un échantillon représentatif de l'ensemble que l'on voulait explorer. Il faut alors opter pour d'autres solutions qui permettront de rester plus longtemps dans le bon ensemble.

La première stratégie qui pourrait être employée serait de coupler l'utilisation de l'algorithme 6 avec un algorithme de génération de graphes. Avec un algorithme de génération de graphes, il serait possible de générer des graphes dans les différentes composantes qui ne sont pas connectées par échange de liens, ce qui permettrait ensuite de faire des échanges de liens dans chacune des composantes. Plus précisément, on pourrait commencer par utiliser l'algorithme 6 pour explorer une partie de l'ensemble, puis après un certain nombre d'itérations, on pourrait générer un nouveau graphe et repartir la chaîne à partir de celui-ci. Dans ce cas, la génération de nouveaux graphes reviendrait à donner une probabilité non nulle de faire un saut d'une composante à l'autre, ce qui serait suffisant pour rendre la chaîne irréductible.

Même s'il n'existe pas pour l'instant d'algorithme génératif capable de générer n'importe quel graphe d'un LCM de manière uniforme, un algorithme qui pourrait être suffisant a été développé en parallèle avec ce projet. Ce dernier est capable de générer des graphes avec trois séquence de degrés distinctes : une pour le nombre de demi-liens rouges, une pour le nombre de demi-liens noirs et une pour le nombre de demi-liens verts. L'algorithme n'a pas été assez exploré pour permettre d'affirmer qu'il génère tous les graphes d'un LCM (de manière uniforme ou non), mais il pourrait permettre de combler d'éventuels cas d'ensemble déconnectés sous échange de liens.

Autrement, une deuxième stratégie serait d'alterner entre les échanges de l'algorithme 6 pour le LCM et les échanges de l'algorithme 2 pour le CM. Comme on sait que le CM est connecté, on sait que si l'on retire le test d'échanges de liens du LCM, il serait possible d'atteindre tous les graphes de cet ensemble. En permettant pendant un certain temps à l'algorithme d'effectuer des échanges « illégaux », on pourrait suivre un chemin jusqu'à une nouvelle composante, qui était initialement déconnectée de la composante de base.

Finalement, une troisième option serait de modifier les transformations utilisées, par exemple en ajoutant la possibilité à l'algorithme de faire des échanges entre trois liens. Cette stratégie a été adoptée notamment pour régler le problème de connectivité du CM avec boucles [71].

Dans les trois stratégies, on amène une perturbation à un algorithme qui explore déjà de façon uniforme. Il faudra donc être très prudent lors de l'implémentation et bien s'assurer que les perturbations ne changent pas l'uniformité de la distribution stationnaire.

2.2 Modèles des configurations étagées et corrélées

Une fois le modèle des configurations étagées développé, il est possible d'aller au delà de celui-ci et d'ajouter des contraintes supplémentaires à ce nouvel ensemble. Les méthodes d'échantillonnage des modèles de configurations corrélées présentés à la section 1.8 sont assez simples pour qu'elles soient directement implémentées dans l'algorithme 6. Il est donc possible d'ajouter des contraintes non seulement sur les corrélations de degrés, mais aussi sur les corrélations de couche et sur les corrélations des tuples (degré, couche).

On peut définir les nouveaux ensembles de la manière suivante : soit un graphe G avec un ensemble de noeuds $V = \{v_i\}_{i=1}^n$, une séquence de degrés $\mathbf{d}$, une séquence de coeurs $\mathbf{c}$, une séquence de couches $\mathbf{OD}$ et une matrice de corrélation de type e. Les corrélations de types viennent en trois « saveurs » : degré-degré, couche-couche, ou (degré, couche)-(degré, couche). On définit les *modèles des configurations étagées et corrélées* de G (LCCMs [4]), notés respectivement $\Gamma^d_{LCCM}(G)$, $\Gamma^\ell_{LCCM}(G)$ et $\Gamma^{(d,\ell)}_{LCCM}(G)$, comme les ensembles générés par une contrainte rigide sur la séquence de degrés $\mathbf{d}$, la séquence de coeurs $\mathbf{c}$, la séquence de couches $\mathbf{OD}$ et la matrice de corrélation e.

2.2.1 Algorithmes d'échantillonnage : contraintes rigides

Comme il a été mentionné précédemment, la méthode de la section 1.8 pour les contraintes rigides peut être directement appliquée sur l'algorithme 6 pour échantillonner l'ensemble. On obtient alors l'algorithme 7. On note que t dans la description de l'algorithme peut prendre trois valeurs, soient d pour les degrés, ℓ pour les couches ou (d, ℓ) pour les tuples (degré, couche). Dans le LCM, lorsque l'on considérait les transformations possibles, on considérait toujours l'entièreté des échanges qui existent, c'est-à-dire les $\binom{E}{2}$ paires de liens possibles. Dans cet algorithme cependant, les échanges qui sont proposés sont beaucoup moins nombreux puisqu'on élimine d'emblée tous ceux qui ne sont pas entre des noeuds de même groupe E_t. Ceci a pour principal avantage d'accélérer la convergence de l'algorithme.

Algorithme 7 MCMC pour le LCCM rigide

entrée : Graphe initial G_0

sortie : Échantillons de graphes $\{G_i\}$

créer les ensembles E_t et leur attribuer une probabilité $P(E_t)$ proportionnelle au nombre de liens dans chaque ensemble

pour $i <$ taille de l'échantillon **faire**

4. de l'anglais *Layered and Correlated Configuration Models.*

choisir un ensemble E_t avec probabilité $P(E_t)$
choisir deux liens (u, v) et (x, y) uniformément au hasard parmi les liens de E_t
si $t_u = t_v$ ou $t_x = t_y$ **alors**
choisir un des deux échanges possibles au hasard
sinon
choisir le seul échange approprié
fin si
si l'échange crée une boucle ou un multilien **alors**
$G_i \leftarrow G_{i-1}$
sinon si l'échange change la décomposition en oignon (alg. 5) **alors**
$G_i \leftarrow G_{i-1}$
sinon
procéder à l'échange de liens pour obtenir G_i
fin si
ajouter G_i à l'échantillon
fin pour

Pour cet algorithme, on peut appliquer la même démonstration que pour le théorème 20 pour l'apériodicité. On peut aussi se convaincre que les transitions de la chaîne de Markov sont symétriques : pour ce faire, on veut montrer que les probabilités $P(G \to G')$ de faire une transition du graphe G au graphe G' et $P(G' \to G)$ de faire la transition inverse sont identiques pour tous les graphes, c'est-à-dire $P(G \to G') = P(G' \to G)$ pour tout G, G'. D'abord, si le graphe G' contient des boucles ou des multiliens ou s'il n'a pas la bonne décomposition en oignon, on a que $P(G \to G') = 0$ puisque l'échange est refusé par l'algorithme. On pose dans ce cas que $P(G' \to G) = 0$ également, ce qui permet d'avoir des probabilités de transitions symétriques. Autrement, les probabilités de transition sont données par la probabilité de choisir une paire de liens multipliée par la probabilité de choisir un des deux échanges possibles.

La probabilité de choisir un groupe particulier est donnée par la probabilité $P(E_t) := |E_t|/|E|$ et la probabilité de choisir un lien parmi cet ensemble est $|E_t|^{-1}$. Donc, la probabilité de choisir le premier lien est $|E_t|/|E| \times |E_t|^{-1} = |E|^{-1}$, et la probabilité de choisir une paire de liens $(u, v); (x, y)$ est alors $[|E|(|E_t| - 1)]^{-1}$. Ensuite, si $t_u = t_v$ ou si $t_x = t_y$, l'algorithme choisit automatiquement le bon échange, donc la probabilité de choisir cet échange est 1. Pour les autres cas, on a que la probabilité de choisir chacun des deux échanges est de 1/2. Donc, les probabilités de transitions $P(G \to G')$ ne dépendent que des normes des groupes E_t. Comme ces groupes sont invariants d'un graphe à l'autre, les probabilités seront toujours égales, ce qui montre que les probabilités sont toutes symétriques.

Au final, l'apériodicité de la chaîne et la symétrie des transitions fait en sorte que l'algorithme convergera toujours vers la distribution uniforme que l'on cherche. Il resterait à vérifier l'unicité

de cette distribution stationnaire en prouvant l'irréductibilité de la chaîne. On sait déjà que la méthode employée ici est irréductible dans le cas où il n'y a pas de contraintes sur la décomposition en oignon [26], mais ce résultat n'est pas directement applicable une fois que l'on ajoute de nouvelles contraintes. Le problème à résoudre est probablement plus complexe que celui pour le LCM, et peu de travail à été fait jusqu'à présent pour élucider la question.

La seule intuition qui peut être donnée sur le problème est celle obtenue lorsque l'on effectue la même expérience numérique qui a été effectuée pour le LCM. L'expérience sur les trois LCCMs a été effectuée en parallèle avec celle sur le LCM. Le résultat de ces expériences est assez intéressant : dans les trois ensembles, aucun cas d'espace déconnecté n'a été trouvé. C'est-à-dire que même dans les ensembles où le LCM était déconnecté, comme par exemple dans celui présenté à la figure 2.1, les contraintes supplémentaires éliminaient toujours le ou les graphes déconnectés et faisaient en sorte que les graphes trouvés par l'algorithme étaient les mêmes que ceux trouvées en « taillant » le CM. Évidemment, ce résultat ne concerne qu'une petite quantité de petits graphes, et une généralisation théorique de ce résultat reste une avenue à explorer.

2.2.2 Algorithmes d'échantillonnage : contraintes souples

Pour explorer les ensembles avec des contraintes souples, il est possible d'appliquer le Metropolis-Hastings proposé par Newman et qui a aussi été présenté à la section 1.8. Les probabilités d'acceptation du Metropolis-Hastings sont donc

$$\mathcal{A}(A'|A) = \min\left\{1, \frac{e(t_u, t_x)e(t_v, t_y)}{e(t_u, t_v)e(t_x, t_y)}\right\}, \tag{2.1}$$

où t peut encore une fois prendre les valeurs d, ℓ ou (d, ℓ). Cette méthode est cependant une application naïve d'une méthode qui n'est pas complètement appropriée pour le problème, et il faut donc faire attention pour s'assurer que les résultats obtenus sont ceux escomptés.

Les problèmes viennent encore une fois du fait que la distribution de probabilité est une approximation. D'abord, il ne faut pas oublier que la distribution de probabilité proposée représente un ensemble dans lequel on conserverait la distribution de degrés et non la séquence de degrés. De plus, comme on conserve la décomposition en oignon dans les ensembles, il faut prendre en compte cette nouvelle contrainte dans le calcul de la distribution, ce qui n'est pas fait lorsque l'on applique directement l'équation (2.1). Les résultats obtenus peuvent donc être plus ou moins biaisés selon le type de corrélation que l'on veut contraindre, et il faut faire des tests numériques pour voir si cette méthode est bel et bien appropriée.

La méthode pour échantillonner le LCCM à contrainte souple est décrite par l'algorithme 8. La convergence de cet algorithme est majoritairement déterminée par les probabilités d'acceptation, il n'est donc pas nécessaire de faire une étude détaillée des propriétés du graphe de graphes comme pour les ensembles à contraintes rigides. Cependant, on note qu'il est possible

que le graphe de graphes ne soit pas toujours connecté, surtout pour de petits graphes. En principe, il s'agit d'un problème de taille finie qui est corrigé lorsque l'on augmente la taille des réseaux. Le problème est présenté à l'annexe C, mais on peut en général supposer que le graphe de graphes est connecté sans causer de biais important.

Algorithme 8 MCMC pour le LCCM souple

entrée : Graphe initial G_0
sortie : Échantillons de graphes $\{G_i\}$
  calculer la matrice conjointe e
  pour $i <$ taille de l'échantillon **faire**
    choisir deux liens (u, v) et (x, y) uniformément au hasard
    choisir au hasard l'échange $(u, v); (x, y) \to (u, x); (v, y)$ ou $(u, v); (x, y) \to (u, y); (v, x)$
    si l'échange crée une boucle ou un multilien **alors**
      $G_i \leftarrow G_{i-1}$
    sinon si si l'échange change la décomposition en oignon (alg. 5) **alors**
      $G_i \leftarrow G_{i-1}$
    sinon
      tirer un nombre $0 \leq r < 1$ uniformément au hasard
      si $r \leq \min\left\{1, \frac{e(t_u,t_x)e(t_v,t_y)}{e(t_u,t_v)e(t_x,t_y)}\right\}$ [5] **alors**
        procéder à l'échange de liens pour obtenir G_i
      sinon
        $G_i \leftarrow G_{i-1}$
      fin si
    fin si
    ajouter G_i à l'échantillon
  fin pour

2.3 Librairie C++ et résultats numériques sur la convergence

Une partie importante du projet a été de développer une librairie utilisant l'ensemble des algorithmes présentés ; la librairie du nom de Onion Graph REshuffler (OGRE) [6] est une librairie codée majoritairement dans le langage C++ qui permet d'échantillonner les 10 ensembles avec des échanges de liens présentés jusqu'à présent. Le but de cette librairie est de fournir à la communauté des outils pour la génération de modèles nuls. Pour rendre le code plus versatile et pour augmenter sa compatibilité avec d'autres librairies de réseaux, les fonctionnalités de la librairie ont aussi été liées dans le langage Python.

5. ou si $r \leq \min\left\{1, \frac{e(t_u,t_y)e(t_v,t_x)}{e(t_u,t_v)e(t_x,t_y)}\right\}$, selon l'échange choisi.

6. Le code n'est pas encore disponible publiquement au moment du dépôt de ce mémoire, un lien ne peut donc être partagé ici.

Dans cette section, la librairie servira à vérifier l'efficacité des méthodes proposées, notamment en observant la convergence des chaînes vers les distributions recherchées. On commence ici par présenter à la figure 2.2 des premiers résultats de convergence pour les algorithmes à contraintes rigides. Cette figure affiche l'erreur quadratique moyenne entre la distribution uniforme et la distribution obtenus avec les algorithmes en fonction du nombre d'itérations. L'erreur quadratique moyenne est calculée avec la formule suivante :

$$E_{\text{rigide}} = \frac{1}{T} \sum_{G} [P(G) - \hat{P}(G)]^2 \tag{2.2}$$

où T est la taille de l'échantillon, $P(G) = \frac{1}{N}$ la probabilité d'obtenir le graphe G selon la distribution uniforme (N étant le nombre de graphes différents dans l'échantillon), et $\hat{P}(G) = \frac{\text{Nombre d'apparition de } G \text{ dans l'échantillon}}{\text{T}}$ la probabilité d'obtenir le graphe G selon l'échantillonnage.

On note que les différents ensembles LCCMs seront différenciés avec la notation LCCM DxD pour les corrélations en degrés, c'est-à-dire pour $t = d$, LCCM CxC pour $t = \ell$ et LCCM CDxCD pour $t = (d, \ell)$.

Dans la figure 2.2, on voit que plus les algorithmes sont appliqués longtemps, plus l'erreur entre la distribution uniforme et la distribution de l'échantillon diminue. Cette figure montre donc bien que tous les algorithmes explorent l'espace des graphes sans biais, comme les différentes preuves présentées jusqu'à présent l'avait prédit.

On peut aussi noter le fait que la courbe pour le CM semble avoir beaucoup moins de fluctuations que les autres courbes. Ceci est causé par la différence de taille des ensembles, puisque pour le graphe étudié, le CM était un ensemble de 1093 graphes, et le LCM un ensemble de 67 graphes (et donc les autres ensembles sont de tailles plus petites ou égales à 67). L'aspect aléatoire des algorithmes est plus significatif pour les ensembles plus petits, et c'est ce qui est reflété dans ces fluctuations. À noter que pour cette même raison de différence de taille, on ne peut comparer la vitesse de convergence des différents algorithmes.

On peut faire un exercice similaire pour les ensembles à contraintes souples : la figure 2.3 montre la convergence de la moyenne des matrices de corrélations vers la matrice de contrainte pour les différentes ensembles. L'erreur quadratique moyenne est dans ce cas calculée avec

$$E_{\text{souple}} = \frac{1}{M} \sum_{i,j} [e_{ij} - \hat{e}_{ij}]^2 \tag{2.3}$$

où M est le nombre d'éléments de la matrice conjointe, e est la matrice conjointe, et $\hat{e} = \frac{1}{N} \sum_k e_k$ la matrice moyenne obtenue en sommant sur les matrices e_k de l'échantillon.

On observe d'abord une convergence rapide pour tous les types de corrélations, qui arrive à convergence en beaucoup moins d'itérations que pour les ensembles à contraintes rigides. Cependant, les convergences observées ici ne sont pas les mêmes que dans la figure 2.2, puisqu'ici

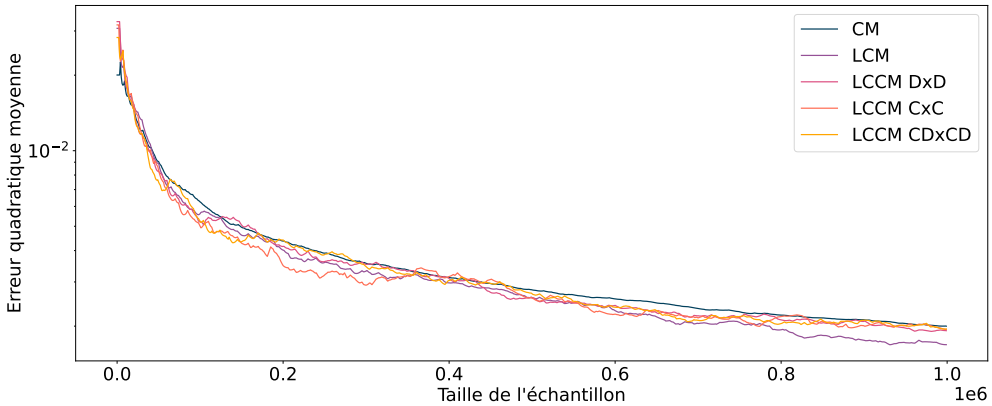

FIGURE 2.2 – Erreur quadratique moyenne entre la distribution uniforme et la distribution de l'algorithme en fonction de la taille de l'échantillon pour les différents ensembles à contraintes rigides. DxD indique les corrélations en degré, CxC les corrélations en couche, et CDxCD les deux corrélations en même temps. Les algorithmes sont appliqués sur un graphe de 8 noeuds.

on observe l'erreur entre des matrices plutôt que la distribution de probabilité sur l'entièreté de l'ensemble. Comparer la distribution obtenue à la distribution désirée ne serait pas réaliste dans le cas des contraintes souples, puisque la distribution recherchée est a priori inconnue. On ne peut donc toujours pas conclure sur la vitesse de convergence des différents algorithmes à partir de ces données ; la vitesse de convergence des chaînes reste encore une question ouverte, qui n'est toujours pas entièrement résolue même pour le CM [24, 25, 29, 31, 39, 40, 41, 49] ou le CCM [3].

La figure 2.3 sert surtout à comparer les différentes matrices de corrélations moyennes obtenues, et ainsi conclure à l'efficacité des probabilités d'acceptations. Plus l'erreur entre la matrice initiale et la matrice moyenne est petite, plus on sait que l'équation (1.9) est respectée par l'algorithme et donc plus on sait que la distribution de l'échantillon obtenu par ce dernier converge vers celle des ensembles à contraintes souples. Idéalement, on voudrait une erreur qui tend vers 0, mais des effets de taille finie empêchent une convergence exacte de l'algorithme. Parmi ces effets, on rappelle d'abord le fait que les probabilités (2.1) ne sont qu'une approximation et qu'elles sont exactes seulement dans la limite thermodynamique $n \to \infty$ (voir section 1.8). De plus, en général, obtenir une moyenne de matrice de corrélations exacte est impossible, puisque les matrices dépendent de configurations de graphes, qui sont des objets finis, ce qui fait qu'il n'existe pas nécessairement assez de configurations pour balancer exactement la moyenne vers la contrainte originale. Ceci est encore plus vrai pour les plus petits graphes puisqu'il y a encore moins de configurations possibles [7]. Finalement, comme mentionné lors de la description de l'algorithme à la section 2.2.2, l'ensemble n'est pas nécessairement connecté

7. Pour un nombre de noeuds n donné, il existe $2^{\binom{n}{2}}$ différents graphes, ce qui donne seulement un nombre fini de matrices de corrélations possibles.

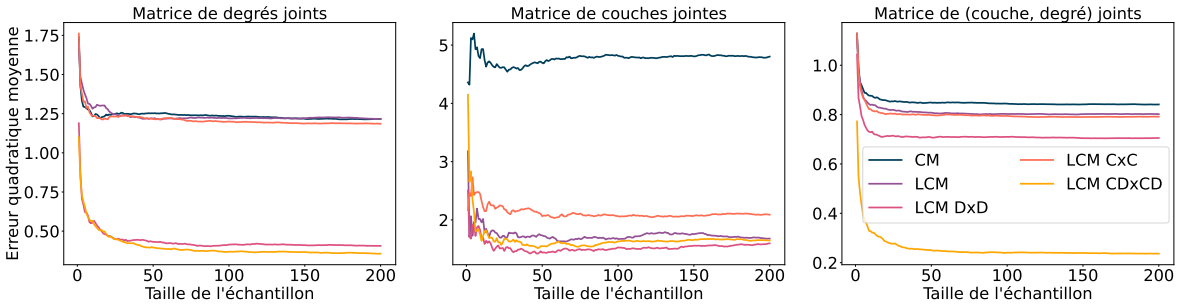

FIGURE 2.3 – Erreur quadratique moyenne entre la matrice de corrélation du graphe initial (la contrainte) et la moyenne des matrices de corrélations sur les différents ensembles, en fonction de la taille de l'échantillon. Les ensembles LCCM sont ici les ensembles à contraintes souples. Le graphe utilisé est un $G(n, m)$ avec $n = 100$ et $p = 0.02$, mais les résultats sont similaires pour tous les autres graphes testés.

et ceci pourrait exclure certaines matrices qui aideraient à réduire l'erreur restante.

Si on compare les erreurs obtenues pour la matrice de degrés joints (à gauche dans la figure), on remarque que les algorithmes qui amènent une corrélation sur les degrés, soient le LCCM DxD et le LCCM CDxCD donnent des erreurs significativement plus basses, ce qui permet de conclure que les bprobabilités d'acceptation amènent la bonne distribution po r ces ensembles. On observe aussi la même chose pour la matrice de (couche, degré) joints (à droite), où l'erreur pour le LCCM CDxCD est beaucoup plus petite que celle pour les autres ensembles. Cependant, ce gain significatif n'est pas observable pour la matrice de couches jointes (au centre) : la ligne d'erreur pour le LCCM CxC, l'ensemble qui devrait conserver la corrélation en couche en moyenne, est plus élevée que pour tous les autres ensembles qui conservent la décomposition en oignon. Ceci montre bien que les probabilités d'acceptation utilisées ne sont pas les bonnes et que l'ensemble est mal biaisé, puisque l'on s'attendrait plutôt à avoir une erreur plus petite si l'on utilisait les bonnes probabilités d'acceptation.

Pour mieux comprendre le problème de biais du LCCM CxC, on peut directement observer les matrices de couches jointes, qui sont affichées à la figure 2.4. On remarque, en observant la différence entre les deux matrices, que les éléments directement à coté de la diagonale sont toujours plus grands dans la matrice moyenne que dans la matrice du graphe initial. L'ensemble surévalue donc ces entrées, qui correspondent au nombre de liens entre des noeuds de couche ℓ et des noeuds de couche $\ell - 1$. Autrement dit, les probabilités d'acceptation font en sorte qu'il y a beaucoup plus de liens rouge-noirs qu'il ne devrait y avoir. Lorsque l'on fait l'expérience de calculer la probabilité d'acceptation moyenne de briser un lien noir-rouge contre celle de les créer, on observe en effet que ce type de lien est privilégié par les probabilités d'acceptation. Sans pouvoir expliquer pourquoi les probabilités (2.1) pour les corrélations en couche privilégient ce type de liens, on peut tout de même comprendre que le déséquilibre ainsi engendré explique l'erreur trop élevée de l'algorithme observée à la figure 2.3.

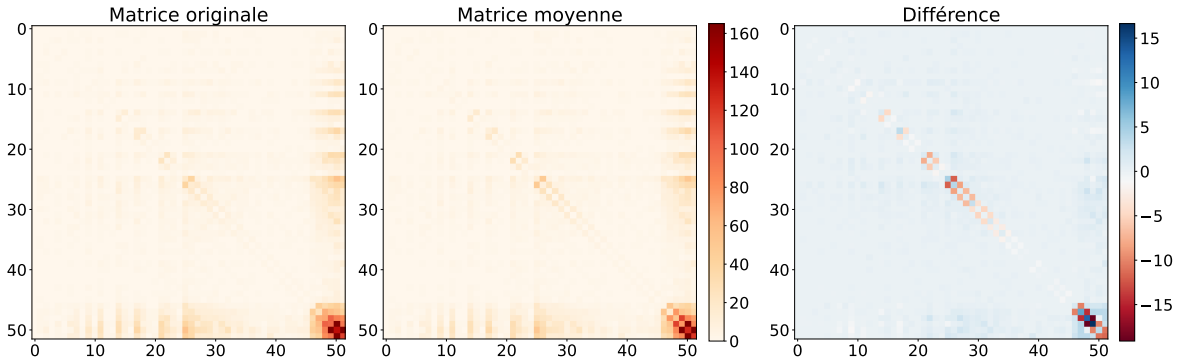

FIGURE 2.4 – (À gauche) Matrice de couches jointes pour le connectome du *C. elegans* [21] qui est utilisée comme contrainte sur l'ensemble LCCM CxC. (Au centre) Moyenne de 200 matrices de corrélations de graphes issus de l'algorithme pour le LCCM CxC. (À droite) Différence entre les deux matrices (une valeur positive indique que la valeur de la matrice originale est plus grande que celle de la matrice moyenne). On note que les éléments contiennent le nombre de liens entre les types à la place des fractions entre les liens.

Finalement, si on retourne à la figure 2.3, on peut observer les différents effets que conserver une corrélation spécifique peut engendrer sur les autres types de corrélations. D'abord, à la figure de gauche, on observe que le fait de conserver la décomposition en oignon ne réduit pas l'erreur quadratique moyenne. En effet, l'erreur pour le CM et celle pour le LCM convergent vers la même valeur, ce qui indique que la contrainte ajoutée (la décomposition en oignon) ne contribue pas à obtenir une matrice de degrés joints plus près de celle de base. Le fait qu'il ne semble pas y avoir de lien important entre la contrainte sur la décomposition en oignon et la corrélations en degrés est un résultat assez surprenant, puisque la décomposition en oignon est une mesure qui est indirectement basée sur le degré des noeuds. Une expérience numérique à la prochaine section viendra ajouter une profondeur à ce résultat préliminaire.

Pour ce qui est des comparaisons à gauche de la figure 2.3, on ne peut plus rien conclure sur le lien entre la corrélation en degrés et la corrélation en couches, puisque l'on sait maintenant que l'ensemble converge vers la mauvaise distribution. En observant maintenant le centre de la figure, on remarque que l'ajout de la contrainte sur la décomposition en oignon contribue à conserver les corrélation en couches, puisque les erreurs sont toujours plus petites sur les ensembles issus du LCM que sur le CM. Ce résultat n'est pas particulièrement surprenant [8], mais il est tout de même intéressant de le voir confirmé numériquement ici. Finalement, pour la figure 2.3 de droite, on observe que l'ajout des contraintes sur la corrélation en degrés diminue l'erreur du LCCM DxD par rapport aux CM et LCM, mais que la contrainte n'est pas suffisante pour diminuer l'erreur de manière aussi importante que pour le LCCM CDxCD.

8. Il va de soi que le fait de conserver la séquence de couche joue un rôle important sur les corrélations en couches, puisque ceci limite déjà grandement le nombre de matrices de couches jointes possibles parmi toutes les configurations de graphes qui existent. On pourrait en fait s'imaginer un cas similaire pour un ensemble sans contrainte sur la séquence de degrés, qui donnerait certainement une erreur sur la matrice de degrés joints beaucoup plus grande que celle obtenue par le CM.

Encore une fois, on ne peut rien conclure sur l'ajout des corrélations en couches.

Pour résumer les résultats importants de cette section, on note d'abord la convergence vers la distribution uniforme des ensembles à contraintes dures de la figure 2.2, ce qui confirme que les algorithmes utilisés sont tous valides. Ensuite, pour les contraintes souples, on a que l'algorithme de Metropolis-Hastings avec les probabilités (2.1) fonctionne bien pour les ensembles qui ont des corrélations de degrés, soient le LCCM DxD et le LCCM CDxCD. Pour les corrélations en couches seules cependant, l'algorithme donne un mauvais biais et il faudra donc revoir les probabilités d'acceptations pour obtenir des résultats plus représentatifs de l'ensemble à explorer. On note que bien que les résultats dans cette section sont présentés pour un nombre limité de graphes, de nombreux autres graphes ont été testés et les résultats amènent aux mêmes conclusions.

Une prochaine question, qui a été mentionnée à quelques reprises dans cette section, est celle du temps de convergence des algorithmes. Il serait très intéressant et utile de faire une comparaison de la convergence pour les techniques d'échantillonage. Celle-ci pourrait très certainement se baser sur un test statistique évaluant la convergence de l'assortativité des réseaux [9], en adaptant une méthode récemment développée [29]. L'analyse n'a pas encore été effectuée pour ce projet, et fait partie d'éventuels travaux futurs.

Faute de meilleure solution, la méthode privilégiée pour l'application des algorithmes sur des données réelles sera de faire une énorme quantité d'itérations, le nombre suffisant devant être calibré par essai et erreur. Ceci permettra de s'assurer d'avoir une convergence, même si le nombre d'itérations risque d'être beaucoup plus grand que nécessaire. L'échantillonnage qui a été appliqué par défaut pour toutes les expériences de la prochaine section est un mélange de 10 fois le nombre de liens avant d'ajouter un graphe à l'échantillon. On note que le fait de faire plusieurs échanges entre l'ajout d'un seul graphe ne change rien à la convergence de l'algorithme vers la bonne distribution, à condition que le nombre d'échange soit assez grand pour considérer que chaque ajout de graphe soit une mesure indépendante. Le nombre 10 est arbitraire, mais le fait de réduire ou augmenter ce nombre n'a rien changé aux résultats obtenus.

2.4 Application sur des réseaux réels

Jusqu'à présent, un total de 10 ensembles et algorithmes d'échantillonnage ont été présentés dans ce document : le CM, les deux CCMs, le LCM et finalement les 6 LCCMs. Le LCCM à contrainte souple sur les corrélations en couches diminue le nombre d'algorithmes d'échantillonnages qui fonctionnent à 9. Une fois les algorithmes développés, on veut les appliquer

9. En général, la mesure d'assortativité (ou n'importe quelle autre mesure scalaire) du réseau original est assez loin de la moyenne sur l'ensemble. Lorsque l'on calcule cette mesure scalaire pour chaque itération du réseau, on finit par voir une convergence vers la valeur moyenne de l'ensemble, ce qui permet de supposer que la chaîne a atteint l'équilibre.

sur des réseaux réels pour observer l'effet que les différentes contraintes peuvent avoir sur les propriétés des ensembles. Ceci permettra entre autre de déterminer dans quels contextes les contraintes amènent de nouvelles informations sur les systèmes étudiés.

Cette section présente différents résultats obtenus lorsque l'on applique les algorithmes d'échantillonage sur des réseaux réels. Les résultats sont séparés en deux parties : la première partie consiste à prendre différentes mesures sur les graphes d'un ensemble et à comparer les distributions ou moyenne de ces mesures obtenues sur les différents ensembles. La deuxième partie se concentrera sur l'application de dynamiques sur les ensembles, et permettra de voir comment les différentes contraintes peuvent influencer l'évolution des dynamiques.

2.4.1 Mesures sur les ensembles

La première mesure à être évaluée est l'*assortativité* des graphes. Cette mesure scalaire est définie comme étant le coefficient de corrélation de Pearson entre les degrés des noeuds connectés [64].

Elle indique la tendance des noeuds d'un certain degré à se connecter à des noeud du même degré. L'assortativité se calcule à partir de la matrice conjointe des degrés e avec la formule suivante [65] :

$$r = \frac{\sum_i e(i,i) - \sum_i a(i)b(i)}{1 - \sum_i a(i)b(i)} \tag{2.4}$$

avec les quantités définies à l'équation (1.2) et i des indices sur les éléments de la diagonale. Sa valeur varie entre 1 et -1, une valeur de 1 indiquant une corrélation maximale entre les noeuds de même degré et une valeur de -1 indiquant la corrélation inverse où les noeuds ont plutôt tendance à se connecter à des noeuds de degré différent (les noeuds de hauts degrés seront alors connectés à des noeuds de faibles degrés). Une valeur de 0 indique qu'il n'y a pas de tendance particulière.

Par définition, les graphes des ensembles à contraintes dures sur les corrélations de degré auront toujours exactement la même valeur d'assortativité. L'analyse ne sera donc pas faite sur ces ensembles, mais cette mesure permet tout de même de comparer les 6 autres ensembles, soient le CM, le CCM souple, le LCM, le LCCM CxC et les LCCM souples DxD et CDxCD. La figure 2.5 montre la distribution d'assortativité pour les ensembles issus de deux réseaux sociaux.

On note d'abord que dans les deux cas, la distribution du modèle des configurations s'éloigne beaucoup de la valeur d'assortativité de base. On observe donc que le mélange effectué par les échanges de liens enlève la corrélation entre les noeuds de même degré. Il s'agit d'un résultat attendu, puisque la contrainte sur la séquence de degrés a peu d'influence sur la corrélation des degrés. En fait, ce résultat se généralise à tous les graphes, et on note que le modèle des configurations donne généralement des graphes avec une assortativité nulle. Ceci est vrai

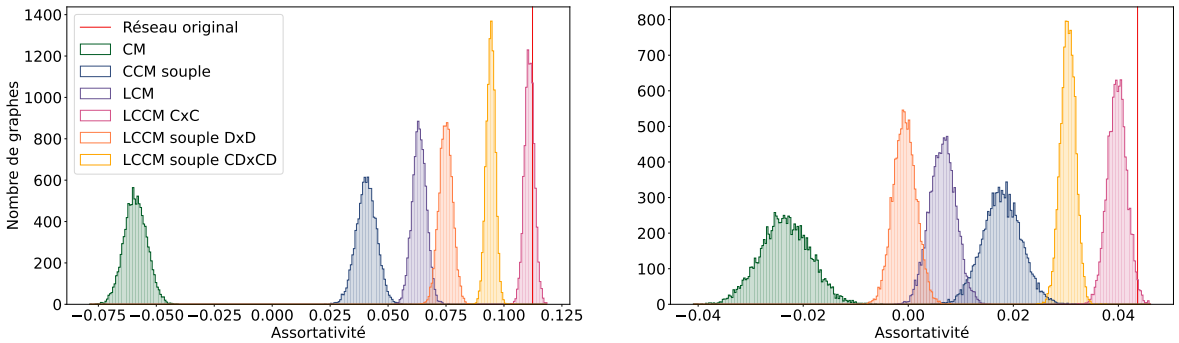

FIGURE 2.5 – Distribution d'assortativité sur des échantillons de 10000 graphes des ensembles. Les algorithmes ont été appliqués sur deux réseaux d'amitiés de Facebook [32], celui de gauche ayant 1429 noeuds et celui de droite 5793. La barre rouge indique la valeur d'assortativité de ces deux réseaux réels.

pour les graphes avec un grand nombre de noeuds. Il est possible que des effets de taille finie viennent empêcher une assortativité complètement nulle, et c'est ce qu'on observe dans la figure 2.5.

La figure 2.5 permet maintenant de répondre à un questionnement qui était resté en suspens à la section précédente, à savoir si l'ajout des contraintes sur la décomposition en oignon joue sur la corrélation des degrés. En effet, les résultats de la figure 2.3 (de gauche) semblent indiquer que l'ajout de la contrainte ne change pas l'erreur entre les deux matrices de degrés joints, et il n'était ainsi pas possible de conclure sur l'influence que la décomposition en oignon a sur les corrélations. En observant la figure 2.5, on voit que le LCM a des distributions d'assortativité bien différentes de celles du CM, ce qui montre finalement qu'il y a bien une corrélation entre l'assortativité et la décomposition en oignon comme on pouvait s'y attendre. Comme la contrainte sur la décomposition en oignon force la conservation des k-coeurs, on pourrait s'attendre à ce que l'algorithme conserve plus les liens entre les noeuds de haut degrés en empêchant des noeuds de haut coeur de se mélanger avec des noeuds de plus petits coeurs. L'absence d'effet à la figure 2.3 peut s'expliquer par la composition du graphe utilisé. En effet, une grande partie des noeuds du $G(n, m)$ utilisé sont dans le même coeur et dans les mêmes couches, ce qui réduit la diversité d'information conservée par l'ajout de la contrainte sur la décomposition en oignon. En général, l'ajout de cette contrainte donnera plus d'information sur les graphes qui ont une grande diversité de couches et de coeurs, ce qui est le cas dans la figure 2.5.

Finalement, la figure 2.5 permet de voir comment les différents ensembles peuvent aider à rejeter des hypothèses nulles. On utilise d'abord les valeurs d'assortativité obtenues pour chaque ensemble et on fait un ajustement de courbe gaussienne de la forme

$$f(x) = Ae^{\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}. \tag{2.5}$$

L'optimiseur ajuste ainsi les paramètres A, μ et σ. On note qu'en laissant le paramètre A libre la fonction obtenue n'est pas normalisée, mais cette forme permet de faciliter grandement la convergence de l'optimiseur numérique. L'ajustement d'une gaussienne permet d'estimer la distribution d'assortativité pour un ensemble donné, et les paramètres μ et σ représentent, respectivement, la moyenne et l'écart-type de la distribution. La distribution normale est choisie en première approximation seulement ; il n'est en effet pas possible d'appliquer directement le théorème central limite sur la mesure d'assortativité, puisque les variables aléatoires sur lesquelles on somme (les entrées de la matrice de corrélation de degrés) ne sont pas indépendantes en raison des contraintes sur les ensembles. Des versions du théorème pour des variables aléatoires faiblement dépendantes existent et pourraient expliquer la forme de la distribution de l'assortativité [7, 34], mais ces théorèmes ne seront pas explorés dans ce mémoire.

Une fois l'estimation de la distribution obtenue, il est possible de calculer la valeur-p de l'assortativité du graphe original, c'est-à-dire la probabilité qu'un graphe issu de l'ensemble ait une valeur d'assortativité au moins aussi extrême que celle du graphe original. Cette valeur permet finalement de quantifier à quel point l'hypothèse nulle est significative d'un point de vue statistique, c'est-à-dire à quel point les contraintes imposées sur l'ensemble peuvent expliquer la mesure sur le graphe original. La valeur-p est ici calculée avec

$$p(a) = \frac{\int_a^\infty f(x)dx}{\int_{-\infty}^\infty f(x)dx} \tag{2.6}$$

où a est la valeur extrême que l'on veut évaluer, et où la division par l'intégrale sur tout le domaine permet de normaliser le résultat obtenu. Dans le cas des deux réseaux d'amitié de la figure 2.5, les valeurs-p obtenues sont toutes presque nulles, sauf dans le cas du LCCM avec contrainte sur les corrélations de couches, où on obtient une valeur-p de $0,26$ pour le réseau de gauche, et une valeur-p de $0,018$ pour celui de droite. Ainsi, on voit que les contraintes uniquement sur la séquence de degrés, sur la décomposition en oignon, ou les contraintes souples sur la corrélations en degrés ne permettent pas d'expliquer la valeur d'assortativité beaucoup plus élevée dans le réseau original.

On peut répéter le même exercice avec une autre mesure scalaire. On choisit ici le *coefficient d'agrégation global* ou *coefficient de clustering global*, qui est une mesure indiquant la tendance des noeuds à se regrouper [68]. L'adage « Les amis de mes amis sont mes amis » représente bien cette mesure, qui compte le nombre de triangles [10] qui sont dans le graphe comparé au nombre de triplets de noeuds qui sont connectés sans être nécessairement fermés. On calcule le coefficient global d'agrégation avec la formule suivante [68] :

$$C = \frac{3 \times \text{Nombre de triangles}}{\text{Nombre de triplets connectés}}. \tag{2.7}$$

10. Les triangles correspondent à des cycles de longueur 3 dans un graphe.

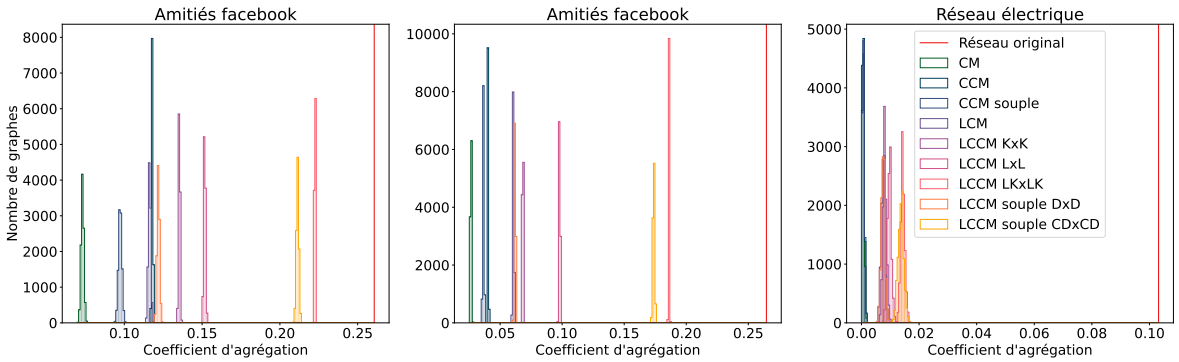

FIGURE 2.6 – Distribution du coefficient d'agrégation global sur des échantillons de 10000 graphes des ensembles. Les algorithmes ont été appliqués sur deux réseaux d'amitiés de Facebook [32], de 1429 noeuds (gauche) et de 5793 noeuds (centre), et sur le réseau électrique de l'ouest des États-Unis [90]. La barre rouge indique la valeur du coefficient pour les réseaux réels.

La valeur varie entre 1, qui indique que tous les triangles existent dans le graphe, et 0, qui arrive s'il n'y a pas de triangles, par exemple pour des graphes en arbres ou des réseaux en treillis avec uniquement des carrés.

La figure 2.6 montre les différentes distributions des coefficients d'agrégations globaux pour les différents ensembles. Comme pour l'assortativité, on note d'abord que le mélange avec le modèle des configurations enlève les triangles dans le graphe d'origine. Il s'agit d'un résultat attendu pour de grands graphes : le modèle des configurations donne des graphes localement en arbre dans la limite thermodynamique, et la valeur du coefficient tombe en $1/n$ [68, 69]. Tout dépendamment de la forme des matrices conjointes ou de la décomposition en oignon, on pourrait s'attendre à ce que les autres ensembles aient aussi une agrégation nulle dans la limite thermodynamique. En effet, ces mesures devraient avoir peu d'influence sur la formation de triangles dans un réseau.

On remarque que ce n'est pas le cas pour les réseaux réels explorés dans la figure 2.6 : les distributions des ensembles avec beaucoup de contraintes ont une valeur d'agrégation généralement plus élevée (surtout dans le cas des réseaux d'amitiés). On peut expliquer ce résultat par le fait que les réseaux ont un nombre de noeuds suffisamment petit faisant en sorte que certains échanges sont empêchés par des effets de taille finie. Le mélange conserve ainsi certaines structures importantes qui contribue grandement à la mesure observée. Ceci est aussi vrai pour le coefficient d'agrégation que pour l'assortativité de la figure 2.5.

Finalement, la figure 2.6 permet de déduire la plausibilité des hypothèses nulles pour expliquer l'agrégation des graphes réels. Cette fois, les valeur-p obtenues pour tous les ensembles sont toujours très proches de zéro pour tous les ensembles et tous les réseaux, et on voit donc qu'aucune des contraintes ne peut expliquer à elle seule les valeurs d'agrégation originales.

On peut tout de même observer une différence qualitative dans le fait que les ensembles plus contraints ont des distributions qui sont plus près de la valeur d'origine. Ainsi, on peut supposer que les contraintes jouent un rôle sur l'agrégation des réseaux, même si d'autres hypothèses (d'autres contraintes) seraient nécessaires pour se rapprocher du réseau réel. On note cependant que pour le réseau électrique, la valeur élevée d'agrégation n'est reproduite par aucun des ensembles, et on peut en conclure que le rôle de la décomposition et des corrélations de degrés ou de couches est moins significatif que dans le cas des réseaux d'amitié. Ceci illustre bien la différence topologique qu'il y a entre des réseaux sociaux et un réseau électrique.

2.4.2 Applications de dynamiques

Une autre manière de comparer l'influence des contraintes est de comparer le comportement de dynamiques sur les ensembles. On choisit ici de travailler avec une dynamique de propagation, soit une dynamique Susceptible-Infecté-Susceptible ou SIS à temps continu [62]. Dans cette dynamique, les noeuds peuvent prendre deux états différents, soient l'état *infecté* (I) ou l'état *susceptible* (S). Lorsqu'un noeud est infecté, il peut transmettre son infection à ses voisins susceptibles avec un taux τ, ou guérir de son infection pour devenir susceptible à un taux γ. La dynamique peut avoir deux comportements différents : la propagation peut mourir, auquel cas on atteint un état d'équilibre lorsque I, le nombre de noeuds infectés, atteint 0. Autrement, on peut atteindre un état dit *endémique*, où la valeur I atteindra un certain nombre de noeuds et restera autour de cette valeur pour le reste de la simulation.

La figure 2.7 montre le nombre de noeuds infectés selon le temps de simulation obtenu en moyenne lorsque l'on applique cette dynamique sur différents ensembles et différents réseaux. On y remarque d'abord une grande différence de comportement selon l'ensemble observé ; l'ajout de contraintes change grandement l'allure des courbes et donc le comportement de la propagation dans les réseaux. Ceci montre bien comment les différentes contraintes peuvent conserver des structures qui influencent directement la dynamique.

Si on observe le comportement sur le réseau de collaborations arXiv, on remarque qu'il semble y avoir une convergence vers des états stationnaires différents selon les différents ensembles. Pour des raisons de mémoire, le calcul n'a pas pu être effectué plus longtemps et il est ainsi difficile de conclure clairement sur ce sujet, mais le fait que certains ensembles donnent des états stationnaires différents a déjà été observé dans la littérature [43]. Néanmoins, on remarque que les modèles qui obtiennent les courbes les plus proches de la courbe du réseau réel sont ceux qui ont le plus de contraintes, plus particulièrement les ensembles avec des contraintes rigides sur les corrélations en couches.

On remarque la même chose pour le réseaux d'amitié, c'est-à-dire que les nouveaux ensembles permettent de mieux épouser la courbe de propagation du réseau réel. Pour le réseau électrique, la situation est un peu différente, du fait que les seuls ensembles qui se distinguent

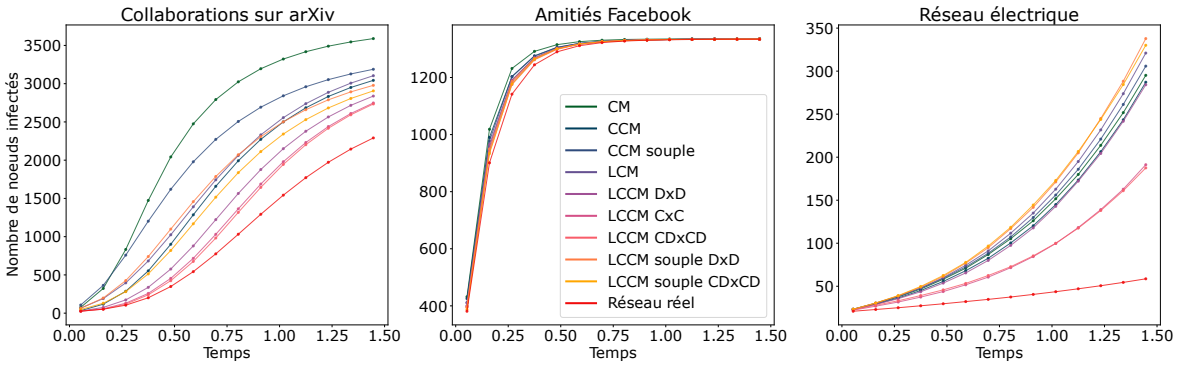

FIGURE 2.7 – Nombre moyen de noeuds infectés en fonction du temps pour une dynamique SIS avec $\tau = 1,2$ et $\gamma = 1$. 20 simulations ont été effectuées sur un échantillon de 1000 graphes issus des différents ensembles, la moyenne pour chaque ensemble a été calculée sur ces 20000 simulations. Les algorithmes ont été appliqués sur des réseaux réels représentant (à gauche) un réseau de collaborations sur arXiv [53] de 26197 noeuds, le réseau d'amitiés Facebook de 1429 noeuds [32] et le réseau électrique [90]. La courbe rouge indique le comportement moyen sur 20000 simulations sur le graphe réel.

sont ceux avec les contraintes rigides sur les corrélations en couche. Ceci indique que la séquence de degrés, la corrélation de degrés et la décomposition ne sont probablement pas des hypothèses suffisantes pour expliquer la lente propagation sur un réseau électrique. La lenteur de la propagation peut certainement s'expliquer par la structure du réseau, dans laquelle les chemins entre les noeuds sont généralement très longs. Cette organisation fait aussi en sorte qu'il y a une grande variété de couches, et le fait de conserver les corrélations entre ces couches force la conservation de ces longs chemins, ce qui permet aux modèles avec cette contrainte de mieux reproduire la lente propagation. Pour ce réseau, il est possible que les états stationnaires soient différents, mais la propagation est trop atypique et la simulation trop courte pour pouvoir conclure à ce sujet avec certitude.

Les résultats obtenus dans cette section, autant sur les propriétés des modèles que sur le comportement de dynamiques sur ces derniers, ont d'abord permis de comparer les propriétés des différents ensembles, mais ils ont aussi aidé à justifier l'utilité de ces ensembles. En effet, les résultats obtenus montrent que les nouvelles contraintes permettent d'obtenir des distributions que les modèles existants (CM et CCMs) ne pouvaient pas obtenir. Il serait évidemment très intéressant de continuer à pousser cette exploration. Entre autres, il serait intéressant d'ajouter des comparaisons à d'autres modèles de graphes, notamment ceux avec des contraintes sur les k-coeurs [42, 86].

Un autre type d'exploration qui pourrait apporter beaucoup d'intuition sur les ensembles serait l'analyse du spectre des matrices d'adjacence des ensembles. L'analyse spectrale des graphes est un domaine en soi et recèle de résultats très intéressants [19, 60, 70]. Une analyse de ce genre a déjà été faite pour le LCM seul dans le cadre d'un cours, et plusieurs résultats en

sont sortis, notamment le fait que le LCM conserve beaucoup de motifs, qu'il sépare bien les communautés de noeuds selon leur coeur, et aussi qu'il ne crée jamais de cycle dans un graphe en arbre. Une continuation de cette analyse, pour le LCM et les autres ensembles, fait partie de la liste de travaux futurs.

Conclusion

Les modèles de graphes aléatoires sont essentiels pour bien comprendre les réseaux et, ultimement, les systèmes complexes dont ils représentent la structure. Tout particulièrement, le modèle des configurations, de par sa simplicité, a été utilisé comme modèle nul à de nombreuses reprises. Le travail présenté dans ce mémoire vient complexifier ce modèle en ajoutant des contraintes rigides sur la décomposition en oignon et des contraintes rigides ou souples sur les corrélations de degrés, de couches, ou de (degré,couche), le but de cette complexification étant d'obtenir des réseaux qui possèdent des caractéristiques plus réalistes et ainsi obtenir des comparaisons plus précises.

L'échantillonnage de ces modèles de graphes aléatoires est un défi qui demande une vérification minutieuse des algorithmes utilisés. Une mauvaise implémentation peut amener des ensembles mal biaisés, et éventuellement donner des conclusions erronées à propos des données réelles [6, 15, 35]. Dans ce mémoire, une attention particulière a été portée sur le développement des algorithmes d'échantillonnages et sur la théorie derrière ces algorithmes, ce qui permet de bien comprendre les possibles erreurs et voir comment les éviter. La théorie sur les chaînes de Markov et le lien avec l'échantillonnage de graphes est éparpillée dans la littérature sous différents domaines ; la série de théorèmes présentée au chapitre 1 est une compilation des concepts pertinents pour le projet et se veut une référence qui unifie ces nombreuses bribes d'information.

Les contributions originales du projet se retrouvent au chapitre 2. On y présente les 7 nouveaux ensembles, issus du modèle des configurations, ainsi que les algorithmes pour les échantillonner. Des preuves montrant que les algorithmes convergent vers les distributions recherchées sont présentées, et une vérification numérique permet de confirmer que 6 algorithmes sur 7 obtiennent les bons échantillons. Finalement, le chapitre 2 se termine avec une application des ensembles sur des données réelles qui s'avère être courte, mais qui montre clairement la pertinence des nouveaux ensembles en tant que modèles nuls. On y voit que les ensembles ont des distributions d'assortativité et de coefficient d'agrégation qui dépendent des contraintes, et que les contraintes influencent grandement le comportement de dynamiques SIS.

Il reste de nombreuses questions non élucidées sur ce projet. D'abord pour l'échantillonnage, il reste encore à régler la question de l'irréductibilité pour savoir si les échanges de liens sont

suffisants pour explorer l'entièreté des ensembles, de même que trouver les bonnes probabilités d'acceptation pour les contraintes souples sur les corrélations en couche. Autrement, il serait possible d'améliorer les algorithmes en utilisant une autre stratégie d'échantillonnage que le *switch and hold*. Ceci reste un autre problème sans réponse, et sa résolution permettrait d'augmenter grandement l'efficacité de l'échantillonnage.

L'approche MCMC au problème d'échantillonnage permet d'assurer une convergence vers la bonne distribution peu importe l'état initial de la chaîne et est une méthode relativement simple à implémenter, du fait qu'elle nécessite tout simplement de trouver les bonnes règles ou probabilités d'acceptations. Cependant, malgré ses avantages, elle vient aussi avec certains inconvénients, notamment le fait qu'elle nécessite de nombreuses itérations avant de converger vers la bonne distribution. Un algorithme génératif est une autre approche possible, qui pourrait éventuellement assurer une convergence instantanée vers la bonne distribution. Il n'existe peut-être pas de tel algorithme, ou peut-être pas d'algorithme avec une distribution uniforme sur l'ensemble, mais un mélange de MCMC qui utilise les échanges de liens avec un algorithme génératif pourrait possiblement accélérer l'échantillonnage. À ce sujet, la vitesse de convergence des algorithmes, qui a été mentionnée à plusieurs reprises dans le mémoire, reste une question importante et devrait inéluctablement faire partie de travaux plus détaillés sur les algorithmes. Il serait ainsi plus facile de s'assurer de la convergence lors d'application des algorithmes sur des réseaux réels ; la stratégie employée jusqu'à présent est naïvement de faire beaucoup d'échanges et espérer que la chaîne soit assez avancée pour avoir une bonne convergence. On note cependant que les résultats présentés dans le chapitre 2 restent valides du fait qu'ils ont été testés pour différents nombres d'itérations et que les résultats n'étaient jamais changés. Dans cette optique, un travail futur important sera d'appliquer les outils récents [29] pour évaluer le temps de convergence des ensembles du projet.

Pour ce qui est des applications sur les réseaux réels, les premières expériences effectuées dans la section 2.4 montrent que les nouvelles hypothèses nulles sont utiles et permettent d'expliquer des propriétés des réseaux ou des comportements des dynamiques. Il serait donc intéressant de continuer à approfondir ces résultats, notamment en observant plus de mesures et plus de dynamiques, et en augmentant la quantité de données réelles. La librairie développée avec le projet va dans cette direction, l'idée étant que la communauté scientifique puisse utiliser les algorithmes directement et rapidement sur leurs données. Le projet aura donc permis d'augmenter considérablement la disponibilité des modèles nuls dans la littérature.

Dans cette avenue, un travail futur important sera de comparer les résultats des nouveaux ensembles à plus de modèles nuls déjà existants. Notamment, la comparaison à des ensembles qui conservent exactement la décomposition en k-coeurs [86] sera extrêmement révélatrice, puisqu'elle permettra de bien comprendre la pertinence d'ajouter les couches à cette mesure, de même que la pertinence de conserver la séquence de degrés.

Pour terminer, nous notons une dernière fois la pertinence de l'approche utilisée pour la comparaison des données réelles à une hypothèse nulle : la stratégie basée sur les données de créer des algorithmes d'échantillonnage permet d'obtenir rapidement des ensembles de graphes, et, avec des développements numériques de plus en plus efficaces, de comparer efficacement de nombreuses hypothèses. Continuer le développement de cette branche permettra d'augmenter la collection de techniques pour obtenir des échantillons, ce qui permettra à son tour d'agrandir la quantité de modèles disponibles. Couplée avec un développement théorique solide qui permettra notamment d'obtenir des intuitions sur les hypothèses dans les limites des très grands graphes, cette approche saura sans nul doute éclairer notre compréhension des systèmes complexes.

Annexe A

Test d'échange pour la décomposition en oignon - version colorée

Tout au long du mémoire, il est utile de penser en terme de couleurs de demi-liens lorsque l'on traite de la décomposition en oignon. Ceci est encore plus vrai lorsque l'on tente de développer des preuves théoriques. Une traduction du test est donc offerte dans cet annexe :

Algorithme 5 Test d'échange pour la décomposition en oignon - version colorée

entrée : Graphe G_{i-1} de l'itération précédente, noeud a, voisin b actuel de a et voisin c potentiel de a

sortie : **vrai** si l'échange change la décomposition en oignon, **faux** sinon

```
c ← coeur du noeud a
ℓ_a ← couche du noeud a
s_b ← couleur du demi-lien associé à b
ℓ_c ← couleur du demi-lien associé à c (une fois l'échange effectué)
n_r ← nombre de demi-liens rouges de a
n_rb ← nombre de demi-liens rouges et noirs de a
si l_a est la première couche du coeur c alors
  si s_b est rouge alors
    si s_c est rouge alors
      retourne faux
    sinon
      retourne vrai
    fin si
  sinon si s_b est noir ou vert alors
    si s_c est rouge alors
      retourne faux
    sinon
```

```
        retourne vrai
      fin si
    fin si
  sinon
    si s_b est rouge alors
      si n_rb = c + 1 alors
        si s_c est rouge ou noir alors
          retourne faux
        sinon
          retourne vrai
        fin si
      sinon
        retourne faux
      fin si
    sinon si s_b est noir alors
      si n_rb = c + 1 alors
        si s_c est rouge ou noir alors
          retourne faux
        sinon
          retourne vrai
        fin si
      sinon si n_r = c alors
        si s_c est noir ou vert alors
          retourne faux
        sinon
          retourne vrai
        fin si
      sinon
        retourne faux
      fin si
    sinon
      si n_r = c alors
        si s_c est noir ou vert alors
          retourne faux
        sinon
          retourne vrai
        fin si
      sinon
        retourne faux
```

```
      fin si
    fin si
  fin si
```

Annexe B

Piste de preuve alternative pour l'apériodicité du LCM

La preuve 20 présentée pour l'apériodicité repose sur la méthode de *swap and hold*. Comme il a été mentionné dans le texte, cette méthode n'est pas la plus efficace, et trouver une solution qui ne propose que les échanges valides permettrait d'améliorer la vitesse de convergence et ainsi de réduire la taille nécessaire pour l'échantillon. Cependant, avec une telle méthode, il faudra revisiter la preuve de l'apériodicité pour qu'elle soit applicable sans que l'on repose sur les boucles dans le graphe de graphes. L'idée serait alors de rejoindre la preuve du théorème 18, qui prouve l'existence d'un 2-cycle et d'un 3-cycle pour tous les graphes, et donc qui montre que le graphe de graphes sera toujours apériodique. Le présent annexe n'offre pas de preuve complète, mais offre des pistes de solution qui pourraient éventuellement être appliquées.

L'existence du 2-cycle se justifie de la même manière que pour le modèle des configurations, c'est-à-dire par la réversibilité des échanges de liens. C'est l'existence du 3-cycle qui devient plus ardue, puisqu'il n'est peut-être pas toujours possible de trouver des liens qui peuvent faire la suite $(u, v); (x, y) \to (u, x); (v, y) \to (u, y); (v, x) \to (u, v); (x, y)$.

Un premier cas à considérer, qui est assez simple, est le cas où deux liens rouge-rouge existent dans la même couche. Deux liens rouge-rouge dans la même couche peuvent toujours s'échanger, et donc le 3-cycle est valide. Autrement, il existe d'autres configurations du genre qui peuvent être utilisées. Trois cas sont présentés à la figure B.1, qui comprennent des noeuds dans des couches ℓ et $\ell - 1$. Ces trois cas peuvent aussi être généralisés pour des liens dans des couches ℓ et ℓ' quelconques, c'est-à-dire pour des liens rouge-verts.

Pour montrer que le graphe de graphes du LCM est apériodique pour tous les graphes, il suffirait alors de montrer qu'il existe toujours au moins une de ces paires de liens. Il est probable que les 7 configurations présentées ici soient suffisantes, il resterait à le montrer rigoureusement. Autrement, il faudra trouver d'autres cas qui permettent de généraliser la

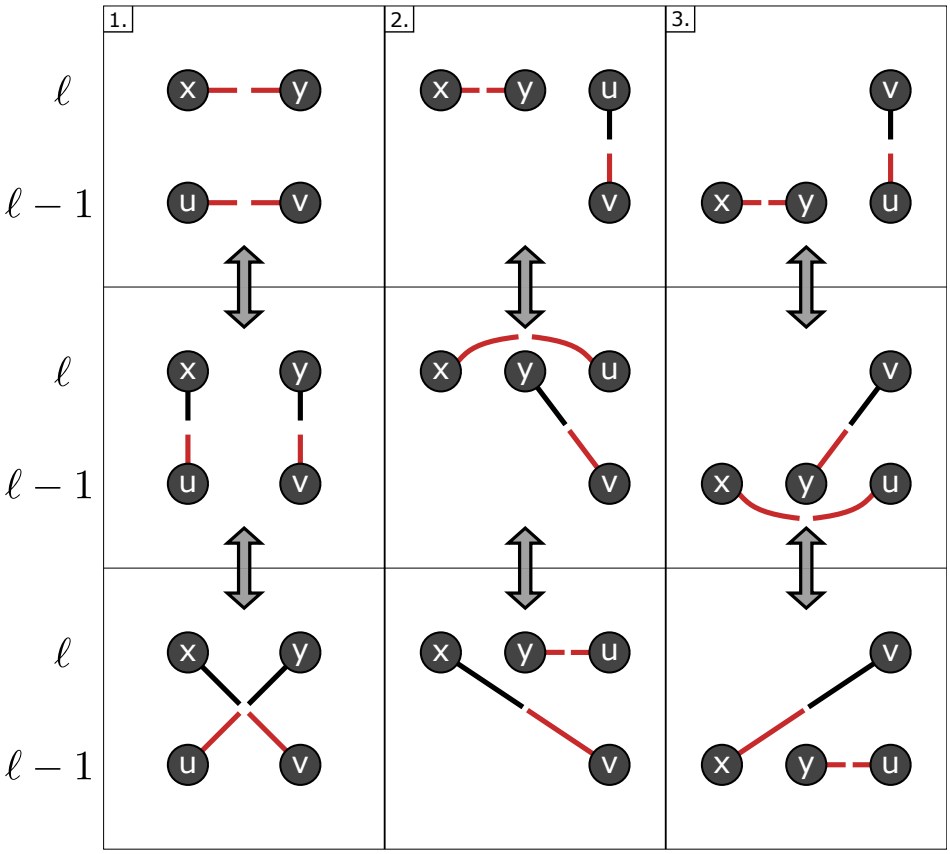

FIGURE B.1 – Exemples de 3-cycles qui peuvent être utilisés pour la preuve d'apériodicité. Les cas sont présentés en colonnes, c'est-à-dire que l'on peut échanger de haut en bas, et aussi de la ligne du haut à la ligne du bas. Les couches des noeuds sont indiquées par leur hauteur. Les couleurs de demi-liens sont affichées.

preuve pour tous les graphes.

Annexe C

Réductibilité du LCCM souple

Comme il s'agit d'une « recalibration » d'un ensemble à contraintes rigides, un ensemble à contraintes souples devrait en théorie contenir exactement les mêmes graphes que ce que ses contraintes rigides lui permettent. Pour le LCCM CxC par exemple, on s'attendrait à ce que cet ensemble contienne les mêmes graphes que le LCM, puisqu'il possède les mêmes contraintes rigides que ce dernier. De plus, d'un point de vue mathématique, lorsque l'on cherche à résoudre l'équation de Lagrange pour un modèle canonique, la solution pour la distribution est une exponentielle avec des paramètres $\lambda_i \in \mathbb{R}$ (avec un paramètre λ_i par contrainte) [23]. Pour avoir une probabilité nulle sur une configuration particulière, il faudrait avoir $P(G) \propto \exp\{\lambda_i\}$ avec un $\lambda_i = -\infty$, ce qui n'est pas possible dans le domaine donné pour les λ_i. Donc, pour un ensemble canonique, toutes les configurations devraient au moins avoir une probabilité plus grande que zéro d'être obtenues [1].

Les probabilités 2.1 pour l'algorithme 8 ne permettent cependant pas toujours cela. En effet, il existe certains cas où l'algorithme donne des contraintes trop fortes pour obtenir certains graphes de l'ensemble, c'est-à-dire que l'algorithme 8 n'est pas toujours irréductible même s'il sert à explorer un ensemble à contraintes souples.

L'exemple de la figure C.1 montre bien le problème créé par les probabilités d'acceptations. Les deux graphes illustrés dans la figure font partie du LCCM CxC souple puiqu'ils ont la

1. Cette justification n'est pas rigoureuse et il ne serait pas impossible d'avoir des exemples de cas où un sous-ensemble se retrouve avec une distribution nulle pour maximiser l'entropie (surtout dans notre cas où le support est discrétisé). Dans cette optique, le LCM et le LCCM souple ne contiendraient pas exactement les mêmes graphes comme on vient de l'assumer. Cependant, ce n'est pas le cas pour l'exemple présenté dans cette annexe ; le problème de connectivité est bien causé par l'algorithme.

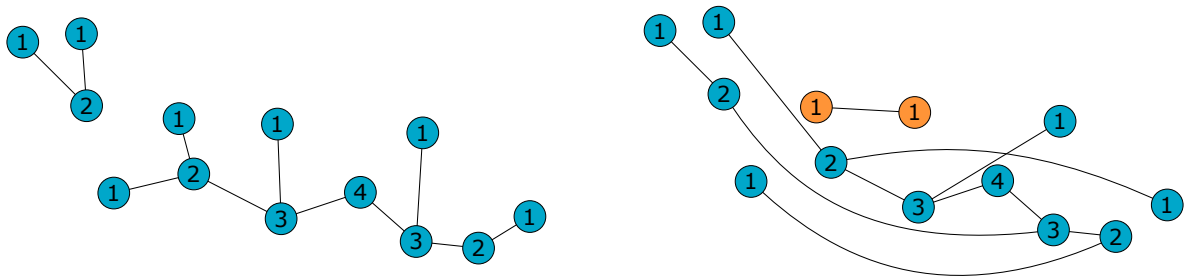

FIGURE C.1 – Deux graphes faisant partie du même ensemble LCM, et donc par extension du même ensemble LCCM souple CxC. Les noeuds font tous partie du 1-coeur, et la couche de chaque noeud est indiqué sur le noeud. On peut passer du graphe de gauche au graphe de droite par la suite d'échanges suivante : $(3,4);(10,12) \to (3,10);(4,12)$, $(1,2);(4,5) \to (1,5);(4,12)$ puis $(1,5);(6,7) \to (1,6);(5,7)$. Les noeuds oranges indiquent le lien qui cause problème avec les probabilités d'acceptation.

même décomposition en oignon. La matrice de couches jointes du graphe de gauche est

$$e = \begin{pmatrix} 0 & 5 & 2 & 0 \\ 5 & 0 & 2 & 0 \\ 2 & 2 & 0 & 2 \\ 0 & 0 & 2 & 0 \end{pmatrix}, \tag{C.1}$$

et celle du graphe de droite est

$$e = \begin{pmatrix} 1 & 4 & 1 & 0 \\ 4 & 0 & 3 & 0 \\ 1 & 3 & 0 & 2 \\ 0 & 0 & 2 & 0 \end{pmatrix}. \tag{C.2}$$

Supposons que l'algorithme débute à partir du graphe de gauche. Lorsque l'on calcule les probabilités d'acceptation, on utilise la première matrice. Pour obtenir le graphe de droite, il faut invariablement créer le lien illustré par des noeuds oranges. Comme on essaie de créer un lien de type $(\ell = 1, \ell = 1)$, la probabilité pour n'importe quel échange qui crée ce lien sera toujours

$$\mathcal{A} = \min\left\{1, \frac{e(1,1)e(\ell_b,\ell_d)}{e(\ell_a,\ell_b)e(\ell_c,\ell_d)}\right\} = 0 \tag{C.3}$$

puisque $e(1,1) = 0$. L'échange sera donc toujours refusé, et le graphe est exclu de l'ensemble LCCM échantillonné, même s'il devrait techniquement pouvoir en faire partie. Donc, dans ce cas, les probabilités d'acceptations sont responsables de couper la connectivité de l'ensemble.

Si la chaîne commençait plutôt à partir du graphe de droite, la matrice utilisée pour calculer les probabilité d'acceptation serait la deuxième. Dans ce cas, il serait possible de faire le chemin vers le graphe de gauche, puisqu'il n'y a pas d'entrée nulle qui empêche la création d'un lien.

Au final, cet exemple montre bien comment les probabilités d'acceptation peuvent briser la connectivité de l'ensemble, même si on travaille dans des contraintes souples. Cette situation survient à chaque fois qu'une entrée de la matrice jointe qui sert de contrainte est nulle. Il s'agit donc techniquement d'un effet de taille finie, puisque plus le graphe possède de noeuds, moins il est probable d'avoir une matrice de corrélation avec des entrées nulles. En général, ce problème sera donc ignoré, mais il est pertinent de le garder en tête puisqu'il pourrait nuire à l'échantillonnage pour certains graphes.

Bibliographie

[1] A. Allard et L. Hébert-Dufresne : Percolation and the effective structure of complex networks. *Phys. Rev. X*, 9:011023, 2019.

[2] G. Amanatidis, B. Green et M. Mihail : Graphic realizations of joint-degree matrices, 2015.

[3] G. Amanatidis et P. Kleer : Rapid Mixing of the Switch Markov Chain for 2-Class Joint Degree Matrices. *SIAM J. Discrete Math.*, 36:118–146, 2022.

[4] P. W. Anderson : More is different. *Science*, 177:393–396, 1972.

[5] P. W. Anderson : Is complexity physics ? is it science ? what is it ?, Jul 1991.

[6] Y. Artzy-Randrup et L. Stone : Generating uniformly distributed random networks. *Phys. Rev. E*, 72:056708, 2005.

[7] N. K. Bakirov : Central limit theorem for weakly dependent variables. *Math. Notes*, 41:63–67, 1987.

[8] D. L. Banks et K. M. Carley : Models for network evolution. *J. Math. Sociol.*, 21:173–196, 1996.

[9] A.-L. Barabási : The network takeover. *Nat. Phys.*, 8:14–16, 2012.

[10] A.-L. Barabási et R. Albert : Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

[11] V. Batagelj et M. Zaveršnik : Fast algorithms for determining (generalized) core groups in social networks. *Advances in Data Analysis and Classification*, 5:129–145, 2011.

[12] H. E. Bell : Gershgorin's theorem and the zeros of polynomials. *The American Mathematical Monthly*, 72:292–295, 1965.

[13] L. Bogacz, Z. Burda, W. Janke et B. Waclaw : A program generating homogeneous random graphs with given weights. *Comput. Phys. Commun.*, 173:162–174, 2005.

[14] Z. Burda, J. D. Correia et A. Krzywicki : Statistical ensemble of scale-free random graphs. *Phys. Rev. E*, 64:046118, 2001.

[15] C. J. Carstens et K. J. Horadam : Switching edges to randomize networks : What goes wrong and how to fix it. *J. Complex Netw.*, 5:337–351, 2017.

[16] D. Centola, V. M. Eguíluz et M. W. Macy : Cascade dynamics of complex propagation. *Physica A*, 374:449–456, 2007.

[17] F. Chung et L. Lu : The average distances in random graphs with given expected degrees. *Proc. Natl. Acad. Sci. U.S.A.*, 99:15879–15882, 2002.

[18] F. Chung, L. Lu, T. G. Dewey et D. J. Galas : Duplication models for biological networks. *Journal of Computational Biology*, 10:677–687, 2003.

[19] F. R. K. Chung : *Spectral Graph Theory.* American Mathematical Society, 1997.

[20] E. F. Connor et D. Simberloff : The assembly of species communities : Chance or competition ? *Ecology*, 60:1132–1140, 1979.

[21] S. J. Cook, T. A. Jarrell, C. A. Brittin, Y. Wang, A. E. Bloniarz, M. A. Yakovlev, K. C. Q. Nguyen, L. T.-H. Tang, E. A. Bayer, J. S. Duerr, H. E. Bülow, O. Hobert, D. H. Hall et S. W. Emmons : Whole-animal connectomes of both Caenorhabditis elegans sexes. *Nature*, 571:63–71, 2019.

[22] A. C. C. Coolen, A. De Martino et A. Annibale : Constrained markovian dynamics of random graphs. *J. Stat. Phys.*, 136:1035–1067, 2009.

[23] T. Coolen, A. Annibale et E. Roberts : *Generating Random Networks and Graphs.* Oxford University Press, 2017.

[24] C. COOPER, M. DYER et C. GREENHILL : Sampling regular graphs and a peer-to-peer network. *Comb. Probab. Comput.*, 16:557–593, 2007.

[25] C. Cooper, M. Dyer et C. Greenhill : Corrigendum : Sampling regular graphs and a peer-to-peer network, 2012.

[26] É. Czabarka, A. Dutle, P. L. Erdős et I. Miklós : On realizations of a joint degree matrix. *Discrete Applied Mathematics*, 181:283–288, 2015.

[27] J. A. Davis et S. Leinhardt : *The Structure of Positive Interpersonal Relations in Small Groups*, p. 141–145. Springer, Wiesbaden, 1967.

[28] S. N. Dorogovtsev et J. F. F. Mendes : Evolution of networks. *Advances in Physics*, 51:1079–1187, 2002.

[29] U. Dutta, B. K. Fosdick et A. Clauset : Sampling random graphs with specified degree sequences, 2021.

[30] P. Erdős et A. Rényi : On Random Graphs I. *Publicationes Mathematicae Debrecen*, 6:290–297, 1959.

[31] P. L. Erdös, I. Miklós et L. Soukup : Towards random uniform sampling of bipartite graphs with given degree sequence, 2010.

[32] M. Fire, R. Puzis et Y. Elovici : Organization mining using online social networks, 2013.

[33] R. A. Fisher : *The design of experiments.* Oliver & Boyd, 1935.

[34] M. Fleermann et W. Kirsch : The central limit theorem for weakly dependent random variables by the moment method, 2022.

[35] B. K. Fosdick, D. B. Larremore, J. Nishimura et J. Ugander : Configuring random graph models with fixed degree sequences. *SIAM Rev.*, 60:315–355, 2018.

[36] O. Frank et D. Strauss : Markov graphs. *J. Am. Stat. Assoc.*, 81:832–842, 1986.

[37] C. W. Gardiner *et al.* : *Handbook of stochastic methods.* springer, 1985.

[38] E. N. Gilbert : Random Graphs. *Annals of Mathematical Statistics*, 30:1141–1144, 1959.

[39] C. Greenhill : A polynomial bound on the mixing time of a markov chain for sampling regular directed graphs, 2011.

[40] C. Greenhill : The switch markov chain for sampling irregular graphs (extended abstract). *Proc. Annu. ACM-SIAM Symp. Discrete Algorithms*, 2015:1564–1572, 2015.

[41] C. Greenhill et M. Sfragara : The switch markov chain for sampling irregular graphs and digraphs. *Theoretical Computer Science*, 719:1–20, 2018.

[42] L. Hébert-Dufresne, A. Allard, J.-G. Young et L. J. Dubé : Percolation on random networks with arbitrary k-core structure. *Phys. Rev. E*, 88:062820, 2013.

[43] L. Hébert-Dufresne, J. A. Grochow et A. Allard : Multi-scale structure and topological anomaly detection via a new network statistic : The onion decomposition. *Sci. Rep.*, 6:31708, 2016.

[44] P. W. Holland et S. Leinhardt : An exponential family of probability distributions for directed graphs. *J. Am. Stat. Assoc.*, 76:33–50, 1981.

[45] R. A. Horn et C. R. Johnson : *Matrix Analysis.* Cambridge University Press, 2012.

[46] S. Horvát et C. D. Modes : Connectedness matters : Construction and exact random sampling of connected networks. *J. Phys. Complex.*, 2:015008, 2021.

[47] L. Hébert-Dufresne, J.-G. Young, A. Daniels et A. Allard : Network onion divergence : Network representation and comparison using nested configuration models with fixed connectivity, correlation and centrality patterns, 2022.

[48] S. Itzkovitz, R. Milo, N. Kashtan, G. Ziv et U. Alon : Subgraphs in random networks. *Phys. Rev. E*, 68:026127, 2003.

[49] R. Kannan, P. Tetali et S. Vempala : Simple markov-chain algorithms for generating bipartite graphs and tournaments. *Random Struct. Algorithms*, 14:293–308, 1999.

[50] S. R. Kharel, T. R. Mezei, S. Chung, P. L. Erdős et Z. Toroczkai : Degree-preserving network growth. *Nat. Phys.*, 18:100–106, 2022.

[51] M. Kitsak, L. K. Gallos, S. Havlin, F. Liljeros, L. Muchnik, H. E. Stanley et H. A. Makse : Identification of influential spreaders in complex networks. *Nat. Phys.*, 6:888–893, 2010.

[52] D.-S. Lee, K.-I. Goh, B. Kahng et D. Kim : Evolution of scale-free random graphs : Potts model formulation. *Nuclear Physics B*, 696:351–380, 2004.

[53] J. Leskovec, J. Kleinberg et C. Faloutsos : Graph evolution : Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data*, 1:2–es, 2007.

[54] D. A. Levin et Y. Peres : *Markov chains and mixing times*. American Mathematical Society, 2017.

[55] C. Lowcay, S. Marsland et C. McCartin : Constrained Switching in Graphs : A Constructive Proof. *In 2013 International Conference on Signal-Image Technology & Internet-Based Systems*, p. 599–604, 2013.

[56] F. D. Malliaros, C. Giatsidis, A. N. Papadopoulos et M. Vazirgiannis : The core decomposition of networks : Theory, algorithms and applications. *The VLDB Journal*, 29:61–92, 2020.

[57] R. D. Malmgren, J. M. Ottino et L. A. Nunes Amaral : The role of mentorship in protégé performance. *Nature*, 465:622–626, 2010.

[58] S. Maslov, K. Sneppen et A. Zaliznyak : Detection of topological patterns in complex networks : Correlation profile of the internet. *Physica A*, 333:529–540, 2004.

[59] C. D. Meyer : *Matrix Analysis and Applied Linear Algebra*. Society for Industrial and Applied Mathematics, 2000.

[60] P. v. Mieghem : *Graph Spectra for Complex Networks*. Cambridge University Press, 2010.

[61] J. C. Miller : Percolation and epidemics in random clustered networks. *Phys. Rev. E*, 80:020901, 2009.

[62] J. C. Miller et T. Ting : Eon (epidemics on networks) : A fast, flexible python package for simulation, analytic approximation, and analysis of epidemics on networks. *J. Open Source Softw.*, 4:1731, 2019.

[63] J. L. Moreno et H. H. Jennings : Statistics of social configurations. *Sociometry*, 1:342–374, 1938.

[64] M. E. J. Newman : Assortative Mixing in Networks. *Phys. Rev. Lett.*, 89:208701, 2002.

[65] M. E. J. Newman : Mixing patterns in networks. *Phys. Rev. E*, 67:26126, 2003.

[66] M. E. J. Newman : Component sizes in networks with arbitrary degree distributions. *Phys. Rev. E*, 76:045101, 2007.

[67] M. E. J. Newman : Random Graphs with Clustering. *Phys. Rev. Lett.*, 103:058701, 2009.

[68] M. E. J. Newman : *Networks : An introduction.* Oxford University Press, 2010.

[69] M. E. J. Newman, S. H. Strogatz et D. J. Watts : Random graphs with arbitrary degree distributions and their applications. *Phys. Rev. E*, 64:026118, 2001.

[70] B. Nica : *A Brief Introduction to Spectral Graph Theory.* EMS Press, 2018.

[71] J. Nishimura : The connectivity of graphs of graphs with self-loops and a given degree sequence. *J. Complex Netw.*, 6:927–947, 2018.

[72] R. Pastor-Satorras, E. Smith et R. V. Solé : Evolving protein interaction networks through gene duplication. *J. Theor. Biol.*, 222:199–210, 2003.

[73] J. Rosales et P. García-Sánchez : *Numerical Semigroups.* Developments in Mathematics. Springer, 2009.

[74] A. Sanil, D. Banks et K. Carley : Models for evolving fixed node networks : Model fitting and model testing. *Soc. Networks*, 17:65–81, 1995.

[75] S. B. Seidman : Network structure and minimum degree. *Soc. Networks*, 5:269–287, 1983.

[76] E. Seneta : Markov and the birth of chain dependence theory. *Int. Stat. Rev.*, 64:255–263, 1996.

[77] S. S. Shen-Orr, R. Milo, S. Mangan et U. Alon : Network motifs in the transcriptional regulation network of Escherichia coli. *Nature Genetics*, 31:64–68, 2002.

[78] K. Siegrist : Periodicity of discrete-time chains, Consulté le 6 août 2022.

[79] T. Snijders : Markov chain monte carlo estimation of exponential random graph models. *J. Soc. Struct.*, 3, 2002.

[80] T. A. B. Snijders : The statistical evaluation of social network dynamics. *Sociol. Methodol.*, 31:361–395, 2001.

[81] G. St-Onge, J.-G. Young, E. Laurence, C. Murphy et L. J. Dubé : Phase transition of the susceptible-infected-susceptible dynamics on time-varying configuration model networks. *Phys. Rev. E*, 97:022305, 2018.

[82] I. Stanton et A. Pinar : Constructing and sampling graphs with a prescribed joint degree distribution. *J. Exp. Algorithmics*, 17:3.1–3.25, 2012.

[83] D. B. Stouffer, J. Camacho, W. Jiang et L. A. N. Amaral : Evidence for the existence of a robust pattern of prey selection in food webs. *Proc. R. Soc. B*, 274:1931–1940, 2007.

[84] D. Strauss et M. Ikeda : Pseudolikelihood estimation for social networks. *J. Am. Stat. Assoc.*, 85:204–212, 1990.

[85] R. Taylor : Constrained switchings in graphs. *In Combinatorial Mathematics VIII*, p. 314–336, 1981.

[86] K. Van Koevering, A. Benson et J. Kleinberg : Random graphs with prescribed k-core sequences : A new null model for network analysis. *In Proceedings of the Web Conference 2021*, p. 367–378, 2021.

[87] A. Vázquez, A. Flammini, A. Maritan et A. Vespignani : Modeling of protein interaction networks. *Complexus*, 1:38–44, 2003.

[88] S. Wasserman et P. Pattison : Logit models and logistic regressions for social networks : I. an introduction to markov graphs and p*. *Psychometrika*, 61:401–425, 1996.

[89] D. J. Watts, P. S. Dodds, J. D. served as editor et T. E. served as associate editor for this article. : Influentials, networks, and public opinion formation. *Journal of Consumer Research*, 34:441–458, 2007.

[90] D. J. Watts et S. H. Strogatz : Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, 1998.

[91] W. Weaver : Science and complexity. *Am. Sci.*, 36:536–544, 1948.

[92] T. G. Will : Switching distance between graphs with the same degrees. *SIAM J. Discrete Math.*, 12:298–306, 1999.

[93] J. B. Wilson : Methods for detecting non-randomness in species co-occurrences : A contribution. *Oecologia*, 73:579–582, 1987.

[94] G. Zhang : Traversability of graph space with given degree sequence under edge rewiring. *Electronics Letters*, 46:351, 2010.