HAL
open science

Réseaux et signal : des outils de traitement du signal pour l'analyse des réseaux

Nicolas Tremblay

To cite this version:

Nicolas Tremblay. Réseaux et signal : des outils de traitement du signal pour l'analyse des réseaux. Autre [cond-mat.other]. Ecole normale supérieure de lyon - ENS LYON, 2014. Français. NNT : 2014ENSL0938 . tel-01078956

HAL Id: tel-01078956

https://theses.hal.science/tel-01078956v1

Submitted on 30 Oct 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Thèse

en vue de l'obtention du grade de :

Docteur de l'Université de Lyon,
délivré par l'École Normale Supérieure de Lyon

Discipline : **Physique**

Laboratoire de Physique

École Doctorale de Physique et d'Astrophysique de Lyon

présentée et soutenue publiquement le 9 Octobre 2014 par :

Nicolas TREMBLAY

Réseaux et signal : des outils de traitement du signal pour l'analyse des réseaux

Directeur de thèse :
Pierre BORGNAT

Après avis de :

M. Jean-Loup GUILLAUME	Professeur, L3i, La Rochelle	Rapporteur
M. Pierre VANDERGHEYNST	Professeur, EPFL, Lausanne	Rapporteur

Devant la commission d'examen formée de :

M. Alain BARRAT	Directeur de recherche, CPT, Marseille	Examinateur
M. Pierre BORGNAT	Chargé de recherche, ENS de Lyon	Directeur
M. Eric FLEURY	Professeur, ENS de Lyon	Examinateur
M. Jean-Loup GUILLAUME	Pofesseur, L3i, La Rochelle	Rapporteur
M. Pierre VANDERGHEYNST	Professeur, EPFL, Lausanne	Rapporteur

Sommaire

Introduction **1**

1 Généralités sur les graphes et le traitement du signal sur graphes **11**
1 Généralités sur les graphes 12
2 Notions de base du traitement du signal sur graphes 16
3 La transformée en ondelettes sur graphes 31

2 Détection multiéchelle de communautés à l'aide d'ondelettes **39**
1 Une communauté dans un graphe 41
2 Partitionner un graphe en communautés 44
3 État de l'art des méthodes de détection multiéchelle 48
4 Ondelettes sur graphes et partitionnement multiéchelle 57
5 Illustrations et comparaison avec d'autres méthodes 71
6 Application à un graphe de terrain 89
7 Définition et utilisation de fonctions d'échelle sur graphe 91
8 Réinterprétation de quelques méthodes multiéchelles 95
9 Conclusions et perspectives 104

3 Rééchantillonnage de groupes de nœuds dans un réseau **109**
1 Les outils classiques 110
2 Une méthode bootstrap pour des groupes de nœuds dans un réseau . 115
3 Étude contrôlée sur un modèle de réseaux complexes 123
4 Application à un réseau social 128
5 Discussion 140

Conclusion et perspectives **143**

A Mesures de similarité entre deux partitions **149**

B Modèle de graphe aléatoire : le modèle de Chung-Lu pondéré **151**

C La correction de Bonferroni **153**

D Strip, Bind, Search : Identifying Abnormal Energy Consumption **155**

E Graph Empirical Mode Decomposition **175**

Bibliographie **189**

Table des matières **205**

« Seule compte la démarche. Car c'est elle qui dure et non le but qui n'est qu'illusion du voyageur quand il marche de crête en crête comme si le but atteint avait un sens. »

– A. de Saint-Exupéry, *Citadelle*

Introduction

« Where is the Life we have lost in living ?
Where is the wisdom we have lost in knowledge ?
Where is the knowledge we have lost in information ? »

– T. S. Eliot, *The Rock*

Les travaux rapportés dans ce manuscrit sont ancrés dans une discipline scientifique à la frontière entre l'informatique, les mathématiques, la physique et les statistiques : le traitement du signal. Comment débruiter un signal ? Comment détecter si des données contiennent de l'information structurée, pertinente ? Comment utiliser les connaissances *a priori* que l'on peut avoir sur les données pour rendre l'analyse plus fine ? Le traitement du signal a historiquement pour objectifs de répondre à ces questions, et est aujourd'hui de plus en plus sollicité pour étudier des données toujours plus volumineuses et complexes.

L'augmentation conjointe des capacités de stockage et de calcul bouleverse la manière dont les sciences considèrent les données. Ces données, avant l'informatique, étaient nécessairement à taille humaine (c'est-à-dire lisibles en intégralité par un ou plusieurs humains), et étaient soigneusement récoltées dans le but de vérifier telle ou telle hypothèse préalablement formulée. Aujourd'hui, la situation est radicalement différente : de plus en plus, les données sont récoltées de manière systématique. Que ce soit par des entreprises comme Facebook, Google ou Amazon qui cherchent à avoir des profils d'utilisateurs précis, par nos banques pour des raisons de sécurité, par les sociétés de transport qui récoltent nos données dans le but, entre autres, d'améliorer le réseau de connexions ; par nous-mêmes, aussi, qui numérisons de plus en plus notre vie privée via nos photos, nos films, nos réseaux sociaux sur Internet, ... Étant donné qu'une partie non-négligeable de ces données est collectée automatiquement et sans buts précis, nous avons affaire à des données de moins en moins structurées, parsemées d'informations inutiles. L'augmentation de la taille des données, alliée à la diminution de leur structure, posent de nouvelles questions fondamentales dans le domaine du traitement du signal.

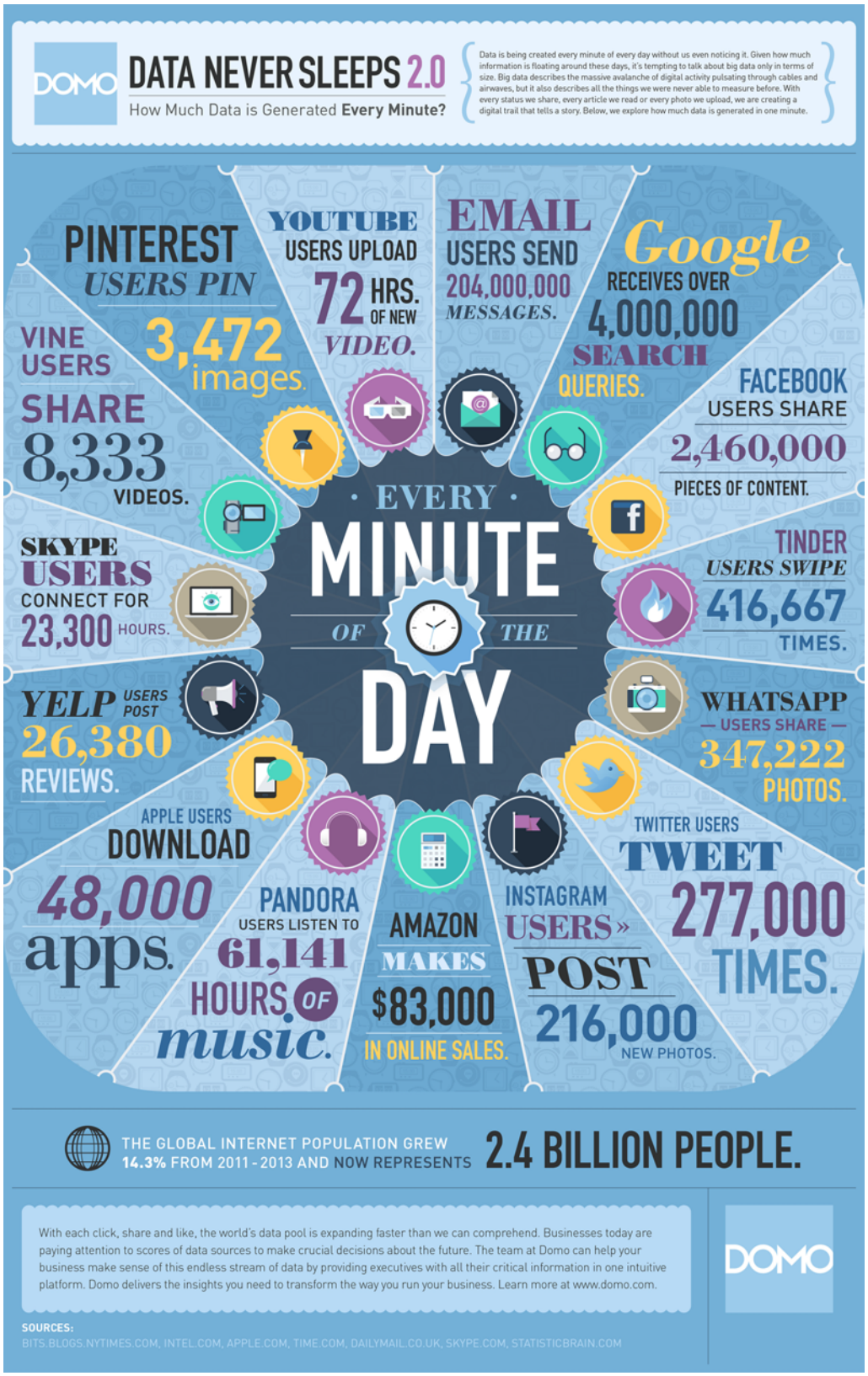


Figure 0.1: Quelques ordres de grandeurs de création de données numériques dans le monde *par minute* [14].

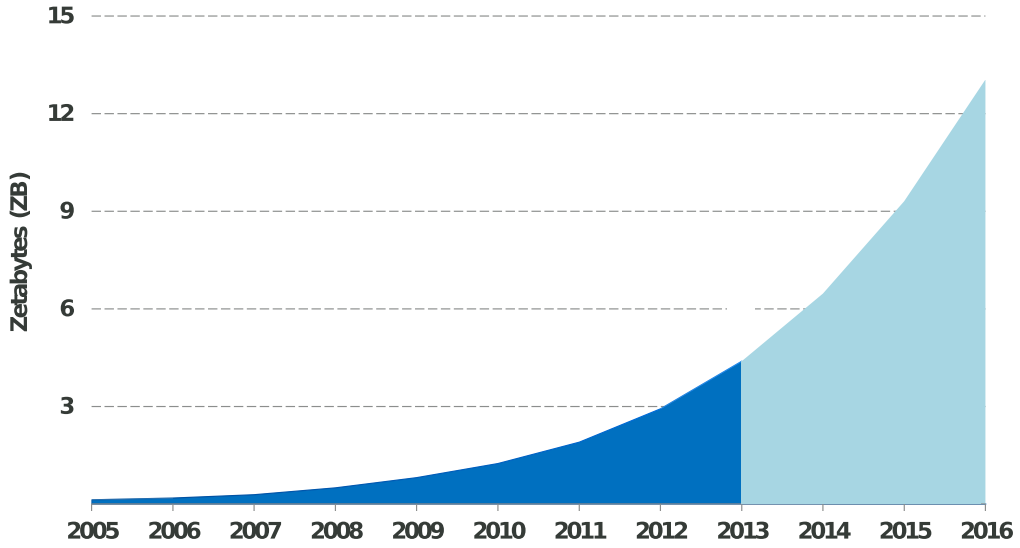


FIGURE 0.2: Volume des données numériques existantes dans l'univers. Figure tirée de "Internet Trends" [1]. Les 2/3 des données numériques existantes sont crées ou utilisées par des particuliers.

La datamasse. Pour donner un ordre de grandeur de l'avalanche de données numériques dont nous sommes témoins, chaque minute, 204 millions de courriers électroniques sont envoyés dans le monde, 2,5 millions de contenus sont partagés sur Facebook, Amazon vend pour 83 000 dollars d'articles, le moteur de recherche de Google est sollicité 4 millions de fois [14] (voir la Fig. 0.1)... Tous les six jours, 100 ans de vidéo sont mis en ligne sur Youtube [15] ! Le volume brut des données numériques mondiales augmente actuellement de 50% par an et a été estimé supérieur à 4 Zo (zettaoctets) en 2013 [1] (voir la Fig. 0.2), c'est-à-dire quatre mille milliards de gigaoctets.

En termes énergétiques, le coût en électricité du numérique revient aujourd'hui à 10% de la génération d'électricité mondiale. À l'échelle d'un utilisateur, regarder une heure de vidéo sur Internet par semaine, revient à la consommation électrique de deux réfrigérateurs, quand on prend en compte la consommation énergétique de tous les serveurs en jeu [5].

Il existe un grand nombre d'autres domaines où la taille des données explose. Pensons aux expériences du Large Hadron Collider (LHC), l'accélérateur de particules du CERN, qui a produit, en 2010, 13 petaoctets (13 millions de gigaoctets), ce qui remplirait une pile de CDs de 14 kilomètres de haut. Deux cent mille processeurs répartis dans 34 pays sont dédiés à l'analyse de ces données [39]. En biologie, l'apparition de séquenceurs de gènes de plus en plus rapides ont permis de créer des bases de données qui atteignent aussi de grandes tailles. L'institut européen de bioinformatique (EBI), au Royaume-Uni, qui fait partie du laboratoire de biologie moléculaire

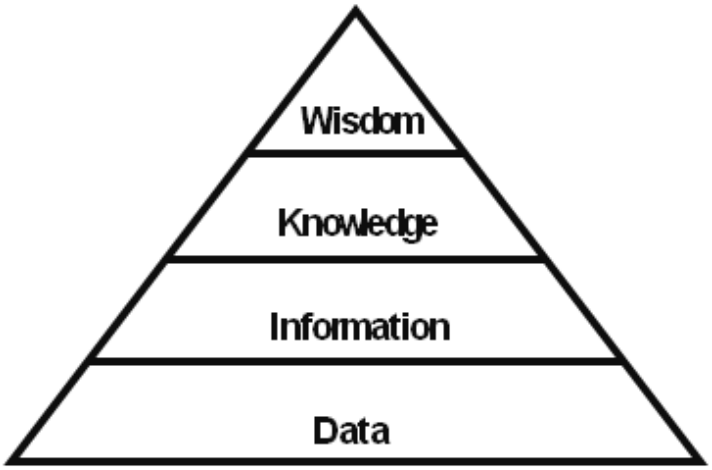


Figure 0.3: Diagramme Données-Information-Connaissance-Sagesse (Data Information Knowledge Wisdom – DIKW en anglais) publié sur Wikipédia [4]. Dans cette thèse, nous nous intéressons au passage du socle au premier étage de cette pyramide : comment passer des données brutes à de l'information ?

européen (EMBL), stockait, en 2013, 20 petaoctets de données séquencées sur des gènes, des protéines ou de petites molécules [144]. En astronomie, l'ordre de grandeur de la taille des données est similaire aujourd'hui. Citons l'exemple du SKA, le Square Kilometer Array, radiotélescope en cours de construction qui aura une surface de mesure de signaux radio d'environ un kilomètre carré. Ce radiotélescope, qui devrait être complètement opérationnel autour de 2025, génèrera 1 exaoctet (un milliard de gigaoctet) *par jour* [11] : l'augmentation du volume des données à analyser est loin d'être terminée !

La pyramide DIKW. Présentons tout d'abord le concept pyramidal « Données Information Connaissance Sagesse »(Data Information Knowledge Wisdom – DIKW en anglais) représenté sur la Fig. 0.3. Les données forment la base de cette pyramide et représentent la couche la plus volumineuse. Ces données sont aujourd'hui essentiellement numériques, c'est-à-dire une suite binaire de zéros et de uns, codant toutes sortes de mesures. Ces données, en tant que telles, n'ont pas de sens. Afin de les rendre intelligibles, il faut les ordonner, les agencer dans un sens qui permette d'en extraire de l'information : c'est la deuxième couche de cette pyramide. La troisième couche consiste en l'analyse de ces informations pour en tirer une forme de connaissance ; connaissance qui permettrait, dans l'idéal, d'atteindre une forme de sagesse, ultime étage de la pyramide. Nous nous intéressons ici aux deux premiers étages de cette pyramide et plus précisément au passage de l'un à l'autre.

L'extraction automatisée d'informations : un défi actuel. Les données qui forment le socle de la pyramide DIKW sont de plus en plus volumineuses, à tel point qu'en extraire l'information pertinente nécessite aujourd'hui des systèmes de traitement automatisés, parfois supervisés (c'est-à-dire qu'au moins un humain, quelque part, vérifie les traitements des algorithmes) et d'autres fois non-supervisés (on fait

FIGURE 0.4: Le réseau Facebook, publié sur le site de la compagnie [17].

assez confiance aux algorithmes pour ne plus vérifier ce qu'ils font). Des exemples simples à l'échelle de l'utilisateur sont les filtres anti-spam qui décident à notre place si un e-mail nous est vraiment adressé, ou les moteurs de recherche qui cherchent pour nous du contenu parmi plusieurs milliards de pages internet. En quelque sorte, devant un tel volume de données, nous sommes contraints à faire appel à la machine pour extraire l'information pertinente. Mais quels outils informatiques, quels algorithmes utiliser ? Ce grand défi a des implications sociétales majeures car nous sommes aujourd'hui dépendants de ces systèmes d'extraction, et nous leur faisons de plus en plus confiance : qui va parcourir la centaine de courriers indésirables qu'il reçoit par jour et vérifier si son filtre anti-spam fonctionne bien ? Qui prend le temps de vérifier que les résultats de la deuxième page de Google concordent avec le premier résultat proposé ? Une étude publiée par Chitika [46] montre que le premier résultat de Google est choisi plus du tiers des fois, et cette proportion chute sous les 1% pour les résultats de la deuxième page.

Données sous forme de réseaux. Nous nous intéressons ici à un type de grandes données assez particulier mais en pleine expansion : des données structurées sous forme de réseau, c'est-à-dire un ensemble d'entités plus ou moins reliées entre elles. Nous allons, comme l'indique le titre de cette thèse, développer des outils de traitement du signal destinés à extraire l'information automatiquement de données sous formes de réseaux. Pour illustrer la notion de réseau, nous pouvons penser à divers domaines où des réseaux existent. Le réseau Facebook, par exemple, est constitué de plus d'un milliard de profils et chacun des profils est relié à d'autres profils s'ils sont "amis" sur le réseau. Cela peut donner une représentation comme celle de la Fig. 0.4. L'étude de ce genre de réseaux permet par exemple de mieux comprendre à quelle vitesse une information peut se propager dans un réseau social et pourquoi.

Un autre exemple est le réseau de neurones du cerveau. La Fig. 0.5 montre une image de quelques neurones reliés entre eux sous forme de réseau. Nous avons environ

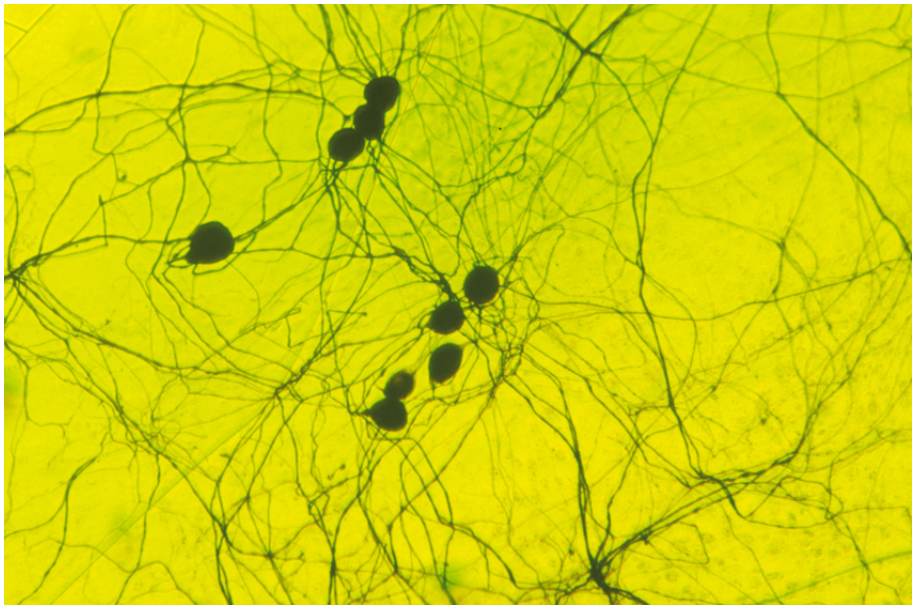


Figure 0.5: Image d'un réseau formé de quelques neurones, publiée sur le site de l'INSERM [3].

10^{11} neurones dans le cerveau et 10^{15} connexions entre neurones [3]. Notre réseau de neurones est un objet d'étude pharaonique qui permet l'étude de nombreuses questions, notamment le lien entre la topologie et la fonction de certaines parties du réseau.

Un autre exemple, du côté de l'informatique, est le réseau Internet. Chaque ordinateur, chaque routeur du réseau internet sont reliés entre eux par des câbles physiques. Cette interconnexion des ordinateurs du monde entier forme un grand réseau que l'on peut représenter, comme le fait par exemple le projet OPTE [16], par l'image de la Fig. 0.6. Sur ce genre de réseaux, les questions de sécurité sont souvent abordées : à quel point ce réseau est-il résistent à des cyber-attaques aléatoires, ou à des cyber-attaques ciblées ?

Un autre type de réseau sont les réseaux de transports. À titre d'exemple, évoquons un réseau de transport spécifique : celui formé par les trajets de Vélo'v, les vélos partagés de Lyon [34]. On peut en faire une représentation sur un fond de carte de la ville de Lyon sur la Fig. 0.7. L'étude de ces réseaux de transport permet par exemple de savoir où le trafic se concentre et quels petits changements sur le réseau pouvons-nous effectuer pour le rendre plus fluide.

Un autre exemple, qui montre si c'est encore nécessaire que ce type de données sous forme de réseaux se retrouve dans beaucoup de domaines disciplinaires très différents les uns des autres, est le réseau de collaboration entre acteurs de cinéma, dans lequel deux acteurs sont connectés entre eux s'ils ont déjà joué ensemble dans un même film. On trace sur la Fig. 0.8 une représentation d'une partie de ce réseau. Le genre de questions intéressantes sur ces réseaux de collaboration peut être des questions de centralité par exemple : quels acteurs sont centraux dans telle ou telle branche du cinéma ? Comment définit-on la centralité sur ce genre de réseaux ?

Beaucoup d'autres réseaux sont couramment étudiés, chacun avec son lot de

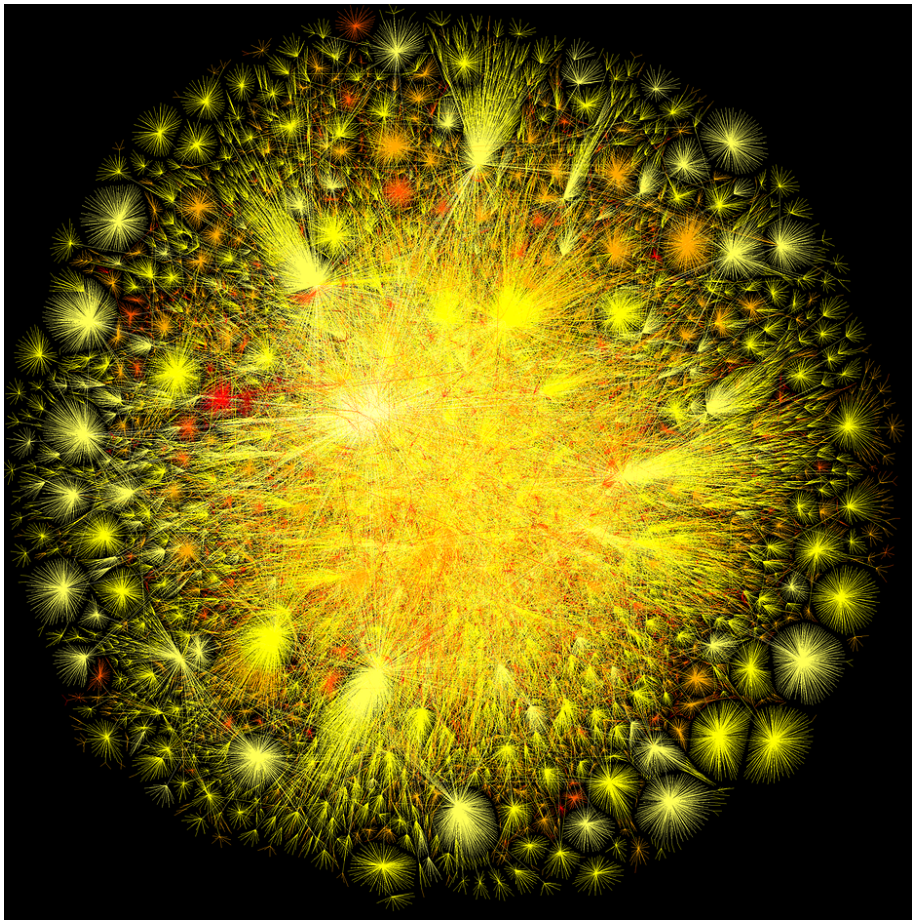


FIGURE 0.6: Une image du réseau internet mondial, selon le projet OPTE [16].

questions spécifiques à répondre : les réseaux de chaîne d'alimentation parmi les animaux, les réseaux d'associations de mots où les mots sont reliés entre eux en fonction de la fréquence à laquelle l'un est associé à l'autre dans nos pensées, les réseaux de protéines, où les protéines interagissent entre elles de manière complexe, les réseaux de flux financiers ... Il serait bien trop long de citer ici tous les types de réseaux que l'on peut rencontrer, mais retenons simplement qu'ils proviennent de domaines très variés.

Réseaux modélisés par des graphes. Nous allons modéliser ces réseaux à l'aide de graphes, objets mathématiques bien identifiés, constitués de nœuds (qui feront office, selon l'application, de profils Facebook, de régions neuronales, de routeurs, ...), reliés ensemble par des liens. Une fois que nous nous sommes décidés sur le graphe représentant le réseau, nous avons déjà parcouru la moitié du deuxième étage de la pyramide DIKW. En effet, cette modélisation de données sous forme de graphe, est, comme toute modélisation, approximative, et nous devons nécessairement faire des choix qui orientent l'analyse. Pour parcourir la deuxième moitié de l'étage, reste à extraire des informations pertinentes de ce graphe, et c'est ce sur ce quoi se concentre cette thèse : une fois un graphe donné, comment en extraire de l'information ?

Du point de vue du traitement du signal, étudier un graphe présente des difficul-

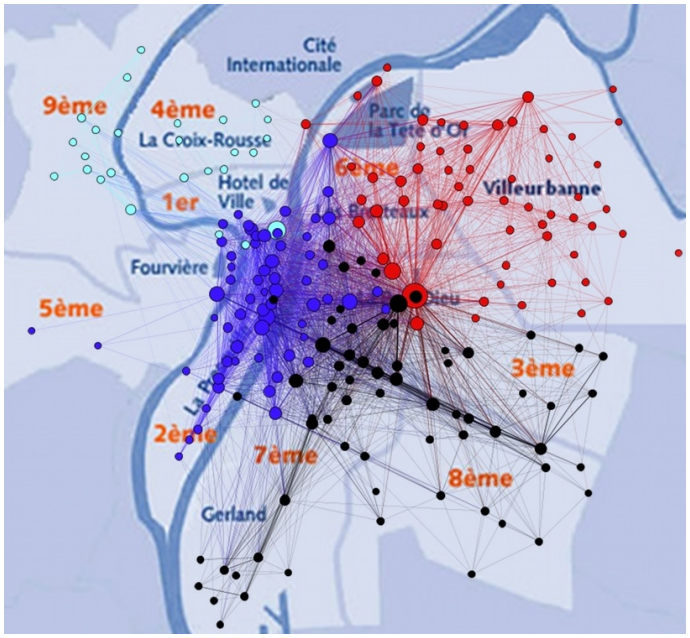


Figure 0.7: Le réseau de vélos partagés Vélo'v sur le fond de carte de la ville de Lyon. Chaque cercle représente une station, plus il est grand, plus la station associée est active. Les liens entre chaque station correspondent à des flux de vélos entre les stations. Cette image est extraite de la publication [34].

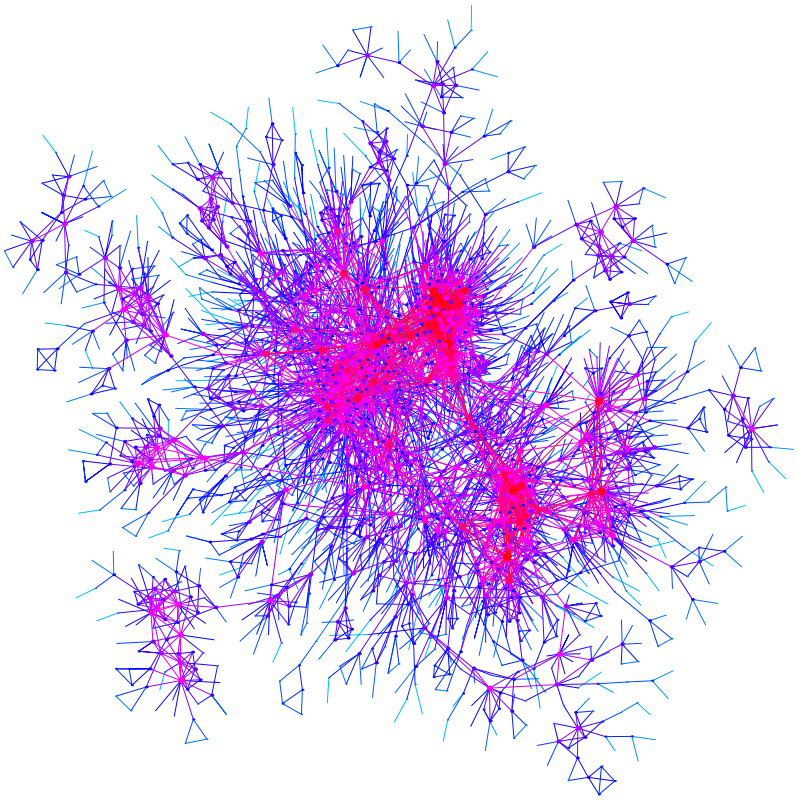


Figure 0.8: Représentation d'une partie du réseau de collaboration entre acteurs de cinéma. Figure tirée de [49].

tés particulières. Ce n'est ni un signal à une dimension comme une courbe de température en fonction du temps, ni un signal à deux dimensions comme une image, mais c'est un objet potentiellement très irrégulier. Dans une courbe temporelle, chaque point de mesure a deux voisins : le point d'avant et le point d'après. Dans une image, chaque pixel a quatre voisins. Dans un graphe, chaque point a un nombre arbitraire de voisins : certains ont 3 amis sur Facebook, d'autres plusieurs milliers. Les outils classiques de traitement du signal comme la transformée de Fourier, le débruitage, la convolution ou la transformée en ondelettes n'ont pas été créés pour ce genre de topologie irrégulière, et il faut les adapter à ces nouvelles données.

Guide de lecture de cette thèse. Ce manuscrit est une présentation détaillée de certains travaux déjà publiés (voir la liste des publications page 187), et d'autres travaux en cours. Dans le premier chapitre de cette thèse, nous récapitulerons le contexte du traitement du signal sur graphe. Nous définirons ce qu'est un signal sur un graphe et évoquerons les outils qui ont été proposés ces dernières années pour pouvoir traiter ces signaux.

Les deuxième et troisième chapitre représentent les deux principales contributions de cette thèse. Dans le deuxième chapitre, nous proposons une nouvelle méthode de détection multiéchelle de communautés dans des graphes, basée sur une définition récente d'ondelettes sur graphe. Nous utiliserons les nouveaux outils théoriques du traitement du signal sur graphe dans le but d'apporter un nouvel éclairage sur le problème phare de l'analyse de grands graphes : la détection automatique de communautés. Plus précisément, nous verrons que les ondelettes peuvent très bien s'adapter à ce problème et qu'elles permettent même de définir un cadre commun à de nombreuses méthodes de détection multiéchelle existantes.

Le troisième chapitre décrit une méthode statistique de rééchantillonnage de groupes de nœuds dans des graphes, à des fins d'estimation de certaines caractéristiques. En effet, un graphe de terrain peut être souvent vu comme étant une seule réalisation d'un processus aléatoire. Dans ces circonstances, nous faisons appel à un rééchantillonnage contrôlé afin de remonter à des intervalles de confiance pour certaines caractéristiques mesurées et de développer un test statistique dans le but de détecter si un groupe de nœuds a un comportement anormal.

Pour faciliter la navigation et en plus d'une table des matières détaillée à la fin du manuscrit, un sommaire est associé à un court texte de présentation en début de chaque chapitre.

Chapitre 1

Généralités sur les graphes et le traitement du signal sur graphes

« Pure mathematics is, in its way, the poetry of logical ideas. »

– A. Einstein, *Obituary for Emmy Noether*

Dans ce chapitre, nous allons rappeler quelques généralités sur les graphes, et plus précisément sur les graphes complexes, avant d'évoquer les bases théoriques du domaine émergent du traitement du signal sur graphes. La première partie de ce chapitre paraîtra triviale au lecteur habitué aux notions inhérentes aux graphes, mais sera j'espère utile au lecteur intéressé par l'aspect "traitement du signal" de cette thèse et pas forcément au fait des définitions du domaine. Inversement, les deux dernières parties s'adressent de préférence au lecteur qui a besoin d'être un peu guidé dans le domaine du "traitement du signal". En tout état de cause, le traitement du signal sur graphe est un domaine suffisamment émergent pour que chaque lecteur trouve dans ce premier chapitre quelques éclaircissements.

Sommaire

1 Généralités sur les graphes **12**
- 1.1 Qu'est-ce qu'un graphe ? 12
- 1.2 Matrice d'adjacence 13
- 1.3 Les graphes pondérés 13
- 1.4 Quelques définitions 13
- 1.5 Matrice laplacienne 15
- 1.6 Modèles de graphes aléatoires 15

2 Notions de base du traitement du signal sur graphes **16**
- 2.1 Qu'est-ce qu'un signal sur graphe ? 16
- 2.2 Le traitement du signal discret : un cas particulier 16
- 2.3 Quelques résultats du traitement du signal discret classique 17
- 2.4 Le traitement du signal sur graphes selon Sandryhaila et Moura 20
- 2.5 Le traitement du signal sur graphes selon Vandergheynst et al. 22
- 2.6 Autres analogies possibles 24
- 2.7 Résultats du traitement du signal sur graphe 26

3 La transformée en ondelettes sur graphes **31**
- 3.1 La famille d'ondelettes classiques 31
- 3.2 La famille d'ondelettes sur graphes 32
- 3.3 La transformée en ondelettes sur graphe 33
- 3.4 Algorithme rapide de transformée en ondelettes sur graphe 34
- 3.5 Le noyau de filtre d'ondelettes 35
- 3.6 Quelques illustrations 36

1 Généralités sur les graphes

Cette première partie reprend quelques définitions classiques de la théorie des graphes, que l'on peut retrouver par exemple dans les livres de Bondy et Murty [33] ou de Clark et Holton [50].

1.1 Qu'est-ce qu'un graphe ?

Dans sa définition la plus simple, un graphe est un ensemble de nœuds reliés entre eux par des liens. Formellement, nous définissons le graphe $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ par son ensemble de nœuds $\mathcal{V}$ et son ensemble de liens $\mathcal{E}$ connectant les nœuds entre eux. En principe, deux nœuds peuvent être connectés par plusieurs liens : on dit alors que ces liens sont parallèles. Aussi, un lien peut connecter un nœud à lui-même : on dit que ce lien est une boucle. Dans la suite, nous considérons le plus souvent des graphes dits simples : c'est-à-dire qui ne contiennent ni liens parallèles ni boucles.

Un lien est orienté : le lien (ij) par exemple connecte le nœud i au nœud j, et est schématiquement représenté par une flèche allant de i à j :

Dans le cas où le lien (ji) existe aussi, au lieu de dessiner une double flèche, on dessine simplement :

Si pour tout lien (ij) du graphe, (ji) existe aussi, on dit que le graphe est non-orienté. Sinon, il est dit orienté.

1.2 Matrice d'adjacence

Dans cette thèse, nous prendrons le plus souvent le point de vue d'une représentation algébrique de $\mathcal{G}$, nécessaire au développement d'outils de traitement du signal sur graphe ; au détriment des approches en mathématiques discrètes, chères à une partie de la communauté mathématique de la théorie des graphes. Nous attachons donc beaucoup d'importance à la matrice d'adjacence A d'un graphe qui code les connections entre les noeuds $\mathcal{V}$. Si on note N le nombre de nœuds du graphe, A est de taille $N \times N$ et :

$$\begin{aligned} A_{ij} &= 1 \text{ si le lien } (ij) \text{ existe} \\ A_{ij} &= 0 \text{ sinon.} \end{aligned} \tag{1.1}$$

La matrice d'adjacence d'un graphe non-orienté (resp. orienté) est nécessairement symmétrique (resp. non symmétrique), comme le montre le haut (resp. bas) de la Fig. 1.1.

1.3 Les graphes pondérés

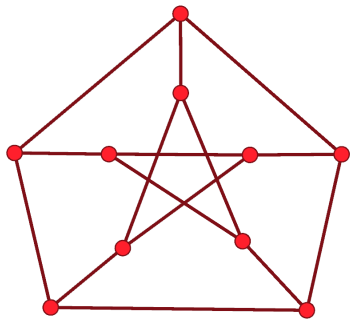
Les liens des graphes que nous venons d'évoquer ont deux états : soit ils existent, soit ils n'existent pas. On dit que ces graphes sont binaires (leurs matrices d'adjacence sont composées uniquement de 0 et de 1).

Un graphe pondéré, en revanche, est un graphe dont chaque lien est associé à un poids qui code l'intensité de l'interaction entre les deux nœuds reliés. Nous notons W la matrice d'adjacence pondérée d'un tel graphe, où $W_{ij} \in \mathbb{R}$ est le poids du lien connectant le nœud i au nœud j. Pour un graphe binaire, $W = A$. Plus le poids W_{ij} est élevé plus le lien entre i et j est "fort".

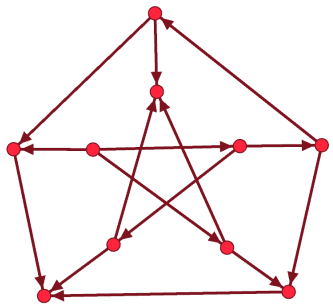
Dans ce manuscrit, sauf si expressément indiqué, nous considérons uniquement des graphes simples, pondérés et non-orientés.

1.4 Quelques définitions

Un nœud i est dit *voisin* d'un nœud j si ils sont connectés par un lien, i.e. si $W_{ij} \neq 0$. On note $i \sim j$ si i et j sont voisins.



$$\longleftrightarrow \qquad A = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \end{pmatrix}$$



$$\longleftrightarrow \qquad A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Figure 1.1: Équivalence graphe – matrice d'adjacence. Ici pour le graphe orienté (en bas) ou non (en haut) dit de Peterson.

Le *degré* d_i d'un nœud i est son nombre de voisins. Formellement, le degré s'écrit :

$$d_i = \sum_{j=1}^{N} \mathbb{I}(W_{ij} \neq 0), \tag{1.2}$$

où $\mathbb{I}(W_{ij} \neq 0) = 1$ si $W_{ij} \neq 0$ est vraie, 0 sinon. On note d le vecteur des degrés (de taille N), et D la matrice diagonale où $\forall i \quad D_{ii} = d_i$.

La *force* s_i d'un nœud i est la somme des poids des liens connectés à i. Formellement, la force d'un nœud s'écrit :

$$s_i = \sum_{j=1}^{N} W_{ij}. \tag{1.3}$$

On note s le vecteur des forces (de taille N) et S la matrice diagonale où $\forall i \quad S_{ii} = s_i$. Dans le cas où le graphe est binaire, la force et le degré d'un nœud sont la même chose.

Un graphe est dit *régulier* si tous ses nœuds ont même degré. Par exemple, le graphe du haut de la Fig 1.1 est régulier : tous ses nœuds ont degré 3.

Un *chemin* entre deux nœuds i et j est une suite de liens consécutifs reliant i à j.

Un *cycle* est un chemin ne passant pas deux fois par un même lien et dont les deux extrémités sont identiques (départ et arrivée du chemin sur le même nœud).

Un graphe est dit *connecté* si il existe un chemin entre toute paire de nœuds.

Un *arbre* est un graphe connecté sans cycle.

Un graphe est dit *bipartite* si son ensemble de sommets $\mathcal{V}$ peut-être séparé en deux sous-ensembles $\mathcal{U}_1$ et $\mathcal{U}_2$ tel que chaque lien dans $\mathcal{E}$ ait une extrémité dans $\mathcal{U}_1$ et l'autre dans $\mathcal{U}_2$.

Soit $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ un graphe. Son *graphe adjoint* noté $L(\mathcal{G})$ et communément appelé *line graph* en anglais, est le graphe d'adjacence des liens de $\mathcal{G}$. Les nœuds de $L(\mathcal{G})$ sont les liens $\mathcal{E}$ de $\mathcal{G}$, et deux nœuds dans $L(\mathcal{G})$ sont connectés si leurs liens associés dans $\mathcal{G}$ sont adjacents. Son usage est généralement réservé aux graphes binaires, mais des extensions aux graphes pondérés ont été proposées par certains auteurs [75].

1.5 Matrice laplacienne

Dans ce manuscrit, nous évoquerons trois matrices laplaciennes différentes :

- la matrice laplacienne, aussi appelée laplacien combinatoire ou simplement *laplacien*, d'un graphe, définie à partir de sa matrice d'adjacence pondérée W et de sa matrice de force S :

$$L = S - W. \tag{1.4}$$

  Dans le cas d'un graphe binaire, on retrouve la formule plus usuelle : $L = D - A$.
- la matrice laplacienne normalisée, ou *laplacien normalisé*, que nous noterons $\mathcal{L}$ et qui s'écrit :

$$\mathcal{L} = S^{-\frac{1}{2}} L S^{-\frac{1}{2}} = I_N - S^{-\frac{1}{2}} W S^{-\frac{1}{2}}, \tag{1.5}$$

  où I_N est la matrice identité de taille $N \times N$. Dans le cas d'un graphe binaire, $\mathcal{L}$ s'écrit : $\mathcal{L} = I_N - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$.
- le *laplacien de marche aléatoire*, que nous noterons $\mathcal{L}_{rw}$ et qui s'écrit :

$$\mathcal{L}_{rw} = S^{-1} L = I_N - S^{-1} W. \tag{1.6}$$

1.6 Modèles de graphes aléatoires

Il existe des modèles de graphes aléatoires de toutes sortes : le modèle de petit monde de Watts-Strogatz [223], le modèle d'attachement préférentiel de Barabási-Albert [24], ... Nous présentons ici deux modèles de graphes binaires et non-dirigés que nous allons citer et utiliser dans la suite de cette thèse : le modèle d'Erdös-Rényi

et le modèle de Chung-Lu.

Le modèle d'Erdös-Rényi a été introduit par Erdös et Rényi en 1959 [73]. Ce modèle est paramétré par N le nombre de nœuds et p la probabilité d'existence d'un lien entre toute paire de nœuds. Pour le construire, générer N nœuds, puis parcourir une à une toutes les paires de nœuds et décider pour chacune d'entre elles si ou non un lien existe entre les deux nœuds (avec une probabilité p). À titre d'exemple, une probabilité de $p = 1$ créé nécessairement le graphe complet à N nœuds. Les graphes générés par ce modèle n'ont aucune sorte de corrélation.

Le modèle de Chung-Lu est aussi appelé modèle de configuration et est un modèle de graphe aléatoire dont les paramètres sont le nombre de nœuds N et une collection de degrés $(k_1, k_2, \ldots, k_N)$ qui sera l'espérance de la distribution de degrés des graphes générés sous ce modèle. Pour créer un graphe de Chung-Lu, générer N nœuds. Puis, pour chaque paire de nœuds i et j, le lien (ij) est créé avec une probabilité $\min(1, \frac{k_i k_j}{2k_{tot}})$ où $k_{tot} = \sum_i k_i$ est l'espérance du nombre total de liens. Chung, Lu, Molloy et Reed ont particulièrement travaillé sur ce modèle [152, 49].

2 Notions de base du traitement du signal sur graphes

2.1 Qu'est-ce qu'un signal sur graphe ?

Étant donné un graphe $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, on définit un signal sur graphe f sur les nœuds du graphe :

$$\begin{aligned} f : \quad & \mathcal{V} \to \mathbb{R} \\ & i \to f(i) \end{aligned} \tag{1.7}$$

On peut le représenter directement sur l'image du graphe en coloriant chaque nœud i du graphe proportionellement à $f(i)$. Par exemple, un signal aléatoire sur un graphe quelconque (ici, un graphe d'Erdös-Rényi avec $N = 15$ nœuds et $p = 0.2$) peut s'illustrer à l'aide de couleurs comme sur la Fig. 1.2.

Une autre alternative de signal sur graphe est de définir f sur les liens du graphe. Dans ce cas, on peut se ramener au cas précédent en considérant non pas le graphe $\mathcal{G}$ mais son graphe adjoint $L(\mathcal{G})$: en effet, un signal défini sur les liens de $\mathcal{G}$ est défini sur les nœuds de $L(\mathcal{G})$. Dans la suite, nous ne considérerons que les signaux définis sur les nœuds d'un graphe.

2.2 Le traitement du signal discret : un cas particulier du traitement du signal sur graphe

Le traitement du signal discret, que nous appellerons cas "classique", peut être vu comme un cas particulier du traitement du signal sur graphes : en effet, nous pouvons considérer qu'un signal classique f de taille N n'est autre qu'un signal défini sur le graphe boucle de taille N (voir Fig. 1.3). Ici nous représentons le graphe boucle non-orienté. Certains auteurs préfèrent le considérer orienté [183].

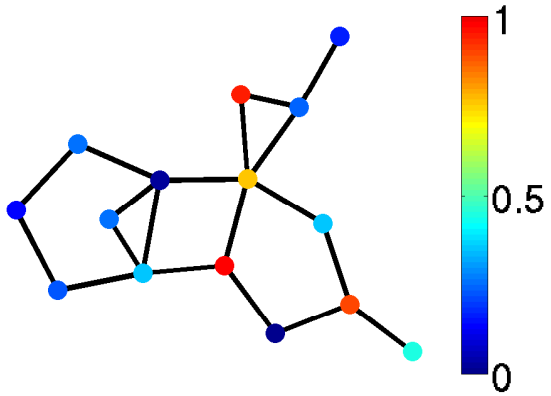


FIGURE 1.2: Exemple de signal sur un graphe d'Erdös-Rényi avec $N = 15$ nœuds et $p = 0.2$. Le signal est représenté en couleur sur chaque nœud.

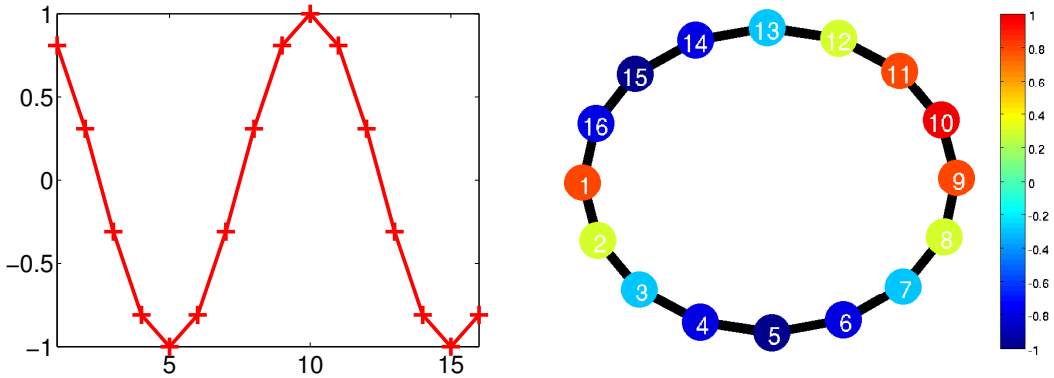


FIGURE 1.3: Deux représentations d'un même signal. À gauche, la représentation linéaire classique d'une sinusoïde discrète. À droite, la même sinusoïde représentée en couleurs sur le graphe boucle non-orienté.

L'objectif des théoriciens du traitement du signal sur graphe est de constituer une théorie cohérente qui, quand réduite au graphe boucle, redonne les résultats bien connus du cas classique. Cette généralisation est loin d'être unique et nous allons dans la suite donner les grandes lignes de deux théories qui sont à ce jour les plus abouties : celle de Sandryhaila et Moura [183] et celle de Vandergheynst et collaborateurs [193]. Avant cela, nous allons récapituler les propriétés du traitement du signal classique que nous voulons retrouver.

2.3 Quelques résultats du traitement du signal discret classique

Les résultats de cette partie peuvent être retrouvés par exemple dans [198]. Nous considérons dans la suite un signal discret classique f de taille N, dont le premier terme est indicé par 1 (et non par 0 comme c'est souvent le cas).

2.3.1 La transformée de Fourier discrète

La transformée de Fourier discrète de f s'écrit :

$$\forall k \in [1, N] \qquad \hat{f}(k) = \frac{1}{\sqrt{N}} < e^{\frac{2i\pi(k-1)(\cdot-1)}{N}}, f(\cdot) >= \frac{1}{\sqrt{N}} \sum_{j=1}^{j=N} f(j) e^{-\frac{2i\pi(k-1)(j-1)}{N}}. \tag{1.8}$$

Elle peut également s'écrire à l'aide de la matrice de Fourier F de taille $N \times N$ définie par $F(j,k) = \frac{1}{\sqrt{N}} \omega^{(j-1)(k-1)}$, où ω est la première racine N-ième de l'unité : $\omega = exp(\frac{-2i\pi}{N})$. A un facteur de normalisation $\frac{1}{\sqrt{N}}$ près, F est la matrice de Vandermonde associée aux racines N-ième de l'unité :

$$F = \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \cdots & \omega^{N-1} \\ 1 & \omega^2 & \omega^4 & \omega^6 & \cdots & \omega^{2(N-1)} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \cdots & \omega^{3(N-1)} \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ 1 & \omega^{N-1} & \omega^{2(N-1)} & \omega^{3(N-1)} & \cdots & \omega^{(N-1)(N-1)} \end{bmatrix}. \tag{1.9}$$

Et on réécrit donc l'équation 1.8 sous forme vectorielle :

$$\hat{f} = Ff. \tag{1.10}$$

F étant une matrice de Vandermonde associée aux racines de l'unité qui sont toutes distinctes, F est inversible. La transformée de Fourier inverse s'écrit :

$$f = F^{-1}\hat{f}. \tag{1.11}$$

De plus, l'identité de Parseval est vérifiée :

$$< f, f >=< \hat{f}, \hat{f} >, \tag{1.12}$$

où $<,>$ désigne le produit scalaire usuel :

$$\forall (x, y) \in \mathbb{C}^N \times \mathbb{C}^N \qquad < x, y >= \sum_{i=1}^{N} x^*(i) y(i). \tag{1.13}$$

2.3.2 L'opérateur de convolution circulaire

L'opérateur de convolution circulaire permet de calculer la convolution entre deux vecteurs finis de taille N. Rappelons que l'opérateur de convolution entre deux vecteurs infinis f et g définis sur $\mathbb{Z}$ s'écrit :

$$\forall i \in \mathbb{Z} \qquad (f * g)(i) = \sum_{j=-\infty}^{+\infty} f(j) g(i-j). \tag{1.14}$$

Soit f un vecteur de taille N. Notons $\tilde{f}$ son prolongement sur $\mathbb{Z}$ par périodisation, c'est-à-dire que :

$$\forall i \in [1, N] \quad \forall j \in \mathbb{Z} \qquad \tilde{f}(i + jN) = f(i). \tag{1.15}$$

L'opérateur de convolution ciculaire, que nous noterons aussi $*$ par souci de simplicité, entre deux vecteurs f et g de taille N s'écrit :

$$\forall i \in [1, N] \qquad (f * g)(i) = \sum_{j=1}^{N} \tilde{f}(j)\tilde{g}(i - j + 1), \tag{1.16}$$

De plus, étant donné la définition de la transformée de Fourier, on retrouve bien la propriété classique qu'une multiplication dans l'espace de Fourier correspond à une convolution dans l'espace direct :

$$\forall k \in [1, N] \qquad \widehat{(f * g)}(k) = \sqrt{N}\hat{f}(k)\hat{g}(k). \tag{1.17}$$

2.3.3 La translation

La translation du signal f par $i_0 \in [1, N]$ peut s'écrire à l'aide de l'opérateur de convolution circulaire :

$$\mathcal{T}_{i_0} f = f * \delta_{i_0} = \begin{bmatrix} \tilde{f}(2 - i_0) \\ \tilde{f}(3 - i_0) \\ \vdots \\ \tilde{f}(N + 1 - i_0) \end{bmatrix}. \tag{1.18}$$

où δ_{i_0} est le Dirac centré en i_0 c'est-à-dire le vecteur de taille N tel que :

$$\begin{aligned} \forall i \in [1, N] \qquad \delta_{i_0}(i) &= 1 \quad \text{si } i = i_0, \\ &= 0 \quad \text{sinon.} \end{aligned} \tag{1.19}$$

2.3.4 La modulation

La modulation d'un signal f par $k_0 \in [1, N]$ correspond à le multiplier par une exponentielle complexe :

$$\forall j \in [1, N] \qquad \mathcal{M}_{k_0} f(j) = f(j) e^{\frac{-2i\pi(k_0-1)(j-1)}{N}} = \sqrt{N} f(j) F(j, k_0). \tag{1.20}$$

2.3.5 Le filtrage

Le filtrage de f par un filtre linéaire $h \in \mathbb{R}^N$ peut être défini comme une convolution dans l'espace direct :

$$f^h = h * f, \tag{1.21}$$

ou comme une multiplication dans l'espace de Fourier :

$$\forall k \in [1, N] \qquad \hat{f}^h(k) = \sqrt{N}\hat{h}(k)\hat{f}(k). \tag{1.22}$$

Pour écrire l'équation 1.22 en vectoriel, on définit la matrice diagonale (dans l'espace de Fourier) :

$$\hat{H} = \begin{bmatrix} \hat{h}(1) & 0 & 0 & \cdots & 0 \\ 0 & \hat{h}(2) & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 & \hat{h}(N) \end{bmatrix}. \tag{1.23}$$

$\hat{H}$ est l'expression du filtre h dans l'espace de Fourier. On réécrit alors l'équation 1.22 :

$$\hat{f}^h = \sqrt{N}\hat{H}\hat{f}, \tag{1.24}$$

ce qui donne, si on revient dans l'espace direct :

$$f^h = \sqrt{N}F^{-1}\hat{H}\hat{f} = \sqrt{N}F^{-1}\hat{H}Ff. \tag{1.25}$$

2.3.6 L'opérateur de retard

L'opérateur de retard, noté $\mathcal{S}$, translate un signal f de un :

$$\mathcal{S}f = f * \delta_1, \tag{1.26}$$

et a pour expression matricielle :

$$\mathcal{S} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & 0 & 1 \\ 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 & 0 \end{bmatrix}. \tag{1.27}$$

C'est-à-dire :

$$f = \begin{bmatrix} f(1) \\ f(2) \\ f(3) \\ \vdots \\ f(N) \end{bmatrix} \quad \longrightarrow \quad \mathcal{S}f = \begin{bmatrix} f(N) \\ f(1) \\ f(2) \\ \vdots \\ f(N-1) \end{bmatrix}. \tag{1.28}$$

On montre aisément que $\forall i \in [1, N] \quad \mathcal{S}^i f = f * \delta_i$, ce qui nous permet de réécrire l'équation 1.21 en :

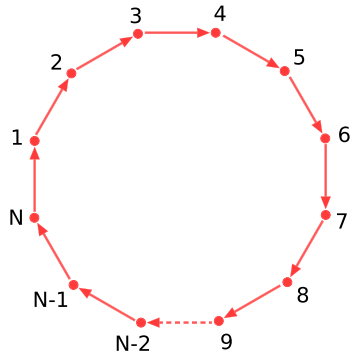
$$f^h = \sum_{i=1}^{N} h(i)\mathcal{S}^i f. \tag{1.29}$$

2.4 Le traitement du signal sur graphes selon Sandryhaila et Moura

Sandryhaila et Moura décrivent dans [183, 184, 185, 186], une possibilité de généralisation du traitement du signal discret classique au traitement du signal sur graphe. Pour cela, ils se reposent sur l'analogie fondamentale suivante.

2.4.1 Analogie fondamentale de Sandryhaila et Moura

Pour les auteurs, le graphe qui correspond au cas classique est le graphe boucle orienté :



de matrice d'adjacence binaire et non-symmétrique :

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & 0 & 0 & \cdots & 0 & 0 \end{bmatrix}. \tag{1.30}$$

On observe que pour un signal f défini sur les nœuds du graphe on a :

$$A^\top f = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & 0 & 1 \\ 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 & 0 \end{bmatrix} f = \begin{bmatrix} f(N) \\ f(1) \\ f(2) \\ \vdots \\ f(N-1) \end{bmatrix} = \mathcal{S} f. \tag{1.31}$$

Par analogie, pour tout graphe $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, orienté ou non, de matrice d'adjacence pondérée W, on définit l'opérateur de retard sur graphe : $\mathcal{S} = W^\top$.

2.4.2 Filtrer un signal sur graphe

Soit $h \in \mathbb{R}^N$ un filtre et f un signal défini sur les nœuds du graphe. On a, d'après l'équation 1.29, et par analogie :

$$f^h = \sum_{i=1}^{N} h(i) \left(W^\top\right)^i f. \tag{1.32}$$

2.4.3 Transformée de Fourier sur graphe

En remarquant, dans le cas classique, que $\mathcal{S}$ est une matrice circulante , elle est forcément diagonalisable dans la base de Fourier.
L'analogie des auteurs est d'avancer : pour tout graphe $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, orienté ou non, de matrice d'adjacence pondérée W, la base de Fourier est obtenue en calculant les vecteurs propres généralisés de la décomposition de Jordan de $W^\top$. En effet, dans le

cas d'un graphe quelconque, la matrice d'adjacence n'est pas forcément circulante et de manière générale pas forcément diagonalisable. Si bien que cette définition de la base de Fourier d'un graphe doit faire appel à la décomposition de Jordan.

Les valeurs propres généralisées obtenues n'ont par contre pas forcément le sens fréquentiel attendu, et afin d'obtenir l'équivalent de la fréquence de chaque mode de Fourier, les auteurs passent par un calcul *ad hoc* de variation totale [184].

2.4.4 En résumé

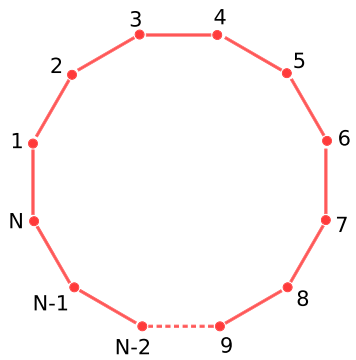
L'analogie est de proposer $W^\top$ comme équivalent de la matrice de retard pour un graphe. De la même manière qu'une matrice circulante classique est un polynôme en $\mathcal{S}$, une matrice circulante sur graphe est donc définie comme étant un polynôme en $W^\top$, et la matrice de Fourier sur graphe est définie comme étant la matrice qui diagonalise (ou qui Jordannise) toute matrice circulante sur graphe, donc qui diagonalise (ou qui Jordannise) $W^\top$.

2.5 Le traitement du signal sur graphes selon Vandergheynst et collaborateurs

Un point de vue que nous allons préférer dans ce manucrit est celui adopté pas Vandergheynst et collaborateurs [96, 193, 195] qui se base sur la base de Fourier obtenue en diagonalisant le laplacien du graphe.

2.5.1 Analogie fondamentale adoptée par Vandergheynst et collaborateurs

Dans cette analogie, le graphe associé au traitement du signal discret classique est le graphe boucle non orienté :



de matrice d'adjacence binaire et symmétrique :

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & 0 & 0 & \cdots & 1 & 0 \end{bmatrix}, \tag{1.33}$$

et de matrice laplacienne symétrique :

$$L = D - A = \begin{bmatrix} 2 & -1 & 0 & 0 & \cdots & 0 & -1 \\ -1 & 2 & -1 & 0 & \cdots & 0 & 0 \\ 0 & -1 & 2 & -1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 2 & -1 \\ -1 & 0 & 0 & 0 & \cdots & -1 & 2 \end{bmatrix}. \tag{1.34}$$

L étant circulante, elle est donc diagonalisable dans la base de Fourier classique F de l'équation 1.9.

Par analogie, pour tout graphe $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ non-orienté de matrice d'adjacence W, la base de Fourier de ce graphe est constituée des vecteurs propres du laplacien du graphe $L = S - W$. On note $\boldsymbol{\chi} = (\boldsymbol{\chi}_1 | \boldsymbol{\chi}_2 | \cdots | \boldsymbol{\chi}_N)$ cette matrice de Fourier du graphe, chaque colonne $\boldsymbol{\chi}_i$ étant vecteur propre du laplacien de valeur propre λ_i :

$$\forall i \in [1, N] \qquad L\boldsymbol{\chi}_i = \lambda_i \boldsymbol{\chi}_i. \tag{1.35}$$

L est forcément diagonalisable car nous nous restreignons ici aux graphes non-orientés, et la symmétrie du laplacien assure :

- sa diagonalisation dans $\mathbb{R}$: $\forall i \in [1, N] \quad \lambda_i \in \mathbb{R}$ et $\boldsymbol{\chi} \in \mathcal{M}_N(\mathbb{R})$.
- $\boldsymbol{\chi}$ est orthornomale.

2.5.2 Quelques précisions importantes

En classique, le graphe associé est le graphe boucle non-orienté, le laplacien celui de l'équation 1.34, et la base de Fourier est la matrice F de l'équation 1.9. On remarque en effet que L est diagonalisable dans F :

$$LF = \begin{bmatrix} \alpha_1 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & \alpha_2 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \alpha_3 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \alpha_{N-1} & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & \alpha_N \end{bmatrix} F, \tag{1.36}$$

avec :

$$\begin{aligned} \forall j \in [1, N] \qquad \alpha_j &= 2 - \omega^{j-1} - \omega^{(j-1)(N-1)} \\ &= 2 - \omega^{j-1} - \omega^{1-j} \\ &= 2\left(1 - \Re(\omega^{j-1})\right) \\ &= 2\left(1 - \cos\left(\frac{2\pi(j-1)}{N}\right)\right). \end{aligned} \tag{1.37}$$

De plus, on remarque que :

$$\forall j \in [2, N] \qquad \alpha_j = \alpha_{N+2-j}. \tag{1.38}$$

On note les valeurs propres du laplacien du cas classique α pour les différencier du cas d'un graphe quelconque que nous notons λ.
Si N est impair, il n'existe qu'un seul sous-espace propre (sous-espace vectoriel associé à une valeur propre) de dimension 1, c'est celui associé à $\alpha_1 = 0$ et au vecteur propre réel constant égal à $\frac{1}{\sqrt{N}}$ (première colonne de F).
Si N est pair, il n'existe que deux sous-espaces propres de dimension 1 : en plus de celui associé à $\alpha_1 = 0$, il y a celui associé à $\alpha_{\frac{N}{2}+1} = 4$ et au vecteur propre réel qui alterne $\frac{(-1)^{l-1}}{\sqrt{N}}$ et qui correspond à la $(\frac{N}{2}+1)^{\text{ème}}$ colonne de F.
Toutes les autres valeurs propres sont de multiplicité 2 : les sous-espaces propres associés sont donc définis par deux vecteurs propres qui ne sont pas uniques. Dans la matrice F de l'équation 1.9, nous avons fait le choix des exponentielles complexes. Nous savons aussi que l'on peut faire le choix des cosinus et des sinus réels. Ce choix, fondamentalement, est dû à la multiplicité des valeurs propres.
Dans le cas d'un graphe quelconque, la multiplicité de chacune des valeurs propres est en général 1 ; et on n'a plus ce choix entre "exponentielles complexes" et "cosinus".

2.5.3 La fréquence d'un mode de Fourier

Un autre intérêt d'utiliser la matrice laplacienne vient du fait que chaque valeur propre α_i correspond à la fréquence au carré du mode de Fourier i (la $i^{\text{ème}}$ colonne de F). En effet, la matrice laplacienne correspond à la version discrète de l'opérateur de dérivée seconde, qui, quand appliqué à des exponentielles complexes fait émerger la fréquence au carré.
De nouveau par analogie, on va définir la fréquence du mode de Fourier du graphe $\boldsymbol{\chi}_i$ comme étant la racine carrée de sa valeur propre associée λ_i.

2.6 Autres analogies possibles

Les deux analogies précédentes reviennent au même dans les cas où le graphe est non-orienté et régulier (c'est-à-dire que tous les nœuds ont la même force s_0). En effet, dans ce cas, le laplacien peut s'écrire : $L = S - W = s_0(W^\top)^0 - (W^\top)^1$, c'est-à-dire comme un polynôme en $W^\top$: diagonaliser L revient donc au même que diagonaliser $W^\top$. Dans tous les autres cas, les deux analogies proposent des matrices de Fourier différentes, et donc une théorie du traitement du signal différente.

Cette multiplicité est potentiellement problématique, d'autant plus qu'il existe *a priori* de nombreuses autres analogies possibles. Pour en trouver une troisième, il suffit de considérer un type générique de matrice M qui ne soit en général ni polynôme en $W^\top$, ni polynôme en L (sinon on retrouvera un des deux cas précédents en diagonalisant) et dont la matrice pour le graphe boucle orienté ou non-orienté soit circulante. On pourra alors dire : la matrice étant circulante dans le cas classique, elle est donc diagonalisable dans Fourier. Ainsi, en diagonalisant M, on trouvera une nouvelle définition de matrice de Fourier.

2.6.1 La matrice laplacienne normalisée

On peut par exemple considérer la matrice laplacienne normalisée $\mathcal{L} = I_N - S^{-\frac{1}{2}}WS^{-\frac{1}{2}}$. De manière générale, elle n'est ni polynôme en $W^\top$ ni en L. Dans le cas du graphe boucle non-orienté, $\mathcal{L}$ est circulante et s'écrit :

$$\mathcal{L} = I_N - S^{-\frac{1}{2}}WS^{-\frac{1}{2}} = \begin{bmatrix} 1 & -0.5 & 0 & 0 & \cdots & 0 & -0.5 \\ -.5 & 1 & -0.5 & 0 & \cdots & 0 & 0 \\ 0 & -0.5 & 1 & -0.5 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 & -0.5 \\ -0.5 & 0 & 0 & 0 & \cdots & -0.5 & 1 \end{bmatrix}. \tag{1.39}$$

On a donc accès à une troisième définition d'une matrice de Fourier sur graphe : la base qui diagonalise le laplacien normalisé $\mathcal{L}$ du graphe. Nous noterons aussi $\boldsymbol{\chi} = (\boldsymbol{\chi}_1|\boldsymbol{\chi}_2|\cdots|\boldsymbol{\chi}_N)$ cette matrice de Fourier du graphe, et λ_i les valeurs propres associées. Dans des applications où la distribution des degrés est homogène, le choix du laplacien (normalisé ou non) est peu important car les deux matrices sont similaires. Dans les cas où la distribution des degrés est hétérogène, le choix du laplacien peut avoir des conséquences importantes sur les résultats. Les auteurs qui se sont penchés sur la question (lire notamment le chapitre 9 de Chung et Lu [48] ou la partie 8.5 du tutoriel sur le partitionnement spectral de von Luxburg [221]) penchent néanmoins pour l'utilisation du laplacien normalisé pour les raisons suivantes :

1. le spectre du laplacien normalisé est intimement lié à des invariants de graphe comme la constante de Cheeger,
2. la première valeur propre non nulle λ_2 est intimement liée à la vitesse de convergence de marcheurs aléatoires sur le graphe,
3. le spectre du laplacien normalisé a l'avantage calculatoire et conceptuel d'être borné entre 0 et 2,
4. certains travaux comme [61] montrent que l'utilisation du laplacien normalisé donne de meilleurs résultats que le laplacien classique pour la détection de communautés – sans toutefois expliquer pourquoi.

Pour toutes ces raisons, nous considérerons dans la suite le laplacien normalisé, et les notations $\boldsymbol{\chi}$ et λ_i se réfèreront à la diagonalisation de ce laplacien, sauf si expressément indiqué. On gardera néanmoins à l'esprit que le choix final du laplacien est lié au cadre d'emploi.

2.6.2 Extension aux graphes orientés

L'analogie de Sandryhaila et Moura permet de définir la matrice de retard sur les graphes orientés. Un des ennuis de cette analogie est qu'il n'y a pas de lien intuitif entre les valeurs propres et la notion intuitive de fréquence que nous avons. Dans ce sens, l'analogie faite au niveau du laplacien est plus intéressante, mais elle est restreinte, dans la majorité de la littérature, aux graphes non-orientés, car on peut ainsi garantir leur diagonalisation dans $\mathbb{R}$ avec une matrice de passage orthornomée.

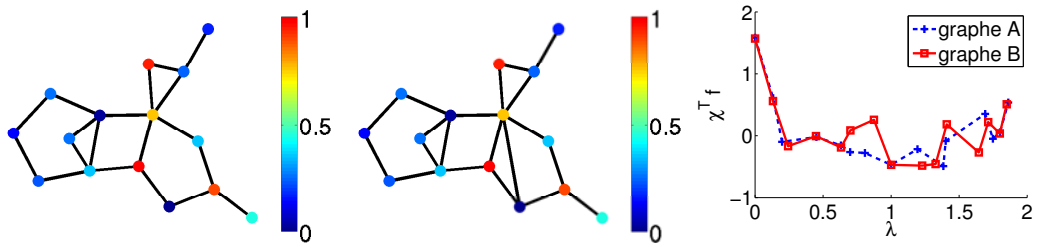


Figure 1.4: Le même signal f sur deux graphes différents qui ne diffèrent que d'un seul lien. Les graphes étant légèrement différents, la transformée de Fourier du signal f change aussi légèrement selon qu'on le considère défini sur un graphe plutôt que l'autre.

Néanmoins, on peut *a priori* étendre la notion de laplacien aux graphes orientés, en choisissant tout d'abord si on considère le poids des liens qui sortent de chaque nœud ou le poids des liens qui arrivent en chaque nœud pour remplacer la matrice S dans la définition du laplacien. Ensuite, on peut argumenter que toute matrice laplacienne ainsi définie n'est pas forcément diagonalisable (même dans $\mathbb{C}$). Qu'à cela ne tienne : les matrices diagonalisables dans $\mathbb{C}$ sont denses dans l'espace des matrices et les graphes que nous analysons sont, en pratique, mesurés d'une manière ou d'une autre. Si par manque de chance on tombe sur une matrice non-diagonalisable, en la changeant légèrement (et tout en restant dans les barres d'erreur de mesure), on peut la diagonaliser. On aura alors accès à des vecteurs et valeurs propres en général complexes qui peuvent définir une transformée de Fourier sur graphe. Une autre idée, proposée dans un preprint récent par Mendes et al. [148], est de symmétriser le laplacien L en $LL^\top$, avant de le diagonaliser.

Dans cette thèse, nous n'explorerons pas plus avant ces possibilités. Le domaine de recherche sur le traitement du signal sur graphes étant pour l'instant peu exploré, nous nous cantonnerons aux cas plus simples des graphes non-orientés où la transformée de Fourier est définie avec la matrice laplacienne normalisée. L'extension aux graphes orientés pouvant en principe être faite d'une des deux manières que nous venons d'évoquer.

2.7 Résultats du traitement du signal sur graphe

2.7.1 La transformée de Fourier sur graphe

La transformée de Fourier de f est ainsi définie par :

$$\forall k \in [1, N] \qquad \hat{f}(k) =< \boldsymbol{\chi}_k, f >= \sum_{l=1}^{N} \boldsymbol{\chi}_k(l) f(l), \tag{1.40}$$

ce qui s'écrit en matriciel :

$$\hat{f} = \boldsymbol{\chi}^\top f. \tag{1.41}$$

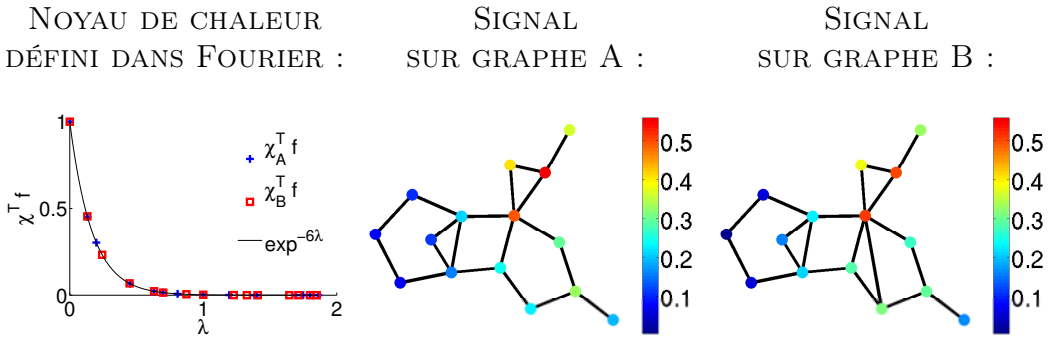


Figure 1.5: Transformée de Fourier inverse du noyau de la chaleur sur deux graphes qui ne diffèrent que d'un seul lien. Les transformées inverses sont similaires mais pas identiques.

La Fig. 1.4 montre un même signal f sur deux graphes qui ne diffèrent que d'un seul lien, et leurs transformées de Fourier respectives. On observe qu'un signal est intimement lié à la topologie du graphe sur lequel il est défini : les deux transformées de Fourier sont similaires mais pas identiques.

La transformée de Fourier inverse de $\hat{f}$ s'écrit, en matriciel :

$$f = \boldsymbol{\chi}\hat{f}, \tag{1.42}$$

car $\boldsymbol{\chi}^{-1} = \boldsymbol{\chi}^\top$ étant donné que $\boldsymbol{\chi}$ est orthonormée. On a l'identité de Parseval :

$$\forall (f,g) \in \mathbb{R}^N \times \mathbb{R}^N \qquad <\hat{g},\hat{f}>= \hat{g}^\top \hat{f} = g^\top \boldsymbol{\chi}\boldsymbol{\chi}^\top f = g^\top f =< g, f > . \tag{1.43}$$

La Fig. 1.5 montre deux signaux $\chi_A^\top f$ et $\chi_B^\top f$ définis à partir du noyau de la chaleur $\exp^{-6\lambda}$ continu dans l'espace de Fourier. Même si définis à partir du même noyau, les deux signaux sont différents car les valeurs propres des deux graphes ne sont pas tout à fait identiques. Les signaux associés dans l'espace direct sont également similaires, mais pas identiques.

2.7.2 La fréquence d'un mode de Fourier

Pour donner une intuition quant à la notion de fréquence d'un mode de Fourier sur graphe, définissons son nombre de passages à zéro. Soit un graphe $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ et un signal sur ce graphe f. Le nombre de passages à zéro $N_{pz}(f)$ de f est le nombre de liens connectant deux nœuds associés à des valeurs de signes opposés. Formellement :

$$N_{pz}(f) = \#\left\{(ij) \in \mathcal{E} \text{ tel que } f(i)f(j) < 0\right\}. \tag{1.44}$$

Pour différents types de graphes, nous traçons sur la Fig. 1.6 $N_{pz}(\boldsymbol{\chi}_i)$ en fonction de λ_i. Même si l'augmentation du nombre de passages par zéro en fonction de la fréquence au carrée λ est moins graduelle et stable que pour le cas classique, on observe qu'il existe une forte corrélation entre le nombre de passages par zéro et λ.

2.7.3 Phénomène de localisation des modes de Fourier

De manière surprenante, les modes de Fourier sur graphe peuvent être localisés, c'est-à-dire peuvent concentrer leur énergie sur quelques nœuds. La Fig. 1.7 compare

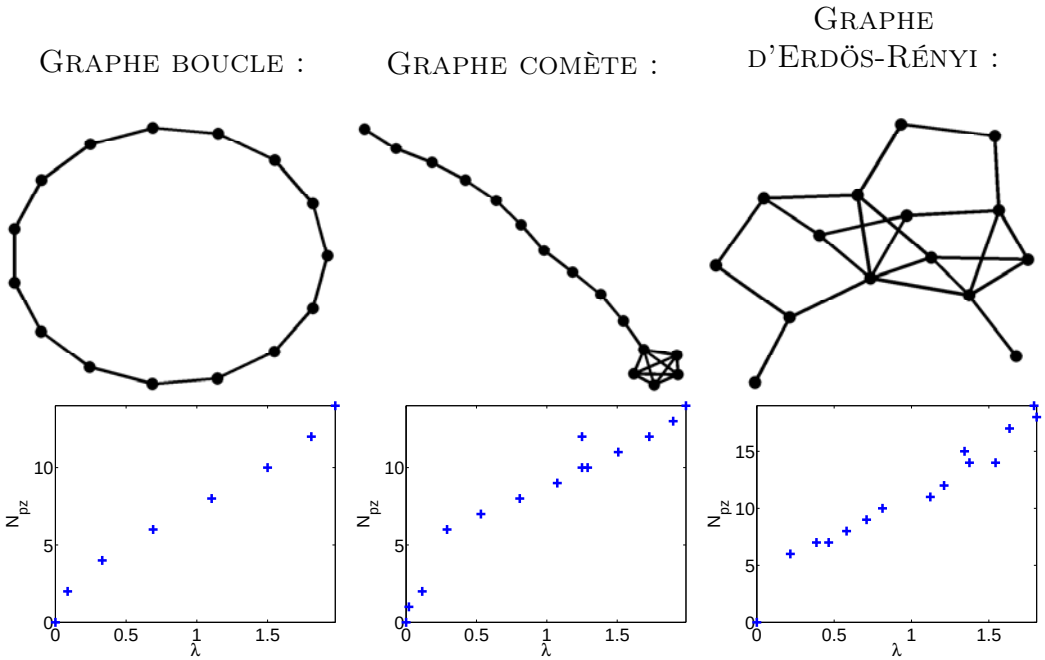


Figure 1.6: Pour chacun des trois graphes, est tracé le nombre de passages par zéro de chacun des vecteurs propres $N_{pz}(\boldsymbol{\chi}_i)$ du laplacien, en fonction de leur valeur propre associée λ_i.

les modes de Fourier classiques (obtenus avec le graphe boucle) et les modes de Fourier du graphe comète. Alors que les sinusoïdes classiques ont une énergie distribuée sur l'ensemble des nœuds, ce n'est pas le cas pour certains modes de Fourier du graphe comète : en particulier pour les modes numéro 8, 9 et 10 (qui correspondent aux trois colonnes entre $\lambda = 1.24$ inclus et $\lambda = 1.28$ exclus) qui sont localisés au niveau de la tête de la comète. On a donc ici des modes de Fourier qui sont à la fois localisés en fréquence et en espace !

On peut également voir ce phénomène de localisation de la manière suivante. Considérons le graphe boucle dont tous les liens ont un poids de 1 sauf deux qui ont un poids w qui peut varier. Quand w tend vers zéro, le graphe tend vers deux graphes déconnectés et les modes de Fourier tendent vers des modes complètement localisés sur l'un ou l'autre sous-graphe.

Ce phénomène de localisation est encore mal compris, même si beaucoup étudié [195, 153, 146, 20]. Etant donné qu'il viole le principe d'incertitude selon lequel on ne peut avoir localisation en espace et en fréquence simultanément, il est important de le garder en mémoire. Comme dans [195], nous noterons μ le maximum en valeur absolue de la matrice de Fourier du graphe :

$$\mu = \max |\boldsymbol{\chi}|. \tag{1.45}$$

μ est un indicateur de régularité du graphe appelé cohérence. En effet, $\mu \in \left[\frac{1}{\sqrt{N}}, 1\right]$ vaut $\frac{1}{\sqrt{N}}$ uniquement dans le cas classique et μ est d'autant plus proche de 1 que le

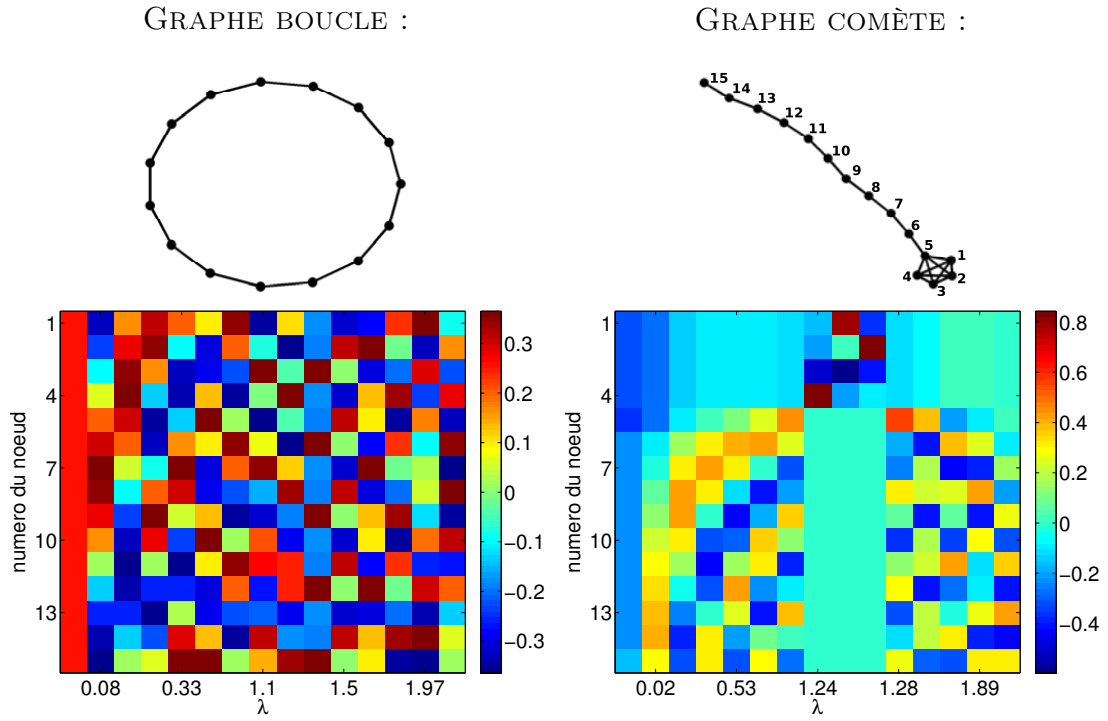


Figure 1.7: Pour chacun des deux graphes, la matrice représente χ, la matrice de Fourier du graphe. On observe un phénomène de localisation pour le graphe comète qui n'existe pas pour le graphe boucle.

graphe est différent du cas classique. Dans le cas du graphe comète de la Fig. 1.7, $\mu = 0.84$, alors que $\mu = 0.26$ dans le cas classique avec même nombre de noeuds.

2.7.4 Point clé

Un des points clés de cette analogie est qu'elle propose une équivalence entre un signal défini sur des nœuds d'un graphe (qui n'ont pas d'ordre naturel les uns par rapport aux autres et qui ont des liens non-triviaux entre eux) et un signal défini sur le spectre du graphe qui, lui, est ordonné ($\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_N$). Par exemple, nous verrons que la translation sur graphe est difficile à définir directement parce qu'il n'y a pas d'équivalent naturel de la convolution (à cause de l'absence d'ordre dans l'espace indexé par les nœuds). Nous passerons donc, suivant Shuman et al. [195] par l'espace de Fourier pour la définir. Nous ferons de même pour d'autres définitions qui posent problème.

2.7.5 Opérations usuelles

Nous reprenons les opérations usuelles classiques rappelées dans la section 2.3, et nous les adaptons au cas des graphes [193].

La convolution de deux signaux sur graphe f et g se définit comme une multiplication dans l'espace de Fourier, comme précisé dans la sous-partie 2.3.2 :

$$\forall k \in [1, N] \qquad \widehat{(f * g)}(k) = \sqrt{N}\hat{f}(k)\hat{g}(k). \tag{1.46}$$

Signal original f : | Signal translaté $\mathcal{T}_{15}f$: | Signal translaté $\mathcal{T}_{13}f$:

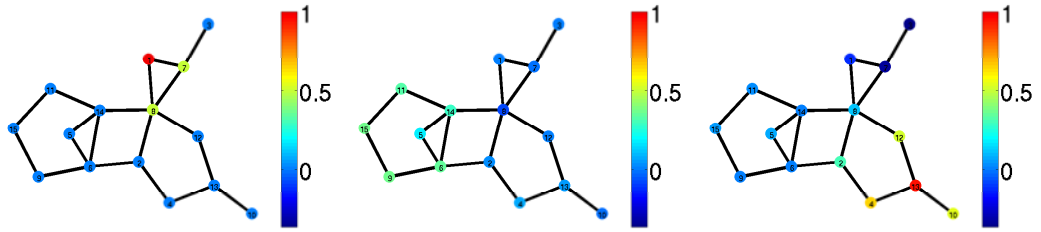


Figure 1.8: Un signal f (à gauche) translaté soit autour du nœud 15 (au milieu) soit autour du nœud 13 (à droite).

Donc, en repassant dans l'espace direct :

$$\forall i \in [1, N] \qquad (f * g)(i) = \sqrt{N} \sum_{l=1}^{N} \boldsymbol{\chi}_l(i) \hat{f}(l) \hat{g}(l). \tag{1.47}$$

La translation d'un signal f par $i_0 \in [1, N]$ peut s'écrire en classique comme la convolution $\mathcal{T}_{i_0} f = f * \delta_{i_0}$. Ainsi, d'après l'équation 1.47 :

$$\forall i \in [1, N] \qquad (\mathcal{T}_{i_0} f)(i) = \sqrt{N} \sum_{l=1}^{N} \boldsymbol{\chi}_l(i) \hat{f}(l) \hat{\delta}_{i_0}(l). \tag{1.48}$$

Or, d'après l'équation 1.41 : $\forall l \in [1, N] \quad \hat{\delta}_{i_0}(l) = \boldsymbol{\chi}_l(i_0)$. Ainsi :

$$\forall i \in [1, N] \qquad (\mathcal{T}_{i_0} f)(i) = \sqrt{N} \sum_{l=1}^{N} \boldsymbol{\chi}_l(i) \hat{f}(l) \boldsymbol{\chi}_l(i_0). \tag{1.49}$$

Un exemple de translation de signal sur graphe est donné sur la Fig. 1.8.

La modulation d'un signal f par $k_0 \in [1, N]$ revient à une multiplication dans l'espace direct. L'équation 1.20 donne, en remplaçant l'exponentielle complexe $e^{\frac{-2i\pi(k_0-1)(j-1)}{N}}$ par le mode de Fourier sur graphe $\boldsymbol{\chi}_{k_0}(j)$:

$$\forall j \in [1, N] \qquad \mathcal{M}_{k_0} f(j) = \sqrt{N} f(j) \boldsymbol{\chi}_{k_0}(j) \tag{1.50}$$

Un exemple de modulation de signal sur graphe est donné sur la Fig. 1.9.

Le filtrage d'un signal f par un filtre $h \in \mathbb{R}^N$ s'écrit, en reprenant l'équation 1.25 :

$$f^h = \sqrt{N} \boldsymbol{\chi} \hat{H} \boldsymbol{\chi}^\top f, \tag{1.51}$$

où $\hat{H} = \mathrm{diag}(\hat{h}) = \mathrm{diag}(\boldsymbol{\chi}^\top h)$.

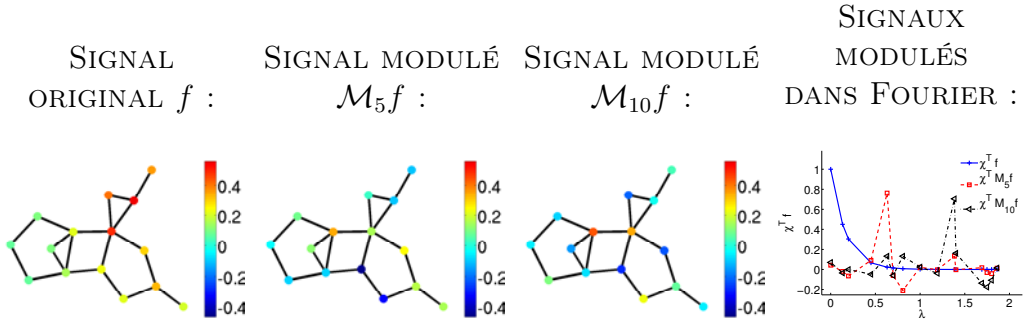


Figure 1.9: Le noyau de la chaleur (à gauche), modulé autour de la 5ème valeur propre ($\mathcal{M}_5 f$) ou de la 10ème valeur propre ($\mathcal{M}_{10} f$). À droite : les trois signaux dans l'espace de Fourier.

D'autres opérations usuelles ont été développées que nous n'allons pas présenter en détail ici. On peut citer les efforts faits pour définir une interpolation sur graphe [93, 154], un processus de lifting [155] ou une transformée de Fourier locale [194]. Forts de ces opérations usuelles d'un signal sur graphe, nous allons maintenant présenter un des outils que nous allons beaucoup utiliser dans cette thèse : les ondelettes sur graphe.

3 La transformée en ondelettes sur graphes

Il existe à ce jour plusieurs définitions d'ondelettes sur graphe. Notons la proposition de Crovella et Kolaczyk [55] qui définissent des ondelettes en termes de diffusion sur l'entourage de chaque nœud. Nous pouvons également citer le travail de Coifman et Maggioni [53], qui définissent des "ondelettes de diffusion", qui utilisent l'opérateur de diffusion pour lisser les signaux définis sur le graphe et proposer une version multiéchelle du signal. Citons également des transformées en ondelettes sur graphe définies par lifting [155, 118] ou par bancs de filtre [156, 69], et des ondelettes définies sur des graphes particuliers, comme les arbres [87]. Dans la suite, nous nous intéresserons aux ondelettes dites spectrales proposées par [96], car elles s'adaptent particulièrement bien au problème que nous aborderons dans le deuxième chapitre : la détection de communautés. En effet, elle se base sur le laplacien du graphe, comme les techniques spectrales de partitionnement, et on peut régler certains paramètres de ces ondelettes pour qu'elles soient particulièrement sensibles à la structure en communautés (comme nous le verrons dans le deuxième chapitre). Les ondelettes sur graphe sont plus faciles à introduire en procédant par analogie par rapport au cas classique *continu*. Nous commençons par écrire une manière de définir les ondelettes continues classiques dans l'espace de Fourier, pour pouvoir ensuite mener à bien l'analogie.

3.1 La famille d'ondelettes classiques

Le lecteur familier avec les ondelettes trouvera ici un rappel sur les notations utilisées et quelques définitions bien connues. En revanche, le lecteur qui découvre

les ondelettes trouvera toute information nécessaire à leur sujet dans le livre de Mallat [141]. La transformée en ondelettes d'un signal f est sa décomposition sur la famille Ψ dite d'ondelettes, composée de translations et d'homothéties d'une ondelette mère. Formellement, les ondelettes filles $\psi_{s,\tau}$ (où $(s,\tau) \in \mathbb{R}^{+*} \times \mathbb{R}$) qui composent la famille d'ondelettes Ψ se déterminent à partir de l'ondelette mère ψ :

$$\forall t \in \mathbb{R} \qquad \psi_{s,\tau}(t) = \frac{1}{\sqrt{s}}\psi\left(\frac{t-\tau}{s}\right). \tag{1.52}$$

τ est le paramètre de translation et s le paramètre d'échelle. L'ondelette mère ψ est une fonction de carré intégrable et de moyenne nulle, son choix précis caractérise entièrement l'analyse multiéchelle rendue possible par la transformée. Observons que :

$$\begin{aligned}
\frac{1}{\sqrt{s}}\hat{\psi}_{s,\tau}(\omega) &= \int_{-\infty}^{\infty} \frac{1}{s}\psi\left(\frac{t-\tau}{s}\right) e^{-i\omega t} dt \\
&= e^{-i\omega\tau} \int_{-\infty}^{\infty} \frac{1}{s}\psi\left(\frac{T}{s}\right) e^{-i\omega T} dT \\
&= e^{-i\omega\tau} \int_{-\infty}^{\infty} \psi\left(T'\right) e^{-is\omega T'} dT' \\
&= \hat{\delta}_\tau(\omega)\, \hat{\psi}(s\omega) \qquad \text{où} \quad \delta_\tau = \delta(t-\tau).
\end{aligned} \tag{1.53}$$

Donc, une écriture possible de l'ondelette fille $\hat{\psi}_{s,\tau}$ est :

$$\psi_{s,\tau}(t) \;=\; \sqrt{s}\int_{-\infty}^{\infty} \hat{\delta}_\tau(\omega)\hat{\psi}(s\omega)\exp^{i\omega t}\, d\omega. \tag{1.54}$$

$\hat{\psi}(s\omega)$ peut être considéré comme un banc de filtre obtenu en dilatant le noyau de filtre $\hat{\psi}$. De plus, l'ondelette mère étant de moyenne nulle et d'énergie finie, $\hat{\psi}$ est un passe-bande (tend vers zéro aux basses fréquences et aux hautes fréquences). Finalement, l'ondelette fille $\psi_{s,\tau}$ peut être interprétée (à un facteur $\sqrt{s}$ près) comme un Dirac centré en τ filtré par le passe-bande dilaté $\hat{\psi}(s\cdot)$.

3.2 La famille d'ondelettes sur graphes

L'équivalent sur graphe de la variable temporelle t est le numéro de nœud i. Ainsi, le Dirac centré en τ devient un Dirac discret centré sur le nœud a. On peut *a priori* garder le paramètre d'échelle s continu et si on définit la fonction continue $g : \mathbb{R}^+ \to \mathbb{R}$ l'équivalent du noyau de filtre passe-bande $\hat{\psi}$, la transformée de Fourier de l'ondelette fille $\psi_{s,a}$ s'écrit :

$$\forall k \in [1, N] \qquad \hat{\psi}_{s,a}(k) = \sqrt{Ns}\hat{\delta}_a(k)g(s\lambda_k). \tag{1.55}$$

En notant $G_s = \mathrm{diag}(g(s\lambda_1)|g(s\lambda_2)|\cdots|g(s\lambda_N))$ la matrice de filtre, on peut écrire en matriciel :

$$\hat{\psi}_{s,a} = \sqrt{Ns}G_s\hat{\delta}_a = \sqrt{Ns}G_s\boldsymbol{\chi}^\top\delta_a, \tag{1.56}$$

si bien que, dans l'espace direct :

$$\psi_{s,a} = \sqrt{Ns}\boldsymbol{\chi}G_s\boldsymbol{\chi}^\top\delta_a. \tag{1.57}$$

La matrice d'ondelettes sur graphe à l'échelle s, notée Ψ_s s'écrit :

$$\Psi_s = (\psi_{s,1}|\psi_{s,2}|\cdots|\psi_{s,N}) = \sqrt{Ns}\boldsymbol{\chi}G_s\boldsymbol{\chi}^\top. \tag{1.58}$$

L'énergie de l'ondelette fille $\psi_{s,a}$ se calcule à partir de l'équation 1.55 :

$$||\psi_{s,a}||_2^2 = \sum_{k=1}^{N} \hat{\psi}_{s,a}(k)^2 = Ns\sum_{k=1}^{N} \boldsymbol{\chi}_k(a)^2 g(s\lambda_k)^2. \tag{1.59}$$

En général, de par le caractère discret du spectre d'un graphe et malgré le facteur $\sqrt{s}$ qui permet la normalisation dans le cas continu, et le facteur $\sqrt{N}$ qui vient de l'équation de filtrage sur graphe, la norme de l'ondelette fille est, sauf exception, différente de la norme de l'ondelette mère :

$$\sum_{i=1}^{N} \psi_{s,a}(i)^2 \neq \sum_{k=1}^{N} g(\lambda_k)^2. \tag{1.60}$$

L'existence du facteur de normalisation $\sqrt{Ns}$ n'étant plus justifiée, on simplifie l'équation 1.58 en :

$$\Psi_s = (\psi_{s,1}|\psi_{s,2}|\cdots|\psi_{s,N}) = \boldsymbol{\chi}G_s\boldsymbol{\chi}^\top, \tag{1.61}$$

tout en sachant qu'elle définit des ondelettes non-normées. Pour les normer, on le fait "à la main", et on les note :

$$\forall(s,a) \in \mathbb{R}^+ \times \mathcal{V} \qquad \tilde{\psi}_{s,a} = \frac{\psi_{s,a}}{||\psi_{s,a}||_2}. \tag{1.62}$$

On note la matrice des ondelettes normées à l'échelle s :

$$\tilde{\Psi}_s = \left(\tilde{\psi}_{s,1}|\tilde{\psi}_{s,2}|\cdots|\tilde{\psi}_{s,N}\right). \tag{1.63}$$

Le paramètre d'échelle étant continu, il existe une infinité de matrices d'ondelettes, toutes différentes. En pratique, nous sélectionnons M paramètres d'échelle $(s_1|s_2|\cdots|s_M)$ d'une manière qui dépendra de l'application. La famille de toutes les ondelettes sur graphe est alors la réunion de toutes les ondelettes à toutes les échelles :

$$\Psi = \bigcup_{j=1}^{M} \Psi_{s_j}. \tag{1.64}$$

Il y a N ondelettes par échelle s_j, Ψ est donc composée de NM ondelettes.

3.3 La transformée en ondelettes sur graphe

Soit un signal sur graphe f. Son coefficient d'ondelette $Wf_{s,a}$ au nœud a à l'échelle s s'écrit :

$$Wf_{s,a} = \psi_{s,a}^\top f. \tag{1.65}$$

La réunion de tous ces coefficients est la transformée en ondelettes de f. On notera parfois Wf_s le vecteur de taille N regroupant les coefficients à l'échelle s :

$$Wf_s = (Wf_{s,1}|Wf_{s,2}|\cdots|Wf_{s,N})^\top = \Psi_s f. \tag{1.66}$$

La famille des ondelettes ne peut pas être une base étant donné qu'elle est potentiellement très redondante. On introduit communément le concept de trame (*frame* en anglais) ci-après. Une famille $\{\phi_n\}_{n\in\Gamma}$ est une frame d'un espace vectoriel H s'il existe deux constantes $A > 0$ et $B > 0$ telles que pour tout $f \in H$:

$$A||f||^2 \leq \sum_{n\in\Gamma} | < f, \phi_n > |^2 \leq B||f||^2. \tag{1.67}$$

La famille d'ondelettes telle que donnée par l'équation 1.64 n'est pas une frame car l'inégalité ci-dessus n'est pas vérifiée pour les signaux constants. Il faut dans un premier temps ajouter à Ψ la matrice $\boldsymbol{\chi}\text{diag}(g_0(\lambda_k))\boldsymbol{\chi}^\top$ obtenue en définissant le filtre passe-bas g_0 : on note cette famille élargie Ψ_0. Ψ_0 est alors une frame si et seulement si :

$$\exists (A, B) > 0 \quad \forall k \in [1, N] \qquad A \leq g_0(\lambda_k) + \sum_{j=1}^{M} g(s_j\lambda_k) \leq B. \tag{1.68}$$

Dans ce cas, tout signal est reconstructible à partir de sa décomposition dans Ψ_0.

3.4 Algorithme rapide de transformée en ondelettes sur graphe

Une des difficultés rencontrées en traitement du signal sur graphe est la dépendance de tous les opérateurs à la topologie du graphe : on ne peut pas calculer, comme dans le cas classique, des ondelettes une bonne fois pour toutes car justement ces ondelettes dépendent du graphe analysé. En effet, l'équation 1.61 montre que le calcul de la matrice des ondelettes à l'échelle s nécessite la connaissance de la matrice de Fourier $\boldsymbol{\chi}$, elle-même calculée en diagonalisant le laplacien du graphe. Or, la diagonalisation d'une matrice de taille N se fait au mieux en un temps de calcul cubique en N, ce qui interdit son utilisation pour des graphes de plus de quelques milliers de nœuds.
Pour surmonter cette difficulté et calculer la transformée en ondelettes d'un signal f rapidement, on ne va pas chercher à rendre explicites les ondelettes. Comme l'expliquent Shuman et al. [196], on va plutôt chercher à approximer chaque filtre $g(s_j\cdot)$ en un polynôme de Tchebychev tronqué au degré m. Considérons l'équation 1.66. Pour obtenir les coefficients d'ondelettes à l'échelle s d'un signal f, soit on calcule explicitement Ψ_s, et cela revient à diagonaliser le laplacien. Soit nous approximons $g(s\cdot)$ en un polynôme de degré m :

$$\forall \lambda \in \mathbb{R}^+ \qquad g(s\lambda) \simeq \sum_{i=1}^{m} \alpha_i \lambda^i, \tag{1.69}$$

i.e. :

$$G_s \simeq \sum_{i=1}^{m} \alpha_i \Lambda^i. \tag{1.70}$$

On observe alors que :

$$Wf_s = \Psi_s f = \boldsymbol{\chi} G_s \boldsymbol{\chi}^\top f \simeq \sum_{i=1}^{m} \alpha_i \boldsymbol{\chi} \Lambda^i \boldsymbol{\chi}^\top f = \sum_{i=1}^{m} \alpha_i \mathcal{L}^i f. \tag{1.71}$$

Ainsi, au lieu de devoir calculer la diagonalisation de $\mathcal{L}$, nous accédons aux coefficients d'ondelette Wf_s uniquement via des multiplications matrice-vecteur. Pour des raisons de rapidité de calcul et d'optimisation de l'approximation évoquées dans [96], le polynôme utilisé dans l'approximation de l'équation 1.69 est la troncature au degré m du développement en polynôme de Tchebychev du filtre $g(s\cdot)$. Le temps de calcul pour obtenir les NM coefficients chute à $O(M|\mathcal{E}| + NMm)$ où $|\mathcal{E}|$ est le nombre de liens. Pour les graphes creux, c'est-à-dire les graphes dont le nombre de liens est de l'ordre du nombre de nœuds (beaucoup de graphes de terrain ont cette propriété), le temps de calcul de l'algorithme est linéaire en N. Évidemment, la contrepartie est que nous obtenons uniquement une approximation des coefficients Wf_s (qui est d'autant meilleure que m est grand).

Dans la suite, nous noterons $\mathcal{FWT}_{s,m}$ l'opérateur de transformée en ondelettes rapide à l'échelle s (avec une justesse d'approximation paramétrée par m) :

$$Wf_s \simeq \mathcal{FWT}_{s,m} f. \tag{1.72}$$

3.5 Le noyau de filtre d'ondelettes

La famille d'ondelettes Ψ est entièrement déterminée, pour un graphe donné, par son noyau de filtre g et le choix des M échelles considérées. Nous verrons plus tard comment choisir ces échelles, mais nous pouvons d'ores et déjà nous intéresser à g. En classique, la construction de son équivalent $\hat{\psi}$ a été l'objet de beaucoup d'efforts de recherche depuis trente ans et des ondelettes de toutes formes existent maintenant, chacune plus ou moins adaptée au type de signaux étudiés. En ondelettes sur graphe, pour assurer la localisation des ondelettes, la seule contrainte qui existe sur g porte sur son comportement à l'origine qui doit être en x^α ($\alpha > 1$). Pour l'instant, une des formes génériques de passe-bande étudiées, est la forme "en cloche" avec un comportement en x^α à l'origine et en $x^{-\beta}$ aux grandes valeurs propres. D'autres propositions plus sophistiquées ont été proposées [197, 137], qui prennent en compte le spectre particulier de chaque graphe, mais nous n'allons pas étudier ces variantes qui sont plus compliquées et demandent à calculer l'intégralité du spectre, ce qui exclut les grands graphes. Plus précisément, g est tel que :

$$g(x;\alpha,\beta,x_1,x_2) = \begin{cases} x_1^{-\alpha}\, x^\alpha & \text{pour} \quad x < x_1 \\ p(x) & \text{pour} \quad x_1 \leq x \leq x_2 \\ x_2^{\beta}\, x^{-\beta} & \text{pour} \quad x > x_2. \end{cases} \tag{1.73}$$

où $p(x)$ est l'unique interpolation polynomiale cubique qui conserve la continuité de g et de sa dérivée g'. α, β, x_1, x_2 sont les paramètres du filtre. Nous verrons aussi dans le deuxième chapitre comment les ajuster en fonction de ce que l'on cherche à analyser, mais pour l'instant, dans le but d'illustrer les ondelettes, nous les choisissons comme dans [96] : $\alpha = \beta = 2$, $x_1 = 1$ et $x_2 = 2$. La Fig. 1.10a montre le noyau de filtre dilaté par 5 paramètres d'échelle différents. On définit ces filtres comme des fonctions continues par commodité, mais en réalité, ces filtres sont discrets quand on les applique à des graphes : ils sont définis uniquement sur le spectre du graphe. On montre la version discrète du même banc de filtre sur la Fig. 1.10b, en utilisant le spectre du graphe de la Fig. 1.2.

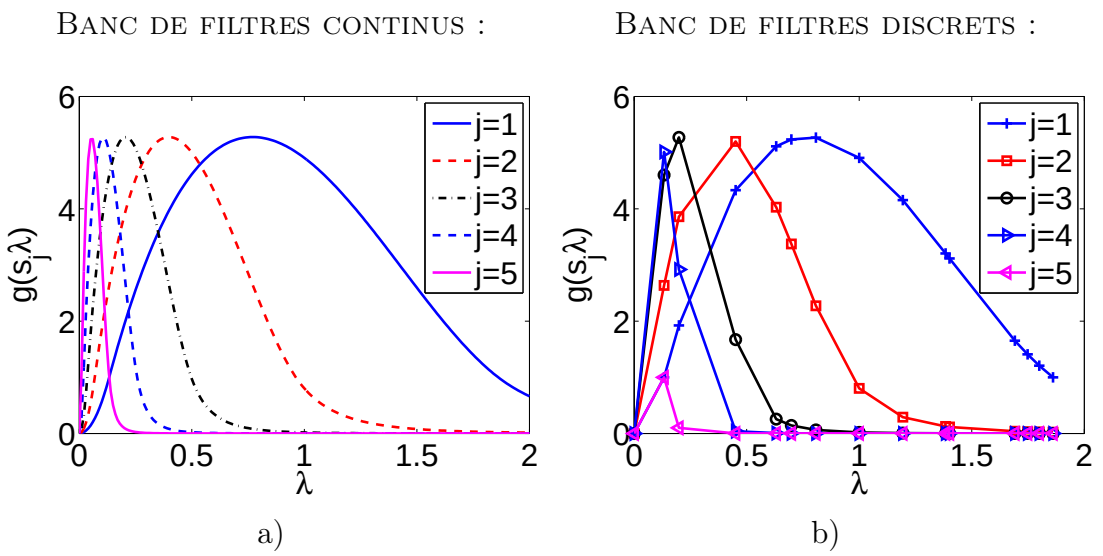


Figure 1.10: a) Version continue et b) discrète (définie sur le spectre du graphe de la Fig. 1.2) d'un banc de filtres passe-bande définissant des ondelettes sur graphe.

3.6 Quelques illustrations

Illustrons quelques ondelettes du graphe de la Fig. 1.2 correspondant au banc de filtres de la Fig. 1.10b. Dans un premier temps, on dessine sur la Fig. 1.11 dix ondelettes : les ondelettes centrées aux nœuds 8 et 15 aux cinq échelles considérées dans le banc de filtres. Par exemple, on observe très bien l'oscillation de l'ondelette $\psi_{s_1,15}$. Aussi, on observe qu'à petite échelle et loin du nœud autour duquel l'ondelette est centrée, il n'y a presque plus d'énergie (les nœuds sont alors en vert clair). À grandes échelles, au contraire, l'énergie est distribuée sur tout le graphe, à tel point qu'il n'est plus possible de savoir à l'œil sur quel nœud l'ondelette est censée être centrée.

Pour donner une intuition sur le comportement de la transformée en ondelettes, on représente sur la Fig. 1.12 un signal très simple (le Dirac centré sur le nœud 15) et sa transformée en ondelettes. La transformée est représentée sous forme de matrice où l'élément de la ligne i et de la colonne j est le coefficient d'ondelette $Wf_{s_i,j}$ calculé à l'échelle s_i et au nœud j. On observe l'équivalent de la cascade d'énergie centrée autour des singularités d'un signal dans une transformée en ondelettes classique. Sauf que dans notre cas, il faut reconstruire mentalement la cascade étant donné qu'il n'y a pas d'ordre naturel dans l'espace des nœuds.

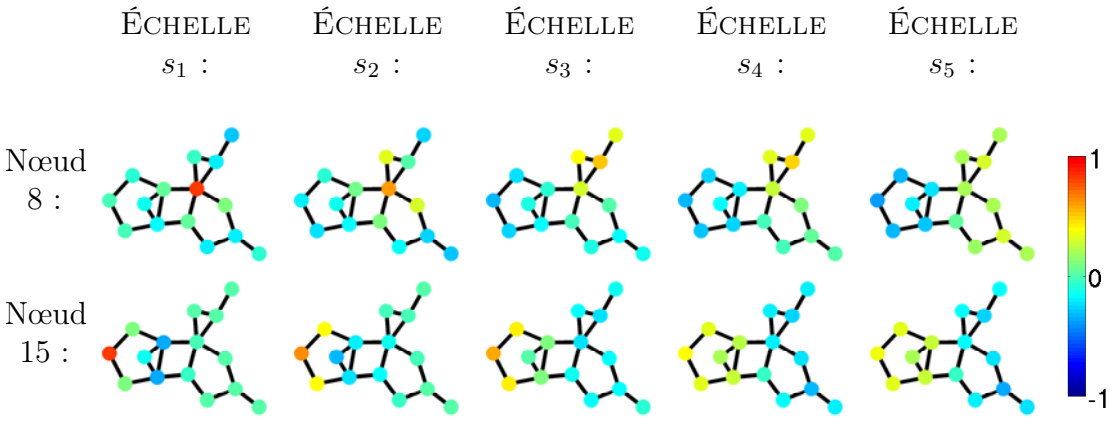


Figure 1.11: Exemples d'ondelettes sur le graphe de la Fig. 1.2 aux cinq échelles considérées dans le banc de filtres de la Fig. 1.10. Nous ne représentons que les ondelettes centrées autour des nœuds 8 et 15.

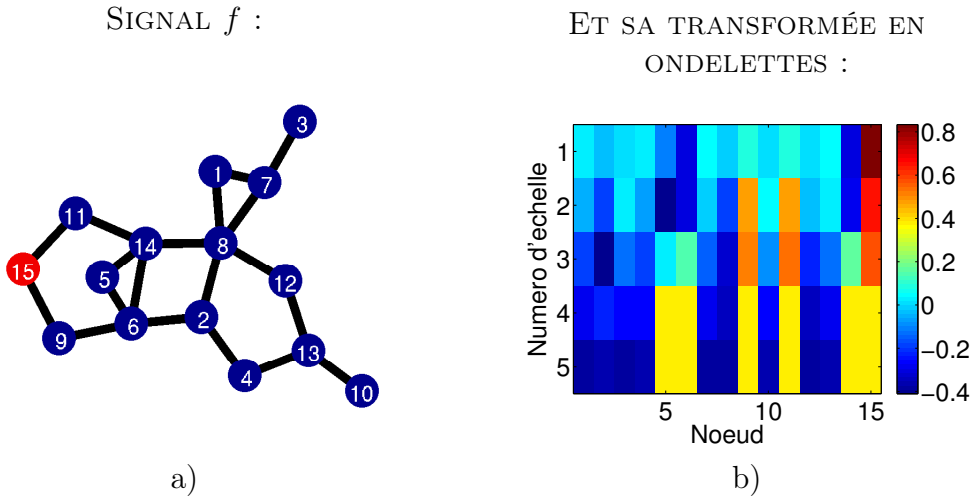


Figure 1.12: Le delta centré autour du nœud 15 (a) et sa transformée en ondelettes (b) pour les cinq échelles considérées dans le banc de filtres de la Fig. 1.10.

Chapitre 2

Détection multiéchelle de communautés à l'aide d'ondelettes

« No man is an island, entire of itself ; every man is a piece of the continent, a part of the main. »

– John Donne, *No Man Is An Island*

La première partie de ce chapitre est dédiée à une courte présentation générale sur la notion de communautés et l'utilité de leur détection. Nous rappellerons dans la deuxième partie quelques travaux classiques de partitionnement d'un graphe en communautés. Le choix des sujets abordés pourra paraître arbitraire au lecteur au fait de la volumineuse littérature sur le sujet : en effet, le but n'est pas de proposer une liste exhaustive des méthodes existantes, mais d'introduire le lecteur à quelques outils de base auxquels nous ferons appel par la suite. Ces deux premières parties sont surtout destinées au lecteur peu familier de la notion de communauté. La troisième partie, en revanche, porte plus précisément sur les méthodes multiéchelles existantes de détection en communautés et permet de poser précisément le problème abordé. Les quatre parties suivantes (parties 4, 5, 6 et 7) sont le cœur de ce chapitre et proposent une nouvelle méthode multiéchelle basée sur les ondelettes sur graphe. Elles reprennent avec plus de détails les idées essentielles déjà publiées dans [212, 216, 215, 211, 210]. La huitième partie présente un point de vue original sur une partie des méthodes multiéchelles existantes, les regroupant toutes sous un formalisme unique de modularité filtrée. Nous évoquerons dans la neuvième et dernière partie de ce chapitre les perspectives de ces travaux.

Sommaire

1 Une communauté dans un graphe **41**
1.1 Qu'est-ce qu'une communauté ? 41
1.2 De l'utilité de la recherche de communautés 42
1.3 Une brève histoire de la recherche de communautés . . . 43
1.4 Partitionnement d'un graphe en communautés 43
2 Partitionner un graphe en communautés **44**
2.1 La modularité : une mesure de la qualité d'une partition 44
2.2 L'algorithme de Louvain 45
2.3 Clusterings hiérarchiques et dendrogrammes 45
2.4 Un algorithme spectral 47
3 État de l'art des méthodes de détection multiéchelle **48**
3.1 Les limites de la modularité 50
3.2 L'intérêt d'une vision multiéchelle 50
3.3 Poser le problème multiéchelle 51
3.4 Trouver une partition par échelle : méthodes existantes . 52
3.5 Trouver les échelles pertinentes : méthodes existantes . . 55
4 Ondelettes sur graphes et partitionnement multiéchelle **57**
4.1 Pourquoi une autre méthode ? 58
4.2 Le noyau de filtre d'ondelette et bornes du paramètre d'échelle 58
4.3 Détection de communautés à une échelle s 62
4.4 Détection rapide de communautés à une échelle s 67
4.5 Mesure de stabilité de l'échelle s 69
4.6 Test statistique 70
5 Illustrations et comparaison avec d'autres méthodes **71**
5.1 Illustration sur un modèle de graphe hiérarchique 71
5.2 Comparaison de mesures de stabilité 75
5.3 Comparaison avec d'autres algorithmes multiéchelles . . 79
5.4 Illustration du test statistique 84
5.5 Illustration sur un modèle de graphe avec une seule échelle 85
5.6 Conclusion 89
6 Application à un graphe de terrain **89**
7 Définition et utilisation de fonctions d'échelle sur graphe **91**
8 Réinterprétation de quelques méthodes multiéchelles **95**
8.1 Quelques notations 95
8.2 Forme canonique de modularité filtrée 95
8.3 La modularité filtrée de la méthode de Delvenne et al. . 97
8.4 La modularité filtrée de la méthode d'Arenas et al. . . . 98
8.5 La modularité filtrée de la méthode de Reichardt et Bornholdt 100
8.6 La modularité filtrée de la méthode de Ronhovde et Nussinov 101

8.7 Deux nouvelles propositions de modularité filtrée 101
8.8 Les filtres équivalents de la modularité classique 102
8.9 Comparaison et discussion 102
9 Conclusions et perspectives **104**
9.1 Conclusions 104
9.2 Perspectives 105

Notations Dans ce chapitre, nous nous efforcerons de noter :
- en petits caractères gras les vecteurs, comme $\boldsymbol{f}$ la notation générique d'un signal défini sur les nœuds d'un graphe ;
- en grands caractères gras les matrices, comme $\boldsymbol{W}$ la matrice d'adjacence pondérée d'un graphe ;
- en grands caractères italiques les ensembles, comme $\mathcal{E}$ l'ensemble des nœuds ;
- en caractères non-gras et non-italiques, grands ou petits, les scalaires, comme les valeurs propres λ_i, ou les fonctions comme le filtre d'ondelettes g.

1 Une communauté dans un graphe

1.1 Qu'est-ce qu'une communauté ?

Une communauté est un ensemble de nœuds plus connectés entre eux qu'avec le reste du graphe. Cette définition, nécessairement floue, a l'avantage d'être assez générale pour correspondre à une majorité des définitions, souvent plus précises, que donne chaque auteur travaillant sur le sujet. À mon sens, la notion même de communautés ne peut être envisagée sans le but que l'on se donne, sans savoir pourquoi on cherche des communautés. Il me semble que chercher des communautés dans un graphe, c'est chercher à regrouper assez de nœuds ensemble pour permettre la lecture et l'analyse du graphe, mais ne pas trop les regrouper pour garder au moins une partie des détails structurels du graphe. Il faudra donc garder en tête que la recherche en communautés est un équilibre à trouver entre trop de détails rendant le graphe difficilement analysable et trop d'agrégation qui fait éventuellement disparaître l'information pertinente du graphe.

Cette recherche de communautés est donc naturellement spécifique au contexte du graphe étudié, à la finesse de l'analyse souhaitée, même à l'expérimentateur qui devra choisir parmi toutes les méthodes de recherche de communautés et donc introduire un biais à ce niveau là. Prenons un exemple classique et simple pour illustrer ce propos : le graphe social d'un club de karaté [229] observé sur une période de 3 ans dans les années 1970 dans le cadre d'une étude sur l'apparition de clivages dans un réseau social. Le graphe est binaire et composé de 34 nœuds, chacun correspondant à un membre du club, et un lien existe entre deux nœuds si les deux personnes ont des interactions sociales en dehors du club de karaté. Ce graphe est représenté sur la Fig. 2.1a. Considérons le nœud colorié de la Fig. 2.1b. Dans quelle communauté appartient-il ? Quelque part, chacune des 4 propositions (Figs. 2.1c-f)

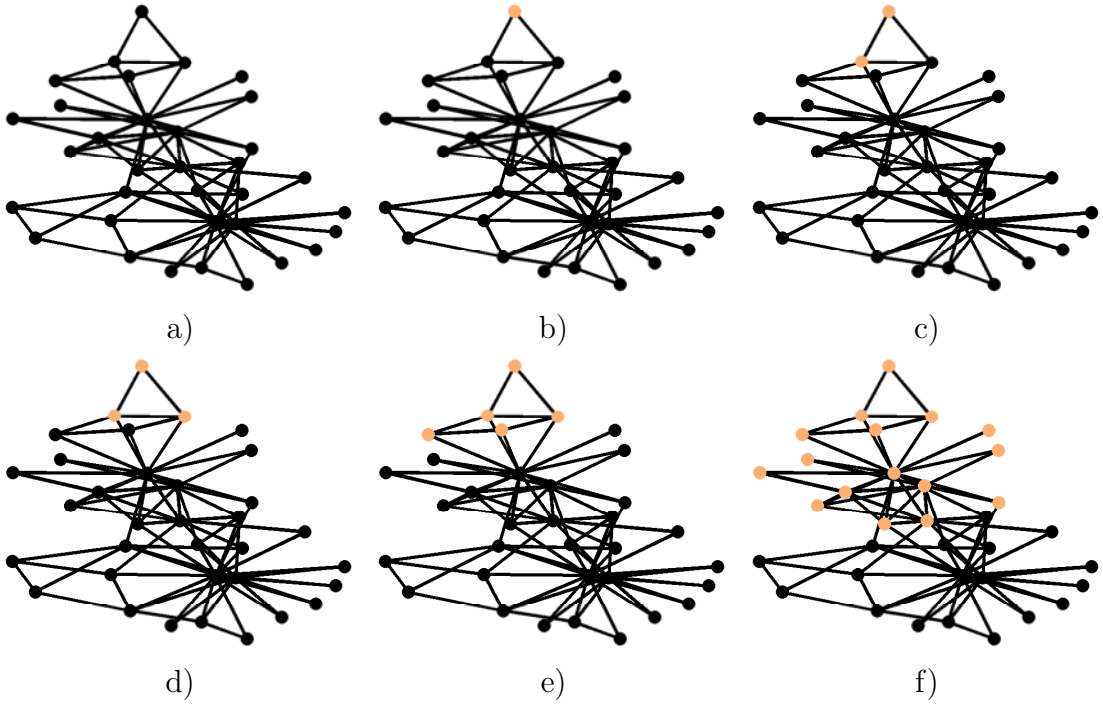


Figure 2.1: a) Graphe du club de karaté de Zachary [229] : chaque nœud représente un des membres du club et un lien existe entre deux nœuds si les deux membres ont des interactions sociales en dehors du club. Considérons le nœud colorié de l'illustration b). Dans quelle communauté appartient-il ? Les nœuds coloriés des illustrations c-f) sont autant de réponses possibles.

de communautés (représentées par les nœuds coloriés sur les figures) correspondent à une interprétation possible de la définition de communauté donnée au début de cette partie. Et aucune de ces propositions n'est mieux que l'autre : les propositions c et d sont justifiées parce que ce sont des cliques, c'est-à-dire des groupes de nœuds tous connectés les uns aux autres ; ce n'est pas le cas pour la proposition e mais celle-ci regroupe 5 nœuds qui sont tous connectés au reste du graphe par l'intermédiaire d'un seul nœud ; et la proposition f coupe le graphe en deux parties plus ou moins égales et peu connectées entre elles. C'est d'ailleurs cette dernière proposition qui correspond à la vérité de terrain discutée dans [229] : ce réseau social s'est finalement clivé en deux (selon les couleurs de f) et les deux sous-réseaux se sont finalement retrouvés dans deux clubs de karaté différents. Chacune des propositions de communautés est justifiable et justifiée. Et c'est uniquement l'étude de terrain et donc le contexte du graphe qui nous permet de savoir quelle acceptation du sens de communauté est pertinente.

1.2 De l'utilité de la recherche de communautés

À la lecture du cas précédent, on est en droit de questionner l'utilité de la recherche de communautés : définition floue, multiplicité des solutions, nécessité de connaître le contexte du graphe pour savoir quel sens donner à la notion de communauté... En fait, pour se convaincre de son utilité, il faut comprendre la recherche

en communautés comme une prospection, qui ne donnera jamais dans les graphes réels une réponse claire et définitive, mais plutôt un ensemble de réponses, autant d'hypothèses plausibles qu'il faudra ensuite étudier en détail avec les outils spécifiques au domaine d'application. Dans ce sens, la recherche de communautés est une méthode de fouille de données, où la donnée est un graphe.

1.3 Une brève histoire de la recherche de communautés

C'est en sociologie qu'est né le domaine de la recherche de communautés, même si aujourd'hui le domaine d'application est bien plus vaste. En effet, quelques travaux précurseurs de sociologues [54, 225, 175, 103] cherchent à grouper les humains en fonction de leurs opinions politiques, de leurs interactions sociales, etc... Dans les années 1960/1970, un groupe de chercheurs à la frontière entre les mathématiques et l'informatique a attaqué le problème de partitionnement de graphes [123, 82], essayant de formaliser le concept de communauté et développer des algorithmes de détection automatique, proposant par exemple le théorème flot-max/coupe-min (*mincut/maxflow* en anglais) qui stipule que le flot maximum pouvant aller d'un nœud s à un nœud t est égal à la somme minimale des poids des liens à retirer du graphe pour séparer s de t. Un des développements de ce groupe de chercheurs est le partitionnement spectral, initié par Donath et Hoffman [59] et poursuivi et amélioré depuis lors (voir le tutoriel de von Luxburg [221] pour une présentation plus générale du partitionnement spectral). Dans les années 2000, Girvan et Newman [89], plutôt ancrés en physique statistique sur réseaux, proposent une mesure de qualité d'une partition donnée appelée modularité. Cette mesure, couplée à des algorithmes très rapides comme celui de Louvain [32] et à des données de plus en plus accessibles et nombreuses, ont fait exploser le nombre de travaux sur la recherche de communautés. Aujourd'hui, ce domaine est à la frontière de l'informatique, des mathématiques, de la physique statistique, et de la science des réseaux. L'étendue des travaux dans ce domaine pourra être mesurée dans la revue de Fortunato [83].

1.4 Partitionnement d'un graphe en communautés

Le problème classique dans le domaine de la détection de communautés dans un graphe, est de trouver une partition de l'ensemble des nœuds $\mathcal{V}$ qui sépare le mieux le graphe en communautés. Par définition, une telle partition, notée P , est un ensemble de communautés deux à deux disjointes, dont la réunion équivaut à $\mathcal{V}$:

$$P = \{C_i\} \text{ tq } \forall (i,j) \quad C_i \cap C_j = \emptyset \text{ et } \bigcup_i C_i = \mathcal{V}. \tag{2.1}$$

Des améliorations peuvent être apportées à l'énoncé de ce problème, par exemple en autorisant les communautés à être recouvrantes, c'est-à-dire qu'un même nœud peut appartenir à plusieurs communautés. Le recouvrement existe bel et bien dans de nombreuses applications, mais les algorithmes sont souvent d'abord présentés sur le cas "simple" sans recouvrement avant d'être généralisés au cas avec recouvrement. Étant donné que nous allons présenter un nouvel algorithme dans la suite, nous examinerons ici uniquement des communautés non-recouvrantes et laisserons le recouvrement pour des travaux futurs.

2 Partitionner un graphe en communautés

Nous référons le lecteur à la revue de Fortunato [83] qui recense et synthétise une grande partie de l'imposant état de l'art dans le domaine de la détection de communautés (recouvrantes ou non). Nous évoquerons ici uniquement quelques éléments utiles à la compréhension de la suite.

Tout d'abord, posons le problème de la détection d'une partition d'un graphe en communautés. Nous nous cantonnons ici aux cas plus simples de graphes non-dirigés et de la recherche de communautés non-recouvrantes, c'est-à-dire d'une partition du graphe au sens stricte du terme. Nous nous posons le problème du "meilleur découpage" d'un graphe en communautés disjointes, et il se décompose, dans de nombreux travaux sur la question, en deux temps :

1. définir ce qu'est le "meilleur découpage" d'un graphe en communautés, c'est-à-dire proposer une mesure de la qualité d'une partition donnée. Nous allons présenter dans la partie 2.1 la mesure la plus connue appelée modularité.
2. maximiser cette mesure sur l'ensemble des partitions possibles. En pratique, cela est impossible étant donné qu'il existe beaucoup $(\frac{1}{N+1}\binom{2N}{N})$ de partitions possibles d'un graphe à N nœuds et que le problème de maximiser une mesure comme la modularité sur cet ensemble a été prouvé NP-complet [36] : on doit se résoudre à chercher une approximation de la solution par diverses heuristiques. Nous présentons dans la suite deux heuristiques classiques : l'algorithme de Louvain dans la partie 2.2, et l'algorithme spectral de Donetti dans la partie 2.4. Pour pouvoir détailler ce dernier, il faudra d'abord introduire le lecteur aux techniques de clustering hiérarchique et aux dendrogrammes, ce que nous ferons dans la partie 2.3.

2.1 La modularité : une mesure de la qualité d'une partition

Traduire la définition qualitative de communautés en équation quantitative est un des défis à surmonter et a été à l'origine de nombreux travaux. Parmi ceux-ci, une mesure a eu beaucoup de succès ces dix dernières années : la modularité [162]. Notée Q, c'est une mesure qui, pour une partition donnée $\{C_i\}$, quantifie à quel point la partition sépare $\mathcal{V}$ en "bonnes" communautés :

$$Q(\{C_i\}) = \frac{1}{2\omega} \sum_{i\in\mathcal{V}, j\in\mathcal{V}} \left[\boldsymbol{W}_{ij} - \frac{s_i s_j}{2\omega} \right] \delta(C_i, C_j), \tag{2.2}$$

où $\boldsymbol{W}$ est la matrice d'adjacence du graphe, s_i la force du nœud i, $\delta(C_i, C_j) = 1$ si i et j sont dans la même communauté, $= 0$ sinon, et $\omega = \frac{1}{2}\sum_{i\in\mathcal{V}, j\in\mathcal{V}} \boldsymbol{W}_{ij}$. En fait, la modularité compare la quantité totale d'interactions intra-communautés, à la quantité à laquelle on s'attendrait si le graphe était aléatoire (au sens du modèle aléatoire de Chung-Lu décrit dans la partie 1.6 du Chapitre 1). La modularité d'une partition en K communautés est comprise entre $-(1-1/K)$ et $1-1/K$ [220]. Ainsi, à chaque partition du graphe est associée une mesure de qualité de son caractère communautaire.

2.2 L'algorithme de Louvain

Nous présentons l'algorithme de Louvain, appelé ainsi parce qu'il a été développé par une équipe de l'Université de Louvain [32], d'une part parce qu'il est incontournable dans le domaine, mais aussi parce que nous en ferons référence par la suite. C'est un algorithme glouton qui itère deux étapes.
Tout d'abord, à chaque nœud est associée une communauté différente : l'algorithme commence donc avec N communautés. Ensuite, pour chaque nœud i, on considère chacun de ses voisins j et on évalue le gain en modularité correspondant à enlever i de sa communauté pour le mettre dans la communauté de j. On place ensuite le nœud i dans la communauté voisine qui fait le plus augmenter la modularité. Si on ne peut déplacer i dans aucune communauté voisine parce que de telles actions feraient baisser la modularité, on laisse i dans sa communauté. On itère ce processus sur tous les nœuds, et on le répète jusqu'à qu'il n'y ait plus d'augmentation possible de la modularité (notons que chaque nœud est considéré plusieurs fois). La première étape est alors terminée.
La deuxième étape consiste à créer un nouveau réseau pour lequel les nœuds sont les communautés ainsi trouvées. Les liens entre deux nœuds sont alors la somme des poids des liens liant les deux communautés. Chaque nœud du nouveau graphe est aussi lié à lui-même avec un poids qui correspond à la somme des poids de tous les liens internes à cette communauté. Cela termine la deuxième étape de l'algorithme. Un "passage" de l'algorithme est la combinaison de ces deux étapes. On itère ces passages jusqu'à que la modularité maximale trouvée à la fin de la première étape ne soit pas supérieure à la modularité trouvée au passage d'avant. Le résultat est une partition P et sa modularité $Q(P)$.
Notons que la solution trouvée par l'algorithme de Louvain dépend de son initialisation (l'ordre de la suite des nœuds considérée) : lancer l'agorithme plusieurs fois donne en général autant de solutions possibles, toutes des approximations de la solution recherchée.

Le succès de l'algorithme de Louvain s'explique grâce à la structure souvent hiérarchique des graphes complexes auxquels nous avons affaire. En effet, sans cette structure hiérarchique, la deuxième étape de l'algorithme ne serait pas justifiée. La deuxième raison de son succès est sa rapidité de calcul qui lui permet de traiter rapidement des graphes à plusieurs millions de nœuds.

2.3 Clusterings hiérarchiques et dendrogrammes

Parmi l'ensemble des méthodes de classification supervisées ou non qui existent dans la littérature [101], nous rappelons uniquement le fonctionnement du clustering hiérarchique pour deux raisons : l'algorithme spectral de Donetti présenté dans la partie suivante, ainsi que la méthode introduite dans cette thèse (parties 4 à 7 de ce chapitre) feront appel à ce clustering particulier. Le clustering hiérarchique (voir Section 5.1 de [116] ou [117]) est une technique qui permet à partir d'un ensemble d'objets caractérisés par des vecteurs (qui portent le nom de *vecteurs caractéristiques*) de taille n, de séparer les objets en groupes d'objets. Nous verrons par la suite que nous pouvons utiliser des méthodes de clusterings pour chercher

des communautés dans un graphe à partir du moment où nous pouvons associer à chaque nœud un vecteur caractéristique en fonction de sa position dans le graphe. Les techniques de clustering ont été développées dans le domaine de la fouille de données. En pratique, la donnée du problème consiste en N objets, dont chaque objet i est caractérisé par un vecteur noté f_i. A partir de cette famille de N vecteurs, on peut calculer une matrice de distance dont l'élément (i, j) est la distance entre les nœuds i et j. La distance est à choisir et dépend du problème que l'on considère : euclidienne, angulaire, la distance de corrélation, la norme p, ... On applique à cette matrice de distance l'algorithme de clustering hiérarchique qui suit. Commençons avec N groupes, avec un nœud par groupe. Les deux groupes les plus proches sont concaténés, et ce, de façon itérative jusqu'à ce qu'il n'existe plus qu'un seul groupe. Lors d'une itération, pour calculer la proximité de deux groupes, on peut utiliser différentes définitions qui vont donner chacune un algorithme différent :

- toutes les paires de nœuds possibles, en prenant un nœud dans chaque groupe, sont considérées. La distance minimale trouvée parmi toutes ces paires définit la proximité des deux groupes. Cela définit la méthode de chaînage simple (*single-linkage clustering*).
- au lieu de prendre la distance minimale parmi toutes les paires de nœuds on définit la proximité entre deux groupes comme la distance maximale. Cela définit la méthode de chaînage complet (*complete-linkage clustering*).
- une autre possibilité est de prendre la distance moyenne entre les paires de nœuds. Cela définit la méthode de chaînage moyenné (*average-linkage clustering*).
- encore une autre possibilité est de représenter chaque groupe par son barycentre dans l'espace à n dimensions dans lequel les objets sont définis. La proximité entre deux groupes est alors la distance entre ses barycentres. Cela définit la méthode de chaînage barycentrique (*centroid-linkage clustering*).

On représente cette suite de concaténations par un diagramme appelé dendrogramme. Un exemple de dendrogramme est présenté sur la Fig. 2.2a. L'axe des abscisses représente les différents objets que l'on cherche à regrouper (ici 14 objets numérotés de 1 à 14, habituellement ordonnés dans l'ordre de regroupement, afin de bien mettre en évidence la structure en arbre du dendrogramme). L'axe des ordonnées représente la distance à laquelle les groupes ont été concaténés. Les groupes se regroupent quand on augmente la distance jusqu'à en obtenir un seul (ici à partir de la distance 1.5). Afin d'obtenir une partition, l'utilisateur coupe le dendrogramme horizontalement à une distance de son choix, comme dans les exemples de la Fig. 2.2b (coupé à une distance de 1, on obtient une partition en deux groupes) et de la Fig. 2.2c (coupé à une distance de 0.5, on obtient une partition en cinq groupes dont un singleton).

En résumé, l'algorithme de clustering hiérarchique ordonne les objets de manière hiérarchique à l'aide d'un dendrogramme. En revanche, pour obtenir une partition, l'utilisateur doit faire un choix (souvent difficile) pour savoir où couper le dendrogramme. Pour cela, l'utilisateur peut s'appuyer sur plusieurs outils :

- citons le critère d'information d'Akaike [21] (AIC) qui cherche à trouver un modèle le plus vraisemblant possible tout en pénalisant le nombre de para-

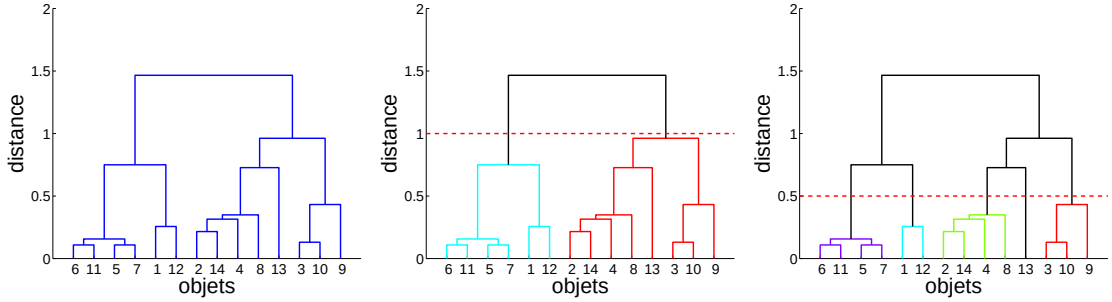


FIGURE 2.2: Exemple de dendrogramme où 14 objets sont hiérarchisés. Pour obtenir une partition des objets à partir d'un dendrogramme, il faut faire une coupe horizontale. L'illustration b) (resp. c) représente une coupe (pointillés rouges horizontaux) possible donnant une partition en deux (resp. cinq) groupes.

mètres pour satisfaire le critère de parcimonie, ou d'autres critères inspirés de celui-ci comme le critère d'information bayésien [188] (BIC).
- un autre outil classique est la validation croisée où l'ensemble des données est séparé en plusieurs sous-ensembles (pas forcément disjoints) et on mesure la consistance du résultat sur tous les sous-ensembles (voir par exemple le chapitre 7.10 de [101]).
- une méthode appelée *Gap Statistics* [206] a été proposée par Tibshirani et al. permet d'estimer le nombre de clusters en confrontant les données à une hypothèse nulle soigneusement choisie.
- une autre possibilité est de maximiser une fonction de qualité sur l'ensemble des coupes possibles du dendrogramme, fonction qui va dépendre de la spécificité des données. Ce sera par exemple le cas de l'algorithme spectral discuté ci-dessous.

2.4 Un algorithme spectral

Le partitionnement spectral regroupe toutes les méthodes de partitionnement en communautés qui font appel aux valeurs propres de la matrice d'adjacence du graphe, ou d'une de ses matrices dérivées comme le laplacien. Cette idée date déjà un peu [59] et a connu de nombreux raffinements au fur et à mesure des années [199, 221]. Il existe de nombreux liens entre les spectres de la matrice d'adjacence ou de la matrice laplacienne et les propriétés du graphe. Par exemple, la multiplicité de la valeur propre nulle du laplacien est égal au nombre de composantes connexes (une composante connexe est un sous-graphe maximal connecté) du graphe. Par extension, la valeur de la deuxième valeur propre λ_2 nous indique à quel point le graphe peut être approximé par deux composantes connexes : plus λ_2 est proche de zéro et plus le graphe est scindé en deux. De plus, Fiedler [78] montre que le deuxième vecteur propre $\boldsymbol{\chi}_2$ coupe le graphe en deux : d'un côté les nœuds dont les composantes de $\boldsymbol{\chi}_2$ sont positives et de l'autre les nœuds dont ces composantes sont négatives. Le lecteur pourra retrouver ces résultats – et bien d'autres – dans

deux livres dédiés à la théorie spectrale des graphes complexes : celui de Van Mieghem [220] et celui de Chung et Lu [49].

Dans ce cadre, nous présentons ici l'algorithme de Donetti [60], qui utilise les idées du partitionnement spectral pour optimiser la modularité : l'intuition sous-jacente au partitionnement spectral est que les nœuds "proches" dans le réseau ont des composantes de vecteurs propres similaires. En pratique dans cet algorithme, les nœuds du graphe sont considérés comme des points dans un espace à D dimensions, dont les coordonnées sont les composantes des D premiers vecteurs propres non triviaux du laplacien. C'est-à-dire, à chaque nœud i on associe le vecteur caractéristique suivant :

$$\boldsymbol{f_i} = [\boldsymbol{\chi}_2(i)|\boldsymbol{\chi}_3(i)|\cdots|\boldsymbol{\chi}_D(i)]^\top . \tag{2.3}$$

Les auteurs utilisent ensuite une distance (euclidienne, angulaire, ...) pour obtenir une matrice de distance $N \times N$ qui dépend simplement de D. Ils utilisent ensuite des techniques de clustering hiérarchique pour grouper les nœuds en fonction de leur distance relative et ainsi obtenir un dendrogramme (voir la partie 2.3 pour les détails sur ces techniques et la définition d'un dendrogramme). Ce qui rend cet algorithme original par rapport aux algorithmes spectraux usuels est la manière dont les auteurs choisissent de couper le dendrogramme : pour obtenir une partition du graphe en communautés, les auteurs cherchent la coupe horizontale du dendrogramme qui maximise la modularité : ils obtiennent ainsi la partition P_D de modularité Q_D. Le nombre D de vecteurs propres à prendre en compte n'est *a priori* pas connu : l'algorithme est donc répété pour un D de plus en plus grand, jusqu'à ce que la modularité optimale Q_D commence à décroître. La solution retenue est alors la partition qui maximise la modularité :

$$Q_{D^*} = \max(Q_D) \quad \text{et} \quad P = P_{D^*}. \tag{2.4}$$

Les auteurs montrent dans [61], sans pouvoir l'expliquer, que les résultats sont meilleurs en utilisant la version normalisée du laplacien qu'en utilisant le laplacien combinatoire.

3 État de l'art des méthodes de détection multiéchelle

Il existe de nombreuses autres méthodes qui permettent de trouver une partition en communautés dans un graphe [83] que nous n'allons pas rappeler ici, ce serait au-delà des prétentions de cette thèse. Mais nous pouvons dire que la grande majorité d'entre elles donnent une unique solution : une seule partition qui maximise une mesure de qualité de la partition, que ce soit la modularité ou une autre. L'unicité de la solution trouvée a à la fois ses avantages en termes de simplicité de l'analyse et de rapidité de temps de calcul, mais aussi les inconvénients de représenter que partiellement une réalité comunautaire qui peut être beaucoup plus complexe. Prenons l'exemple du graphe jouet de la Fig. 2.3 : c'est un graphe de 128 nœuds qui a été créé avec quatre échelles intrinsèques. Les quatre partitions associées à ces

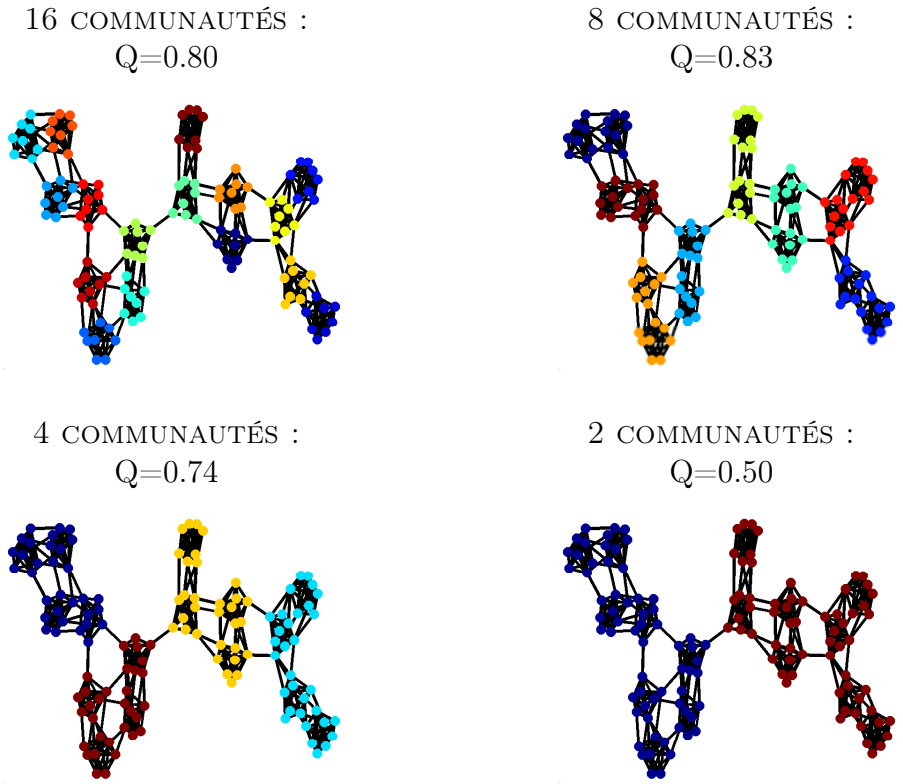


FIGURE 2.3: Graphe jouet de 128 nœuds créé avec quatre échelles intrinsèques : il peut être naturellement découpé en seize, huit, quatre et deux communautés. Sur chaque graphe, deux nœuds sont de la même couleur si ils sont dans la même communauté. Malgré le fait que chacune de ces partitions est juste, on voit que chercher à maximiser la modularité revient à choisir uniquement la partition à huit communautés. Ce genre de cas nécessite une méthode multiéchelle.

échelles sont représentées avec des couleurs sur la figure (deux nœuds avec la même couleur sont dans la même communauté) : il y a une partition à seize communautés, à huit communautés, à quatre communautés, et à deux communautés. Chacune de ces partitions a sa raison d'être : le graphe a été spécialement créé pour cela ! Et pourtant, les algorithmes classiques vont devoir choisir une seule partition parmi ces quatre. La maximisation de la modularité, par exemple, trouvera typiquement une solution à une échelle "moyenne" : ici la solution à 8 communautés.

C'est pourquoi certains auteurs ont préféré développer des méthodes dites multiéchelles, qui vont proposer une solution par échelle d'analyse, en introduisant une notion d'échelle de différentes manières. Dans la suite, nous évoquerons tout d'abord dans la partie 3.1 quelques limites de la modularité, puis nous montrerons dans la partie 3.2 l'intérêt et parfois la nécessité de faire appel à des méthodes multiéchelles. Ce qui nous permettra dans la partie 3.3 de poser proprement le double problème de détection multiéchelle de communautés : d'abord définir une notion d'échelle pour un graphe et trouver une partition par échelle dont nous détaillerons quelques méthodes existantes dans la partie 3.4 ; puis dégager le(s) échelle(s) pertinente(s) pour

savoir quelle(s) partition(s) retenir, dont quelques idées existantes sont présentées dans la partie 3.5.

3.1 Les limites de la modularité

La modularité a deux principales limites : une limite de résolution (évoquée en premier dans [84]) qui l'empêche de détecter des groupes de nœuds de petite taille, et une limite due au plateau très irrégulier qu'elle présente autour de son maximum global : il existe souvent de très nombreux maxima locaux, proches du maximum global, et qui pourtant correspondent à des partitions très différentes.
Pour illustrer la limite de résolution, on peut considérer, comme dans [92], un graphe composé de k cliques de c nœuds reliées entre elles en cercle. On représente un tel graphe sur la Fig. 2.4a avec $k = 18$ et $c = 4$. La partition qu'on aimerait obtenir (et qui correspond à la manière dont on a construit le graphe) sépare toutes les cliques entre elles (Fig. 2.4b). Or on observe dans le cas présenté que la modularité qui correspond à cette partition $Q = 0.802$ est inférieure à la modularité qui correspond à la partition qui agrège les cliques deux à deux (Fig. 2.4c). On montre [92] que la modularité va préférer agréger les cliques entre elles à partir d'un nombre critique k^* de cliques :

$$k^* = \binom{c}{2} + 2. \tag{2.5}$$

Par de tels arguments, on montre que la maximisation de la modularité favorise une échelle intrinsèque, au détriment de petites communautés qui ne pourront pas être détectées.

Quant à la dégénérescence du maximum global de la modularité (c'est-à-dire l'existence de très nombreux maxima locaux très proches du maximum global qui correspondent à des partitions parfois très différentes), ce défaut provient plus de la volonté d'obtenir une unique partition plus que de la modularité en tant que telle. En effet, l'existence de nombreux maxima locaux est la signature de la multiplicité et la complexité des structures modulaires du réseau. Quelque part, tous les nœuds sont "en compétition" pour que leur entourage propre soit considéré comme une seule communauté (dans le cadre de communautés non recouvrantes formant une partition). Très souvent, donner une seule partition signifie privilégier une solution parmi beaucoup d'autres tout autant intéressantes et qui peuvent être très différentes (voir [92] pour plus de détails).

3.2 L'intérêt d'une vision multiéchelle

Pour parer à la première limite évoquée (celle de résolution), il est souvent nécessaire de faire appel à des algorithmes qui vont donner plusieurs solutions de partition. Une manière de procéder est de définir une notion d'échelle dans le graphe, et de chercher, à chaque échelle donnée, la partition qui sépare le mieux le graphe en communautés. De cette manière, nous serons sûrs de ne pas passer à côté de petites communautés non détectables autrement, et proposer ainsi une description plus complète des sous-structures du graphe. Pour parer à la seconde limite, il faut nécessairement associer à une méthode multiéchelle une mesure de stabilité qui va

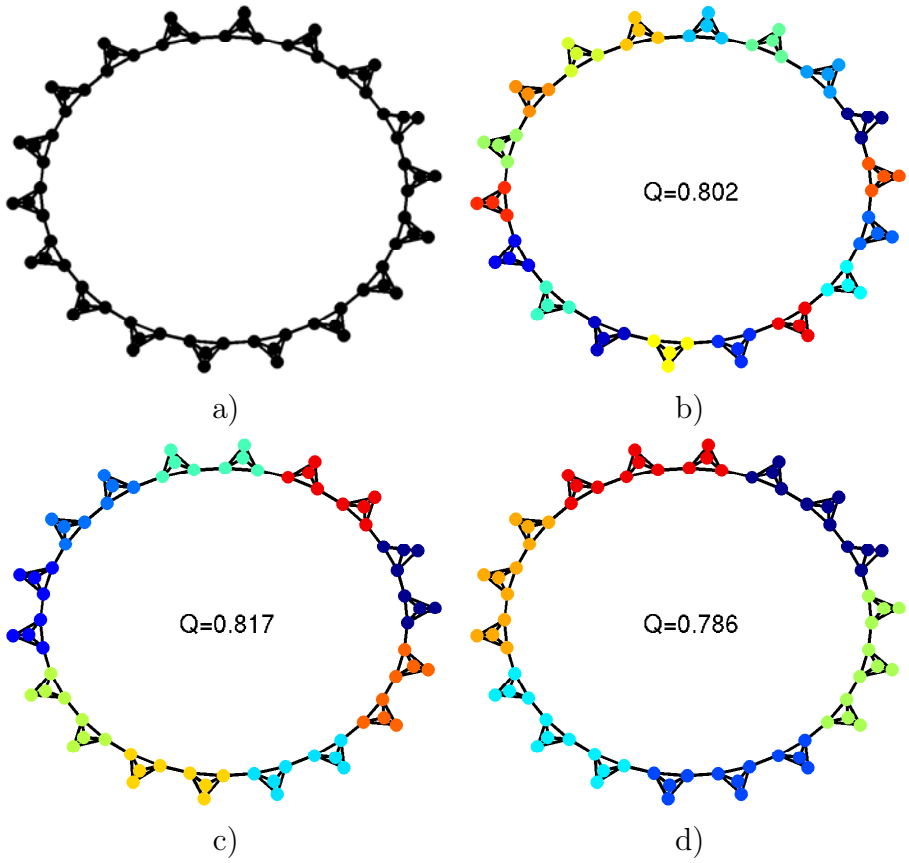


FIGURE 2.4: a) Graphe de 18 cliques de 4 nœuds reliées en cercle. La partition qu'on aimerait obtenir est illustrée en b). Or, la partition qui obtient le meilleur score de modularité est la solution c) qui agrège les cliques deux-à-deux : la modularité favorise une certaine échelle intrinsèque.

permettre à l'utilisateur de choisir les échelles spécifiques pour lesquelles les partitions sont pertinentes.

3.3 Poser le problème multiéchelle

Nous posons à présent le problème de détection multiéchelle de communautés, qui se décompose comme suit, pour la majorité des travaux existants (Chapitre 12 de [83]) :

1. Définir une notion d'échelle pertinente. Cette échelle, qu'on notera de manière générique s, est continue et définie sur un intervalle réel.
2. Discrétiser s en M échelles discrètes $\mathcal{S} = \{s_i\}_{i \in [1,M]}$.
3. Proposer une mesure de qualité dépendante de l'échelle et optimiser cette mesure pour chaque échelle s_i, afin d'obtenir la partition en communautés à cette échelle P_{s_i}. On obtient ainsi l'ensemble de partitions $\mathcal{P} = \{P_{s_i}\}_{s_i \in \mathcal{S}}$.
4. Détecter, parmi les M échelles proposées, lesquelles sont vraiment pertinentes. Pour cela, proposer une mesure de pertinence de chaque échelle.

Dans la partie 3.4, nous présenterons l'état de l'art concernant les points 1 et 3 du problème. Puis, dans la partie 3.5, nous détaillerons certaines mesures de pertinence qui peuvent répondre au point 4 du problème. Le point 2 est un point peu discuté dans la littérature et il y a souvent une part d'arbitraire à la fois au niveau des bornes du paramètre d'échelle et de la façon dont les échelles sont discrétisées sur cet intervalle. Nous verrons dans la partie 4.2 que notre méthode permet de calculer automatiquement les bornes du paramètre d'échelle qui seront pertinentes.

3.4 Trouver une partition par échelle : méthodes existantes

3.4.1 Inspirée de la physique statistique : la méthode de Reichardt et Bornholdt

Reichardt et Bornholdt [173] proposent une méthode multiéchelle qui se base sur une analogie entre la détection en communautés, et le modèle de Potts en physique statistique, un modèle d'interactions entre spins. Nous présentons ce travail dans sa version initiale qui considère uniquement des matrices binaires. Le modèle de Potts, qui est une généralisation du modèle d'Ising, associe à chaque spin i (i.e. chaque nœud i) une valeur de spin σ_i qui correspond dans l'analogie à la communauté du nœud i. Le principe du modèle veut que les spins adjacents aient le plus possible des valeurs similaires, et les spins non-adjacents aient le plus possible des valeurs différentes. Pour mettre cela en équation, le modèle pénalise (resp. favorise) les liens connectant des spins de valeurs différentes (resp. égales) et les non-liens entre des spins de valeurs égales (resp. différentes). L'état physique d'un tel système est caractérisé par la liste $\{\sigma\}$ de tous les σ_i (i.e. une partition) , et son énergie vaut :

$$\mathcal{H}_\gamma(\{\sigma\}) = -\sum_{i<j}(\boldsymbol{A}_{ij} - \gamma p_{ij})\delta(\sigma_i, \sigma_j), \tag{2.6}$$

où $\boldsymbol{A}_{ij}$ est l'élément (i, j) de la matrice d'adjacence, $\gamma \in \mathbb{R}^+$ et p_{ij} la probabilité d'avoir un lien entre les nœuds i et j pour un modèle nul au choix. Trouver l'état $\{\sigma\}$ qui minimise cette énergie revient à chercher des communautés dans le graphe. C'est p_{ij} qui contient l'information du modèle aléatoire auquel on veut confronter l'existence de structures en communautés. Par exemple, $p_{ij} = p$ si on veut le confronter au modèle d'Erdös-Rényi. Mais, plus juste pour les graphes complexes usuels, les auteurs utilisent le modèle de Chung-Lu : $p_{ij} = d_i d_j / 2d_{tot}$ qui impose une distribution réaliste des degrés. Dans ce cas, notons que pour $\gamma = 1$ on retrouve à un facteur multiplicatif près l'expression de la modularité. En ce sens, on peut voir $\mathcal{H}_\gamma(\{\sigma\})$ comme une modularité généralisée paramétrée par la donnée d'un modèle aléatoire de graphe et de γ. En réalité, γ joue le rôle de paramètre d'échelle : on peut passer de la solution à une communauté contenant tous les nœuds pour $\gamma = 0$ à la solution à N communautés chacune contenant un seul nœud pour $\gamma \longrightarrow \infty$. Les auteurs utilisent une forme d'algorithme de recuit simulé pour minimiser $\mathcal{H}_\gamma(\{\sigma\})$, et trouver ainsi une solution à chaque échelle. Une extension de ce modèle aux graphes pondérés est proposée dans [102], et aux graphes dirigés dans [208].

3.4.2 Ajout de boucles : la méthode d'Arenas

Arenas et al. [22] proposent une méthode basée sur l'ajout de boucles (liens liant les nœuds à eux-mêmes) de poids r. C'est-à-dire que la matrice d'adjacence pondérée $\boldsymbol{W}$ est remplacée par $\boldsymbol{W_r} = \boldsymbol{W} + r\boldsymbol{I}$ où $\boldsymbol{I}$ est la matrice identité. La modularité $Q_r = Q(\boldsymbol{W_r})$ est donc aussi paramétrée par r, qui joue ici le rôle de paramètre d'échelle. Ce paramètre $r \in [2\omega/N, +\infty[$, en augmentant, va permettre de scanner les échelles les plus petites, et en diminuant, les échelles les plus grandes. Les auteurs utilisent une version de l'algorithme de recherche tabou [90] pour maximiser cette modularité paramétrée, et trouver ainsi une partition à chaque échelle.

3.4.3 Marcheurs aléatoires : la méthode de Delvenne

Delvenne et al. [58] proposent une méthode basée sur des marcheurs aléatoires. Ils se basent sur une interprétation de la matrice $\boldsymbol{W}$ en terme de matrice de transition d'une chaîne de Markov. Pour être précis, $\boldsymbol{W}$ n'est pas exactement une matrice de transition car elle n'est pas normée à 1. On définit la matrice $\boldsymbol{M} = \boldsymbol{S}^{-1}\boldsymbol{W}$ où nous rappelons que $\boldsymbol{S}$ est la matrice diagonale des forces des nœuds. Ceci assure à $\boldsymbol{M}$d'avoir ses lignes qui somment à 1, ce qui permet d'en faire une interprétation probabiliste. La diffusion d'un marcheur aléatoire tend vers la distribution stationnaire $\boldsymbol{\pi} = \boldsymbol{s}/2\omega$ où nous rappelons que $\boldsymbol{s}$ est le vecteur des forces, et 2ω la somme totale des forces. En mettant ce vecteur sur la diagonale d'une matrice nulle, ou obtient $\boldsymbol{\Pi} = \mathrm{diag}(\boldsymbol{\pi})$. Notons qu'une partition dont on cherche à mesurer la qualité peut se coder sous la forme d'une matrice $\boldsymbol{H}$ de taille $N \times K$ où K est le nombre de partitions. $H_{ij} = 1$ si le nœud i appartient à la communauté j, $= 0$ sinon. Forts de ces notations, les auteurs proposent d'étudier la matrice $\boldsymbol{R_t}$ de taille $K \times K$ ci-dessous :

$$\boldsymbol{R_t} = \boldsymbol{H}^\top(\boldsymbol{\Pi}\boldsymbol{M}^t - \boldsymbol{\pi}^\top\boldsymbol{\pi})\boldsymbol{H}. \tag{2.7}$$

$R_t(i,j)$ correspond à la probabilité, pour un marcheur aléatoire, de commencer dans la communauté i et de finir dans la communauté j après t pas de temps, à laquelle on retranche la probabilité que deux marcheurs indépendants soient dans i et j dans l'état stationnaire. Ainsi, une bonne partition est une partition stable vis-à-vis d'un marcheur aléatoire, et est caractérisée par des termes diagonaux élevés de $\boldsymbol{R_t}$. Les auteurs définissent la stabilité d'une partition $\boldsymbol{H}$ à un temps t :

$$r(t, \boldsymbol{H}) = \min_{0 \leq s \leq t} \mathrm{trace}(\boldsymbol{R_s}). \tag{2.8}$$

Cette stabilité est donc une fonction décroissante du temps. En effet, une partition a de fortes chances d'être stable à temps petit car les marcheurs aléatoires n'ont pas encore eu le temps de diffuser sur le graphe. Les auteurs généralisent la définition de stabilité pour un paramètre de temps t continu défini sur $\mathbb{R}^+$. Ce paramètre t joue ici le rôle de paramètre d'échelle. À une échelle t, rechercher une partition qui découpe le graphe en communautés devient un problème de maximisation de $r(t, \boldsymbol{H})$ sur l'espace des partitions $\boldsymbol{H}$. Pour $t = 1$, on peut montrer que maximiser $r(1, \boldsymbol{H})$ revient à maximiser la modularité. De plus, les auteurs montrent qu'une version linéarisée de $r(t, \boldsymbol{H})$ permet de retrouver les cas de Reichardt et d'Arenas. Pour maximiser $r(t; \boldsymbol{H})$, Le Martelot et al. [133] ont proposé un algorithme glouton similaire à celui de Louvain, adapté à l'optimisation de la stabilité.

3.4.4 Autres méthodes

Rosvall et Bergstrom, dans [181], généralisent au cas multiéchelle leur algorithme `infomap` [180], qui fonctionne de la manière suivante. La fonction de qualité que les auteurs cherchent à optimiser est l'efficacité théorique maximale avec laquelle on peut coder des marches aléatoires sur le graphe. En effet, si un graphe présente des structures en communautés fortes, les marches aléatoires sur ce graphe vont régulièrement boucler au sein des communautés, rendant leur codage optimal facile et court. Les auteurs se basent donc sur des outils entropiques de théorie de l'information. Pour optimiser cette fonction de qualité sur l'ensemble des partitions, les auteurs utilisent un algorithme rapide glouton [179] similaire à celui de Louvain. Une des limites de la version multiéchelle de cet algorithme est qu'il est nécessairement hiérarchique (i.e. il ne peut trouver que des partitions exactement imbriquées les unes dans les autres).

Lancichinetti et al. [129] proposent des solutions qui s'appliquent surtout à la recherche de communautés recouvrantes. Les auteurs proposent une mesure de la qualité d'une communauté C (et non d'une partition), à un paramètre d'échelle α :

$$f_C^\alpha = \frac{s_{in}^C}{(s_{in}^C + s_{out}^C)^\alpha}, \tag{2.9}$$

où s_{in} est la somme des poids internes à C et s_{out} la somme des poids des liens liant C au reste du graphe. Étant donné que cette mesure de qualité ne s'applique pas à une partition, on dit qu'elle est locale à défaut d'être globale. L'algorithme fonctionne de la manière suivante. D'abord choisir un nœud i au hasard. Chercher sa "communauté naturelle", c'est-à-dire le groupe de nœuds contenant i tel qu'aucune suppression de nœud du groupe ou aucun ajout de nœuds voisins de C ne permet d'augmenter f_C^α. Puis, choisir un nœud au hasard parmi les nœuds encore non alloués à une communauté. Recommencer le processus jusqu'à qu'il n'y ait plus de nœuds non alloués. On obtient alors un découpage en communautés recouvrantes. Ici, le paramètre α joue le rôle de paramètre d'échelle. Cette méthode a l'inconvénient de beaucoup dépendre de quels nœuds on choisit dans la phase de tirage aléatoire. Nous verrons que ce caractère stochastique a aussi ses avantages.

Huang et al. [105] proposent une notion de similarité $s(u, v)$ entre deux nœuds u et v :

$$s(u,v) = \frac{\sum\limits_{x \sim u \text{ et } x \sim v} w(u,x)w(v,x)}{\sqrt{\sum\limits_{x \sim u} w^2(u,x)}\sqrt{\sum\limits_{x \sim v} w^2(v,x)}}. \tag{2.10}$$

Ils définissent alors une mesure de qualité locale, qu'ils appellent contraction (*tightness* en anglais) et qui s'applique à une communauté C (et non à une partition) :

$$T(C) = \frac{S_{in}^C}{S_{in}^C + S_{out}^C}, \tag{2.11}$$

où $S_{in}^C = \sum\limits_{u \in C, v \in C, u \sim v} s(u,v)$ est deux fois la somme des similarités entre toutes les

paires de nœuds adjacents de C ; et $S_{out}^C = \sum_{u \in C, v \notin C, u \sim v} s(u,v)$ la somme des similarités entre toutes les paires de nœuds adjacents dont un est dans C et l'autre à l'extérieur de C. Le gain de contraction $\tau_C^\alpha(a)$ obtenu quand une communauté C intègre un nouveau nœud a est paramétré par un paramètre d'échelle α :

$$\tau_C^\alpha(a) = \frac{S_{out}^C}{S_{in}^C} - \frac{\alpha S_{out}^a - S_{in}^a}{2S_{in}^a}. \tag{2.12}$$

Ils utilisent alors un algorithme d'optimisation similaire à celui de Lancichinetti et al. pour trouver des communautés.

Pons et Latapy [169] proposent une modularité paramétrée par un paramètre d'échelle α, mais à peu de choses près, on retrouve le cas de Reichardt en écrivant $\gamma = (1-\alpha)/\alpha)$ (section 12.1 de [83]).

Ronhovde et Nussinov [177] proposent également une méthode multiéchelle proche de celle de Reichardt, qu'ils nomment "méthode de Potts absolue" parce qu'elle ne fait pas intervenir un modèle de graphe aléatoire comme la méthode de Reichardt. Ceci dit, les deux méthodes restent conceptuellement similaire.

Citons pour finir les travaux de **Le Martelot et Hankin** [133, 134] qui ont proposé non pas d'autres fonctions de qualité à optimiser, mais deux algorithmes optimisés pour la détection multiéchelle de communautés. En effet, leur point de vue est de dire que la détection d'une partition à une échelle $s + \delta s$ peut-être effectuée plus rapidement en s'inspirant du résultat déjà obtenu à l'échelle s. Les auteurs proposent un algorithme pour optimiser les fonctions de qualité globales comme celles de Reichardt, d'Arenas, de Delvenne, de Pons ou de Ronhovde ; et un autre algorithme pour optimiser les fonctions de qualité locales comme celles de Lancichinetti ou de Huang.

3.5 Trouver quelle(s) échelle(s) sont pertinentes : méthodes existantes

En résumé, les méthodes multiéchelles présentées ci-dessus définissent toutes une fonction de qualité paramétrée par une notion d'échelle différente pour chaque auteur. Puis, cette fonction de qualité est optimisée avec un algorithme heuristique parfois choisi parce qu'il correspond bien à la fonction de qualité en question, d'autres fois choisi pour sa rapidité ou efficacité générale comme l'algorithme de Louvain. Dans la grande majorité des cas, une valeur particulière du paramètre d'échelle permet de retrouver la modularité classique. Dans tous ces travaux, les auteurs proposent un intervalle réel sur lequel est défini le paramètre d'échelle.
Étant donné que l'on a accès en théorie aux partitions P_s qui correspondent à toutes les valeurs de s sur cet intervalle, il est primordial d'associer à ces méthodes une mesure de pertinence des échelles considérées. Cette pertinence est mesurée en termes de stabilité de la partition associée. Il existe plusieurs types de stabilités que nous détaillons dans la suite. Nous verrons d'abord dans la partie 3.5.1 une stabilité mesurée en perturbant directement le graphe, puis dans la partie 3.5.2 une autre stabilité définie par la taille de l'intervalle d'échelles pour lesquelles la partition est trouvée, et finalement dans la partie 3.5.3 un dernier type d'instabilité qui utilise à profit la

stochasticité de certains algorithmes d'optimisation. Ces trois stabilités sont toutes basées sur des calculs de similarité entre partitions. Il existe plusieurs notions de similarité entre deux partitions que nous rappelons dans l'annexe A.

3.5.1 Stabilité par perturbation directe du graphe

Soit une partition P détectée dans un graphe $\mathcal{G}$. Dans les méthodes de perturbation directe du graphe, estimer la stabilité d'une partition revient à :

- créer de nombreux graphes perturbés $\mathcal{G}_1, \mathcal{G}_2, \cdots$ parfois appelés bootstraps,
- recalculer la partition optimale pour chaque graphe $P_1, P_2, \cdots$,
- et mesurer la similarité entre P et cette famille de partitions $\{P_k\}$ (voir l'annexe A).

La méthode de Gfeller et al. [88], par exemple, consiste à créer chaque graphe perturbé $\mathcal{G}_k$ en tirant, pour chaque lien (ij) de poids w_{ij} dans $\mathcal{G}$, un nouveau poids dans une distribution uniforme sur $[-\sigma w_{ij}, \sigma w_{ij}]$. Pour mesurer la similarité entre P et cette nouvelle famille de partitions P_k, les auteurs définissent la probabilité intra-cluster p_{ij} comme le nombre de fois que les deux nœuds i et j se sont retrouvés dans la même communauté sur le nombre total de graphes perturbés. Ensuite, ils proposent de considérer les liens tels que $p_{ij} < \theta$ comme des liens "externes", qui sont ensuite retirés de $\mathcal{G}$ pour donner un nouveau graphe souvent déconnecté, avec plusieurs composantes connexes. Les auteurs proposent finalement une mesure de similarité entre ces composantes connexes et la partition P trouvée au départ, avec un indice de similarité basé sur l'entropie.

La méthode de Karrer et al. [119], pour perturber le graphe, procède comme suit. Chaque lien (ij) est considéré et l'algorithme décide de le supprimer avec une probabilité α. Si c'est le cas, une paire de nœuds (i, j) est connectée aléatoirement avec une probabilité $p_{ij} = k_i k_j / 2k_{tot}$. A chaque suppression/création de liens, le graphe perturbé se rapproche du modèle aléatoire de Chung-Lu. La similarité utilise la variation d'information.

Même si le but de leur travail est de suivre les évolutions de communautés dans des graphes dynamiques, nous pouvons citer Rosvall et Bergstrom [178] qui proposent, pour créer des graphes perturbés, de tirer aléatoirement chaque nouveau poids w'_{ij} dans une distribution de Poisson de moyenne le poids initial w_{ij}. Ils définissent alors des noyaux de communautés, les groupes de nœuds maximaux qui se retrouvent dans la même communauté dans 95% des $\{P_k\}$.

Citons finalement la méthode de Lambiotte [127] qui consiste, pour perturber le graphe, à ajouter ou retirer $\pm 10\%$ à chaque lien. La stabilité est mesurée grâcé à la moyenne de la variation d'information entre tous les couples de la famille $\{P_k\}$.

Toutes ces méthodes sont paramétrées par la force de la perturbation et plus le graphe est perturbé, plus on va mesurer que la partition P est instable ! De plus, ces techniques ne sont pas très adaptées au problème de la détection multiéchelle. Pour deux raisons : la première vient du fait que la perturbation ne tient pas en compte de la taille des communautés. Par exemple, perturber un triangle en modifiant de $\pm 10\%$ ses liens aura plus d'impact sur cette structure que perturber une grande communauté de 100 nœuds. La deuxième raison est plus subtile. Admettons que l'on cherche à mesurer la stabilité de la partition P_{s_i} trouvé à l'échelle s_i. Après

perturbations du graphe, on se retrouve avec une famille de partitions $\{P_{s_i,k}\}$ toutes calculées *à la même échelle* s_i. Mais si on perturbe le graphe, le même paramètre d'échelle s_i ne correspond plus exactement à la même échelle physique du graphe.

3.5.2 Stabilité en terme d'intervalle d'échelle

Cette notion de stabilité est inhérente aux méthodes multiéchelles et a été proposée notamment par Arenas et al. [22], Lambiotte [127], Pons et al. [169] dans des formulations plus ou moins similaires. Soit P_s la partition trouvée à l'échelle s. Définisson l'intervalle d'existence de P_s, noté I_s, le plus grand intervalle d'échelle incluant s tel que :

$$\forall s' \in I_s \qquad P_{s'} = P_s. \tag{2.13}$$

Cet intervalle existe forcément étant donné que l'échelle est un paramètre continu. L'idée sous-jacente est la suivante : une partition P_s trouvée à l'échelle s est d'autant plus stable que son intervalle d'existence I_s est grand. Le problème de cette mesure de stabilité est double. D'une part, il y a peu de chances que la longueur d'un intervalle d'échelle à grandes échelles (par exemple l'intervalle I_s qui correspond à un découpage en deux communautés) soit vraiment équivalent à la longueur d'un intervalle d'échelle à petites ou moyennes échelles. D'autre part, le paramètre d'échelle est échantillonné uniquement M fois et la mesure de I_s va dépendre de la densité d'échantillonnage du paramètre, ce qui rend ce genre de mesures peu fiable.

3.5.3 Stabilité par stochasticité de l'algorithme d'optimisation

Certains algorithmes qui permettent d'optimiser les fonctions de qualité sur l'ensemble des partitions sont stochastiques : ils ne donnent pas à chaque fois le même résultat selon leur initialisation. C'est le cas, par exemple, de l'algorithme de Louvain (voir 2.2) ou l'algorithme de Lancichinetti (voir 3.4.4). Ces algorithmes ne sont pas adaptés aux deux précédentes mesures de stabilité. En revanche, certains auteurs [127, 129] en ont profité pour tourner cette stochasticité, *a priori* gênante, en mesure de stabilité. Ici, la famille de nouvelles partitions $\{P_k\}$ est générée en lançant autant de fois l'algorithme d'optimisation. Ce type de stabilité est semble-t-il le plus adapté au problème multiéchelle, car il ne risque pas de perturber les échelles en perturbant le graphe, et il n'implique pas de comparer des longueurs d'intervalle à différentes échelles qui ne sont pas forcément comparables.

4 Les ondelettes sur graphes et le partitionnement multiéchelle en communautés

Dans cette partie, nous allons développer les détails d'une nouvelle méthode de partitionnement multiéchelle en communautés, basée sur la corrélation d'ondelettes sur graphes.

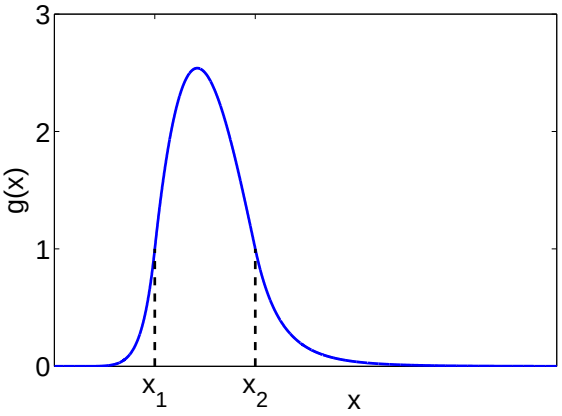


Figure 2.5: Forme "en cloche" du noyau de filtre passe-bande de l'équation 2.14.

4.1 Pourquoi une autre méthode ?

L'idée est de développer une méthode qui soit ancrée dans le domaine du traitement du signal pour plusieurs raisons. La première est de bénéficier de la précision de la définition d'échelle d'une ondelette, mathématiquement solidement fondée. Aussi, l'idée est de créer un pont entre l'analyse de graphe complexes et le traitement du signal, le but ultime étant de pouvoir un jour transposer les connaissances, algorithmes et techniques d'analyse du traitement du signal classique aux signaux sur graphes, pour aider l'analyse des graphes complexes. De plus, la question de la détection de communautés est un sujet central dans le domaine de l'analyse des graphes complexes et c'est une porte d'entrée idéale pour commencer à appliquer les premiers résultats du traitement du signal sur graphes. Finalement, un des buts est de pouvoir employer à terme les mêmes outils pour discuter des propriétés des graphes et des signaux sur graphes.

4.2 Le noyau de filtre d'ondelette et les bornes du paramètre d'échelle

Nous avons vu dans la partie 3.2 du chapitre 1 que les ondelettes sont entièrement déterminées par le noyau de filtre g. Rappelons que nous utilisons g tel que [96] :

$$g(x;\alpha,\beta,x_1,x_2) = \begin{cases} x_1^{-\alpha}\,x^{\alpha} & \text{pour} \quad x < x_1 \\ p(x) & \text{pour} \quad x_1 \leq x \leq x_2 \\ x_2^{\beta}\,x^{-\beta} & \text{pour} \quad x > x_2. \end{cases} \tag{2.14}$$

où $p(x)$ est l'unique interpolation polynomiale cubique qui conserve la continuité de g et de sa dérivée g'. α, β, x_1, x_2 sont les paramètres du filtre. La Fig. 2.5 montre la forme générale "en cloche" de ce noyau de filtre.

La plus grande échelle qui va nous intéresser est en fait codée dans $\boldsymbol{\chi}_2$, le vecteur de Fiedler, qui coupe le graphe en deux [78] : les nœuds i tel que $\boldsymbol{\chi}_2(i)$ est positif d'un côté et les nœuds j tel que $\boldsymbol{\chi}_2(j)$ est négatif de l'autre. Dans notre but de détecter des communautés, on pose la première contrainte : à l'échelle maximale s_{max}, le

résultat de la détection multiéchelle doit coïncider avec la solution du vecteur de Fiedler. Cela contraint le filtre $g(s_{max}\lambda)$ à être centré et piqué autour de $\lambda = \lambda_2$. Pour cela, on décide de s'assurer que $g(s_{max}x)$ commence à décroître comme une loi de puissance à partir de $x = \lambda_2$, i.e. $s_{max}\lambda_2 = x_2$. De plus, on s'assure que $g(s_{max}\lambda_3)$ (et *a fortiori* $g(s_{max}\lambda_i) \quad \forall i > 3$) est fortement atténué par rapport à $g(s_{max}\lambda_2)$: par exemple $g(s_{max}\lambda_3) = 10g(s_{max}\lambda_2)$, soit :

$$\beta = \frac{1}{\log_{10}(\frac{\lambda_3}{\lambda_2})}. \tag{2.15}$$

De plus, au vu de l'importance de $\boldsymbol{\chi}_2$ dans les algorithmes spectraux, nous ajoutons une deuxième contrainte : tous les filtres, même le filtre à l'échelle la plus petite s_{min}, doivent être sensible à $\boldsymbol{\chi}_2$. Ainsi $g(s_{min}x)$ est contraint à balayer toutes les valeurs propres avec notamment $s_{min}\lambda_2 = x_1$ et $s_{min}\lambda_{max} = x_2$. Avec λ_{max} la valeur propre de coupure du filtre que nous préciserons dans la suite.

Récapitulons, on a :

$$\begin{aligned} \beta &= \frac{1}{\log_{10}(\frac{\lambda_3}{\lambda_2})}, \\ s_{min}\lambda_{max} &= x_2, \\ s_{min}\lambda_2 &= x_1, \\ s_{max}\lambda_2 &= x_2. \end{aligned} \tag{2.16}$$

Les trois dernières équations lient quatre paramètres entre eux : x_1, x_2, s_{min} et s_{max} et se réécrivent :

$$\begin{aligned} s_{min} &= x_1\frac{1}{\lambda_2}, \\ x_2 &= x_1\frac{\lambda_{max}}{\lambda_2}, \\ s_{max} &= x_1\frac{\lambda_{max}}{\lambda_2^2}, \end{aligned} \tag{2.17}$$

où nous mettons en évidence que le paramètre x_1 a pour unique effet de translater l'intervalle d'échelles sur l'axe des réels, et n'a donc aucune conséquence ni sur les filtres, ni *a fortiori* sur les ondelettes. Sans perte de généralité, x_1 est fixé à 1, ce qui fixe à leur tour :

$$\begin{aligned} s_{min} &= \frac{1}{\lambda_2}, \\ x_2 &= \frac{\lambda_{max}}{\lambda_2}, \\ s_{max} &= \frac{\lambda_{max}}{\lambda_2^2}. \end{aligned} \tag{2.18}$$

En pratique, il est souvent inutile de prendre en compte les valeurs propres plus grandes que 1. En effet, les vecteurs propres associés ont de grandes chances d'être trop localisés et peu instructifs. Aussi, on sait [156] par exemple que dans un graphe bipartite, chaque valeur propre inférieure à 1 a une valeur propre associée supérieure à 1 : pour ces graphes ou pour des graphes quasi-bipartites, on ne gagne pas plus d'information en prenant en compte les valeurs propres supérieures à 1. Nous fixons

donc la valeur propre de coupure du filtre à plus petite échelle à $\lambda_{max} = 1$ [1]. On obtient finalement le jeu de paramètres :

$$\begin{aligned} s_{min} &= \frac{1}{\lambda_2}, \\ x_2 &= \frac{1}{\lambda_2}, \\ s_{max} &= \frac{1}{\lambda_2^2}. \end{aligned} \tag{2.19}$$

Notons que $\lambda_2 \leq 1$ pour tout graphe qui n'est pas complet. En effet, l'équation 1.13 de [48] donne la propriété suivante pour λ_2 dans le cas d'un graphe pondéré :

$$\lambda_2 = \inf_{f \text{ tq } \sum f(i)s_i = 0} \frac{\sum_{i \sim j} (f(i) - f(j))^2 w(i,j)}{\sum_{i \in \mathcal{V}} f(i)^2 s_i}, \tag{2.20}$$

où s_i est la force du nœud i. Soit un graphe non-complet, c'est-à-dire qu'il existe deux nœuds a et b qui ne sont pas connectés. Considérons un signal $\boldsymbol{f_1}$ défini sur les nœuds d'un graphe tel que :

$$f_1(v) = \begin{cases} s_a \text{ si } v = b, \\ s_b \text{ si } v = a, \\ 0 \text{ sinon.} \end{cases} \tag{2.21}$$

On a bien $\sum_{i \in \mathcal{V}} f_1(i) s_i = 0$ et en appliquant l'équation 2.20 avec $\boldsymbol{f_1}$, on obtient le même résultat obtenu dans le Lemme 1.7 de [48] pour les graphes binaires :

$$\lambda_2 \leq \frac{\sum_{i \sim j} (f_1(i) - f_1(j))^2 w(i,j)}{\sum_{i \in \mathcal{V}} f_1(i)^2 s_i} = 1. \tag{2.22}$$

Ainsi, le jeu de paramètres de l'équation 2.19 respecte toujours $s_{min} \leq s_{max}$ sauf pour les graphes complets (qui de toutes façons nécessitent peu d'outils d'analyse et encore moins d'outils de recherche de communautés !).

Reste à discuter le paramètre α. La seule contrainte sur α vient de la contrainte de localisation des ondelettes : $\alpha > 1$. En ondelettes classiques, α correspond au nombre de moments nuls. Mais cette interprétation n'est pas valide dans notre cas car, pour aucun filtre à aucune échelle on a $s\lambda$ assez petit devant $x_1 = 1$ pour que l'atténuation en loi de puissance α se fasse sentir sur les valeurs propres. En effet le minimum de $s\lambda$ est atteint pour $s = s_{min}$ et $\lambda = \lambda_2$ et dans ce cas : $s\lambda = s_{min}\lambda_2 = 1 = x_1$ (rappelons que le spectre $\{\lambda_i\}$ est discret, et donc très différent du cas classique où le spectre

1. Dans un cas extrême où nous souhaitons sonder des échelles très fines, on peut toujours décider de travailler avec $\lambda_{max} = 2$, ce qui veut dire que l'échelle la plus fine prend en compte toutes les valeurs propres.

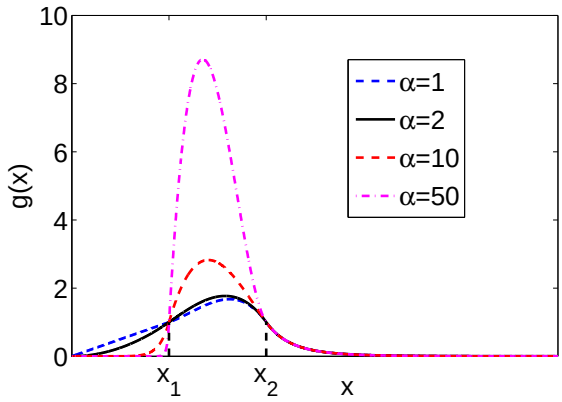


FIGURE 2.6: Effet du paramètre α sur la forme du filtre. Augmenter α revient à augmenter le maximum du filtre, et donc, indirectement, à rendre le filtre plus sélectif.

est continu). En fait, l'effet de α est indirect et se voit sur le maximum atteint par $g(x)$. Le maximum de $g(x)$ est toujours atteint pour une valeur x_M comprise entre x_1 et x_2, qui se rapproche d'autant plus de x_1 que α est grand. Aussi, le maximum $g(x_M)$ est d'autant plus grand que α est grand. Ces effets indirects sont illustrés sur la Fig. 2.6.

En définitive, plus α est grand, et plus le filtre est sélectif entre x_1 et x_2. Cette sélectivité est vraiment souhaitable à grand échelle (sélectivité autour de λ_2), mais ceci est assuré en fixant d'autres paramètres comme vu précédemment. L'effet de α va surtout se faire sentir aux petites et moyennes échelles, pour lesquelles on souhaite conserver une partie de l'information des petites valeurs propres : on ne souhaite donc pas être trop sélectifs à ces échelles. On contraint donc α à être petit. Or, α doit être supérieur à 1, donc on garde le choix de [96] : $\alpha = 2$ pour toute la suite.

Notons finalement que ce choix n'a pas un gros impact sur les ondelettes, comme illustré sur la Fig. 2.7 : on montre le pourcentage de passages par zéro (voir équation 1.44) moyenné sur toutes les ondelettes en fonction de l'échelle pour un graphe de Sales-Pardo (voir la partie 5.1.1). Le pourcentage de passages par zéro d'une ondelette est une manière de mesurer l'équivalent de la taille (ou de la "fréquence" moyenne) d'une ondelette sur graphe : plus l'ondelette a une "fréquence" moyenne élevée, moins elle est étendue sur le graphe, et plus son pourcentage de passages par zéro sera élevé. Deux remarques sont à tirer de la figure. D'abord, comme attendu, plus l'échelle est grande, moins les ondelettes passent par zéro, donc plus elles sont étendues sur le graphe. Aussi, notons que le choix de la valeur de α n'a pas un gros impact sur la taille des ondelettes.

En remarque finale, notons que le noyau de filtre g adapté au problème de détection de communautés, ainsi que les bornes intéressantes du paramètre d'échelle sont entièrement déterminés uniquement par la donnée de deux valeurs propres : λ_2 et λ_3. Ceci permettra par la suite d'ajuster les filtres en n'estimant que les deux premières valeurs propres non-nulles (à l'aide d'algorithmes comme l'itération d'Arnoldi [136]) au lieu de diagonaliser l'intégralité du laplacien.

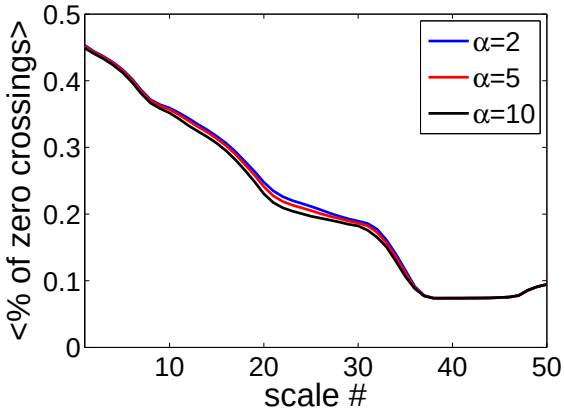


Figure 2.7: Effet du paramètre α sur le pourcentage de passages par zéro moyenné sur toutes les ondelettes en fonction de l'échelle, pour un graphe de Sales-Pardo (voir la partie 5.1.1).

4.3 Détection de communautés à une échelle s

4.3.1 Calcul des ondelettes

Soit un graphe $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ dont on diagonalise le laplacien normalisé $\boldsymbol{\mathcal{L}}$. Nous rappelons que nous préférons le laplacien normalisé au détriment du laplacien combinatoire pour toutes les raisons récapitulées dans la partie 2.6.1 du chapitre 1. Nous obtenons ainsi le spectre $\{\lambda_i\}_{i=[1,N]}$ que l'on ordonne : $0 = \lambda_1 < \lambda_2 \leq \lambda_3 \leq \cdots \leq \lambda_N \leq 2$. En effet, la première valeur propre λ_1 est toujours nulle. Nous allons considérer uniquement des graphes connectés (λ_1 est donc de multiplicité 1) et 2 est la valeur maximale du spectre quel que soit le graphe. Pour une démonstration de ces propriétés, voir les premières pages de [48]. La diagonalisation nous permet également d'obtenir les vecteurs propres associés que nous ordonnons dans la matrice $\boldsymbol{\chi}$:

$$\boldsymbol{\chi} = (\boldsymbol{\chi}_1 | \boldsymbol{\chi}_2 | \cdots | \boldsymbol{\chi}_N) \,. \tag{2.23}$$

Nous avons vu dans le premier chapitre la définition de l'ondelette sur graphe centrée autour du nœud a et à l'échelle s :

$$\boldsymbol{\psi}_{s,a} = \boldsymbol{\chi} G_s \boldsymbol{\chi}^\top \delta_a. \tag{2.24}$$

Nous pouvons regrouper ces ondelettes à l'échelle s dans une matrice, notée $\boldsymbol{\Psi}_s$:

$$\boldsymbol{\Psi}_s = (\boldsymbol{\psi}_{s,1} | \boldsymbol{\psi}_{s,2} | \cdots | \boldsymbol{\psi}_{s,N}) = \boldsymbol{\chi} \boldsymbol{G}_s \boldsymbol{\chi}^\top, \tag{2.25}$$

où $\boldsymbol{G}_s$ est la forme matricielle du noyau de filtre g dilaté par s :

$$\boldsymbol{G}_s = \mathrm{diag}(g(s\lambda_1) | g(s\lambda_1) | \cdots | g(s\lambda_N)). \tag{2.26}$$

4.3.2 Création du dendrogramme à l'échelle s

Considérons à présent une échelle $s \in [s_{min}, s_{max}]$. Comment obtient-on, à partir de $\boldsymbol{\Psi}_s$, la partition P_s qui sépare le graphe en communautés à l'échelle s ? L'intuition

est de considérer que l'ondelette centrée autour du nœud a est une vision "égo-centrée" du graphe, et deux nœuds dans la même communauté vont avoir à peu de choses près la même vision du graphe. Autrement dit, et c'est l'idée centrale de l'algorithme, nous classons ensemble les nœuds à voisinage (tel que défini par les ondelettes) similaire. En calculant la corrélation entre les ondelettes centrées autour des nœuds a et b, on peut mesurer à quel point ces nœuds sont proches topologiquement. On créé ainsi une matrice de distance $\boldsymbol{\mathcal{D}}_s$ où la distance $\mathcal{D}_s(a,b)$ entre les nœuds a et b est la distance de corrélation entre les ondelettes $\boldsymbol{\psi}_{s,a}$ et $\boldsymbol{\psi}_{s,b}$, c'est-à-dire :

$$\forall (a,b) \in \mathcal{V}^2 \qquad \mathcal{D}_s(a,b) = 1 - \frac{\psi_{s,a}^\top \psi_{s,b}}{||\psi_{s,a}||_2 ||\psi_{s,b}||_2}. \tag{2.27}$$

Notons que $\mathcal{D}_s(a,b)$ est bien une distance de corrélation car les ondelettes sont de moyenne nulle. En effet, la moyenne d'un signal $\boldsymbol{f}$ est son produit scalaire avec $\boldsymbol{\chi}_1$, le vecteur propre constant du laplacien. Ici, comme nous utilisons le laplacien normalisé $\boldsymbol{\mathcal{L}}$, la moyenne de $\boldsymbol{f}$ s'écrit :

$$\bar{f} = \chi_1^\top \boldsymbol{f} = \frac{1}{\sqrt{\sum_i s_i}} \sum_{i=1}^{N} \sqrt{s_i} f(i). \tag{2.28}$$

Avec cette définition de la moyenne (utilisée par exemple dans la partie 5.1 de [96]), on a :

$$\bar{\psi_{s,a}} = \boldsymbol{\chi}_1^\top \boldsymbol{\psi}_{s,a} = \boldsymbol{\chi}_1^\top \boldsymbol{\chi} \boldsymbol{G}_s \boldsymbol{\chi} \boldsymbol{\delta}_a = g_s(1)\chi_1(a), \tag{2.29}$$

car $\boldsymbol{\chi}$ est ortnormale. Par définition d'un filtre d'ondelettes, sa composante constante doit être nulle : $g_s(1) = 0$, si bien que $\bar{\psi_{s,a}} = 0$. Notons que si nous avions utilisé le laplacien combinatoire $\boldsymbol{L}$, dont le premier vecteur propre est constant égal à $1/\sqrt{N}$, alors la moyenne d'un signal $\boldsymbol{f}$ s'écrirait de manière classique : $\bar{f} = \frac{1}{N}\sum_{i=1}^{N} f(i)$.

Une fois $\boldsymbol{\mathcal{D}}_s$ calculée, on utilise un algorithme de clustering hiérarchique qui ordonne les nœuds au sein d'un dendrogramme, comme expliqué dans la partie 2.3. En pratique, on utilise la méthode de chaînage moyenné car elle est reconnue comme étant un compromis entre la sensitivité du chaînage complet aux points aberrants ou très éloignés de la moyenne, et la tendance du chaînage simple à créer de longues chaînes qui ne correspondent pas à la réalité de groupes denses et cohérents [143].

4.3.3 Couper le dendrogramme

À l'échelle s, nous obtenons donc un dendrogramme. Mais nous n'avons pas encore une partition ! En effet, reste à choisir quel critère utiliser pour automatiquement savoir où couper le dendrogramme ?

Comme expliqué en détail dans la partie 2.4, Donetti et. al [60] décident par exemple de considérer toutes les coupes possibles et de garder uniquement la partition qui maximise la modularité. De notre côté, nous cherchons à nous affranchir de la modularité qui favorise une échelle intrinsèque. Le choix le plus naturel est de couper le dendrogramme au niveau du saut maximal entre deux nœuds du dendrogramme, c'est-à-dire au niveau de l'intervalle de distance maximal dans lequel il n'y a aucune concaténation de groupe. Illustrons ce type de coupure sur la Fig. 2.8 sur deux dendrogrammes. Couper au niveau du plus grand saut est une idée connue

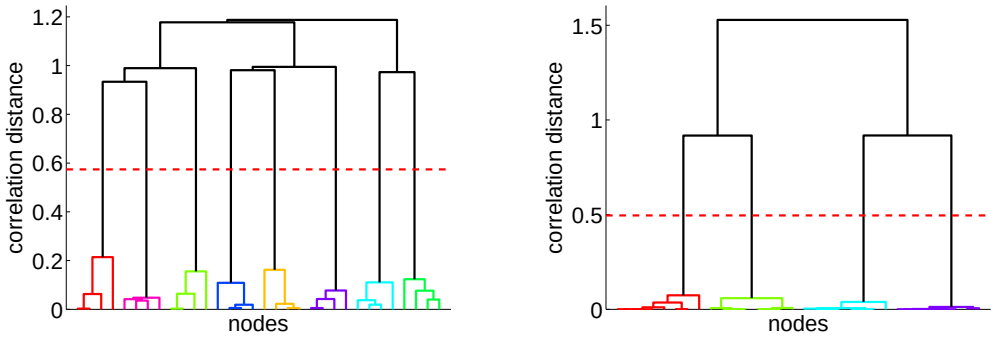


Figure 2.8: Couper le dendrogramme au niveau de son plus grand saut : illustration sur deux dendrogrammes.

mais nécessite pour être justifiée un rééchantillonnage et un test statistique (cf. *gap statistics* [101]).

Effectuer ce test aurait un coût en temps de calcul trop important étant donné qu'il devrait être fait à chaque échelle. Et si on ne fait pas le test statistique, montrons que simplement couper au plus grand saut est sensible à l'ajout de nœuds aberrants. Pour ce faire, considérons le graphe très structuré de la Fig. 2.9a). À une échelle s, son dendrogramme est représenté sur la Fig. 2.9c) et couper au saut maximal (tirets horizontaux) correspond à une partition en 4 communautés. Perturbons ce graphe en ajoutant quatre nœuds aberrants, c'est-à-dire qui ne sont pas du tout connectés au reste du graphe comme les autres nœuds : ils connectent des points *a priori* éloignés sur le graphe sans aucune cohérence. Ces quatre nœuds sont au centre de la représentation de la Fig. 2.9b) et leurs liens sont en rouge. Cet ajout perturbe forcément le dendrogramme qui devient celui de la Fig. 2.9d). Les quatre liens verts du dendrogramme correspondent aux quatre points aberrants : on peut voir qu'ils s'agrègent avec le reste du graphe à des distances élevées et perturbent le dendrogramme, si bien que le nouveau plus grand saut est celui qui coupe le graphe en deux communautés.

Étant donné que la plupart des graphes réels ont des points aberrants (dus à des erreurs de mesure par exemple), il faut affiner cette première méthode de coupe de dendrogramme pour la rendre plus robuste. Pour cela, nous décrivons dans ce paragraphe une nouvelle méthode que nous illustrons au fur et à mesure sur le dendrogramme de la Fig. 2.10a. Considérons un nœud a et son chemin de dendogramme associé : c'est le chemin entre la feuille du dendrogramme qui correspond au nœud a et la racine du dendrogramme (le nœud du dendrogramme qui a la plus grande distance de corrélation). On illustre un tel chemin en rouge sur la Fig. 2.10a. Pour ce nœud a, on peut tracer sa fonction de sauts Γ_a construite ainsi : suivre le chemin de dendrogramme en commençant à la distance de corrélation nulle. Pour chaque distance de corrélation, le chemin est entre deux nœuds du dendrogramme : tracer le saut entre eux. La fonction de sauts correspondant au chemin représenté sur Fig. 2.10a est tracée sur la Fig. 2.10b. En sommant toutes les fonctions de sauts

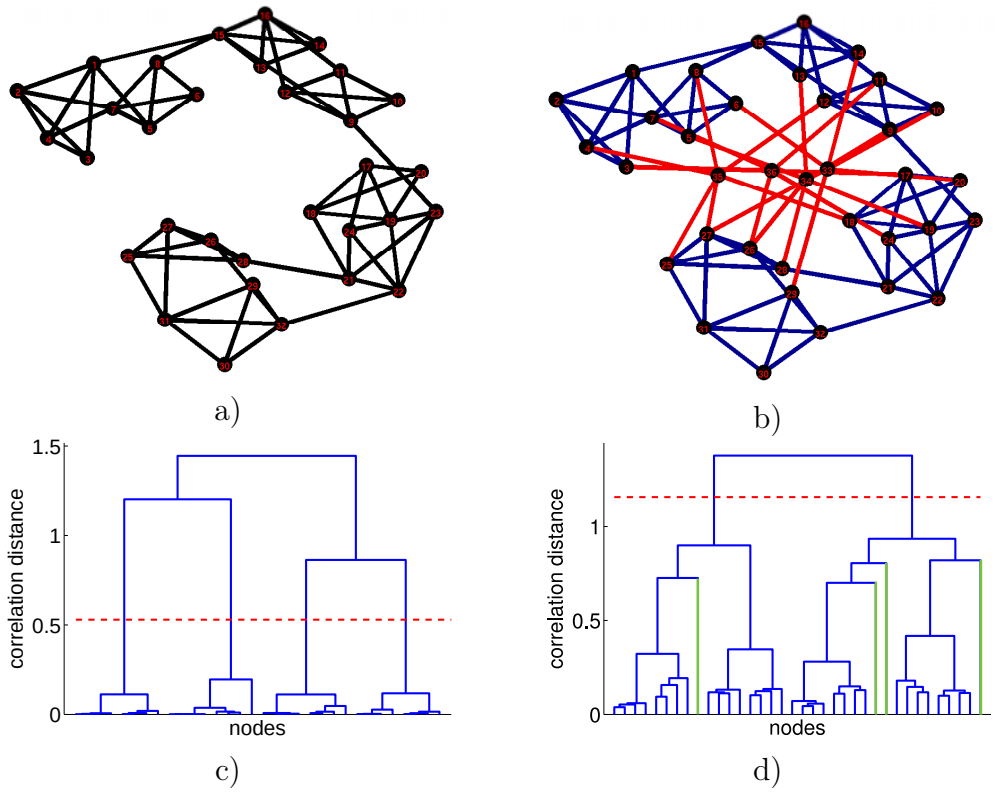


FIGURE 2.9: a) un graphe très structuré et c) le dendrogramme associé à une certaine échelle s. b) Une version perturbée du graphe a) où nous avons ajouté quatre nœuds aberrants (ils sont au centre de l'illustration et connectent faiblement avec des liens rouges beaucoup de nœuds du graphe). d) Le dendrogramme associé au graphe b) : les quatre liens verticaux verts correspondent aux quatre nœuds aberrants. On voit que le plus grand saut du dendrogramme non-perturbé est coupé en plusieurs petits sauts, si bien que le nouveau plus grand saut est celui qui coupe le graphe en deux.

associées à chaque nœud du graphe, on obtient la fonction de sauts globale :

$$\Gamma = \frac{1}{N\max(\text{corr. dist.})} \sum_{a \in V} \Gamma_a, \tag{2.30}$$

tracée sur la Fig. 2.10c. En suivant l'intuition derrière le *gap statistics*, on considère que la meilleure coupe du dendrogramme correspond au maximum de Γ (voir les Figs. 2.10c et 2.10d pour une illustration).

C'est l'action de la moyenne sur tous les nœuds qui rend cette méthode robuste. Elle a le désavantage d'être plus coûteuse en temps de calcul. Mais on peut raisonnablement estimer le maximum de Γ en calculant la moyenne uniquement sur un sous-ensemble aléatoire des chemins de dendrogramme, ce qui peut réduire le temps de calcul si besoin est. La Fig. 2.11 compare les deux méthodes de coupe de dendrogramme sur le dendrogramme de la Fig. 2.9d) : Γ est tracée sur la Fig. 2.11a), la coupe qui correspond au maximum de Γ est représentée en tirets noirs sur la Fig. 2.11b) : on retrouve les quatre communautés trouvés sur l'exemple non per-

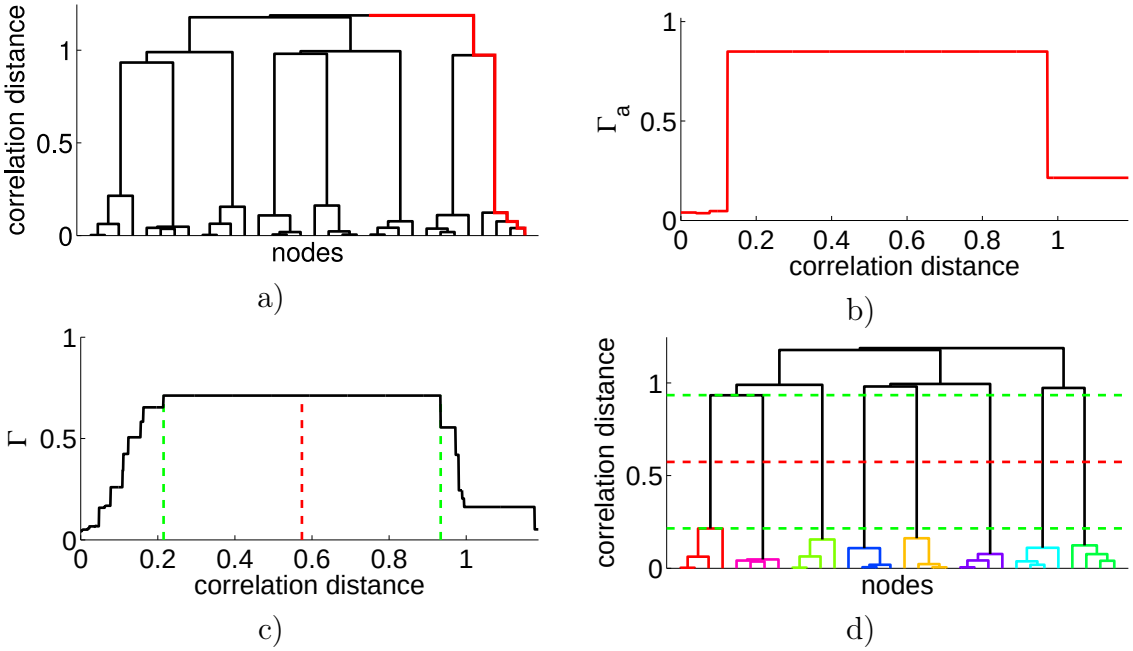


Figure 2.10: a) représente un dendrogramme dont un nœud a a été sélectionné : on représente en rouge son chemin feuille-racine. La fonction de sauts Γ_a associée est représentée en b). c) est la fonction de sauts globale Γ, où les pointillés verts représentent l'intervalle pour lequel Γ est maximal. d) Nous coupons le dendrogramme au niveau de cet intervalle.

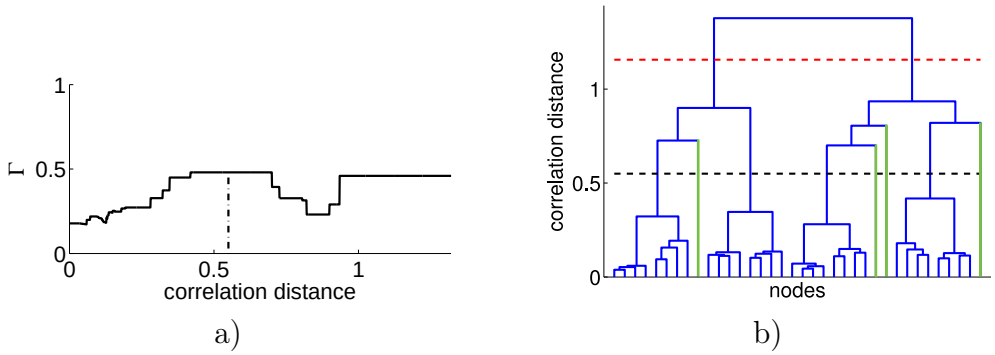


Figure 2.11: a) Fonction de sauts globale du dendrogramme de la figure b). En coupant le dendrogramme au maximum de Γ (pointillés noirs), nous retrouvons la solution en quatre partitions qui correspond à la solution du graphe non-perturbé de la Fig. 2.9a) : cette méthode est robuste aux points aberrants.

turbé alors que la méthode du saut maximal (dont la coupe est rappelée en rouge) correspond à une partition en seulement deux communautés.

4.3.4 Bilan d'étape : algorithme avec calcul intégral des ondelettes

Soit un graphe $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, et une échelle $s \in [s_{min}, s_{max}]$. Obtenir la partition P_s à cette échelle se fait en quatre étapes :

1. Calculer les ondelettes $\mathbf{\Psi}_s$.

2. Calculer la matrice de distance $\boldsymbol{\mathcal{D}}_s$.
3. Créer le dendrogramme associé à l'aide de l'algorithme hiérarchique de chaînage moyenné.
4. Couper le dendrogramme selon la méthode robuste présentée.

4.4 Détection rapide de communautés à une échelle s

4.4.1 Utiliser la transformée en ondelettes rapide de quelques signaux aléatoires

Un inconvénient important que nous allons traiter dès à présent est le temps de calcul nécessaire aux deux premières étapes de cette proposition. En effet, calculer les ondelettes nécessite au préalable de diagonaliser le laplacien, ce qui devient rapidement prohibitif au-delà de quelques milliers de nœuds. Nous avons vu dans le premier chapitre (partie 3.4) qu'il existe une transformée en ondelettes rapide qui permet d'approximer la transformée en ondelettes d'un signal sans diagonaliser le laplacien. Rappelons que $\mathcal{FWT}_{s,m}$, l'opérateur de transformée rapide à l'échelle s (avec une justesse d'approximation paramétrée par m) est tel que :

$$\boldsymbol{W}\boldsymbol{f}_s \simeq \mathcal{FWT}_{s,m}\boldsymbol{f}. \tag{2.31}$$

Une idée est d'approximer chaque ondelette en calculant :

$$\boldsymbol{\psi}_{s,a} \simeq \mathcal{FWT}_{s,m}\boldsymbol{\delta}_a, \tag{2.32}$$

et ensuite de calculer les corrélations pour estimer la matrice de distance. Mais il y a encore plus efficace. Nous montrons à présent qu'il suffit de calculer la transformée en ondelettes rapide de quelques signaux aléatoires sur le graphe pour avoir une bonne estimation de la matrice de distance $\boldsymbol{\mathcal{D}}_s$.

Considérons un signal aléatoire $\boldsymbol{r} \in \mathbb{R}^N$ défini sur les nœuds du graphe, composé de N variables aléatoires gaussiennes et indépendantes de moyenne nulle et de variance σ^2. Définissons $f_{s,a} \in \mathbb{R}$ la projection de ce signal aléatoire sur l'ondelette $\boldsymbol{\psi}_{s,a}$:

$$f_{s,a} = \boldsymbol{\psi}_{s,a}^\top \boldsymbol{r} = \sum_{k=1}^{N} \psi_{s,a}(k) r(k). \tag{2.33}$$

Étant la somme de N variables aléatoires gaussiennes et indépendantes, $f_{s,a}$ est aussi une variable aléatoire gaussienne d'espérance :

$$\mathbb{E}(f_{s,a}) = \boldsymbol{\psi}_{s,a}^\top \mathbb{E}(\boldsymbol{r}) = 0 \tag{2.34}$$

et de variance :

$$\begin{aligned} \mathrm{Var}(f_{s,a}) &= \mathbb{E}((f_{s,a} - \mathbb{E}(f_{s,a}))^2) = \mathbb{E}(f_{s,a}^2) \\ &= \boldsymbol{\psi}_{s,a}^\top \mathbb{E}(\boldsymbol{r}^\top \boldsymbol{r}) \boldsymbol{\psi}_{s,a} = \sigma^2 ||\boldsymbol{\psi}_{s,a}||_2^2. \end{aligned} \tag{2.35}$$

Considérons la corrélation entre les variables aléatoires $f_{s,a}$ associée au nœud a et $f_{s,b}$ associée à un autre nœud b. Par définition :

$$\begin{aligned}\mathrm{Cor}(f_{s,a}, f_{s,b}) &= \frac{\mathbb{E}((f_{s,a} - \mathbb{E}(f_{s,a}))(f_{s,b} - \mathbb{E}(f_{s,b})))}{\sqrt{\mathrm{Var}(f_{s,a})\mathrm{Var}(f_{s,b})}} \\ &= \frac{\mathbb{E}(f_{s,a}f_{s,b})}{\sigma^2||\boldsymbol{\psi}_{s,a}||_2||\boldsymbol{\psi}_{s,b}||_2}. \end{aligned} \tag{2.36}$$

Calculons la covariance :

$$\begin{aligned}\mathbb{E}(f_{s,a}f_{s,b}) &= \mathbb{E}((\boldsymbol{\psi}_{s,a}^\top \boldsymbol{r})(\boldsymbol{\psi}_{s,b}^\top \boldsymbol{r})) \\ &= \mathbb{E}((\sum_{k=1}^{N} \psi_{s,a}(k)r(k))(\sum_{k'=1}^{N} \psi_{s,b}(k')r(k'))) \\ &= \sum_{k\neq k'} \psi_{s,a}(k)\psi_{s,b}(k')\mathbb{E}(r(k)r(k')) + \sum_{k=1}^{N} \psi_{s,a}(k)\psi_{s,b}(k)\mathbb{E}(r(k)^2) \\ &= \sigma^2 \boldsymbol{\psi}_{s,a}^\top \boldsymbol{\psi}_{s,b}. \end{aligned} \tag{2.37}$$

Si bien que :

$$\mathrm{Cor}(f_{s,a}, f_{s,b}) = \frac{\boldsymbol{\psi}_{s,a}^\top \boldsymbol{\psi}_{s,b}}{||\boldsymbol{\psi}_{s,a}||_2||\boldsymbol{\psi}_{s,b}||_2}. \tag{2.38}$$

La corrélation entre $f_{s,a}$ et $f_{s,b}$ est exactement la corrélation entre les ondelettes centrées en a et b. Avant de passer à l'estimation de cette corrélation, montrons que $f_{s,a}$ et $f_{s,b}$ sont conjointement gaussiennes, i.e. que toute combinaison linéaire $cf_{s,a} + df_{s,b}$ ($(c,d) \in \mathbb{R}^2$) est gaussienne. En fait :

$$cf_{s,a} + df_{s,b} = \sum_{k=1}^{N} (c\psi_{s,a}(k) + d\psi_{s,b}(k))r(k) \tag{2.39}$$

est une somme de variables indépendantes gaussiennes, donc elle-même gaussienne.

Pour estimer la corrélation de l'équation (2.38), nous utilisons l'estimateur de corrélation classique. Considérons η réalisations de $\boldsymbol{r}$ que nous stockons dans la matrice :

$$\boldsymbol{R} = (\boldsymbol{r}_1|\boldsymbol{r}_2|\dots|\boldsymbol{r}_\eta) \in \mathbb{R}^{N\times\eta} \tag{2.40}$$

où la $i^{\text{ème}}$ colonne $\boldsymbol{r}_i$ est la $i^{\text{ème}}$ réalisation de $\boldsymbol{r}$. Notons $f_{s,a}^i = \boldsymbol{\psi}_{s,a}^\top \boldsymbol{r_i}$ la $i^{\text{ème}}$ réalisation de $f_{s,a}$, et concaténons toutes ces η réalisations dans le vecteur caractéristique $\boldsymbol{f}_{s,a}$:

$$\boldsymbol{f}_{s,a}^\top = \boldsymbol{\psi}_{s,a}^\top \boldsymbol{R} \qquad (\boldsymbol{f}_{s,a} \in \mathbb{R}^\eta). \tag{2.41}$$

L'estimateur du coefficient de corrélation entre $\boldsymbol{f}_{s,a}$ et $\boldsymbol{f}_{s,b}$ s'écrit :

$$\hat{C}_{ab,\eta} = \frac{(\boldsymbol{f}_{s,a} - \bar{\boldsymbol{f}}_{s,a})^\top(\boldsymbol{f}_{s,b} - \bar{\boldsymbol{f}}_{s,b})}{||\boldsymbol{f}_{s,a} - \bar{\boldsymbol{f}}_{s,a}||_2||\boldsymbol{f}_{s,b} - \bar{\boldsymbol{f}}_{s,b}||_2}, \tag{2.42}$$

où $\bar{\boldsymbol{f}}_{s,a}$ est le vecteur constant égal à la moyenne de $\boldsymbol{f}_{s,a}$: si on note $\mathbb{1}$ le vecteur constant égal à 1, $\bar{\boldsymbol{f}}_{s,a} = \frac{1}{\eta}\mathbb{1}^\top \boldsymbol{f}_{s,a}\mathbb{1}$.

Comme $f_{s,a}$ et $f_{s,b}$ sont conjointement gaussiennes, cet estimateur est asymptotiquement consistent, i.e. :

$$\lim_{\eta \to +\infty} \hat{C}_{ab,\eta} = \mathrm{Cor}(f_{s,a}, f_{s,b}) = \frac{\boldsymbol{\psi}_{s,a}^\top \boldsymbol{\psi}_{s,b}}{||\boldsymbol{\psi}_{s,a}||_2 ||\boldsymbol{\psi}_{s,b}||_2}. \tag{2.43}$$

Ainsi :

$$\lim_{\eta \to +\infty} 1 - \hat{C}_{ab,\eta} = \boldsymbol{D}_s(a,b). \tag{2.44}$$

En pratique, les expériences montrent qu'on retrouve les bonnes structures en communautés pour η relativement petit devant N (plus les communautés sont évidentes à trouver, plus η peut être petit). Ainsi, au lieu de calculer la transformée en ondelettes rapides de N Diracs pour obtenir les ondelettes, puis de calculer la corrélation de N vecteurs de taille N, il suffit de calculer la transformée en ondelettes d'un petit nombre η de vecteurs aléatoires, et ensuite calculer la matrice de corrélation de N vecteurs de taille η.

4.4.2 Bilan d'étape : algorithme avec vecteurs aléatoires

Soit un graphe $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, et une échelle $s \in [s_{min}, s_{max}]$. L'algorithme de base résumé dans la partie 4.3.4 peut être amélioré en termes de temps de calcul grâce à ces vecteurs aléatoires. Selon le nouvel algorithme, la partition P_s à l'échelle s se calcule en quatre étapes :

1. Générer une matrice de η vecteurs aléatoires gaussiens de moyenne nulle et de variance 1 :
$$\boldsymbol{R} = (\boldsymbol{r}_1 | \boldsymbol{r}_2 | \dots | \boldsymbol{r}_\eta) \in \mathbb{R}^{N \times \eta}.$$
Et calculer la transformée en ondelettes rapide de ces η vecteurs aléatoires, pour obtenir un vecteur caractéristique $\boldsymbol{f}_{s,a}$ par nœud :
$$\mathcal{FWT}_{s,m} \boldsymbol{R} = [\boldsymbol{f}_{s,1}^\top | \boldsymbol{f}_{s,2}^\top | \cdots | \boldsymbol{f}_{s,N}^\top]^\top. \tag{2.45}$$
2. Estimer la matrice de distance $\boldsymbol{\mathcal{D}}_s$:
$$\forall (a,b) \in \mathcal{V}^2 \qquad \boldsymbol{\mathcal{D}}_s(a,b) \simeq 1 - \hat{C}_{ab,\eta} = 1 - \frac{(\boldsymbol{f}_{s,a} - \bar{\boldsymbol{f}}_{s,a})^\top (\boldsymbol{f}_{s,b} - \bar{\boldsymbol{f}}_{s,b})}{||\boldsymbol{f}_{s,a} - \bar{\boldsymbol{f}}_{s,a}||_2 ||\boldsymbol{f}_{s,b} - \bar{\boldsymbol{f}}_{s,b}||_2}. \tag{2.46}$$
3. Créer le dendrogramme associé à l'aide de l'algorithme hiérarchique de chaînage moyenné.
4. Couper le dendrogramme selon la méthode présentée dans 4.3.3 pour obtenir P_s.

4.5 Mesure de stabilité de l'échelle s

Estimer rapidement la matrice de distance $\boldsymbol{\mathcal{D}}_s$ n'est pas le seul avantage à utiliser des vecteurs aléatoires. En effet, grâce à cela, l'algorithme est maintenant stochastique, et nous pouvons utiliser à bon escient cette stochasticité pour estimer la stabilité de la partition P_s obtenue (voir la partie 3.5.3). L'intuition est la suivante :

si en répétant J fois l'algorithme avec J tirages différents de la matrice $\boldsymbol{R}$ on obtient J versions très différentes de P_s, on peut considérer la partition comme instable et l'échelle peu pertinente. En revanche, si on retrouve à chaque fois à peu près la même solution, la partition est jugée stable et l'échelle s pertinente. Tant pour décrire une partition P_s que son échelle associée s, nous utiliserons les adjectifs stable/pertinente (ou instable/non-pertinente si c'est le cas) indifféremment.

En pratique, pour estimer la stabilité $\gamma_a(s)$ de l'échelle s, considérons J réalisations de $\boldsymbol{R}$ (typiquement $J = 20$) et calculons l'ensemble des partitions associées $\{P_s^j\}_{j\in J}$. La stabilité $\gamma_a(s)$ est définie comme la moyenne des similarités calculées deux à deux de toutes les partitions de $\{P_s^j\}_{j\in J}$:

$$\gamma_a(s) = \frac{2}{J(J-1)} \sum_{(i,j)\in J, i\neq j} \texttt{simi}(P_s^i, P_s^j), \tag{2.47}$$

où `simi` est une mesure au choix parmi celles décrites dans l'annexe A. Nous utiliserons principalement l'indice de Rand ajusté, mais cela a peu d'impact sur γ_a.

4.6 Test statistique

Motivation. Le résumé de la partie 4.4.2 ainsi que le paragraphe précédent nous permettent de détecter l'ensemble de partitions $\mathcal{P} = \{P_{s_i}\}_{s_i\in S}$ et de calculer la stabilité γ_a associée à chacune d'entre elles. De ces informations, on peut extraire les K "meilleures" partitions, c'est-à-dire les K partitions les plus stables. Les méthodes multiéchelles de l'état de l'art s'arrêtent là et ne vont pas plus loin. Alors qu'il manque semble-t-il encore un élément à l'analyse. En effet, l'algorithme produira les K meilleures partitions d'un graphe aléatoire d'Erdös-Rényi si on le lui demande, ce qui n'a pas beaucoup de sens car un graphe aléatoire n'a – par construction – aucune structure en communautés. Mais il existera quand même des partitions plus stables que les autres (uniquement dues à la chance !) qui seront extraites artificiellement. Il est donc important de pousser l'analyse encore plus loin et de proposer un moyen de donner une valeur intrinsèque à la stabilité mesurée. Nous proposons dans cette partie un tel moyen basé sur un test statistique où nous comparons les stabilités mesurées, à des stabilités typiquement mesurées dans des versions rendues aléatoires du même graphe. Cette question de "valeur intrinsèque" se pose également dans les algorithmes basées sur une fonction de qualité, comme la modularité [161, 127, 52]. En effet, comment savoir que telle ou telle valeur de la modularité est vraiment pertinente ? Dans le langage de la détection de communautés telle que nous la proposons, cette question devient : quelle est la valeur seuil γ_a^{th} au dessus de laquelle on peut juger une partition suffisamment stable ? Nous développons dans la suite un test statistique permettant d'estimer cette valeur seuil γ_a^{th}.

Le test. Considérons un graphe $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ et l'ensemble de partitions $\mathcal{P} = \{P_{s_i}\}_{s_i\in S}$ trouvé par l'algorithme ainsi que leur mesure de stabilité $\{\gamma_a(s_i)\}_{s_i\in S}$. Afin de tester quelles échelles sont intrinsèquement intéressantes, nous allons comparer la mesure de stabilité $\gamma_a(s_i)$ de chaque échelle à la stabilité calculée pour des versions rendues aléatoires (selon la méthode de l'annexe B) du graphe. Cela se fait en quatre étapes :

1. Énoncer l'hypothèse nulle H_0 : $\mathcal{G}$ n'a pas de structure en communautés à aucune échelle.

2. Générer un grand nombre R de versions aléatoires de $\mathcal{G} = (\mathcal{V}, \mathcal{E})$: on obtient une collection de graphes de Chung-Lu pondérés $\{\mathcal{G}_r\}_{r\in[1,R]}$.

3. Calculer les stabilités $\{\gamma_a^r(s_i)\}_{s_i\in S}$ de chaque graphe aléatoire $\mathcal{G}_r$. Concaténer toutes ces valeurs dans $\mathcal{S}_{\gamma_a} = \{\gamma_a^r\ ,\ r \in [1, R]\}$, ce qui nous donne accès à la distribution empirique de γ_a sous hypothèse nulle.

4. Pour toute échelle $s_i \in S$, si $\gamma_a(s_i)$ est plus grand que le α-quantile supérieur γ_a^{th} de $\mathcal{S}_{\gamma_a}$, alors H_0 est rejetée avec une confiance $1 - 1/\alpha$ (si $R >> \alpha$) : $\mathcal{G}$ a une structure en communautés à cette échelle. Typiquement, on utilise $\alpha = 100$ et R assez grand pour que $\mathcal{S}_{\gamma_a}$ ait un cardinal supérieur à $10\alpha = 1000$, i.e. $R \sim \frac{1000}{M}$ (en effet, chaque graphe aléatoire contribue à hauteur de M valeurs à $\mathcal{S}_{\gamma_a}$).

Finalement, le résultat du test est un ensemble d'échelles pour lesquelles l'hypothèse nulle est rejetée : $\hat{S} = \{s_i \in S \text{ t.q. } \gamma_a(s_i) \geq \gamma_a^{\text{th}}\} \subset S$, c'est-à-dire pour lesquelles les partitions associées $\hat{\mathcal{P}} = \{P_k, k \in \hat{S}\}$ sont stables avec un niveau de signification statistique à $1 - 1/\alpha = 99\%$.

Remarque. Notons que la méthode de l'annexe B est une manière de rendre aléatoire un graphe parmi beaucoup d'autres. L'idée ici est de créer des graphes aléatoires les plus réalistes possibles, qui vont détruire toute structure en communautés, mais pas toutes les corrélations qui peuvent exister dans le graphe : cela permet d'avoir un test statistique avec moins de faux positifs. En effet, admettons qu'un graphe ne possède aucune structure en communautés mais possède d'autres types de corrélations. Si on le compare à une version complètement aléatoire pour laquelle on ne garde aucune structure (par exemple un graphe d'Erdös-Rényi avec une probabilité telle que l'espérance du nombre total de liens soit respectée), on risque de classer les partitions du graphe comme étant stables simplement parce qu'on détecte les corrélations au sein du graphe et non une structure de communauté pertinente !

5 Illustrations et comparaison avec d'autres méthodes

Dans cette partie, nous allons illustrer la nouvelle méthode que nous venons de présenter sur des exemples et la comparer avec d'autres méthodes multiéchelles sur des modèles de graphes hiérarchiques.

5.1 Illustration sur un modèle de graphe hiérarchique

5.1.1 Un modèle de graphe hiérarchique : le modèle de Sales-Pardo

Ce modèle de graphe hiérarchique est binaire et non-dirigé et a été introduit par Sales-Pardo et al. [182], et par la suite utilisé entre autres par Lambiotte et al. [127] pour tester des outils de détection multiéchelle de communautés. Nous rappelons ici la définition et la construction d'un graphe de Sales-Pardo. Considérons N nœuds, et trois structures en communautés imbriquées les unes dans les autres : N/N_3 communautés de N_3 nœuds (la petite échelle), imbriquées dans N/N_2 communautés

à N_2 nœuds (l'échelle moyenne), elles-mêmes imbriquées dans N/N_1 communautés à N_1 nœuds (la grande échelle), où $N_3 < N_2 < N_1 < N$. Chaque nœud adhère donc à trois communautés : une à chaque échelle. Soit un nœud i. On définit S_x le nombre de nœuds qui ont x adhésions en commun avec i. Ici :

- tous les nœuds qui ne sont pas dans la plus grande comunauté de i n'ont aucune adhésion en commun avec i : $S_0 = N - N_1$.
- tous les nœuds qui sont dans la même grande communauté que i, mais pas dans la même moyenne communauté ont une seule adhésion en commun avec i : $S_1 = N_1 - N_2$.
- tous les nœuds qui sont dans la même moyenne communauté que i (et donc *a fortiori* dans la même grande communauté que i) mais pas dans la même petite communauté ont deux adhésions en commun : $S_2 = N_2 - N_3$.
- tous les nœuds (différents de i) qui sont dans la même petite communauté que i ont trois adhésions en commun avec i : $S_3 = N_3 - 1$.

Considérons $\bar{k_3}$ le degré moyen intra-petite communauté, $\bar{k_2}$ le degré moyen intra-moyenne communauté (mais extra-petite communauté), $\bar{k_1}$ le degré moyen intra-grande communauté (mais extra-petite et moyenne communautés) et $\bar{k_0}$ le degré moyen extra-grande communauté. Définissons :

$$\begin{aligned} \bar{k_0} &= p_0 S_0, & \bar{k_1} &= p_1 S_1, \\ \bar{k_2} &= p_2 S_2, & \bar{k_3} &= p_3 S_3, \end{aligned} \tag{2.48}$$

où p_x est la probabilité d'existence d'un lien entre deux nœuds qui ont x adhésions en commun. Un premier paramètre ρ contrôle à quel point les différentes échelles sont séparées :

$$\rho = \frac{\bar{k_0}}{\bar{k_1}} = \frac{\bar{k_0} + \bar{k_1}}{\bar{k_2}} = \frac{\bar{k_0} + \bar{k_1} + \bar{k_2}}{\bar{k_3}}. \tag{2.49}$$

Le plus petit est ρ, le plus séparées sont les échelles entre elles, le plus facile ce sera d'extraire la structure hiérarchique en communautés. Un deuxième paramètre, le degré moyen $\bar{k}$, contrôle la densité du graphe :

$$\bar{k} = \bar{k_0} + \bar{k_1} + \bar{k_2} + \bar{k_3}. \tag{2.50}$$

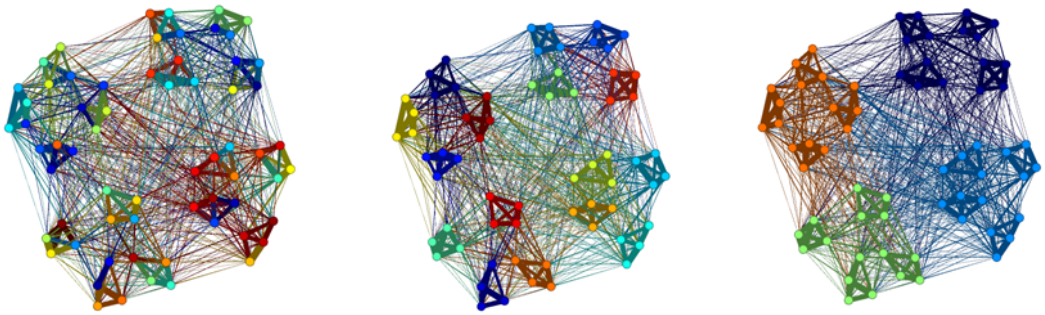
Le plus petit est $\bar{k}$, le plus parcimonieux est le graphe, le plus difficile ce sera de retrouver les communautés. Pour un jeu de paramètres donné $(\rho, \bar{k})$, on obtient le système d'équations suivant pour les probabilités p_i :

$$\begin{aligned} p_0 &= \frac{\rho^3}{(1+\rho)^3}\frac{\bar{k}}{S_0}, & p_1 &= \frac{\rho^2}{(1+\rho)^3}\frac{\bar{k}}{S_1}, \\ p_2 &= \frac{\rho}{(1+\rho)^2}\frac{\bar{k}}{S_2}, & p_3 &= \frac{\bar{k}}{(1+\rho)S_3}. \end{aligned} \tag{2.51}$$

Les p_i étant des probabilités entre 0 et 1, on a la contrainte implicite suivante :

$$\frac{\bar{k}}{1+\rho} \leq S_3. \tag{2.52}$$

Dans cette thèse, à moins que ce soit explicitement indiqué, nous utilisons des graphes de Sales-Pardo à $N = 640$ nœuds, et trois structures en communautés imbriquées : 64 petites communautés de $N_3 = 10$ nœuds imbriquées dans 16 moyennes



(1) (2) (3)

FIGURE 2.12: Schéma d'un graphe de Sales-Pardo à 640 nœuds : chaque nœud illustré est en fait une petite communauté de 10 nœuds. La figure 1 représente la partition à petite échelle avec 64 communautés de 10 nœuds, imbriquées dans 16 communautés de 40 nœuds (l'échelle moyenne de la figure 2), elles-mêmes imbriquées dans 4 communautés de 160 nœuds (la grande échelle de la figure 3).

communautés à $N_2 = 40$ nœuds, elles-mêmes imbriquées dans 4 grandes communautés à $N_1 = 160$. Ainsi, $S_0 = 480$, $S_1 = 120$, $S_2 = 30$ and $S_3 = 9$.

Pour générer un graphe de Sales-Pardo à $(\rho, \bar{k})$ fixés, se référer au système d'équations 2.51 pour obtenir les probabilités d'existence de lien et générer ensuite le graphe aléatoirement en respectant ces probabilités. Une réalisation d'un tel graphe est représentée sur la Fig. 2.12 où on met en couleurs les trois différentes structures en communautés.

5.1.2 Illustration en calculant toutes les ondelettes

Nous appliquons ici l'algorithme basé sur le calcul intégral des ondelettes et de leurs corrélations résumé dans 4.3.4 sur la réalisation d'un graphe de Sales-Pardo (avec $\rho = 1$ et $\bar{k} = 16$) illustrée sur la Fig. 2.12. Le calcul de λ_2 et λ_3 donne $\beta = 41$, $s_{min} = x_2 = 7$ et $s_{max} = 47$. De plus, nous choisissons de scanner $M = 50$ échelles logarithmiquement espacées entre s_{min} et s_{max}. La Fig. 2.13 représente les filtres pour quatre échelles différentes. Pour information, nous montrons également le spectre du graphe sur cette figure.

L'avantage d'utiliser un modèle de graphe hiérarchique est que l'on connaît la "vérité de terrain", c'est-à-dire les partitions théoriques en communauté. L'algorithme donnant l'ensemble de partitions $\mathcal{P} = \{P_{s_i}\}_{i\in[1,M]}$, il suffit de calculer la similarité (par exemple en utilisant l'indice de Rand ajusté) entre toutes les partitions de $\mathcal{P}$ et les trois partitions théoriques pour avoir une idée de la performance de l'algorithme. C'est ce que nous illustrons sur la Fig. 2.14. On observe notamment des intervalles d'échelles (représentés par des doubles flèches) pour lesquels toutes les partitions à ces échelles correspondent exactement à la partition théorique.

5.1.3 Illustration de l'algorithme rapide

Nous appliquons à présent l'algorithme rapide résumé dans la partie 4.4.2 et basé sur la transformée en ondelettes de quelques vecteurs aléatoires, sur le même graphe.

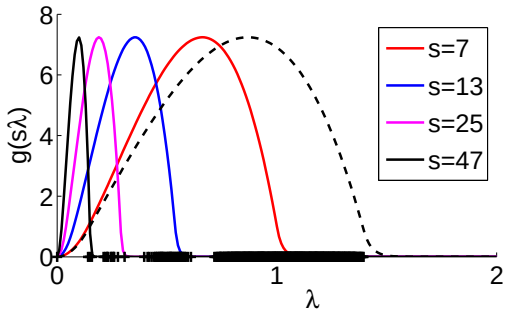


Figure 2.13: Les filtres d'ondelettes à quatre échelles différentes. Le spectre est indiqué par des croix sur l'axe des abscisses. On représente en pointillés un filtre à une échelle trop petite pour être intéressante dans la majeure partie des cas : on s'arrête en effet aux filtres qui s'annulent après $\lambda = 1$.

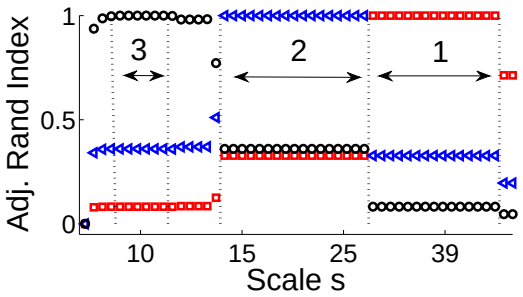


Figure 2.14: Résultat de l'algorithme : l'indice Rand ajusté entre la partition trouvée à chaque échelle et la partition théorique à petite (resp. moyenne, grande) échelle est représenté en noir (resp. bleu, rouge). Les trois intervalles d'échelle représentés par des double flèches sont les intervalles pour lesquels l'algorithme retrouve exactement les trois partitions théoriques.

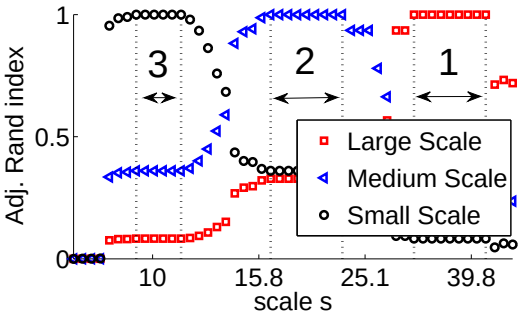


Figure 2.15: Résultat de l'algorithme rapide en utilisant $\eta = 60$.

On représente sur la Fig. 2.15 le résultat en utilisant $\eta = 60$ vecteurs aléatoires. On retrouve de nouveau les trois structures en communautés, sur des intervalles d'échelles néanmoins plus petits. Afin de quantifier la performance de l'algorithme en fonction du nombre de vecteurs aléatoires η nous définissons les trois ratios suivants :

- Le ratio de rappel à grande échelle (en anglais *Large Scale Recall ratio* : LSR) qui est le maximum de similarité entre les partitions $\mathcal{P}$ trouvées par l'algorithme et la partition théorique à grande échelle.
- Le ratio de rappel à moyenne échelle (en anglais *Medium Scale Recall ratio* : MSR) qui est le maximum de similarité entre les partitions $\mathcal{P}$ trouvées par l'algorithme et la partition théorique à moyenne échelle.
- Le ratio de rappel à petite échelle (en anglais *Small Scale Recall ratio* : SSR) qui est le maximum de similarité entre les partitions $\mathcal{P}$ trouvées par l'algorithme et la partition théorique à petite échelle.

En calculant les moyennes et médianes de ces ratios sur 100 réalisations du graphe de Sales-Pardo, on peut mesurer la capacité de l'algorithme à retrouver les trois structures en communauté. Les résultats sont exposés sur la Fig. 2.16. Il est très intéressant de noter que plus on utilise de vecteurs aléatoires, plus on arrive à retrouver les structures fines du graphe. Si on regarde la Fig. 2.16d), on voit que, en médiane, il suffit de 7 (resp. 15, 40) vecteurs aléatoires pour retrouver exactement la structure à grande (resp. moyenne, petite) échelle.

Il y a ici un rapprochement à faire avec l'acquisition comprimée (en anglais *compressive sensing* [62]), qui utilise des acquisitions aléatoires de données parcimonieuses pour sonder rapidement un grand espace. Dans notre cas, la donnée parcimonieuse est la somme des trois partitions en communautés, le grand espace l'espace des partitions possibles, et l'acquisition aléatoire le calcul de corrélation de transformée en ondelettes de vecteurs aléatoires. On note, et c'est un résultat empirique (comme ceux présentés au paragraphe précédent) que nous avons retrouvé dans de nombreux exemples, que le nombre de vecteurs aléatoires nécessaires pour détecter une partition en x communautés est de l'ordre de x. Cela est très utile à savoir : en effet, on peut avoir affaire à de grands graphes pour lesquels on veut une vision assez simpliste de la réalité pour commencer. On n'a alors pas besoin d'utiliser beaucoup de vecteurs aléatoires, mais uniquement autant que le niveau d'approximation souhaité.

En termes de temps de calcul, on montre empiriquement sur la Fig. 2.17 que le temps de calcul est linéaire en η. L'extrapolation linéaire de la courbe rejoint le temps de calcul nécessaire en passant par le calcul des corrélations de toutes les ondelettes pour $\eta = N$.

5.2 Comparaison de mesures de stabilité

Montrons tout d'abord la stabilité γ_a mesurée pour le graphe précédent. La Fig. 2.18 montre côte à côte le résultat de la détection de communautés pour $\eta = 60$ (figure a) et l'instabilité $1 - \gamma_a$ associée (figure b). Les trois intervalles d'échelles où les partitions théoriques sont retrouvées exactement correspondent à une stabilité

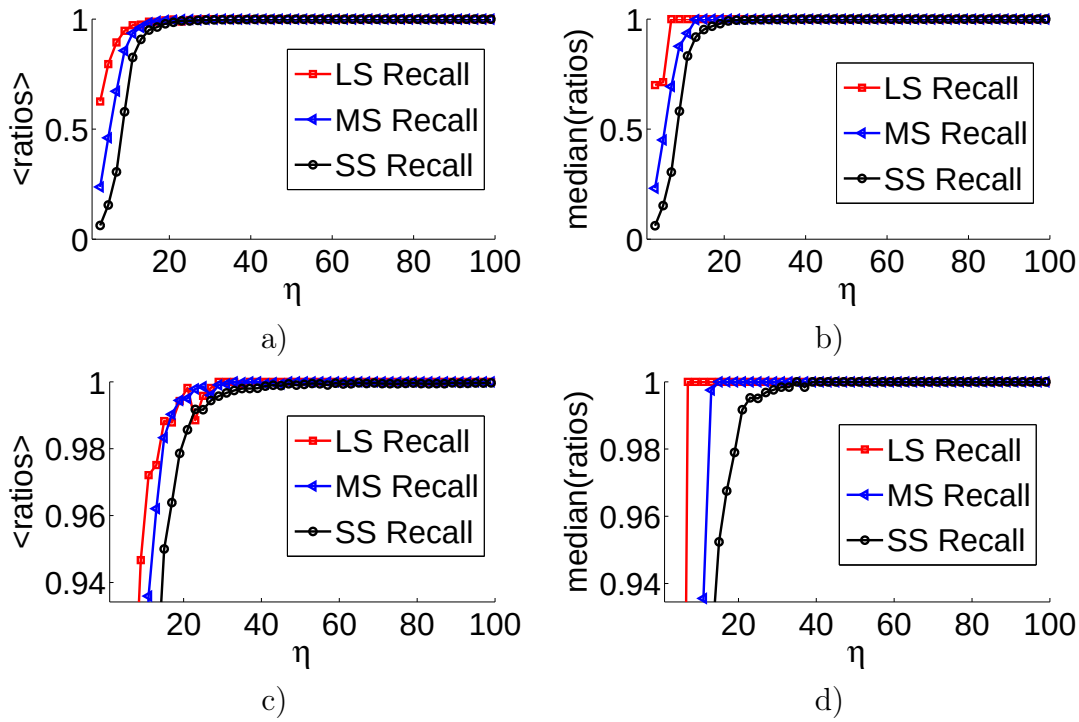


Figure 2.16: a) (resp. b) Moyenne (resp. médiane) des ratios de rappel en fonction du nombre de vecteurs aléatoires calculée sur 100 réalisations de Sales-Pardo. c) et d) sont simplement des zooms autour de 1 pour voir plus précisément comment converge l'algorithme vers la bonne solution quand on augmente η.

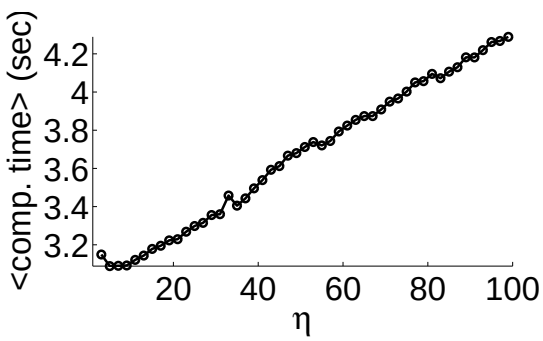
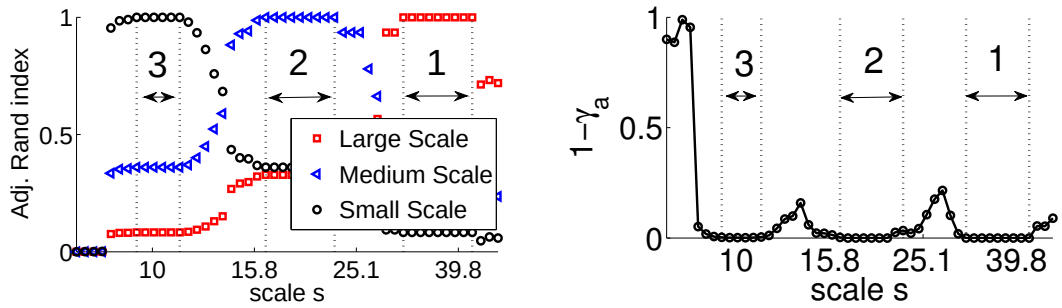


Figure 2.17: Moyenne du temps de calcul en fonction du nombre η de vecteurs aléatoires.



a) b)

FIGURE 2.18: Résultat de l'algorithme comparé à la mesure de stabilité proposée. Les trois intervalles où les partitions théoriques sont retrouvées sont reportés sur la figure d'instabilité : ils correspondent très bien aux intervalles où l'instabilité $1-\gamma_a$ est minimale.

parfaite de 1, c'est-à-dire qu'à ces échelles-là, les $J = 20$ ensembles de $\eta = 60$ vecteurs aléatoires donnent exactement les mêmes résultats. Nous allons comparer cette mesure de stabilité avec deux autres mesures classiques de la littérature :

- Une première stabilité qui entre dans la catégorie des stabilités mesurées par perturbation directe du graphe (voir la partie 3.5.1). Nous utilisons ici la proposition de Lambiotte et al. [127] : créer J versions perturbées du graphe en ajoutant ou retirant $\pm p\%$ à chaque lien. A chaque échelle s, obtenir ainsi une partition par graphe perturbé, c'est-à-dire l'ensemble $\{P_s^j\}_{j\in J}$. La stabilité $\gamma_r(s)$ est alors définie comme la moyenne des similarités calculées deux à deux de toutes les partitions de $\{P_s^j\}_{j\in J}$:

$$\gamma_r(s) = \frac{2}{J(J-1)} \sum_{(i,j)\in J, i\neq j} \texttt{simi}(P_s^i, P_s^j), \tag{2.53}$$

  où nous utiliserons l'indice de Rand ajusté pour la mesure de similarité `simi`. Cette mesure est donc paramétrée par p.
- Une deuxième stabilité qui entre dans la catégorie des stabilités mesurées en terme de largeur d'intervalle d'échelles (voir la partie 3.5.2). Nous utilisons ici aussi la proposition de Lambiotte et al. [127] : à chaque échelle s, on calcule la similarité moyenne entre la partition trouvée à cette échelle et celle trouvée aux échelles voisines. Le voisinage est paramétré par τ et on définit la stabilité $\gamma_v(s)$:

$$\gamma_v(s) = \frac{1}{2*\tau} \sum_{k\in[s-\tau,s+\tau], k\neq s} \texttt{simi}(P_s, P_k). \tag{2.54}$$

On illustre ces instabilités sur la Fig. 2.19 où les partitions $\{P_s\}_{s\in S}$ de la figure a) ont été calculées avec l'algorithme basé sur le calcul intégral des ondelettes et de leurs corrélations. On voit en effet que ces définitions d'instabilité sont pertinentes : les minima globaux correspondent bien aux trois structures en communautés du graphe.

Néanmoins, il y a plusieurs problèmes inhérents à ces mesures. Pour γ_r, tout d'abord, il est coûteux de recalculer pour chacun des J graphes perturbés, le laplacien et sa diagonalisation. De plus, le paramètre p semble *ad hoc* : en effet, selon si on

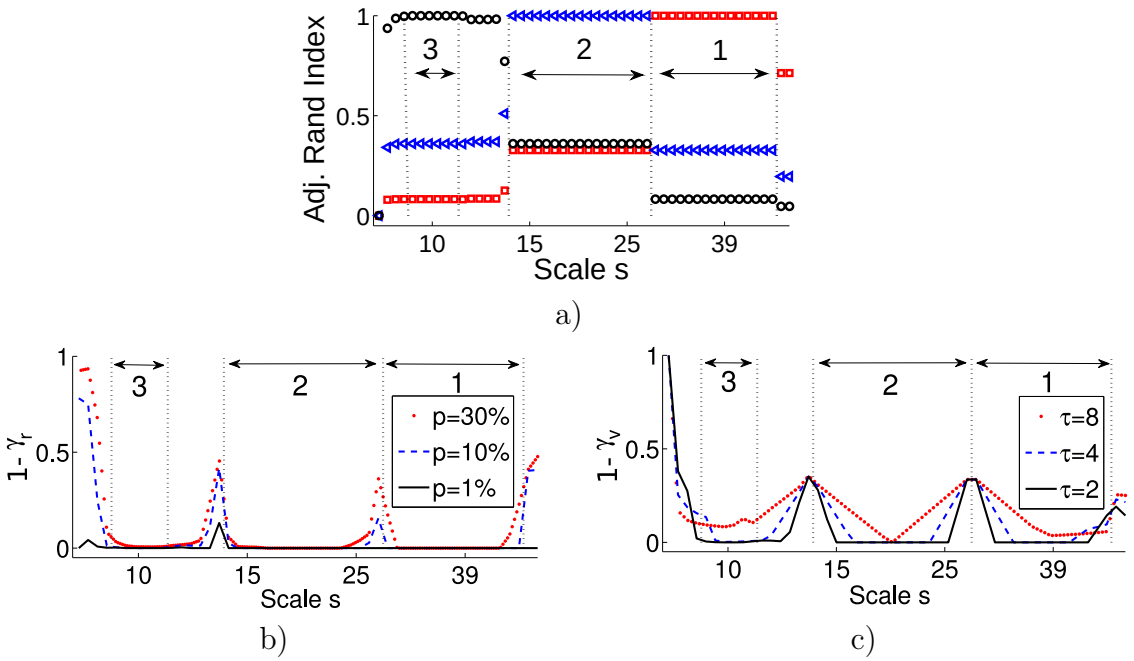


Figure 2.19: a) Résultat de l'algorithme utilisant le calcul de toutes les ondelettes. Les trois intervalles pour lesquels les partitions théoriques sont exactement retrouvées sont reportés sur les figures d'instabilité b) et c). b) Instabilité $1 - \gamma_r$ mesurée en perturbant directement le graphe pour différents degrés de perturbation p. c) Instabilité $1-\gamma_v$ mesurée en comparant les partitions trouvées avec les partitions trouvées aux échelles voisines, pour différentes tailles de voisinage τ.

choisit une petite ou une grande valeur de p, les partitions vont être jugées plus ou moins stables. Aussi, en observant la courbe pour $p = 1\%$, les partitions à petite échelle semblent avoir plus de chances d'être jugées instables que les grandes échelles. En effet, comme expliqué dans la partie 3.5.1, perturber tous les liens de manière équivalente va favoriser les partitions avec de grandes communautés.

Le calcul de γ_v a l'avantage d'être beaucoup moins coûteux étant donné qu'il ne nécessite qu'une seule diagonalisation du laplacien. Ici, la difficulté vient du fait qu'il est impossible de prévoir la valeur de τ à utiliser, d'autant plus que sa valeur dépend de la densité de l'échantillonnage du paramètre d'échelle !

En résumé, la mesure de stabilité γ_a introduite a des performances similaires à l'état de l'art de la littérature, avec le double avantage de ne pas être paramétrée et de s'inscrire naturellement dans l'algorithme rapide de détection de communautés proposé. On pourrait argumenter que la variance σ^2 des vecteurs aléatoires utilisés est un paramètre de la stabilité γ_a, mais comme on ne s'intéresse qu'aux corrélations, la variance n'a en fait aucun impact. Le seul paramètre que l'on pourrait trouver est η mais ce paramètre est inhérent à l'algorithme de détection de communautés et n'est pas ajouté pour la mesure de stabilité.

5.3 Comparaison avec d'autres algorithmes multiéchelles

Nous comparons ici notre méthode avec trois autres méthodes de la littérature résumées dans la partie 3.4 : la méthode d'Arenas [22], la méthode de Reichardt [173] et la méthode de Delvenne [58]. Nous comparons ici uniquement la capacité de chaque méthode à extraire les partitions théoriques de modèles de graphes hiérarchiques. Nous utiliserons d'abord dans la partie 5.3.1 le modèle de Sales-Pardo, puis dans la partie 5.3.2 le modèle LFR (du nom de ses trois auteurs) développé dans [132, 128]. Finalement, dans la partie 5.3.3, nous comparerons les performances de notre algorithme selon que l'on utilise le laplacien normalisé ou le laplacien de marche aléatoire (voir le rappel de ces définitions dans la partie 1.5 du chapitre 1). Afin de comparer les méthodes entre elles, nous allons utiliser un ensemble $\mathcal{S}$ de 50 échelles, logarithmiquement échantillonnées entre les bornes du paramètre. Notons que seule notre méthode propose une borne supérieure et une borne inférieure pour le paramètre d'échelle. La méthode d'Arenas possède une borne inférieure mais pas de borne supérieure et les méthodes de Delvenne et de Reichardt ne possèdent pas de bornes. Pour la méthode d'Arenas, nous décidons d'aller de la borne inférieure des auteurs à $s = 500$. Pour la méthode de Delvenne (resp. Reichardt), l'intervalle d'échelles $s \in [0.01, 100]$ (resp. $s \in [0.1, 100]$) fonctionne bien pour les graphes que nous allons étudier.

Finalement, pour maximiser les fonctions de qualité de ces trois travaux, nous utilisons l'implémentation optimisée de LeMartelot et Hankin [134], disponible sur le site internet de l'auteur [9].

5.3.1 Comparaison sur le modèle de graphes de Sales-Pardo

Nous allons comparer les ratios de rappel des trois partitions théoriques en fonction du jeu de paramètres $(\rho, \bar{k})$ qui contrôle la difficulté de la détection et de l'algorithme utilisé. Nous obtenons les résultats de la Fig. 2.20. Pour tracer ces figures, nous générons pour chaque jeu de paramètres 20 graphes aléatoires, et nous représentons ici la moyenne des ratios de rappel ainsi que l'intervalle à 90%. On peut remarquer que la méthode que nous venons de présenter (ici avec $\eta = 100$ vecteurs aléatoires) a des résultats presque aussi bons que les autres méthodes de l'état de l'art, mais se retrouve quand même dans la majorité des cas légèrement en-deça des autres méthodes. Nous pouvons aussi remarquer que le temps de calcul est, dans ce cas d'un petit graphe (640 nœuds), deux à cinq fois plus court avec notre méthode qu'avec les autres méthodes.

Essayons de comprendre ce résultat. Après investigations, nous constatons qu'il n'est pas nécessaire de détecter les partitions avec plus de vecteurs aléatoires ni même avec les ondelettes calculées complètement. C'est au niveau de la coupe du dendrogramme que la différence se fait. Observons la Fig. 2.21a) qui trace, pour un graphe de Sales-Pardo avec $(\rho = 2, \bar{k} = 11)$, le nombre de communautés trouvées en fonction de l'échelle. On observe une différence de comportement flagrante : notre méthode détecte autant de partitions que de nœuds jusqu'à une échelle élevée et ensuite chute brutalement à un nombre de communautés proche de la valeur théorique à moyenne échelle (16 communautés), en ne détectant pas au passage de partitions

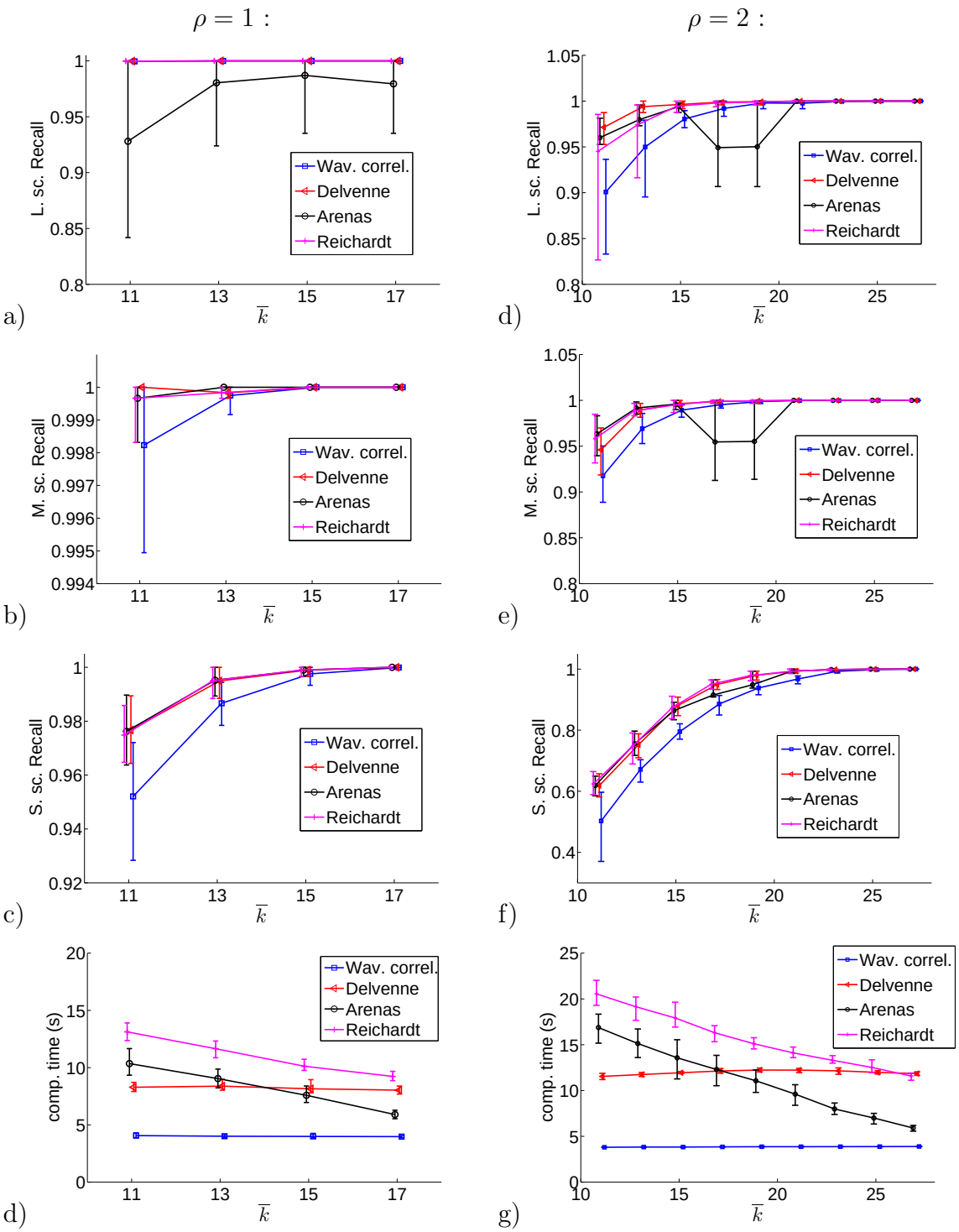


Figure 2.20: Comparaison de différents algorithmes multiéchelles sur le modèle de graphe de Sales-Pardo pour différents jeux de paramètres $(\rho, \overline{k})$. La colonne de gauche (resp. droite) correspond à $\rho = 1$ (resp. $\rho = 2$). La première (resp. deuxième, troisième) ligne représente le ratio de rappel à grande (resp. moyenne, petite) échelle. La dernière ligne compare les temps de calcul pour obtenir ces résultats. Les résultats sont calculés sur 20 graphes, et on représente la moyenne et l'intervalle à 90%.

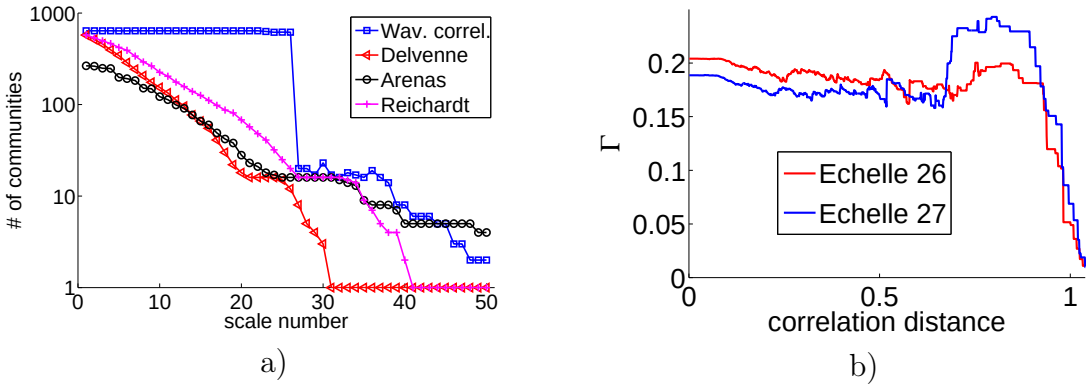


FIGURE 2.21: a) Nombre de communautés de la partition trouvée en fonction de l'échelle, pour différentes méthodes multiéchelles : il existe une différence de comportements flagrante. Le saut observé pour la méthode de corrélations d'ondelettes se trouve entre les échelles numéro 26 et 27 : on représente sur la figure b) la fonction de sauts globale Γ pour ces deux échelles.

à nombre de communautés intérmédiaire. Les autres méthodes ont un comportement plus linéaire : elles ont donc forcément plus de chances de détecter toutes les partitions théoriques. Le saut observé avec notre méthode se fait entre les échelles 26 et 27. Nous montrons, à ces deux échelles, la fonction de sauts globale Γ de leur dendrogramme associé sur la Fig. 2.21b) : on observe en effet que son maximum est à faible distance de corrélation pour $s = s_{26}$ découpant ainsi le dendrogramme en de très nombreuses communautés, et à grande distance de corrélation pour $s = s_{27}$ découpant ainsi le dendrogramme en peu de communautés. Les maxima locaux intermédiaires que l'on peut observer sur ces deux courbes ne sont pas détectés alors que certains d'entre eux correspondent en réalité à la partition théorique à petite échelle. On est donc en droit de se demander si la fonction de sauts globale Γ, définie par l'équation 2.30 est judicieuse et si c'est bien celle-là qu'il faut chercher à maximiser pour savoir où couper le dendrogramme. Toutes les fonctions de saut Γ_a associées à tous les nœuds a forment un faisceau de courbes. Nous considérons ici sa moyenne (Γ) mais on pourrait tout aussi bien considérer son enveloppe maximale Γ_{max} par exemple, et chercher à maximiser cette enveloppe. On montre sur la Fig. 2.22 les performances des algorithmes en fonction que l'on décide de couper le dendrogramme en fonction de Γ, de Γ_{max} ou au niveau du saut maximal entre deux nœuds du dendrograme, comme évoqué dans la partie 4.3.3 (méthode nommée MG – pour l'anglais *maximal gap* – dans la suite). Il n'y a pas de différences bien nettes de performance entre toutes ces possibilités. Nous avons vu dans la partie 4.3.3 que la méthode MG est sensible aux points aberrants et qu'il vaut mieux l'éviter. Dans la suite, nous continuerons d'utiliser la méthode du maximum de Γ.

De futurs travaux devront s'attarder un peu plus longtemps sur la coupe du dendrogramme en s'appuyant éventuellement sur une mesure de qualité globale qui fait le succès des trois autres méthodes. Malgré ces petites différences, on peut néanmoins dire que les quatre méthodes ont des performances similaires sur ce genre de graphes.

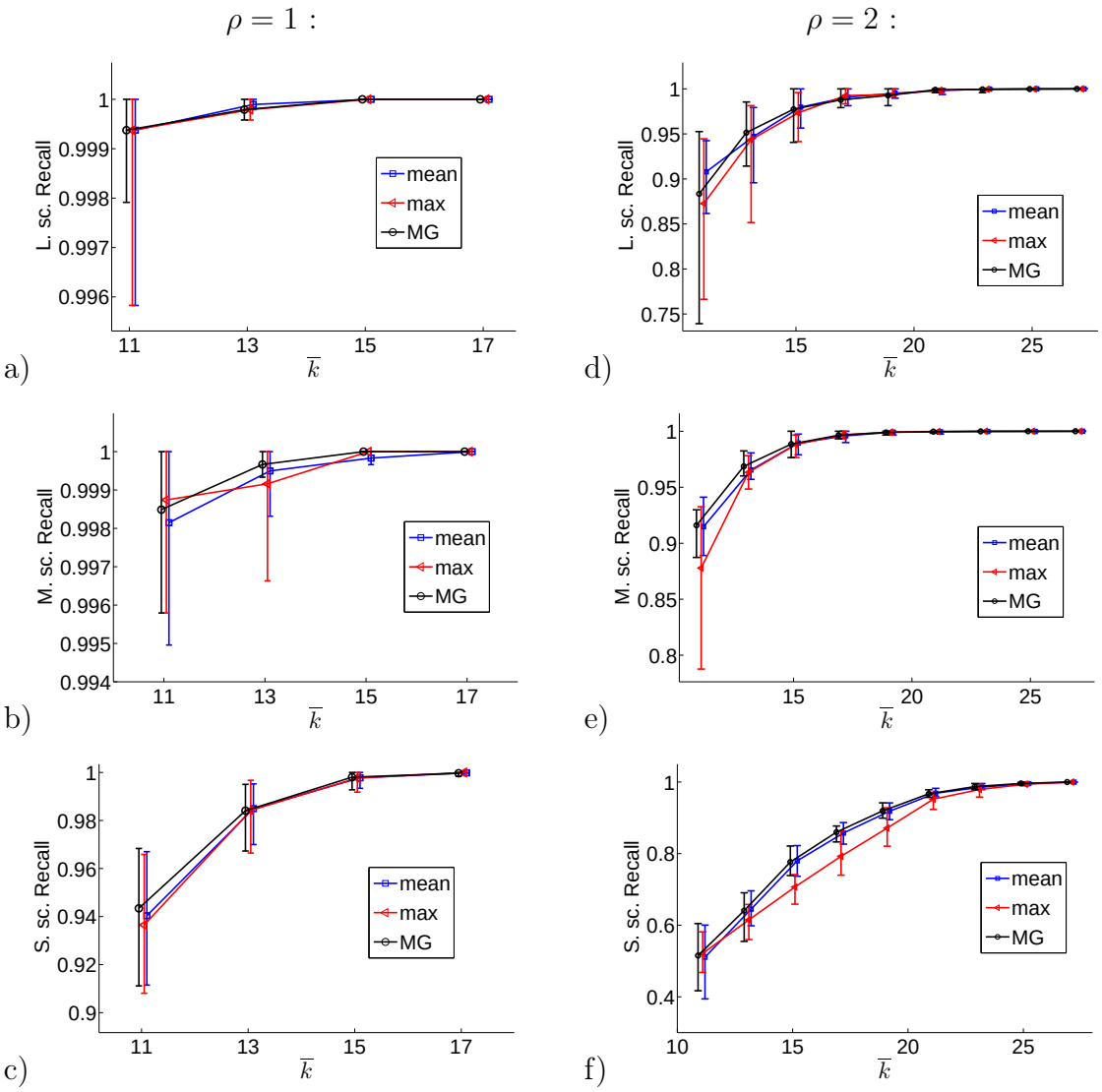


Figure 2.22: Comparaison des résultats sur un Sales-Pardo pour différents jeux de paramètres $(\rho, \overline{k})$ en fonction que l'on décide de couper le dendrogramme à son saut maximal (MG), au maximum de sa fonction de sauts globale Γ (mean), ou au maximum de Γ_{max}, l'enveloppe supérieure du faisceau de courbes Γ_a (max) : il n'y a pas de différences notables.

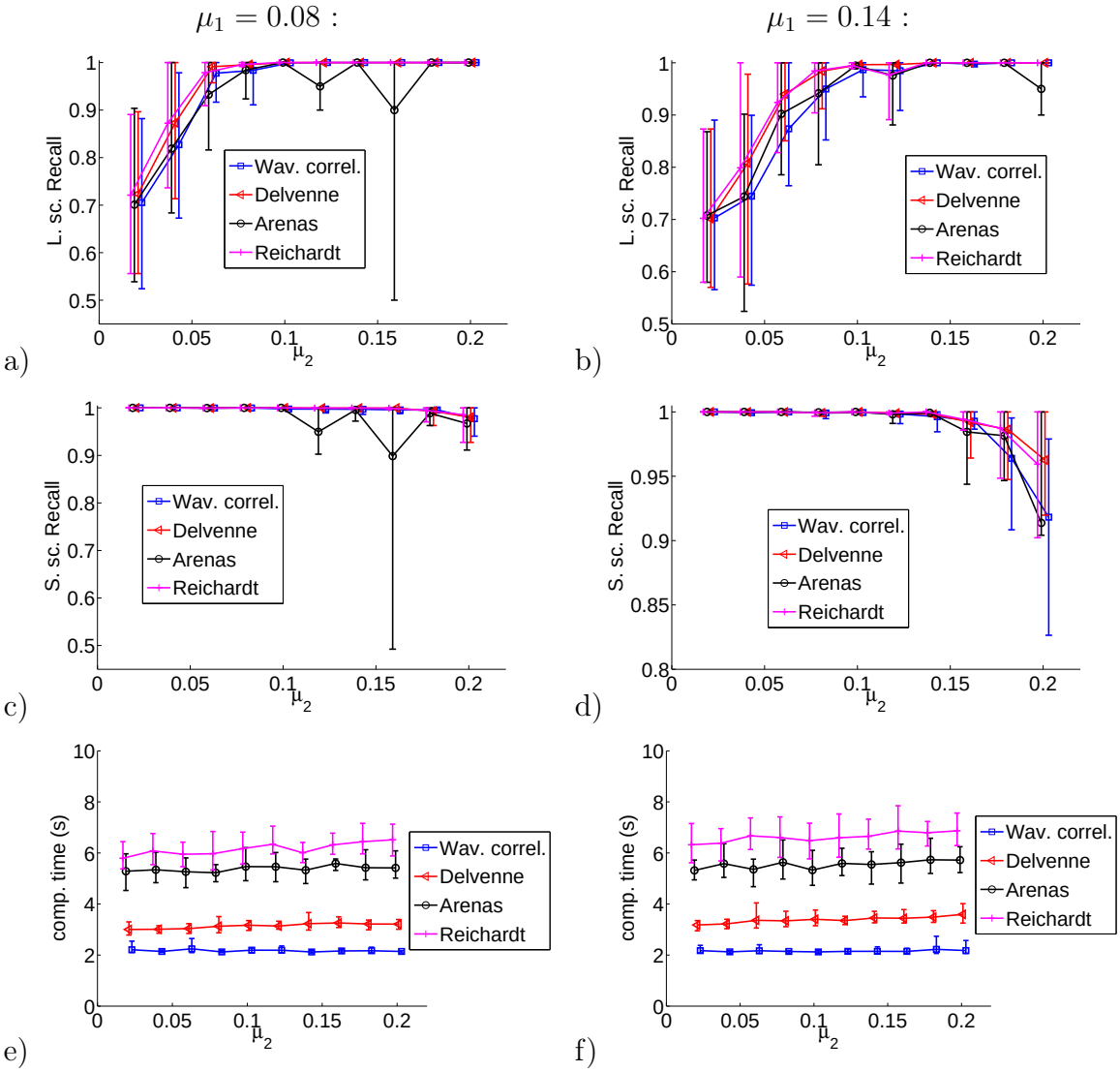


FIGURE 2.23: Comparaison de différents algorithmes multiéchelles sur le modèle de graphe LFR pour différents jeux de paramètres (μ_1, μ_2). La colonne de gauche (resp. droite) correspond à $\mu_1 = 0.08$ (resp. $\mu_1 = 0.14$). La première (resp. deuxième) ligne représente le ratio de rappel à grande (resp. petite) échelle. La dernière ligne compare les temps de calcul pour obtenir ces résultats. Les résultats sont calculés sur 20 graphes, et on représente la moyenne et l'intervalle à 90%.

5.3.2 Comparaison sur un autre modèle de graphes hiérarchiques

Nous comparons à présent les mêmes algorithmes multiéchelles sur un autre modèle hiérarchique moins structuré : le modèle LFR [132, 128]. Ce modèle permet de créer des graphes aléatoires avec une distribution de degrés et une distribution de la taille des communautés en loi de puissance, tout en ayant une structure en communautés à deux échelles. Les codes pour générer les graphes sont disponibles sur le site internet des auteurs [8]. On créé des graphes aléatoires avec les paramètres suivants : $N = 300$ nœuds, un degré moyen de $k = 10$, un degré maximal de $k_{max} = 25$, un exposant de loi de puissance de $t_1 = -2$ pour la distribution des degrés, un exposant

de loi de puissance de $t_2 = -1$ pour la distribution de la taille des communautés, un minimum de $min_c = 10$ nœuds et un maximum de $max_c = 50$ nœuds pour les communautés à petite échelle, et un minimum de $min_C = 20$ nœuds et un maximum de $max_C = 80$ nœuds pour les communautés à grande échelle, et des paramètres de recouvrement de communautés qui interdisent le recouvrement ($on = om = 0$). La difficulté de retrouver les communautés théoriques dépend de deux paramètres de mixage : μ_1 (resp. μ_2) qui quantifie le mixage entre les communautés à grande (resp. petite) échelle. Plus ces paramètres sont grands, plus les deux partitions théoriques sont difficiles à retrouver. Nous comparons les performances sur la Fig. 2.23. Comme précédemment, pour tracer ces figures, nous générons pour chaque jeu de paramètres 20 graphes aléatoires, et nous représentons ici la moyenne des ratios de rappel ainsi que l'intervalle à 90%. Pour ce modèle de graphes, les performances des quatre algorithmes sont plus homogènes, à l'exception peut-être de l'algorithme d'Arenas qui a parfois des valeurs aberrantes.

5.3.3 Laplacien normalisé ou laplacien de marche aléatoire ?

Dans la communauté du clustering spectral [221], et comme discuté précédemment dans la partie 2.6.1 du chapitre 1, il est communément admis que les laplaciens normalisés donnent de meilleurs résultats que le laplacien classique. La question est éventuellement de savoir lequel des laplaciens normalisés est le plus adapté à notre algorithme : $\boldsymbol{\mathcal{L}} = \boldsymbol{S}^{-\frac{1}{2}}\boldsymbol{L}\boldsymbol{S}^{-\frac{1}{2}}$ ou $\boldsymbol{\mathcal{L}}_{rw} = \boldsymbol{S}^{-1}\boldsymbol{L}$? Von Luxburg [221] écrit que le laplacien de marche aléatoire $\boldsymbol{\mathcal{L}}_{rw}$ est souvent plus efficace. Est-ce le cas ici ? Nous comparons sur la Fig. 2.24 les performances de l'algorithme en fonction du laplacien utilisé. Nous observons que les résultats ne dépendent pas du laplacien choisi.

5.4 Illustration du test statistique

Illustrons le test statistique sur deux exemples :

a. L'exemple de la Fig. 2.18.

b. Une version rendue aléatoire (comme expliqué dans l'annexe B) de cet exemple.

Le résultat du test est en fait une valeur seuil de stabilité au-delà de laquelle les partitions sont considérées pertinentes. Nous montrons sur la Fig. 2.25 les stabilités avec la valeur seuil (sous forme de tirets horizontaux) donnée par le test pour les deux exemples ci-dessus.

La vérité de terrain du cas a) est qu'au moins les échelles au sein des trois intervalles sont au-dessus de la valeur seuil de stabilité, ce qui est le cas (rappelons que nous traçons $1-\gamma_a$ sur les figures). Il y a des faux positifs, c'est-à-dire des échelles qui sont plus stables que la valeur seuil mais dont la partition associée n'est pas une des trois partitions théoriques. Ceci est en fait attendu étant donné que certaines partitions entre deux intervalles sont des combinaisons de partitions théoriques, et sont donc plus stables que des partitions dans un graphe aléatoire. Dans le cas b), la vérité de terrain est qu'aucune des partitions devraient être plus stable que la valeur seuil étant donné que le graphe est aléatoire : ce qui est le cas.

Dans la suite, les partitions qui ne sont pas plus stables que la valeur seuil ne sont pas considérées, et quand des intervalles d'échelles entiers sont au-dessus du

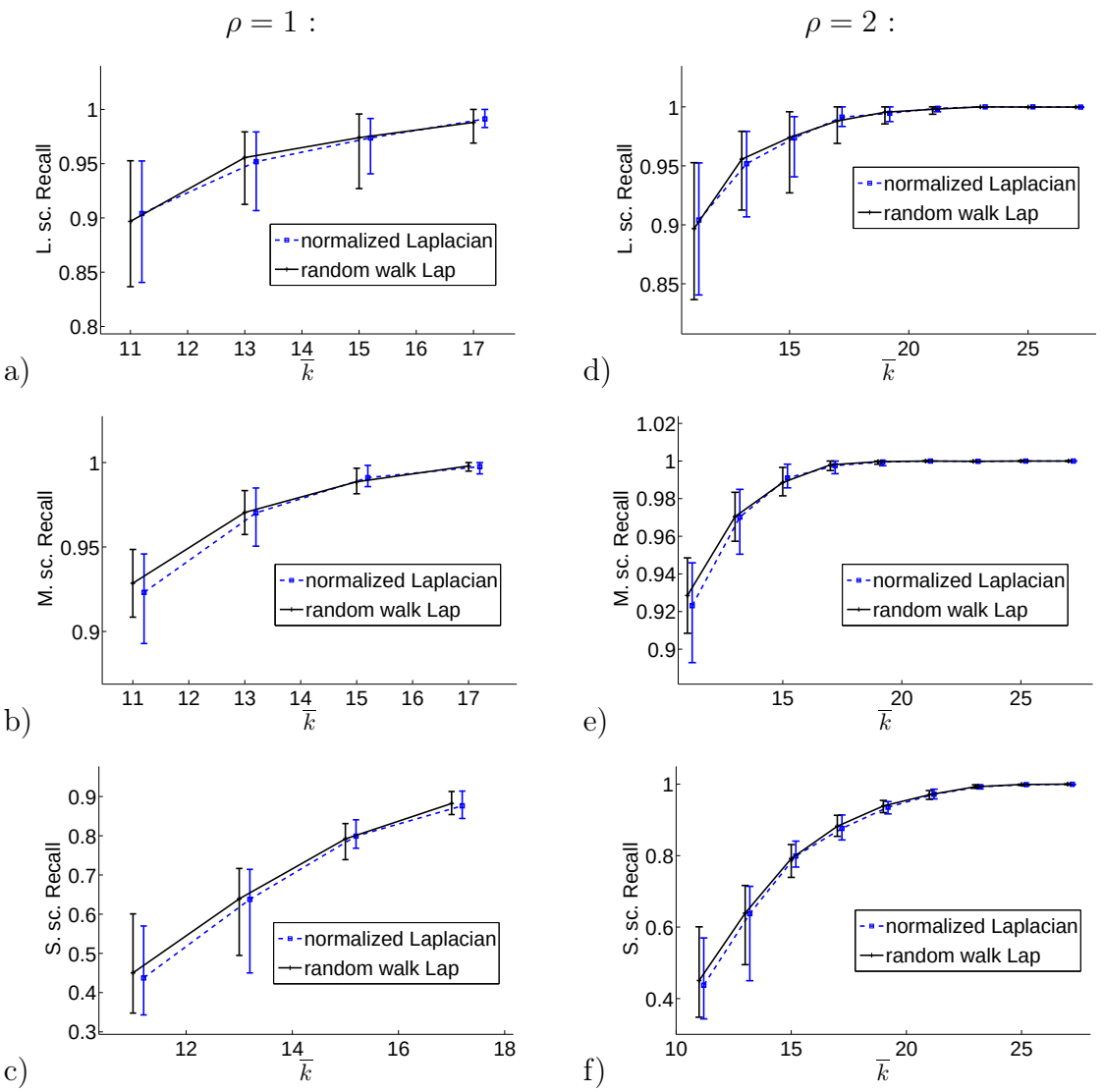


FIGURE 2.24: Comparaison de performances selon que l'algorithme utilise le laplacien normalisé $\boldsymbol{\mathcal{L}} = \boldsymbol{S}^{-\frac{1}{2}}\boldsymbol{L}\boldsymbol{S}^{-\frac{1}{2}}$ ou le laplacien normalisé de "marche aléatoire" $\boldsymbol{\mathcal{L}}_{rw} = \boldsymbol{S}^{-1}\boldsymbol{L}$, sur des graphes de Sales-Pardo. Les résultats sont globalement équivalents.

seuil, on gardera le maximum local significatif (maximum local de γ_a ou minimum local de $1 - \gamma_a$).

5.5 Illustration sur un modèle de graphe avec une seule échelle

Il est très intéressant d'observer le comportement de l'algorithme dans le cas où il n'existe qu'une seule échelle de description. Nous allons de nouveau utiliser ici le modèle LFR, mais dans sa version non-hiérarchique [132]. Il existe donc dans ce modèle une seule échelle pertinente, i.e. une seule partition théorique. Afin de comparer à d'autres méthodes de détection, nous utilisons les mêmes paramètres que dans [131] pour créer les graphes : un degré moyen de $k = 20$, un degré maximal de $k_{max} = 50$, un exposant de loi de puissance de $t_1 = -2$ pour la distribution

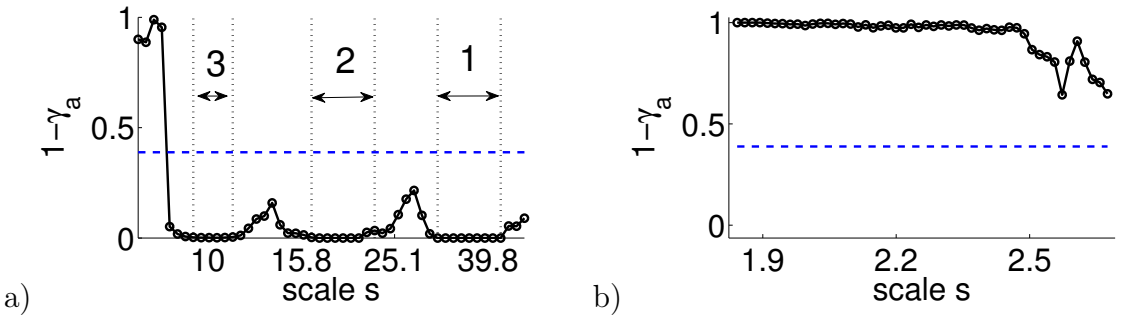


Figure 2.25: a) Illustration du test statistique sur un graphe de Sales-Pardo. La valeur seuil de l'instabilité $1-\gamma_a$ est représentée en pointillés horizontaux : toute valeur d'instabilité en-deça de ce seuil correspond à une partition jugée pertinente. Les trois intervalles d'échelles où les partitions théoriques sont exactement retrouvées sont bien jugées pertinentes. b) Même calcul pour une version rendue aléatoire du graphe utilisé pour la Fig. a) : sans surprises, aucune partition n'est jugée pertinente.

des degrés, un exposant de loi de puissance de $t_2 = -1$ pour la distribution de la taille des communautés, et des paramètres de recouvrement de communautés qui interdisent le recouvrement ($on = om = 0$). Suivant [131], nous allons étudier 4 cas différents :

- Le cas 1000_S : $N = 1000$ nœuds avec des petites communautés : un minimum de $min_c = 10$ nœuds et un maximum de $max_c = 50$ nœuds par communauté.
- Le cas 5000_S : $N = 5000$ nœuds avec des petites communautés : un minimum de $min_c = 10$ nœuds et un maximum de $max_c = 50$ nœuds par communauté.
- Le cas 1000_B : $N = 1000$ nœuds avec des communautés plus grandes : un minimum de $min_c = 20$ nœuds et un maximum de $max_c = 100$ nœuds par communauté.
- Le cas 5000_B : $N = 5000$ nœuds avec des communautés plus grandes : un minimum de $min_c = 20$ nœuds et un maximum de $max_c = 100$ nœuds par communauté.

Attardons-nous un instant sur le cas 1000_S et générons une réalisation pour différents niveaux de difficulté d'extraction de la partition théorique (différentes valeurs du paramètre de mixage μ) : $\mu = 0.2$, 0.4, 0.6, 0.65, 0.7 et 0.75. Pour chacune de ces réalisations, nous lançons notre algorithme avec $\eta = 100$ vecteurs aléatoires et nous faisons aussi le test statistique. Nous obtenons les résultats récapitulés sur la Fig. 2.26 : pour chaque valeur de μ, nous traçons la stabilité en fonction de l'échelle, et nous sélectionnons le minimum local significatif de $1-\gamma_a$ en-dessous du seuil donné par le test : il est représenté en rouge (si il y a un plateau minimal alors il y a plusieurs points rouges). S'il n'y a pas d'intervalle d'échelles en dessous du seuil, aucune échelle n'est sélectionnée : l'algorithme ne trouve pas de solution significative. Nous observons que lorsqu'une échelle est sélectionnée car jugée stable par l'algorithme, alors la partition correspondante est égale à la partition théorique, ou dans les cas difficiles, proche de la partition théorique. On observe ici que jusqu'à $\mu = 0.65$, la solution trouvée par l'algorithme est à peu près juste, et au-delà l'algorithme ne trouve pas de solution. Ce comportement est très satisfaisant et correspond bien à ce qu'on attend du test statistique.

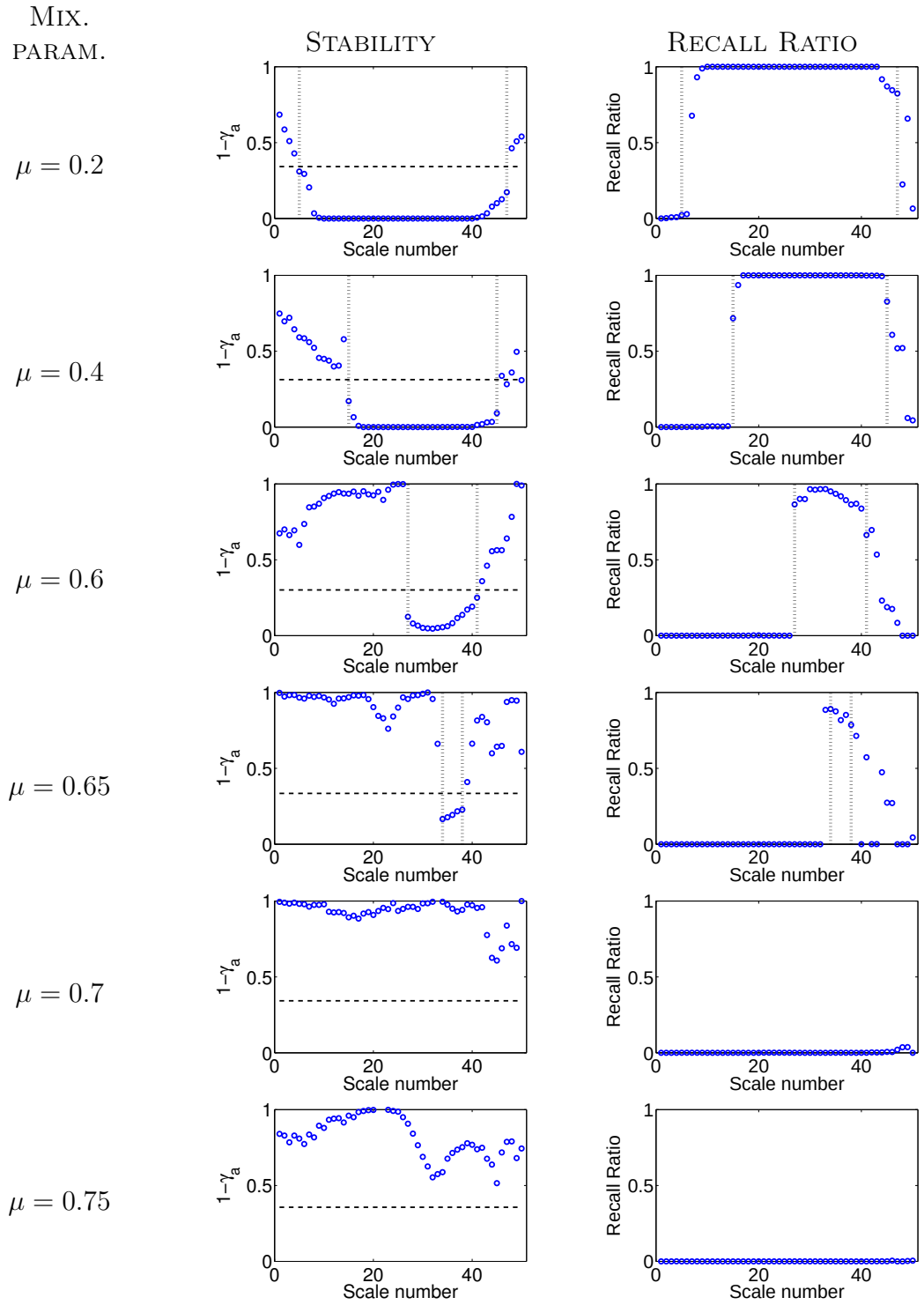


Figure 2.26: Résultat de l'algorithme sur une réalisation d'un modèle de graphes mono-échelles, paramétré par μ. Pour différentes valeurs de ce paramètre, nous traçons la stabilité de la partition obtenue à chaque échelle (colonne du milieu), ainsi que le ratio de rappel de la partition théorique à chaque échelle (colonne de droite). Nous représentons en rouge le minimum local significatif de $1 - \gamma_a$ en-dessous du seuil donné par le test statistique (pointillés horizontaux). S'il n'y a pas d'échelles en-dessous du seuil, aucune échelle n'est sélectionnée : l'algorithme ne trouve pas de solution pertinente.

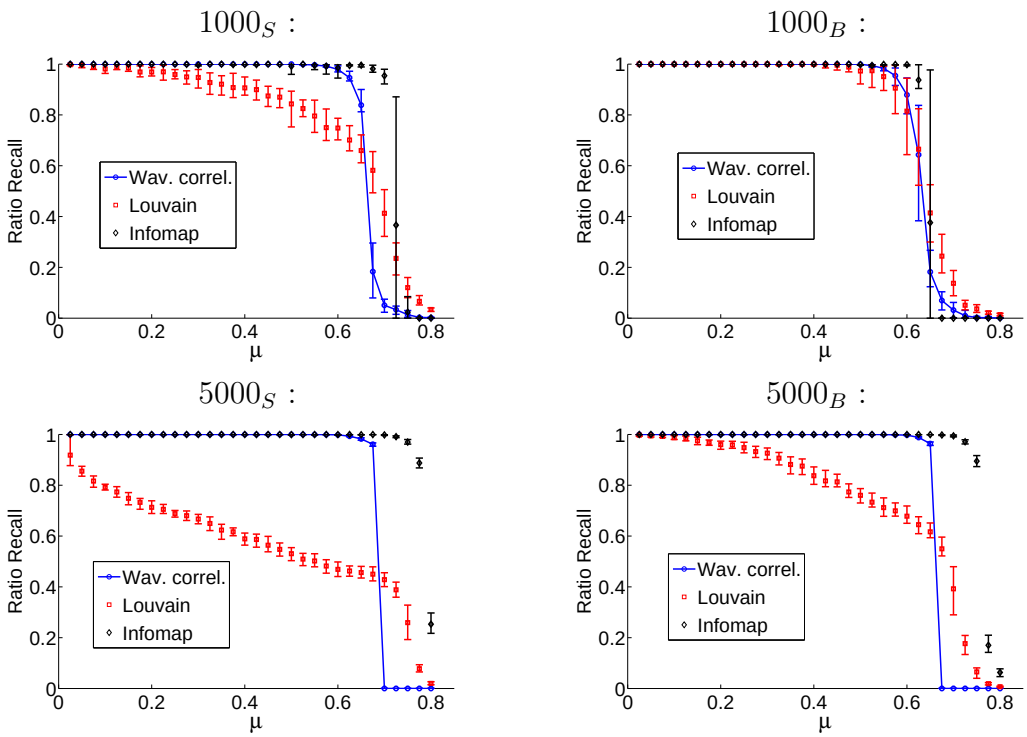


Figure 2.27: Comparaison de notre algorithme avec deux autres algorithmes de la littérature sur un modèle de graphes mono-échelle, paramétré par μ. On représente les moyennes et les intervalles à 90% sur 20 réalisations pour chaque paramètre μ. Les quatre différentes figures correspondent aux quatre différents cas présentés dans le texte.

Nous avons montré une réalisation par valeur de paramètre μ. Générons à présent 20 réalisations de chacun des 4 cas listés ci-dessus, pour un paramètre de mixage μ échantillonné entre 0.025 (cas le plus facile) et 0.8 (cas le plus difficile). Pour chacune de ces réalisations, nous calculons le ratio de rappel de la partition théorique et traçons la performance moyenne sur la Fig. 2.27. Nous comparons le résultat de notre algorithme avec deux algorithmes jugés très performants sur ce type d'exemples par Lancichinetti et al. [131] : l'algorithme de Louvain basé sur la modularité et `infomap`, la méthode de Rosvall et Bergstrom [180]. Ces figures peuvent être directement comparées à la Fig. 2 de [131]. Comme ces auteurs, nous observons que la méthode de Louvain, basée sur la modularité, devient moins performante quand la taille des communautés est hétérogène ou quand la taille typique des communautés est petite devant le nombre total de nœuds. On retrouve aussi que la méthode `infomap` est très efficace sur ce type de graphes, légèrement plus efficace que notre algorithme sur ces paramètres là.

Prenons garde de ne pas généraliser ces performances à tout type de graphe ! Dans le domaine de la détection de communautés, gardons à l'esprit que les performances sont toujours mesurées sur un type de graphe particulier, qui favorise forcément une méthode plutôt qu'une autre. A titre d'exemple, étudions le cas où les tailles de communautés sont très hétérogènes ($N = 1000$ nœuds, un minimum de $min_c = 10$

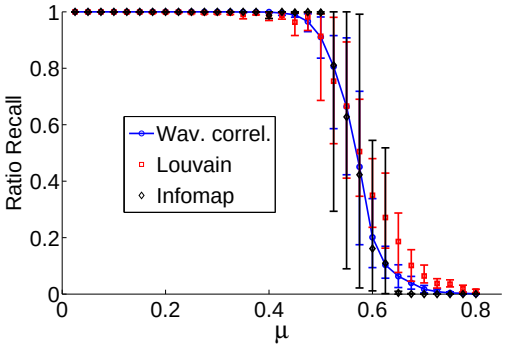


FIGURE 2.28: Comparaison de notre algorithme avec deux autres algorithmes de la littérature sur un modèle de graphes mono-échelle, paramétré par μ, pour un graphe avec des tailles de communautés très hétérogènes.

nœuds et un maximum de $max_c = 200$ nœuds dans chaque communauté). Les résultats sont présentés sur la Fig. 2.28 : on trouve, dans ce cas, que les trois méthodes ont des performances similaires !

5.6 Conclusion

Plusieurs éléments sont à retenir de cette partie :

- La notion de stabilité γ_a que nous introduisons est performante. Elle est non-paramétrée et s'appuie sur la stochasticité de l'algorithme.
- La capacité de notre algorithme à retrouver des partitions à différentes échelles dans des modèles de graphes multiéchelles est légèrement en-deça des meilleures méthodes de la littérature.
- Après investigations, nous avons remarqué que c'est au niveau de la coupe du dendrogramme qu'il faudra améliorer la méthode.
- Le test statistique que nous avons développé a un comportement très satisfaisant.

6 Application à un graphe de terrain

Dans cette partie, nous appliquons la méthode à un graphe de terrain. Il s'agit d'un graphe d'interactions sociales mesuré à l'aide de la plateforme de mesure Sociopatterns [10]. Les détails de cette plateforme sont donnés dans la partie 4.1.1 du chapitre 3. Dans le cas qui nous intéresse ici, nous allons travailler avec la matrice d'adjacence pondérée de taille $N \times N$ où N est le nombre de personnes participant à l'expérience. W_{ij} est ici le temps cumulé de contact entre les personnes i et j pendant toute la durée de l'expérience. Deux personnes sont dites "en contact" à un certain moment si elles sont face-à-face et à moins d'$1,5$m de distance l'une de l'autre.

La méthode est utilisée sur un graphe de contacts entre 242 enfants et professeurs des écoles d'une école primaire, mesuré en 2009 [202]. L'expérience a duré deux jours complets et nous travaillons donc avec le graphe de contacts cumulés sur les deux

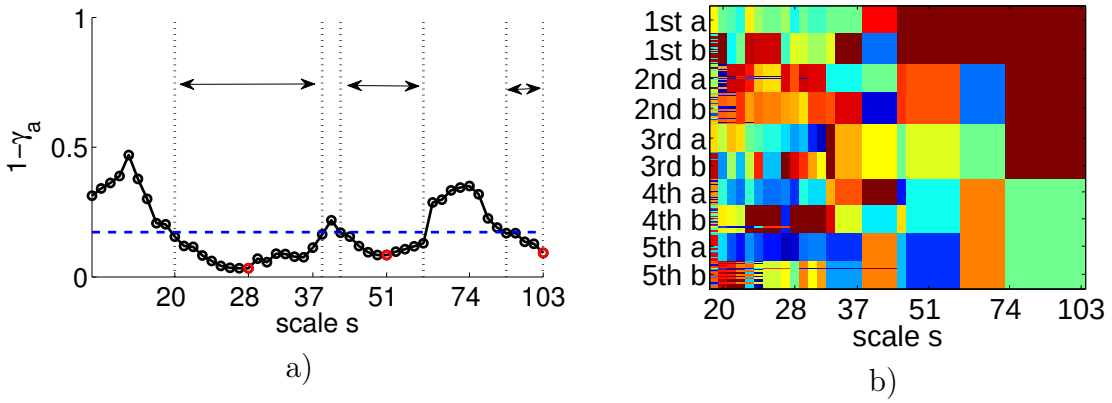
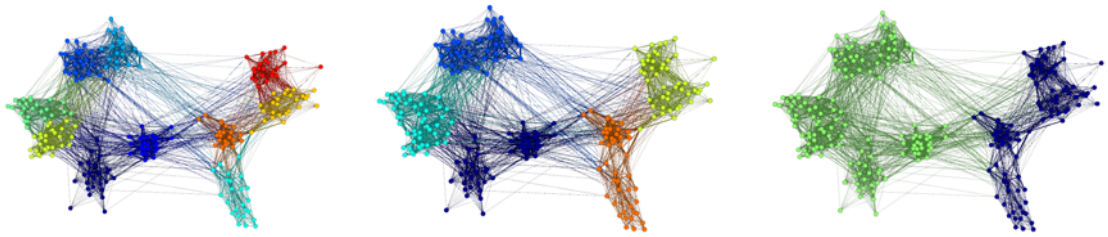


Figure 2.29: a) Résultat du test de stabilité pour le graphe de contacts dans une école primaire. $\eta = 30$ vecteurs alétoires sont utilisés. La ligne horizontale pointillée bleue correspond à la valeur seuil de stabilité. Les trois points rouges représentent les minima significatifs des trois intervalles d'échelle en-dessous du seuil : ce sont les trois partitions pertinentes détectées par la méthode. b) montre les partitions associées à chaque paramètre d'échelle : en ordonnée de la matrice, les nœuds du graphe sont placés dans l'ordre de leur classe respective. Chaque colonne correspond à la partition trouvée au paramètre d'échelle correspondant. Dans une colonne, deux nœuds ont la même couleur s'ils appartiennent à la même communauté. Les partitions correspondant aux huit premiers paramètres d'échelle ne sont pas représentées ici : elles contiennent trop de communautés pour ce mode d'illustration.

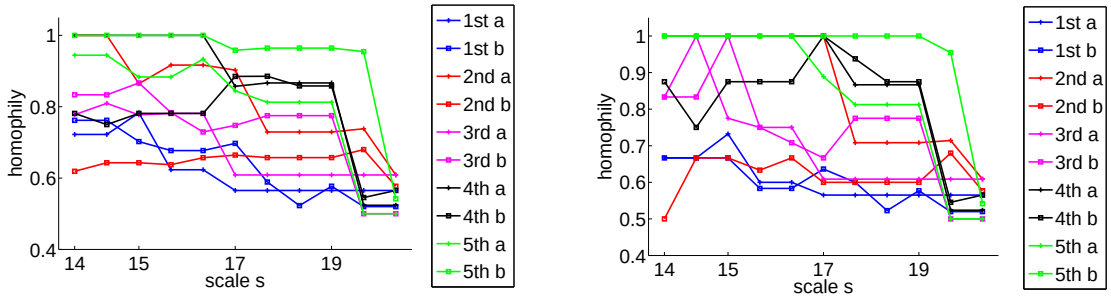
jours. L'école primaire est composée de cinq niveaux : du CP au CM2, et il y a deux classes par niveau. Le résultat de l'algorithme est montré sur la Fig. 2.29. Nous sélectionnons un minimum local de $1 - \gamma_a$ par intervalle d'échelles en dessous du seuil : ce sont les trois échelles représentées en rouge. Chacune de ces trois échelles est en effet bien représentative de leur intervalle d'échelles : l'indice de similarité moyen des partitions de l'intervalle à petite (resp. moyenne, grande) échelle est de 0.93 (0.92, 1). À grande échelle ($s = 103$), les enfants plus âgés (CM1 et CM2) sont séparés des plus jeunes (CP, CE1 et CE2) : il n'y a que deux grandes communautés. À une échelle intermédiaire ($s = 51$), chaque niveau est séparé des autres : il existe 5 communautés. Finalement, à une petite échelle ($s = 28$), l'algorithme détecte une partition en 10 communautés qui sépare chaque classe des autres. Ces trois partitions sont illustrées sur la Fig. 2.30.

Il est également intéressant de regarder des échelles un peu moins stables entre $s = s_{min} = 14$ et $s = 20$. À ces échelles, le paramètre d'échelle est assez petit pour commencer à séparer les groupes au sein même des classes. On observe alors un phénomène intéressant illustré sur la Fig. 2.31. Nous traçons, pour chaque classe, la moyenne (figure de gauche) et la médiane (figure de droite) de la proportion d'individus de même sexe dans chaque communauté de la classe (on ne prend en compte que les communautés qui contiennent au moins 2 individus). Cet indicateur vaut 1 si tous les groupes au sein d'une classe sont unisexes et tendent vers 0.5 si tous les groupes au sein d'une classe sont paritaires. Mise à part la classe de CE1a (2nd a) qui a un comportement un peu différent des autres petites classes, on observe



(1) (2) (3)

FIGURE 2.30: Les trois partitions stables détectées par le test statistique, qui correspondent aux trois points rouges de la Fig. 2.29. La figure 1 (resp. 2, 3) montre une partition en 10 (resp. 5, 2) communautés (les nœuds de même couleur sont dans la même communauté). Le graphe est illustré ici en utilisant `ForceAtlas2` implémenté dans Gephi [27].



a) b)

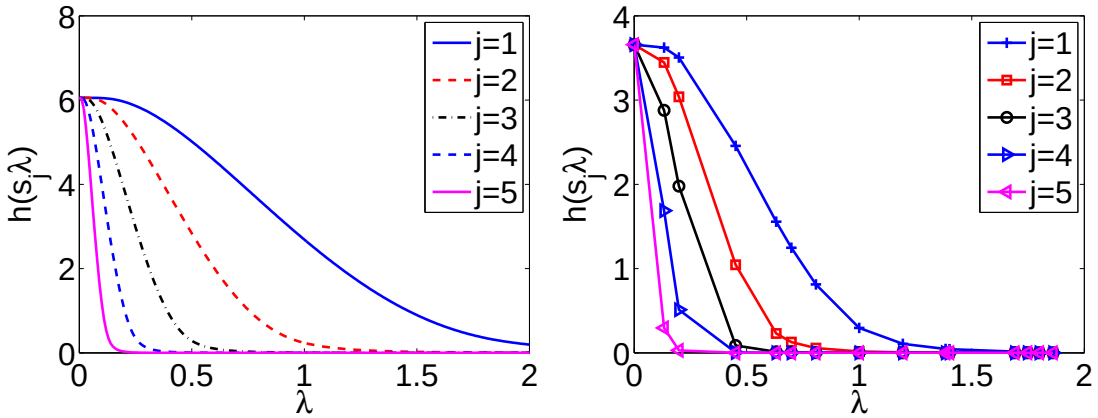
FIGURE 2.31: Pour les petites échelles (entre $s_{min} = 14$ et $s = 20$) où les communautés trouvées découpent les classes en petits groupes, et pour chaque classe, la figure de gauche (resp. droite) montre la moyenne (resp. médiane) de la proportion d'individus de même sexe dans chaque communauté de la classe (on ne prend en compte que les communautés qui contiennent au moins 2 individus). Le point tout à droite de chaque figure montre, pour chaque classe, le pourcentage d'individus de même sexe, c'est-à-dire $\max\left(\frac{\#\text{ filles}}{\#\text{ tot}}, \frac{\#\text{ garçons}}{\#\text{ tot}}\right)$ ce qui permet, au moins à l'œil, de normaliser les courbes tracées.

que plus les élèves des classes sont âgés, plus les communautés intra-classe tendent à être unisexes ; inversement, plus les enfants sont jeunes, plus leurs communautés intra-classe sont paritaires. Le point tout à droite de chaque figure montre le pourcentage d'individus de même sexe, c'est-à-dire $\max\left(\frac{\#\text{ filles}}{\#\text{ tot}}, \frac{\#\text{ garçons}}{\#\text{ tot}}\right)$ pour chaque classe, ce qui permet de normaliser, au moins à l'œil, les courbes tracées. Ce phénomène dit d'homophilie est bien connu en sciences sociales [203].

7 Définition et utilisation de fonctions d'échelle sur graphe

Définition. Nous introduisons ici une définition possible de fonction d'échelle et nous montrons comment elles peuvent remplacer les ondelettes dans l'algorithme

Banc de filtres continus : Banc de filtres discrets :



a) b)

Figure 2.32: Bancs de filtres passe-bas continus (a) et discrets (b) associés aux bancs de filtres passe-bande d'ondelettes de la Fig. 1.10.

présenté. L'équation 4.16 de [42] donne l'équation du noyau de filtre passe-bas définissant des fonctions d'échelles dans le cadre des ondelettes continues :

$$|\hat{\phi}(\omega)|^2 = \int_1^{+\infty} |\hat{\psi}(s\omega)|^2 \frac{ds}{s} = \int_\omega^{+\infty} \frac{|\hat{\psi}(\xi)|^2}{\xi} d\xi, \tag{2.55}$$

où $\hat{\psi}$ est le noyau de filtre d'ondelettes continues et ω la fréquence dans l'espace de Fourier. Par analogie, et de la même manière que les ondelettes sur graphes ont été définies, nous proposons le noyau de filtre passe-bas h définissant les fonctions d'échelles sur graphe suivant :

$$h(\lambda) = \left(\int_\lambda^{+\infty} \frac{g(x)^2}{x} dx \right)^{1/2}, \tag{2.56}$$

où g est le noyau de filtre d'ondelettes sur graphe. En notant :

$$\boldsymbol{H}_s = \mathrm{diag}(h(s\lambda_1)|h(s\lambda_2)|\cdots|h(s\lambda_N)) \tag{2.57}$$

la matrice de filtre passe-bas, on peut écrire la fonction d'échelle $\boldsymbol{\phi}_{s,a}$ centrée sur le nœud a et à l'échelle s sous forme matricielle :

$$\boldsymbol{\phi}_{s,a} = \boldsymbol{\chi} \boldsymbol{H}_s \boldsymbol{\chi}^\top \boldsymbol{\delta}_a. \tag{2.58}$$

La matrice de fonctions d'échelle sur graphe à l'échelle s, notée $\boldsymbol{\Phi}_s$ s'écrit :

$$\boldsymbol{\Phi}_s = (\boldsymbol{\phi}_{s,1}|\boldsymbol{\phi}_{s,2}|\cdots|\boldsymbol{\phi}_{s,N}) = \boldsymbol{\chi} \boldsymbol{H}_s \boldsymbol{\chi}^\top. \tag{2.59}$$

Nous montrons sur la Fig. 2.32 les bancs de filtres passe-bas continus et discrets associés aux bancs de filtres passe-bande d'ondelettes de la Fig. 1.10.

Afin d'illustrer quelques fonctions d'échelles, nous présentons sur la Fig. 2.33 les fonctions d'échelles associées à chacune des ondelettes de la Fig. 1.11. Remarquons

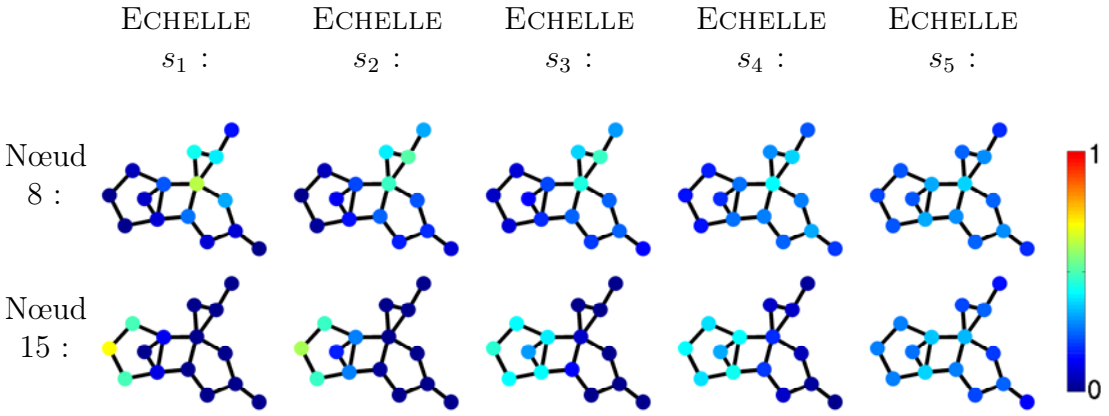


Figure 2.33: Fonctions d'échelles associées à chacune des ondelettes de la Fig. 1.11.

qu'elles sont toujours positives (c'est dû au fait que h atteint toujours son maximum pour la valeur propre $\lambda = 0$ qui correspond au seul vecteur propre toujours positif).

Utilisation. Dans l'algorithme présenté, nous avons choisi de calculer les corrélations d'ondelettes pour mesurer des distances entre les nœuds. La distance entre deux nœuds a et b s'écrivaient (voir l'équation 2.27) :

$$\forall (a,b) \in \mathcal{V}^2 \qquad \mathcal{D}_s(a,b) = 1 - \mathrm{Cor}(\boldsymbol{\psi}_{s,a}, \boldsymbol{\psi}_{s,b}) = 1 - \frac{\boldsymbol{\psi}_{s,a}^\top \boldsymbol{\psi}_{s,b}}{||\boldsymbol{\psi}_{s,a}||_2 ||\boldsymbol{\psi}_{s,b}||_2}. \tag{2.60}$$

Nous pouvons tout à fait envisager à présent d'utiliser les corrélations de fonctions d'échelle. On a, dans ce cas, la formule suivante (légèrement plus compliquée car les fonctions d'échelles ne sont pas de moyenne nulle) :

$$\forall (a,b) \in \mathcal{V}^2 \qquad \mathcal{D}_s(a,b) = 1 - \mathrm{Cor}(\boldsymbol{\phi}_{s,a}, \boldsymbol{\phi}_{s,b}) = 1 - \frac{(\boldsymbol{\phi}_{s,a} - \bar{\boldsymbol{\phi}}_{s,a})^\top (\boldsymbol{\phi}_{s,b} - \bar{\boldsymbol{\phi}}_{s,b})}{||\boldsymbol{\phi}_{s,a} - \bar{\boldsymbol{\phi}}_{s,a}||_2 ||\boldsymbol{\phi}_{s,b} - \bar{\boldsymbol{\phi}}_{s,b}||_2}, \tag{2.61}$$

où $\bar{\boldsymbol{\phi}}_{s,a}$ (resp. $\bar{\boldsymbol{\phi}}_{s,b}$) est le vecteur constant égal à la moyenne de $\boldsymbol{\phi}_{s,a}$ (resp. $\boldsymbol{\phi}_{s,b}$). Rappelons que, quand on utilise le laplacien normalisé, la moyenne s'écrit (voir la discussion autour de l'équation 2.28) : $< \boldsymbol{\phi}_{s,a} >= \boldsymbol{\chi}_1^\top \boldsymbol{\phi}_{s,a}$.

Une autre possibilité pour la matrice de distance $\mathcal{D}_s$ est de calculer la corrélation de l'énergie des ondelettes :

$$\forall (a,b) \in \mathcal{V}^2 \qquad \mathcal{D}_s(a,b) = 1 - \mathrm{Cor}(\boldsymbol{\psi}_{s,a}^2, \boldsymbol{\psi}_{s,b}^2). \tag{2.62}$$

Nous comparons sur la Fig. 2.34 le résultat de l'algorithme sur une réalisation d'un graphe de Sales-Pardo (avec $\rho = 1$ et $\bar{k} = 16$) selon que l'on décide de calculer les corrélations d'ondelettes, d'énergies d'ondelettes, ou de fonctions d'échelles : on obtient le même type de résultats.

Pour être plus complet, nous comparons ces trois manières de calculer la matrice de distance sur le modèle de Sales-Pardo en faisant varier les paramètres. Nous montrons les performances moyennes sur la Fig. 2.35. Observons que les fonctions d'échelles détectent mieux les partitions à grande échelle, les énergies d'ondelettes

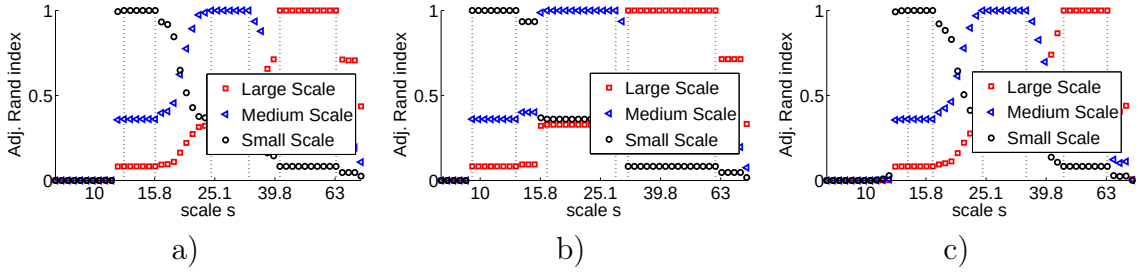


Figure 2.34: Comparaison de trois notions de distances sur la même réalisation d'un graphe de Sales-Pardo. La distance utilisée est a) la corrélation d'ondelettes, b) la corrélation de fonctions d'échelles, ou c) la corrélation des énergies des ondelettes.

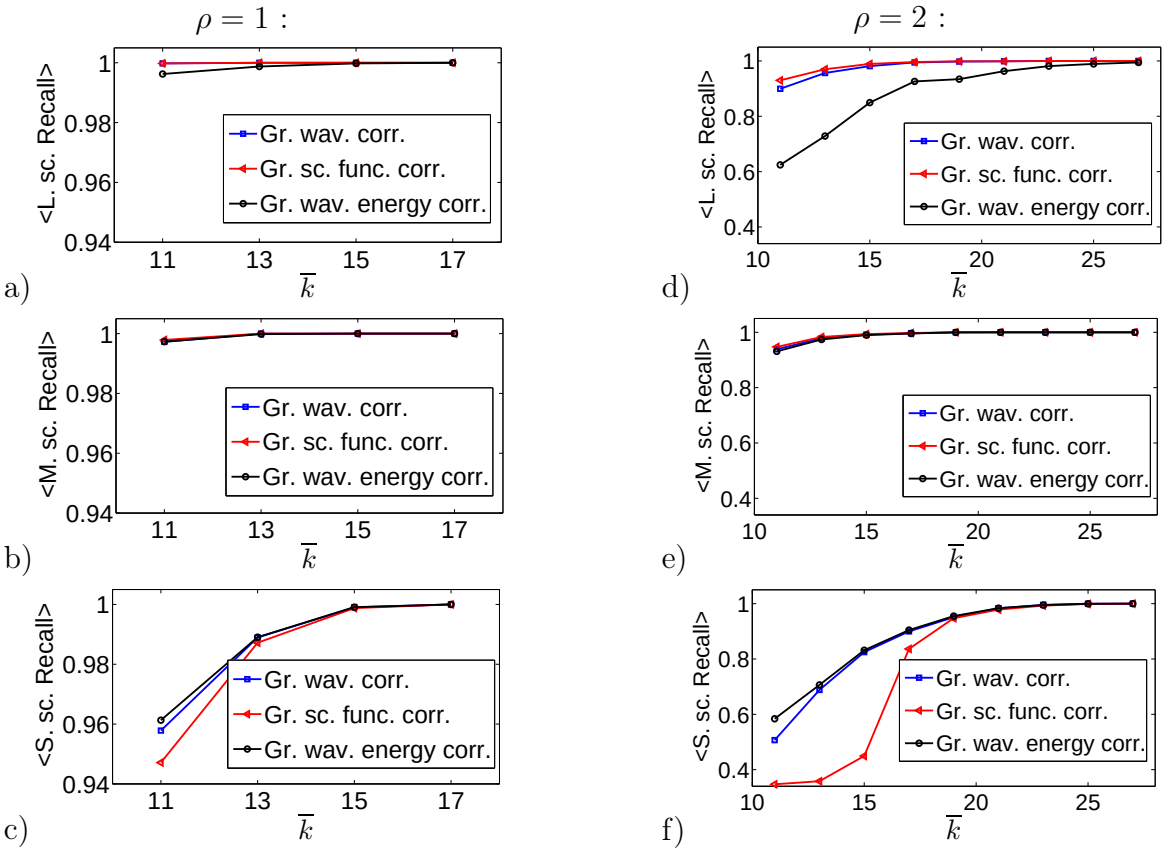


Figure 2.35: Comparaison des performances moyennes sur le modèle de graphe de Sales-Pardo avec différents jeux de paramètres $(\rho, \bar{k})$, selon que l'on choisisse de calculer les corrélations d'ondelettes, de fonctions d'échelles ou d'énergie des ondelettes. Les résultats présentés sont des moyennes sur 100 réalisations de graphes.

détectent mieux les partitions à petite échelle et les trois méthodes détectent aussi bien les partitions à moyenne échelle. Les ondelettes s'avèrent néanmoins plus polyvalentes, avec de bons résultats pour les trois échelles. De plus, le calcul justifiant la version rapide de l'algorithme (en calculant les transformées en ondelettes de vecteurs aléatoires) est mené uniquement dans le cas des ondelettes : les ondelettes restent notre choix de prédilection pour mesurer la matrice de distance entre les nœuds à différentes échelles.

8 Réinterprétation de quelques méthodes multiéchelles

Dans cette partie, nous réinterprétons certaines méthodes multiéchelles de la partie 3.4 en termes d'optimisation de modularité filtrée.

8.1 Quelques notations

Soit une partition P séparant un graphe en J communautés. Cette partition peut être codée à l'aide d'une matrice $\boldsymbol{C}$ de taille $N \times J$:

$$\boldsymbol{C} = (\mathbb{1}_{C_1}|\mathbb{1}_{C_2}|\dots|\mathbb{1}_{C_J}), \tag{2.63}$$

où $\mathbb{1}_{C_j}$ est la fonction indicatrice de la communauté C_j, i.e. :

$$\begin{aligned} \mathbb{1}_{C_j}(i) &= 1 \text{ si } i \in C_j \\ &= 0 \text{ sinon.} \end{aligned} \tag{2.64}$$

Pour calculer la modularité Q d'une partition, on peut passer par la matrice de modularité $\boldsymbol{B}$, définie à partir de la matrice d'adjacence pondérée $\boldsymbol{W}$:

$$\boldsymbol{B}(\boldsymbol{W}) = \frac{1}{2\omega}\left(\boldsymbol{W} - \frac{\boldsymbol{s}\boldsymbol{s}^\top}{2\omega}\right), \tag{2.65}$$

où $\boldsymbol{s}$ est le vecteur des forces (ou des degrés, si $\boldsymbol{W}$ est binaire), et 2ω la somme du vecteur force, c'est-à-dire, $2\omega = \sum_{i=1}^{N} s(i) = \sum_{i,j} W(i,j)$. La modularité d'une partition codée par $\boldsymbol{C}$ dans un graphe de matrice de modularité $\boldsymbol{B}$ s'écrit :

$$Q = \mathrm{Tr}(\boldsymbol{C}^\top \boldsymbol{B}\boldsymbol{C}). \tag{2.66}$$

On peut montrer aisément que cette formule est équivalente à celle de l'équation 2.2.

8.2 Forme canonique de modularité filtrée

Soit une matrice d'adjacence filtrée qui s'écrit sous la forme canonique suivante :

$$\boldsymbol{W}_s = \boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}\boldsymbol{K}_s\boldsymbol{\chi}^\top\boldsymbol{S}^{-\frac{1}{2}}\boldsymbol{W}, \tag{2.67}$$

où $\boldsymbol{K}_s$ est une matrice diagonale dans l'espace de Fourier, de diagonale $K_s(i,i) = k_s(\lambda_i)$ où k_s est un filtre quelconque (à part la contrainte nécessaire à l'existence de certaines équations de la suite : $k_s(1) \neq 0$). Rappelons que la matrice laplacienne normalisée $\boldsymbol{\mathcal{L}}$ s'écrit :

$$\boldsymbol{\mathcal{L}} = \boldsymbol{I} - \boldsymbol{S}^{-\frac{1}{2}}\boldsymbol{W}\boldsymbol{S}^{-\frac{1}{2}}. \tag{2.68}$$

De plus la matrice des vecteurs propres $\boldsymbol{\chi}$ diagonalise $\boldsymbol{\mathcal{L}}$: $\boldsymbol{\mathcal{L}} = \boldsymbol{\chi}\boldsymbol{\Lambda}\boldsymbol{\chi}^\top$ où $\boldsymbol{\Lambda}$ est la matrice diagonale des valeurs propres ($\Lambda_{ii} = \lambda_i$). En inversant l'équation 2.68, on obtient :

$$\boldsymbol{W} = \boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}(\boldsymbol{I} - \boldsymbol{\Lambda})\boldsymbol{\chi}^\top\boldsymbol{S}^{\frac{1}{2}}, \tag{2.69}$$

si bien que $\boldsymbol{W}_s$ peut se réécrire ainsi :

$$\boldsymbol{W}_s = \boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}\boldsymbol{K}_s(\boldsymbol{I}-\boldsymbol{\Lambda})\boldsymbol{\chi}^\top\boldsymbol{S}^{\frac{1}{2}}. \tag{2.70}$$

Le vecteur des forces de la matrice filtrée $\boldsymbol{W}_s$, noté $\boldsymbol{s'}$, s'écrit en vectoriel $\boldsymbol{s'} = \boldsymbol{W}_s\mathbb{1}$ où $\mathbb{1}$ est le vecteur constant égal à 1. Pour calculer $\boldsymbol{s'}$, nous avons besoin de remarquer que :

- $\boldsymbol{S}^{\frac{1}{2}}\mathbb{1}=(\sqrt{s_1}|\sqrt{s_2}|\cdots|\sqrt{s_N})^\top = \sqrt{2\omega}\boldsymbol{\chi}_1$.
- $\Lambda(1,1) = \lambda_1 = 0$.
- $\boldsymbol{\chi}$ est orthonormale.

On a donc :

$$\begin{aligned}
\boldsymbol{s'} &= \boldsymbol{W}_s\mathbb{1} \\
&= \boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}\boldsymbol{K}_s\left(\boldsymbol{I}-\boldsymbol{\Lambda}\right)\boldsymbol{\chi}^\top\boldsymbol{S}^{\frac{1}{2}}\mathbb{1} \\
&= \boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}\boldsymbol{K}_s\left(\boldsymbol{I}-\boldsymbol{\Lambda}\right)\boldsymbol{\chi}^\top\left(\sqrt{2\omega}\boldsymbol{\chi}_1\right) \\
&= \boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}\boldsymbol{K}_s\left(\boldsymbol{I}-\boldsymbol{\Lambda}\right)\left(\sqrt{2\omega}\boldsymbol{\delta}_1\right) \\
&= \boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}\left(k_s(1)\sqrt{2\omega}\boldsymbol{\delta}_1\right) \\
&= k_s(1)\boldsymbol{S}^{\frac{1}{2}}\left(\sqrt{2\omega}\boldsymbol{\chi}_1\right) \\
&= k_s(1)\boldsymbol{s}.
\end{aligned} \tag{2.71}$$

Le vecteur des forces ne change donc pas, à $k_s(1)$ près. On a donc aussi : $\omega' = \frac{1}{2}\sum_{i=1}^N s'(i) = \frac{1}{2}\sum_{i=1}^N k_s(1)s(i) = k_s(1)\omega$. La matrice de modularité filtrée s'écrit alors :

$$\begin{aligned}
\boldsymbol{B}_s = \boldsymbol{B}(\boldsymbol{W}_s) &= \frac{1}{2\omega'}\left(\boldsymbol{W}_s - \frac{\boldsymbol{s'}\boldsymbol{s'}^\top}{2\omega'}\right) \\
&= \frac{1}{2\omega k_s(1)}\left(\boldsymbol{W}_s - \frac{k_s(1)\boldsymbol{s}\boldsymbol{s}^\top}{2\omega}\right) \\
&= \frac{1}{2\omega k_s(1)}\left(\boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}\boldsymbol{K}_s(\boldsymbol{I}-\boldsymbol{\Lambda})\boldsymbol{\chi}^\top\boldsymbol{S}^{\frac{1}{2}} - \frac{k_s(1)\boldsymbol{s}\boldsymbol{s}^\top}{2\omega}\right).
\end{aligned} \tag{2.72}$$

Premièrement, notons que $\forall i \quad \boldsymbol{\chi}_1(i) = \frac{\sqrt{s_i}}{\sqrt{2\omega}}$, si bien que :

$$\frac{\boldsymbol{s}\boldsymbol{s}^\top}{2\omega} = \boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}_1\boldsymbol{\chi}_1^\top\boldsymbol{S}^{\frac{1}{2}}. \tag{2.73}$$

Deuxièmement, notons que, pour une matrice carrée quelconque $\boldsymbol{M}$ de colonnes $\boldsymbol{M}_i$ et une matrice diagonale quelconque $\boldsymbol{D}$, on a :

$$\boldsymbol{M}\boldsymbol{D}\boldsymbol{M}^\top = \sum_{i=1}^N D(i,i)\boldsymbol{M}_i\boldsymbol{M}_i^\top. \tag{2.74}$$

L'équation 2.72 peut donc se réécrire en :

$$\boldsymbol{B}_s = \frac{1}{2\omega}\sum_{i=2}^N \frac{k_s(i)}{k_s(1)}(1-\lambda_i)\boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}_i\boldsymbol{\chi}_i^\top\boldsymbol{S}^{\frac{1}{2}}, \tag{2.75}$$

où la somme commence à l'indice $i = 2$. La modularité filtrée, Q_s s'écrit donc, pour une partition $\boldsymbol{C}$:

$$\begin{aligned} Q_s(\boldsymbol{C}) = \mathrm{Tr}(\boldsymbol{C}^\top \boldsymbol{B}_s \boldsymbol{C}) &= \frac{1}{2\omega} \sum_{i=2}^{N} \frac{k_s(i)}{k_s(1)} (1-\lambda_i) ||\boldsymbol{C}^\top \boldsymbol{S}^{\frac{1}{2}} \boldsymbol{\chi}_i||_2^2 \\ &= \frac{1}{2\omega} \sum_{i=1}^{N} f_s(i) ||\boldsymbol{C}^\top \boldsymbol{S}^{\frac{1}{2}} \boldsymbol{\chi}_i||_2^2, \end{aligned} \tag{2.76}$$

où on introduit f_s le filtre effectif qui va compter dans la maximisation de la modularité filtrée Q_s :

$$f_s(i) = \begin{cases} \dfrac{k_s(i)}{k_s(1)}(1-\lambda_i) & \text{pour} \quad i \geq 2, \\ 0 \quad \text{pour} \quad i = 1 \end{cases} \tag{2.77}$$

Interprétée ainsi, maximiser la modularité filtrée Q_s c'est maximiser la colinéarité entre les fonctions indicatrices $\mathbb{1}_{C_j}$ et les vecteurs propres du laplacien normalisé (à une multiplication par $\boldsymbol{S}^{1/2}$ près) qui ont un poids $f_s(i)$ important.

Nous allons voir dans la suite que certaines méthodes multiéchelles peuvent se réinterpréter sous forme de maximisation de modularité filtrée et nous allons exhiber les filtres f_s équivalents. Dans les cas où $f_s(1) = 0$, et grâce à l'équation 2.77, nous pourrons ensuite remonter aux filtres k_s associés. L'équation 2.77 donne :

$$\forall i \in [2, N] \qquad k_s(i) = k_s(1) \frac{f_s(i)}{1-\lambda_i}, \tag{2.78}$$

et $k_s(1)$ reste *a priori* libre. On remarque que $k_s(1)$ n'est en fait qu'un facteur multiplicatif devant tout le filtre : on peut le fixer à 1 sans perte de généralités. Une fois qu'on aura trouvé des filtres f_s, pour remonter à k_s il suffira d'écrire :

$$k_s(1) = 1 \qquad \text{et} \qquad \forall i \in [2, N] \qquad k_s(i) = \frac{f_s(i)}{1-\lambda_i}. \tag{2.79}$$

8.3 La modularité filtrée de la méthode de Delvenne et al.

Delvenne et al. cherchent à maximiser la trace d'une matrice d'autocovariance $\boldsymbol{R_t}$ qui dépend du temps de Markov t, dont l'équation est (équation 2 de [58]) :

$$\boldsymbol{R_t} = \boldsymbol{H}^\top (\boldsymbol{\Pi} \boldsymbol{M}^t - \boldsymbol{\pi}^\top \boldsymbol{\pi}) \boldsymbol{H}. \tag{2.80}$$

On peut traduire cette expression, avec nos notations, en :

$$\boldsymbol{R_t} = \frac{1}{2\omega} \boldsymbol{C}^\top \left(\boldsymbol{S} (\boldsymbol{S}^{-1} \boldsymbol{W})^t - \frac{\boldsymbol{s}\boldsymbol{s}^\top}{2\omega} \right) \boldsymbol{C}. \tag{2.81}$$

Le terme entre parenthèses peut être interprété comme une modularité filtrée. On définit la matrice de modularité filtrée de Delvenne :

$$\boldsymbol{B}_t^D = \frac{1}{2\omega} \left[\boldsymbol{S} (\boldsymbol{S}^{-1} \boldsymbol{W})^t - \frac{\boldsymbol{s}\boldsymbol{s}^\top}{2\omega} \right]. \tag{2.82}$$

Or, $\boldsymbol{W} = \boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}(\boldsymbol{I}-\boldsymbol{\Lambda})\boldsymbol{\chi}^\top\boldsymbol{S}^{\frac{1}{2}}$, donc :

$$\begin{aligned}(\boldsymbol{S}^{-1}\boldsymbol{W})^t &= (\boldsymbol{S}^{-\frac{1}{2}}\boldsymbol{\chi}(\boldsymbol{I}-\boldsymbol{\Lambda})\boldsymbol{\chi}^\top\boldsymbol{S}^{\frac{1}{2}})^t \\ &= \boldsymbol{S}^{-\frac{1}{2}}\boldsymbol{\chi}(\boldsymbol{I}-\boldsymbol{\Lambda})^t\boldsymbol{\chi}^\top\boldsymbol{S}^{\frac{1}{2}}.\end{aligned} \tag{2.83}$$

$\boldsymbol{B}_t^D$ se réécrit en :

$$\boldsymbol{B}_t^D = \frac{1}{2\omega}\left[\boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}(\boldsymbol{I}-\boldsymbol{\Lambda})^t\boldsymbol{\chi}^\top\boldsymbol{S}^{\frac{1}{2}} - \frac{\boldsymbol{s}\boldsymbol{s}^\top}{2\omega}\right]. \tag{2.84}$$

Au vu des équations 2.73 et 2.74, on peut écrire :

$$\boldsymbol{B}_t^D = \frac{1}{2\omega}\sum_{i=2}^{N} sgn(1-\lambda_i)|1-\lambda_i|^t\boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}_i\boldsymbol{\chi}_i^\top\boldsymbol{S}^{\frac{1}{2}}, \tag{2.85}$$

où nous avons écrit $sgn(1-\lambda_i)|1-\lambda_i|^t$ au lieu de simplement $(1-\lambda_i)^t$ pour les cas où $\lambda_i > 1$ et $t \in]0,1[$. À des différences de notations près, nous retrouvons l'équation 5 de [58] écrite par les auteurs. Notons que les λ_i de l'équation 5 de [58] représentent le spectre de $\boldsymbol{S}^{-\frac{1}{2}}\boldsymbol{W}\boldsymbol{S}^{-\frac{1}{2}}$ et non de $\boldsymbol{\mathcal{L}}$, comme dans notre équation (les deux spectres ne diffèrent que de 1).

Par identification à l'équation 2.76, nous avons :

$$f_s^D(i) = \begin{cases} sgn(1-\lambda_i)|1-\lambda_i|^s & \text{pour} \quad i \geq 2, \\ 0 \quad \text{pour} \quad i = 1 \end{cases} \tag{2.86}$$

sachant que $s \in \mathbb{R}^+$ pour cette méthode. De plus, l'équation 2.79 nous renseigne sur le filtre k_s^D :

$$\forall i \in [1,N] \qquad k_s^D(i) = |1-\lambda_i|^{s-1}. \tag{2.87}$$

8.4 La modularité filtrée de la méthode d'Arenas et al.

Arenas et al. cherchent à maximiser la modularité d'un graphe auquel on rajoute des boucles de poids r sur tous les nœuds. Dans le formalisme ci-dessus, on peut écrire la matrice d'adjacence paramétrisée d'Arenas [22] :

$$\boldsymbol{W}_r^A = \boldsymbol{W} + r\boldsymbol{I}, \tag{2.88}$$

de matrice de modularité filtrée :

$$\boldsymbol{B}_r^A = \boldsymbol{B}(\boldsymbol{W}_r^A). \tag{2.89}$$

On poursuit ici le calcul uniquement dans le cas appelé *corrected Arenas* [2] par Lambiotte (non publié) qui est une variante de la méthode d'Arenas classique :

$$\boldsymbol{W}_r^{CA} = \boldsymbol{W} + \frac{r}{k}\boldsymbol{S} \tag{2.90}$$

2. La méthode appelée *corrected Arenas* est équivalente à la méthode d'Arenas classique dans le cas d'un graphe régulier.

où $k = \frac{1}{N}\sum_i s_i$ est la moyenne du vecteur force. On a :

$$\frac{r}{k}\boldsymbol{S} = \frac{r}{k}\boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}\boldsymbol{\chi}^\top\boldsymbol{S}^{\frac{1}{2}}. \tag{2.91}$$

La matrice $\boldsymbol{W}_r^{CA}$ s'écrit alors :

$$\boldsymbol{W}_r^{CA} = \boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}\left(\boldsymbol{I}(1+\frac{r}{k}) - \boldsymbol{\Lambda}\right)\boldsymbol{\chi}^\top\boldsymbol{S}^{\frac{1}{2}}. \tag{2.92}$$

La matrice de modularité filtrée s'écrit alors :

$$\boldsymbol{B}_r^{CA} = \frac{1}{2\omega'}\left(\boldsymbol{W}_r^{CA} - \frac{\boldsymbol{s'}\boldsymbol{s'}^\top}{2\omega'}\right), \tag{2.93}$$

où $\boldsymbol{s'}$ est le vecteur force (et ω' sa somme) de $\boldsymbol{W}_r^{CA}$. On a :

$$\boldsymbol{s'} = \boldsymbol{s} + \frac{r}{k}\boldsymbol{s} = (1+\frac{r}{k})\boldsymbol{s}, \tag{2.94}$$

si bien que :

$$\boldsymbol{s'}\boldsymbol{s'}^\top = \left(1+\frac{r}{k}\right)^2\boldsymbol{s}\boldsymbol{s}^\top \qquad \text{et} \qquad 2\omega' = \sum_k s'(k) = (1+\frac{r}{k})2\omega. \tag{2.95}$$

Ainsi :

$$\frac{\boldsymbol{s'}\boldsymbol{s'}^\top}{(2\omega')^2} = \frac{\boldsymbol{s}\boldsymbol{s}^\top}{(2\omega)^2}. \tag{2.96}$$

L'équation 2.93 devient :

$$\begin{aligned}\boldsymbol{B}_r^{CA} &= \frac{1}{(1+\frac{r}{k})2\omega}\boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}\left(\boldsymbol{I}(1+\frac{r}{k}) - \boldsymbol{\Lambda}\right)\boldsymbol{\chi}^\top\boldsymbol{S}^{\frac{1}{2}} - \frac{\boldsymbol{s}\boldsymbol{s}^\top}{(2\omega)^2} \\ &= \frac{1}{(1+\frac{r}{k})2\omega}\sum_{i=1}^{N}(1+\frac{r}{k}-\lambda_i)\boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}_i\boldsymbol{\chi}_i^\top\boldsymbol{S}^{\frac{1}{2}} - \frac{\boldsymbol{s}\boldsymbol{s}^\top}{(2\omega)^2}.\end{aligned} \tag{2.97}$$

Or, $\frac{\boldsymbol{s}\boldsymbol{s}^\top}{2\omega} = \boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}_1\boldsymbol{\chi}_1^\top\boldsymbol{S}^{\frac{1}{2}}$, et donc :

$$\boldsymbol{B}_r^{CA} = \frac{1}{2\omega}\sum_{i=2}^{N}\left(1 - \frac{\lambda_i}{1+\frac{r}{k}}\right)\boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}_i\boldsymbol{\chi}_i^\top\boldsymbol{S}^{\frac{1}{2}}. \tag{2.98}$$

Par identification à l'équation 2.76, nous avons :

$$f_s^{CA}(i) = \begin{cases} 1 - \dfrac{\lambda_i}{s} & \text{pour} \quad i \geq 2, \\ 0 \quad \text{pour} \quad i = 1 \end{cases} \tag{2.99}$$

en écrivant $s \sim 1 + \frac{r}{k}$. Or $r_{min} = -\frac{2\omega}{N}$ selon les auteurs, si bien que la variable d'échelle canonique s a une valeur minimale de $s_{min} = 1 + \frac{r_{min}}{k} = 1 - \frac{2\omega}{Nk} = 0$ car $Nk = \sum_i s_i = 2\omega$. On a donc $s \in]0, +\infty]$.
De plus, l'équation 2.79 indique que :

$$\forall i \in [1, N] \qquad k_s^{CA}(i) = \frac{1}{s}\left(\frac{s-\lambda_i}{1-\lambda_i}\right). \tag{2.100}$$

8.5 La modularité filtrée de la méthode de Reichardt et Bornholdt

De la même manière que précédemment, nous pouvons écrire une modularité filtrée pour la méthode de Reichardt et Bornholdt [173] :

$$\boldsymbol{B}_{\gamma}^{RB} = \frac{1}{2\omega}(\boldsymbol{W} - \gamma\frac{\boldsymbol{s}\boldsymbol{s}^{\top}}{2\omega}), \tag{2.101}$$

qui peut se réécrire sous la forme :

$$\boldsymbol{B}_{\gamma}^{RB} = \frac{1}{2\omega}(1-\gamma)\boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}_1\boldsymbol{\chi}_1^{\top}\boldsymbol{S}^{\frac{1}{2}} + \frac{1}{2\omega}\sum_{i=2}^{N}(1-\lambda_i)\boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}_i\boldsymbol{\chi}_i^{\top}\boldsymbol{S}^{\frac{1}{2}}. \tag{2.102}$$

La modularité filtrée de Reichardt et Bornholdt s'écrit donc, pour une partition $\boldsymbol{C}$:

$$Q_{\gamma}^{RB}(\boldsymbol{C}) = \frac{1}{2\omega}(1-\gamma)||\boldsymbol{C}^{\top}\boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}_1||_2^2 + \frac{1}{2\omega}\sum_{i=2}^{N}(1-\lambda_i)||\boldsymbol{C}^{\top}\boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}_i||_2^2. \tag{2.103}$$

Par identification à l'équation 2.76, nous avons :

$$f_s^{RB}(i) = \begin{cases} 1-\lambda_i & \text{pour} \quad i \geq 2, \\ 1-s & \text{pour} \quad i = 1 \end{cases} \tag{2.104}$$

où $s \sim \gamma \in \mathbb{R}^+$ dans ce cas. Etant donné que $f_s^{RB}(1) \neq 0$, on ne peut exactement identifier cette équation à l'équation 2.76, si bien qu'on ne peut extraire un filtre équivalent sur la matrice d'adjacence k_s^{RB}.

Équivalence entre la méthode de Reichardt et Bornholdt et la méthode "corrected Arenas" : les équations 2.99 et 2.104 sont en fait très proches l'une de l'autre. En effet, on a :

$$f_s^{RB} = sf_s^{CA} + 1 - s. \tag{2.105}$$

On a alors :

$$\begin{aligned} Q_s^{RB} &= \sum_{i=1}^{N} f_s^{RB}(i)||\boldsymbol{C}^{\top}\boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}_i||_2^2 \\ &= s\sum_{i=1}^{N} f_s^{CA}(i)||\boldsymbol{C}^{\top}\boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}_i||_2^2 + (1-s)\sum_{i=1}^{N}||\boldsymbol{C}^{\top}\boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}_i||_2^2 \\ &= sQ_s^{CA} + (1-s)2\omega. \end{aligned} \tag{2.106}$$

Ainsi, à une échelle donnée s, la partition (codée par la matrice $\boldsymbol{C}$) qui maximise Q_s^{CA} maximise aussi Q_s^{RB} (car $s > 0$) : les deux méthodes sont équivalentes.

8.6 La modularité filtrée de la méthode de Ronhovde et Nussinov

De la même manière que précédemment, nous pouvons écrire une modularité filtrée pour la méthode de Ronhovde et Nussinov [177] :

$$\boldsymbol{B}_{\gamma}^{RN} = \frac{1}{2\omega}(\boldsymbol{W} - \gamma(\boldsymbol{I} - \boldsymbol{W})), \tag{2.107}$$

qui peut se réécrire sous la forme :

$$\begin{aligned}\boldsymbol{B}_{\gamma}^{RN} &= \frac{1}{2\omega}\boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}\left[\boldsymbol{I}(1+\gamma) - \gamma(\boldsymbol{I}-\boldsymbol{\Lambda})^{-1}\right](\boldsymbol{I}-\boldsymbol{\Lambda})\boldsymbol{\chi}^{\top}\boldsymbol{S}^{\frac{1}{2}} \\ &= \frac{1}{2\omega}\boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}_1\boldsymbol{\chi}_1^{\top}\boldsymbol{S}^{\frac{1}{2}} + \frac{1}{2\omega}\sum_{i=2}^{N}(1+\gamma-\frac{\gamma}{1-\lambda_i})\boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}_i\boldsymbol{\chi}_i^{\top}\boldsymbol{S}^{\frac{1}{2}}.\end{aligned} \tag{2.108}$$

La modularité filtrée de Ronhovde et Nussinov s'écrit donc :

$$Q_{\gamma}^{RN} = \frac{1}{2\omega}||\boldsymbol{C}^{\top}\boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}_1||_2^2 + \frac{1}{2\omega}\sum_{i=2}^{N}(1+\gamma-\frac{\gamma}{1-\lambda_i})||\boldsymbol{C}^{\top}\boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}_i||_2^2. \tag{2.109}$$

Par identification à l'équations 2.76, nous avons :

$$f_s^{RN}(i) = \begin{cases}(1+s-\dfrac{s}{1-\lambda_i}) & \text{pour} \quad i \geq 2, \\ 1 \quad \text{pour} \quad i = 1\end{cases} \tag{2.110}$$

où $s \sim \gamma \in \mathbb{R}^+$ dans ce cas. Etant donné que $f_s^{RN}(1) \neq 0$, on ne peut exactement identifier cette équation à l'équation 2.76, si bien qu'on ne peut extraire un filtre équivalent sur la matrice d'adjacence k_s^{RN}.

8.7 Deux nouvelles propositions de modularité filtrée

On peut imaginer deux autres manières de filtrer la modularité :

1. en utilisant les filtres d'ondelettes :

$$\forall i \in [1, N] \qquad f_s^g(i) = g_s(\lambda_i). \tag{2.111}$$

   Ce qui équivaut à un filtre k_s^g sous la forme :

$$k_s^g(i) = \begin{cases}\dfrac{g_s(\lambda_i)}{1-\lambda_i} & \text{pour} \quad i \geq 2, \\ 1 \quad \text{pour} \quad i = 1\end{cases} \tag{2.112}$$

2. en utilisant les fonctions d'échelles :

$$f_s^h(i) = \begin{cases}h_s(\lambda_i) & \text{pour} \quad i \geq 2, \\ 0 \quad \text{pour} \quad i = 1\end{cases} \tag{2.113}$$

   Ce qui équivaut à un filtre k_s^h sous la forme :

$$k_s^h(i) = \begin{cases}\dfrac{h_s(\lambda_i)}{1-\lambda_i} & \text{pour} \quad i \geq 2, \\ 1 \quad \text{pour} \quad i = 1\end{cases} \tag{2.114}$$

8.8 Les filtres équivalents de la modularité classique

La modularité classique s'écrit :

$$\begin{aligned} \boldsymbol{B}(W) &= \frac{1}{2\omega}\left(\boldsymbol{W} - \frac{\boldsymbol{s}\boldsymbol{s}^\top}{2\omega}\right) \\ &= \frac{1}{2\omega}\sum_{i=2}^{N}(1-\lambda_i)\boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}_i\boldsymbol{\chi}_i^\top\boldsymbol{S}^{\frac{1}{2}}, \end{aligned} \tag{2.115}$$

si bien qu'elle est associée au filtre f suivant :

$$f(i) = \begin{cases} 1-\lambda_i & \text{pour} \quad i \geq 2, \\ 0 & \text{pour} \quad i = 1 \end{cases} \tag{2.116}$$

De nouveau, l'équation 2.79 indique que :

$$\forall i \in [1, N] \qquad k(i) = 1. \tag{2.117}$$

8.9 Comparaison et discussion

Nous récapitulons les filtres de modularité f_s et les filtres de la matrice d'adjacence associés k_s (quand ils existent) sur la Fig. 2.36, où nous traçons chaque filtre à quatre différentes échelles pour bien saisir comment ils se comportent. Une des premières propriétés que l'on peut observer est la divergence de tous les filtres k_s en $\lambda = 1$. L'explication est la suivante. Les filtres k_s tendent à peu près vers $\frac{1}{1-\lambda_i}$ quand le paramètre d'échelle tend vers sa valeur minimale. En effet, dans ce cas, la limite de l'équation 2.67 devient :

$$\begin{aligned} \lim_{s\to 0}\boldsymbol{W}_s &= \boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}(\boldsymbol{I}-\boldsymbol{\Lambda})^{-1}\boldsymbol{\chi}^\top\boldsymbol{S}^{-\frac{1}{2}}\boldsymbol{W} \\ &= \boldsymbol{S}^{\frac{1}{2}}\boldsymbol{\chi}(\boldsymbol{I}-\boldsymbol{\Lambda})^{-1}(\boldsymbol{I}-\boldsymbol{\Lambda})\boldsymbol{\chi}^\top\boldsymbol{S}^{\frac{1}{2}} \\ &= \boldsymbol{S}. \end{aligned} \tag{2.118}$$

Ce qui signifie que le cas à petite échelle extrême tend vers le graphe où chaque nœud est déconnecté des autres : les méthodes de détection de communautés vont naturellement trouver un nœud par communauté à cette échelle : c'est bien ce que l'on cherche ! La divergence des filtres k_s en $\lambda = 1$ est bien nécessaire si l'on souhaite sonder les très petites échelles du graphe. La valeur particulière $\lambda = 1$ pose naturellement des questions, qu'ont de si spécial les vecteurs propres de valeur propre proche de 1 ? Que se passe-t-il pour les graphes qui ont une valeur propre égale à 1, et pour lesquels ces filtres divergent ? Ces questions motiveront de futures recherches.

Une deuxième propriété est la suivante. Plus l'échelle est petite, plus il y a de vecteurs propres $\boldsymbol{\chi}_i$ qui ne sont pas filtrés, c'est-à-dire plus les méthodes prennent en compte l'information de tous les vecteurs propres.

De plus, rappelons que plus un vecteur propre est associé à une valeur propre élevée, plus il a de chances d'être localisé, et donc, *a priori*, moins il est utile pour la détection à grande échelle. Sachant cela, nous sommes en droit de nous demander pourquoi certains filtres f_s à grande échelle continuent à laisser passer les modes à

FILTRES DE MODULARITÉ f_s : FILTRES DE MAT. D'ADJ. k_s :

$$Q_s(C) = \frac{1}{2\omega}\sum_{i=1}^{N} f_s(i)||C^\top S^{\frac{1}{2}}\chi_i||_2^2 \qquad W_s = S^{\frac{1}{2}}\chi \mathrm{diag}(k_s)\chi^\top S^{-\frac{1}{2}}W$$

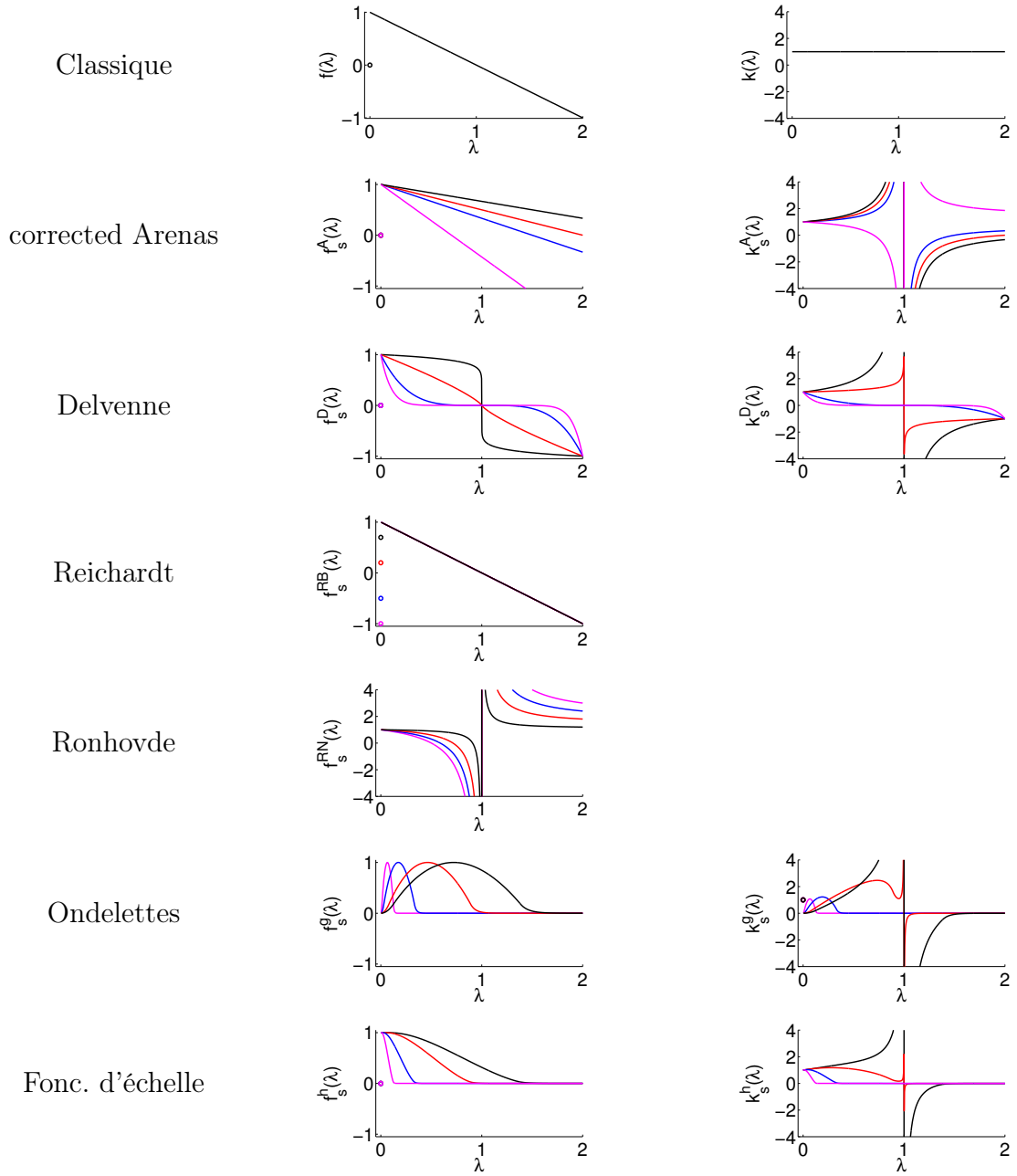


FIGURE 2.36: Comparaison des filtres de modularité f_s (colonne de gauche) et des filtres de matrice d'adjacence associés k_s (colonne de droite, quand ils existent) pour différentes méthodes de détection de communautés multiéchelle. Pour chaque banc de filtres, nous traçons quatre échelles dont le code de couleurs est, de la plus petite à la plus grande échelle : noir, rouge, bleu et magenta.

λ élevé, comme les filtres d'Arenas, de Reichardt et de Ronhovde. À l'inverse, les filtres de Delvenne et ceux construits avec les filtres d'ondelettes ou de fonctions d'échelles, filtrent les λ élevés à grande échelle. Une explication possible est de dire que les valeurs de filtre pour les λ élevés ne sont pas importantes. En effet, optimiser la colinéarité de tous les $\boldsymbol{\chi}_i$ alors qu'ils sont localisés, et orthogonaux, c'est semble-t-il une optimisation de termes contradicatoires vouée à l'échec. Émettons l'hypothèse que, au-delà d'une échelle très petite où la localisation des $\boldsymbol{\chi}_i$ est utile, les termes importants des filtres sont les termes pour lesquels les vecteurs propres $\boldsymbol{\chi}_i$ associés ne sont pas trop localisés, c'est-à-dire ceux entre 0 et 1 (sachant que plus λ_i est proche de 1, plus le vecteur propre associé a de chances d'être localisé). En comparant les filtres dans l'intervalle $[0, 1]$, on observe des filtres de forme générale similaire.

Finalement, réécrire toutes ces méthodes sous forme de modularité filtrée permet une compréhension plus fine des choix des auteurs, et une interprétation de la modularité sous toutes ses formes, d'un point de vue spectral. Pratiquement, cette réécriture a deux intérêts principaux. Tout d'abord, elle permet de relire le problème de conception de modularité filtrée comme un problème d'ingénierie de filtre : comment créer les filtres les plus adaptés ? Un autre intérêt, qui existe à partir du moment où il existe un filtre k_s équivalent au filtre f_s, est qu'optimiser la modularité filtrée par f_s d'un graphe de matrice d'adjacence $\boldsymbol{W}$ est exactement équivalent à optimiser la modularité classique d'un graphe de matrice d'adjacence $\boldsymbol{W}_s$ filtrée par k_s : on peut donc utiliser toute la panoplie d'algorithmes d'optimisation de modularité classique déjà existants sur les matrices $\boldsymbol{W}_s$. Un bémol néanmoins provient du fait que les matrices $\boldsymbol{W}_s$ ne sont pas forcément positives, tout dépend des filtres k_s ; si bien qu'il faut utiliser des algorithmes permettant de gérer ce genre de situations. Ces deux directions (design de filtre et utilisation d'algorithmes classiques d'optimisation pour trouver les communautés de graphes filtrés $\boldsymbol{W}_s$) méritent d'être poursuivies plus avant : ce sera l'objet de travaux futurs.

9 Conclusions et perspectives

9.1 Conclusions

Nous avons, au fur et à mesure de ce chapitre, développé une méthode riche qui tire profit des ondelettes sur graphes et de la définition naturelle d'une échelle qu'elles impliquent, afin de permettre la détection multiéchelle de communautés au sein d'un réseau. La performance de l'algorithme développé est comparable aux performances des meilleurs algorithmes disponibles dans la littérature. Nous avons eu le souci durant tout ce travail, certes de développer une méthode utile à la communauté, mais surtout de montrer que les outils de traitement du signal sur graphes sont des outils adaptés –ou en tous les cas adaptables– aux problématiques rencontrées en science des réseaux. Nous attachons également de l'importance à la dernière partie (partie 8) de ce chapitre, qui montre que le formalisme et concepts du traitement du signal sur graphe permettent d'unifier différentes méthodes existantes de détection multiéchelle de communautés.

9.2 Perspectives

9.2.1 La décomposition en modes empiriques d'un signal sur graphe

Ce projet a débuté dans le cadre d'une visite de deux mois effectuée au laboratoire du Pr. Esaki à l'université de Todai, à Tokyo, au Japon. Nous avions accès aux données récoltées par un réseau de capteurs : capteurs de puissance reliés aux ampoules et aux systèmes de chauffage ou refroidissement de toutes les salles d'un immeuble de l'université. Ce sont autant de séries temporelles qui renseignent sur l'utilisation en temps réel de la lumière et des systèmes de chauffage et refroidissement. Le but du projet est de créer une méthode qui puisse détecter des anomalies dans la consommation d'énergie du bâtiment. Pour cela, nous caractérisons une utilisation "normale" du réseau en calculant une matrice de corrélation "normale" entre tous ces signaux. Puis, en temps réel, dès que les signaux d'un couple de capteurs ne sont pas corrélés alors qu'ils devraient l'être, nous détectons ce couple comme une anomalie. Une situation typique qui a inspiré cette méthodologie est la suivante : dans tel bureau, on observe "normalement" que la lumière et la climatisation sont activées ou désactivées simultanémenent. Et dès que l'utilisateur de ce bureau sort de son bureau en éteignant la lumière mais pas la climatisation, alors une anomalie est détectée.
Une des difficultés rencontrées a été de séparer les séries temporelles en différentes bandes de fréquence afin que les corrélations mesurées pour décrire une utilisation "normale" ne soit pas biaisée par des tendances à très basses fréquences comme les cycles diurnes ou même saisonniers. Tous les détails de ce travail ont fait l'objet de deux publications [81, 79], dont la première [81] (qui est plus détaillée) est rappelée en annexe D (en anglais).

Nous avons également cherché à aborder ce problème de détection en considérant que la matrice de corrélation des signaux définissait un graphe : plus deux signaux sont corrélés, et plus le poids du lien qui relie les deux nœuds associés est élevé. L'idée est alors de caractériser une utilisation normale du réseau, non pas à l'aide de la simple matrice de corrélation, mais via la structure multiéchelle en communautés du graphe sous-jacent. Ensuite, il suffit de calculer cette structure multiéchelle en temps réel et détecter une anomalie dès que l'environnement topologique d'un nœud (caractérisé par les communautés à différentes échelles auxquelles il appartient) change par rapport à son environnement normal. Malgré nos efforts, cette idée n'a pas abouti pour une raison principale : le passage d'une matrice de similarité à une matrice d'adjacence pondérée définissant un graphe n'est pas trivial. En effet, il existe plusieurs méthodes classiques pour faire cela (voir 2.2 de [221]), que ce soit, à partir de matrices de similarité, ou, de manière équivalente, de matrices de distances :

- le graphe de voisinage à ϵ : tous les poids inférieurs à *epsilon* sont supprimés.
- le graphe des k plus proches voisins : pour chaque nœud, on ne garde que les liens le reliant à ses k plus proches voisins. Notons que cela ne rend pas le graphe forcément régulier car la relation n'est pas symétrique (c'est n'est pas parce qu'un nœud donné fait partie des k plus proches voisins d'un autre nœud que le contraire est vrai).

– Appliquer un noyau gaussien à toutes les similarités avec un écart-type donné σ.

Toutes ces méthodes de transformation entre matrice de similarité et matrice d'adjacence d'un graphe ont pour conséquence de rendre le graphe plus parcimonieux, ce qui est plus adapté à la notion de communautés sur graphe. Mais la question reste tout de même ouverte : lorsque les données brutes sont sous forme de matrice de similarité ou de distance, quel est l'intérêt de passer par la définition d'un graphe (et donc de devoir choisir de manière un peu arbitraire une des trois méthodes et surtout le paramètre ϵ, k ou σ associé) pour détecter des communautés, plutôt que de garder la matrice de similarité ou de distance telle quelle et appliquer des méthodes classiques de classification (k-means, clustering hiérarchique,...) ?

Ce séjour au Japon a également inspiré un travail d'adaptation d'une méthode de traitement du signal classique, la Décomposition en Modes Empiriques [106, 142] (EMD en anglais), au cas de signaux définis sur graphe. L'EMD est un algorithme piloté par les données qui cherche à séparer localement les oscillations lentes des oscillations rapides dans un signal. L'EMD étant locale et adaptative, elle est particulièrement utile quand le signal est non-stationnaire et/ou avec un spectre recouvrant, i.e. où une analyse simple par filtrage dans l'espace de Fourier ne fonctionne pas. La difficulté d'adapter l'EMD classique à une EMD sur graphe, est de transposer la notion d'oscillation à un signal défini par un graphe, où chaque nœud a un nombre arbitraire de voisins, et où la notion de maximum ou minimum local, et la notion d'interpolation, entre autres, doivent être revisitées. Ces travaux ont fait l'objet d'une publication [217] que nous reproduisons dans l'annexe E (en anglais), et où sont présentés quelques résultats obtenus pour des signaux simulés sur des exemples de réseaux de capteurs générés artificiellement.

9.2.2 Description multirésolution d'un signal sur graphe

Une des motivations sous-jacentes de créer une méthode de détection de communautés multiéchelle à l'aide d'ondelettes est de pouvoir proposer, à terme, une analyse multirésolution *conjointe* du graphe et du signal sur ce graphe. Pour ce faire, trois opérations sont importantes : la notion de filtrage pour le signal (que nous avons vu dans la partie 2.7.5), de sous-échantillonnage (*downsampling* en anglais) du graphe, et de réduction du graphe.

La première idée est généralement de créer un banc de filtres avec deux (ou plus) canaux qui vont séparer le signal en une somme de deux (ou plus) signaux en fonction de leur bande de fréquences. Ce filtrage se fait généralement dans l'espace de Fourier défini par la diagonalisation du laplacien (et non l'espace de Fourier défini par diagonalistaion de la matrice d'adjacence) : c'est le cas dans les propositions de Shuman et al. [192], de Narang et Ortega [157, 158, 156, 159], et de Ekambaram et al. [70].

Puis, pour réduire l'information qui est alors redondante, l'idée est de sous-échantillonner le graphe en ne conservant que quelques nœuds du graphe, ou en concaténant certains nœuds ensemble. Plusieurs idées ont été étudiées pour cela :

- Narang et Ortega préfèrent définir une transformée multirésolution pour les graphes bipartites. En effet, pour ces graphes, le sous-échantillonnage est trivial : il suffit, comme dans le cas classique, de garder un point sur deux. Puis, si le graphe n'est pas bipartite, alors les auteurs proposent de le décomposer en plusieurs sous-graphes bipartites grâce à un algorithme initialement proposé par Harary et el. [98].
- Ekambaram et al. travaillent également sur des graphes particuliers : non pas sur des graphes bipartites, mais sur des graphes circulants (c'est-à-dire dont il existe une numérotation de nœuds qui rend la matrice d'adjacence circulante). Dans ce cas, il y a assez de symétries dans le graphe pour pouvoir définir un sous-échantillonnage en prenant un nœud sur deux.
- Shuman et al. préfèrent ne garder que les nœuds dont les composantes du vecteur propre du Laplacien associé à la plus grande valeur propre sont positifs. En effet, ce vecteur propre a la particularité de séparer, par sa polarité, le graphe en deux. Dans le cas d'un graphe bipartite, les deux ensembles de nœuds naturels du graphe sont retrouvés par la polarité de ce vecteur propre, et on retrouve donc le cas de Narang et Ortega. Dans le cas d'un arbre, la polarité de ce vecteur propre sépare les niveaux de profondeur paire des niveaux de profondeur impaire, comme proposé dans [190].

Finalement, reste à définir la troisième opération : une fois que le signal est filtré et que nous avons identifié les nœuds que nous allons conserver dans la description à plus grande échelle du signal sur graphe, reste à créer le nouveau graphe, i.e. reste à trouver comment lier les nœuds conservés entre eux. Une méthode régulièrement utilisée est la réduction dite de Kron (appelée également réduction de Schur) [63]. Cette réduction peut s'écrire algébriquement de la manière suivante. Soit α l'ensemble des nœuds que l'on décide de conserver, et α^C son complémentaire. Notons $\boldsymbol{L}_1$ la réduction de la matrice laplacienne : $\boldsymbol{L}_1 = \boldsymbol{L}(\alpha, \alpha)$. Notons également : $\boldsymbol{L}_2 = \boldsymbol{L}(\alpha, \alpha^C)$ et $\boldsymbol{L}_3 = \boldsymbol{L}(\alpha^C, \alpha^C)$. Alors la matrice réduite de Kron du laplacien $\boldsymbol{L}$ (aussi appelée le complément de Schur de $\boldsymbol{L}_1$) s'écrit :

$$\hat{\boldsymbol{L}} = \boldsymbol{L}_1 - \boldsymbol{L}_2 \boldsymbol{L}_3^{-1} \boldsymbol{L}_2^\top. \tag{2.119}$$

Il se trouve que $\hat{\boldsymbol{L}}$ défini un laplacien valide et donc un graphe.

L'idée naturelle qui découle de ce chapitre et qui pourrait enrichir l'ensemble des descriptions multirésolutions sur graphes déjà existantes, se trouve au niveau de la deuxième opération évoquée ci-dessus : le sous-échantillonnage. L'idée est de sous-échantillonner le graphe par rapport à sa structure en communautés. En effet, l'existence de communautés dans beaucoup de graphes donne la possibilité de réduire la taille du graphe naturellement. De plus, pour de nombreux graphes réels, il existe des structures en communautés à différentes échelles. Un signal sur ce graphe pourrait donc être décrit de manière multiéchelle en tirant profit de la structure multiéchelle du graphe sous-jacent. En plus de suivre les structures en communautés, on souhaite que le banc de filtre n'introduise pas de phénomène de repliement de spectre (*aliasing* en anglais), qu'il ne soit pas sur-échantillonné, et qu'il permette une reconstruction parfaite du signal. Les calculs déjà existants dans les travaux

cités ont permis de résoudre certaines difficultés, mais pas toutes. Ce travail est en cours et constitue une des perspectives prometteuses de ce deuxième chapitre de thèse.

Chapitre 3

Rééchantillonnage de groupes de nœuds dans un réseau

« I could prove God statistically »

– G. Gallup

Ce chapitre est consacré à une méthode inspirée du bootstrap classique, mais spécifiquement adaptée à des groupes de nœuds au sein d'un réseau. Pour le lecteur peu familier des techniques de bootstrap, nous rappelons dans la partie 1 quelques définitions et utilisations classiques. La littérature sur le sujet étant conséquente, nous rappellerons uniquement les quelques concepts qui nous seront utiles dans la suite. Les trois parties qui suivent sont essentiellement inspirées de l'article que nous avons publié sur le sujet [214] (dont une version préliminaire a fait l'objet d'un article de conférence [213]), avec quelques ajouts et commentaires en plus. La partie 2 décrit la méthode de bootstrap sur graphes, la partie 3 propose de vérifier la méthode sur un modèle de graphe aléatoire, et la partie 4 est une application de la méthode sur un jeu de données collecté pendant cette thèse : un réseau de contacts humains entre les chercheurs d'une conférence scientifique de cinq jours. Nous finirons par conclure et apporter un regard critique sur la méthode dans la partie 5.

Sommaire

1 Les outils classiques **110**
1.1 Le bootstrap classique 110
1.2 Cas moins classique : bootstrap de séries temporelles corrélées 114
2 Une méthode bootstrap pour des groupes de nœuds dans un réseau **115**
2.1 Problème et état de l'art 115
2.2 Méthode 117
3 Étude contrôlée sur un modèle de réseaux complexes **123**
3.1 Vérification du test statistique 124
3.2 Contrôler la taille de l'espace bootstrap et la puissance du test 124
4 Application à un réseau social **128**
4.1 Présentation du jeu de données 129
4.2 Application de la méthode de bootstrap sur graphes . . . 134
5 Discussion **140**

1 Les outils classiques

Le lecteur intéressé par la littérature sur le bootstrap classique pourra par exemple se référer au livre de Davison et Hinkley [56]. Nous rappellerons dans la suite uniquement les quelques notions utiles à la compréhension du chapitre. Dans la partie 1.1, nous rappelons la définition d'un estimateur et d'un échantillon bootstrap, avant d'illustrer l'utilité du bootstrap dans le cadre d'un test statistique de corrélation. Dans la partie 1.2, nous donnons quelques précisions sur les méthodes de bootstrap par bloc, utiles dans le cas où l'utilisateur souhaite garder quelques corrélations de l'échantillon initial dans les échantillons bootstrap.

1.1 Le bootstrap classique

Le bootstrap est un terme générique qui regroupe des méthodes d'inférence et de test statistiques à l'aide de rééchantillonnage avec remise. C'est Efron [68] qui est à l'origine de ce terme, et du domaine de recherche consacré au bootstrap. La différence du bootstrap avec d'autres méthodes de rééchantillonnage comme le jacknife [170, 218] ou d'autres méthodes (par exemple [99, 145]) est principalement que le bootstrap est une méthode de rééchantillonnage indépendante et identiquement distribuée (i.i.d.) avec remise [67].

Les situations dans lesquelles le bootstrap est utile sont typiquement quand nous avons affaire à un seul échantillon de données. En effet, dans ce cas, il est possible d'estimer la moyenne par exemple, mais pas possible d'estimer à quel point cette moyenne est proche ou non de la moyenne théorique de la distribution sous-jacente des données – c'est-à-dire le biais de l'estimateur de moyenne. Afin de pouvoir estimer certaines caractéristiques de la distribution sous-jacente dont sont tirées les

données, les méthodes bootstrap proposent de rééchantillonner autant de fois que nécessaire l'échantillon initial et de les considérer comme autant de réalisations de la distribution sous-jacente.

Le bootstrap est également utilisé à des fins de test statistique [56] où l'utilisateur peut rééchantillonner autant de fois que nécessaire l'échantillon de départ en s'assurant que les échantillons ainsi obtenus vérifient une hypothèse nulle donnée. Cela permet de tester si oui ou non cette hypothèse nulle est rejetée pour l'échantillon de départ. C'est à des fins de test statistique que nous allons utilisé le bootstrap, et nous mettrons naturellement l'accent dessus dans la suite.

Dans la partie 1.1.1, nous rappellerons quelques définitions et notations utiles sur les estimateurs. Dans la partie 1.1.2 nous rappellerons quelques notations spécifiques aux méthodes bootstraps. Puis, dans la partie 1.1.3, nous montrerons en quoi le bootstrap peut être utile pour des tests statistiques et nous donnerons un exemple de test simple que nous pouvons effectuer avec cette méthode.

1.1.1 Estimateurs

Considérons un échantillon $\mathcal{Y}^0 = \{y_1, y_2, \ldots, y_n\}$ de n observations, toutes tirées de n variables aléatoires $Y_1, Y_2, \ldots, Y_n$ indépendentes et identiquement distribuées de fonction de répartition inconnue F, i.e. :

$$\forall i \in [1, n] \qquad \mathbb{P}(Y_i \leq y) = F(y). \tag{3.1}$$

Notons θ une caractéristique d'intérêt de F, calculée à l'aide d'une fonctionnelle T, appelée estimateur :

$$\theta = T(F). \tag{3.2}$$

θ peut par exemple être la moyenne, la variance ou la médiane de F. Notons F_n la fonction de répartition empirique, directement calculée à partir de l'échantillon :

$$F_n(y) = \frac{\text{nombre d'éléments dans l'échantillon} \leq y}{n} = \frac{1}{n} \sum_{i=1}^{n} H(y - y_i), \tag{3.3}$$

où H est la fonction de Heaviside définie sur $\mathbb{R}$ telle que :

$$H(x) = \begin{cases} 0 \text{ si } x < 0, \\ 1 \text{ si } x \geq 0. \end{cases} \tag{3.4}$$

Pour estimer θ, deux cas s'offrent à nous :

- La distribution F étant inconnue, on peut essayer de l'estimer à l'aide d'un modèle paramétrique en cherchant à ajuster un modèle gaussien, poissonien, etc... (de paramètres ω) sur la fonction de répartition empirique F_n. On obtient ainsi une estimation $\hat{F}_\omega$ de F qui nous permet d'estimer θ :

  $$\hat{\theta} = T(\hat{F}_\omega). \tag{3.5}$$

  Ce genre de méthodes se regroupent sous le terme générique de *méthodes paramétriques*. Elles sont avantageuses dans les cas où les données sont bien connues et où on sait qu'elles sont généralement distribuées selon une certaine

loi. Dans le cadre de cette thèse, néanmoins, nous n'évoquerons pas les méthodes paramétriques, car elles sont pour l'instant difficiles à implémenter dans le cas des graphes. En effet, la modélisation des réseaux complexes a fait couler beaucoup d'encre et il n'est pas forcément aisé de choisir quel modèle convient le mieux. Le lecteur intéressé par ces modèles peut par exemple se référer aux livres de Newman [160], de Watts [224] ou de Durrett [65]. Définir un modèle est une chose, pouvoir générer des graphes aisément à partir de ces modèles ou pouvoir comparer un graphe de terrain à des modèles en est une autre, comme le montre par exemple la thèse de Tabourier [204]. D'autres travaux [149, 150], qui peuvent entrer dans la catégorie des méthodes paramétriques, se sont attelés à la question de la détection de sous-graphes déterministes dans un graphe globalement aléatoire.

– La manière la plus simple d'estimer F à partir de l'échantillon est de l'estimer par la fonction de répartition empirique F_n :

$$\hat{F} = F_n, \tag{3.6}$$

ce qui nous permet, de nouveau, d'écrire l'estimation de θ sous la forme :

$$\hat{\theta} = T(\hat{F}) = T(F_n) = T(\mathcal{Y}). \tag{3.7}$$

Les méthodes d'estimation basées sur cette idée se nomment *méthodes non-paramétriques*. La méthode que nous allons présenter dans ce chapitre se range dans cette catégorie.

1.1.2 L'échantillon et l'estimation bootstrap

Un *échantillon bootstrap* est un échantillon de la même taille que l'échantillon initial $\mathcal{Y}^0$ mais tiré aléatoirement à partir de la fonction de répartition empirique F_n (dans le cas des méthodes paramétriques, les échantillons bootstrap sont tirés à partir de l'estimation paramétrée F_ω de la fonction de répartition). La fonction de répartition empirique associe un poids $1/n$ à chaque élément de l'échantillon, donc tirer un échantillon bootstrap, noté $\mathcal{Y}_b = \{y_1^b, y_2^b, \ldots, y_n^b\}$ à partir de F_n c'est simplement tirer n éléments avec remise dans $\mathcal{Y}^0$. Chaque échantillon bootstrap $\mathcal{Y}_b$ donne une *estimation bootstrap* de θ :

$$\theta_b = T(\mathcal{Y}_b). \tag{3.8}$$

Pour un nombre B d'échantillons bootstrap, on obtient une collection $\{\theta_b\}_{b\in[1,B]}$ d'estimations bootstrap de θ. Cette collection d'estimations bootstrap est très utile en statistique, pour estimer le biais ou la variance d'un estimateur par exemple, ou pour créer des intervalles de confiance et des tests statistiques. Nous allons maintenant nous intéresser à l'utilité du bootstrap pour la conception de tests statistiques, car c'est dans cette optique que nous utiliserons le bootstrap dans la suite du chapitre.

1.1.3 Le bootstrap au service de tests statistiques

Développons ici un exemple simple : l'exemple d'un test de corrélation. Les données se présentent sous la forme d'un échantillon de n paires (u, x) :

$$\mathcal{Y}^0 = \{(u_1, x_1), (u_2, x_2), \ldots, (u_n, x_n)\}, \tag{3.9}$$

et on cherche à savoir si les variables aléatoires U et X sont corrélées. La mesure usuelle pour estimer la corrélation est le coefficient de corrélation r^0 entre les listes $\{u_i\}$ et $\{x_i\}$. La question qui se pose est la suivante : étant donnée la taille de l'échantillon, le coefficient de corrélation mesuré est-il significatif ? Nous pouvons répondre à cette question de la manière suivante :

1. Formuler l'hypothèse nulle H_0 suivante : U et X ne sont pas corrélées.
2. Créer artificiellement B échantillons bootstraps non corrélés. Pour faire cela, il suffit de garder l'ordre d'une des listes, $\{u_i\}$ par exemple, et de permuter aléatoirement l'ordre des $\{x_i\}$. Un échantillon bootstrap s'écrit alors :

$$\mathcal{Y}_b = \{(u_1, x_{p(1)}, (u_2, x_{p(2)}), \ldots, (u_n, x_{p(n)})\}, \tag{3.10}$$

   où p est une permutation aléatoire des indices.
3. On obtient ainsi une collection $\{r_b\}$ de B coefficients de corrélation qui vérifient l'hypothèse nulle.
4. On peut alors estimer la p-valeur de r^0, c'est-à-dire la probabilité qu'on ait observé r^0 si U et X n'étaient pas corrélées :

$$p \simeq \frac{1 + \#\{r_b > r^0\}}{B+1}. \tag{3.11}$$

   Cette p-valeur n'est qu'une estimation dont la précision augmente quand le nombre d'échantillons B augmente. Le but ici est de rejeter H_0 si U et X sont corrélées. Pour que la p-valeur soir correctement estimée, un ensemble de $B = 10/p$ échantillons est souvent suffisant. Typiquement, une p-valeur inférieure à 0.01 suffit à la majorité des utilisateurs, ce qui nécessite donc $B = 1000$ échantillons.

Exemple. Illustrons ce test de corrélation sur un exemple. Considérons un réseau de contacts humains comme celui que nous étudierons dans la partie 4, et qui est détaillé dans la partie 4.1. Dans ce genre de réseaux, on observe régulièrement une corrélation entre le degré d'un nœud (le nombre de nœuds avec qui il a été en contact) et sa force (son temps total de contact), comme illustrée sur la Fig. B.1 de l'annexe B. La question que l'on peut poser est : cette corrélation est-elle significative ? Le problème est que nous n'avons accès qu'à un seul jeu de données et qu'on ne peut pas recommencer l'expérience avec les mêmes conditions initiales. Nous pouvons néanmoins créer des échantillons bootstrap et suivre le protocole de test statistique précédemment décrit pour estimer à quel point la corrélation mesurée est significative. L'échantillon original se met sous la forme :

$$\mathcal{Y}^0 = \{(d_1, s_1), (d_2, s_2), \ldots, (d_n, s_n)\}, \tag{3.12}$$

où d_i est le degré et s_i la force du nœud i. La figure de gauche de Fig. 3.1 trace la force en fonction du degré pour chaque nœud. Le coefficient de corrélation entre les deux vaut $r^0 = 0.67$. Comme décrit précédemment, pour créer un échantillon bootstrap il suffit de garder l'ordre d'une des listes et de permuter aléatoirement l'ordre de l'autre liste. Nous en créons $B = 1000$ et chaque échantillon bootstrap b est à l'origine d'un coefficient de corrélation bootstrap r_b. Nous traçons trois exemples

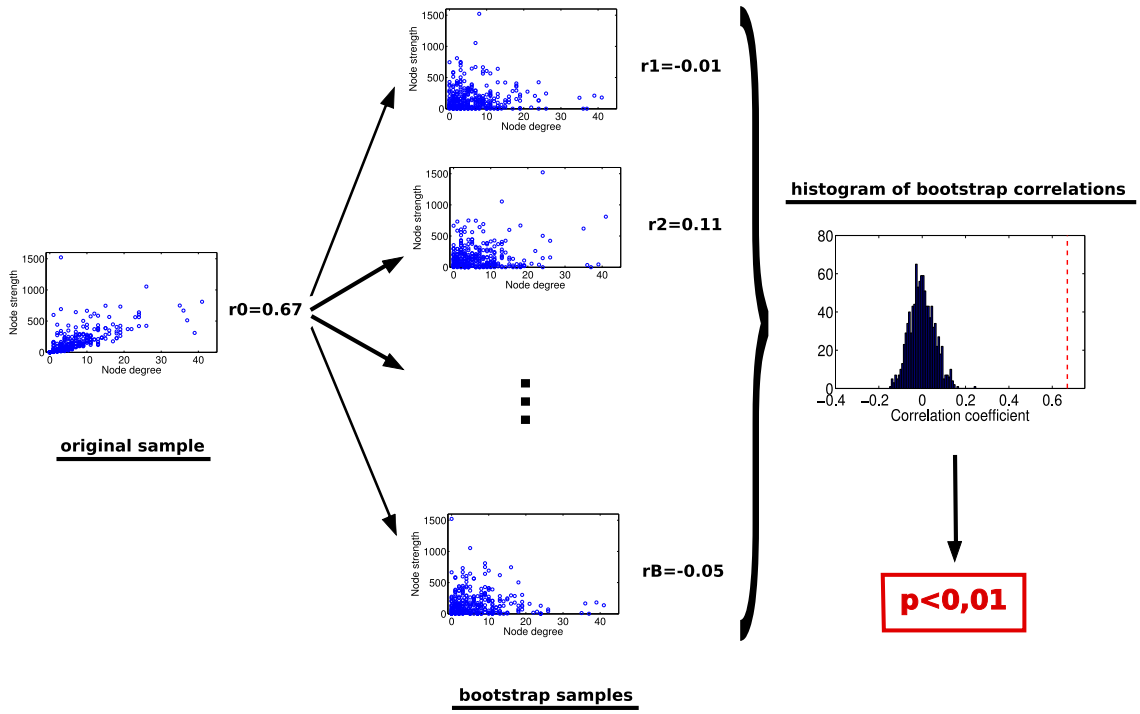


Figure 3.1: Illustration du test statistique de corrélation entre le degré et la force d'un nœud dans le réseau de contact humain présenté dans la partie 4.1. L'échantillon initial est représenté sur la figure de gauche. Trois exemples d'échantillon bootstrap –parmi les 1000 générés– sont montrés au milieu de la figure. Chaque échantillon est tracé à côté de son coefficient de corrélation associé. Ces 1000 coefficients de corrélation sont distribués comme illustré par l'histogramme de droite. Les tirets verticaux rouges représentent le coefficient de corrélation pour l'échantillon de départ : il est très éloigné de la distribution bootstrap, si bien que l'hypothèse nulle est rejetée avec une p-valeur inférieure à 10^{-2}. La corrélation entre le degré et la force des nœuds est belle et bien significative.

d'échantillon bootstrap (avec leur coefficient de corrélation correspondant) sur les 3 figures du milieu. Tous ces coefficients $\{r_b\}_{b\in[1,1000]}$ définissent une distribution représentée par l'histogramme de la figure de droite. La valeur originale r^0 est très éloignée de la distribution. En effet, aucun échantillon bootstrap n'a un coefficient de corrélation supérieur à 0.3 : on peut donc en conclure que la p-valeur de l'observation r^0 est inférieure à 10^{-2}, l'hypothèse nulle est donc rejetée : la corrélation entre le degré et la force des nœuds est belle et bien significative. Afin d'être plus précis sur l'estimation de cette p-valeur, nous avons créé 10 millions d'échantillons bootstrap qui nous permettent de savoir que la p-valeur est en réalité inférieure à $10^{-6}\%$.

1.2 Cas moins classique : le bootstrap de séries temporelles avec corrélation

Dans certains cas, l'échantillon de départ $\mathcal{Y}^0 = \{y_1, y_2, \ldots, y_n\}$ contient des dépendances et on ne peut plus supposer que ses termes sont tous i.i.d. (indépendants et identiquement distribués), c'est le cas par exemple de séries temporelles corré-

lées. Dans ce cas, on cherche à créer des échantillons bootstraps qui conservent ces corrélations afin qu'elles ne biaisent pas l'estimation ou le test statistique que nous souhaitons effectuer. Pour cela, l'idée de base est de créer des échantillons bootstraps en concaténant des blocs (suite contigüe de $\{y_i\}$) de taille l. Comme en bootstrap classique, l'échantillon bootstrap doit avoir la même taille que l'échantillon de départ (il est alors plus simple de choisir l qui divise n), et un même bloc peut-être tiré *a priori* plusieurs fois étant donné que le tirage se fait avec remise. Les échantillons bootstraps ainsi créés conservent les corrélations jusqu'à l.

Selon les propositions, les blocs sont non-recouvrants [95, 41], ou recouvrants [125, 140], ils peuvent aussi être extraits de la série entière préalablement bouclée sur elle-même [167], ou de taille l variable [168]. D'autres travaux [165] proposent de rendre les concaténations de blocs plus lisses. Un article de Lahiri [126] compare ces méthodes de bootstrap par blocs. Dans tous les cas, la discussion de la taille l des blocs est centrale. Si $l = 1$, on retrouve le cas classique d'Efron où les échantillons bootstraps sont i.i.d. Mais dans ce cas, aucune corrélation n'est conservée. Quand l tend vers n, en revanche, les corrélations sont conservées, mais on n'a plus accès qu'à un ensemble très restreint d'échantillons bootstraps : les échantillons bootstraps ressemblent de plus en plus à l'échantillon de départ, ce qui implique un risque de biais important et des tests statistiques peu puissants (avec un haut risque de faux négatif). En effet, dans les cas extrêmes où l est proche de n, les échantillons bootstraps sont très similaires à l'échantillon original et tout test statistique ne fera que comparer l'échantillon original avec lui-même : quelle que soit H_0, elle ne pourra pas être rejetée, même si elle est fausse !

Il existe donc un compromis dans la taille l des blocs. Malheureusement, la taille idéale l^* n'est pas connue dans le cas général et dépend du problème posé [94].

2 Une méthode bootstrap pour des groupes de nœuds dans un réseau

Dans ce chapitre nous allons nous intéresser à une question bien particulière : comment peut-on affirmer ou non qu'un groupe de nœuds donné au sein d'un réseau a un comportement anormal ? Nous verrons dans la partie 4 que nous pouvons adapter cette question pour savoir à quel point un groupe d'individus se mélange à un autre groupe d'individus. Nous pouvons également imaginer des applications en détection de comportement anormal sur réseau. Pour répondre à cette question, il va falloir comparer le groupe avec des échantillons bootstrap qui seront eux-mêmes des groupes de nœuds du réseau. Nous verrons dans la partie 2.1 une présentation du problème plus général d'estimation de caractéristiques d'un réseau et son état de l'art, avant de détailler notre méthode de bootstrap dans la partie 2.2.

2.1 Problème et état de l'art

Le problème de l'estimation de caractéristiques d'un réseau présente plusieurs facettes. Évoquons en premier lieu l'erreur de mesure que l'on fait quand on créé un graphe pondéré. Il y en a (au moins) de deux sortes :

- L'erreur due au sous-échantillonnage. En effet, la grande majorité des graphes étudiés sont en fait des sous-graphes d'un graphe généralement beaucoup plus grand. Les caractéristiques mesurées sur ce genre de graphes ne donnent donc pas exactement les caractéristiques véritablement recherchées du grand graphe sous-jacent [138, 230, 111, 174].
- L'erreur sur la mesure de la structure du graphe. Certains graphes, commes les graphes bayésiens, sont créés en détectant la structure de corrélation au sein d'un jeu de données qui n'est pas, à la base, naturellement sous forme de graphe. Les graphes phylogénétiques [112] ou les graphes de régulation de gènes [72] sont des exemples classiques de graphes bayésiens. Pour estimer ces graphes, il existe de nombreuses méthodes [40], dont certaines [86, 77] qui utilisent des idées de bootstrap mais appliquées au jeu de données, et pas directement au graphe. Étant donné la manière de créer ces graphes, ils sont toujours associés à une certaine erreur de mesure. D'autres types de graphe ont des topologies incertaines, comme les graphes créés à partir d'une matrice de similarité. Il existe différentes manières de créer un graphe à partir d'une telle matrice [221], et, dans tous les cas, l'erreur sur la mesure de similarité entre deux objets se répercute sur la structure de graphe obtenue. Il existe aussi des graphes dont la topologie est certaine (ou presque) mais dont les poids des liens sont soumis à des erreurs de mesure. Un exemple est le graphe du réseau routier où chaque nœud est une intersection, chaque lien une route liant deux intersections, et le poids de chaque lien correspond au trafic mesuré sur la route correspondante. Dans tous les cas, on a accès à un graphe auquel est associé une confiance. Une des questions qui se pose en termes d'estimation est : comment se propage l'erreur sur la mesure du graphe sur la mesure de différentes caractéristiques locales (degré, ...) ou plus globales (coefficient de clustering, centralités, ...) du graphe ? [23, 120].

Une deuxième grande ligne du problème de l'estimation de caractéristiques sur des graphes, et c'est cette ligne qui nous intéresse dans ce chapitre, n'est pas en lien à ces erreurs de mesure mais provient essentiellement du fait qu'un réseau de terrain n'est qu'une réalisation tirée d'une distribution inconnue de graphes ayant certaines caractéristiques ; et qu'il est généralement difficile –voire impossible– de tirer une autre réalisation de la même distribution. Dans le cas des humains, par exemple, on ne peut pas avoir accès à deux réseaux sociaux provenant exactement de la même condition initiale (rien qu'à cause de l'effet mémoire). L'analyste n'a donc généralement accès qu'à une seule réalisation pour estimer des caractéristiques de la distribution sous-jacente inconnue. En d'autres termes, même si on omet l'erreur de mesure, le calcul, par exemple, du coefficient de clustering, n'est qu'une estimation du vrai coefficient de clustering de la distribution sous-jacente du graphe étudié, au même titre que $\hat{\theta}$ n'est qu'une estimation de θ dans la partie 1. Pour pallier cette difficulté, on propose de rééchantillonner le graphe pour créer artificiellement d'autres échantillons qui nous permettront de mieux estimer θ.

Il existe plusieurs façons de rééchantillonner un graphe. Certaines sont paramétrées, i.e. l'idée est de dire que le graphe étudié appartient à un certain modèle (par

exemple : tous les graphes avec même séquence de degrés, même coefficient de clusterings, même première valeur propre de la matrice d'adjacence, etc...), de générer autant de graphes que nécessaire appartenant à ce modèle, et d'estimer ainsi des intervalles de confiance sur les caractéristiques mesurées [30, 233, 97, 228].

D'autres sont non-paramétrées [76, 71] et utilisent le graphe mesuré pour générer des échantillons bootstrap sans faire appel à un modèle de graphe. Certaines de ces méthodes de rééchantillonnage sont utilisées notamment dans le but de mesurer la stabilité d'une partition d'un graphe (voir la partie 3.5.1 et [88, 119, 178, 130]). Toutes ces méthodes existantes permettent de créer des échantillons bootstrap qui sont eux-mêmes des graphes et permettent de répondre à certaines questions avec succès.

Dans ce chapitre, nous présentons une méthode bootstrap qui va créer, non des graphes, mais des groupes de nœuds au sein du graphe. De nouveau, nous avons le choix entre deux directions possibles : soit on utilise un modèle de groupes de nœuds au sein d'un graphe [231, 201, 200] pour développer une méthode bootstrap paramétrée ; soit, et c'est la direction que nous avons préféré prendre, nous cherchons dans le graphe lui-même des groupes de nœuds qui feront office d'échantillon bootstrap.

2.2 Méthode

Notre objectif principal est d'estimer à quel point un groupe de nœuds donné au sein d'un réseau ressemble aux autres groupes. Une méthode standard est de formuler une hypothèse nulle H_0 définissant le comportement normal d'un groupe, de tirer des échantillons bootstrap qui respectent cette hypothèse nulle, et finalement de décréter si ou non l'hypothèse nulle peut être rejetée. Dans cette partie, nous allons définir une méthode de rééchantillonnage spécifique au cas de groupes de nœuds dans un réseau afin de créer des échantillons bootstraps qui permettront de tester de telles hypothèses nulles.

2.2.1 Des observables pertinentes de groupes de nœuds dans un réseau

Soit $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ le graphe représentant le réseau en cours d'étude, avec $\mathcal{V}$ son ensemble de nœuds et $\mathcal{E}$ son ensemble de liens. On note $X^0 \subset \mathcal{V}$ le groupe de nœuds dont on souhaite étudier le comportement. Notons $R^0 \subset \mathcal{V}$ le complémentaire de X^0 dans $\mathcal{V}$ ($R^0 = \mathcal{V} \backslash X^0$).
Nous quantifions le "comportement" de X^0 en nous intéressant à plusieurs observables caractéristiques de la structure du groupe. Dans le contexte de réseaux sociaux, des observables pertinentes vont par exemple mesurer la force des contacts au sein du groupe, possiblement supérieure à la force des contacts avec le reste du réseau. En pratique, et au vu de l'application qui va nous importer dans la suite, nous proposons les sept observables suivantes, en plus du cardinal M de X^0 :

- N^0_{XX} le nombre total de liens de $\mathcal{E}$ entre les nœuds de X^0 ;
- N^0_{RR} le nombre total de liens de $\mathcal{E}$ entre les nœuds de R^0 ;
- N^0_{XR} le nombre total de liens de $\mathcal{E}$ connectant les deux groupes de nœuds ;
- T^0_{XX} le poids total des liens de $\mathcal{E}$ entre les nœuds de X^0 ;
- T^0_{RR} le poids total des liens de $\mathcal{E}$ entre les nœuds de R^0 ;
- T^0_{XR} le poids total des liens de $\mathcal{E}$ connectant les deux groupes de nœuds.

– Q_X^0 la modularité de la partition de $\mathcal{V}$ en deux groupes : X^0 et R^0.

Rappelons que la modularité mesure à quel point une partition sépare le graphe en communautés bien distinctes. Voir l'équation 2.2 du chapitre 2 pour sa formulation précise. Dans le cas présent d'une partition en deux, la modularité ne prend que des valeurs entre -0.5 et 0.5 [1] (une valeur proche de 0.5 signifie deux communautés bien marquées).

On considère ainsi un ensemble de $F = 7$ observables (en plus du cardinal M du groupe) qui ne sont pas complètement indépendantes. On est en droit de questionner cette redondance : on pourrait même avancer que la modularité et le cardinal suffisent à caractériser un groupe. Dans certains cas, ces deux observables suffisent peut-être mais, en général, la modularité n'est pas suffisante pour discriminer certains groupes qui peuvent avoir la même modularité mais pour des raisons très différentes. De plus, X^0 ne forme pas forcément de communauté au sein du réseau : il est donc nécessaire d'ajouter des observables qui sont certes en partie corrélées (voir [139] qui étudie les corrélations entre observables pour différents modèles de graphes) mais qui vont rendre la description d'un groupe plus riche, ce qui nous permettra de mieux discriminer les groupes entre eux.

En fonction de l'application et de la nature du réseau complexe étudié, certaines observables seront plus pertinentes que d'autres pour décrire le comportement des groupes. Nous sommes ici guidés par une application à un réseau de contacts humains face-à-face (les détails sont donnés dans la partie 4.1), mais nous soulignons bien que cette méthode est directement applicable à de nombreux contextes : il suffit simplement d'adapter ces observables au problème étudié.

2.2.2 Rééchantillonnage bootstrap pour des tests statistiques

Une fois les F observables Z sont choisies (nous noterons Z la notation générique d'une observable, là où Z^0 est la valeur prise par l'observable pour le groupe X^0, et où Z_b est la valeur prise pour un échantillon bootstrap X_b), la procédure est la suivante :

1. D'abord, nous formulons une hypothèse nulle qui va spécifier à quel point nous voulons que certaines corrélations soient conservées dans l'ensemble des échantillons bootstraps. Prenons un exemple d'hypothèse nulle pour illustrer :
   H_0 *: X^0 se comporte comme n'importe quel groupe de même taille.*
   Ici, les échantillons bootstrap vont uniquement conserver la taille de X^0 : créer un échantillon bootstrap X_b c'est simplement tirer M nœuds différents dans le réseau.
   Une autre hypothèse nulle plus compliquée est :
   H_0 *: X^0 se comporte comme n'importe quel groupe de même taille et de même modularité à δ près.*
   On garde ici la contrainte sur le cardinal de l'échantillon bootstrap, et on ajoute une contrainte paramétrée par δ sur sa modularité : chaque échantillon bootstrap X_b devra non seulement avoir le même nombre de nœuds mais il devra

1. En effet, la modularité d'une partition en K communautés est bornée par $1 - 1/K$ [220]

aussi vérifier une condition sur sa modularité Q_b : $Q^0(1-\delta) \leq Q_b \leq Q^0(1+\delta)$. La valeur de $\delta > 0$ paramétrise la force de la contrainte de conservation de corrélations (le choix de δ est discuté dans la partie 2.2.6). On note f le nombre d'observables (en plus du cardinal) contraintes par l'hypothèse nulle. Dans le premier exemple, $f = 0$, dans le deuxième exemple, $f = 1$.

2. Ensuite, créons un ensemble de B échantillons bootstrap en rééchantillonnant avec remise des groupes de nœuds qui respectent l'hypothèse nulle. Pour trouver ces groupes sous contraintes, on fait appel à un algorithme de recuit simulé décrit dans la partie 2.2.3.
3. Pour un B suffisamment grand, on peut estimer la distribution de chaque observable des groupes de l'ensemble bootstrap. Ces distributions nous renseignent et définissent, de manière entièrement pilotée par les données, le "comportement normal" des groupes dans le réseau sous l'hypothèse nulle choisie.
4. Choisissons ensuite un seuil de signification α pour tester l'hypothèse nulle, i.e. la probabilité de rejeter l'hypothèse nulle même si elle est vraie doit être inférieure ou égale à α. Autrement dit, α est la limite supérieure du taux de fausse alarme que l'on s'autorise. Dans la littérature, α est également appelée probabilité de fausse détection [25]. Parce que nous avons affaire à un test multiple (plusieurs observables sont partiellement corrélées), nous utilisons la correction de Bonferonni (rappelée dans l'annexe C) : un seuil de signification $\alpha' = \alpha/(F-f)$ est défini et utilisé pour tester séparément les $F-f$ observables non contraintes par l'hypothèse nulle (voir partie 2.2.4).
5. Afin de décider si l'hypothèse nulle peut être rejetée avec un niveau de signification α, et estimer à quel point le groupe d'interêt X^0 est éloigné de l'hypothèse nulle, nous introduisons une divergence d (définie dans la partie 2.2.4) calculée en comparant les distributions empiriques obtenues par bootstrap des observables Z et les valeurs Z^0 effectivement mesurées pour X^0. Quand $d = 0$, l'hypothèse nulle ne peut pas être rejetée ; quand d est supérieure à zéro, elle mesure à quel point X^0 à un comportement qui dévie par rapport aux échantillons bootstraps, c'est-à-dire à quel point X^0 a un comportement anormal.
6. Finalement, deux indicateurs de la taille de l'espace bootstrap sont calculés (définis dans la partie 2.2.5) pour vérifier si les contraintes ne sont pas trop importantes, ce qui aurait des impacts indésirables sur la puissance du test, comme discuté plus en détail dans la partie 2.2.6.

2.2.3 Le recuit simulé pour obtenir des échantillons bootstrap sous contraintes

Un point technique important de la méthode de rééchantillonnage est qu'elle doit nous permettre de tirer des groupes de nœuds qui satisfassent des contraintes préalablement choisies. La contrainte la plus simple est la contrainte sur le cardinal et est trivialement atteinte pour les échantillons bootstrap : il suffit de tirer (sans remise) autant de nœuds que dans X^0.

D'autres hypothèses nulles imposent des contraintes moins triviales. Par exemple, une contrainte possible sur l'échantillon bootstrap est d'avoir le même nombre de

liens intra-groupe (N_{XX}) à δ près, que X^0 (en plus d'avoir la même taille). Vu la difficulté du problème, il faut faire appel à des algorithmes d'optimisation qui vont chercher à minimiser $|N_{X^*X^*} - N_{X^0X^0}|$ sur l'ensemble des groupes à M nœuds du réseau. Nous utilisons un algorithme de recuit simulé [37] comme décrit ci-dessous pour tirer les échantillons bootstrap sous contraintes.

L'algorithme commence avec un groupe aléatoire X de nœuds, de même cardinal que X^0. Le coût C de X est défini comme la valeur absolue de la différence entre la valeur de Z dans le groupe courant et Z^0. Une "température" auxiliaire T est initalisée à une certaine valeur (ici $T_{ini} = 0.5$). À chaque itération de l'algorithme, on garde quelques nœuds du groupe courant X et on change le reste. Plus précisément, on essaie de changer $\min(M \times |r| \times T, M)$ nœuds parmi les M nœuds du groupe, où r est une variable aléatoire gaussienne de moyenne nulle et de variance 1. Si le coût C' du nouveau groupe est inférieur à C, alors on accepte le changement. Si au contraire $C' > C$, alors on accepte le changement avec une probabilité

$$p = \min\left(\exp\left(\frac{C - C'}{T}\right), 1\right). \tag{3.13}$$

Quand le coût cesse de décroître pendant plusieurs itérations, on diminue la température auxiliaire ($T \leftarrow 0.85T$) et on réitère le processus. L'algorithme est arrêté dès que X vérifie la contrainte, i.e. dès que $C = 0$ pour une contrainte forte ($\delta = 0$), ou dès que Z est entre $Z^0(1-\delta)$ et $Z^0(1+\delta)$ pour une contrainte relâchée : on obtient alors l'échantillon bootstrap $X_b = X$ avec $Z_b = Z$.

Ce processus est répété B fois pour obtenir l'ensemble des échantillons bootstrap. Le recuit simulé, replacé dans le contexte général de l'optimisation, est un algorithme d'optimisation stochastique qui propose des solutions approchées.

2.2.4 Normalisation des observables, test de chaque observable et choix de la divergence d

Chaque observable Z est normalisée en une quantité adimensionnelle z, appelée le score Z (*Z-score* en anglais) :

$$z = \frac{Z - \bar{Z}^\dagger}{\sigma_Z^\dagger}, \tag{3.14}$$

où $\bar{Z}^\dagger$ est l'espérance et $\sigma_Z^\dagger$ l'écart-type de l'observable Z dans un graphe aléatoire avec la même séquence de poids que le graphe $\mathcal{G}$. Pour estimer ces deux valeurs, on procède comme suit. Des graphes aléatoires sont générés en réallouant aléatoirement les poids de la liste complète des poids (qui inclut les poids nuls qui correspondent aux liens absents). Ceci rend aléatoire le degré des nœuds, leur force, ainsi que les structures topologiques locales, et ne conserve que la séquence des poids. $\bar{Z}^\dagger$ et $\sigma_Z^\dagger$ sont la moyenne et l'écart-type calculés sur un ensemble de cent tels graphes. Cette normalisation peut paraître arbitraire, mais ce mode de représentation est choisi pour sa clarté (on peut tracer les distributions des $F = 7$ observables normalisées sur la même figure) et, plus important, nous permet de comparer les résultats entre groupes de tailles différentes.

Pour chaque observable normalisée z, la fonction de distribution empirique $\hat{D}_z$ est dérivée de l'ensemble bootstrap. On procède alors à un test statistique sur X^0 avec un seuil de signification $\alpha' = \alpha/(F - f)$. Pour cela, on créé l'intervalle d'acceptation à $1 - \alpha'$: c'est l'intervalle qui comprend une proportion $(1 - \alpha')$ des échantillons bootstrap et qui laisse $\frac{\alpha'}{2}$ des valeurs aberrantes sous l'intervalle et les $\frac{\alpha'}{2}$ restantes au-dessus de l'intervalle. Le test est rejeté pour cette observable spécifique si z^0 n'est pas compris dans cet intervalle.

Finalement, définissons une divergence d qui quantifie à quel point X^0 est différent de l'ensemble bootstrap. Pour chaque observable Z, on définit d_z comme étant la distance minimale entre z^0 et l'intervalle d'acceptation de z précédemment détaillé. Si z^0 est dans l'intervalle, alors $d_z = 0$. La divergence d est alors définie comme la somme de toutes les divergences d_z : elle mesure à quel point X^0 est éloigné de l'ensemble bootstrap. Si $d > 0$, l'hypothèse nulle est rejetée avec un taux de fausse alarme de α et le plus grand est d, le plus éloigné X^0 est de l'hypothèse nulle. En revanche, si $d = 0$, l'hypothèse nulle ne peut pas être rejetée.

2.2.5 Deux indicateurs de la taille de l'espace bootstrap

Dans le bootstrap non-contraint classique, la validité des tests statistiques repose sur un tirage aléatoire non-biaisé des échantillons [234]. Dans le cas présent, en imposant des contraintes sur les échantillons bootstrap afin de conserver une partie des corrélations, une partie de l'aléatoire est perdue et cela introduit quelques dépendances : alors que la divergence d est suffisante pour résumer le résultat d'un test statistique non-contraint, on a ici besoin de garder en plus un œil sur le biais introduit par les contraintes. Dans la suite, nous proposons deux indicateurs pratiques qui permettent cela.

Le premier indicateur est l'écart-type σ_u de la distribution du nombre de fois où chaque nœud est choisi dans un échantillon bootstrap. Il mesure l'uniformité avec laquelle un nœud est choisi dans un échantillon bootstrap : le plus petit est σ_u, le plus uniforme est le choix des nœuds dans l'ensemble bootstrap.

Le deuxième indicateur mesure si les nœuds de X^0 sont choisis plus – ou moins – souvent dans les échantillons bootstrap que si il n'y avait que la contrainte sur le cardinal. Pour cela, nous comparons la distribution empirique du nombre de nœuds de X^0 qui sont dans un échantillon bootstrap à la distribution théorique du nombre de nœuds de X^0 qu'il y aurait s'il n'y avait que la contrainte sur le cardinal. Cette distribution théorique est celle de tirer k nœuds de X^0 en $M = |X^0|$ tirages sans remise dans un ensemble de $V = |\mathcal{V}|$ nœuds. Elle est donc donnée par la loi hypergéométrique suivante :

$$P(k) = \frac{\binom{M}{k}\binom{V-M}{M-k}}{\binom{V}{M}}. \tag{3.15}$$

On calcule ensuite la distance χ^2 entre les deux distributions. Afin de comparer différents χ^2 correspondant à des tests bootstrap différents, chaque distance χ^2 est calculée avec un échantillonnage en 10 classes qui contiennent au moins 5 valeurs chacune. Un point important est que nous n'utilisons pas le χ^2 comme un test d'adéquation qui mesurerait à quel point la distribution théorique ajuste bien la distribution empirique. On s'attend en effet à ce que χ^2 augmente dès qu'on assigne

une contrainte forte aux échantillons bootstrap. On préfère utiliser χ^2 et σ_u comme deux paramètres de contrôle du caractère uniforme de la procédure de bootstrap, et prendre garde qu'ils restent raisonnablement petits.

En résumé, le résultat du test proposé à un seuil de signification α et à un paramètre de relaxation δ donnés, est le triplet (d, χ^2, σ_u). Le plus grand est d, le plus éloigné est X^0 de l'hypothèse nulle ; les plus petits sont χ^2 et σ_u, le moins biaisé est le choix des échantillons bootstrap.

2.2.6 Compromis entre la force de(s) contrainte(s) et la puissance du test

Le paramètre δ contrôle la "force" d'une contrainte donnée : le plus petit est δ, la plus forte est la contrainte. Considérons une contrainte très forte. Dans ce cas, l'espace bootstrap (i.e. l'espace des échantillons bootstrap) peut être drastiquement réduit à un point où les seuls échantillons bootstrap qui vérifient la contrainte sont très similaires au groupe testé X^0 ! Le test sera alors naturellement incapable de rejeter l'hypothèse nulle ($d = 0$) même si X^0 est anormal (i.e. le test a une faible puissance statistique). En d'autres termes, considérons un groupe anormal X^0. On peut toujours trouver une contrainte suffisamment forte (ou un ensemble de contraintes) qui classifie X^0 comme normal : il existe une valeur minimal de δ en-dessous de laquelle le test perd de sa puissance.

Néanmoins, le but de développer une méthode de bootstrap sous contraintes est de tester des groupes avec des hypothèses nulles les plus spécifiques possibles, pour comprendre le plus précisément possible pourquoi et dans quel sens un groupe est anormal. On souhaite donc un δ le plus petit possible pour avoir des échantillons bootstrap les plus représentatifs de l'hypothèse nulle.

Ainsi, pour chaque contrainte, il existe une valeur optimale δ^* de δ qui maximise à la fois la puissance et la précision du test. L'existence d'une valeur seuil δ^* pour δ se transpose en l'existence de maxima autorisés χ^{2*} et σ_u^* pour χ^2 et σ_u.

Mentionnons un point important. La force de la contrainte δ n'est pas le seul mécanisme qui peut réduire la taille de l'espace bootstrap. En effet, le cardinal de l'échantillon M peut réduire l'espace s'il se rapproche trop de la taille totale du graphe, ou si il est trop petit. Par exemple, il n'existe qu'un seul groupe de taille $M = V$ dans un réseau à V nœuds ; et il n'existe que V "groupes" de taille $M = 1$ dans un tel réseau. Ces deux cas extrêmes ne sont pas adaptés à la méthode que nous présentons ici car l'espace bootstrap n'est pas assez grand. Une analogie est possible entre M et la taille des blocs l de la partie 1.2 qui contrôle la portée des corrélations que l'utilisateur souhaite conserver dans les échantillons bootstrap. Malheureusement, même dans le cas plus simple du bootstrap par blocs, il n'existe pas de valeur théorique pour l^*, et la question de la valeur optimale reste ouverte. Nous définissons ici une borne inférieure M_l et une borne supérieure M_u pour que l'espace bootstrap de taille $\binom{V}{M}$ (avec uniquement la contrainte de taille) soit assez grand. Étant donné que l'on tire $B = 1000$ échantillons bootstrap parmi les $\binom{V}{M}$ qui existent, on décide que l'on veut au moins 10000 fois plus d'échantillons possibles que le nombre réellement tiré. Dans le cas où le nombre de nœuds total vaut $V = 320$

(comme dans le cas de la partie 4) :

$$\binom{V}{M} > 10^7 \longrightarrow M_l = 4 \leq M \leq M_u = 316. \tag{3.16}$$

Cette réflexion nous amène à une remarque d'ordre général. Le bootstrap que nous définissons dans cette partie de thèse est en partie une généralisation du bootstrap par blocs. En effet, là où le bootstrap par blocs se cantonne à la conservation de corrélations le long d'une seule direction (typiquement temporelle), le bootstrap de groupes sur graphe propose de conserver des corrélations sur une topologie plus complexe définie par la structure du graphe. Mais ce n'est pas non plus *exactement* une généralisation au vu des différences suivantes. Dans le bootstrap par blocs, le signal est une série temporelle et les échantillons bootstrap sont créés en concaténant avec remise des blocs du signal. Dans le bootstrap sur graphes, le signal est la combinaison du graphe $\mathcal{G}$ et du groupe X^0. Pour créer un échantillon bootstrap, nous ne coupons pas X^0 en plusieurs "sous-groupes" (qui seraient l'équivalent des blocs) pour ensuite les "concaténer". Plutôt, nous cherchons parmi tous les signaux autorisés (tous les groupes de $\mathcal{G}$) des signaux similaires à X^0. C'est bien la puissance combinatoire des groupes d'un graphe (il existe $\binom{V}{M}$ groupes de M nœuds dans un graphe de taille V) qui nous permet cela, que l'on tient à conserver en imposant $M \in [M_l, M_u]$.

Pour résumer cette discussion, le test a trois types de résultats possibles :

1. $d > 0, \forall(\chi^2, \sigma_u, M)$. Dans ce cas où $d > 0$, il n'y a pas besoin de discuter les valeurs de χ^2 et σ_u, ni même de M. En effet, même si $\chi^2 > \chi^{2*}$ et/ou $\sigma_u > \sigma_u^*$, i.e. même si les échantillons bootstraps sont dangereusement similaires à X^0, le comportement de X^0 est quand même différent des échantillons bootstrap : l'hypothèse nulle est rejetée.
2. $d = 0, \chi^2 < \chi^{2*}, \sigma_u < \sigma_u^*, M_l \leq M \leq M_u$. L'espace bootstrap est assez vaste, le test garde sa puissance : l'hypothèse nulle n'est pas rejetée.
3. $d = 0, \chi^2 > \chi^{2*}$ et/ou $\sigma_u > \sigma_u^*$ et/ou $M \notin [M_l, M_u]$. Dans ce cas, nous sommes dans la situation discutée ci-dessus : le test n'est pas assez puissant et aucune conclusion ne peut être tirée. Dans ce cas, il est conseillé de relâcher la (ou les) contrainte(s) en augmentant δ, jusqu'à que l'espace bootstrap soit de nouveau suffisamment grand pour que les indicateurs retombent sous les seuils, et qu'on se retrouve alors dans le cas 2.

Afin de poursuivre la procédure proposée dans cette partie, reste à estimer δ^*, χ^{2*}, σ_u^* et les bornes de M. Une estimation théorique n'étant pas envisageable, nous estimons ces valeurs sur un modèle contrôlé de graphes, pour différents types de contraintes, différentes tailles de groupes, et estimons dans chaque cas les valeurs seuil correspondantes. C'est ce que nous détaillons dans la partie suivante.

3 Étude contrôlée sur un modèle de réseaux complexes

Nous procédons dans cette partie à une validation de la méthodologie proposée, et nous estimons les paramètres δ^*, χ^{2*} et σ_u^* sur des graphes contrôlés. Nous utilisons

des ersatz de graphes complexes : les graphes de Chung-Lu pondérés présentés dans l'annexe B pour lesquels nous contrôlons les degrés, les poids, ainsi que la corrélations entre la force de chaque nœud et son degré. Dans la partie 3.1, ces graphes sont utilisés pour vérifier que le test statistique présenté dans la partie 2 a bien un taux de fausse alarme moyen de α. Ensuite, dans 3.2, nous estimons empiriquement δ^*, χ^{2*} et σ_u^* pour différentes contraintes et différentes tailles de groupes sur ce modèle de graphes.

3.1 Vérification du test statistique

Une approche Monte-Carlo est utilisée pour valider la méthode proposée dans le cas de graphes de Chung-Lu pondérés. Le but est ici de vérifier le taux de fausse alarme du test (le taux auquel les groupes normaux sont rejetés). À cette fin, nous créons 1000 réalisations de graphes de Chung-Lu pondérés générés à partir des distributions empiriques du réseau de contacts humains mesurées lors d'une conférence à Salt Lake City. Ce jeu de données, noté simplement SLC, est décrit en détail dans la partie 4.1 : c'est sur ce jeu que nous allons appliquer la méthode, il est donc logique d'utiliser les distributions empiriques de ces données pour paramétrer le modèle de Chung-Lu pondéré que nous utilisons ici. L'hypothèse nulle considérée ici est : H_0 *: X^0 se comporte de la même façon que les autres groupes de même taille et de même modularité à δ près.* C'est en effet une des hypothèses nulles les plus pertinentes en ce qui concerne le comportement de groupes dans des réseaux de contacts humains. La méthode est appliquée successivement sur 1000 groupes aléatoires (un groupe par graphe de Chung-Lu pondéré) de taille $M = 39$ parmi les 320 nœuds du graphe (c'est une taille de groupe que nous allons étudier dans la partie 4). Chacun des groupes est testé pour différents seuils de signification α (de 0.01 à 0.1) et différentes forces de contrainte δ sur la modularité (de 3% à 100%, avec un cas additionnel $\delta = \infty$ qui revient à ne pas contraindre la modularité). Par construction, un groupe aléatoire dans un graphe de Chung-Lu pondéré devrait être classifié comme normal, et ne devrait pas rejeter le test. C'est bien ce que l'on vérifie ici.

Le taux de fausse alarme moyen obtenu sur ces 1000 groupes, divisé par le seuil de signification voulu α est montré sur la Figure 3.2 en fonction de α et de δ. Afin que le test soit valide, on devrait obtenir uniquement des valeurs inférieures à 1, ce qui est bien le cas. De plus, la valeur mesurée est souvent bien plus petite que 1 (entre 0.3 et 0.6) : c'est le signe que la correction de Bonferonni est pessimiste (i.e. α' pourrait être plus grand que $\frac{\alpha}{F-f}$, le test multiple aurait toujours un taux de fausse alarme contrôlé par α). Finalement, cette partie montre que le taux de fausse alarme est bien contrôlé.

3.2 Contrôler la taille de l'espace bootstrap et la puissance du test

La suite logique de notre investigation serait d'estimer le taux de faux positifs, c'est-à-dire le taux auquel des groupes anormaux ne sont pas rejetés par le test. La difficulté apparaît tout de suite : qu'est-ce qu'un groupe anormal ? Alors qu'il était

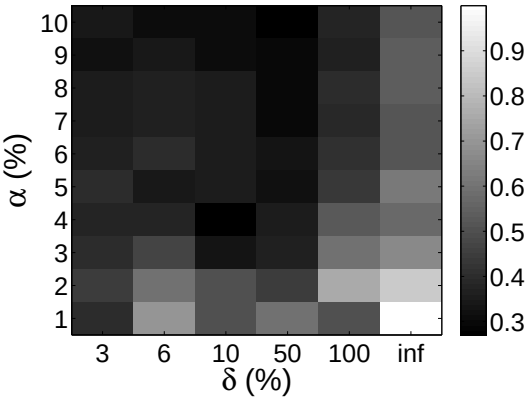


FIGURE 3.2: Ratio du taux de fausse alarme sur le niveau de signification théorique α du test, mesuré sur 1000 groupes aléatoirement tirés dans 1000 réalisations de graphes de Chung-Lu pondérés. Le niveau de signification α agit comme attendu par la correction de Bonferonni : comme une borne supérieure du vrai taux de fausse alarme (i.e. toutes les valeurs de la matrice sont inférieures à 1). Le test est ici effectué avec l'hypothèse nulle de même cardinal et même modularité à δ près. Les résultats sont montrés en fonction de α et de δ. De manière générale, quand α augmente, ou quand δ diminue, la borne de Bonferonni devient de moins en moins ajustée.

aisé de créer des groupes normaux en les choisissant de manière aléatoire parmi les nœuds du réseau, il est plus ambigü de décider ce qu'est un groupe anormal ; et c'est d'ailleurs un des objectifs de ce travail de définir la normalité d'un groupe ! Nous passons donc directement au problème d'estimer les valeurs seuil δ^*, χ^{2*} et σ^*. Autrement dit nous tentons maintenant d'estimer à partir de quelle taille l'espace bootstrap est jugé assez grand pour que le test soit valide.

Dans cette optique, nous prenons un tout autre point de vue et introduisons la notion de rareté : quand une valeur est dans le mode principal d'une distribution, elle est considérée comme assez commune, et quand elle est dans les queues d'une distribution, elle est considérée comme trop rare. À titre d'illustration, nous considérons l'hypothèse nulle "même cardinal et même modularité à δ près". Une fois de plus, considérons 1000 réalisations de graphes Chung-Lu pondérés, et tirons 10000 groupes de cardinal $M = 39$ dans chacun de ces graphes. L'histogramme de la modularité de ces 10^7 groupes (plus exactement : de ces 10^7 partitions du graphe en un groupe et son complémentaire) est montré sur la Figure 3.3. Typiquement, un groupe a une modularité faible, entre -0.03 et 0.03. Un groupe est dit *rare* si sa modularité est dans les extrêmités de cette distribution : soit plus grande que le 10^6-quantile supérieur, soit plus petite que le 10^6-quantile inférieur (ces quantiles sont raisonnablement estimés étant donné que nous avons 10^7 échantillons). Le choix de ces quantiles particuliers est certes arbitraire, mais peut facilement être changé selon l'application et n'influe pas sur la méthode générale. Ces quantiles nous donnent deux bornes de modularité, $Q_l^* = -0.050$ et $Q_u^* = 0.076$, qui séparent les groupes en groupes communs (dont la modularité est dans l'intervalle) et en groupes rares.

Nous proposons que le test ne devrait pas rejeter l'hypothèse nulle pour les

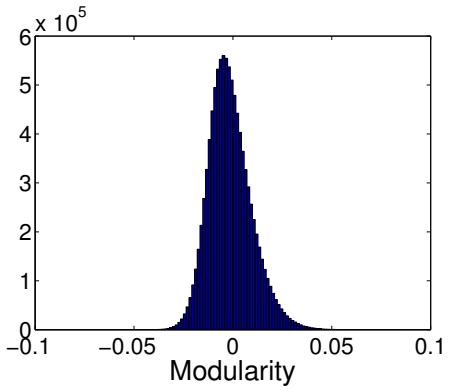


Figure 3.3: Histogramme de la modularité pour un groupe de 39 nœuds dans un graphe de Chung-Lu pondéré de 320 nœuds.

groupes communs, c'est-à-dire que le résultat du test pour un groupe commun quelconque devrait être, au seuil de signification près α, l'option numéro 2 dans la liste de la partie 2.2.6. En effet, les réalisations du modèle de Chung-Lu pondéré sont aléatoires, et des groupes aléatoires dans des graphes aléatoires n'ont aucune raison en général d'être anormaux. Considérons un groupe X^0 dont la modularité est autour de -0.005 (la valeur la plus probable de la distribution de modularité). Pour un δ donné, l'algorithme de recuit simulé va tirer des échantillons de cette distribution, et il y a de grandes chances que les échantillons aient une modularité proche de -0.005 aussi, étant donné que c'est la valeur la plus probable. Ainsi, quel que soit δ, l'espace bootstrap sera assez grand et le résultat du test sera toujours $d = 0, \chi^2 < \chi^{2*}, \sigma_u < \sigma_u^*$. En revanche, au fur et à mesure que nous choisissons des groupes X^0 proches des bornes de modularité Q^*, le test va commencer à rejeter à tort H_0 pour un δ assez grand. En effet, considérons un groupe X^0 avec une modularité proche de Q_u^*. Pour un δ trop grand, la modularité des échantillons bootstrap va continuer à tendre vers -0.005, car ces échantillons ont plus de chances d'être tirés. Si on veut que tous les groupes communs ne rejettent pas le test, il faut avoir un δ assez petit. Ce qui définit une première borne supérieure δ_u pour $\delta : \delta < \delta_u$. Le même argument tient pour les groupes communs dont la modularité est proche de la borne inférieure Q_l^*. On obtient ainsi une autre borne supérieure pour $\delta : \delta < \delta_l$. L'un dans l'autre, on obtient une borne supérieure de δ^* :

$$\delta^* \leq \min(\delta_l, \delta_u). \tag{3.17}$$

Afin de décider quelle valeur de δ^* choisir entre 0 et $\min(\delta_l, \delta_u)$, nous mettons à profit la discussion de la partie 2.2.6 sur le compromis existant entre précision de l'hypothèse nulle et puissance du test. Nous donnons la priorité à la puissance du test et nous choisissons ainsi la valeur maximale autorisée de δ^* :

$$\delta^* = \min(\delta_l, \delta_u). \tag{3.18}$$

En pratique, comme illustré sur le Tableau 3.1-a, δ^* est compris entre 0.05 et 0.15 : les hypothèses nulles conservent une précision raisonnable.

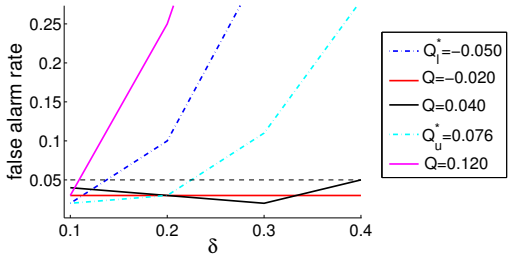


FIGURE 3.4: Taux de fausse alarme en fonction de δ pour des groupes de cardinal 39 de modularités différentes (décrivant des groupes communs et rares) dans des graphes de Chung-Lu pondérés. Les groupes de modularité Q en-dehors de l'intervalle $([Q_l^* = -0.05, Q_u^* = 0.076])$ sont considérés comme rares, et les autres comme assez communs. Pour garder le taux de fausse alarme en-deça du niveau de signification théorique α (ici égal à 5% et représenté par les tirets horizontaux) pour tous les groupes communs, les valeurs maximales $\delta_l^* = 0.15$ et $\delta_u^* = 0.20$ sont lues sur le graphe. Ce qui implique une valeur seuil de $\delta^* = \min(\delta_l^*, \delta_u^*) = 0.15$.

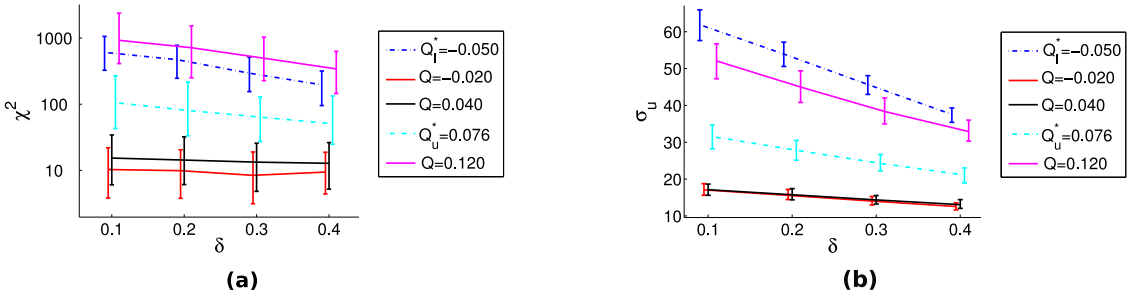


FIGURE 3.5: χ^2 (a) et σ_u (b) en fonction de δ pour des groupes de modularités différentes dans des graphes de Chung-Lu pondérés. Comme le test est créé pour être valide pour tous les groupes communs (i.e. avec modularité $Q \in [Q_l^* = -0.05, Q_u^* = 0.076]$) si on uilise $\delta^* = 0.15$, alors les valeurs seuil σ_u^* et χ^{2*} sont lues de ces courbes. On obtient : $\chi^{2*} = 950$ et $\sigma_u^* = 60$.

Reste donc à estimer δ_l et δ_u. Pour ce faire, la Fig. 3.4 montre le taux de fausse alarme moyen obtenu pour des groupes de 39 nœuds de modularités différentes (des groupes communs et des groupes rares). Comme attendu, pour des groupes très communs (comme ceux de modularité -0.02), le taux de fausse alarme est constant respectivement à δ. En revanche, le plus proche de Q_l^* ou Q_u^* sont les modularités des groupes choisis, le plus rapidement le taux augmente en fonction de δ. Chacune de ces courbes est calculée à partir de 100 groupes sur 100 réalisations de graphes de Chung-Lu (un groupe par graphe). Pour garder ce taux de fausse alarme sous la barre du taux de fausse alarme théorique du test α (ici, $\alpha = 5\%$) pour tous les groupes communs, une valeur maximale δ^* pour δ est lue sur cette Fig. 3.4. Pour le quantile inférieur Q_l^*, on lit $\delta_l = 0.15$, et pour le quantile supérieur Q_u^*, on peut lire $\delta_u = 0.20$. Si bien que la borne de δ pour la contrainte de modularité pour des groupes de taille 39 est :

$$\delta^* = \min(\delta_l, \delta_u) = 0.15. \tag{3.19}$$

Autrement dit, si on choisit $\delta < \delta^*$, tous les groupes communs quelconques ne rejetteront pas le test (avec une tolérance de $\alpha = 5\%$).

La prochaine étape est de trouver les seuils σ_u^* et χ^{2*}. On indique sur la Fig. 3.5 comment varient ces indicateurs en fonction de δ pour des groupes de 39 nœuds de modularités différentes. Étant donné que nous avons simulé chaque cas sur 100 groupes, nous avons tracé l'intervalle de confiance à 95% pour chaque valeur de σ_u et de χ^2. Comme attendu, χ^2 et σ_u augmentent de manière monotone avec la rareté des groupes considérés (pour une valeur donnée de δ). Étant donné que nous avons pris le point de vue de faire en sorte que tous les groupes communs ne rejettent pas le test si $\delta < \delta^* = 0.15$, on lit σ_u^* et χ^{2*} sur les courbes : ce sont les valeurs maximales atteintes en $\delta = 0.15$ par les groupes de modularité Q_l^* ou Q_u^*. On obtient :

$$\chi^{2*} = \max\left(\chi^2(\delta^*; Q = Q_l^*), \chi^2(\delta^*; Q = Q_u^*)\right) = 950 \tag{3.20}$$

et

$$\sigma_u^* = \max\left(\sigma_u(\delta^*; Q = Q_l^*), \sigma_u(\delta^*; Q = Q_u^*)\right) = 60. \tag{3.21}$$

Pour conclure cette partie consacrée à la validation de notre méthode : une fois que sont décidés le taux de fausse alarme théorique α, le cardinal du groupe d'intérêt, et le critère pour décider quand est-ce qu'un groupe est assez commun (i.e. pas trop rare), il est possible de quantifier l'approche par bootstrap sur graphe présentée dans la partie 2 et proposer le triplet de valeurs seuil : δ^*, χ^{2*} et σ_u^*. Dans le Tableau 3.1-a sont récapitulés les triplets $(\delta^*, \chi^{2*}, \sigma_u^*)$ pour toutes les contraintes et toutes les tailles de groupes que nous allons rencontrer dans la partie 4.2. La contrainte notée "contrainte Q_X" (resp. "contrainte N_{XX}", "contrainte T_{XX}") est la contrainte de même modularité (resp. de même nombre de liens entre les nœuds de X, de même somme totale des poids des liens entres les nœuds de X). Dans la partie 4.2, nous allons comparer le résultat du test pour quatre différents groupes avec différentes hypothèses nulles. Afin de pouvoir comparer les résultats proprement, nous utiliserons une valeur unique de δ^* pour chaque hypothèse nulle. Au risque que le test perde un peu de puissance mais pour simplifier l'analyse, nous décidons de prendre, pour chaque hypothèse nulle, le minimum des quatre δ^* trouvés (un pour chaque cardinal). On obtient aussi le σ_u^* et χ^{2*} associés que nous récapitulons dans le dernier Tableau 3.1-b. Ce sont ces valeurs seuil, obtenues grâce au modèle de graphes de Chung-Lu, que nous utiliserons dans l'application proposée dans la partie 4.

4 Application à un réseau social

Nous appliquons cette méthode à un jeu de données collecté dans le cadre de cette thèse à Salt Lake City (SLC) en Novembre 2011 lors de deux conférences scientifiques colocalisées. Le déploiement de la plate-forme de mesure de proximité humaine SocioPatterns [10, 43] nous a permis de mesurer le réseau social dynamique des chercheurs pendant les cinq jours de conférence. Les deux conférences étaient co-organisées par la DPP (Division Physique des Plasmas) de la Société Américaine de Physique (APS) et par Gaseous Electronic Conference (GEC). La DPP regroupe plus

Hypothèse nulle	$M = 39$	$M = 73$	$M = 99$	$M = 106$
contrainte Q_X	$(0.15, 950, 60)$	$(0.15, 110, 45)$	$(0.15, 40, 35)$	$(0.15, 40, 30)$
contrainte N_{XX}	$(0.15, 3500, 85)$	$(0.05, 2700, 115)$	$(0.05, 2600, 125)$	$(0.05, 2200, 125)$
contrainte T_{XX}	$(0.15, 3600, 80)$	$(0.1, 2200, 100)$	$(0.05, 3700, 135)$	$(0.05, 2700, 135)$

a)

Hypothèse nulle	$M = 39$	$M = 73$	$M = 99$	$M = 106$
contrainte Q_X avec $\delta^* = 0.15$	$(950, 60)$	$(110, 45)$	$(40, 35)$	$(40, 30)$
contrainte N_{XX} avec $\delta^* = 0.05$	$(5200, 95)$	$(2700, 115)$	$(2600, 125)$	$(2200, 125)$
contrainte T_{XX} avec $\delta^* = 0.05$	$(4300, 90)$	$(3700, 120)$	$(3700, 135)$	$(2700, 135)$

b)

TABLE 3.1: a) Triplets $(\delta^*, \chi^{2*}, \sigma_u^*)$ pour différentes contraintes et différents tailles de groupes. Le triplet $(\delta^*, \chi^{2*}, \sigma_u^*)$ pour la contrainte sur la modularité Q_X et $M = 39$ est lu sur les Figs. 3.4 et 3.5 comme expliqué dans le texte. Les figures utilisées pour déterminer les autres triplets ne sont pas montrées. b) Pour chaque contrainte, nous décidons de garder un unique δ^* : le minimum des quatre δ^* (obtenus pour chacune des tailles). On montre ici les valeurs seuil (χ^{2*}, σ_u^*) associées.

de chercheurs universitaires là où la GEC regroupe plus de chercheurs industriels ; et les deux institutions avaient l'objectif commun de co-localiser les deux conférences dans le but affiché de mélanger les communautés de chercheurs.

Et c'est la question sous-jacente qui a guidé notre étude : les chercheurs de la DPP et les chercheurs de la GEC forment naturellement deux communautés et les meilleurs efforts possibles ne pourront pas mélanger assez les communautés à tel point qu'elles ne soient plus distingables, mais ces deux communautés se sont-elles significativement mélangées lors de cette conférence en particulier ? Nous nous retrouvons exactement dans le genre de situation où nous n'avons qu'une seule mesure qui peut être vue comme une seule réalisation d'un phénomène en partie aléatoire : nous allons devoir faire appel à des échantillons bootstraps tels que décrit précédemment pour estimer le poids statistique de cette réalisation particulière. Dans un premier temps, nous présentons la méthode expérimentale de mesure du réseau de contacts humains, ainsi que quelques distributions classiques de certaines caractéristiques du réseau dans la partie 4.1. Puis, nous appliquerons la méthode de bootstrap sur graphes dans la partie 4.2 pour répondre à cette question de mélange entre les deux communautés de chercheurs.

4.1 Présentation du jeu de données

Le réseau de contacts humains que nous allons étudier a été mesuré pendant cette thèse à l'aide de badges radio portés par les participants à l'expérience. Dans la partie 4.1.1, nous détaillons le protocole expérimental de ce genre de mesures. Puis, dans la partie 4.1.2, nous détaillons l'organisation prélable à l'expérience, surtout au niveau du dimensionnement : combien d'antennes de mesures sont nécessaires ?

Cela permettra également d'avoir en tête l'agencement de l'espace pendant la conférence, ce qui sera utile à la compréhension de certains résultats. Finalement, dans la partie 4.1.3, nous examinons quelques distributions classiques que l'on peut extraire de ce genre de données.

4.1.1 Méthode expérimentale

Sur les 2081 participants aux deux conférences, 320 ont accepté de participer à l'étude : 281 de DPP et 39 de GEC. La participation était sur la base du volontariat si bien qu'il n'y a pas de biais d'échantillonnage spécifique. La proximité face-à-face des participants est mesurée à l'aide de la plate-forme de mesure SocioPatterns, basée sur des badges RFID (Radio Frequency IDentification) actifs qui peuvent être glissés dans les badges de conférence. Deux badges échangent des paquets radio à faible puissance uniquement si les deux personnes qui les portent sont face-à-face (le corps humain absorbe les paquets à si faible puissance) et à une distance inférieure à la portée du signal qui est de 1 à 1,5 mètre. Quand un badge A reçoit un paquet d'un badge B (le paquet contient essentiellement le numéro d'identification de B), il envoie à plus forte puissance l'information "A a vu B" qui est ensuite enregistrée par les antennes placées dans l'environnement. Le temps auquel l'information est enregistrée est aussi précisément noté à la milli-seconde près. En pratique, autant de précision n'est pas utile, mais c'est surtout pour garder l'ordre des contacts lorsqu'il y a beaucoup d'activité d'un coup. À la fin de la conférence, les données brutes consistent en un log de tous les contacts enregistrés : c'est une liste de lignes (t, r, i, j) où t est le temps auquel l'antenne r a reçu l'information que les individus portant les badges i et j étaient proches et face-à-face (on dit qu'ils sont "en contact"). Étant donnés les paramètres de fonctionnement des badges (puissance, fréquence à laquelle ils envoient leur numéro d'identification), la proximité de deux individus portant des badges RFID peut être détectée avec plus de 99% de chances si elle dure au moins 20 secondes [43], ce qui est une échelle de résolution temporelle assez fine pour étudier la mobilité humaine lors d'évènements sociaux. Nous décidons donc d'agréger les données brutes sur des fenêtres temporelles de 20 secondes. Pour cela, procédons de la manière suivante. Commençons par partitionner les cinq jours de mesure en périodes de 20 secondes. À chacune de ces périodes t, associons une matrice d'adjacence binaire A^t qui représente le graphe agrégé sur 20 secondes : $A^t_{ij} = A^t_{ji} = 1$ si et seulement si i et j ont échangé au moins un paquet pendant la fenêtre temporelle t, et $A^t_{ij} = A^t_{ji} = 0$ sinon.

Finalement, les données définissent un réseau de contact dynamique dans lequel chaque nœud représente un individu, et un lien entre deux nœuds au temps t signifie que les deux individus étaient proches et face-à-face à ce moment-là. Le réseau dynamique peut à son tour être agrégé sur toute la durée de la conférence, définissant un réseau de contacts pondéré où chaque nœud est un individu et le poids de chaque lien représente le temps cumulé de contact sur toute la durée de l'expérience entre les deux individus associés. Ce réseau agrégé sur la durée de la conférence est ensuite légèrement modifié : on enlève les liens qui ont un poids inférieur à 1 minute. Le seuil de 1 minute est choisi car les contacts plus courts peuvent être considérés comme du bruit : deux personnes qui se sont croisées à plusieurs reprises dans le couloir sans vraiment intéragir par exemple. Nous avons vérifié que tous les résultats présentés

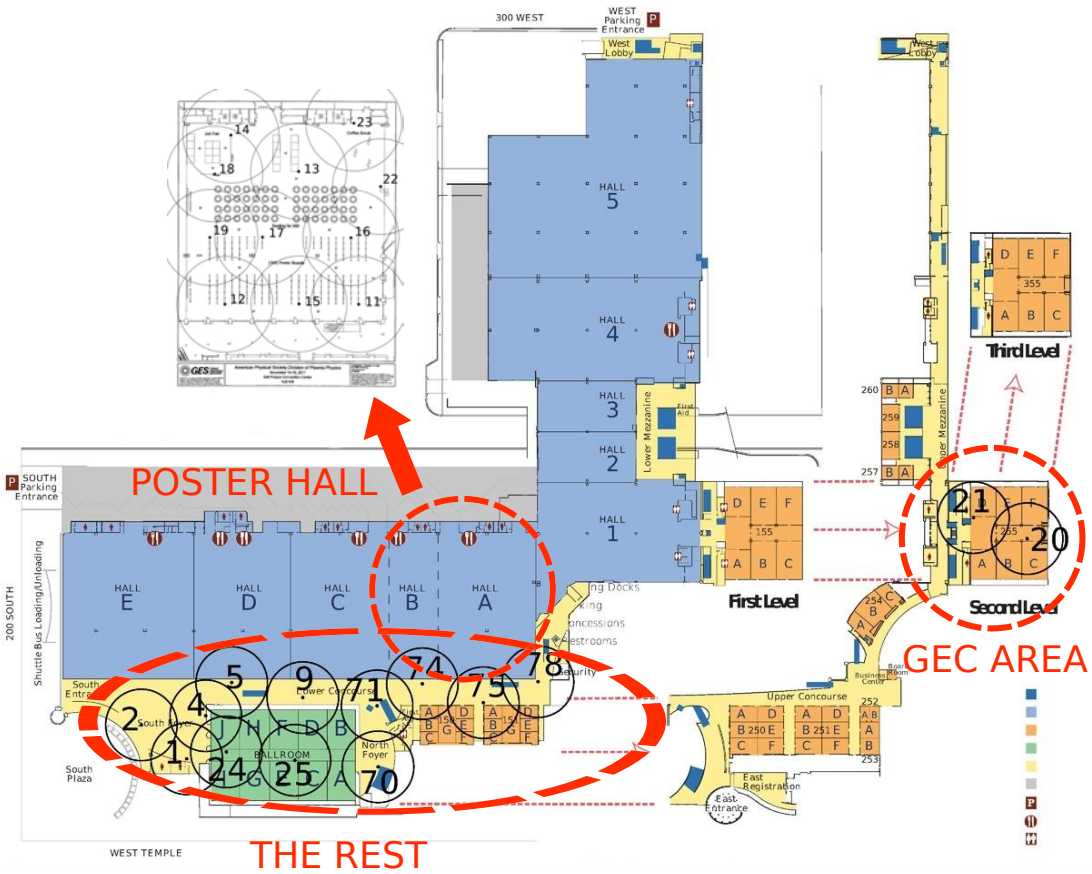


FIGURE 3.6: Plan du centre de conférences de Salt Lake City. Chaque cercle noir correspond à une des 25 antennes radio déployées pour mesurer les interactions sociales entre les participants. Les trois cercles rouges pointillés représentent trois grandes zones d'activité des conférences.

dans ce manuscrit sont robustes par rapport à ce seuil : des résultats similaires sont obtenus pour des seuils à 3 ou à 5 minutes. Dans la suite, certaines distributions seront tirées du graphe dynamique (comme la distribution du temps de contact) et d'autres seront tirées du graphe agrégé (comme la distribution du poids des liens). Enfin, la méthode de bootstrap introduite dans cette thèse s'appliquera sur le graphe agrégé.

4.1.2 Dimensionnement du déploiement

La Fig. 3.6 montre les plans du centre de conférences de Salt Lake City, immense bâtiment dont les deux conférences n'occuppaient qu'une petite partie malgré les plus de deux mille participants. En fait, il y avait trois zones principales, entourées en rouge sur la carte :

1. Le hall des présentations poster était véritablement un espace commun aux deux conférences. C'est là où nous anticipions le maximum de contacts et c'est pourquoi nous avons pris soin de paver l'espace correctement en déployant

	HTT09	SFHH	SLC		
			GEC	DPP	TOUT
Nombre de badges	113	418	39	281	320
Taux d'échantillonnage	75%	33%	12%	16%	15%
Nombre de jours	2	2	5		
Nombre de contacts	9582	27434	1189	21519	23920
Temps total de contacts (heures)	102	414	18	306	339

Table 3.2: Statistiques basiques concernant trois jeux de données mesurées dans trois conférences scientifiques différentes.

onze antennes dans cette grande salle.

2. L'espace GEC était à plus de 500 mètres (!) de là, au deuxième étage : c'est là que GEC organisait ses présentations orales. Des pauses café étaient proposées dans la salle entre les salles de présentation : c'est cet espace que nous avons couvert avec des antennes. La distance de cette zone par rapport au reste de l'activité a été un obstacle évident à l'interaction entre les deux groupes.
3. Le dernier espace, que nous avons appelé "le reste" comprend essentiellement le bureau des inscriptions de DPP, quelques couloirs entre certaines salles de présentation de DPP, et surtout l'espace des pauses café de DPP.

Le dimensionnement du déploiement a été fait au vu de ces plans, au préalable de la conférence. En effet, étant donnée la portée des antennes (ou plutôt : la portée des badges qui émettent les informations de contact aux antennes) et la géométrie particulière, nous avons disposé 25 antennes dans le bâtiment, comme le montre la Fig. 3.6.

4.1.3 Distributions classiques

Dans un premier temps, nous comparons les données obtenues avec d'autres jeux de données obtenues dans des contextes similaires de conférence scientifique avec la même plate-forme de mesure. Le Tableau 3.2 présente quelques statistiques basiques du jeu de données SLC, en comparaison avec les jeux de données mesurées pour la conférence ACM HypertText de 2009 (HT09) [114] et pour la conférence de la *Société Française d'Hygiène Hospitalière* (SFHH) de 2009 [43]. Notons que la somme de tous les contacts (et le temps total de contacts) intra-DPP et intra-GEC ne correspond pas exactement à la somme générale de tous les contacts (TOUT) : le complémentaire est l'interaction entre DPP et GEC. Les données SLC font état d'un nombre relativement faible de contacts, en comparaison avec les autres conférences, au vu du nombre de participants et du nombre de jours des conférences : c'est dû au faible taux d'échantillonnage de la population des conférences de SLC.

Néanmoins, la Fig. 3.7 montre que plusieurs propriétés statistiques des réseaux de contact sont très similaires dans les trois jeux de données. Parmi celles-ci, notons :

– la distribution des temps de contacts. Un contact entre deux badges i et j est une suite continue de 1 dans la liste $\{A_{ij}^t\}$. La durée d'un contact est la longueur de cette liste (multipliée par le pas de temps qui est ici 20 secondes).

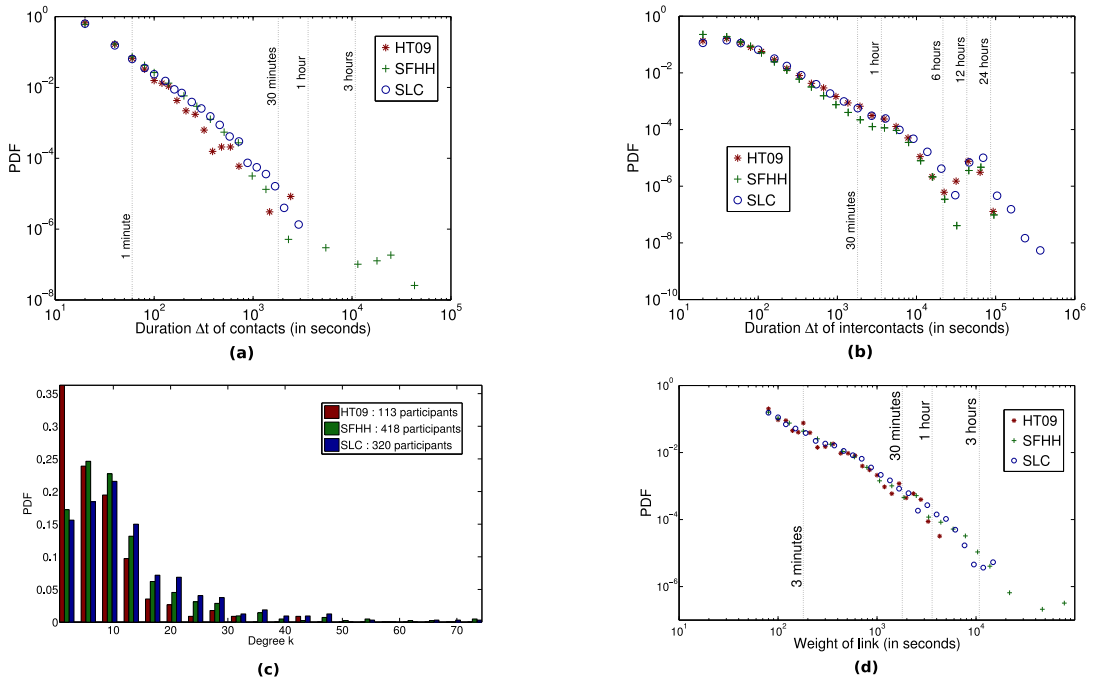


FIGURE 3.7: Comparaison, entre trois jeux de données mesurés lors de conférences scientifiques, de (a) la distribution de temps de contacts (b) la distribution de temps d'intercontacts, (c) la distribution des degrés dans le réseau agrégé, et (d) la distribution des poids des liens.

- la distribution des temps d'intercontacts. Un intercontact est le temps, pour un nœud, entre deux débuts de contacts.
- la distribution des degrés du graphe agrégé sur la durée des conférences. Dans ce cas, le degré d'un nœud qui représente une personne est le nombre d'individus que la personne a contacté au moins une fois.
- la distribution des poids des liens du graphe agrégé sur la durée des conférences. Le poids d'un lien entre deux nœuds donne le temps total de contact entre les participants correspondants.

Ceci confirme la robustesse des propriétés statistiques principales des réseaux de proximité face-à-face entre humains, déjà relevée dans plusieurs travaux [114, 66, 113].

Dans les données SLC, on peut distinguer trois catégories de contacts : les contacts intra-DPP, intra-GEC, et les contacts entre les deux groupes. La Fig. 3.8 montre que même si le nombre de contacts intra-DPP est bien plus grand que intra-GEC (voir le Tableau 3.2), les distributions de temps de contact se superposent remarquablement bien : on n'observe pas de différences de comportement entre ces trois catégories de contacts. Notons également dès à présent que nous ne sommes pas intéressés par une quelconque modélisation de ces distributions (par exemple par des lois de puissance ou log-normales), car la méthode que nous proposons est entièrement pilotée par les données. Il est néanmoins intéressant de remarquer que les formes élargies et hétérogènes des distributions impliquent qu'une méthode paramétrique serait difficile à implémenter [51, 91], et on s'attend à ce qu'une méthode

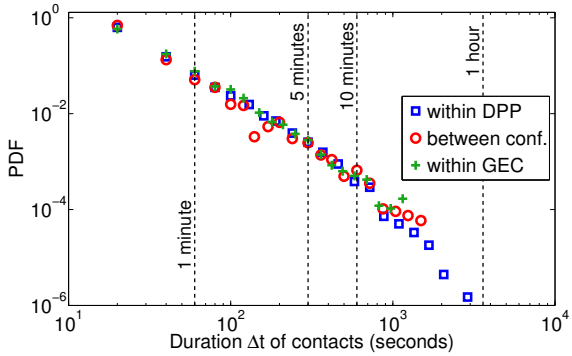


Figure 3.8: Distributions cumulées de la durée de contacts intra-DPP, intra-GEC, et entre les deux conférences du jeu de données SLC.

pilotée par les données soit plus appropriée.

4.1.4 Distributions de contacts en fonction du lieu

L'activité mesurée est spatialement hétérogène, avec en particulier les trois grandes zones évoquées sur le plan de la Fig. 3.6 : la zone GEC, le hall des posters, et le reste. Comme la méthode de mesure nous permet d'avoir une résolution spatiale du graphe de contacts dans la mesure où chaque contact est enregistré par une antenne bien précise et localisée, il est intéressant de se demander si les statistiques de contacts sont différentes selon le lieu où ils sont mesurés.

À cette fin, les histogrammes des temps de contacts séparés par catégorie de contacts et par lieu de contacts sont tracés sur la Fig. 3.9. Pour les contacts intra-DPP (figure de gauche), les distributions mesurées dans les différents lieux ont des formes similaires, et les différences proviennent essentiellement du nombre de contacts mesurés dans chaque lieu (les membres de DPP n'ont pas beaucoup visité la zone GEC). En revanche, les contacts entre les deux groupes (figure du milieu) et pour les contacts intra-GEC (figure de droite), différentes pentes sont observées selon le lieu des contacts. Des distributions plus hétérogènes sont obtenues dans le hall des posters, en particulier pour les contacts entre les deux groupes : le hall des posters est le lieu le plus favorable aux longs contacts inter-groupes. Ce qui nous amène à une remarque assez évidente : organiser des évènements dans des lieux communs favorise le mélange entre individus de groupes différents !

4.2 Application de la méthode de bootstrap sur graphes

Comme indiqué précédemment, la question posée par les organisateurs et motivant cette étude est la suivante : est-ce que le but affiché des organisateurs de mélanger les deux communautés de chercheurs a été atteint ? Afin de donner une réponse quantitative à cette question, nous avons besoin de comparer les interactions observées entre GEC et DPP à une certaine référence. La question de cette référence est centrale. En effet, nous ne pouvons pas "réitérer" l'expérience pour voir à quel point elle est reproductible, nous devons estimer une signification statistique

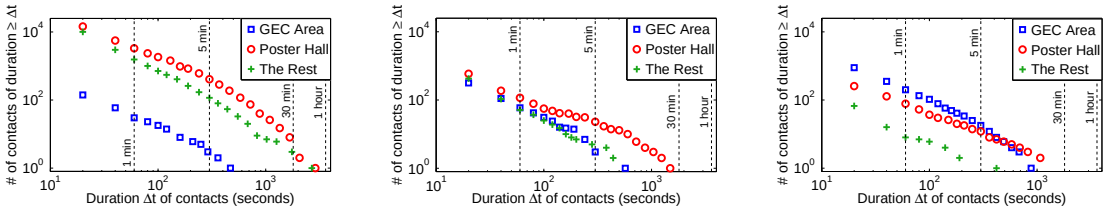


FIGURE 3.9: Histogrammes cumulés des temps de contacts en fonction du lieu à l'intérieur du centre de conférence. Résultats pour : (à gauche) les contacts intra-DPP, (au milieu) les contacts entre les deux groupes, (à droite) les contacts intra-GEC.

uniquement à partir des données que l'on a. La méthode proposée dans ce chapitre est un candidat naturel pour répondre à cette question.

4.2.1 Choisir les groupes et les hypothèses nulles

Les $F = 7$ observables retenues pour caractériser le "comportement" d'un groupe de nœuds incluent des caractéristiques intra-groupes comme N_{XX} et T_{XX}, des caractéristiques mesurées entre le nœuds qui n'appartiennent pas au groupe N_{RR} et T_{RR}, et des caractéristiques qui mesurent à quel point les groupes intéragissent N_{XR}, T_{XR} et Q_X. La terminologie "comportement du groupe" est utilisée pour simplifier, mais il est clair que les caractéristiques choisies quantifient aussi le comportement du complémentaire du groupe ainsi que les interactions entre les deux. Quantifier par exemple le "comportement de GEC" est en fait une manière de mesurer le mélange entre GEC et DPP (DPP est le complémentaire de GEC). La méthode exposée dans la partie 2 est un moyen de quantifier et de valider statistiquement, la normalité – ou l'anormalité – du comportement de GEC par rapport à différents hypothèses nulles. La méthode est ainsi appliquée au groupe des chercheurs de GEC, qui prend le rôle du groupe étudié X^0. Tous les tests statistiques reportés dans la suite sont fait à un seuil de signification $\alpha = 5\%$.

Il est difficile de décider la normalité d'un groupe en n'utilisant qu'une seule hypothèse nulle qui restreint forcément la définition même de normalité. Il existe peu de modèles qui pourraient décrire le comportement attendu d'un groupe dans ce genre de données (e.g., [201, 231, 200]) et aucun n'a été pensé pour décrire l'intégralité des caractéristiques que nous prenons en compte ici. Notre approche permet de tester le groupe GEC par rapport à un ensemble d'hypothèses nulles qui, ensemble, proposent une description plus riche qu'une seule hypothèse qui aurait pu paraître arbitraire. L'objectif de l'étude n'est pas seulement de détecter si GEC se comporte différemment des autres groupes, mais de quelle(s) manière(s) il est différent ou similaire aux autres groupes.

Les différentes hypothèses nulles que nous allons étudier sont au nombre de quatre et peuvent toutes s'écrire sous la forme :

H_0 : X^0 se comporte comme tout autre groupe du réseau...

1. *...de même taille.* Cette hypothèse nulle permet d'éliminer toute différence de comportement due à la taille. Dans ce cas, aucune des F observables n'est

Groupe	Cardinal	N_{XX}	N_{XR}	N_{RR}	T_{XX}	T_{XR}	T_{RR}	Q_X
GEC	39	101	120	1907	58820	45740	947100	0.100
STP	106	384	850	894	252900	356220	442540	0.145
JUP	73	183	766	1179	97600	303800	650260	0.073
SEP	99	226	704	1198	124280	310740	616640	0.095

Table 3.3: Le cardinal et les sept caractéristiques retenues pour les quatre groupes de nœuds étudiés. T_{XX}, T_{XR} et T_{RR} sont en secondes.

contrainte : $f = 0$.

2. *... de même taille et de même modularité.* Dans ce cas, la modularité reste égale à celle de la partition $(X^0, \mathcal{V}\backslash X^0)$ à δ près. Ceci permet de limiter la détection de différence de comportement due simplement au caractère communautaire d'un groupe. Dans ce cas, $f = 1$.
3. *... de même taille et de même nombre de liens intra-groupe N_{XX}.* Ceci permet d'éliminer toute différence de comportement due à la densité d'intra-connections d'un groupe. Dans ce cas, $f = 1$.
4. *... de même taille et de même temps total d'interactions intra-groupe T_{XX}.* Ceci permet d'éliminer toute différence de comportement due à la durée totale de contacts au sein d'un groupe. Dans ce cas aussi, $f = 1$.

Notons que la deuxième hypothèse nulle (celle qui contraint la modularité) permet de limiter la détection de différence de comportement due simplement au caractère communautaire d'un groupe, mais pas de l'éliminer complètement. En effet, il existe de nombreuses interprétations possibles de la notion de communauté (voir le chapitre 2) et la modularité n'en mesure qu'une partie. Pour être surs que les différences de comportement potentielles ne sont pas uniquement dues au caractère communautaire d'un groupe, nous allons faire le test sur trois autres groupes du réseau qui ont *a priori* un comportement communautaire : les étudiants de DPP, i.e. les participants qui n'ont pas encore soutenu leur thèse (STP), les juniors de DPP, i.e. les chercheurs qui ont soutenu leur thèse il y a moins de dix ans (JUP), et les seniors de DPP, i.e. les chercheurs qui ont soutenu leur thèse il y a plus de dix ans (SEP). Notons qu'à eux quatre, ces groupes forment une partition du réseau total. Le Tableau 3.3 liste les caractéristiques mesurées pour GEC, STP, JUP et SEP. On s'attend à ce que chacun de ces groupes forme une communauté dans le réseau de contact au vu de similarités d'âge ou de statut profesionnel. Les modularités reportées sur le Tableau sont en effet élevées. Il est donc intéressant de comparer les résultats du test pour GEC avec ceux pour ces trois autres groupes : si leur comportement est similaire, on pourrait argumenter que GEC se comporte comme un sous-groupe de DPP, et la conclusion serait que la colocalisation des deux conférences était un moyen efficace de mélanger les deux populations de chercheurs. Si au contraire, GEC est significativement plus anormal que les trois autres groupes, ce serait un argument fort pour douter de l'efficacité de la co-organisation de l'évènement.

Dans la suite nous allons tester ces quatre groupes (i.e., le groupe noté X^0 dans la partie 2 sera alternativement GEC, STP, JUP, ou SEP) vis-à-vis des mêmes

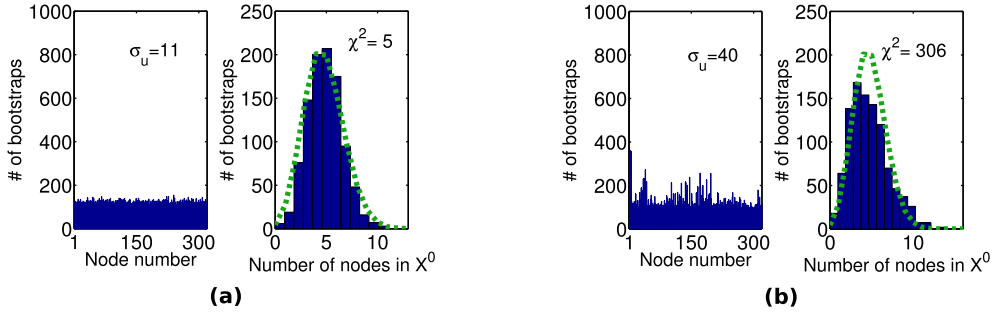


FIGURE 3.10: σ_u et χ^2 pour $X^0 =$ GEC pour (a) le test avec contrainte de cardinal (détaillé dans la partie 4.2.2, (b) le test avec contrainte de cardinal et de modularité (avec $\delta = 15\%$). L'histogramme de gauche représente le nombre de fois que chaque nœud a été sélectionné dans un échantillon bootstrap : l'uniformité de cette distribution est indiquée par son écart-type σ_u. L'histogramme de droite est le nombre de nœuds de X^0 qui ont été choisis dans chaque échantillon bootstrap : sa distance χ^2 avec la distribution hypergéométrique en tirets verts mesure à quel point l'échantillonnage est éloigné du cas où il n'y a que la contrainte de cardinal.

hypothèses nulles et nous comparerons à quel point les tests sont rejetés pour chacun de ces groupes.

4.2.2 Contrainte de cardinal

Dans un premier temps, nous considérons l'hypothèse nulle de même cardinal. Plus précisément, nous testons si GEC se comporte comme n'importe quel autre groupe aléatoire de $M = 39$ individus dans la conférence. Formellement, l'hypothèse nulle est : H_0 *: GEC se comporte comme tout autre groupe de même taille.*

En appliquant la méthode de la partie 2, on tire avec remise $B = 1000$ échantillons bootstrap de 39 nœuds. Pour chacun de ces échantillons, nous mesurons les sept observables associées et les normalisons comme proposé dans la partie 2.2.4. Pour chacune des observables normalisées z, nous obtenons ainsi la distribution empirique $\hat{D}_z$. Les intervalles d'acceptation à $1 - \alpha' = 1 - \frac{\alpha}{F-f} = 1 - \frac{0.5}{7} = 99,3\%$ de ces distributions empiriques définissent ce qu'on appelle le "comportement normal" d'un groupe de 39 nœuds dans le graphe. Puis, on calcule la divergence d_z associée à chaque z, pour finalement obtenir la divergence totale d.

La Fig. 3.10.a montre deux histogrammes qui illustrent ce que cherchent à mesurer les deux indicateurs σ_u et χ^2. L'histogramme de gauche montre le nombre de fois que chaque nœud a été sélectionné dans un échantillon bootstrap : son écart-type σ_u quantifie à quel point cette sélection est uniforme ou non. L'histogramme de droite montre le nombre de fois qu'un nœud de GEC a été sélectionné dans les échantillons bootstraps : le χ^2 mesure une distance entre cet histogramme empirique et l'histogramme théorique hypergéométrique tracé en pointillés verts. Dans le cas présent où les nœuds sont choisis aléatoirement sans contraintes (à part le cardinal), les deux valeurs σ_u et χ^2 approchent de leur valeur minimale.

La figure en haut à gauche de la Fig. 3.11 résume le résultat du test pour GEC : les intervalles d'acceptation à $99,3\%$ de chacune des distributions empiriques $\hat{D}_z$ sont

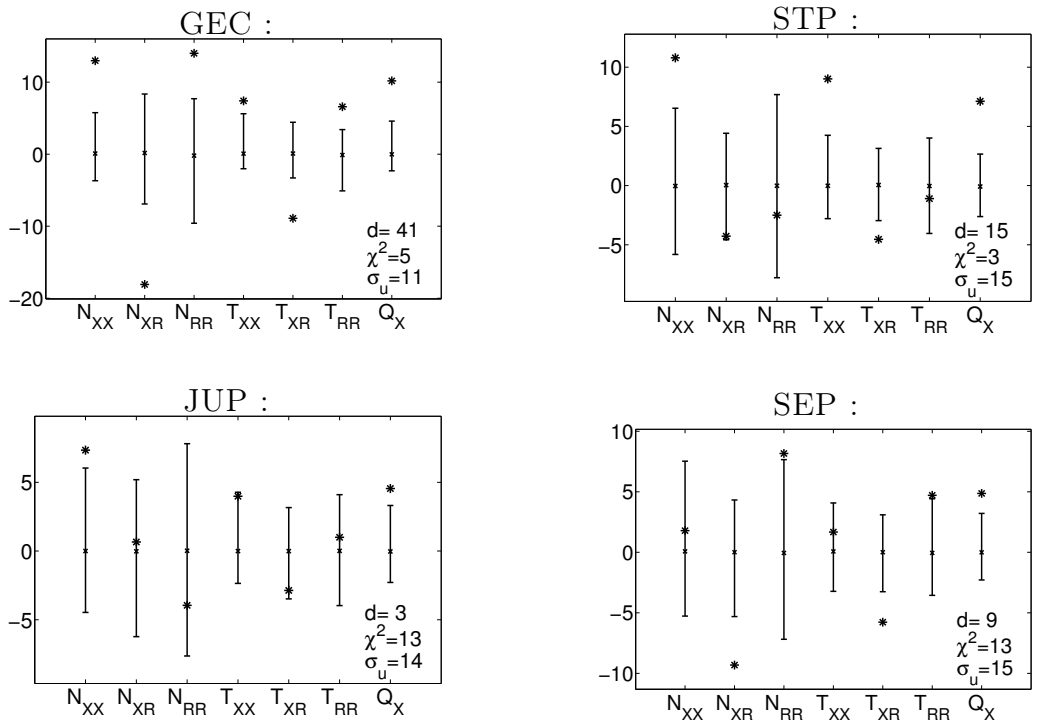


Figure 3.11: Résultats du test avec la contrainte de cardinal pour les quatres groupes : GEC, STP, JUP, et SEP. Pour chaque observable normalisée z, la distribution obtenue à l'aide des échantillons bootstrap est indiquée par son intervalle d'acceptation à $1-\alpha' = 1-\frac{\alpha}{F-f} = 1-\frac{0.05}{7} = 99.3\%$ et sa médiane (petite croix). La valeur mesurée z^0 est indiquée avec des étoiles. En bas à droite de chaque figure, le résultat du test est résumé avec le triplet (d, χ^2, σ_u).

tracés en noir (la médiane est symbolisée par une petite croix noire). Les valeurs effectivement mesurées pour GEC sont tracées à l'aide d'étoiles noires. Finalement, le triplet (d, χ^2, σ_u) est donné en bas à droite de la figure. Les trois autres figures montrent les résultats du test pour chacun des trois autres groupes (STP, JUP, SEP). Les résultats de tous les groupes correspondent au cas 1 ($d > 0, \forall(\chi^2, \sigma_u, M)$) de la partie 2.2.6 : les quatre tests sont valables.

Pour les quatre groupes, d est non-nulle et l'hypothèse nulle est rejetée. En d'autres termes, aucun de ces groupes d'individus ne se comporte comme un groupe aléatoire de même taille. Ce qui n'est pas surprenant, étant attendu qu'ils se comportent comme des communautés. C'est bien ce qu'on observe : comparés aux échantillons bootstrap, chaque groupe tend à avoir des valeurs Q_X, N_{XX}, N_{RR}, T_{XX}, T_{RR} plus grandes et des valeurs N_{XR}, T_{XR} plus petites. Il est remarquable que la divergence de GEC est clairement plus grande que celle des autres groupes : ce premier test, même si naïf, donne des pistes expliquant la particularité de GEC par rapport aux autres.

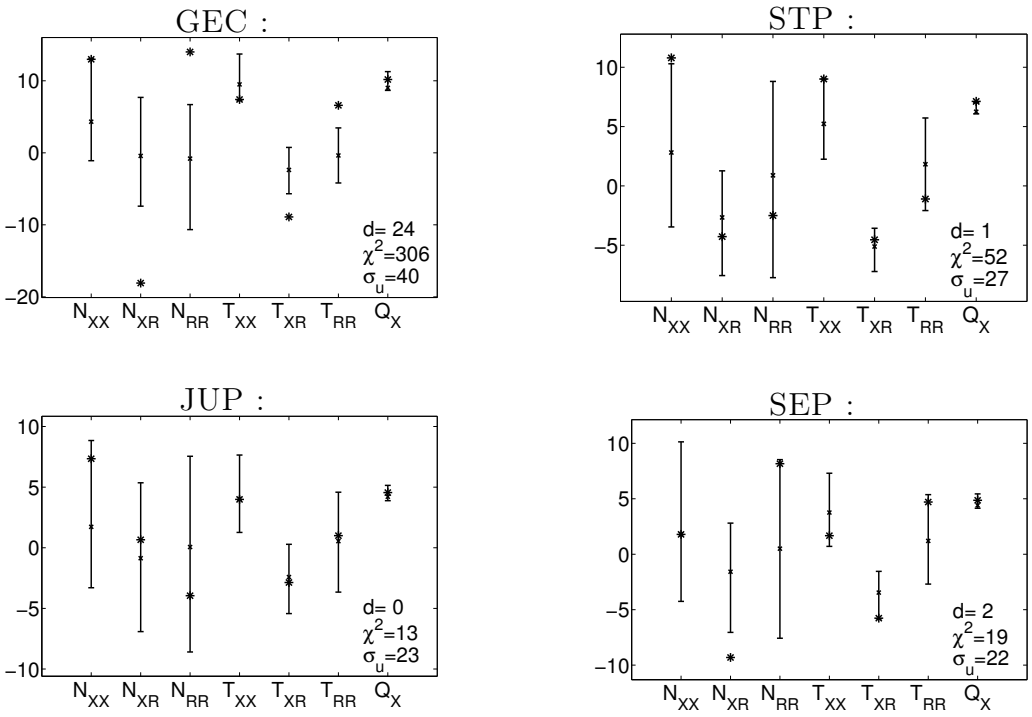


FIGURE 3.12: Résultats du test avec les contraintes de cardinal et de modularité (avec $\delta = 15\%$) pour les quatres groupes : GEC, STP, JUP, et SEP. Pour chaque observable normalisée z, la distribution obtenue à l'aide des échantillons bootstrap est indiquée par son intervalle d'acceptation à $1 - \alpha' = 1 - \frac{\alpha}{F-f} = 1 - \frac{0.05}{6} = 99.2\%$ et sa médiane (petite croix). La valeur mesurée z^0 est indiquée avec des étoiles. En bas à droite de chaque figure, le résultat du test est résumé avec le triplet (d, χ^2, σ_u).

4.2.3 Contraintes plus fines

Afin de discriminer plus précisément le comportement de GEC par rapport aux autres groupes, nous faisons appel à des hypothèses nulles plus fines, i.e. conservant plus de corrélations (avec des contraintes plus fortes sur les échantillons bootstraps). Comme discuté dans la partie 3, le paramètre de la force des contraintes, δ, est choisi égal à la valeur seuil δ^* associée à l'hypothèse nulle considérée, lue dans le Tableau 3.1.

L'hypothèse nulle que nous considérons à présent est celle sur la modularité. Formellement, elle s'écrit : H_0 *: GEC se comporte comme tout autre groupe de même taille et de même modularité.* Cette hypothèse nulle montre des performances correctes simulées sur le modèle de Chung-Lu pondéré en prenant $\delta = \delta^* = 0.15$ (voir la partie 3).

La Fig 3.10.b montre les deux mêmes histogrammes que la Fig. 3.10.a mais dans le cas où il existe une telle contrainte sur les échantillons bootstrap. Comme attendu, des valeurs plus grandes sont mesurées pour σ_u et χ^2, mais qui restent plus petites que les valeurs maximales autorisées estimées dans la partie 3 et rappelées dans le Tableau 3.1-b : $\sigma_u < \sigma_u^* = 60$ et $\chi^2 < \chi^{2*} = 950$.

Les résultats du test pour les quatre groupes sont résumés sur la Fig. 3.12. Remarquons d'abord que les intervalles d'acceptation ne sont pas centrés autour de

Hypothèse nulle	GEC	STP	JUP	SEP
aucune contrainte (cardinal seul)	(41, 5, 11)	(15, 3, 15)	(3, 13, 14)	(9, 13, 15)
contrainte Q_X avec $\delta^* = 0.15$	(24, 306, 40)	(1, 52, 27)	(0, 13, 23)	(2, 19, 22)
contrainte N_{XX} avec $\delta^* = 0.05$	(69, 1960, 94)	(15, 287, 97)	(4, 110, 68)	(21, 13, 23)
contrainte T_{XX} avec $\delta^* = 0.05$	(40, 277, 59)	(7, 728, 121)	(0, 7, 52)	(15, 12, 30)

Table 3.4: Résultats résumés de la partie 3 pour différents types de contraintes. Chaque entrée du tableau donne le triplet (d, χ^2, σ_u).

zéro ; ils ont en effet besoin d'être en accord avec une grande modularité (typiquement : N_{XX}, T_{XX} grands et N_{XR}, T_{XR} petits). Les résultats du test pour les quatre groupes correspondent aux cas 1 ou 2 de la partie 2.2.6 : les quatre tests sont valables. La divergence de JUP devient nulle quand on corrige le test pour prendre en compte sa grande modularité : JUP agit comme n'importe quel groupe de même modularité. En ce sens il est normal. Les divergences de STP et SEP sont dix fois infértieures à celle de GEC. Ceci montre que le comportement de GEC est bien particulier en comparaison avec d'autres groupes.

Les deux autres hypothèses nulles (imposant N_{XX} ou T_{XX}) sont deux manières différentes de garder les corrélations dues à la quantité d'interactions entre les nœuds de chaque groupe. Chacune des contraintes est imposée avec un $\delta = \delta^*$ lu dans le Tableau 3.1-b. Les résultats sont résumés dans le Tableau 3.4. Tous les résultats correspondent au cas 1 $(d > 0, \forall(\chi^2, \sigma_u))$ ou au cas 2 $(d = 0, \chi^2 < \chi^{2*}, \sigma_u < \sigma_u^*)$ discutés dans la partie 2.2.6. La divergence obtenue pour GEC est, pour ces deux hypothèses nulles supplémentaires, bien supérieure à la divergence obtenue pour les trois autres groupes.

Même si la contrainte sur la modularité est celle qui discrimine le mieux GEC par rapport aux autres groupes, les trois autres tests proposent des arguments supplémentaires pour montrer que le comportement de GEC est bien anormal. Les résultats de tous les tests sont cohérents, et montrent non seulement que GEC se comporte différemment, mais explicite exactement à quel niveau ce groupe se comporte différemment. Par exemple, sous la contrainte de modularité, les intervalles d'acceptation pour GEC montrent qu'il a des valeurs de N_{RR}, T_{RR} particulièrement élevées et des valeurs de N_{XR}, T_{XR} particulièrement basses. En revanche, T_{XX} et N_{XX} ne semblent pas en cause dans cette différence de comportement. Les raisons exactes de sa différence sont soulignées grâce à la méthode proposée.

5 Discussion

L'originalité et la difficulté de cette méthode se résument très bien en récapitulant la liste des trois résultats possibles du test statistique :

- CAS 1 : $d > 0, \forall(\chi^2, \sigma_u, M)$. Dans ce cas, le test est rejeté. Ce cas est en fait le plus simple de tous, et est celui qui est ni soumis à interprétation, ni soumis à des choix arbitraires. Si la divergence est non-nulle, même si l'espace bootstrap est petit et que les échantillons sont très similaires à X^0, X^0 est quand même

assez différent des échantillons pour rejeter l'hypothèse nulle.
- Ce sont les cas où $d = 0$ qui sont plus compliqués, et qui ont nécessité tous les efforts de la partie 3 pour essayer de faire la différence entre un test non-rejeté parce que le groupe est réellement normal (CAS 2 : $\chi^2 < \chi^{2*}, \sigma_u < \sigma_u^*, M_l \leq M \leq M_u$) et un test non-rejeté parce que l'espace bootstrap est trop petit (CAS 3 : $\chi^2 > \chi^{2*}$ et/ou $\sigma_u > \sigma_u^*$, et/ou $M \notin [M_l, M_u]$). Ces deux derniers cas sont malheureusement soumis à plusieurs choix arbitraires que nous avons dû effectuer, notamment pour déterminer les bornes de M et pour les quantiles choisis pour définir la limite entre groupes communs et trop rares.

Nous n'avons pas la prétention ici d'avoir mis au point une méthode de rééchantillonnage robuste et entièrement aboutie. En effet, cette méthode est fragile dans le sens où on ne dispose d'aucune garantie théorique quant à ses performances. Aussi, le choix du modèle de graphes utilisé – les graphes de Chung-Lu pondérés – est tout à fait discutable. Nous avons fait ce choix pour garder certaines corrélations que nous pensions importantes dans le cadre de l'étude, mais d'autres choix peuvent être justifiables également, du simple Erdös-Rényi avec le même degré moyen, à des modèles de graphes conservant des caractéristiques plus complexes comme la moyenne de la longueur des plus courts chemins, ou la première valeur propre de la matrice d'adjacence W [228, 97]. Finalement, il est nécessaire de bien être conscient des tenants et aboutissants de la méthode pour proposer une juste interprétation des résultats obtenus ; ce qui est souvent le cas pour de nombreuses méthodes statistiques.

Une autre remarque : nous rangeons notre méthode dans la catégorie des méthodes de bootstrap non-paramétrées dans la mesure où le groupe d'intérêt X^0 est comparé à des groupes issus du graphe lui-même : nous n'avons pas besoin de modéliser le comportement d'un groupe pour générer des échantillons bootstrap. Néanmoins, nous avons bel et bien utilisé un modèle de graphe pour, d'une part valider la méthode, et pour trouver les valeurs seuil δ^*, χ^{2*} et σ_u^* qui s'avèrent être importantes pour l'interprétation du résultat du test statistique. Nous pouvons éventuellement considérer le test en lui-même comme non-paramétrique, mais son interprétation comme étant paramétrique.

En ce qui concerne l'étude du réseau de contacts humains SLC, nous avons montré dans un premier temps que certaines de ces distributions sont similaires à celles issues d'étude de réseaux similaires : les distributions de temps de contact, de temps d'intercontact du graphe dynamique ; les distributions de degré, de poids des liens du graphe agrégé, ou la corrélation entre force et degré d'un nœud sont autant de caractéristiques robustes et universelles. Nous avons également profité de la localisation spatiale des antennes pour caractériser les différences dans les distributions de contacts intra-DPP, intra-GEC et entre les deux conférences, selon le lieu de contacts. Nous avons mis en évidence, sans surprise, que les contacts entre les conférences, étaient particulièrement longs dans le hall aux posters, seul lieu véritablement partagé entre les deux conférences.

L'application de la méthode de bootstraps sur ce réseau dans le but de caractériser à quel point GEC agit de manière anormale au sein du réseau a donné des résultats également attendus. Comme le montre le test de même cardinal, le groupe GEC a un comportement communautaire fort, mais au même titre que les trois groupes de DPP identifiés en fonction des âges : STP, JEP et SEP. Afin de discriminer GEC par rapport à ces trois groupes, il a fallu considérer des hypothèses nulles plus fines, plus précises. GEC ne se comporte clairement pas comme un groupe aléatoire de même cardinal et de même modularité : son caractère communautaire n'est pas la seule raison de son comportement anormal. Alors que les trois autres groupes ressemblent beaucoup plus à des groupes de même cardinal et même modularité. Chacun des tests effectués est autant d'arguments qui nous permettent de conclure que GEC a un comportement anormal au sein du réseau, ce qui met en avant à quel point les deux conférences n'ont pas véritablement interagi.

L'intérêt de ce travail est d'avoir mis au jour les difficultés que l'on peut rencontrer quand on essaie de définir une méthode de bootstrap sur réseau. On propose ici une version possible, qui n'est certes pas pleinement satisfaisante, mais qui a néanmoins abouti à des résultats intéressants, et qui a l'avantage d'être facilement adaptable à tout type de réseaux et à de nombreuses problématiques. Nous avons su séparer les difficultés surmontables des difficultés qui nécessitent des choix arbitraires. Un des développements que nous avons envisagé et qui serait très intéressant à ajouter à la méthode est de prendre en compte des observables réellement dynamiques comme le temps de contact moyen au sein d'un groupe, ou le temps d'intercontact moyen au sein d'un groupe entre tel jour et tel jour. Ce serait une manière de faire la différence entre deux groupes qui présentent des caractéristiques "agrégées" similaires pour des raisons dynamiques potentiellement très différentes.

Conclusion et perspectives

« Que sçai-je ? »

– M. de Montaigne, *Essais*, II, 12

Les différents travaux présentés dans ce manuscrit ont pour point commun :
- l'objet d'étude : les graphes,
- l'éclairage apporté venu du traitement du signal.

C'est un travail qui s'inscrit dans la continuité des efforts récemment menés en traitement du signal sur graphes [18]. Je me suis concentré tout au long de cette thèse à adapter ces outils émergents, et parfois à en créer, dans le but de participer au rapprochement du traitement du signal sur graphes à certaines problématiques de la science des réseaux.

Une des contributions majeures de cette thèse est le développement d'une méthode de détection multiéchelle de communautés dont une toolbox Matlab est téléchargeable sur mon site internet [13], accessible à tous. Rappelons les points clés. Nous avons vu dans la partie 4.4 du chapitre 2 que l'algorithme de base résumé dans la partie 4.3.4 peut être grandement amélioré par la transformée en ondelettes de quelques vecteurs aléatoires. L'amélioration existe à plusieurs niveaux :
- en passant par quelques vecteurs aléatoires, nous gagnons en temps de calcul. Nous avons remarqué que le nombre de vecteurs aléatoires nécessaires dépendait de la résolution voulue par l'utilisateur. Si l'utilisateur ne souhaite pas voir des échelles plus fines qu'une dizaine de communautés, il suffit de calculer la transformée en ondelettes rapide d'une dizaine de vecteurs aléatoires, au lieu de calculer autant de transformées en ondelettes que de nœuds dans le graphe ;
- l'aléatoire de ces quelques vecteurs rend l'algorithme stochastique, ce qui nous permet de définir une stabilité à chaque échelle : à une échelle donnée, si quel que soit le jeu de vecteurs aléatoires donné, on retombe sur la même partition, c'est qu'elle est stable et qu'elle mérite d'être analysée ;
- conceptuellement, il est très intéressant de remarquer que la transformée en ondelettes de chaque vecteur aléatoire agit comme une mesure du graphe, et

qu'un ensemble restreint de mesures permet tout de même de remonter à la structure en communautés du graphe. Le lien avec l'acquisition comprimée (en anglais *compressive sensing* [62]) est remarquable. Tout semble se comporter comme si la structure en communautés est le signal parcimonieux que l'on cherche à reconstruire en acquisition comprimée, même si cette analogie reste encore à développer.

Nous avons également montré que cette méthode est aussi performante que l'état de l'art sur les modèles de graphes hiérarchiques classiques. Nous ne nous sommes pas attachés à chercher les classes de graphes pour lesquels notre méthode est plus performante que les autres, ceci aurait eu un intérêt limité. Le fait qu'elle présente des performances similaires à l'état de l'art est déjà très positif, d'autant plus que notre méthode présente d'autres avantages conceptuels :

- la notion d'échelle est rigoureusement définie comme le paramètre de dilatation de filtres dans l'espace de Fourier du graphe ;
- la transformée en ondelettes permet d'analyser des signaux définis sur le graphe. Nous en avons ici détourné l'utilisation pour analyser la structure du graphe. Nous avons ainsi montré qu'il est possible d'utiliser ces outils d'analyse de signaux sur graphe pour sonder la topologie du graphe sous-jacent. Dans un contexte où les exemples de signaux réellement définis sur des graphes sont encore peu nombreux, et où les exemples d'analyse de structures de graphes sont au contraire florissants, nous avons développé un outil potentiellement utile à un grand nombre d'utilisateurs ;
- finalement, la partie 8 montre que le formalisme que nous avons développé permet une généralisation des méthodes multiéchelles existantes, en les réinterprétant sous forme de bancs de filtres dans l'espace de Fourier.

Nous avons détaillé dans la partie 9 les perspectives possibles de ces travaux. Nous avons vu qu'une suite logique de ce travail est de proposer une cascade multirésolution de filtres, qui sous-échantillonne le signal en ne prenant qu'une seule valeur par communauté.

Une deuxième contribution de cette thèse a été de développer une méthode de rééchantillonnage entièrement pilotée par les données pour les groupes de nœuds dans des graphes. Une fois un graphe et un groupe de nœuds donnés, l'idée est de comparer ce groupe à d'autres groupes similaires (similarité définie par une hypothèse nulle) et de tester si oui ou non, le groupe étudié se comporte statistiquement comme les autres groupes du graphe. Dans un contexte où nous n'avons accès qu'à une seule réalisation d'un processus complexe et en partie aléatoire, l'intérêt de la méthode que nous avons développé est double :

- elle nous permet de décider, à un niveau de confiance donné, si oui ou non un groupe se comporte anormalement. C'est une contribution intéressante en détection d'anomalies.
- elle nous permet aussi, étant donnée une hypothèse nulle, de calculer empiriquement les intervalles de confiance de caractéristiques choisies.

Nous avons vu dans la partie 2.2.6 du chapitre 3 que la grande difficulté réside dans le compromis nécessaire entre la précision d'une hypothèse nulle et la taille de l'espace bootstrap : plus une hypothèse nulle est précise (et donc intéressante pour

l'analyste), plus l'espace bootstrap est restreint, et donc plus le test statistique perd de sa puissance. Il s'avère ainsi nécessaire de contrôler la taille de l'espace bootstrap, ce que nous faisons en la mesurant indirectement à l'aide du triplet (χ^2, σ_u, M). Toute la question est alors de savoir quelles sont les valeurs seuil de ce triplet au-delà desquelles on ne peut tirer de conclusions du test statistique. Ces valeurs seuil sont estimées empiriquement sur des graphes de Chung-Lu pondérés, un modèle de graphe aléatoire que nous avons introduit.

Cette thèse a également permis d'ajouter deux nouveaux jeux de données à l'ensemble des données de Sociopatterns, grâce à deux déploiements :

- en novembre 2011 : la semaine de mesures à la conférence de Salt Lake City, étudiée en détail dans le chapitre 3 ;
- en juin 2013 : deux fois une semaine de mesures au sein du Laboratoire de Physique de l'ENS Lyon, où nous avons mesuré les interactions sociales entre une centaine de chercheurs, post-doctorants, doctorants et stagiaires du laboratoire. Ce jeu de données n'a pas été évoqué dans ce manuscrit mais a fait l'objet du stage de licence 3 de Rémi Menaut que j'ai co-encadré [171].

Les autres contributions de cette thèse se rattachent de près ou de loin à ces thématiques. La collaboration avec les collègues du Japon, dont le fruit est présenté en Annexe D, est peut-être la contribution qui s'en éloigne le plus. Mais nous retrouvons néanmoins l'aspect graphe des réseaux de capteurs. De plus, cette collaboration a inspiré le travail d'EMD sur graphes présenté en Annexe E. Finalement, j'ai participé à d'autres travaux non évoqués dans ce manuscrit, dont une contribution à un chapitre de livre sur les réseaux dynamiques, plus particulièrement sur les données des vélos partagés de Lyon, les Vélo'v [35].

Je ne peux terminer ce manuscrit sans deux mises en perspective éthiques. Premièrement, de nombreux réseaux sont aujourd'hui mesurés, par des agences de renseignement comme la NSA [7] ou la DGSE [6] ; par des entreprises de l'Internet comme Google qui ont accès par exemple à son réseau de courriers électroniques, ou des entreprises de télécommunication comme Orange. Afin d'argumenter que les vies privées ne sont pas espionnées, ces institutions avancent souvent qu'ils n'enregistrent que les métadonnées, c'est-à-dire qu'ils n'analysent pas le contenu des courriers électroniques ou des conversations téléphoniques, mais enregistrent uniquement l'expéditeur, le destinataire, l'heure et la date de l'envoi et éventuellement quelques autres informations annexes. Mais il suffit par exemple de faire tourner, sur sa propre boîte mail, le code de l'application Immersion [2] créée par des chercheurs du MIT pour se rendre compte que les métadonnées permettent, à elles seules, de bien reconstruire son propre réseau social. Un deuxième argument avancé par certaines institutions est d'affirmer que leurs bases de données sont anonymisées, c'est-à-dire que les analystes ayant accès aux données ont accès à la structure en réseau des données, mais ne connaissent pas l'identité des nœuds du réseau. Une étude est intéressante à lire à ce sujet [57] : dans une base de données constituée des données de mobilité (mesurée via les téléphones portables) d'un demi million de personnes sur une durée de quinze mois, il suffit de savoir où était une personne donnée à quatre moments

différents (ce qui est peu en quinze mois !), pour pouvoir l'identifier dans la base de données avec 95% de chances. L'anonymat est donc tout relatif, tant la structure de nos connexions au sein de notre réseau social nous sont en fait spécifiques. Les métadonnées permettent de reconstruire une structure en réseau qui, comme nous le voyons à travers ces exemples, mais comme nous avons aussi pu le remarquer à travers tout ce manuscrit, contient beaucoup d'informations. Nos vies privées sont en partie codées dans ces structures en réseau et il convient d'en prendre garde.

Comme indiqué en introduction, ce manuscrit s'intègre dans le domaine plus général de l'extraction automatique d'information. Je vais m'attarder sur l'exemple d'une requête Google, plus simple à discuter, mais la détection de communautés multiéchelles est également une forme d'extraction automatique d'informations. Le résultat d'une requête Google est une liste ordonnée de dix pages web (une quantité négligeable de personnes visitent la deuxième page de résultats [46]) parmi les plusieurs milliards de pages internet indexées. Alors que l'on sait que l'algorithme `PageRank`, qui est à la base de l'algorithme plus sophistiqué (et secret) de Google, donne du poids aux pages internet les plus référencées, n'y a-t-il pas un danger d'uniformisation des sources d'information, et par là même, un danger d'uniformisation des idées ? En effet, une page très référencée a plus de chances d'être encore plus référencée par la suite, et donc à gagner encore plus de poids dans l'algorithme, et ainsi de suite (c'est d'ailleurs la base du modèle d'attachement préférentiel de Barabási et Albert [24]). Cette page devient en quelque sorte très référencée grâce au simple fait qu'elle était déjà bien référencée à la base, pas forcément parce que son contenu est particulièrement pertinent. Ce phénomène d'amplification ne monte pas sur des piédestaux les informations les plus utiles ou les plus réfléchies de notre société, si on en croit les vidéos les plus vues de Youtube [12]. La meilleure solution serait de rester prudents face aux résultats d'une requête d'informations automatisée, de bien garder à l'esprit que ces systèmes ne sont que des des outils (aussi utiles et performants soient-ils), d'être conscients de leurs défauts et qualités, de ne pas accepter le premier résultat comme une vérité nécessaire. Mais l'augmentation du volume des données continue, et l'utilisation que nous faisons des algorithmes augmente en conséquence, si bien que nous avons de moins en moins le temps d'être prudents. L'automatisation a révolutionné notre accès à l'information, et va d'ailleurs de pair avec l'augmentation du volume de données actuelle. En effet, sans automatisation, il est inutile de stocker autant de données qu'aucun humain ne pourra jamais lire (pensons aux 100 ans de vidéo mises en ligne tous les six jours sur Youtube). Mais je pense que l'augmentation exponentielle des données augmente nécessairement la confiance que nous sommes contraints à donner à la machine qui en extrait l'information, et il convient de garder à l'esprit que l'outil est formidablement puissant mais présente aussi des écueils à éviter.

Les intérêts privés qui se jouent derrière l'ordre des résultats d'une requête Google, où chacun essaie d'obtenir un meilleur référencement parfois à l'aide d'entreprises spécialisées, sont une autre raison, à mes yeux, de garder un œil critique devant ces résultats. Une page web est en première position d'une requête parce qu'elle contient vraiment le "meilleur" contenu, ou parce que les gérants du site web ont dépensé plus d'argent et d'énergie que les autres pour être classés premiers ?

Finalement, je pense qu'il est important de poursuivre l'effort de recherche académique publique, libre et ouverte à tous, dans le domaine de l'extraction d'informations, afin que cette expertise reste la plus transparente possible ; il en va, je pense, de notre libre accès à l'information.

Annexe A

Mesures de similarité entre deux partitions

Considérons $\mathcal{C}$ et $\mathcal{C}'$ deux partitions d'un même graphe. Notons C_k (resp. C'_k) la $k^{\text{ème}}$ communauté de $\mathcal{C}$ (resp. $\mathcal{C}'$) et n_k (resp. $n_{k'}$) son nombre de nœuds. Il existe différentes manières de mesurer à quel point ces deux partitions sont similaires.

Les méthodes basées sur le décompte de paires de nœuds. Ces méthodes comptent dans un premier temps :

- N_{11} le nombre de paires de nœuds qui sont dans la même communauté à la fois dans $\mathcal{C}$ et dans $\mathcal{C}'$.
- N_{00} le nombre de paires de nœuds dans des communautés différentes dans $\mathcal{C}$ et dans $\mathcal{C}'$.
- N_{10} le nombre de paires de nœuds qui sont dans la même communauté dans $\mathcal{C}$ mais pas dans $\mathcal{C}'$.
- N_{01} le nombre de paires de nœuds qui sont dans la même communauté dans $\mathcal{C}'$ mais pas dans $\mathcal{C}$.

Ces quatre nombres somment toujours à $N(N-1)/2$. Wallace [222] propose deux critères de similarités :

$$\mathcal{W}_I(\mathcal{C},\mathcal{C}') = \frac{N_{11}}{\sum_k n_k(n_k-1)/2}, \tag{A.1}$$

$$\mathcal{W}_{II}(\mathcal{C},\mathcal{C}') = \frac{N_{11}}{\sum_{k'} n_{k'}(n_{k'}-1)/2}. \tag{A.2}$$

Fowlkes et Mallows [85] proposent de symmétriser ces deux critères assymétriques en prenant la moyenne géométrique des deux. L'*indice de Fowlkes*, compris entre 0 et 1, s'écrit :

$$\mathcal{F}(\mathcal{C},\mathcal{C}') = \sqrt{\mathcal{W}_I(\mathcal{C},\mathcal{C}')\mathcal{W}_{II}(\mathcal{C},\mathcal{C}')}. \tag{A.3}$$

Un autre indice de similarité est l'*indice de Rand* [172] :

$$\mathcal{R}(\mathcal{C},\mathcal{C}') = \frac{N_{11}+N_{00}}{N(N-1)/2}. \tag{A.4}$$

Compris entre 0 et 1, cet indice vaut 1 quand les deux partitions sont égales et vaut zéro uniquement quand une des partitions est constituée d'une seule partition à N

nœuds et l'autre partition de N partitions à un nœud. Ce scénario est très extrême. De plus, un effet statistique indésirable de cet indice est que deux partitions en deux commuanutés ont plus de chances d'avoir un indice de similarité élevé plutôt que deux partitions en dix communautés. En fait, une des propriétés désirables d'un indice serait que l'espérance de l'indice de deux partitions aléatoires soit nulle. Hubert et Arabie [110] proposent d'ajuster l'indice de Rand pour prendre en compte ces effets statistiques en définissant l'*indice de Rand ajusté* :

$$AR = \frac{\text{Rand} - \mathbb{E}(\text{Rand})}{1 - \mathbb{E}(\text{Rand})}, \tag{A.5}$$

où $\mathbb{E}(\text{Rand})$ est l'espérance de l'indice de Rand calculée sur tous les couples de partitions avec même nombre de communautés (K, K') et mêmes tailles de communautés $\big(\{n_k\}_{k\in[1,K]}, \{n_{k'}\}_{k'\in[1,K']}\big)$. $AR(\mathcal{C}, \mathcal{C}')$ est compris entre -1 et 1, vaut 1 si les deux partitions sont égales, vaut en moyenne zéro si les deux partitions sont tirées aléatoirement, vaut -1 si les deux partitions sont pathologiquement différentes. Évoquons un dernier indice historique mais encore régulièrement utilisé, l'*indice de Jaccard* [115] :

$$\mathcal{J}(\mathcal{C}, \mathcal{C}') = \frac{N_{11}}{N_{11} + N_{01} + N_{10}}. \tag{A.6}$$

L'indice de Jaccard est compris entre 0 et 1.

La variation d'information est une méthode qui se base sur un calcul entropique [147]. La probabilité qu'un nœud se retrouve dans C_k si la partition était aléatoire est $P(k) = n_k/N$. L'entropie associée à la partition $\mathcal{C}$ s'écrit :

$$H(\mathcal{C}) = -\sum_{k=1}^{K} P(k) \log P(k). \tag{A.7}$$

Notons $P(k, k')$ la probabilité jointe qu'un nœud se retouve à la fois dans C_k et C'_k :

$$P(k, k') = \frac{|C_k \cap C'_k|}{N}. \tag{A.8}$$

On définit l'information mutuelle $I(\mathcal{C}, \mathcal{C}')$, l'information que chaque partition a de l'autre :

$$I(\mathcal{C}, \mathcal{C}') = -\sum_{k'=1}^{K'} \sum_{k=1}^{K} P(k, k') \log P(k, k'). \tag{A.9}$$

Si les deux partitions sont égales, alors :

$$I(\mathcal{C}, \mathcal{C}') = H(\mathcal{C}) = H(\mathcal{C}'). \tag{A.10}$$

Et finalement, on peut définir la mesure de variation d'information $VI(\mathcal{C}, \mathcal{C}')$:

$$VI(\mathcal{C}, \mathcal{C}') = H(\mathcal{C}) + H(\mathcal{C}') - 2I(\mathcal{C}, \mathcal{C}'). \tag{A.11}$$

Le plus proche VI est de zéro, le plus similaire sont les partitions. VI est borné entre 0 et $\log N$.

Cette liste est loin d'être exhaustive et donne juste une petite idée de ces mesures. Choisir le type de mesure à utiliser dépend de l'application, et il n'y a pas de réponse claire à ce sujet [209, 110].

Annexe B

Modèle de graphe aléatoire : le modèle de Chung-Lu pondéré

Soit $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ un graphe et $\{d_i\}_{i=1,\dots,V}$ sa liste de degré. Nous rappelons qu'un graphe de Chung-Lu [47, 151] associé à $\mathcal{G}$ est un graphe aléatoire binaire avec la même espérance de liste de degrés. Pour cela, les degrés de chaque nœud sont réalloués aléatoirement et chaque lien (ij) est créé avec une probabilité $\min(1, \frac{d_i d_j}{2d_{tot}})$ où $d_{tot} = \frac{1}{2}\sum_i d_i$ est l'espérance du nombre total de liens.

Le modèle de Chung-Lu est défini uniquement pour les graphes binaires et nous étendons à présent ce modèle aux graphes pondérés : nous proposons ici le modèle de Chung-Lu pondéré. En pratique, nous commençons par créer un graphe aléatoire de Chung-Lu comme décrit au paragraphe précédent qui nous assure que l'espérance de la distribution de degrés est bien respectée. Ensuite, il suffit d'attribuer un poids à chaque lien. Dans des graphes réels, il existe très souvent une corrélation entre le degré et la force, cette corrélation n'est pas forcément trivialement linéaire et est une des caractéristiques du graphe [26]. Pour rendre compte de cette corrélation et proposer un modèle de graphe aléatoire d'autant plus réaliste, nous suggérons de conserver cette corrélation mesurée empiriquement sur le graphe $\mathcal{G}$. Nous estimons donc, à partir de $\mathcal{G}$, pour tout degré d, la distribution $P_d(w)$ des poids des liens connectés aux nœuds de degré d. S'il n'existe pas assez de nœuds de degré d dans $\mathcal{G}$ pour estimer cette distribution correctement, nous considérons les 50 nœuds qui ont des degrés le plus proche d. Ainsi, une fois le graphe binaire de Chung-Lu créé, considérer tour à tour les nœuds du graphe. Pour chaque nœud i de degré d_i, tirer d_i poids de la distribution appropriée $P_{d_i}(w)$ et les attribuer aléatoirement aux liens qui n'ont pas encore de poids attribué. En effet, si i est connecté à un nœud j qui a déjà été visité par l'algorithme, alors le lien (ij) a déjà un poids qui a été tiré à partir de la distribution $P_{d_j}(w)$. Nous obtenons ainsi un graphe aléatoire de Chung Lu pondéré qui est une version aléatoire de $\mathcal{G}$ et qui a la même espérance de distribution de degrés, la même espérance de corrélation dégré/force, et une distribution de forces similaire. À titre d'exemple, nous montrons sur la Fig. B.1, la force moyenne des nœuds en fonction de leur degré pour 3 réseaux sociaux mesurés dans des conférences. Nous superposons le résultat obtenu pour une réalisation d'un graphe de Chung-Lu pondéré, obtenue à partir des distributions mesurées sur le jeu de données "SLC" (voir la partie 4.1 du chapitre 3) : la corrélation est conservée.

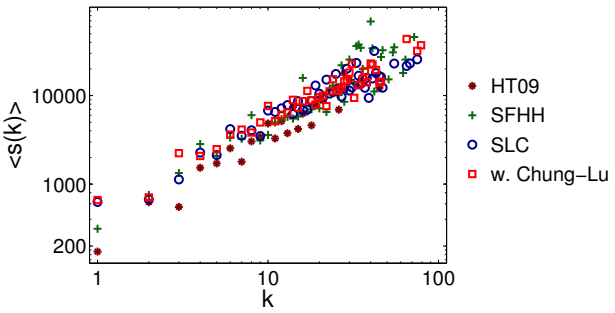


Figure B.1: Force moyenne en fonction du degré des nœuds dans trois réseaux sociaux mesurés lors de conférences. Les carrés rouges montrent en sus le résultat pour une réalisation d'un graphe de Chung-Lu pondéré générée à partir des distributions empiriques du réseau "SLC".

Annexe C

La correction de Bonferroni

Soit un ensemble de m hypothèses nulles (dépendantes ou non) $H_1, H_2, \ldots, H_m$, et $p_1, p_2, \ldots, p_m$ leur p-valeur respectives. La correction de Bonferroni, du nom du mathématicien italien mais proposée en premier par Dunn [64], donne des conditions suffisantes pour rejeter l'ensemble des hypothèses nulles à une incertitude α près : il suffit de rejeter chacune des hypothèses nulles à α/m. *Id est*, si $\forall i \quad p_i < \alpha/m$, alors la famille des hypothèses nulles est rejetée avec une incertitude α. Cette correction est d'autant plus pessimiste que les hypothèses nulles sont corrélées.

Annexe D

Strip, Bind, and Search : A Method for Identifying Abnormal Energy Consumption in Buildings

Romain Fontugne[1,5], Jorge Ortiz[2], Nicolas Tremblay[3], Pierre Borgnat[3]
Patrick Flandrin[3], Kensuke Fukuda[4], David Culler[2] and Hiroshi Esaki[1]

[1]The University of Tokyo [2]University of California, Berkeley
[3]CNRS, École Normale Supérieure de Lyon [4]National Institute of Informatics
[5]Japanese-French Laboratory for Informatics

Published in the proceedings of the IPSN'13 conference
April 8–11, 2013, Philadelphia, Pennsylvania, USA.

A typical large building contains thousands of sensors, monitoring the HVAC system, lighting, and other operational sub-systems. With the increased push for operational efficiency, operators are relying more on historical data processing to uncover opportunities for energy-savings. However, they are overwhelmed with the deluge of data and seek more efficient ways to identify potential problems. In this paper, we present a new approach called the Strip, Bind and Search (SBS) ; a method for uncovering abnormal equipment behavior and in-concert usage patterns. SBS uncovers relationships between devices and constructs a model for their usage pattern relative to other devices. It then flags deviations from the model. We run SBS on a set of building sensor traces ; each containing hundred sensors reporting data flows over 18 weeks from two separate buildings with fundamentally different infrastructures. We demonstrate that, in many cases, SBS uncovers misbehavior corresponding to inefficient device usage that leads to energy waste. The average waste uncovered is as high as 2500 kWh per device.

Contents

1 Introduction **156**
2 Problem description **158**
3 Methodology **159**
3.1 Strip and Bind 159
3.2 Search 163
4 Data sets **164**
4.1 Engineering Building 2 - Todai 165
4.2 Cory Hall - UC Berkeley 165
4.3 Data pre-processing 165
5 Experimental Results **166**
5.1 Shortcomings 166
5.2 Device behavior at different time scales 167
5.3 Anomalies 168
6 Related work **172**
7 Discussion **173**
8 Conclusions **174**

1 Introduction

Buildings are one of the prime targets to reduce energy consumption around the world. In the United States, the second largest energy consumer in the world, buildings account for 41% of the country's total energy consumption [219]. The first measure towards reducing the building's energy consumption is to prevent electricity waste due to the improper use of the buildings equipment.

Large building infrastructure is usually monitored by numerous sensors. Some of these sensors enable building administrators to view device power-draw in real time. This allows administrators to determine proper device behavior and system-wide inefficiencies. Detecting misbehaving devices is crucial, as many are sources of energy waste. However, identifying these saving opportunities is impractical for administrators because large buildings usually contain hundreds of monitored devices producing thousands of streams and it requires continuous monitoring. As such, the goal of this work is to establish a method that automatically reports abnormal device-usage patterns to the administrator by closely examining all of the continuous power streams.

The intuition behind the proposed approach is that each service provided by the building requires a minimum subset of devices. The devices within a subset are used at the same time when the corresponding service is needed and a savings opportunity is characterized by the partial activation of the devices. For example, office comfort is attained through sufficient lighting, ventilation, and air conditioning. These are controlled by the lighting and HVAC (Heating, Ventilation, and Air Conditioning)

system. Thus, when the room is occupied both the air conditioner (heater on a cold day) and lights are used together and should be turned off when the room is empty. In principal, if a person leaves the room and turns off *only* the lights then the air conditioner (or heater) is a source of electricity waste.

Following this basic idea we propose *Strip, Bind and Search* (SBS), an unsupervised methodology that systematically detects electricity waste. Our proposal consists of two key components :

Strip and Bind The first part of the proposed method mines the raw sensor data, identifying inter-device usage patterns. We first *strip* the underlying traces of occupancy-induced trends. Then we *bind* devices whose underlying behavior is highly correlated. This allows us to differentiate between devices that are used together (high correlation), used independently (no correlation), and used mutually exclusively (negative correlation).

Search The second part of the method monitors devices relationships over time and reports deviations from the norm. It learns the normal inter-device usage using a robust, longitudinal analysis of the building data and detect anomalous usages. Such abnormalities usually present an opportunity to reduce electricity waste or events that deserve careful attention (e.g. faulty device).

SBS overcomes several challenges. First, noisy sensor traces that all share a similar trend, making direct correlation analysis non-trivial. Device energy consumption is mainly driven by occupancy and weather, all the devices display a similar daily pattern, in roughly overlapping time intervals and phases. Therefore, one of the main contributions of this work is uncovering the intrinsic device relationships by filtering out the dominant trend. For this task we use Empirical Mode Decomposition [106], a known method for de-trending time-varying signals.

Another key contribution of this work is in using SBS to practically monitor building energy consumption. Moreover, the proposed method is easy to use and functions in any building, as it does not require prior knowledge of the building nor extra sensors. It is also tuned through a single intuitive parameter.

We validate the effectiveness of our approach using 10 weeks of data from a modern Japanese building containing 135 sensors and 8 weeks of data from an older American building containing 70 sensors. These experiments highlight the effectiveness of SBS to uncover device relationships in a large deployment of 135 sensors. Furthermore, we inspect the SBS results and show that the reported alarms correspond to significant opportunities to save energy. The major anomaly reported in the American building lasts 18 days and accounts for a waste of 2500 kWh. SBS also reports numerous small anomalies, hidden deep within the building's overall consumption data. Such errors are very difficult to find without SBS.

In the rest of this paper, we detail the mechanisms of SBS (Section 3) before evaluating it with real data (Section 5) then we discuss different outcomes of the proposed methodology (Section 7) and conclude.

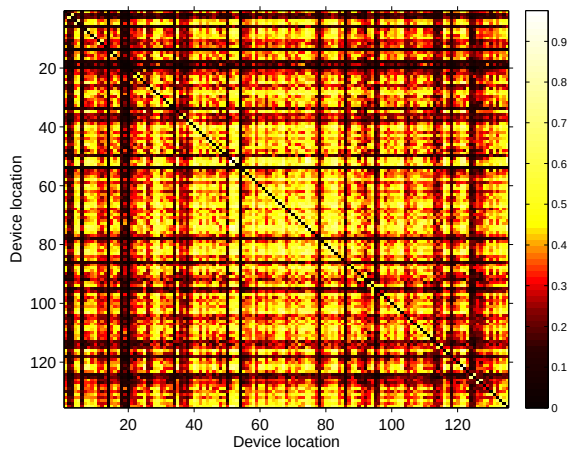


Figure D.1: Correlation coefficients of the raw traces from the Building 1 dataset (Section 4.1). The matrix is ordered such as the devices serving same/adjacent rooms are nearby in the matrix.

2 Problem description

The primary objective of SBS is to determine *how* device usage patterns are correlated across all pairs of sensors and discover when these relationships change. The naive approach is to run correlation analysis on pairs of sensor traces, recording their correlation coefficients over time and examining when there is a statistically-significant deviation from the norm. However, this approach does not yield any useful information when applied to *raw data traces*. For example, the two raw signals shown in Figure D.3 are from two independent HVAC systems, serving different rooms on different floors. Since each space is independently controlled, we expect their power-draw signals to be uncorrelated (or at least distinguishable from other signal pairs). However, their correlation coefficient (0.57), is not particularly informative – it is statistically similar to the correlation between itself and other signals in the trace.

Using a larger set of devices, Figure D.1 shows a correlation matrix with 135 distinct lighting and HVAC systems serving numerous rooms in a building (described later on in Section 4.1). The indices are selected such that their index-difference is indicative of their relative spatial proximity. For example, a device in location 1 is closer in the building to a device in location 2 than it is to a device in location 135. The color of the cell is the average pairwise correlation coefficient for devices in the row-column index. The higher the value, the lighter the color. Devices serving the same room are along the diagonal. Because these devices are used simultaneously, we expect high average correlation scores, lighter shades, along the diagonal figure. However, we observe no such pattern. Most of the signals are correlated with all the others and we see no discernible structure.

An explanation for this is that the daily occupant usage patterns drive these results. Figure D.3 demonstrates this more clearly. It shows two 1-week raw signals traces which feature the same diurnal pattern. This trend is present in almost every sensor trace, and, it hides the smaller fluctuations providing more specific patterns

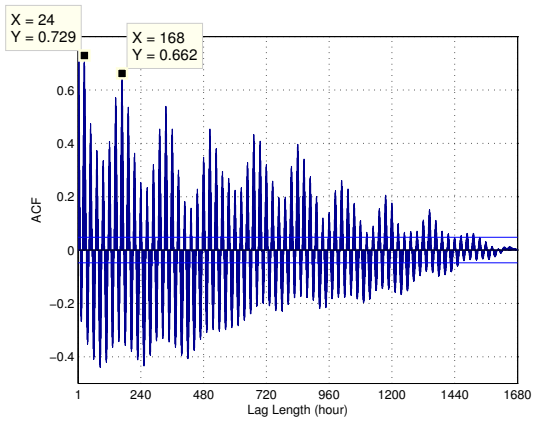


FIGURE D.2: Auto-correlation of a usual signal from the Building 1 dataset. The signal features daily and weekly patterns (resp. $x = 24$ and $x = 168$).

driven by local occupant activity. Upon deeper inspection, we uncovered several dominant patterns, common among energy-consuming devices in buildings [226]. Figure D.2 depicts the auto-correlation of a usual electric power signal for a device. The two highest values in the figure correspond to a lag of 24 hours and 168 hours (one week). Therefore, the signal has some periodicity and similar (though not equal) values are seen at daily and weekly time scales. The daily pattern is due to daily office hours and the weekly pattern corresponds to weekdays and weekends. Correlation analysis on *raw* signals cannot be used to determine meaningful inter-device relationships because periodic components act as non-stationary trends for high-frequency phenomenon, making the correlation function irrelevant. Such trends must be removed in order to make meaningful progress towards our aforementioned goals.

In the next section we describe SBS. We discuss *strip and bind* in section 3.1, which addresses de-trending and relationship-discovery. Then, we describe how we *search* for changes in usage patterns, in section 3.2, to identify potential savings opportunities.

3 Methodology

3.1 Strip and Bind

Discovering devices that are used in concert is non-trivial. SBS decomposes each signal into an additive set of components, called Intrinsic Mode Functions (IMF), that reveals the signal patterns at different frequency bands. IMFs are obtained using Empirical Mode Decomposition (see Figure D.3 and Section 3.1.1). We only consider IMFs with time scales shorter than a day, since we are interested in capturing short-scale usage patterns. Consequently, SBS aggregates the IMFs that fall into this specific time scale (see *IMF agg.* in Figure D.3). The resulting partial signals of different device power traces are compared, pairwise, to identify the devices that

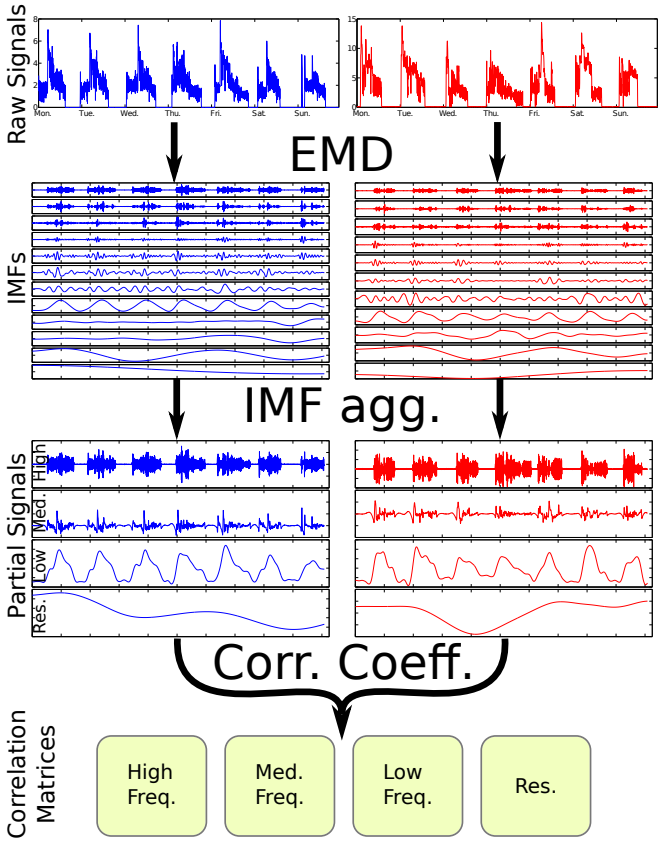


Figure D.3: *Strip and Bind* using two raw signals standing for one week of data from two different HVACs. (1) Decomposition of the signals in IMFs using EMD (top to bottom : c_1 to c_n) ; (2) aggregation of the IMFs based on their time scale ; (3) comparison of the partial signals (aggregated IMFs) using correlation coefficient.

show un/correlated usage patterns (see *Corr. Coeff.* in Figure D.3).

3.1.1 Empirical Mode Decomposition

Empirical Mode Decomposition (EMD) [106] is a technique that decomposes a signal and reveals intrinsic patterns, trends, and noise. This technique has been widely applied to a variety of datasets, including climate variables [135], medical data [31], speech signals [104, 100], and image processing [163]. EMD's effectiveness relies on its empirical, adaptive and intuitive approach. In fact, this technique is designed to efficiently decompose both non-stationary and non-linear signals without requiring any a priori basis functions or tuning.

EMD decomposes a signal into a set of oscillatory components called intrinsic mode functions (IMFs). An IMF satisfies two conditions : (1) it contains the same number of extrema and zero crossings (or differ at most by one) ; (2) the two IMF envelopes defined by its local maxima and local minima are symmetric with respect to zero. Consequently, IMFs are functions that directly convey the amplitude and frequency modulations.

EMD is an iterative algorithm that extracts IMFs step by step by using the so-

called sifting process [106] ; each step seeks for the IMF with the highest frequency by sifting, then the computed IMF is removed from the data and the residual data are used as input for the next step. The process stops when the residual data becomes a monotonic function from which no more IMF can be extracted.

We formally describe the EMD algorithm as follows :

1. Sifting process : For a current signal $h_0 = X$, let m_0 be the mean of its upper and lower envelopes as determined from a cubic-spline interpolation of local maxima and minima.
2. The estimated local mean m_0 is removed from the signal, giving a first component : $h_1 = h_0 - m_0$
3. The sifting process is iterated, h_1 taking the place of h_0. Using its upper and lower envelopes, a new local mean m_1 is computed and $h_2 = h_1 - m_1$.
4. The procedure is repeated k times until $h_k = h_{k-1} - m_{k-1}$ is an IMF according to the two conditions above.
5. This first IMF is designated as $c_1 = h_k$, and contains the component with shortest periods. We extract it from the signal to produce a residual : $r_1 = X - c_1$. Steps 1 to 4 are repeated on the residual signal r_1, providing IMFs c_j and residuals $r_j = r_{j-1} - c_j$, for j from 1 to n.
6. The process stops when residual r_n contains no more than 3 extrema.

The result of EMD is a set of IMFs c_i and the final residue r_n, such as :

$$X = \sum_{i=1}^{n} c_i + r_n$$

where the size of the resulting set of IMFs (n) depends on the original signal X and r_n represents the trend of the data (see *IMFs* in Figure D.3).

For this work we implemented a variant of EMD called Complete Ensemble EMD [207]. This algorithm computes EMD several times with additional noise, it allows us to efficiently analyze signals that have flat sections (i.e. consuming no electricity in our case).

3.1.2 IMF aggregation

By applying EMD to energy consumption signals we obtain a set of IMFs that precisely describe the devices consumption patterns at different frequency bands. Therefore, we can focus our analysis on the smaller time scales, ignoring the dominant patterns that prevent us from effectively analyzing raw signals.

However, comparing the IMFs obtained from different signals is also not trivial, because EMD is empirically uncovering IMFs from the data there is no guarantee that the two IMFs c_i^1 and c_i^2 obtained from two distinct signals S^1 and S^2 represent data at the same frequency domain. Directly comparing c_i^1 and c_i^2 is meaningless unless we confirm that they belong to the same frequency domain.

There are numerous techniques to retrieve IMF frequencies [108]. In this work we take advantage of the Generalized Zero Crossing (GZC) [107] because it is a simple and robust estimator of the instantaneous IMF frequency [108]. GZC is a direct

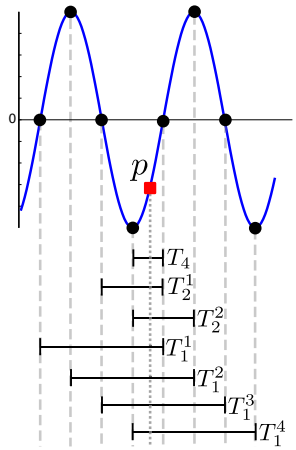


Figure D.4: Generalized Zero Crossing : the local mean period at the point p is computed from one quarter period T_4, two half periods T_2^x and four full periods T_1^y (where $x = 1, 2$, and, $y = 1, 2, 3, 4$).

estimation of IMF instantaneous frequency using critical points defined as the zero crossings and local extrema (round dots in Figure D.4). Formally, given a data point p, GZC measures the quarter (T_4), the two halves (T_2^x), and the four full periods (T_1^y), p belong to (see Figure D.4) and the instantaneous period is computed as :

$$T = \frac{1}{7}\{4T_4 + (2T_2^1 + 2T_2^2) + (T_1^1 + T_1^2 + T_1^3 + T_1^4)\}$$

Since all points p between two critical points have the same instantaneous period GZC is local down to a quarter period. Hereafter, we refer to the time scale of an IMF as the average of the instantaneous periods along the whole IMF. Because the time scale of each IMF depends on the original signal, we propose the following to efficiently compare IMFs from different signals. We cluster IMFs with respect to their time scales and partially reconstruct each signal by aggregating its IMFs from the same cluster. Then, we directly compare the partial signals of different devices.

The IMFs are clustered using four time scale ranges :

- The *high frequencies* are all the IMFs with a time scale lower than 20 minutes. These IMFs capture the noise.
- The *medium frequencies* are all the IMFs with a time scale between 20 minutes and 6 hours. These IMFs convey the detailed devices usage.
- The *low frequencies* are all the IMFs with a time scale between 6 hours and 6 days. These IMFs represent daily device patterns.
- The *residual data* is all data with a time scale higher than 6 days. This is mainly residual data obtained after applying EMD. Also, it highlights the main device trend.

These time scale ranges are chosen based on our experiments and goal. The 20-minute boundary relies on the sampling period of our dataset (5 minutes) and permits us to capture IMFs with really short periods. The 6-hour boundary allows us to analyze all patterns that have a period shorter than the usual office hours. The 6-day boundary allows us to capture daily patterns and weekday patterns.

Aggregating IMFs, within each time scale range, results in 4 partial signals representing different characteristics of the device's energy consumption (see *Partial Signals* in Figure D.3). We do a pairwise device trace comparison, calculating the correlation coefficient of their partial signals. In the example shown in Figure D.3, the correlation coefficient of the raw signals suggests that they are highly correlated (0.57). However, the comparison of the corresponding *partial signals* provides new insights ; the two devices are poorly correlated at high and medium frequencies (respectively -0.01 and -0.04) but highly correlated at low frequencies (0.79) meaning that these devices are not "intrinsically" correlated. They only share a similar daily pattern.

All the devices are compared pairwise at the four different time scale ranges. Consequently, we obtain four correlation matrices that convey device similarities at different time scales. Each line of these matrices (or column, since the matrices are symmetric) reveals the behavior of a device – its relationships with the other devices at a particular time scale. The matrices form the basis for tracking the behavior of devices and to search for misbehavior.

3.2 Search

Search aims at identifying misbehaving devices in an unsupervised manner. Device behavior is monitored via the correlation matrices presented in the previous section. Using numerous observations SBS computes a specific reference that exhibits the normal inter-device usage pattern. Then, SBS compares the computed reference with the current data and reports devices that deviate from their usual behavior.

3.2.1 Reference

We define four reference matrices, which capture normal device behavior at the four time scale ranges defined in Section 3.1.2. The references are computed as follows : (1) we retrieve the correlation matrices for n consecutive time bins. (2) For each pair of devices we compute the median correlation over the n time bins and obtain a matrix of the median device correlations.

Formally, for each time scale range the computed reference matrix for d devices and n time bins is :

$$R_{i,j} = \text{median}(C^1_{i,j}, ..., C^n_{i,j})$$

where i and j ranges in $[1, d]$.

Because anomalies are rare by definition, we assume the data used to construct the reference matrix is an accurate sample of the population ; it is unbiased and accurately captures the range of normal behavior. Abnormal correlation values, that could appear during model construction, are ignored by the median operator thanks to its robustness to outlier (50% breakdown point). However, if that assumption does not hold (more than 50% of the data is anomalous), our model will flag the opposite – labeling abnormal as normal and vice-versa. From close inspection of our data, we believe our primary assumption is sound.

3.2.2 Behavior change

We compare each device behavior, for all time bins, to the one provided by the reference matrix. Consider the correlation matrix C^t obtained from the data for time bin t ($1 \leq t \leq n$). Vector $C^t_{i,*}$ is the behavior of the ith device for this time bin. Its normal behavior is given by the corresponding vector in the reference matrix $R_{i,*}$. We measure the device behavior change at the time bin t with the following Minkowski weighted distance :

$$l^t_i = \left(\sum_{j=1}^{d} w_{ij} \left(C^t_{i,j} - R_{i,j} \right)^p \right)^{1/p}$$

where d is the number of devices and w_{ij} is :

$$w_{ij} = \frac{R_{i,j}}{\sum_{k=1}^{d} R_{i,k}}.$$

The weight w enables us to highlight the relationship changes between the device i and those highly correlated to it in the reference matrix. In other words, our definition of behavior change is mainly driven by the relationship among devices that are usually used in concert. We also set $p = 4$ in order to inhibit small differences between $C^t_{i,j}$ and $R_{i,j}$ but emphasize the important ones.

By monitoring this quantity over several time bins the abnormal device behaviors are easily identified as the outlier values. In order to identify these outlier values we implement a robust detector based on median absolute deviation (MAD), a dispersion measure commonly used in anomaly detection [109, 44]. It is a measure that robustly estimates the variability of the data by computing the median of the absolute deviations from the median of the data. Let $l_i = [l^1_i, ..., l^n_i]$ be a vector representing the behavior changes of device i over n time bins, then its MAD value is defined as :

$$\mathrm{MAD}_i = b\,\mathrm{median}(|l_i - \mathrm{median}(l_i)|)$$

where the constant b is usually set to 1.4826 for consistency with the usual parameter σ for Gaussian distributions. Consequently, we define anomalous behavior, for device i at time t, such that the following equation is satisfied :

$$l^t_i > \mathrm{median}(l_i) + \tau\,\mathrm{MAD}_i$$

Note, τ is a parameter that permits to make SBS more or less sensitive.

The final output of SBS is a list of alarms in the form (t, i) meaning that the device i has abnormal behavior at the time bin t. The priority of the alarms in this list is selected by the building administrator by tuning the parameter τ.

4 Data sets

We evaluate SBS using data collected from buildings in two different geographic locations. One is a new building on main campus of the University of Tokyo and the other is an older building at the University of California, Berkeley.

4.1 Engineering Building 2 - Todai

Engineering building 2, at the University of Tokyo (Todai), is a 12-story building completed in 2005 and is now hosting classrooms, laboratories, offices and server rooms. The electricity consumption of the lighting and HVAC systems of 231 rooms is monitored by 135 sensors. Rather than a centralized HVAC system, small, local HVAC systems are set up throughout the buidling. The HVAC systems are classified into two categories, EHP (Electrical Heat Pump) and GHP (Gas Heat Pump). The GHPs are the only devices that serve numerous rooms and multiple floors. The 5 GHPs in the dataset serve 154 rooms. The EHP and lighting systems serve only pairs of rooms and which are directly controlled by the occupants. In addition, the sensor metadata provides device-type and location information (room number), therefore, the electricity consumption of each pair of rooms is separately monitored.

The dataset contains 10 weeks of data starting from June 27, 2011 and ending on September 5, 2011. This period of time is particularly interesting for two reasons : 1) in this region, the summer is the most energy-demanding season and 2) the building manager actively works to curtail energy usage as much as possible due to the Tohoku earthquake and Fukushima nuclear accident.

Furthermore, this dataset is a valuable ground truth to evaluate the Strip and Bind portions of SBS. Since the light and HVAC of the rooms are directly controlled by the room's occupants, we expect SBS to uncover verifiable devices relationships.

4.2 Cory Hall - UC Berkeley

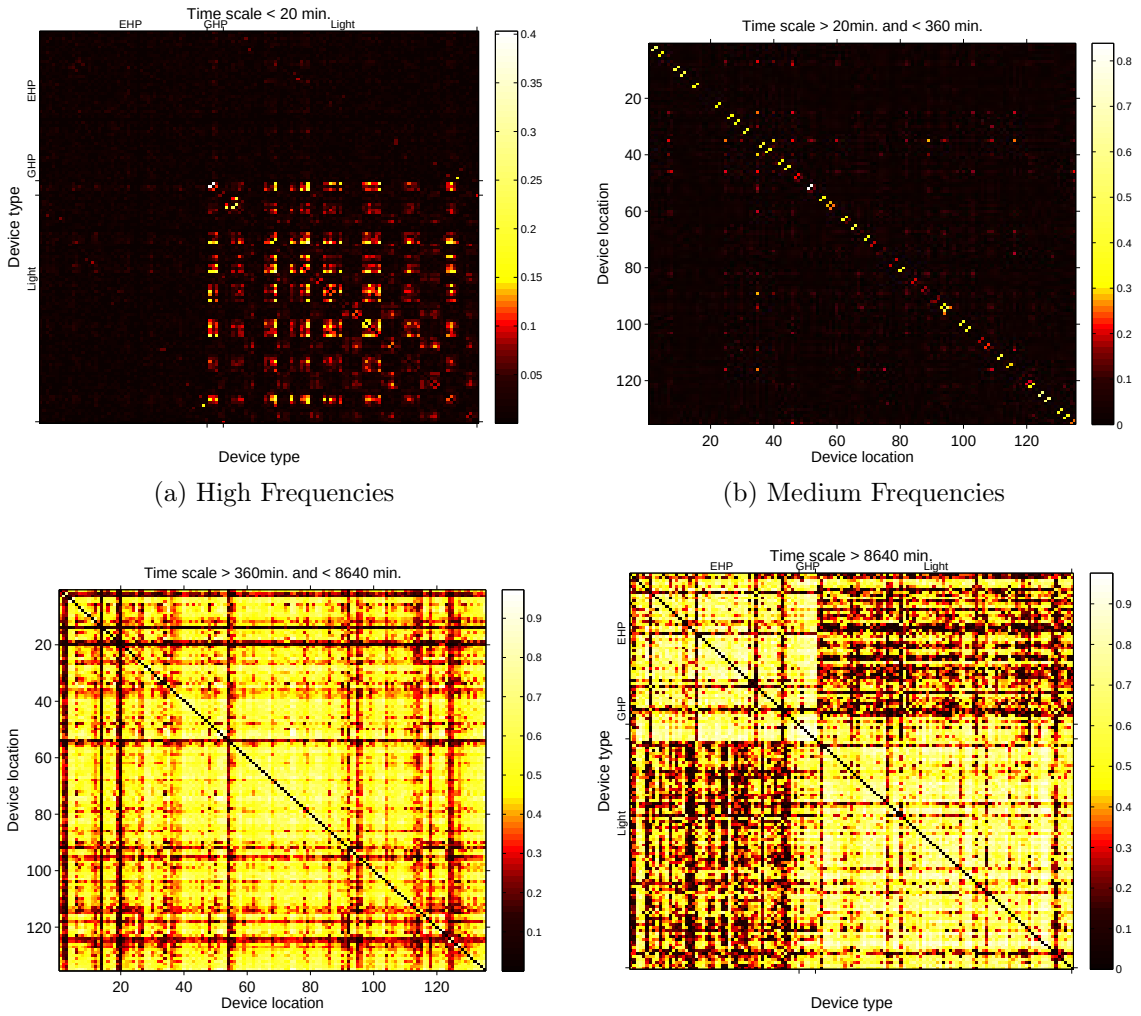
Cory Hall, at UC Berkeley, is a 5-story building hosting mainly classrooms, meeting rooms, laboratories and a datacenter. This building was completed in 1950, thus its infrastructure is significantly different from the Japanese one. The HVAC system in the building is centralized and serves several floors per unit. There is a separate unit for an internal fabricated laboratory, inside the building.

This dataset consists of 8 weeks of energy consumption traces measured by 70 sensors starting on April 5th, 2011. In contrast to the other dataset, a variety of devices are monitored, including, electric receptacles on certain floors, most of the HVAC components, power panels and whole-building consumption.

These two building infrastructures are fundamentally different. This enables us to evaluate the practical efficacy of the proposed, unsupervised method in two very different environments.

4.3 Data pre-processing

Data pre-processing is not generally required for the proposed approach. Nevertheless, we observe in a few exceptional cases that sensors reporting excessively high values (i.e. values higher than the device actual capacity) that greatly alter the performance of SBS by inducing a large bias in the computation of the correlation coefficient. Therefore, we remove values that are higher than the maximum capacity of the devices, from the raw data.



(a) High Frequencies

(b) Medium Frequencies

(c) Low Frequencies

(d) Residual data

Figure D.5: Reference matrices for the four time scale ranges (the diagonal $x = y$ is colored in black for better reading). The medium frequencies highlight devices that are located next to each other thus intrinsically related. The low frequencies contains the common daily pattern of the data. The residual data permits to visually identify devices of the similar type.

5 Experimental Results

In this section we evaluate SBS on our building traces. We demonstrate the benefits of striping the data by monitoring patterns captured at different time scales. Then, we thoroughly investigate the alarms reported by SBS.

5.1 Shortcomings

Because our analysis is done on historical data, some of the faults found by SBS could not be fully corroborated. In order to fully examine the effectiveness of our approach, we must run it in real time and physically check that the problem is

actually occurring. When a problem is detected in the historical trace, months after it has occurred, the current state of the building may no longer reflect what is in the traces. Some of the anomalies discussed in this section uncover interpretable patterns that are difficult to find in practice. For example, simultaneous heating and cooling is a known, recurring problem in buildings, but it is very hard to identify when it is occurring. Some of the anomalies we could not interpret might be interpretable by a building manager, however, we did not consult either building manager for this study. Therefore, the results of this study do not examine the true/false positive rate exhaustively.

The true/false negative rate is impractical to assess. It may be examined through synthetic stimulation of the building via the control system. However, getting co-operation from a building manager to hand over control of the building for experimentation in non-trivial. Therefore, we forgo a full true/false negative analysis in our evaluation.

Because of these challenges, the evaluation of SBS focuses on comparing the output with known fault signatures. We examine anomalies, in either building, where the anomaly is easily interpretable but difficult to find by the building manager. We forego a comparison of SBS with competing algorithms because related algorithms require detailed knowledge of the building, *a priori*. The advantage of SBS is that it requires no such information to provide immediate value.

5.2 Device behavior at different time scales

The Strip and Bind part of SBS is evaluated using the data from Eng. Bldg 2. This dataset is appropriate to measure SBS's performance, since lighting and HVAC systems serving the same room are usually used simultaneously. Consequently, we analyze this data using SBS and verify that the higher correlations at medium frequencies correspond to devices located in the same room.

The dataset is split into 10, one-week bins and each bin is processed by SBS. Using the 10 correlation matrices at each time scale range, SBS uncovers the four reference matrices depicted in Figure D.5.

High frequencies In this work the high frequencies correspond to the signals *noise*, therefore, we do not expect any useful information from the corresponding matrix (Figure D.5a). Indeed, the corresponding reference matrix does not provide any help to determine a device's relative location. Thus, we emphasize that high frequency data should be ignored for uncovering device relationships (in contrast to [80]). Interestingly, we find that the sensors monitoring the lights generate consistent noise.

Medium frequencies Our main focus is on the medium frequencies as it is designed to capture the intrinsic device relationships. Figure D.5b shows the correlation matrix at medium frequencies. It is significantly different from the one obtained with the raw signals (Figure D.1) : high correlation coefficients are concentrated along the matrix diagonal. Since devices serving the same or adjacent rooms are placed

nearby in the matrix it validates our hypothesis : *high correlation scores within the medium frequency band shows strong inter-device relationships.*

Considering this reference matrix as an adjacency matrix of a graph, in which the nodes are the devices, we identify the clusters of correlated devices using a community mining algorithm [32]. As expected we obtain mainly clusters of only two devices (light and HVAC serving the same room), but we also find clusters that are composed of more devices. For example a cluster contains 3 HVAC systems serving the three server rooms. Although these server rooms are located on different floors, SBS shows a strong correlation between these devices. Coincidentally, they are managed similarly. Interestingly, we also observe a couple of clusters that consist of independent devices serving adjacent rooms belonging to the same lab. The bigger cluster contains 33 devices that are 2 GHP devices and the corresponding lights. This correlation matrix and the corresponding clusters highlight the ability for SBS to identify such hidden inter-device usage relationships.

Low frequencies Low frequencies capture daily patterns, embedded in all the device traces. Figure D.5c depicts the corresponding reference matrix which is similar to the one of raw signal traces (Figure D.1) and it shows no particular structure.These partial signals are discarded as they do not help us in identifying inter-device usage patterns.

Residual data The residual data shows the weekly trend, which gives us no information about device relationships. But, surprisingly, by reordering the correlation matrix based on the type of the devices (Figure D.5d) we can visually identify two major clusters. The first cluster consists of HVAC devices (see EHP and GHP in Figure D.5d) and the second one contains only lights. An in-depth examination of the data reveals that long-term trends are inherent to the device types. For example, as the consumption of both the EHP and GHP devices is driven by the building occupancy and the outside temperature, these two types of devices follow the same trend. However, the use of light is independent from the outside temperature thus the lighting systems follow a common trend different from the EHP and GHP one.

We conduct the same experiments by splitting the dataset in 70 bins of 1 day long and observe analogous results at high and medium frequencies but not at lower frequencies. This is because the bins are too short to exhibit daily oscillations and the residual data captures only the daily trend.

5.3 Anomalies

We evaluate the *search* performance of SBS using the traces from the Eng. Bldg 2 and Cory Hall. Due to the lack of historical data, such as room schedule or reports of energy waste, the evaluation is non-trivial. Furthermore, getting ground truth data from a manual inspection of the hundreds traces of our data sets is impractical. The lack of ground truth data prevents us from producing a systematic analysis of the anomalies missed by SBS (i.e. false negatives rate). Nevertheless, we exhibit the relevance of the anomalies uncovered by SBS (i.e. high true positive rate and low false positive rate) by manually checking the output of SBS.

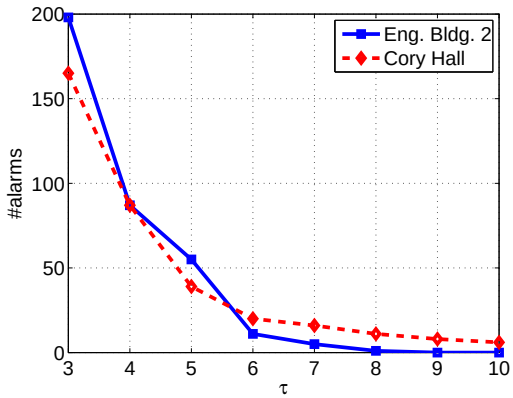


FIGURE D.6: Number of reported alarms for various threshold value ($\tau = [3, 10]$).

	High	Low	Punc.	Missing	Other
Eng. Bldg 2	9 (5)	6 (5)	1 (1)	36 (1)	3 (3)
Cory Hall	25 (7)	7 (3)	4 (4)	0 (0)	3 (3)

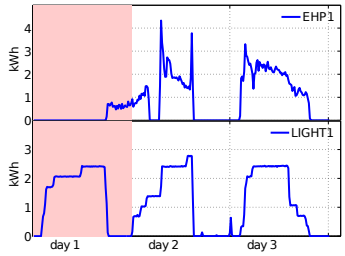
TABLE D.1: Classification of the alarms reported by SBS for both dataset (and the number of corresponding anomalies).

Anomaly classification To validate SBS results we manually inspect the anomalies detected by the algorithm. For each reported alarm (t, i) we investigate the device trace i and the devices correlated to it to determine the reason for the alarm. Specifically, we retrieve the major relationship change that causes the alarm (i.e. $\max(|w_j(C^t_{i,j} - R_{i,j})|)$, see Section 3.2) and examine the metadata associated to the corresponding device. This investigation allows us to classify the alarms into five groups :

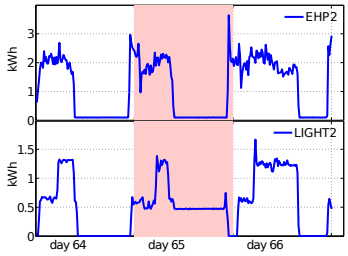
- *High power usage* : alarms corresponding to electricity waste.
- *Low power usage* : alarms representing the abnormally low electricity consumption of a device.
- *Punctual abnormal usage* : alarms standing for short term (less than 2.5 hours) raise or drop of the electricity consumption.
- *Missing data* : alarms raised due to a sensor failure.
- *Other* : alarms whose root cause is unclear.

Experimental setup For each experiment, the data is split in time bins of one day, starting from 09 :00 a.m. – which is approximately the office's opening time. We avoid having bins start at midnight since numerous anomalies appear at night and they are better highlighted if they are not spanning two time bins. Only the data at medium frequencies are analyzed, the other frequency bands are ignored, and the reference matrix is computed from all time bins.

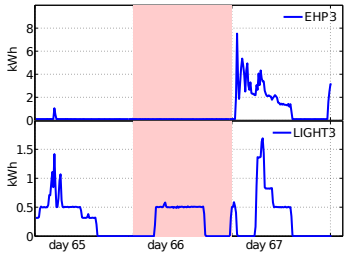
The threshold τ tunes the sensitivity of SBS, hence, the number of reported alarms. Furthermore, by plotting the number of alarms against the value of τ for



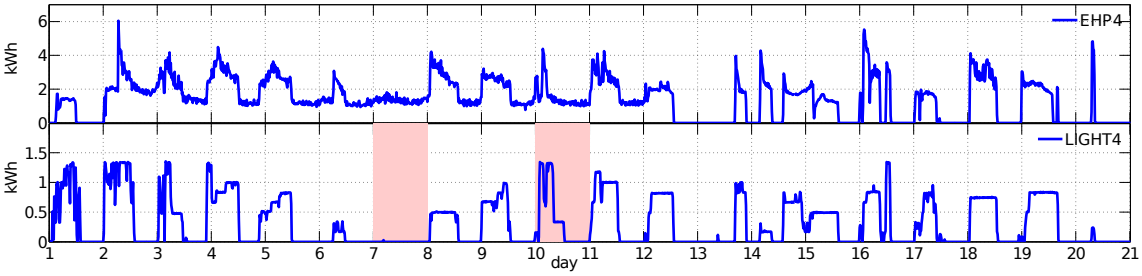
(a) High power usage where the HVAC (EHP) is turned on at night



(b) High power usage where the light is left on at night



(c) Low power usage where the HVAC (EHP) is not used during office hours



(d) Long term high power usage partially detected

Figure D.7: Example of alarms (red rectangles) reported by SBS on the Eng. Bldg 2 dataset

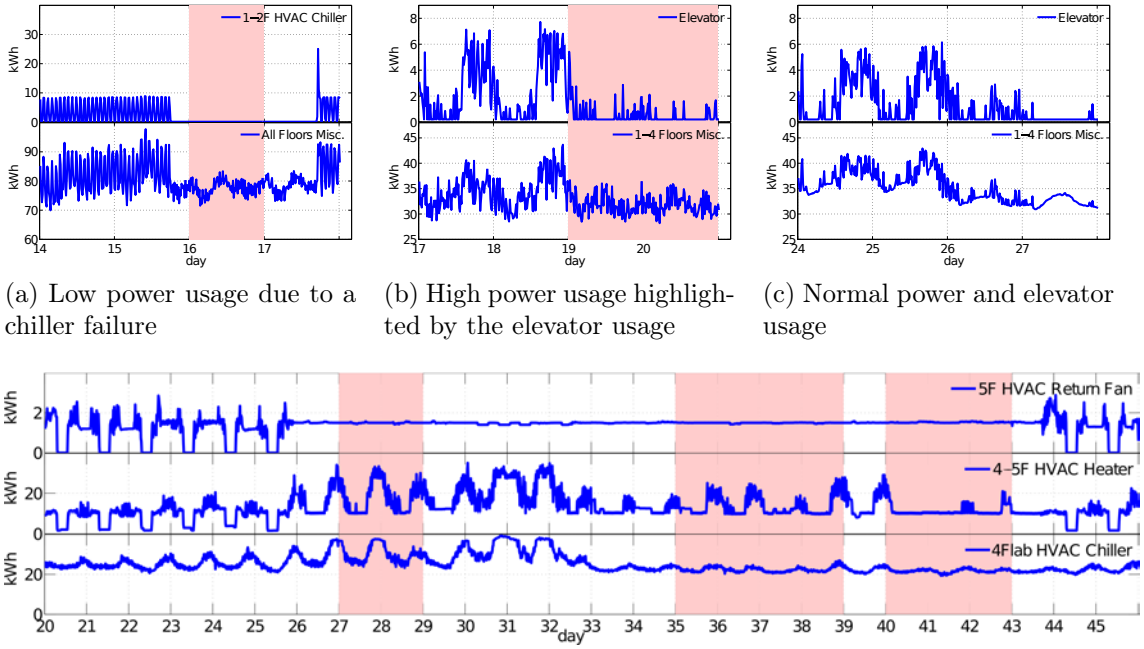
both datasets (Figure D.6) we observe an elbow in the graph around $\tau = 5$. With thresholds lower than this pivot value ($\tau < 5$), the number of alarms significantly increases, causing less important anomalies to be reported. For higher values ($\tau > 5$), the number of alarms is slowly decreasing, providing more conservative results that consist of the most important anomalies. This pivot value provides a good trade off for either data set.

Table D.1 classifies the alarms reported by SBS on both datasets. Anomalies spanning several time bins (or involving several devices) may raise several alarms. We display these in Table D.1 as numbers in brackets – the number of anomalies corresponding to the reported alarms.

5.3.1 Engineering Building 2

SBS reported 55 alarms over the 10 weeks of the Eng. Bldg 2 dataset. However, 36 alarms are set aside because of sensor errors; one GHP has missing data for the first 18 days. Since this device is highly correlated to the GHP in the reference matrix, their relationship is broken for the 18 first bins and for each bin one alarm per device is raised.

In spite of the post-Fukushima measures to reduce Eng. Bldg 2's energy consumption, SBS reported 9 alarms corresponding to high power usage (Table D.1). Figure D.7a depicts the electricity consumption of the light and EHP in the same room where two alarms are raised. Because the EHP was not used during daytime (but is turned on at night, when the light is turned off) the relationship between the



(a) Low power usage due to a chiller failure

(b) High power usage highlighted by the elevator usage

(c) Normal power and elevator usage

(d) Long term high power usage due to competing heating and cooling

FIGURE D.8: Example of alarms (red rectangles) reported by SBS on the Cory Hall dataset

two devices is "broken" and an alarm is raised for each device. Figure D.7b shows another example of energy waste. The light is on at night and the EHP is off. The top-priority anomaly reported by SBS is caused by the 10 days long constant use of an EHP (Figure D.7d) and this waste of electricity accounts for 165 kWh. SBS partially reports this anomaly but lower values of τ permits us to identify most of it.

We observed 6 alarms corresponding to abnormally low power use. Upon further inspection we notice that it corresponds to energy saving initiatives from the occupants – likely due to electricity concerns in Japan. This behavior is displayed in Figure D.7c. The room is occupied at the usual office hours (indicated by light usage) but the EHP is not on in order to save electricity.

5.3.2 Cory Hall

SBS reported 39 alarms for the Cory Hall dataset (Table D.1). 7 are classified as low power usage, however, our inspection revealed that the root causes are different than for the Eng. Bldg 2 dataset. We observe that the low power usage usually corresponds to device failures or misconfiguration. For example, Figure D.8a depicts the electricity consumption of the 2nd floor chiller and a power riser that comprises the consumption of multiple systems, including the chiller. As the chiller suddenly stops working, the correlation between both measurements is significantly altered and an alarm for each device is raised.

SBS also reports 25 alarms corresponding to high power usage. One of the identified anomalies is particularly interesting. We indirectly observe abnormal usage of a device from the power consumption of the elevator and a power panel for equipment

from the 1st to the 4th floor. Figure D.8b and D.8c show the electricity consumption for both devices. SBS uncovers the correlation between the these two signals, as the amount of electricity going through the panel fluctuates along with the elevator power consumption (Figure D.8c). In fact, the elevator is a good indicator of the building's occupancy. Anomalous energy-consumption is identified during a weekend as the consumption measured at the panel is independently fluctuating from the elevator usage. These fluctuations are caused by a device that is not directly monitored. Therefore, we could not identify the root cause more precisely. Nevertheless, the alarm is worthwhile for building operators to start investigating.

The most important anomaly identified in Cory Hall is shown in Figure D.8d. This anomaly corresponds to the malfunctioning of the HVAC heater serving the 4th and 5th floors. The heater is constantly working for 18 consecutive days, regardless of the underlying occupant activity. Moreover, in order to maintain appropriate temperature this also results in an increase of the 4th floor HVAC chiller power consumption and several fans, such as the one depicted in Figure D.8d. This situation is indicative of simultaneous heating and cooling – whereby heating and cooling systems are competing – and it is a well-know problem in building management that leads to significant energy waste. For this example, the electricity waste is estimated around 2500 kWh for the heater. Nevertheless, as the anomaly spans over 18 days, it is hidden in the building's overall consumption, thus, it is difficult to detect by building administrators without SBS.

6 Related work

The research community has addressed the detection of abnormal energy-consumption in buildings in numerous ways [121, 122].

The rule-based techniques rely on a priori knowledge, they assert the sustainability of a system by identifying a set of undesired behaviors. Using a hierarchical set of rules, Schein et al. propose a method to diagnose HVAC systems [187]. In comparison, state machine models take advantage of historical training data and domain knowledge to learn the states and transitions of a system. The transitions are based on measured stimuli identified through a domain expertise. State machines can model the operation of HVAC systems [166] and permit to predict or detect the abnormal behavior of HVAC's components [29]. However, the deployment of these methods require expert knowledge and are mostly applied to HVAC systems.

In [189], the authors propose a simple unsupervised approach to monitor the average and peak daily consumption of a building and uncover outlier, nevertheless, the misbehaving devices are left unidentified.

Using regression analysis and weather variables the devices energy-consumption is predicted and abnormal usage is highlighted. The authors of [38] use kernel regression to forecast device consumption and devices that behave differently from the predictions are reported as anomalous. Regression models are also used with performances indices to monitor the HVAC's components and identify inefficiencies [232]. The implementation of these approaches in real situations is difficult, since it requires a training dataset and non-trivial parameter tuning.

Similar to our approach, previous studies identify abnormal energy-consumption

using frequency analysis and unsupervised anomaly detection methods. The device's consumption is decomposed using Fourier transform and outlier values are detected using clustering techniques [28, 226, 45]. However, these methods assume a constant periodicity in the data and this causes many false positives in alarm reporting. We do not make any assumption about the device usage schedule. We only observe and model device relationships. We take advantage of a recent frequency analysis technique that enables us uncover the inter-device relationships [80]. The identified anomalies correspond to devices that deviate from their normal relationship to other devices.

Reducing a building's energy consumption has also received a lot of attention from the research community. The most promising techniques are based on occupancy model predictions as they ensure that empty rooms are not over conditioned needlessly. Room occupancy is usually monitored through sensor networks [19, 74] or the computer network traffic [124]. These approaches are highly effective for buildings that have rarely-occupied rooms (e.g. conference room) and studies show that such approaches can achieve up to 42% annual energy saving. SBS is fundamentally different from these approaches. SBS identifies the abnormal usage of any devices rather than optimizing the normal usage of specific devices. Nevertheless, the two approaches are complementary and energy-efficient buildings should take advantage of the synergy between them.

7 Discussion

SBS is a practical method for mining device traces, uncovering hidden relationships and abnormal behavior. In this paper, we validate the efficacy of SBS using the sensor metadata (i.e. device types and location), however, these tags are not needed by SBS to uncover devices relationships. Furthermore, SBS requires no prior knowledge about the building and deploying our tool to other buildings requires no human intervention – neither extra sensors nor a training dataset is needed.

SBS is a best effort approach that takes advantage of all the existing building sensors. For example, our experiments revealed that SBS indirectly uncovers building occupancy through device use (e.g. the elevator in the Building 2). The proposed method would benefit from existing sensors that monitor room occupancy as well (e.g. those deployed in [19, 74]). Savings opportunities are also observable with a minimum of 2 monitored devices and building energy consumption can be better understood after using SBS.

SBS constructs a model for normal inter-device behavior by looking at the usage patterns over time, thus, we run the risk that a device that constantly misbehaves is labeled as normal. Nevertheless, building operators are able to quickly identify such perpetual anomalies by validating the clusters of correlated devices uncovered by SBS. The inspection of these clusters is effortless compare to the investigation of the numerous raw traces. Although this kind of scenario is possible it was not observed in our experiments.

In this paper, we analyze only the data at medium frequencies, however, we observe that data at the high frequencies and residual data (Figure D.5) also permits us to determine the device type. This information is valuable to automatically retrieve

and validate device labels – a major challenge in building metadata management.

This paper aims to establish a methodology to identify abnormalities in device power traces and inter-device usage patterns. In addition, we are planning to apply this method to online detection using, for example, a sliding window to compute an adaptive reference matrix that evolve in time. However, designing such system raises new challenges that are left for future work.

8 Conclusions

The goal of this article is to assist building administrators in identifying misbehaving devices in large building sensor deployments. We proposed an unsupervised method to systematically detect abnormal energy consumption in buildings : the Strip, Bind, and Search (SBS) method. SBS uncovers inter-device usage patterns by striping dominant trends off the devices energy-consumption trace. Then, it monitors device usage and reports devices that deviate from the norm. Our main contribution is to develop an unsupervised technique to uncover the true inter-device relationships that are hidden by noise and dominant trends inherent to the sensor data. SBS is used on two sets of traces captured from two buildings with fundamentally different infrastructures. The abnormal consumption identified in these two buildings are mainly energy waste. The most important one is an instance of a competing heater and cooler that caused the heater to waste around 2500 kWh.

Acknowledgments

The authors thank Hideya Ochiai for providing the data from the University of Tokyo. This research was partially supported by the JSPS fellowship program and the CNRS/JSPS Joint Research Project. This work is also supported in part by the National Science Foundation under grants CPS-0932209 and CPS-0931843.

Annexe E

Graph Empirical Mode Decomposition

Nicolas Tremblay, Pierre Borgnat, Patrick Flandrin

CNRS, École Normale Supérieure de Lyon

Published in the proceedings of the EUSIPCO'14 conference
September 1–5, 2014, Lisbon, Portugal.

An extension of Empirical Mode Decomposition (EMD) is defined for graph signals. EMD is an algorithm that decomposes a signal in an addition of modes, in a local and data-driven manner. The proposed Graph EMD (GEMD) for graph signals is based on careful considerations on key points of EMD : defining the extrema, interpolation procedure, and the sifting process stopping criterion. Examples of GEMD are shown on the 2D grid and on two examples of sensor networks. Finally the effect of the graph's connectivity on the algorithm's performance is discussed.

Contents

1 Introduction **176**
2 Algorithm for Graph EMD **176**
2.1 Classical Empirical Mode Decomposition 176
2.2 From CEMD to Graph EMD 177
3 Examples and Discussion **179**
3.1 Application and discussion on the 2D grid 179
3.2 Two examples of sensor networks and discussion 180
3.3 Another definition of local extrema 182
4 Conclusion **184**

1 Introduction

Graphs are a useful coding or representation of relations in data for many applications, e.g., neural, sensor, energy, social or biological networks. A *graph signal* is a signal defined on the nodes of a graph, the structure of this graph being either known *a priori* or inferred from proximity or similarity measures between nodes. Fig. E.1 shows two examples of such signals, in the context of sensor networks, where nodes are sensors spread out in space. Recently, there has been a substantial effort to adapt classical signal processing tools to graph signals [191, 183] such as the graph wavelet transform [96, 137, 53, 156], lifting schemes [155], the windowed Fourier transform [195] or interpolation [154]. We introduce the Graph Empirical Mode Decomposition, an adaptation to graph signals of the now widely used Empirical Mode Decomposition (EMD) [106, 142]. EMD is a data-driven algorithm that aims at locally separating fast from slow oscillations in a signal. As EMD is local and adaptive, it is especially useful when the components of the signal one wants to separate are nonstationary or have overlapping spectra, hence when a simple filtering in the Fourier space fails. For examples and illustrations, we focus on sensor networks, but the method is relevant for any graph signal. In Sec. 2, the Graph EMD (GEMD) is introduced after recalling the classical EMD (CEMD). In Sec. 3, GEMD is applied to signals on the 2D grid, and on the two examples of Fig. E.1. We conclude in Sec. 4.

2 Algorithm for Graph EMD

2.1 Classical Empirical Mode Decomposition

Let us recall classical EMD (CEMD). Given a signal $x(t)$, it separates a local "low frequency" component $m_1(t)$ –the trend– from an Intrinsic Mode Function (IMF) $d_1(t)$ which is a local "high frequency" mode having the same number of extrema and zero crossings, and roughly symmetric with respect to zero. By applying the same decomposition to $m_1(t)$ we obtain $m_1(t) = m_2(t) + d_2(t)$, and, recursively :

$$x(t) = m_K(t) + \sum_{k=1}^{K} d_k(t). \tag{E.1}$$

The signal is decomposed in IMFs until they are all extracted.

Given $m_i(t)$, this separation of the slow oscillating trend $m_{i+1}(t)$ from the fast oscillating IMF $d_{i+1}(t)$, is done in the EMD algorithm by using the so-called *sifting process* [106] :

1. $s = m_i$. While $s(t)$ does not meet the sifting process stopping criterion, repeat steps 2 to 5 :
2. Detect the local extrema of $s(t)$.
3. Interpolate the minima (resp. extrema) to obtain some envelope $e_{min}(t)$ (resp. $e_{max}(t)$).
4. Compute the mean (local trend) $\mu(t) = \frac{e_{min}(t)+e_{max}(t)}{2}$.
5. Subtract it from the signal : $s(t) \leftarrow s(t) - \mu(t)$.

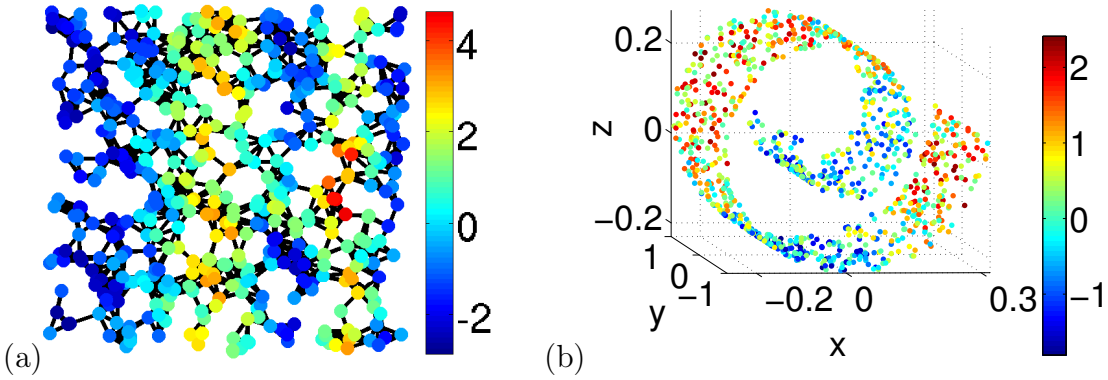


FIGURE E.1: Examples of graph signals in sensor networks, as detailed later on in Sec. 3.2. The values of the graph signals on the nodes is color-coded (as per the colorbars).

6. Set $d_{i+1}(t) = s(t)$ and $m_{i+1}(t) = m_i(t) - s(t)$.

The most conservative stopping criterion is that the loop stops as soon as $s(t)$ is an IMF. This is usually too strong a constraint and it is relaxed to a stopping criterion yielding approximate IMFs [176].

2.2 From CEMD to Graph EMD

Before discussing the elements defining EMD on graphs (extrema, interpolation and stopping criteria), let us study how one creates a graph for data when it is not known beforehand.

2.2.1 Graph creation

We place ourselves in the context of sensor networks, where the signal has a value at each sensor whose locations in space are known. Let V be the set of the sensors, used as nodes for the graph. Among the options to define edges in the graph supporting the signal, we explore two :

1. a weighted graph parametrized by δ : only pairs of sensors (i, j) at a distance $d_{i,j}$ shorter than δ are connected by an edge, with weight $w_{i,j} = \exp(-d_{i,j}^2/2\delta^2)$.
2. a binary graph parametrized by k : each node is connected to its k nearest neighbors (k-NN).

These procedures do not necessarily build connected graphs (typically when δ or k are too small). To avoid interpolation problems, choose a connected component and add the shortest link connecting it to another component – the component grows larger – and repeat this until the graph is connected.

In other contexts, the graph could be known beforehand, or obtained, e.g., by using statistical similarities as distances. Anyway, we end up with a graph $\mathcal{G} = (V, E)$, where E is the set of edges connecting nodes. Let us note A its adjacency matrix and D the diagonal matrix of degrees.

2.2.2 Definition of local extrema

For a signal x defined on V, a node i is a local maximum (resp. minimum) if, for all its neighbours k in $\mathcal{G}$, $x(i) > x(k)$ (resp. $x(i) < x(k)$). Note that other notions of extremum could be introduced : for instance extremum along one direction only, as it is done for images in "Pseudo-2D" EMD where extrema are along lines or columns only [227]. We explore another definition of extrema in Sec. 3.3.

2.2.3 Interpolation procedure

There are several ways to interpolate a graph signal. There are for instance global procedures, like the method discussed in [154], where the authors minimize the highest graph Fourier frequency mode necessary to recover the signal on the known nodes. Since this method is based on global Fourier modes, it would not be appropriate to extract modes having some nonstationnarity within the graph, e.g. a chirp. The highest frequency retained would be globally the highest one, and it contradicts the locality of EMD : at local places in the graph where the modes have lower frequency, this interpolation procedure would never extract the mode.

Instead, we rely on interpolation methods that are local, for instance formulated through a discrete partial differential equation on the graph. Inspired by Grady et al. [93], interpolation is recast as a Dirichlet problem on the graph. Consider the Laplacian $L = D - A$ of graph $\mathcal{G}$, the signal s on nodes V to be interpolated, and B (resp. U) the set of nodes where the signal is known (resp. unknown). Solving the Dirichlet problem comes down to finding s that minimizes $s^\top Ls$ under the constraint $s(b) = s_B(b)$ the known values for $b \in B$. By re-ordering the nodes, one may write $s^\top = [s_B^\top s_U^\top]$ and $L = \begin{bmatrix} L_B & R \\ R^\top & L_U \end{bmatrix}$. Solving the Dirichlet problem boils down to solving the system of linear equations : $L_U s_U = -R^\top s_B$.

2.2.4 Choice of stopping criteria

With the previous elements, the sifting process is easily modified and we propose a stopping criterion for this sifting process from an energy criterion : stop the loop 2-5 as soon as the energy of the mean $\mu(t)$ (computed at step 4) is lower than the energy of the signal $m_i(t)$ analysed divided by 1000. In all the experiments we have made, this criterion converges. This criteria is reminiscent of choices in CEMD, see [106, 176].

2.2.5 The GEMD algorithm

Akin to the CEMD, we define the GEMD from its algorithm. Given a set of sensors V, a set of measures $\{x_i\}_{i\in V}$, and K, the number of modes to be extracted, the algorithm reads :

1. Create the adjacency matrix A for the graph $\mathcal{G}$, here by considering the relative spatial positions (as in 2.2.1).
2. Set $m_0 = x$. Iterating on i, extract from m_i the fast mode d_{i+1} and slow trend m_{i+1} following the sifting process of Sec. 2.1 (where t stands now for

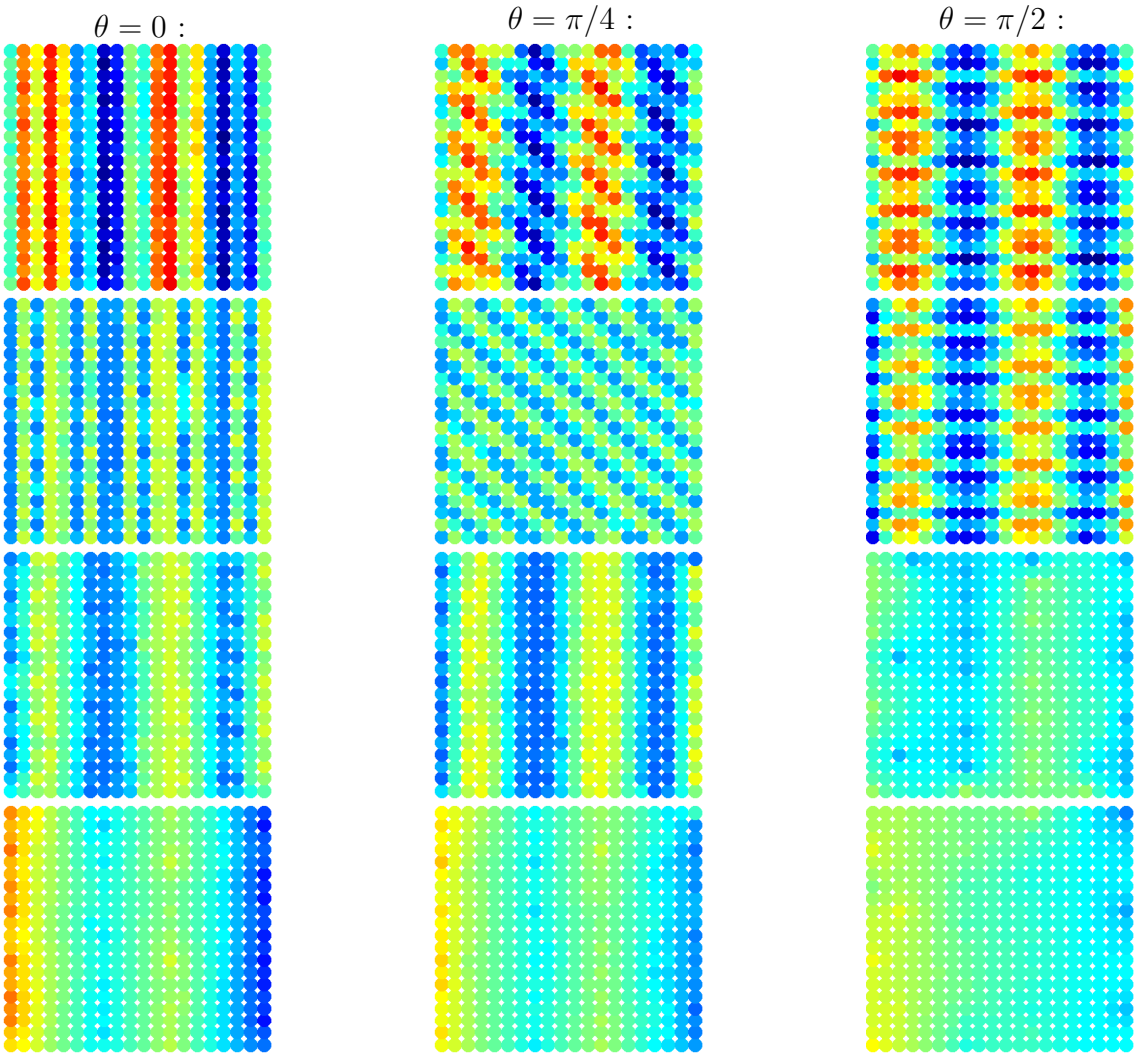


FIGURE E.2: The left (resp. center, right) column represents the result of the GEMD for an angle $\theta = 0$ (resp. $\pi/4$, $\pi/2$) between two sine waves. The first row is the original signal, the second (resp. third) row the first (resp. second) IMF, and the last row is the residue.

the indices of nodes) using the extrema, interpolation procedure and stopping criterion described above.

3. Stop and obtain $x(t) = m_K(t) + \sum_{k=1}^{K} d_k(t)$.

Note that we do not discuss here the number of modes to be extracted K, it is fixed a priori.

3 Examples and Discussion

3.1 Application and discussion on the 2D grid

Consider the case of $N = 400$ sensors distributed on the 2D grid 20×20. Instead of discussing it as a regular image, we adopt the point-of-view of graphs. Let us

consider, as an example, a superposition of sine waves separated by an angle θ ; the signal is the sum of three components :

- a horizontal sine wave of amplitude 2 and frequency 2.
- a sine wave of amplitude 1 and frequency 7, that propagates with an angle θ with the horizontal.
- a uniform noise of amplitude 0.5.

Fig. E.2 shows the results of the GEMD for $\theta = 0, \pi/4$ and $\pi/2$. For the first two cases, the two sines waves are separated as expected (high frequency wave in the first IMF). For orthogonal sine waves ($\theta = \pi/2$), the GEMD does not separate them and the explanation touches the very foundations of EMD : the definition of extrema in 2.2.2. Fig. E.3 displays the first steps of the algorithm. For $\theta = 0$ and $\pi/4$, there are enough extrema to force the envelopes (and thus the mean μ) to follow the low frequency component thereby enabling the separation of components ; whereas for $\theta = \pi/2$, there are not enough extrema and they have approximately the same value : the envelopes are flat, μ has very low energy and the first IMF will contain the whole signal, for no separation. This issue is that this signal is a valid IMF, like it is for EMD in 2D [164], and one should turn to "Pseudo-2D" EMD [227] with another definition of extrema to change that. In fact, the definition of extrema (combined with interpolation and stopping criterion) defines *a posteriori* what is an IMF. Depending on the application, one may change this to force the separation of a component. This issue is discussed in Sec. 3.3.

3.2 Two examples of sensor networks and discussion

3.2.1 A sensor network in 2D space

Consider a network of 512 sensors uniformly distributed on the 1×1 square. We create a weighted graph from their 2D space positions following 2.2.1 with $\delta = 0.075$. On this graph, we create a signal as the sum of 4 components :

- a sine wave of amplitude 1 and frequency 7, propagating with an angle $\theta = \pi/4$ with the horizontal (Fig. E.4a).
- a horizontal linear chirp of amplitude 2 (Fig. E.4d).
- a null signal except for a localized set of nodes of amplitude 2 (Fig. E.4g).
- a uniform noise of amplitude 0.5.

The total signal is plotted in Fig. E.1a. Results are plotted in the center column of Fig. E.4. The first IMF recovers the high frequency sine wave component. The linear chirp is partly in the second IMF and in the residue. The localized signal ends up in the residue. The right column of Fig. E.4 shows that a filtering in the graph Fourier space (as defined using the Laplacian [96]) would have failed because of overlap in the Fourier spectra.

3.2.2 A sensor network in 3D space

Consider a network of $N = 1024$ sensors distributed on a "swiss roll" manifold in 3D space [205] as shown in Fig. E.1b. The 3D positions (X, Y, Z) of the sensors on this manifold are computed in 3 steps : i) create U_1 and U_2, two uniformly random vectors between 0 and 1 of N points ; ii) the 3D coordinate vectors are obtained by

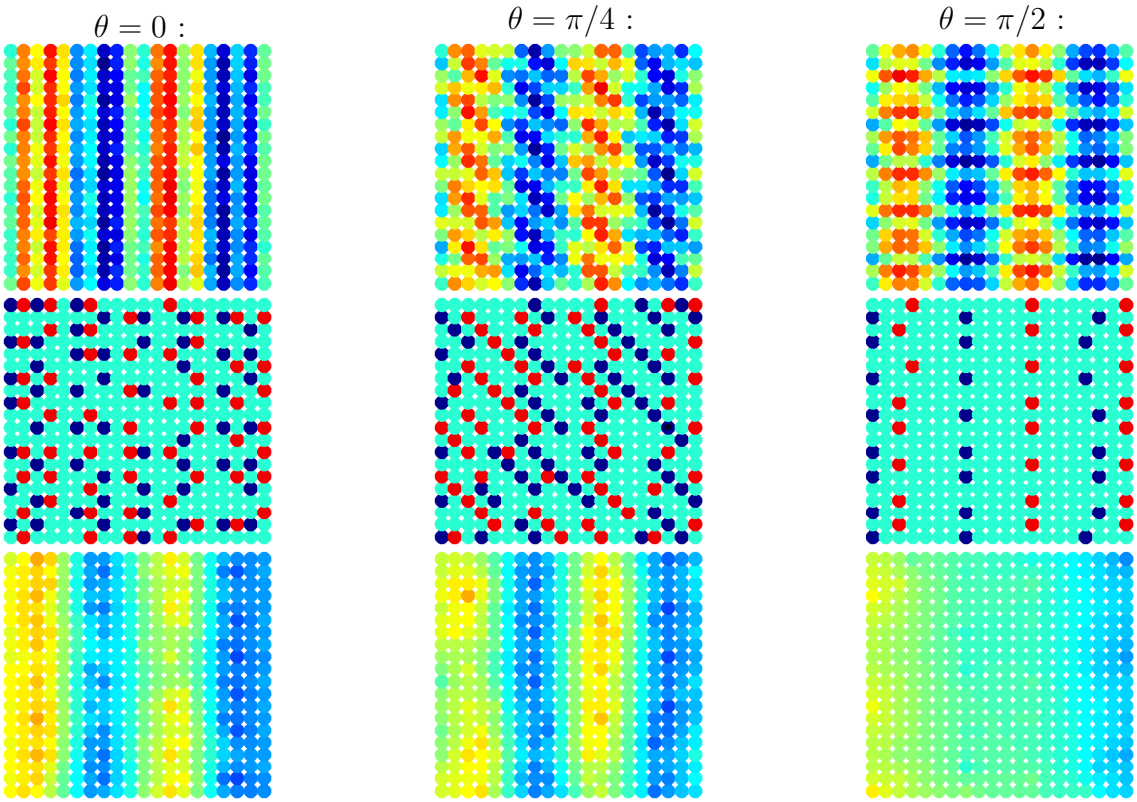


FIGURE E.3: The first steps of the GEMD on the 2D grid for the three angles. The first line is the original signals. The second row shows the extrema (minima in blue and maxima in red). The last row shows the mean μ of the two envelopes interpolated from these extrema.

setting $S_1 = \pi\sqrt{(b^2 - a^2)U_1 + a^2}$, and

$$X = S_1 \cos S_1;\ Y = \pi^2(b^2 - a^2)U_2/2;\ Z = S_1 \sin S_1.$$

Parameter $a\pi$ (resp. $b\pi$) is the starting (resp. ending) angle of the swiss roll. Here, $a = 1$ and $b = 4$. Y is chosen such that the length of the manifold in the Y direction is equal to the total length of the manifold if unrolled : this is to ensure a uniform distribution of sensors ; iii) the swiss roll is finally centered and rescaled to fit in the cube of side length 2.

The corresponding k-NN binary graph is created following section 2.2.1 with $k = 14$. On this graph, create a signal as the sum of 3 components :

- a sine wave in the 3D space : amplitude 1 and frequency 7, that propagates with an angle $\theta = \pi/4$ in the (y,z) plane (Fig.E.5c),
- a linear chirp along the manifold of amplitude 1 (Fig.E.5d),
- a uniform noise of amplitude 0.5.

The total signal is plotted in Fig.E.5a and Fig.E.5b for two different view points. The rest of Fig. E.5 shows the results of the GEMD : the first (resp. second) IMF in (e) (resp. (f)) recovers the 3D sine wave (c) (resp. the chirp on the manifold (d)). Here again, a simple filtering in the Fourier space would not have separated both signals (see Fig. E.5g and h).

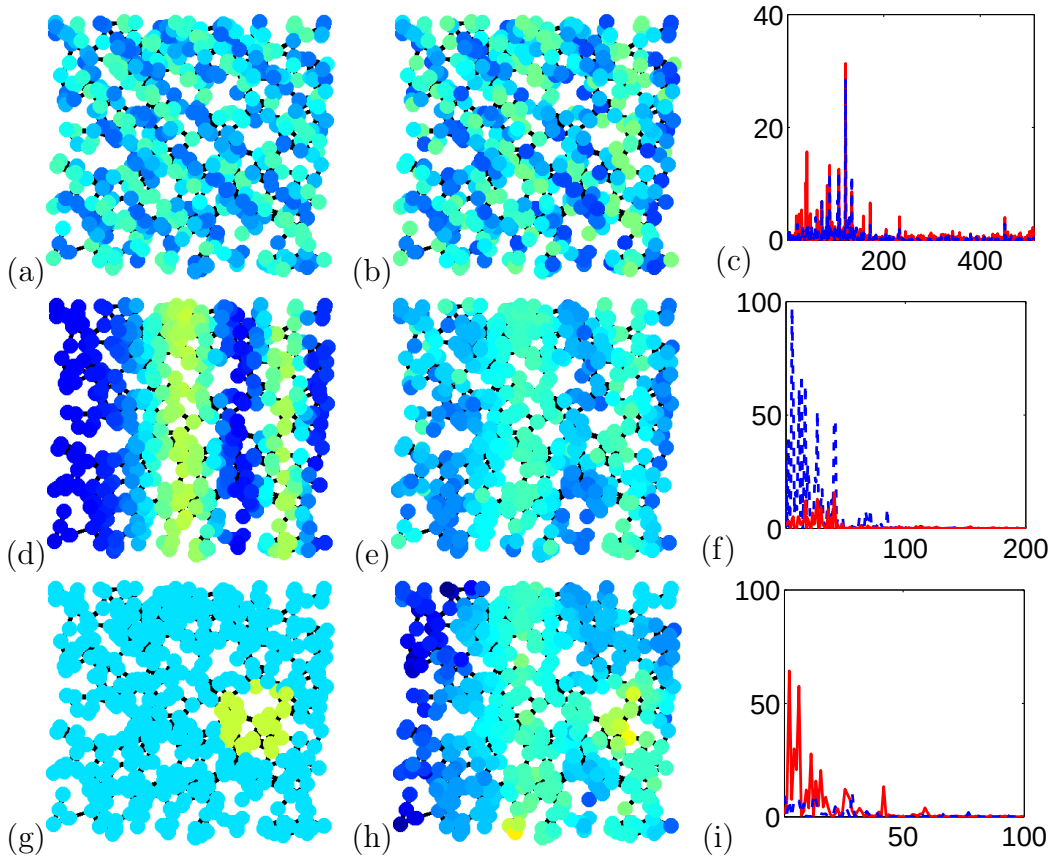


Figure E.4: Sensor network of 512 uniformly distributed nodes. Left column : the three components of the original signal. Middle column : the first two IMFs and the residue uncovered by the GEMD. The colormap is the same as in Fig. E.1a. Right column : theoretical (resp. uncovered) signals in blue (resp. red) in the graph Fourier domain.

Let us now investigate the impact of the construction of the graph from 2.2.1 on the power of recovery of the original components by the GEMD. The recovery rate of the 3D sine wave (resp. the chirp) is measured in terms of its correlation distance with the first (resp. second) IMF. Both methods detailed in section 2.2.1 are investigated and results are plotted in Fig.E.6. They present a similar behavior : when the connectivity is too low, the method is not sensible to slowly varying signals and recovery fails ; when the connectivity is too high, there are too few maxima and the whole signal ends up in the first IMF : recovery fails ; there exists an optimal connectivity for which both signals are reasonably uncovered. However, the sensitivity is not too high for k ; if k is (roughly) between 10 and 20, the recovery appears to be correct.

3.3 Another definition of local extrema

Suppose a different notion of extremum : a node is a local maximum (resp. minimum) if its value is higher (resp. lower) than a portion of its neighbors, like it would be for maximum along lines or columns on a grid. Here, we only explore this definition for half of the neighbors. In Fig. E.7 we compare results obtained with

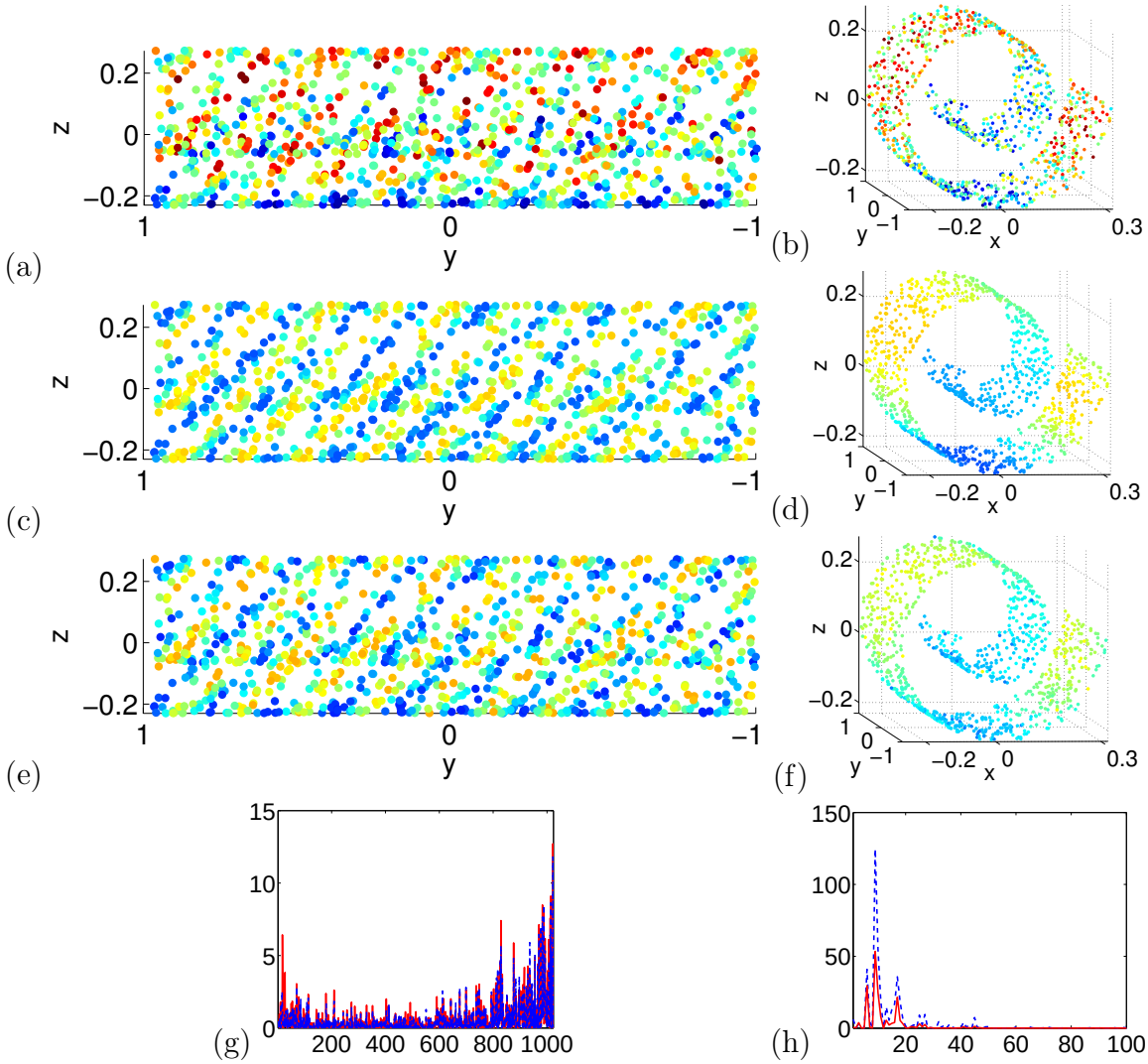


FIGURE E.5: GEMD results on a swiss roll manifold. (a) and (b) are two different views of the original signal composed of a sum of a 3D sine wave (c) and a linear chirp on the manifold (d). (e) and (f) are the first two IMFs. The colormap is the same as in Fig. E.1b. (g) and (h) compare the original (dashed blue) and recovered (red) components in the graph Fourier domain.

this definition (two right columns) with results previously obtained with Sec. 2.2.2's definition (two left columns) on the 2D grid example with $\theta = \pi/2$. We see how this new definition of extrema increase the number of extrema, thereby enabling the extraction of the fast oscillating mode (first IMF) but pushing the slow oscillating mode into the residue. This observation suggests to look for more elaborate notions of extrema that would take into account the topology of the graph.

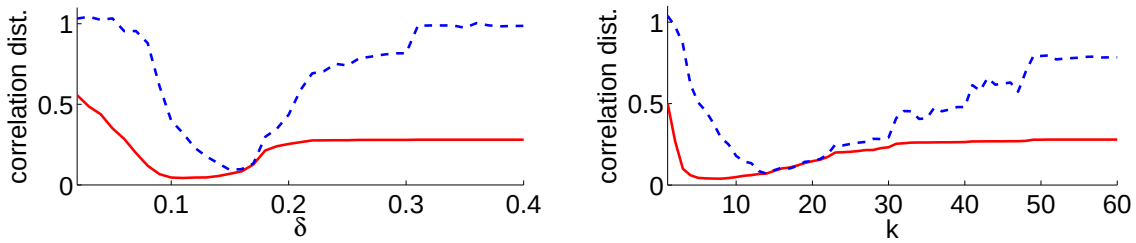


Figure E.6: For the swiss roll example, power of recovery of the 3D sine wave (red) and the chirp on the manifold (dashed blue) vs. the connectivity of the graph.

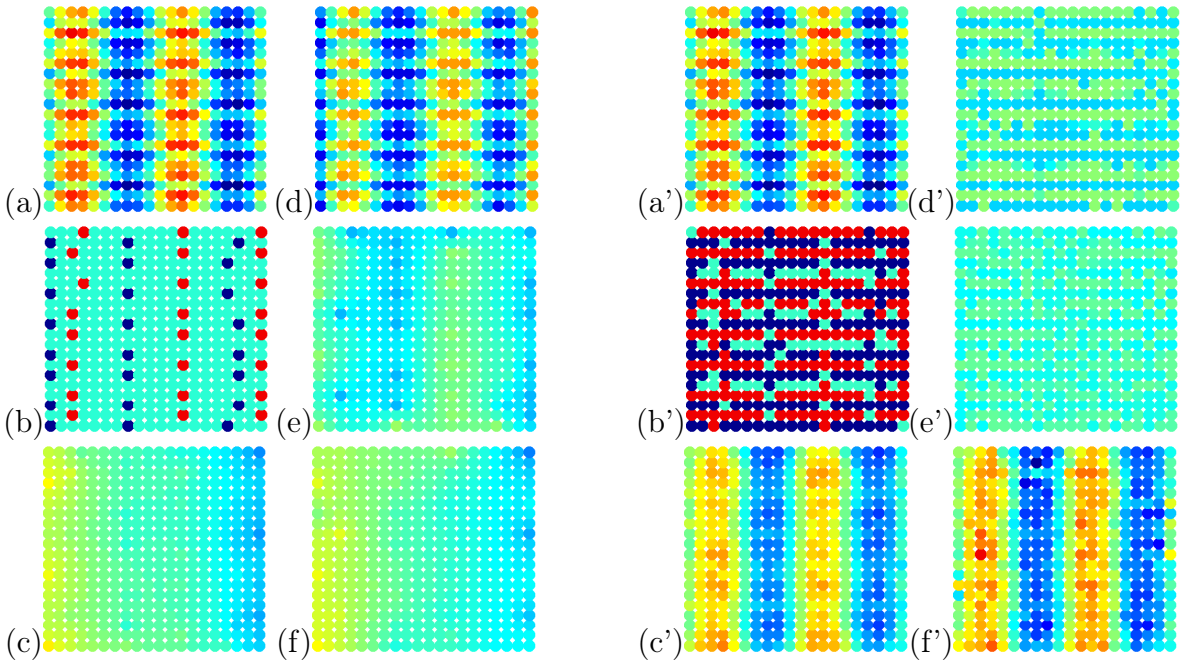


Figure E.7: Comparison of results using two different definitions of extrema. The two columns on the left (resp. right) show results obtained with the notion of extrema described in Sec. 2.2.2 (resp. in Sec. 3.3). Fig. a (resp. a') represent the same original signal described in Sec. 3.1 with $\theta = \pi/2$. Fig. b (b') represent the local maxima (in red) and minima (in blue). Fig. c (c') represent the mean of the two envelopes interpolated from these extrema. Fig. d (d') represent the first IMF, Fig. e (e') the second IMF and Fig. f (f') the residue.

4 Conclusion

A straight-forward adaptation of the classical EMD for graph signals is explored in this communication. The extension of EMD to graph signals opens many degrees of freedom for the key points of EMD : extrema, interpolation, and stopping criterion. In this first communication on the subject, we deliberately chose to use the simplest definitions. We discussed that an additional point is essential to GEMD : the way one chooses to create the graph associated to a given network, more specifically the choice of how connected one chooses to create the graph. In fact, the connectivity has a direct impact on the number of extrema of the signal, therefore a direct impact

on the very definition of what is an IMF. Future work will explore this link between connectivity and local extrema.

Liste de publications

Publications dans des journaux à comité de lecture

- N. Tremblay, P. Borgnat. *Graph wavelets and multiscale community mining*, IEEE Transactions in Signal Processing, accepted, 2014.
- N. Tremblay, P. Borgnat, J-F. Pinton, A. Barrat, M. Nornberg, C. Forest. *Bootstrapping under constraint for the assessment of group behavior in human contact networks*, Physical Review E, Vol. 88, pp. 052812, 2013.

Actes de conférences à comité de sélection

- N. Tremblay, P. Borgnat, P. Flandrin. *Graph empirical mode decomposition*, in EUSIPCO proceedings, Lisbon, Portugal, 2014.
- N. Tremblay, P. Borgnat. *Multiscale community mining in networks using the graph wavelet transform of random vectors*, in GlobalSIP proceedings, Austin, USA, 2013.
- N. Tremblay, P. Borgnat. *Multiscale community mining in networks using spectral graph wavelets*, in EUSIPCO proceedings, Marrakech, Marocco, 2013.
- N. Tremblay, P. Borgnat. *Partitionnement multiéchelle d'un graphe en communautés : détection des échelles pertinentes*, in GRETSI proceedings, Brest, France, 2013.
- N. Tremblay, P. Borgnat. *Multiscale detection of stable communities using wavelets on networks*, in ECCS proceedings, Barcelone, Spain, 2013.
- R. Fontugne, N. Tremblay, P. Borgnat, P. Flandrin, H. Esaki. *Mining anomalous electricty consumption using ensemble empirical mode decomposition*, in ICASSP proceedings, Vancouver, Canada, 2013.
- R. Fontugne , J. Ortiz , N. Tremblay , P. Borgnat, P. Flandrin , K. Fukuda , D. Culler, H. Esaki. *Strip, bind, and search : a method for identifying abnormal energy consumption in buildings*, in IPSN proceedings, Philadelphia, USA, 2013.
- N. Tremblay, P. Borgnat, J-F. Pinton, A. Barrat, M. Nornberg, C. Forest. *Constrained graph resampling for group assessment in human social networks*, in ECCS proceedings, Bruxelles, Belgium, 2012.

Contribution à des ouvrages collectifs

- P. Borgnat, C. Robardet, P. Abry, P. Flandrin, J-B. Rouquier, N. Tremblay. *A dynamical network view of Lyon's Vélo'v shared bicycle system*, in book Dynamics on and of complex networks, Volume 2, pp. 267–284, 2013.

Bibliographie

[1] Analyses sur le développement d'internet, publiées tous les ans. http://www.kpcb.com/internet-trends.

[2] Application créée par des chercheurs du MIT permettant de créer le graphe d'un réseau social à partir des métadonnées d'une boîte mail. https://immersion.media.mit.edu/.

[3] Article sur le neurone de l'INSERM. http://www.inserm.fr/thematiques/neurosciences-sciences-cognitives-neurologie-psychiatrie/dossiers-d-information/neurones.

[4] Article Wikipedia sur la pyramide DIKW. https://en.wikipedia.org/wiki/DIKW_Pyramid.

[5] Etude de l'impact énergétique du numérique. http://www.tech-pundit.com/wp-content/uploads/2013/07/Cloud_Begins_With_Coal.pdf.

[6] La DGSE enregistre nos métadonnées – article du *Monde*. http://www.lemonde.fr/societe/article/2013/07/04/revelations-sur-le-big-brother-francais_3441973_3224.html.

[7] La NSA enregistre les métadonnées – article du *Guardian*. http://www.theguardian.com/world/2013/sep/30/nsa-americans-metadata-year-documents.

[8] Le banc d'essai de Lancichinetti et al. https://sites.google.com/site/andrealancichinetti/files/weighted_networks.tar.gz.

[9] Le site de le Martelot. www.elemartelot.org.

[10] Le site internet de sociopatterns. www.sociopatterns.org.

[11] Le site internet du télescope SKA. http://www.skatelescope.org/.

[12] Les 500 vidéos les plus vues sur youtube. https://www.youtube.com/playlist?list=PLirAqAtl_h2r5g8xGajEwdXd3x1sZh8hC.

[13] MSCD Wav Toolbox. http://perso.ens-lyon.fr/nicolas.tremblay/files/MSCD_Wav_BETA.tar.gz.

[14] Quelques chiffres sur la création de données numériques mondiales. http://www.domo.com/learn/data-never-sleeps-2.

[15] Quelques statistiques de youtube. https://www.youtube.com/yt/press/fr/statistics.html.

[16] Une carte de l'internet, selon le projet OPTE. http://www.opte.org/.

[17] Visualisation du réseau Facebook. https://www.facebook.com/notes/facebook-engineering/visualizing-friendships/469716398919/.

[18] Adaptation and learning over complex networks. IEEE Signal Processing Magazine, Volume 30, 2013.

[19] Yuvraj Agarwal, Bharathan Balaji, Seemanta Dutta, Rajesh K. Gupta, and Thomas Weng. Duty-cycling buildings aggressively : The next frontier in hvac control. In IPSN'11, pages 246–257, Chicago, IL, USA, 2011.

[20] Ameya Agaskar and Yue M Lu. A spectral graph uncertainty principle. Information Theory, IEEE Transactions on, 59(7) :4338–4356, 2013.

[21] Hirotugu Akaike. A new look at the statistical model identification. Automatic Control, IEEE Transactions on, 19(6) :716–723, 1974.

[22] A. Arenas, A. Fernandez, and S. Gomez. Analysis of the structure of complex networks at different resolution levels. New Journal of Physics, 10(5) :053039, 2008.

[23] P. Balachandran, E. Airoldi, and E. Kolaczyk. Inference of Network Summary Statistics Through Network Denoising. ArXiv e-prints, October 2013.

[24] A.L. Barabási and R. Albert. Emergence of scaling in random networks. Science, 286(5439) :509, 1999.

[25] Lindsey R. Barnes, David M. Schultz, Eve C. Gruntfest, Mary H. Hayden, and Charles C. Benight. CORRIGENDUM : false alarm rate or false alarm ratio ? Weather and Forecasting, 24(5) :1452–1454, October 2009.

[26] A. Barrat, M. Barthélemy, R. Pastor-Satorras, and A. Vespignani. The architecture of complex weighted networks. Proc. Natl. Acad. Sci. (USA), 101 :3747–3752, 2004.

[27] Mathieu Bastian, Sebastien Heymann, Mathieu Jacomy, et al. Gephi : an open source software for exploring and manipulating networks. ICWSM, 8 :361–362, 2009.

[28] Gowtham Bellala, Manish Marwah, Martin Arlitt, Geoff Lyon, and Cullen E Bash. Towards an understanding of campus-scale power consumption. In Proceedings of the Third ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings, pages 73–78. ACM, 2011.

[29] Gowtham Bellala, Manish Marwah, Amip Shah, Martin Arlitt, and Cullen Bash. A finite state machine-based characterization of building entities for monitoring and control. In Proceedings of the Fourth ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings, pages 153–160. ACM, 2012.

[30] Ginestra Bianconi, Paolo Pin, and Matteo Marsili. Assessing the relevance of node features for network structure. Proceedings of the National Academy of Sciences, 106(28) :11433–11438, 2009.

[31] Manuel Blanco-Velasco, Binwei Weng, and Kenneth E. Barner. Ecg signal denoising and baseline wander correction based on the empirical mode decomposition. Computers in biology and medicine, 38(1) :1–13, jan. 2008.

[32] V.D. Blondel, J.L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. Journal of Statistical Mechanics : Theory and Experiment, 2008(10) :P10008, 2008.

[33] John Adrian Bondy and Uppaluri Siva Ramachandra Murty. Graph theory with applications, volume 6. Macmillan London, 1976.

[34] P. Borgnat, C. Robardet, P. Abry, P. Flandrin, J-B. Rouquier, and N. Tremblay. A dynamical network view of lyon's vélo'v shared bicycle system. In book "Time-varying dynamical networks", following the workshop "Dynamics on and of complex networks" of the ECCS 2011.

[35] Pierre Borgnat, Céline Robardet, Patrice Abry, Patrick Flandrin, Jean-Baptiste Rouquier, and Nicolas Tremblay. A dynamical network view of lyon's vélo'v shared bicycle system. In Dynamics On and Of Complex Networks, Volume 2, pages 267–284. Springer, 2013.

[36] Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert Gorke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. On modularity clustering. Knowledge and Data Engineering, IEEE Transactions on, 20(2) :172–188, 2008.

[37] S. P. Brooks and B. J. T. Morgan. Optimization using simulated annealing. Journal of the Royal Statistical Society. Series D (The Statistician), 44(2) :pp. 241–257, 1995.

[38] Matthew Brown, Chris Barrington-Leigh, and Zosia Brown. Kernel regression for real-time building energy analysis. Journal of Building Performance Simulation, 5(4) :263–276, 2012.

[39] Geoff Brumfiel. Down the petabyte highway. Nature, 469(20) :282–283, 2011.

[40] Wray L. Buntine. A guide to the literature on learning probabilistic networks from data. Knowledge and Data Engineering, IEEE Transactions on, 8(2) :195–210, 1996.

[41] Edward Carlstein. The use of subseries values for estimating the variance of a general statistic from a stationary sequence. The Annals of Statistics, 14(3) :pp. 1171–1179, 1986.

[42] René Carmona, Wen-Liang Hwang, and Bruno Torrésani. Practical Time-Frequency Analysis : Gabor and Wavelet Transforms, with an Implementation in S, volume 9. Academic Press, 1998.

[43] C. Cattuto, W. Van den Broeck, A. Barrat, V. Colizza, J.F. Pinton, and A. Vespignani. Dynamics of person-to-person interactions from distributed rfid sensor networks. PloS one, 5(7) :e11596, 2010.

[44] Philip Chan, Matthew Mahoney, and Muhammad Arshad. Learning rules and clusters for anomaly detection in network traffic. In Managing Cyber Threats, volume 5 of Massive Computing, pages 81–99. Springer US, 2005.

[45] Chao Chen and Diane J. Cook. Energy outlier detection in smart environments. In Artificial Intelligence and Smarter Living, volume WS-11-07 of AAAI Workshops. AAAI, 2011.

[46] Chitika. Chitika Insights : The Value of Google Result Positioning. Chitika, 2013.

[47] F. Chung and L. Lu. The average distances in random graphs with given expected degrees. Proceedings of the National Academy of Sciences, 99(25) :15879, 2002.

[48] F.R.K. Chung. Spectral graph theory. Number 92. Amer Mathematical Society, 1997.

[49] F.R.K. Chung and L. Lu. Complex graphs and networks. Number 107. Amer Mathematical Society, 2006.

[50] John Clark and Derek Allan Holton. A first look at graph theory, volume 1. World Scientific, 1991.

[51] Aaron Clauset, Cosma Rohilla Shalizi, and Mark EJ Newman. Power-law distributions in empirical data. SIAM review, 51(4) :661–703, 2009.

[52] Stéphan Clémençon, Hector De Arazoza, Fabrice Rossi, and Viet Chi Tran. Hierarchical clustering for graph visualization. arXiv preprint arXiv :1210.5693, 2012.

[53] R.R. Coifman and M. Maggioni. Diffusion wavelets. Applied and Computational Harmonic Analysis, 21(1) :53–94, 2006.

[54] James S Coleman et al. Introduction to mathematical sociology. London Free Press Glencoe., 1964.

[55] M. Crovella and E. Kolaczyk. Graph wavelets for spatial traffic analysis. In INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies, volume 3, pages 1848–1857. Ieee, 2003.

[56] AC Davison and DV Hinkley. Bootstrap methods and their applicationcambridge university press. Cambridge, UK, 193, 1997.

[57] Yves-Alexandre de Montjoye, César A Hidalgo, Michel Verleysen, and Vincent D Blondel. Unique in the crowd : The privacy bounds of human mobility. Scientific reports, 3, 2013.

[58] J.C. Delvenne, S.N. Yaliraki, and M. Barahona. Stability of graph communities across time scales. Proceedings of the National Academy of Sciences, 107(29) :12755–12760, 2010.

[59] William E Donath and Alan J Hoffman. Lower bounds for the partitioning of graphs. IBM Journal of Research and Development, 17(5) :420–425, 1973.

[60] L. Donetti and M.A. Munoz. Detecting network communities : a new systematic and efficient algorithm. Journal of Statistical Mechanics : Theory and Experiment, 2004 :P10012, 2004.

[61] L. Donetti and M.A. Muñoz. Improved spectral algorithm for the detection of network communities. Arxiv preprint physics/0504059, 2005.

[62] David L Donoho. Compressed sensing. Information Theory, IEEE Transactions on, 52(4) :1289–1306, 2006.

[63] Florian Dorfler and Francesco Bullo. Kron reduction of graphs with applications to electrical networks. Circuits and Systems I : Regular Papers, IEEE Transactions on, 60(1) :150–163, 2013.

[64] Olive Jean Dunn. Multiple comparisons among means. Journal of the American Statistical Association, 56(293) :pp. 52–64, 1961.

[65] Richard Durrett, Richard Durrett, and Richard Durrett. Random graph dynamics, volume 200. Cambridge university press Cambridge, 2007.

[66] N. Eagle and A. Pentland. Reality mining : sensing complex social systems. Personal and Ubiquitous Computing, 10(4) :255–268, 2006.

[67] B. Efron. The jackknife, the bootstrap, and other resampling plans, volume 38. Society for Industrial and Applied Mathematics Philadelphia, 1982.

[68] Bradley Efron. Bootstrap methods : another look at the jackknife. The annals of Statistics, pages 1–26, 1979.

[69] Venkatesan N Ekambaram, Giulia Fanti, Babak Ayazifar, and Kannan Ramchandran. Critically-sampled perfect-reconstruction spline-wavelet filterbanks for graph signals. submitted to IEEE GlobeSip, 2013.

[70] Venkatesan N Ekambaram, Giulia C Fanti, Babak Ayazifar, and Kannan Ramchandran. Multiresolution graph signal processing via circulant structures. In Digital Signal Processing and Signal Processing Education Meeting (DSP/SPE), 2013 IEEE, pages 112–117. IEEE, 2013.

[71] H. Eldardiry and J. Neville. A resampling technique for relational data graphs. In Proceedings of the 2nd SNA Workshop, 14th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2008.

[72] Frank Emmert-Streib, Galina Glazko, Ricardo De Matos Simoes, et al. Statistical inference and reverse engineering of gene regulatory networks from observational expression data. Frontiers in genetics, 3 :8, 2012.

[73] P Erdös and A Rényi. On random graphs i. Publicationes Mathematicae, 6 :290–297, 1959.

[74] Varick L. Erickson, Miguel Á. Carreira-Perpiñán, and Alberto Cerpa. Observe : Occupancy-based system for efficient reduction of hvac energy. In IPSN'11, pages 258–269, Chicago, IL, USA, 2011.

[75] TS Evans and R Lambiotte. Line graphs, link partitions, and overlapping communities. Physical Review E, 80(1) :016105, 2009.

[76] Fabrizio De Vico Fallani, Vincenzo Nicosia, Vito Latora, and Mario Chavez. Nonparametric resampling of random walks for spectral network clustering. Physical Review E, 89(1) :012802, 2014.

[77] Joseph Felsenstein et al. Confidence limits on phylogenies : an approach using the bootstrap. Evolution, 39(4) :783–791, 1985.

[78] Miroslav Fiedler. Algebraic connectivity of graphs. Czechoslovak Mathematical Journal, 23(2) :298–305, 1973.

[79] R. Fontugne, N. Tremblay, P. Borgnat, P. Flandrin, and H. Esaki. Mining anomalous electricity consumption using ensemble empirical mode decomposition. In Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on, pages 5238–5242, May 2013.

[80] Romain Fontugne, Jorge Ortiz, David Culler, and Hiroshi Esaki. Empirical mode decomposition for intrinsic-relationship extraction in large sensor deployments. In IoT-App'12, Workshop on Internet of Things Applications, Beijing, China, 2012.

[81] Romain Fontugne, Jorge Ortiz, Nicolas Tremblay, Pierre Borgnat, Patrick Flandrin, Kensuke Fukuda, David Culler, and Hiroshi Esaki. Strip, bind, and search : a method for identifying abnormal energy consumption in buildings. In Proceedings of the 12th international conference on Information processing in sensor networks, pages 129–140. ACM, 2013.

[82] LR Ford and Delbert Ray Fulkerson. Flows in networks, volume 3. Princeton Princeton University Press, 1962.

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

[84] S. Fortunato and M. Barthelemy. Resolution limit in community detection. Proceedings of the National Academy of Sciences, 104(1) :36, 2007.

[85] E. B. Fowlkes and C. L. Mallows. A method for comparing two hierarchical clusterings. Journal of the American Statistical Association, 78(383) :pp. 553–569, 1983.

[86] N. Friedman, M. Goldszmidt, and A. Wyner. Data analysis with bayesian networks : A bootstrap approach. In Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence, pages 196–205. Morgan Kaufmann Publishers Inc., 1999.

[87] Matan Gavish, Boaz Nadler, and Ronald R Coifman. Multiscale wavelets on trees, graphs and high dimensional data : Theory and applications to semi supervised learning. In Proceedings of the 27th International Conference on Machine Learning (ICML-10), pages 367–374, 2010.

[88] David Gfeller, Jean-Cédric Chappelier, and Paolo De Los Rios. Finding instabilities in the community structure of complex networks. Physical Review E, 72(5) :056135, 2005.

[89] M. Girvan and M.E.J. Newman. Community structure in social and biological networks. Proceedings of the National Academy of Sciences, 99(12) :7821, 2002.

[90] Fred Glover. Future paths for integer programming and links to artificial intelligence. Computers & Operations Research, 13(5) :533–549, 1986.

[91] Michel L Goldstein, Steven A Morris, and Gary G Yen. Problems with fitting to the power-law distribution. The European Physical Journal B-Condensed Matter and Complex Systems, 41(2) :255–258, 2004.

[92] Benjamin H. Good, Yves-Alexandre de Montjoye, and Aaron Clauset. Performance of modularity maximization in practical contexts. Physical Review E, 81(4), April 2010.

[93] L. Grady and E. Schwartz. Anisotropic interpolation on graphs : the combinatorial dirichlet problem. Technical Report Boston University, 2003.

[94] P. Hall, J.L. Horowitz, and B.Y. Jing. On blocking rules for the bootstrap with dependent data. Biometrika, 82(3) :561–574, 1995.

[95] Peter Hall. Resampling a coverage pattern. Stochastic Processes and their Applications, 20(2) :231 – 246, 1985.

[96] D.K. Hammond, P. Vandergheynst, and R. Gribonval. Wavelets on graphs via spectral graph theory. Applied and Computational Harmonic Analysis, 30(2) :129–150, 2011.

[97] Sami Hanhijärvi, Gemma C Garriga, and Kai Puolamäki. Randomization techniques for graphs. In SDM, pages 780–791. SIAM, 2009.

[98] Frank Harary, Derbiau Hsu, and Zevi Miller. The biparticity of a graph. Journal of graph theory, 1(2) :131–133, 1977.

[99] John A Hartigan. Using subsample values as typical values. Journal of the American Statistical Association, 64(328) :1303–1317, 1969.

[100] T. Hasan and M.K. Hasan. Suppression of residual noise from speech signals using empirical mode decomposition. Signal Processing Letters, IEEE, 16(1) :2 –5, jan. 2009.

[101] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. The elements of statistical learning : Data Mining, Inference, and Prediction. Springer, 2009.

[102] Tapio Heimo, Jussi M Kumpula, Kimmo Kaski, and Jari Saramäki. Detecting modules in dense weighted networks with the potts method. J. Stat. Mech, page P08007, 2008.

[103] George C Homans. The human group, volume 7. Routledge, 2013.

[104] Hai Huang and Jiaqiang Pan. Speech pitch determination based on hilbert-huang transform. Signal Processing, 86(4) :792 – 803, 2006.

[105] Jianbin Huang, Heli Sun, Yaguang Liu, Qinbao Song, and Tim Weninger. Towards online multiresolution community detection in large-scale networks. PLoS ONE, 6(8) :e23829, 08 2011.

[106] N. Huang, Z. Shen, S. Long, M. Wu, H. Shih, Q. Zheng, N.C. Yen, C.C. Tung, and H. Liu. The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis. Proceedings of the Royal Society of London. Series A : Mathematical, Physical and Engineering Sciences, 454(1971) :903–995, 1998.

[107] Norden E. Huang. Computing frequency by using generalized zero-crossing applied to intrinsic mode functions. U.S. Patent 6,990,436 B1, 2006.

[108] Norden E. Huang, Zhaohua Wu, Steven R. Long, Kenneth C. Arnold, Xianyao Chen, and Karin Blank. On instantaneous frequency. Advances in Adaptive Data Analysis, pages 177–229, 2009.

[109] P.J. Huber and E.M. Ronchetti. Robust Statistics. Wiley Series in Probability and Statistics. Wiley, 2009.

[110] L. Hubert and P. Arabie. Comparing partitions. Journal of classification, 2(1) :193–218, 1985.

[111] Christian Hubler, H-P Kriegel, Karsten Borgwardt, and Zoubin Ghahramani. Metropolis algorithms for representative subgraph sampling. In Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on, pages 283–292. IEEE, 2008.

[112] John P. Huelsenbeck, Fredrik Ronquist, et al. Mrbayes : Bayesian inference of phylogenetic trees. Bioinformatics, 17(8) :754–755, 2001.

[113] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot. Pocket switched networks and human mobility in conference environments. In Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking, pages 244–251. ACM, 2005.

[114] L. Isella, J. Stehlé, A. Barrat, C. Cattuto, J.F. Pinton, and W. Van den Broeck. What's in a crowd ? analysis of face-to-face behavioral networks. Journal of theoretical biology, 271(1) :166–180, 2011.

[115] P. Jaccard. Distribution de la flore alpine dans le bassin des dranses et dans quelques régions voisines. Bulletin de la Société Vaudoise des Sciences Naturelles, 39 :241–272, 1901.

[116] A.K. Jain, M.N. Murty, and P.J. Flynn. Data clustering : a review. ACM computing surveys (CSUR), 31(3) :264–323, 1999.

[117] Anil K Jain and Richard C Dubes. Algorithms for clustering data. Prentice-Hall, Inc., 1988.

[118] Maarten Jansen, Guy P Nason, and Bernard W Silverman. Multiscale methods for data on graphs and irregular multidimensional situations. Journal of the Royal Statistical Society : Series B (Statistical Methodology), 71(1) :97–125, 2009.

[119] Brian Karrer, Elizaveta Levina, and M. E. J. Newman. Robustness of community structure in networks. Phys. Rev. E, 77 :046119, Apr 2008.

[120] Natallia Katenka, Eric D Kolaczyk, et al. Inference and characterization of multi-attribute networks with application to computational biology. The Annals of Applied Statistics, 6(3) :1068–1094, 2012.

[121] S. Katipamula and M.R. Brambley. Review article : Methods for fault detection, diagnostics, and prognostics for building systems—a review, part i. HVAC&R Research, 11(1) :3–25, 2005.

[122] S. Katipamula and M.R. Brambley. Review article : Methods for fault detection, diagnostics, and prognostics for building systems—a review, part ii. HVAC&R Research, 11(2) :169–187, 2005.

[123] Brian W Kernighan and Shen Lin. An efficient heuristic procedure for partitioning graphs. Bell system technical journal, 49(2) :291–307, 1970.

[124] Younghun Kim, Rahul Balani, Han Zhao, and Mani B. Srivastava. Granger causality analysis on IP traffic and circuit-level energy monitoring. In BuildSys'10, pages 43–48, 2010.

[125] Hans R Kunsch et al. The jackknife and the bootstrap for general stationary observations. The Annals of Statistics, 17(3) :1217–1241, 1989.

[126] S.N. Lahiri. Theoretical comparisons of block bootstrap methods. The Annals of Statistics, 27(1) :386–404, 1999.

[127] R. Lambiotte. Multi-scale modularity in complex networks. In Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt), 2010 Proceedings of the 8th International Symposium on, pages 546–553. IEEE, 2010.

[128] A. Lancichinetti and S. Fortunato. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. Physical Review E, 80(1) :016118, 2009.

[129] A. Lancichinetti, S. Fortunato, and J. Kertész. Detecting the overlapping and hierarchical community structure in complex networks. New Journal of Physics, 11 :033015, 2009.

[130] A. Lancichinetti, F. Radicchi, J.J. Ramasco, and S. Fortunato. Finding statistically significant communities in networks. PloS one, 6(4) :e18961, 2011.

[131] Andrea Lancichinetti and Santo Fortunato. Community detection algorithms : a comparative analysis. Physical review E, 80(5) :056117, 2009.

[132] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. Benchmark graphs for testing community detection algorithms. Physical Review E, 78(4) :046110, 2008.

[133] Erwan Le Martelot and Chris Hankin. Multi-scale community detection using stability as optimisation criterion in a greedy algorithm. In Proceedings of the 2011 International Conference on Knowledge Discovery and Information Retrieval (KDIR 2011), pages 216–225, Paris, October 2011. SciTePress.

[134] Erwan Le Martelot and Chris Hankin. Fast multi-scale detection of relevant communities in large-scale networks. The Computer Journal, 56(9) :1136–1150, 2013.

[135] T. Lee and T. B. M. J. Ouarda. Prediction of climate nonstationary oscillation processes with empirical mode decomposition. Journal of Geophysical Research, 116, 2011.

[136] R.B. Lehoucq and D.C. Sorensen. Deflation techniques for an implicitly restarted arnoldi iteration. SIAM J. Matrix Analysis and Applications, 17 :789–821, 1996.

[137] N. Leonardi and D. Van De Ville. Tight wavelet frames on multislice graphs. IEEE Transactions on Signal Processing, 61(13) :3357–3367, 2013.

[138] Jure Leskovec and Christos Faloutsos. Sampling from large graphs. In Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 631–636. ACM, 2006.

[139] C Li, H Wang, W De Haan, CJ Stam, and P Van Mieghem. The correlation of metrics in complex networks with applications in functional brain networks. Journal of Statistical Mechanics : Theory and Experiment, 2011(11) :P11018, 2011.

[140] Regina Y Liu and Kesar Singh. Moving blocks jackknife and bootstrap capture weak dependence. Exploring the limits of bootstrap, 225 :248, 1992.

[141] S. Mallat. A wavelet tour of signal processing. Academic press, 1999.

[142] D. Mandic, N. Rehman, Z. Wu, and N. Huang. Empirical mode decomposition-based time-frequency analysis of multivariate signals. IEEE Signal Processing Magazine, 30(6) :74–86, 2013.

[143] Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. Introduction to information retrieval, volume 1. Cambridge university press Cambridge, 2008.

[144] Vivien Marx. Biology : The big challenges of big data. Nature, 498(7453) :255–260, 2013.

[145] PJ Mc et al. Pseudo-replication : Half samples. Review of the International Statistical Institute, 37(3) :239–264, 1969.

[146] Patrick N McGraw and Michael Menzinger. Laplacian spectra as a diagnostic tool for network structure and dynamics. Physical Review E, 77(3) :031102, 2008.

[147] Marina Meilă. Comparing clusterings—an information based distance. Journal of Multivariate Analysis, 98(5) :873 – 895, 2007.

[148] R Vilela Mendes, Hugo C Mendes, and Tanya Araújo. Signal processing on graphs : Transforms and tomograms. arXiv preprint arXiv :1406.2185, 2014.

[149] Benjamin A Miller, Nadya T Bliss, and Patrick J Wolfe. Toward signal processing theory for graphs and non-euclidean data. In Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on, pages 5414–5417. IEEE, 2010.

[150] Benjamin A Miller, Nadya T Bliss, Patrick J Wolfe, and Michelle S Beard. Detection theory for graphs. Lincoln Laboratory Journal, 20(1), 2013.

[151] J. Miller and A. Hagberg. Efficient generation of networks with given expected degrees. Algorithms and Models for the Web Graph, pages 115–126, 2011.

[152] Michael Molloy and Bruce Reed. A critical point for random graphs with a given degree sequence. Random structures & algorithms, 6(2-3) :161–180, 1995.

[153] Yuji Nakatsukasa, Naoki Saito, and Ernest Woei. Mysteries around the graph laplacian eigenvalue 4. Linear Algebra and Its Applications, 438(8) :3231–3246, 2013.

[154] S. Narang, A. Gadde, and A. Ortega. Signal processing techniques for interpolation in graph structured data. In ICASSP, pages 5445–5449, 2013.

[155] S. Narang and A. Ortega. Lifting based wavelet transforms on graphs. In Proc. of APSIPA Annual Summit and Conference (APSIPA ASC), 2009.

[156] S. Narang and A. Ortega. Perfect reconstruction two-channel wavelet filter banks for graph structured data. IEEE Transactions on Signal Processing, 60(6) :2786–2799, 2012.

[157] Sunil K Narang and Antonio Ortega. Local two-channel critically sampled filter-banks on graphs. In Image Processing (ICIP), 2010 17th IEEE International Conference on, pages 333–336. IEEE, 2010.

[158] Sunil K Narang and Antonio Ortega. Downsampling graphs using spectral theory. In Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on, pages 4208–4211. IEEE, 2011.

[159] Sunil K NARANG and Antonio ORTEGA. Compact support biorthogonal wavelet filterbanks for arbitrary undirected graphs. IEEE transactions on signal processing, 61(17-20) :4673–4685, 2013.

[160] Mark Newman. Networks : an introduction. Oxford University Press, 2010.

[161] M.E.J. Newman. Modularity and community structure in networks. Proceedings of the National Academy of Sciences, 103(23) :8577, 2006.

[162] M.E.J. Newman and M. Girvan. Finding and evaluating community structure in networks. Physical review E, 69(2) :026113, 2004.

[163] J. C. Nunes, S. Guyot, and E. Delechelle. Texture analysis based on local analysis of the bidimensional empirical mode decomposition. Machine Vision and Applications, 16 :177–188, 2005.

[164] J.C Nunes, Y Bouaoune, E Delechelle, O Niang, and Ph Bunel. Image analysis by bidimensional empirical mode decomposition. Image and Vision Computing, 21(12) :1019 – 1026, 2003.

[165] Efstathios Paparoditis and Dimitris N. Politis. Tapered block bootstrap. Biometrika, 88(4) :pp. 1105–1119, 2001.

[166] D. Patnaik, M. Marwah, R.K. Sharma, and N. Ramakrishnan. Temporal data mining approaches for sustainable chiller management in data centers. ACM Transactions on Intelligent Systems and Technology, 2(4), 2011.

[167] Dimitris N Politis and Joseph P Romano. A circular block-resampling procedure for stationary data. Exploring the limits of bootstrap, pages 263–270, 1992.

[168] Dimitris N. Politis and Joseph P. Romano. The stationary bootstrap. Journal of the American Statistical Association, 89(428) :pp. 1303–1313, 1994.

[169] P. Pons and M. Latapy. Post-processing hierarchical community structures : Quality improvements and multi-scale view. Theoretical Computer Science, 412(8) :892–900, 2011.

[170] Maurice H Quenouille. Approximate tests of correlation in time-series. Journal of the Royal Statistical Society. Series B (Methodological), 11(1) :68–84, 1949.

[171] Menaut R. Mesure et analyse d'un réseau social. Rapport de stage de Licence 3, 2013.

[172] William M. Rand. Objective criteria for the evaluation of clustering methods. Journal of the American Statistical Association, 66(336) :pp. 846–850, 1971.

[173] J. Reichardt and S. Bornholdt. Statistical mechanics of community detection. Physical Review E, 74(1) :016110, 2006.

[174] Bruno Ribeiro and Don Towsley. Estimating and sampling graphs with multidimensional random walks. In Proceedings of the 10th ACM SIGCOMM conference on Internet measurement, pages 390–403. ACM, 2010.

[175] Stuart A Rice. The identification of blocs in small political bodies. American Political Science Review, 21(03) :619–627, 1927.

[176] G. Rilling, F. Flandrin, and P. Gonçalves. On empirical mode decomposition and its algorithms. In IEEE-EURASIP Workshop NSIP, June 2003.

[177] Peter Ronhovde and Zohar Nussinov. Local resolution-limit-free potts model for community detection. Physical Review E, 81(4) :046114, 2010.

[178] M. Rosvall and C.T. Bergstrom. Mapping change in large networks. PloS one, 5(1) :e8694, 2010.

[179] Martin Rosvall, Daniel Axelsson, and Carl T Bergstrom. The map equation. The European Physical Journal Special Topics, 178(1) :13–23, 2009.

[180] Martin Rosvall and Carl T Bergstrom. Maps of random walks on complex networks reveal community structure. Proceedings of the National Academy of Sciences, 105(4) :1118–1123, 2008.

[181] Martin Rosvall and Carl T Bergstrom. Multilevel compression of random walks on networks reveals hierarchical organization in large integrated systems. PloS one, 6(4) :e18209, 2011.

[182] M. Sales-Pardo, R. Guimera, A.A. Moreira, and L.A.N. Amaral. Extracting the hierarchical organization of complex systems. Proceedings of the National Academy of Sciences, 104(39) :15224–15229, 2007.

[183] A. Sandryhaila and J.M.F. Moura. Discrete signal processing on graphs. Signal Processing, IEEE Transactions on, 61(7) :1644–1656, April 2013.

[184] A Sandryhaila and J.M.F. Moura. Discrete signal processing on graphs : Frequency analysis. Signal Processing, IEEE Transactions on, 62(12) :3042–3054, 2014.

[185] Aliaksei Sandryhaila and José MF Moura. Discrete signal processing on graphs : Graph filters. In ICASSP, pages 6163–6166, 2013.

[186] Aliaksei Sandryhaila and José MF Moura. Discrete signal processing on graphs : Graph fourier transform. In ICASSP, pages 6167–6170, 2013.

[187] J. Schein and S.T. Bushby. A hierarchical rule-based fault detection and diagnostic method for hvac systems. HVAC&R Research, 12(1) :111–125, 2006.

[188] Gideon Schwarz et al. Estimating the dimension of a model. The annals of statistics, 6(2) :461–464, 1978.

[189] John E. Seem. Using intelligent data analysis to detect abnormal energy consumption in buildings. Energy and Buildings, 39(1) :52 – 58, 2007.

[190] Godwin Shen and Antonio Ortega. Tree-based wavelets for image coding : Orthogonalization and tree selection. In Picture Coding Symposium, 2009. PCS 2009, pages 1–4. IEEE, 2009.

[191] D. Shuman, S. Narang, P. Frossard, A. Ortega, and P. Vandergheynst. The emerging field of signal processing on graphs : Extending high-dimensional data analysis to networks and other irregular domains. IEEE SP Mag., 30(3) :83–98, 2013.

[192] David I Shuman, Mohammad Javad Faraji, and Pierre Vandergheynst. A framework for multiscale transforms on graphs. arXiv preprint arXiv :1308.4942, 2013.

[193] David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The emerging field of signal processing on graphs : Extending high-dimensional data analysis to networks and other irregular domains. Signal Processing Magazine, IEEE, 30(3) :83–98, 2013.

[194] David I Shuman, Benjamin Ricaud, and Pierre Vandergheynst. A windowed graph fourier transform. In Statistical Signal Processing Workshop (SSP), 2012 IEEE, pages 133–136. IEEE, 2012.

[195] David I Shuman, Benjamin Ricaud, and Pierre Vandergheynst. Vertex-frequency analysis on graphs. arXiv preprint arXiv :1307.5708, 2013.

[196] David I Shuman, Pierre Vandergheynst, and Pascal Frossard. Chebyshev polynomial approximation for distributed signal processing. In Distributed Computing in Sensor Systems and Workshops (DCOSS), 2011 International Conference on, pages 1–8. IEEE, 2011.

[197] David I Shuman, Christoph Wiesmeyr, Nicki Holighaus, and Pierre Vandergheynst. Spectrum-adapted tight graph wavelet and vertex-frequency frames. arXiv preprint arXiv :1311.0897, 2013.

[198] Christian Soize. Méthodes mathématiques en analyse du signal. Masson, 1993.

[199] Daniel A Spielmat and Shang-Hua Teng. Spectral partitioning works : Planar graphs and finite element meshes. In Foundations of Computer Science, 1996. Proceedings., 37th Annual Symposium on, pages 96–105. IEEE, 1996.

[200] Michele Starnini, Andrea Baronchelli, and Romualdo Pastor-Satorras. Modeling human dynamics of face-to-face interaction networks. Phys. Rev. Lett., 110 :168701, Apr 2013.

[201] J. Stehlé, A. Barrat, and G. Bianconi. Dynamical and bursty interactions in social networks. Physical review E, 81(3) :035101, 2010.

[202] J. Stehle, N. Voirin, A. Barrat, C. Cattuto, L. Isella, J.F. Pinton, M. Quaggiotto, W. Van Den Broeck, C. Regis, B. Lina, et al. High-resolution measurements of face-to-face contact patterns in a primary school. PloS one, 6(8) :e23176, 2011.

[203] Juliette Stehlé, François Charbonnier, Tristan Picard, Ciro Cattuto, and Alain Barrat. Gender homophily from spatial behavior in a primary school : a sociometric study. Social Networks, 35(4) :604–613, 2013.

[204] Lionel Tabourier. Thèse de doctorat : Méthode de comparaison des topologies de graphes complexes. Applications aux réseaux sociaux. Université Pierre et Marie Curie, Paris, 2010.

[205] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. Science, 290 :2319–2323, 2000.

[206] Robert Tibshirani, Guenther Walther, and Trevor Hastie. Estimating the number of clusters in a data set via the gap statistic. Journal of the Royal Statistical Society : Series B (Statistical Methodology), 63(2) :411–423, 2001.

[207] M.E. Torres, M.A. Colominas, G. Schlotthauer, and P. Flandrin. A complete ensemble empirical mode decomposition with adaptive noise. In IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 4144 –4147, May 2011.

[208] V. A. Traag and Jeroen Bruggeman. Community detection in networks with positive and negative links. Phys. Rev. E, 80 :036115, Sep 2009.

[209] Amanda L Traud, Eric D Kelsic, Peter J Mucha, and Mason A Porter. Comparing community structure to characteristics in online collegiate social networks. SIAM review, 53(3) :526–543, 2011.

[210] N. Tremblay and P. Borgnat. Multiscale community mining in networks using the graph wavelet transform of random vectors. In Global Conference on Signal and Information Processing (GlobalSIP), 2013 IEEE, pages 463–466, Dec 2013.

[211] N. Tremblay and P. Borgnat. Multiscale detection of stable communities using wavelets on networks. European Conference of Complex Systems, 2013.

[212] N. Tremblay and P. Borgnat. Graph wavelets and multiscale community mining. IEEE Transactions on Signal Processing, accepted, 2014.

[213] N. Tremblay, P. Borgnat, J-F. Pinton, A. Barrat, M. Nornberg, and C. Forest. Constrained graph resampling for group assessment in human social networks. European Conference of Complex Systems, 2012.

[214] Nicolas Tremblay, Alain Barrat, Cary Forest, Mark Nornberg, Jean-François Pinton, and Pierre Borgnat. Bootstrapping under constraint for the assessment of group behavior in human contact networks. Physical Review E, 88(5) :052812, 2013.

[215] Nicolas Tremblay and Pierre Borgnat. Multiscale community mining in networks using spectral graph wavelets. In Signal Processing Conference (EUSIPCO), 2013 Proceedings of the 21st European, pages 1–5, Sept 2013.

[216] Nicolas Tremblay and Pierre Borgnat. Partitionnement multi-échelle d'un graphe en communautés : détection des échelles pertinentes. In GRETSI, pages x+4, 2013.

[217] Nicolas Tremblay, Pierre Borgnat, and Patrick Flandrin. Graph empirical mode decomposition. In Signal Processing Conference (EUSIPCO), 2014 Proceedings of the 22st European, pages 1–5, Sept 2014.

[218] John W Tukey. Bias and confidence in not-quite large samples. In Annals of Mathematical Statistics, volume 29, pages 614–614, 1958.

[219] Energy Information Administration (US). Annual Energy Review 2011. Government Printing Office, 2012.

[220] Piet Van Mieghem. Graph spectra for complex networks. Cambridge University Press, 2011.

[221] U. Von Luxburg. A tutorial on spectral clustering. Statistics and computing, 17(4) :395–416, 2007.

[222] David L Wallace. Comment. Journal of the American Statistical Association, 78(383) :569–576, 1983.

[223] D.J. Watts and S.H. Strogatz. Collective dynamics of 'small-world'networks. Nature, 393(6684) :440–442, 1998.

[224] Duncan J Watts. Small worlds : the dynamics of networks between order and randomness. Princeton university press, 1999.

[225] Robert S Weiss and Eugene Jacobson. A method for the analysis of the structure of complex organizations. American Sociological Review, pages 661–668, 1955.

[226] Michael Wrinch, Tarek H.M. EL-Fouly, and Steven Wong. Anomaly detection of building systems using energy demand frequency domain anlaysis. In IEEE Power & Energy Society General Meeting, San-Diego, CA, USA, 2012.

[227] Z. Wua, N.E. Huang, and X. Chan. The multi-dimensional ensemble empirical mode decomposition method. Adv. Adapt. Data Anal., 1(3) :339–372, 2009.

[228] X. Ying and X. Wu. Graph generation with prescribed feature constraints. In Proc. of the 9th SIAM Conference on Data Mining, 2009.

[229] W Zachary. An information flow modelfor conflict and fission in small groups1. Journal of anthropological research, 33(4) :452–473, 1977.

[230] Y. Zhang, E. D. Kolaczyk, and B. D. Spencer. Estimating Network Degree Distributions Under Sampling : An Inverse Problem, with Applications to Monitoring Social Media Networks. ArXiv e-prints, May 2013.

[231] Kun Zhao, Juliette Stehlé, Ginestra Bianconi, and Alain Barrat. Social network dynamics of face-to-face interactions. Phys. Rev. E, 83 :056109, May 2011.

[232] Q. Zhou, S. Wang, and Z. Ma. A model-based fault detection and diagnosis strategy for hvac systems. International Journal of Energy Research, 33(10) :903–918, 2009.

[233] Etay Ziv, Robin Koytcheff, Manuel Middendorf, and Chris Wiggins. Systematic identification of statistically significant network measures. Physical Review E, 71(1) :016110, 2005.

[234] A.M. Zoubir and D.R. Iskander. Bootstrap techniques for signal processing. Cambridge University Press, 2004.

Table des matières

Introduction **1**

1 Généralités sur les graphes et le traitement du signal sur graphes **11**
1 Généralités sur les graphes 12
1.1 Qu'est-ce qu'un graphe ? 12
1.2 Matrice d'adjacence 13
1.3 Les graphes pondérés 13
1.4 Quelques définitions 13
1.5 Matrice laplacienne 15
1.6 Modèles de graphes aléatoires 15
2 Notions de base du traitement du signal sur graphes 16
2.1 Qu'est-ce qu'un signal sur graphe ? 16
2.2 Le traitement du signal discret : un cas particulier 16
2.3 Quelques résultats du traitement du signal discret classique . . 17
2.3.1 La transformée de Fourier discrète 18
2.3.2 L'opérateur de convolution circulaire 18
2.3.3 La translation 19
2.3.4 La modulation 19
2.3.5 Le filtrage 19
2.3.6 L'opérateur de retard 20
2.4 Le traitement du signal sur graphes selon Sandryhaila et Moura 20
2.4.1 Analogie fondamentale de Sandryhaila et Moura . . . 20
2.4.2 Filtrer un signal sur graphe 21
2.4.3 Transformée de Fourier sur graphe 21
2.4.4 En résumé 22
2.5 Le traitement du signal sur graphes selon Vandergheynst et al. 22
2.5.1 Analogie fondamentale de Vandergheynst et al. . . . 22
2.5.2 Quelques précisions importantes 23
2.5.3 La fréquence d'un mode de Fourier 24
2.6 Autres analogies possibles 24
2.6.1 La matrice laplacienne normalisée 25
2.6.2 Extension aux graphes orientés 25
2.7 Résultats du traitement du signal sur graphe 26
2.7.1 La transformée de Fourier sur graphe 26
2.7.2 La fréquence d'un mode de Fourier 27
2.7.3 Phénomène de localisation des modes de Fourier . . . 27

2.7.4 Point clé 29
2.7.5 Opérations usuelles 29
3 La transformée en ondelettes sur graphes 31
3.1 La famille d'ondelettes classiques 31
3.2 La famille d'ondelettes sur graphes 32
3.3 La transformée en ondelettes sur graphe 33
3.4 Algorithme rapide de transformée en ondelettes sur graphe . . 34
3.5 Le noyau de filtre d'ondelettes 35
3.6 Quelques illustrations 36

2 Détection multiéchelle de communautés à l'aide d'ondelettes 39
1 Une communauté dans un graphe 41
1.1 Qu'est-ce qu'une communauté ? 41
1.2 De l'utilité de la recherche de communautés 42
1.3 Une brève histoire de la recherche de communautés 43
1.4 Partitionnement d'un graphe en communautés 43
2 Partitionner un graphe en communautés 44
2.1 La modularité : une mesure de la qualité d'une partition . . . 44
2.2 L'algorithme de Louvain 45
2.3 Clusterings hiérarchiques et dendrogrammes 45
2.4 Un algorithme spectral 47
3 État de l'art des méthodes de détection multiéchelle 48
3.1 Les limites de la modularité 50
3.2 L'intérêt d'une vision multiéchelle 50
3.3 Poser le problème multiéchelle 51
3.4 Trouver une partition par échelle : méthodes existantes 52
3.4.1 De la physique statistique : Reichardt et Bornholdt . 52
3.4.2 Ajout de boucles : la méthode d'Arenas 53
3.4.3 Marcheurs aléatoires : la méthode de Delvenne . . . 53
3.4.4 Autres méthodes 54
3.5 Trouver les échelles pertinentes : méthodes existantes 55
3.5.1 Stabilité par perturbation directe du graphe 56
3.5.2 Stabilité en terme d'intervalle d'échelle 57
3.5.3 Stabilité par stochasticité de l'algorithme 57
4 Ondelettes sur graphes et partitionnement multiéchelle 57
4.1 Pourquoi une autre méthode ? 58
4.2 Le noyau de filtre d'ondelette et bornes du paramètre d'échelle 58
4.3 Détection de communautés à une échelle s 62
4.3.1 Calcul des ondelettes 62
4.3.2 Création du dendrogramme à l'échelle s 62
4.3.3 Couper le dendrogramme 63
4.3.4 Bilan d'étape : algorithme avec calcul des ondelettes 66
4.4 Détection rapide de communautés à une échelle s 67
4.4.1 Transformée en ondelettes de signaux aléatoires . . . 67
4.4.2 Bilan d'étape : algorithme avec vecteurs aléatoires . . 69
4.5 Mesure de stabilité de l'échelle s 69

4.6 Test statistique 70
5 Illustrations et comparaison avec d'autres méthodes 71
5.1 Illustration sur un modèle de graphe hiérarchique 71
5.1.1 Le modèle de graphe hiérarchique de Sales-Pardo . . 71
5.1.2 Illustration en calculant toutes les ondelettes 73
5.1.3 Illustration de l'algorithme rapide 73
5.2 Comparaison de mesures de stabilité 75
5.3 Comparaison avec d'autres algorithmes multiéchelles 79
5.3.1 Comparaison sur le modèle de graphes de Sales-Pardo 79
5.3.2 Comparaison sur un autre modèle hiérarchique . . . 83
5.3.3 Laplacien normalisé ou laplacien de marche aléatoire ? 84
5.4 Illustration du test statistique 84
5.5 Illustration sur un modèle de graphe avec une seule échelle . . 85
5.6 Conclusion 89
6 Application à un graphe de terrain 89
7 Définition et utilisation de fonctions d'échelle sur graphe 91
8 Réinterprétation de quelques méthodes multiéchelles 95
8.1 Quelques notations 95
8.2 Forme canonique de modularité filtrée 95
8.3 La modularité filtrée de la méthode de Delvenne et al. 97
8.4 La modularité filtrée de la méthode d'Arenas et al. 98
8.5 La modularité filtrée de la méthode de Reichardt et Bornholdt 100
8.6 La modularité filtrée de la méthode de Ronhovde et Nussinov 101
8.7 Deux nouvelles propositions de modularité filtrée 101
8.8 Les filtres équivalents de la modularité classique 102
8.9 Comparaison et discussion 102
9 Conclusions et perspectives 104
9.1 Conclusions 104
9.2 Perspectives 105
9.2.1 La décomposition en modes empiriques sur graphe . 105
9.2.2 Description multirésolution d'un signal sur graphe . . 106

3 Rééchantillonnage de groupes de nœuds dans un réseau 109
1 Les outils classiques 110
1.1 Le bootstrap classique 110
1.1.1 Estimateurs 111
1.1.2 L'échantillon et l'estimation bootstrap 112
1.1.3 Le bootstrap au service de tests statistiques 112
1.2 Cas moins classique : bootstrap de séries temporelles corrélées 114
2 Une méthode bootstrap pour des groupes de nœuds dans un réseau . 115
2.1 Problème et état de l'art 115
2.2 Méthode 117
2.2.1 Des observables pertinentes pour des groupes de nœuds 117
2.2.2 Rééchantillonnage pour des tests statistiques 118
2.2.3 Obtenir des échantillons bootstrap sous contraintes . 119
2.2.4 Normalisation, test de chaque observable et divergence 120

2.2.5 Deux indicateurs de la taille de l'espace bootstrap . . 121
2.2.6 Compromis entre contraintes et puissance du test . . 122
3 Étude contrôlée sur un modèle de réseaux complexes 123
3.1 Vérification du test statistique 124
3.2 Contrôler la taille de l'espace bootstrap et la puissance du test 124
4 Application à un réseau social 128
4.1 Présentation du jeu de données 129
4.1.1 Méthode expérimentale 130
4.1.2 Dimensionnement du déploiement 131
4.1.3 Distributions classiques 132
4.1.4 Distributions de contacts en fonction du lieu 134
4.2 Application de la méthode de bootstrap sur graphes 134
4.2.1 Choisir les groupes et les hypothèses nulles 135
4.2.2 Contrainte de cardinal 137
4.2.3 Contraintes plus fines 139
5 Discussion 140

Conclusion et perspectives **143**

A Mesures de similarité entre deux partitions **149**

B Modèle de graphe aléatoire : le modèle de Chung-Lu pondéré **151**

C La correction de Bonferroni **153**

D Strip, Bind, Search : Identifying Abnormal Energy Consumption **155**

E Graph Empirical Mode Decomposition **175**

Bibliographie **189**

Table des matières **205**

Résumé – Cette thèse propose de nouveaux outils adaptés à l'analyse des réseaux : sociaux, de transport, de neurones, de protéines, de télécommunications... Ces réseaux, avec l'essor de certaines technologies électroniques, informatiques et mobiles, sont de plus en plus mesurables et mesurés ; la demande d'outils d'analyse assez génériques pour s'appliquer à ces réseaux de natures différentes, assez puissants pour gérer leur grande taille et assez pertinents pour en extraire l'information utile, augmente en conséquence.

Pour répondre à cette demande, une grande communauté de chercheurs de différents horizons scientifiques concentre ses efforts sur l'analyse des graphes, des outils mathématiques modélisant la structure relationnelle des objets d'un réseau. Parmi les directions de recherche envisagées, le traitement du signal sur graphe apporte un éclairage prometteur sur la question : le signal n'est plus défini comme en traitement du signal classique sur une topologie régulière à n dimensions, mais sur une topologie particulière définie par le graphe. Appliquer ces idées nouvelles aux problématiques concrètes d'analyse d'un réseau, c'est ouvrir la voie à une analyse solidement fondée sur la théorie du signal.

C'est précisément autour de cette frontière entre traitement du signal et science des réseaux que s'articule cette thèse, comme l'illustrent ses deux principales contributions. D'abord, une version multiéchelle de détection de communautés dans un réseau est introduite, basée sur la définition récente des ondelettes sur graphe. Puis, inspirée du concept classique de bootstrap, une méthode de rééchantillonnage de graphes est proposée à des fins d'estimation statistique.

Abstract – This thesis describes new tools specifically designed for the analysis of networks such as social, transportation, neuronal, protein, communication networks... These networks, along with the rapid expansion of electronic, IT and mobile technologies are increasingly monitored and measured. Adapted tools of analysis are therefore very much in demand, which need to be universal, powerful, and precise enough to be able to extract useful information from very different possibly large networks.

To this end, a large community of researchers from various disciplines have concentrated their efforts on the analysis of graphs, well define mathematical tools modeling the interconnected structure of networks. Among all the considered directions of research, graph signal processing brings a new and promising vision : a signal is no longer defined on a regular n-dimensional topology, but on a particular topology defined by the graph. To apply these new ideas on the practical problems of network analysis paves the way to an analysis firmly rooted in signal processing theory.

It is precisely this frontier between signal processing and network science that we explore throughout this thesis, as shown by two of its major contributions. Firstly, a multiscale version of community detection in networks is proposed, based on the recent definition of graph wavelets. Then, a network-adapted bootstrap method is introduced, that enables statistical estimation based on carefully designed graph resampling schemes.