université PARIS-SACLAY

Contributions to statistical analysis of graph-structured data

Quelques contributions à l'analyse statistique de données à structure de graphe

Thèse de doctorat de l'université Paris-Saclay

École doctorale de mathématiques Hadamard n°574 (EDMH)
Spécialité de doctorat : Mathématiques appliquées
Graduate School : Mathématiques, Référent : Faculté des sciences d'Orsay

Thèse préparée dans l'unité de recherche Laboratoire de mathématiques d'Orsay (Université Paris-Saclay, CNRS), sous la direction de Pascal MASSART, professeur, et de Frédéric CHAZAL, directeur de recherche.

Thèse soutenue à Paris-Saclay, le 05 Décembre 2022, par

Etienne LASALLE

Composition du jury

Membres du jury avec voix délibérative

Christophe GIRAUD Professeur, Université Paris-Saclay	Président
Aurélie FISCHER Maîtresse de conférence, HDR, Université de Paris	Rapporteur & Examinatrice
Nicolas VERZELEN Chargé de recherche, HDR, INRAE Montpellier	Rapporteur & Examinateur
Sophie DONNET Chargée de recherche, INRAE MIA Paris-Saclay	Examinatrice
Kathryn HESS BELLWALD Professeure, EPFL	Examinatrice

THESE DE DOCTORAT

NNT : 2022UPASM034

université PARIS-SACLAY | ÉCOLE DOCTORALE de mathématiques Hadamard (EDMH)

Titre : Quelques contributions à l'analyse statistique de données à structure de graphe
Mots clés : graphes, statistiques, analyse topologique des données, processus empiriques, diffusion de la chaleur

Résumé : Avec l'augmentation des capacités d'acquisition et de stockage de données, le développement de méthodes efficaces pour le traitement de données à structure de graphe est devenu un point crucial pour les sciences des données. Nous introduisons et étudions de nouvelles méthodes de comparaison de graphes basées sur la diffusion de la chaleur. La nouveauté de notre approche réside essentiellement dans l'introduction du concept de *processus de distances*. Il s'agit de la famille de toutes les distances calculées sur une plage de temps de diffusion pour une paire de graphes donnée. Cela nous permet de développer une analyse multi-échelles des graphes. De plus, en représentant les graphes via des outils issus de l'*analyse topologique des données*, nous sommes en mesure de comparer des graphes de tailles différentes ou non alignés. L'étude des propriétés statistiques de ces processus se fait par la théorie des processus empiriques. Nous prouvons un théorème central limite (TCL) fonctionnel, ainsi qu'un résultat d'approximation gaussienne nous permettant de montrer que la vitesse de convergence dans le TCL est indépendante de la taille des graphes. Ces résultats sont généraux et peuvent être appliqués à d'autres processus. De plus, ils garantissent la validité asymptotique de méthodes de ré-échantillonage pour la construction de bandes de confiance et de tests à deux échantillons permettant de comparer des populations de graphes. Nous étudions les performances de ces tests sur des jeux de données simulés et nous les appliquons au problème de la détection de changement de distribution dans le cadre de l'apprentissage par réseaux de neurones.

Title : Contributions to statistical analysis of graph-structured data
Keywords : graphs, statistics, topological data analysis, empirical processes, heat diffusion

Abstract : With the increase in data acquisition and storage capabilities, developing efficient methods for processing graph-structured data has become a crucial issue in data science. We introduce and study new methods based on heat diffusion to compare graphs. The novelty of our approach essentially lies in the introduction of the concept of *distance processes*, where we consider the family of all distances computed over a continuous range of diffusion times for a given pair of graphs. This allows us to develop a multi-scale analysis of graphs. Moreover, by representing graphs via tools borrowed from *topological data analysis*, we are able to compare graphs of different sizes or unaligned. The statistical properties of these processes are studied with the theory of empirical processes. We prove a functional central limit theorem (CLT), as well as a Gaussian approximation result allowing us to show that the convergence rate in the CLT is independent of the graphs' sizes. These results are general and can be applied to other processes. Moreover, they guarantee the asymptotic validity of resampling methods for constructing confidence bands and two-sample tests comparing graph populations. We study the performance of these tests on simulated data sets and apply them to the problem of distribution shift detection in the context of neural network learning.

Remerciements

Je tiens tout d'abord à remercier mes directeurs, Pascal Massart et Frédéric Chazal, pour m'avoir permis de réaliser cette thèse et pour la confiance et la liberté dont j'ai pu jouir durant ces trois années. Merci pour les discussions enrichissantes que nous avons pu avoir.

Je suis également très reconnaissant envers les membres du jury pour avoir accepté de participer à ma soutenance. Merci à Christophe Giraud, Sophie Donnet, Kathryn Hess, et tout particulièrement à Aurélie Fischer et Nicolas Verzelen pour votre relecture attentive de mon manuscrit.

Je remercie les membres de l'équipe Datashape pour l'intégration dans l'équipe, aussi bien sur le plan scientifique que personnel. Je tiens également à remercier les assistantes de l'équipe, Stéphanie, Bahar, Laurence et Aïssatou, qui ont toujours fait un travail remarquable, dans la bienveillance et la bonne humeur. Merci aux shapers de mes débuts, pour m'avoir guidé et montré la voie. Je pense à Raphaël, Nicolas, Vincent, Martin, et surtout Théo. Merci à ceux qui ont partagé l'expérience de thèse avec moi : Louis, Wojtek, Alex, Vadim, Olympio, Jérémie. Du Magnan au Cesfo en passant par le Crous à 1€, on aura eu de bons moments de rigolade. Merci aux nouveaux de prendre le relais : Bastien, Alexandre, Laure. Une mention spéciale aux grimpeurs : Louis, Wojtek et Felix. Merci pour ces moments hors du temps passés à Bleau.

Merci à Bertrand et Mathieu pour les discussions sympathiques et stimulantes. C'est un réel plaisir de pouvoir partager un projet avec vous.

Merci aux membres de l'équipe Proba-Stats pour votre accueil (Camille !) et pour les bons moments passés aux cafés, au Cesfo, aux restos... Vous m'avez beaucoup appris, que ce soit en séminaire, en pizzama ou de manière informelle. Je citerai en particulier : Nicolas, Alice, Vincent, Perrine, Guillaume, Jérôme, Camille, Olivier, Guillermo, Zacharie, Paul, Jean-Baptiste, Rémi, Elisabeth, Jean-François, Nathanaël, Gilles. Merci pour l'atmosphère agréable que vous créez dans l'équipe et qui donne envie de se lever le matin.

Merci à tous mes amis, de tous horizons, pour m'avoir rendu le chemin un peu moins long, un peu moins dur, un peu plus drôle. Le quinté dans le désordre. Les petits pédestres : Rémi et Thomas. Les champions d'agreg (et du monde !) : Julie, Julien, Arnaud, Christian, Fabien, Laurène, Valentin. Les Calaisiens : Valentin, Em'line, Valentin, Louis, Clément, Lucas, Lorène. Les François Lerouxiens : Guillaume et Amandine. Les Oléronais ou JPSiens (là ya deux écoles) : Luca, Lucile,

Antoine.

Le *groupe de soutien* : où vous placer dans ces remerciements ? Vous êtes à la fois collègues, amis et famille ! Vous m'avez porté à bout de bras à de multiples occasions, encouragé, fait grandir. Je vous en suis à jamais reconnaissant. Vous rencontrer est sans aucun doute la plus belle chance qui m'ait été donnée durant cette thèse. Merci pour les rires, les délires et les souvenirs.

La grande saucisse, mes skis pour ton soutien sans faille. Tu as toujours su m'apporter de la couleur (pêche ?) dans les périodes sombres. J'ai adoré travailler avec toi, les hob, les séances à Rungis... Mes skis pour toutes les belles découvertes : de la planche au surf, en passant par la Meije (oui, je sais, ce n'est pas le chemin le plus court !). J'ai hâte de partager avec toi le prochain p'tit caf' en haut de Cornaf'. Ne change rien, tu es précieuse !

Merci P'pa, merci M'man. Pour tout. Littéralement. Pour votre accompagnement, votre compréhension, votre soutien. Vous avez toujours cru en moi, et m'avez toujours poussé vers la meilleure version possible de moi-même. Je vous dois beaucoup. On aura vécu des moments forts durant ces trois ans, des bons comme des moins bons. Mais on reste soudés. Vous avez réussi "à fabriquer ce manteau, qui nous protège de la vie. / Ce confort impalpable, ce tremplin, cette béquille, / Ce miracle anodin" vous avez fait une famille.

Merci Gaïus pour le modèle que tu as été et est encore pour moi. Malgré la distance, tu as été là quand j'en avais besoin. Toujours de bons conseils, tu as su m'aider à naviguer durant ces trois années. Merci de m'inspirer au quotidien, que ce soit en sciences ou pour le reste.

Merci Riki'. On aura relevé de nombreux défis, toi et moi, durant ces trois ans. Merci de m'avoir accompagné tout du long, merci de m'avoir soutenu même quand pour toi c'était compliqué. Tu es une personne magnifique et je saisis chaque jour la chance de t'avoir à mes côtés. J'ai maintenant vraiment hâte de regarder vers l'avenir et d'envisager la suite avec toi.

J'ai également une pensée pour mes grands-parents, toute ma famille et belle-famille, les amis de la musique de Guînes, ceux du basket d'Andres, tous mes professeurs et de manière générale tous les gens avec qui j'ai partagé un petit bout de vie.

A tous, je vous dois une part de cette humble réussite.

Contents

List of Figures

Chapter 1

1.1 Introduction

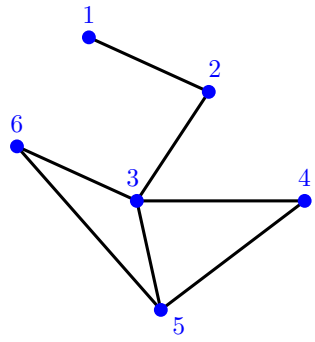
L'un des premiers enjeux en statistiques a été le développement d'outils permettant aux scientifiques de traiter leurs données, notamment lorsqu'elles devenaient trop volumineuses ou trop complexes pour être comprises directement. Les statisticiens ont commencé par étudier des données à valeurs réelles (listes de nombres), puis des données multivariées (tableaux de nombres), et ont continué en s'intéressant à des données de plus en plus complexes : données hétérogènes, images, textes, discours...

Lors de mes précédentes expériences de recherche, j'ai eu l'occasion de travailler sur des problématiques de clustering ainsi que sur de la détection d'anomalies pour des données en lien avec l'analyse topologique des données. Je me suis également intéressé aux propriétés probabilistes de modèles de graphes aléatoires ainsi qu'à celles d'autres structures combinatoires de dimension supérieure, appelées complexes simpliciaux. Cette étude s'est effectuée dans un contexte d'applications aux neurosciences. Elle m'a permis de découvrir la richesse de ces objets combinatoires, la diversité des défis mathématiques qu'ils soulèvent et l'étendue des applications possibles.

Motivés par ces considérations et par l'engouement récent sur ce sujet, nous allons nous concentrer dans cette thèse sur l'étude de données à structure de graphe.

1.1.1 Spécificités des données de graphes

Un graphe est un objet combinatoire qui représente un système d'entités en interaction. Il se définit via son ensemble de sommets (également appelés nœuds) V et son ensemble d'arêtes $E \subset V \times V$. On note $G = (V, E)$. Pour des sommets $u, v \in V$, si $(u, v) \in E$ alors u et v sont connectés dans le graphe, encodant le fait que les entités associées à u et v sont en interaction. Voir Figure 1.1 (gauche) pour un exemple de graphe. Les graphes sont des objets mathématiques puissants permettant de modéliser de nombreuses situations réelles : réseaux sociaux [PBV07, ZAM08], réseaux d'interactions de protéines [BO04, GTH+16], échanges commerciaux [FBA17, BM18], trafic internet [IFM09], activité cérébrale [FZB16, RNS+17], voir Figure 1.2 pour des représentations de tels graphes. En conséquence, l'analyse des données de graphes trouve des applications dans de nombreux domaines. Avec l'augmentation des capa-



$$\begin{array}{c} \\ (1) \\ (2) \\ (3) \\ (4) \\ (5) \\ (6) \end{array} \begin{array}{c} \begin{array}{cccccc} (1) & (2) & (3) & (4) & (5) & (6) \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

FIGURE 1.1 : A gauche, un exemple de graphe non-orienté, avec comme ensemble de sommets $V = \{1, 2, 3, 4, 5, 6\}$ et comme ensemble d'arêtes $E = \{\{1, 2\}, \{2, 3\}, \{3, 4\}, \{3, 5\}, \{3, 6\}, \{4, 5\}, \{5, 6\}\}$. A droite, sa matrice d'adjacence.

cités d'acquisition et de stockage de données, le développement de méthodes efficaces pour traiter les graphes est devenu un point crucial pour les sciences des données.

Comme pour d'autres données complexes, les défis engendrés par l'étude des graphes sont spécifiques à ce type de données. C'est pourquoi des méthodes adaptées doivent être développées. Dans ce qui suit, nous présentons certaines spécificités des graphes ainsi que les problématiques qu'elles mettent en jeux.

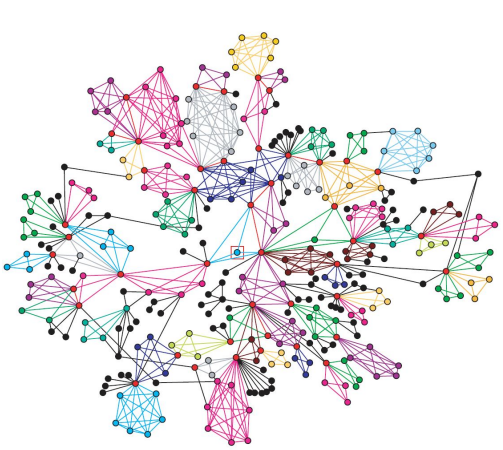
Les graphes sont des objets complexes par nature. En effet, la quantité d'informations requise pour décrire un graphe est quadratique en le nombre de sommets[1]. Pour de grands graphes, l'analyse de toute cette information peut s'avérer laborieuse.

De plus, des structures plus complexes peuvent émerger de cette description locale donnée par l'ensemble des arêtes E. En prenant du recul par rapport à cette description locale et en adoptant un point de vue plus global, on peut capter des informations plus intéressantes, comme par exemple des informations relatives à la connexité du graphe. Les caractéristiques pertinentes d'un graphe peuvent se situer à différentes échelles. Et toutes ces échelles sont parfois nécessaires pour comprendre précisément la structure du graphe. C'est le cas lorsque des structures imbriquées sont présentes par exemple. C'est l'idée de l'analyse multi-échelle.

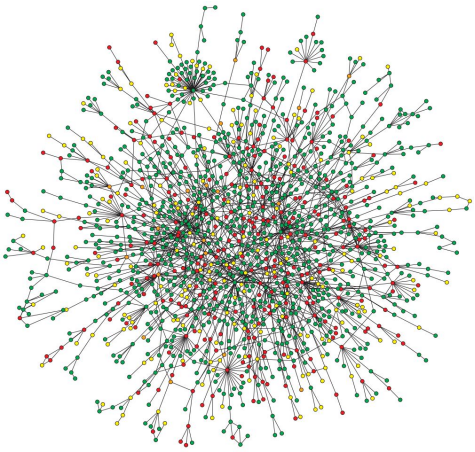
Lorsqu'on travaille sur de l'analyse de graphes, des difficultés peuvent survenir à cause de la diversité des cadres de travail qui existent : graphes pondérés ou non, orientés ou non, alignés ou non, nœuds avec ou sans attributs... Le nombre de combinaisons possibles de ces variations rend difficile la conception de méthodes polyvalentes. De plus, il est souvent nécessaire de développer des méthodes spécifiques au cadre considéré afin d'exploiter au maximum les informations disponibles.

Dans cette thèse, nous considérons des graphes *pondérés*. Pour une paire (u, v) de sommets, au lieu de fournir une information binaire, $(u, v) \in E$ ou $(u, v) \notin E$, nous fournissons un poids $w_{u,v} \in \mathbb{R}$ qui traduit l'intensité de la connexion. Nous interprétons $w_{u,v} = 0$ comme l'absence d'interaction entre u et v. Il est à noter que

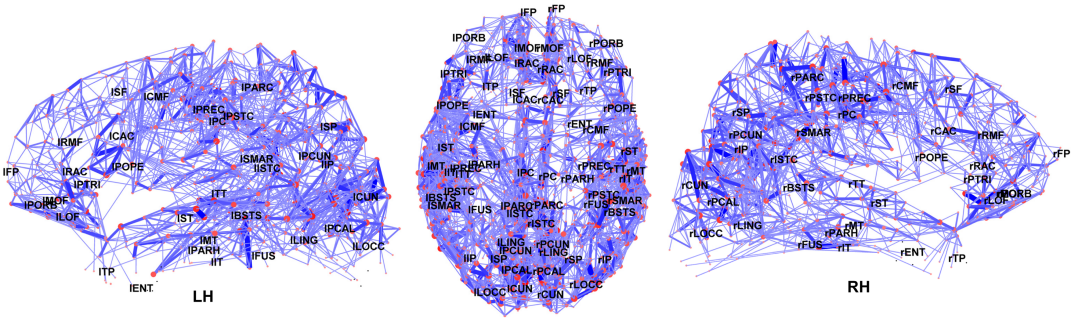
[1]Pour un graphe à n sommets, on a $n(n-1)/2$ arêtes possibles.



(a) Réseau de co-publications.



(b) Réseau d'interactions de protéines.



(c) Réseau du cerveau.

FIGURE 1.2 : Exemples de graphes réels. (a) Un graphe de co-publications, où les nœuds représentent les auteurs et les arêtes sont présentes si les deux auteurs ont une publication en commun. Les couleurs représentent les communautés (sous-graphes denses). Reproduction de [PBV07].
(b) Un graphe représentant les interactions entre les protéines d'une variété de levure, la levure de bière. Les nœuds sont colorés en fonction de l'impact sur la cellule lorsque la protéine est retirée (rouge : létal, vert : non létal, orange : croissance lente, jaune : inconnu). Reproduction de [BO04].
(c) Des graphes représentant la connectivité entre différentes régions du cerveau. La position des nœuds est donnée par la position anatomique des régions. Reproduction de [HCG+08].

les graphes non-pondérés sont un cas particulier des graphes pondérés, où les poids sont dans $\{0, 1\}$. D'après la description faite ci-dessus, il paraitrait naturel que les poids soient définis sur l'ensemble des réels positifs ou nuls $\mathbb{R}_+$. L'exemple qui suit propose la construction d'un graphe pondéré où les poids sont dans $\mathbb{R}$ et non $\mathbb{R}_+$. Il fournit ainsi une intuition sur le sens possible des poids négatifs dans un graphe pondéré.

Example 1.1 (Graphe avec des poids négatifs)**.** Considérons n variables aléatoires à valeurs réelles $X_1, \ldots, X_n$, potentiellement dépendantes et pouvant provenir de distributions différentes. Nous considérons le graphe pondéré tel que l'ensemble de sommets est $\{1, \ldots, n\}$ et pour toute paire de sommets (i, j), $w_{i,j} = \mathrm{Corr}(X_i, X_j)$, où $\mathrm{Corr}(X_i, X_j)$ désigne la corrélation de Pearson[2] entre X_i et X_j. Dans cet exemple, un poids nul signifie que les deux variables ne sont pas corrélées, tandis qu'un poids positif ou négatif traduit une corrélation positive ou négative, respectivement.

Une autre distinction courante provient du caractère symétrique ou asymétrique des connexions entre les sommets. On parle alors de graphes *orientés* ou *non-orientés*. Les représentations matricielles des graphes orientés sont souvent plus difficiles à manipuler, car les matrices ne sont pas symétriques. En particulier, il n'est pas toujours possible d'obtenir des décompositions spectrales. Par souci de simplicité, nous ne considérerons par la suite que des graphes non-orientés.

Une autre difficulté lors de l'étude de données de graphes vient du choix de leur représentation. En effet, les graphes peuvent être encodés de diverses façons, chacune capturant des caractéristiques différentes, ce qui rend leur étude est à la fois intéressante et complexe. Ils contiennent des informations d'une grande richesse, mais qui se retrouvent parfois cachées et difficiles à extraire. Afin de capturer les propriétés adaptées à la tâche à accomplir, il est nécessaire de regarder les graphes sous le bon angle, c'est-à-dire avec la bonne représentation.

Une représentation particulièrement classique est la *matrice d'adjacence* (ou *matrice de poids* dans le cas de graphes pondérés). Pour un graphe de taille n, quitte à numéroter les sommets, on peut considérer que $V = \{1, \ldots, n\}$. Le graphe peut alors être encodé par sa matrice d'adjacence $A \in \mathbb{R}^{n \times n}$, définie comme la matrice binaire où pour tous les $i, j \in V$,

$$A_{i,j} = \begin{cases} 1 & \text{si } (i, j) \in E, \\ 0 & \text{sinon.} \end{cases}$$

Dans le cas des graphes pondérés, la matrice des poids est la matrice W de taille $n \times n$ telle que $W_{i,j} = w_{i,j}$. Les matrices d'adjacence et de poids sont des constructions canoniques, dans le sens où elles encodent les graphes de manière univoque, à une numérotation des nœuds près. Ce sont les représentations matricielles les plus basiques. Elles sont souvent utilisées pour fournir une description brute et complète des graphes. La Figure 1.1 (à droite) donne un exemple de matrice d'adjacence.

[2]Pour deux variables aléatoires réelles X et Y, $\mathrm{Corr}(X, Y) := \mathrm{Cov}(X, Y)/(\sigma_X \sigma_Y)$ où $\mathrm{Cov}(X, Y)$ désigne la covariance et σ_X, σ_Y sont les écarts types de X et Y.

Remarquons que la matrice d'adjacence ou de poids d'un graphe non-orienté est symétrique.

Pour un graphe pondéré avec un ensemble de sommets $V = \{1, \ldots, n\}$, le *degré* du sommet $i \in V$ correspond au poids total des arêtes incidentes à i. Il est noté d_i, tel que $d_i = \sum_j W_{i,j}$. La matrice des degrés D est la matrice diagonale dont les coefficients diagonaux sont les degrés : $D_{i,i} = d_i$ pour tout sommet i. On peut alors définir d'autres représentations matricielles également très employées : les laplaciens. Le laplacien combinatoire (ou non normalisé) est défini par $L = D - W$. Le laplacien normalisé symétrique est défini par $\tilde{L} = I - D^{-1/2}WD^{-1/2}$ où I est la matrice identité. Bien que les laplaciens encodent également les graphes de manière univoque, comme les matrices d'adjacence ou de poids, leur définition est moins intuitive. Une manière de mieux les comprendre est d'adopter le point de vue du traitement du signal sur les graphes. Dans ce cadre, les laplaciens sont vus comme des opérateurs différentiels. A la manière d'une base de Fourier dans le cadre des fonctions périodiques, les vecteurs propres des laplaciens permettent de décomposer et de régulariser les signaux sur les graphes.

D'un point de vue mathématique et théorique, les matrices sont un moyen pratique de représenter les graphes. En effet, ce sont des outils mathématiques bien connus, que nous savons caractériser et étudier. De plus, en tant qu'opérateurs, les matrices d'adjacence et les laplaciens contiennent des informations pertinentes, notamment via leurs spectres. Cependant, d'un point de vue computationnel, les représentations matricielles ne sont pas forcément économes en mémoire, particulièrement lorsque les graphes sont parcimonieux[3].

Au lieu de fournir une description exhaustive des graphes, comme le font les représentations matricielles, nous pouvons utiliser des résumés structurels. Ces derniers se concentrent sur des caractéristiques spécifiques des structures de graphes. Il peut s'agir de statistiques simples comme le nombre d'arêtes, la distribution des degrés ou le nombre de petits sous-graphes donnés (appelés *graphlets*). Il peut également s'agir d'indices plus complexes, comme les coefficients de *clustering* ou la connectivité algébrique. Ces représentations sont pratiques pour classer les graphes en catégories[4]. Cependant, elles sont parfois un peu grossières et peuvent omettre des informations fines sur les structures de graphes.

Une autre façon de représenter les graphes consiste à les plonger dans d'autres espaces, mieux connus et plus pratiques à manipuler. L'idée est de disposer d'une application allant d'un espace de graphes dans un autre espace de nature différente. Cette application peut être explicite, par exemple en concaténant des résumés structurels dans un vecteur. Elle peut également être implicite. Les méthodes à noyau, par exemple, envoient implicitement les données qu'elles traitent dans un espace de Hilbert, généralement abstrait. Les plongements explicites peuvent être utilisés pour de la visualisation, tandis que ceux implicites sont souvent utilisés pour des tâches statistiques plus complexes, comme des problèmes de partitionnement (*clustering*

[3]Les graphes parcimonieux sont des graphes possédant un nombre d'arêtes significativement plus petit que $|V|^2$, l'ordre de grandeur du nombre d'arêtes possibles.

[4]Par exemple, on appelle graphes *scale-free*, les graphes dont la distribution des degrés suit une loi de puissance.

en anglais) ou de classification.

Pour des données complexes, il est en général crucial de disposer de notions pertinentes de distance. Avec les graphes, des soucis liés à l'identification des sommets peuvent émerger. Considérons, par exemple, deux graphes de même taille. Pour pouvoir les comparer, de nombreuses méthodes requièrent la connaissance d'une correspondance entre les sommets des deux graphes. Si cette correspondance est connue, les sommets peuvent être numérotés de manière cohérente. On dit que les graphes sont alignés. En revanche, lorsque ce n'est pas le cas, des problèmes en apparence simples peuvent s'avérer compliqués. C'est le cas du problème d'isomorphisme de graphes (ou d'exact *matching*), où l'on cherche à déterminer si deux graphes sont identiques ou non. Dans le cas général, l'existence d'un algorithme de complexité polynomiale capable de résoudre le problème reste inconnue. Dans le même ordre d'idées, comparer des graphes de tailles différentes semble encore plus compliqué, puisqu'aucune correspondance totale entre les sommets ne peut exister.

Lorsque l'on fait des statistiques sur des données de graphes, il existe deux types d'asymptotiques. D'un part, on peut considérer un modèle de graphes et étudier les propriétés des graphes qu'il génère, lorsque le nombre de sommets tend vers l'infini. Nous pouvons voir dans cette approche un parallèle avec létude asymptotique de processus ponctuels, où le nombre de points tend vers l'infini. Ce type d'asymptotique est utile pour analyser les grands graphes. D'autre part, supposons qu'un échantillon de graphes de tailles raisonnables nous est fourni. Par exemple, nous pouvons penser à des copies *i.i.d.* d'un graphe aléatoire suivant une distribution donnée. Dans ce cas, l'asymptotique concerne plutôt le nombre de graphes. Nous pouvons donc étudier les propriétés statistiques de l'échantillon et donc de la loi de probabilité sous-jacente. C'est alors la taille de l'échantillon qui tend vers l'infini et non plus celle des graphes.

Ces deux situations sont très différentes. Dans la première, les nœuds sont les objets d'intérêt, et la structure du graphe vient comme une information complémentaire. Dans la seconde, les échantillons de graphes constituent l'objet d'étude principal et nous cherchons à en apprendre plus sur eux et la loi de probabilité les ayant générés.

1.1.2 Un aperçu des méthodes statistiques sur des graphes

Dans cette section, nous proposons un aperçu non exhaustif de travaux existants, ciblant un ou plusieurs des défis mentionnés ci-dessus. Nous mentionnerons dans un premier temps quelques contributions à l'étude de grands graphes, avant de détailler différentes approches sur l'étude d'échantillons de graphes. Nous nous concentrerons plus particulièrement sur les notions de distances et sur les méthodes prenant en compte la topologie des graphes. Pour les lecteurs intéressés, chaque paragraphe fournira des références complémentaires.

Un grand graphe. En analyse de graphes, les statisticiens se sont beaucoup penchés sur l'étude des jeux de données composés d'un seul grand graphe. Pour les grands graphes, il peut être difficile de visualiser ou d'interpréter leur structure. Afin de réduire la complexité de telles données et d'en extraire une information plus exploitable, l'une des tâches classiques est le problème de la *détection de communautés*. L'objectif est de partitionner les sommets, c'est-à-dire de les séparer en groupes, de sorte que les sommets de chaque groupe soient similaires en termes de connectivité. Prenons l'exemple classique d'un graphe représentant un réseau social où les personnes sont liées en fonction de leur amitié. Le problème de la détection de communautés vise à identifier les groupes d'amis, de sorte que les personnes à l'intérieur de chaque groupe soient bien connectées, et que les personnes entre les groupes soient moins connectées.

Plusieurs solutions ont été proposées à ce problème. Certaines utilisent les éléments propres de représentations matricielles [Fie73, DH03, VL07]. Ces approches sont connues sous le nom de *clustering spectral*. Elles ont fait l'objet d'analyses théoriques [VLBB08, RCY+11]. D'autres ont proposé des approches basées sur des modèles probabilistes. Dans ce cas, il est supposé que le graphe provient d'un modèle paramétrique (par exemple le modèle à blocs stochastiques) et une procédure d'estimation des paramètres est mise en place. Cette dernière est souvent basée sur de la maximisation de vraisemblance [NS01, DPR08, LBA12].

Dans ce cadre d'étude avec un grand graphe, la plupart des résultats théoriques sont asymptotiques en la taille du graphe.

Jeux de données de graphes. Plutôt que d'être constituées d'un seul grand graphe, les données peuvent parfois se présenter sous la forme de jeux de données de plusieurs graphes. Nous pouvons penser, par exemple, à des graphes représentant les activités cérébrales d'une cohorte de patients [RVM+16, FKL19].

Un autre exemple, que nous exploiterons par la suite, est celui des ensembles de graphes d'activation de réseaux de neurones. Dans le domaine de l'apprentissage statistique, les réseaux de neurones sont des outils ayant gagné en popularité ces derniers temps, grâce à leurs excellentes performances faisant référence sur de nombreux problèmes. Ils sont dotés d'une structure inhérente de graphe, souvent utilisée pour la représentation (voir Figure 1.3). Certains travaux proposent d'aller plus loin, et d'exploiter cette structure de graphes pour étudier le réseau de neurones. [GS17] et [GSH19] ont introduit les graphes d'activation. Ce sont des graphes pondérés associés aux données d'entrée. Chaque graphe d'activation représente la façon dont le réseau de neurones a traité une certaine donnée. Les ensembles de graphes d'activation ont été utilisés pour étudier les phases d'apprentissage par [RTB+18] et pour mesurer l'incertitude des prédictions par [LIC+21].

Dans ce cadre des jeux de données de plusieurs graphes, les problématiques sont généralement des tâches statistiques standards telles que des tests statistiques, du *clustering*, de la classification ou de la régression. Pour aborder ces questions, diverses solutions ont été proposées.

L'une d'entre elles consiste à utiliser une classe de méthodes appelées méthodes à noyau. Essentiellement, ces méthodes utilisent un classifieur linéaire sur des pro-

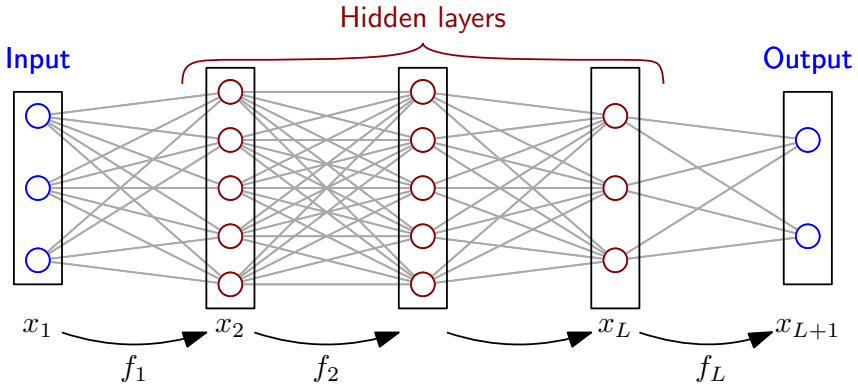


FIGURE 1.3 : Représentation sous forme de graphe d'un réseau de neurones dense.

blèmes non-linéaires. L'astuce consiste à définir un noyau sur l'espace des données, c'est-à-dire une fonction qui mesure la similarité entre données et qui correspond à un produit scalaire dans un espace de Hilbert abstrait. De nombreuses méthodes à noyau existent sur divers problèmes statistiques. Pour les appliquer sur des données de graphes, il suffit de leur fournir un noyau pertinent. Cependant, cette étape est loin d'être triviale. Elle constitue même tout l'enjeu des méthodes à noyau. La plupart des noyaux de graphes se concentrent soit sur des sous-structures locales (comptage de *graphlets* [SVP+09, KM12], de sous-arbres [GFW03, MV09], etc...) ou sur des structures globales (marches aléatoires [GFW03, VSKB10], plus courts chemins [BK05]). Par conséquent, peu de noyaux captent les caractéristiques multi-échelles des graphes. Toutefois, des noyaux multi-échelles ont été introduits plus récemment [SSVL+11, KP16]. De plus, les noyaux sont généralement restreints à certains types de graphes. Certains ne sont définis que lorsque les nœuds sont labellisés ou ont des attributs, la plupart ne s'appliquent pas aux graphes pondérés, etc...

Suite au succès des réseaux de neurones sur d'autres types de données (les images, par exemple), des réseaux de neurones pour graphes (GNN[5]) ont également été proposés pour traiter des jeux de données de graphes. Les premiers GNN étaient des variations de l'algorithme de *message passing* dans lequel des représentations des nœuds sont apprises en propageant des informations le long des arêtes [GMS05, SGT+08, GM10]. Ensuite, motivés par le succès des réseaux de neurones convolutifs pour les images, des variations pour les graphes ont commencé à apparaître. Au lieu de faire des convolutions sur des grilles, comme c'est le cas pour les images, les convolutions ont été redéfinies pour utiliser la structure des graphes [BZSL14, AT16, NAK16, GSR+17]. Toutes ces méthodes sont généralement conçues pour effectuer des tâches d'apprentissage spécifiques, mais fournissent par ailleurs des représentations (souvent en grande dimension) des sommets. Ces approches sont essentiellement fondées sur des études numériques. Pour plus de détails sur les GNN, nous renvoyons le lecteur à l'étude faite dans [WPC+20].

[5]Pour *graph neural network* en anglais.

Un besoin de comparer les graphes. Une attention particulière a été consacrée à la conception de notions de distance entre graphes. En effet, dans les travaux que nous avons mentionnés ci-dessus, beaucoup de techniques (comme les méthodes à noyau) se basent sur des comparaisons de graphes. La conception de ces distances est fortement contrainte par le cadre d'étude. En particulier, différents types d'informations peuvent être utilisés selon que les graphes sont orientés ou non, pondérés ou non, de même taille ou non. Un autre facteur clé pour définir des distances est de savoir si une correspondance entre les sommets des graphes (NC[6]) est connue ou non. Dans le cas d'une NC connue, nous pouvons considérer que les graphes sont définis sur le même ensemble de sommets et les comparaisons peuvent être faites à l'échelle des arêtes. Dans ce contexte, diverses métriques ont été définies en comparant les matrices d'adjacence, les laplaciens, les noyaux de la chaleur [HGJ13] et d'autres matrices [KVF13] dont les entrées représentent des quantités associées aux paires de sommets. D'autre part, lorsque aucune NC n'est disponible, les graphes sont souvent comparés à une échelle mésoscopique ou macroscopique, à l'aide de résumés structurels. Des statistiques globales sur les graphes ont été proposées, comme la distribution des degrés, le diamètre des graphes ou le coefficient de clustering [PCJ04]. Une autre approche bien étudiée consiste à considérer des *graphlets* [PCJ04], c'est-à-dire des petits sous-graphes que l'on dénombre dans les graphes. Diverses méthodes ont été développées pour comparer les nombres de graphlets [Pržo7, YMDD+14, ARR+14, FNC+17]. Certains travaux ont également proposé d'exploiter l'information structurelle portée par les spectres d'opérateurs [WZ08, GAC+18].

Les mêmes remarques que pour les noyaux peuvent être faites sur les méthodes de comparaison de graphes. Généralement, elles ne sont pas multi-échelles ou sont conçues pour des types de graphes spécifiques. De plus, l'alignement des graphes est une condition nécessaire pour appliquer certaines méthodes.

Pour plus de détails sur les comparaisons de graphes, nous renvoyons le lecteur à [SERG14], [ESDS16], [TITP19] et aux références qui s'y trouvent.

Analyse topologique des données pour les graphes. L'analyse topologique des données (TDA[7]) est un domaine récent qui vise à extraire des informations de nature topologique sur données complexes. Elle se base sur la théorie de l'*homologie persistante*. Fondamentalement, elle enregistre l'apparition et la disparition de composantes topologiques (composantes connexes, cycles, cavités et trous de dimension supérieure) le long d'une famille d'espaces topologiques emboîtés. Cet outil est classiquement appliqué aux nuages de points. Pour un nuage de points donné, nous construisons une filtration, c'est-à-dire une famille emboitée de complexes simpliciaux (ensemble de sommets, d'arêtes, de triangles et de simplexes de dimension supérieure) supportés par le nuage de points[8]. Pour chaque composante topologique, nous encodons sa naissance b et sa mort d le long de la filtration, comme un point de coordonnées $(b, d) \in \mathbb{R}^2$. Le multi-ensemble de points qui en résulte est appelé un

[6]Pour *node correspondence* en anglais.
[7]Pour *topological data analysis* en anglais.
[8]Les sommets des complexes simpliciaux correspondent aux points du nuage.

diagramme de persistance et joue le rôle d'un descripteur de la topologie du nuage de points.

Des approches semblables ont été proposées pour appliquer ce cadre aux graphes. L'idée est similaire : construire une famille de complexes simpliciaux par-dessus le graphe. Cette fois, les complexes simpliciaux sont supportés non seulement par les sommets, mais aussi par les arêtes du graphe. Le *flag complex* [ABM05] est un exemple classique de telle construction. Des travaux empiriques utilisent cette approche pour effectuer des analyses de la topologie des graphes [RNS+17, KGB19, LGSL20].

D'autres approches sont basées sur la théorie de l'*homologie persistante étendue*. L'idée est de regarder deux familles croissantes de sous-graphes : les sous-graphes des sous-niveaux et les sous-graphes des sur-niveaux donnés par une fonction sur les sommets à valeur réelle. Une fois encore, on enregistre les différents niveaux d'apparition et de disparition des composantes connexes et des cycles dans ces deux familles et nous les encodons dans un diagramme de persistance. Cette approche a été suivie par [ZW19, CCI+20, ZYCW20].

La théorie de l'homologie persistante s'est avérée pertinente dans l'analyse de données complexes[9] grâce à l'extraction d'informations sur les composantes topologiques de ces données, de manière multi-échelle et robuste. Des travaux récents soutiennent l'idée que la TDA devrait également être capable de capturer des informations pertinentes dans les données de graphes. Nous souhaitons nous appuyer sur ces idées pour proposer des méthodes d'analyse de données de graphes disposant de garanties statistiques.

1.1.3 Contributions

Parmi les travaux présentés ci-dessus, retenons deux lignes directrices essentielles. D'une part, les données composées d'un seul grand graphe ont été bien étudiées. Des solutions spécifiques aux graphes et souvent supportées par des résultats théoriques ont été apportées. D'autre part, dans le cas de jeux de données contenant plusieurs graphes, des outils prometteurs ont été développés : de l'introduction de diverses notions de distance, aux réseaux de neurones pour graphes, en passant par des idées issues de l'analyse topologique des données. La plupart d'entre eux sont étayés par des résultats empiriques et pourraient servir de base à des méthodes plus avancées et disposant de garanties statistiques.

Motivé par le bilan ci-dessus, l'objectif principal de cette thèse peut être résumé comme suit :

> Fournir de nouvelles méthodes pour analyser des jeux de données de graphes, statistiquement fondées et prenant en compte les spécificités des graphes.

[9] Par exemple : formes 3D [TMB14], images [QTT+19, RYB+20], séries temporelles [SDB16, Ume17].

Nous explorons de nouveaux outils de comparaison de graphes. Leur analyse théorique nous permet de concevoir des tests à deux échantillons consistants pour les données de graphes. En guise d'application, nous étudions le problème de la détection de changements de distribution dans le contexte de l'apprentissage par réseaux de neurones.

Résumé des contributions. L'objectif de cette thèse est de fournir des outils statistiquement fondés pour analyser des jeux de données de graphes. Nous nous concentrons sur le cas des graphes pondérés non-orientés. Pour cela, nous introduisons et étudions de nouvelles méthodes de comparaison de graphes basées sur la diffusion de la chaleur. La nouveauté de notre approche ne réside pas spécialement dans la définition de nouvelles métriques ou semi-métriques d_t basées sur la diffusion de chaleur, mais plutôt dans l'introduction du concept de *processus de distances*, où pour une paire de graphes (G, G') nous considérons la famille de distances sur une plage de temps de diffusion $\{d_t(G, G'),\ t \in [0, T]\}$, pour un certain $T > 0$. De plus, en représentant les graphes par des *diagrammes de persistance* (outils issus de l'analyse topologique des données), nous sommes en mesure de comparer des graphes de tailles différentes ou non alignés.

L'étude de ces processus de distances nous amène à la théorie des processus empiriques. Pour des distributions de graphes pondérés de tailles bornées et à poids positifs bornés, nous prouvons un théorème central limite fonctionnel pour nos processus. Nous le raffinons en montrant que la vitesse de convergence est en fait indépendante de la taille des graphes. De plus, bien que les preuves s'appuient sur des résultats standards sur les processus empiriques, la formulation de nos résultats est simple et générale. Ils pourraient facilement être appliqués à d'autres processus de distances ou même à d'autres processus en général.

Ce travail sur les processus de distances nous mène au développement de méthodes d'analyse de jeux de données de graphes théoriquement fondées. Plus précisément, nous proposons des tests à deux échantillons pour des données de graphes consistants. Nous étudions leurs performances sur des jeux de données simulées, et montrons que leurs comportements sur des échantillons de tailles finies sont en accord avec nos résultats asymptotiques. De plus, nous considérons le problème de la détection de changements de distribution dans le cadre de l'apprentissage par réseaux de neurones. L'idée est d'être en mesure d'alerter l'utilisateur d'un réseau de neurones, lorsque ce dernier propose de mauvaises prédictions sur un nouveau jeu de données dont la distribution est différente de celle des jeux d'entraînement et de test. Nous proposons d'utiliser nos tests à deux échantillons sur les familles de graphes d'activation. Notre approche a l'avantage d'être applicable quelle que soit la nature des données d'entrée. En outre, nous montrons que les graphes d'activation couplés à nos tests à deux échantillons capturent avec précision le comportement des réseaux de neurones et sont capables de détecter leurs changements de comportement. Par rapport aux autres méthodes, notre approche n'est pas la plus sensible aux changements de distribution, mais elle est en accord avec la robustesse ou la sensibilité du réseau face à ces changements. Elle pourrait permettre d'éviter d'alerter l'utilisateur

pour des changements de distributions très bien gérés par le réseau de neurones.

Distances basées sur la diffusion de la chaleur. Nous analysons des processus de distances définis à partir de deux distances basées sur la diffusion de la chaleur, à savoir la *heat kernel distance* et la *heat persistence distance*.

Pour un graphe pondéré non orienté G de taille n, soit L son laplacien non normalisé. Le *noyau de chaleur* (*heat kernel* en anglais) au temps $t \geq 0$ de G est la matrice symétrique e^{-tL}. Elle fournit l'ensemble des solutions à l'équation de la chaleur $\partial_t u_t = -Lu_t$, où $u_t \in \mathbb{R}^n$ est le vecteur représentant les températures des sommets dans le graphe. Par conséquent, e^{-tL} encode la façon dont la chaleur s'est diffusée après un temps de diffusion t.

Considérons deux graphes G et G' pour lesquels la correspondance entre leurs sommets est connue. Supposons donc que les sommets sont numérotés afin d'aligner les graphes. Ainsi, nous pouvons comparer leurs noyaux de la chaleur en utilisant n'importe quelle norme matricielle. Nous choisissons la norme de Frobenius $\|\cdot\|_F$ et définissons la *heat kernel distance* (HKD) par

$$D_t((G, G')) = \|e^{-tL} - e^{-tL'}\|_F, \tag{1.1}$$

où L et L' sont les laplaciens de G et G' respectivement[10]. Pour capturer des informations à différentes échelles, nous pouvons faire varier le temps de diffusion. Dans le but de conserver toutes ces informations, nous introduisons la notion de processus de distances. Ainsi, pour un certain $T > 0$, nous considérons la famille $\{D_t((G, G')),\ t \in [0, T]\}$ de toutes les heat kernel distances, et l'appelons le processus HKD.

Soient deux graphes G et G', éventuellement de tailles différentes ou non alignés. Nous suivons l'approche de [CCI+20] qui vise à comparer G et G' via leurs diagrammes de persistance. Ces derniers encodent la structure topologique des graphes, étant donné une fonction filtre basée sur de la diffusion de la chaleur. Considérons la *heat kernel signature* (HKS) définie, pour un graphe $G = (V, E)$ et son laplacien L, comme la fonction $h_t(G) : V \ni i \mapsto (e^{-tL})_{i,i}$. Cette fonction est utilisée pour calculer quatre types de diagrammes de persistance, caractérisant quatre types de composantes topologiques que nous pouvons interpréter comme : des branches descendantes, des branches ascendantes, des composantes connexes et des cycles[11]. Chaque diagramme de persistance est génériquement dénoté par $Dg(G, h_t(G))$. Ces diagrammes peuvent être comparés en utilisant la *distance Bottleneck* d_B, une sorte de distance de transport dans l'espace des diagrammes de persistance. Par conséquent, nous définissons la *heat persistence distance* (HPD) par

$$H_t((G, G')) = \max_{Dg} d_B(Dg(G, h_t(G)), Dg(G', h_t(G'))),$$

[10] Les doubles parenthèses sont insérées dans (1.1) pour correspondre au cadre des processus empiriques, à savoir la paire de graphes (G, G') est évaluée par la fonction $D_t(\cdot)$.

[11] L'orientation haut/bas est prise par rapport à $h_t(G)$, comme si les nœuds de G étaient placés dans un espace où l'axe vertical représente les valeurs de $h_t(G)$.

où le maximum est pris sur les quatre types de diagrammes[12]. Encore une fois, pour considérer tous les filtres HKS possibles, nous étudions le processus HPD défini comme $\{H_t((G, G')),\ t \in [0, T]\}$..

Pour des graphes aléatoires G et G', on peut voir les processus HKD et HPD comme l'évaluation de la paire aléatoire (G, G') par les familles de fonctions $\mathcal{F}_{HKD} = \{D_t,\ t \in [0, T]\}$ et $\mathcal{F}_{HPD} = \{H_t,\ t \in [0, T]\}$ respectivement. Ceci correspond exactement à la notion de processus empirique. C'est à travers ce cadre que nous analysons les processus HKD et HPD.

Résultats sur les processus empiriques. Pour analyser les processus de distances définis ci-dessus, nous tirons parti de la théorie déjà établie des processus empiriques. La première étape de notre analyse consiste à dériver des résultats statistiques sur des processus empiriques lipschitziens généraux. Ensuite, nous montrons que ces résultats s'appliquent à nos processus HKD et HPD.

Considérons un espace de probabilité $\mathcal{X}$, équipé d'une mesure de probabilité P et d'une famille $\mathcal{F}$ de fonctions mesurables $f_t : \mathcal{X} \to \mathbb{R}$ indexée par $[0, T]$. Nous montrons le résultat suivant.

Théorème 1.1. *Supposons qu'il existe $k > 0$ et $M > 0$ tels que pour tout $x \in X$, la fonction $t \mapsto f_t(x)$ est k-Lipschitzienne et bornée par M. Alors pour un échantillon $X_1, \ldots, X_N$ tiré sous P, le processus suivant*

$$\{G_N f_t,\ t \in [0, T]\} := \left\{ \sqrt{N} \left(\frac{1}{N} \sum_{i=1}^{N} f_t(X_i) - \mathbb{E}_P \left[f_t(X) \right] \right),\ t \in [0, T] \right\}$$

converge dans un certain sens vers un processus gaussien $\mathbb{G} = \{G_t,\ t \in [0, T]\}$.

Des compléments sur le type de convergence et la définition des processus gaussiens se trouvent en Section 2.3. Ce théorème constitue une version fonctionnelle du théorème central limite. Par la suite, il sera fondamental pour apporter des garanties asymptotiques à des méthodes de rééchantillonnage.

Pour obtenir des informations sur la vitesse de convergence dans le Théorème 1.1, nous prouvons un résultat d'approximation gaussienne.

Théorème 1.2. *Pour n'importe quel entier N, on pose $r_N = N^{-1/7} \log(N)^{9/14}$. Alors sous les mêmes hypothèses que le théorème précédent, pour tout $\lambda > 1$, il existe une constante ρ et un rang N_0 tels que pour tout $N \geq N_0$, on peut construire sur le même espace de probabilité à la fois l'échantillon $X_1, \ldots, X_N$ et une version $\mathbb{G}^{(N)}$ du processus gaussien limite $\mathbb{G}$ tels qu'ils vérifient*

$$\mathbb{P} \left(\sup_t \left| G_N f_t - \mathbb{G}_t^{(N)} \right| > \rho . r_N \right) \leq N^{-\lambda}.$$

[12] Même remarque sur les doubles parenthèses de $H_t((G, G'))$, comme dans (1.1).

La conséquence importante de ce théorème est que le taux de convergence r_N ne dépend pas des paramètres du problème : la constante de Lipschitz k et la borne M. Ceci est encourageant car appliqué aux processus HKD et HPD, ce résultat signifiera que la vitesse de convergence vers le processus gaussien sera indépendante de la taille des graphes.

Théorème 1.1 et Théorème 1.2 proviennent de résultats déjà établis et plus généraux. Cependant, le fait de travailler dans le cadre spécifique de processus continus indexés par un paramètre réel permet à nos résultats d'avoir des énoncés simples. Les hypothèses requises pour les appliquer sont assez élémentaires, car les processus doivent essentiellement être uniformément lipschitziens. En prenant pour $\mathcal{X}$ des espaces spécifiques de paires de graphes pondérés, nous montrons que nous pouvons appliquer le Théorème 1.1 et le Théorème 1.2 à $\mathcal{F}_{HKD}$ et $\mathcal{F}_{HPD}$. De plus, nous pensons que ces théorèmes pourraient facilement être appliqués à d'autres processus.

Par ailleurs, nous affinons le résultat du Théorème 1.2 en rendant explicite la dépendance de ρ et N_0 par rapport à k et M, et montrons qu'elle est polynomiale. Ce résultat, couplé au fait que le taux r_N est indépendant de k et M, offre de bons indices sur le fait que les méthodes basées sur l'approximation gaussienne fonctionneront bien en pratique, même dans le cadre d'échantillons de tailles finies.

Applications statistiques aux échantillons de paires de graphes. Pour appliquer le Théorème 1.1 aux processus HKD et HPD, la manière la plus naturelle est de considérer des échantillons de paires de graphes. En faisant cela, nous obtenons des garanties asymptotiques sur la construction de bandes de confiance et de tests à deux échantillons, en utilisant des méthodes de rééchantillonnage (aussi appelées méthodes *bootstrap*). Considérons un échantillon de paires de graphes $(G_1, G'_1), \ldots, (G_N, G'_N)$. Nous voulons déterminer la largeur de la bande de confiance autour du processus empirique moyen $\{P_N d_t, t \in [0,T]\} := \{N^{-1}\sum_i d_t(G_i, G'_i), t \in [0,T]\}$, où d_t est soit la HKD, soit la HPD. Ceci peut être fait en effectuant plusieurs rééchantillonnages avec remise $((\hat{G}_1^{(b)}, \hat{G}_1'^{(b)}), \ldots, (\hat{G}_N^{(b)}, \hat{G}_N'^{(b)}))_{1\leq b\leq B}$. D'après le Théorème 1.1, on peut montrer que les processus bootstrap

$$\left(\left\{\sqrt{N}\left(\frac{1}{N}\sum_i d_t(\hat{G}_i^{(b)}, \hat{G}_i'^{(b)}) - \frac{1}{N}\sum_i d_t(G_i, G'_i)\right),\ t \in [0,T]\right\}\right)_{1\leq b\leq B} \tag{1.2}$$

sont presque distribués comme le processus gaussien limite. Par conséquent, en calculant $\hat{c}$, un quantile supérieur empirique du supremum sur t des processus dans (1.2), $N^{-1/2}\hat{c}$ fournit une largeur pour la bande de confiance qui sera valide asymptotiquement.

De manière similaire, nous pouvons construire un test à deux échantillons pour des échantillons de paires de graphes $((G_{1,1}, G'_{1,1}), \ldots, (G_{1,N}, G'_{1,N}))$ et $((G_{2,1}, G'_{2,1}), \ldots, (G_{2,M}, G'_{2,M}))$. On désigne par P_1 et P_2 les lois de probabilité générant les échantillons. Nous voulons effectuer un test qui distingue l'hypothèse nulle $H_0 : P_1 = P_2$ de l'hypothèse alternative $H_1 : P_1 \neq P_2$. Pour ce faire, nous proposons un test basé sur la comparaison des moyennes empiriques des processus de distances, en

calculant la statistique de test T suivante :

$$T = \sqrt{\frac{NM}{N+M}} \sup_{t\in[0,T]} \left| \frac{1}{N}\sum_i d_t(G_{1,i}, G'_{1,i}) - \frac{1}{M}\sum_j d_t(G_{2,j}, G'_{2,j}) \right|. \tag{1.3}$$

Comme pour le calcul de la largeur de la bande de confiance, nous pouvons calculer des équivalents bootstrap $T^{(1)}, \dots, T^{(B)}$ de la statistique de test. Pour n'importe quel b, la statistique bootstrap $T^{(b)}$ est calculée en utilisant (1.3), mais appliquée à des rééchantillons $((\hat{G}^{(b)}_{1,1}, \hat{G}'^{(b)}_{1,1}), \dots, (\hat{G}^{(b)}_{1,N}, \hat{G}'^{(b)}_{1,N}))$ et $((\hat{G}^{(b)}_{2,1}, \hat{G}'^{(b)}_{2,1}), \dots, (\hat{G}^{(b)}_{2,M}, \hat{G}'^{(b)}_{2,M}))$ tirés avec remise depuis l'échantillon concaténé $((G_{1,1}, G'_{1,1}), \dots, (G_{1,N}, G'_{1,N}), (G_{2,1}, G'_{2,1}), \dots, (G_{2,M}, G'_{2,M}))$. Les valeurs $(T^{(b)})_{1\le b\le B}$ fournissent une bonne estimation de la distribution de la statistique de test sous l'hypothèse nulle. Par conséquent, leur quantile empirique supérieur $\tilde{c}_\alpha$ de niveau α peut être utilisé comme seuil de rejet. Si $T > \tilde{c}_\alpha$, on rejette H_0, si $T \le \tilde{c}_\alpha$ on conserve H_0. Grâce au Théorème 1.1, cette procédure assure un contrôle asymptotique du niveau et de la puissance du test. En effet, désignons par $\hat{c}_\alpha$ le quantile supérieur (non empirique) de niveau α de la variable aléatoire $T^{(1)}$, et rappelons que $\tilde{c}_\alpha$ est une bonne approximation de $\hat{c}_\alpha$ pour B suffisamment grand. Alors, nous avons le résultat suivant.

Théorème 1.3. *Soient P_1 et P_2 des distributions de paires de graphes pondérés, tels que les poids sont positifs et bornés. Nous supposons que la taille des graphes est bornée et que les tailles des échantillons, M et N, tendent vers l'infini à des vitesses similaires*[13]*. Alors le test qui rejette H_0 dès que $T > \hat{c}_\alpha$ est consistant. Cela signifie que le niveau asymptotique du test est de α.*

$$\mathbb{P}_{H_0}\left(T > \hat{c}_\alpha\right) \underset{M,N\to\infty}{\longrightarrow} \alpha.$$

De plus, pour toute hypothèse alternative vérifiant, pour un $t \in [0,T]$, $\mathbb{E}_{P_1}\left[d_t(G,G')\right] \neq \mathbb{E}_{P_2}\left[d_t(G,G')\right]$, on a

$$\mathbb{P}\left(T > \hat{c}_\alpha\right) \underset{M,N\to\infty}{\longrightarrow} 1,$$

ce qui nous assure un contrôle de la puissance asymptotique du test.

Applications statistiques à des échantillons de graphes. Les échantillons de paires de graphes ne sont pas le type de données de graphes le plus fréquemment rencontré. Par conséquent, nous proposons une modification de la procédure de test à deux échantillons afin de pouvoir effectuer des tests pour des échantillons de graphes et non de paires de graphes. L'idée est d'ajouter une étape d'appariement aléatoire, où nous créons trois ensembles de paires : des paires constituées d'un graphe de chaque échantillon (paires inter-échantillons), et des paires du même échantillon, soit du premier, soit du second (paires intra-échantillon). Pour préserver les propriétés

[13] Ici, "des vitesses similaires" signifie que $N/(N+M) \to \lambda \in (0,1)$.

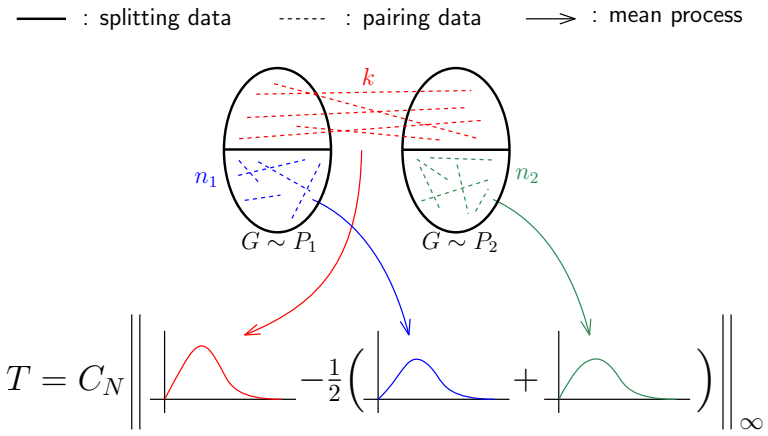


FIGURE 1.4 : Illustration de l'étape d'appariement pour les tests à deux échantillons sur des graphes.

d'indépendance, nous nous assurons que chaque graphe apparaît dans au plus une paire. Ensuite, nous comparons la moyenne des processus de distances calculée sur les paires inter-échantillons aux moyennes calculées sur les paires intra-échantillons (voir Figure 1.4). Un seuil de rejet est calculé par une méthode bootstrap similaire à la procédure pour les échantillons de paires de graphes.

Nous proposons également deux variantes de ce test. La première consiste à inclure des informations supplémentaires lors de l'appariement des graphes. Nous considérons la situation où les graphes sont labellisés : soit parce qu'ils proviennent de données de classification, soit parce qu'un classificateur est disponible. Dans ce cas, au lieu d'apparier les graphes complètement au hasard, nous pouvons les apparier de manière à faire correspondre les labels au sein des paires. Ainsi, les processus de distances comparent désormais des paires de graphes ayant systématiquement le même label, donc des graphes plus similaires. Cela se traduit souvent par une réduction de la variance des processus de distances et donc par une augmentation de la puissance de test. La deuxième variation découle de l'observation que, comme chaque graphe apparaît dans *au plus* une paire, toutes les paires ne peuvent pas être considérées. Par conséquent, l'étape d'appariement ne capture pas toute l'information possible. Nous proposons une approche par tests multiples dans laquelle nous répétons l'étape d'appariement plusieurs fois, nous calculons les p-valeurs associées à chaque test, et nous appliquons une procédure de Benjamini-Hochberg pour finaliser le test. Dans notre cas, bien que les tests ne soient pas indépendants, nous obtenons sur nos simulations un bon contrôle du niveau du test. La procédure de Benjamini-Yekutieli peut également être mise en oeuvre. Bien que plus conservative, elle assure un contrôle théorique du niveau du test, sans hypothèse sur la dépendance des statistiques de test.

Simulations. Toutes ces méthodes conçues pour analyser et comparer des échantillons de graphes ou des paires de graphes sont analysées dans différents contextes dans la partie II. Nous évaluons d'abord la construction de bandes de confiance et de tests à deux échantillons pour des paires de graphes tirés sous différents modèles

de graphes aléatoires (modèle d'Erdős-Rényi (ER), modèle à blocs stochastiques, modèles aléatoires de graphes géométriques). Les expériences confirment le résultat théorique concernant le contrôle asymptotique du niveau et de la puissance même pour des échantillons de quelques centaines de paires de graphes. Pour les modèles ER avec différents paramètres, les calculs analytiques sont réalisables. Par conséquent, nous calculons le seuil déterminant le moment où le test optimal de Neymann-Pearson parvient à faire la distinction entre deux modèles d'ER avec des paramètres différents qui dépendent de la taille des échantillons. Les expériences montrent que notre test à deux échantillons basé sur la HKD peut faire la distinction entre les deux modèles même en travaillant près de la transition de phase, ce qui indique que les performances du test sont satisfaisantes en termes de puissance.

Applications à la détection de changements de distribution. Le reste de la partie II est consacré à la contribution des tests à deux échantillons basés sur la HKD au problème de la détection de changements de distribution dans le contexte de l'apprentissage par réseau de neurones. Nous proposons d'utiliser les graphes d'activation comme données d'entrée de nos tests. En testant sur les graphes d'activation, nous testons si la façon dont les données d'entrée sont traitées par le réseau a changé.

Nous évaluons notre test à deux échantillons basé sur la HKD et ses variantes (appariement par labels et tests multiples) sur plusieurs jeux de données et architectures de réseaux de neurones. Nous avons d'abord mené des expériences sur le jeu de données d'images de chiffres MNIST [LCB10] avec deux architectures de réseaux de neurones de faible complexité : un réseau dense et un réseau convolutif. Ensuite, nous avons considéré des données et des architectures plus complexes en étudiant des données de nuages de points traitées par un réseau de neurones appelé *Ripsnet* [dSHC+22]. Le réseau est conçu pour renvoyer des estimations robustes de descripteurs topologiques de nuages de points appelés *images de persistance*. Dans ces deux applications, nous comparons nos méthodes à la procédure BBSD[14] [LWS18] qui est une méthode de détection de changements de distribution basée sur une procédure de tests multiples utilisant les valeurs de sortie du réseau. Nous comparons également nos méthodes à un test similaire au nôtre. Il utilise les graphes d'activation et la *diffusion distance* [HGJ13] définie comme $\max_{t\in[0,T]} D_t(G, G')$ au lieu du processus HKD. Par rapport à notre méthode, cette alternative calcule le maximum de chaque processus de distances avant de calculer une statistique de test basée sur ces maxima.

En tant que tests à deux échantillons, nos méthodes sont généralement moins puissantes que BBSD. Cela signifie que même pour de légers changements, BBSD a un meilleur taux de détection que nos méthodes. Cependant, dans le contexte de la détection de changements de distribution, BBSD est en fait trop sensible et renverrait des signaux d'alerte même pour des changements de distribution n'ayant qu'un faible impact sur les performances du réseau. D'autre part, il ressort des expériences que la méthode basée sur la *diffusion distance* n'a pas assez de puissance pour détecter les changements avant que la précision du réseau ne chute de manière

[14] *Black Box Shift Detection* en anglais.

significative, ce qui n'est pas le cas pour nos méthodes. Nous expliquons ce phénomène en observant que le temps de diffusion le plus discriminant pour comparer les échantillons n'est pas nécessairement proche des temps de diffusion les plus discriminants utilisés pour comparer les graphes de chaque paire. Au final, nos méthodes basées sur les processus HKD sont performantes dans le contexte de la détection des changements de distribution.

Les conclusions de cette série d'expériences sont doubles. Premièrement, elle démontre l'utilité de considérer les processus de distance au lieu de la *diffusion distance*. Deuxièmement, elle montre que les graphes d'activation sont de bons descripteurs pour surveiller le comportement des réseaux de neurones et que la méthode basée sur le processus HKD que nous avons introduite ainsi que ses variations sont bien adaptées pour analyser ces descripteurs.

1.1.4 Plan

Chapter 2: Background. Cette thèse tire parti de plusieurs domaines, allant de la théorie des graphes, à l'analyse topologique des données, en passant par les statistiques et l'apprentissage par réseaux de neurones. Ce chapitre introduit ces notions et privilégie la compréhension du lecteur plutôt que l'exhaustivité. Le chapitre commence par une introduction à la théorie des graphes et accorde une attention particulière aux considérations spectrales. Il est suivi d'une présentation de l'analyse topologique des données, du cadre classique et des alternatives pour les graphes. Ensuite, nous présentons les définitions et les résultats classiques des processus empiriques. Enfin, nous introduisons les réseaux de neurones, plus précisément les réseaux de neurones denses et convolutifs. Nous définissons également des descripteurs du comportement des réseaux de neurones, appelés graphes d'activation.

Part I – Theory –

Chapter 3: Multi-scale comparisons of graphs. Dans ce chapitre, nous présentons de nouveaux outils de comparaison pour les graphes. Nous définissons d'abord la *heat kernel distance* (HKD) et la *heat persistence distance* (HPD), en nous appuyant sur les travaux de [HGJ13] et [CCI+20]. Nous relions ces choix à des arguments spectraux et à des considérations spécifiques aux graphes avant de présenter la contribution principale de cette thèse. Nous introduisons le nouveau concept de *processus de distances* : une famille de distances (*par exemple*, HKD ou HPD) indexée par une gamme de paramètres d'échelle.

Chapter 4: Statistical properties of real-valued continuous empirical processes indexed by a real parameter. Ce chapitre développe les résultats théoriques nécessaires à l'analyse de nos processus de distances. La théorie classique des processus empiriques en est le support. Nous dérivons des résultats spécifiques pour les processus empiriques continus indexés par un paramètre réel. Ces résultats sont simples à énoncer et facilement généralisables à d'autres processus. Ils fournissent des garanties asymptotiques qui étayent nos méthodes d'analyse de graphes. De plus,

des études fines de la vitesse de convergence du théorème central limite fonctionnel soutiennent la validité de ces méthodes même dans le cadre d'échantillons de tailles finies.

Chapter 5: Application to confidence bands and two-sample tests for graphs. En s'appuyant sur les résultats du chapitre précédent, nous proposons ici des méthodes pour construire des bandes de confiance consistantes et des tests à deux échantillons consistants pour des données de graphes. Des alternatives au test à deux échantillons sont proposées pour pouvoir traiter des échantillons de graphes et non de paires de graphes. On propose aussi des variations permettant d'augmenter la puissance du test. Elles mettent en place des schémas d'appariement et des tests multiples.

Chapter 6: Appendix. Les preuves du Chapitre 4 et Chapitre 5 sont reportées dans ce chapitre.

Part II – Numerical Illustrations –

Chapter 7: From generative random graph models. Ce chapitre est consacré à l'étude empirique des tests à deux échantillons basés sur la HKD et HPD sur des paires de graphes. Nous menons des expériences sur des modèles de graphes aléatoires et évaluons les performances des tests en estimant leurs niveaux et leurs puissances.

Chapter 8: Distribution shift detection from activation graphs in neural network learning. Dans ce dernier chapitre, nous abordons le problème de la détection des mauvaises performances des réseaux de neurones lorsqu'on leur présente de nouvelles données issues de nouvelles distributions. Nous proposons d'appliquer notre test à deux échantillons aux graphes d'activation du nouveau jeu de données et du jeu de données test. Nous comparons cette méthode à d'autres méthodes de détection et confirmons que les graphes d'activation sont des descripteurs pertinents pour saisir les changements de comportement des réseaux de neurones. De plus, nos tests à deux échantillons détectent les changements avec une meilleure corrélation avec les mauvaises performances du réseau, par rapport aux autres méthodes qui sont soit trop sensibles, soit pas assez puissantes.

Les travaux des chapitres 3-7 ont fait l'objet d'une soumission [Las21].

ENGLISH VERSION

1.2 Introduction

One of the first concerns in statistics was to develop tools allowing scientists to analyze their data, especially when they were becoming too big or too complex to be understood directly. Statisticians started by studying real-valued data (lists of numbers), then multivariate data (tables of numbers), and continued by studying more complex data: heterogeneous data, images, texts, or speeches.

In my previous research experiences, I had the opportunity to work on clustering problems as well as anomaly detection for data related to topological data analysis. I also got interested in the probabilistic properties of random graph models and other higher dimensional combinatorial structures, namely simplicial complexes. This study was done in the context of neurosciences. It allowed me to discover the richness of these combinatorial objects, the diversity of the mathematical challenges they raise, and the extent of the potential applications.

Motivated by all these considerations and the recent enthusiasm around this subject, we focus in this thesis on the study of graph-structured data.

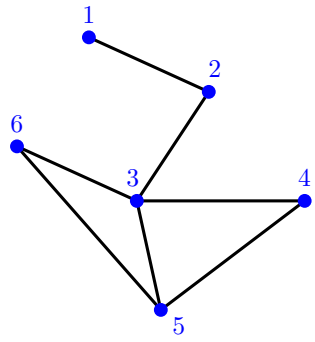
1.2.1 Specificity of graph data

A graph is a combinatorial object that represents a system of interacting entities. It is formally defined through its set of vertices (also called nodes) V and its set of edges $E \subset V \times V$. We note $G = (V, E)$. For vertices $u, v \in V$, if $(u, v) \in E$ then u and v are connected in the graph; encoding the fact that the entities relative to u and v are interacting. See Figure 1.5 (left) for an example of a graph. Graphs are powerful mathematical objects that model various real-life situations: social networks [PBV07, ZAM08], protein-protein interaction networks [BO04, GTH+16], trade markets [FBA17, BM18], web traffic [IFM09], brain activities [FZB16, RNS+17]; see Figure 1.6 for representations of such graphs. As a consequence, graph data analysis finds applications in many areas. With the increase in data acquisition and storage capacities, developing efficient analysis methods for graph data is becoming a pivotal point in data sciences.

Like other complex data, graph data come with their own challenges. That is why dedicated methods need to be developed. In the following, we present some of the specific features and challenges that come with graph data.

Graphs are inherently complex objects. The amount of information required to describe a graph is quadratic in the number of vertices[15]. For large graphs, analyzing all this information can be cumbersome. Additionally, more global structures can arise from the local description given by the edge set E. By looking at a bigger picture, we might capture more meaningful and relevant graph characteristics, for example information about connectedness. Indeed, pertinent graph features might be located at different scales. And all these scales are sometimes necessary to accurately understand the graph structure, *e.g.,* in the presence of nested structures.

[15]For a graph with n vertices, there are $n(n-1)/2$ possible edges.



$$\begin{array}{c} \\ (1) \\ (2) \\ (3) \\ (4) \\ (5) \\ (6) \end{array} \begin{array}{c} \begin{array}{cccccc} (1) & (2) & (3) & (4) & (5) & (6) \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

Figure 1.5: On the left, an example of an undirected graph with vertex set $V = \{1, 2, 3, 4, 5, 6\}$ and edge set $E = \{\{1, 2\}, \{2, 3\}, \{3, 4\}, \{3, 5\}, \{3, 6\}, \{4, 5\}, \{5, 6\}\}$. On the right, its adjacency matrix.

This is the idea of multi-scale analysis.

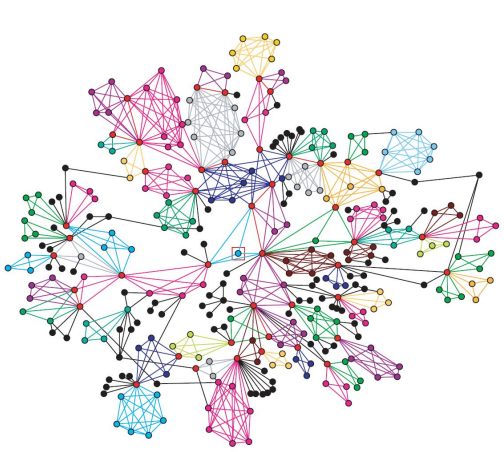
When working on graph analysis, difficulties can arise because of the various frameworks that exist: weighted or unweighted, directed or undirected, aligned or unaligned, nodes with or without attributes... The number of combinations of these variations makes the design of broadly applicable methods challenging. Furthermore, framework-specific methods often need to be developed to benefit from the available information.

In this thesis, we will consider *weighted* graphs. For a pair (u, v) of vertices, instead of a binary information, either $(u, v) \in E$ or $(u, v) \notin E$, we provide a weight $w_{u,v} \in \mathbb{R}$ that accounts for the strength of the connection. We interpret $w_{u,v} = 0$ as the absence of interaction between u and v. Note that unweighted graphs are a special case of weighted graphs where weights are in $\{0, 1\}$. From the description above, it would seem natural for weights to be defined on the set of non-negative real numbers $\mathbb{R}_+$. The following example constructs a weighted graph where weights are in $\mathbb{R}$ and not $\mathbb{R}_+$. It provides an intuition about the possible meaning of negative weights in a weighted graph.

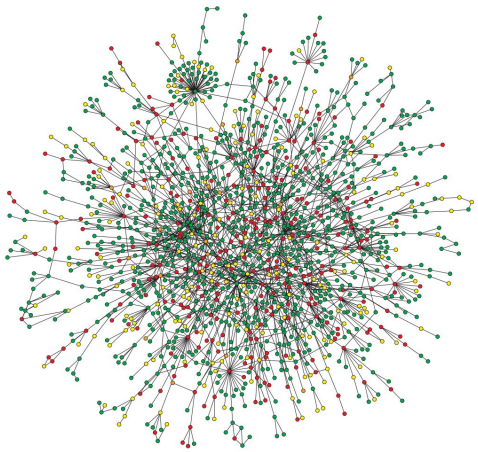
Example 1.2 (Graph with negative weights)**.** Consider any n real-valued random variables $X_1, \ldots, X_n$, potentially dependent and potentially having different distributions. We can construct the weighted graph with vertex set $\{1, \ldots, n\}$ and set for any pair (i, j), $w_{i,j} = \mathrm{Corr}(X_i, X_j)$, where $\mathrm{Corr}(X_i, X_j)$ denotes the Pearson correlation[16] between X_i and X_j. In this case, a zero weight means that the two variables are uncorrelated, while a positive or negative weight means a positive or negative correlation, respectively.

Another common distinction is whether connections are symmetric or asymmetric. We say that graphs are either *undirected* or *directed*. Matrix representations of directed graphs are often more difficult to handle as the matrices are not symmetric.

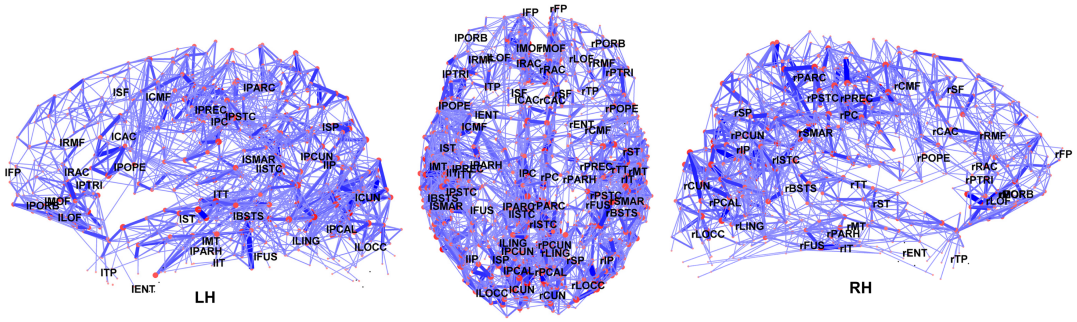
[16]Let X and Y be two real-valued random variables, then $\mathrm{Corr}(X, Y) := \mathrm{Cov}(X, Y)/(\sigma_X \sigma_Y)$ where $\mathrm{Cov}(X, Y)$ is the covariance and σ_X, σ_Y are the standard deviations of X and Y.



(a) Co-authorship network.



(b) Protein-protein interaction network.



(c) Brain network.

Figure 1.6: Examples of real-world graphs. (a) A co-authorship graph, where nodes represent authors and edges are present if both authors have a publication in common. Colors represent communities (densely connected subgraphs). Reproduction of [PBV07].
(b) A graph representing the interactions between proteins for a variety of yeast, the brewer's yeast. Node are colored according to the impact on the cell when the protein is removed (red: lethal, green: non-lethal, orange: slow-growth, yellow: unknown). Reproduction of [BO04].
(c) Graphs representing the connectivity between brain regions. Node positions are given by anatomical locations. Reproduction of [HCG+08].

In particular, spectral analysis becomes more complicated, as spectral decompositions do not always exist. For simplicity, in the following, we will only consider undirected graphs.

Another difficulty when studying graph data comes from the choice of their representation. It is because graphs can be encoded in various ways, each capturing different characteristics, that their study is both interesting and complex. They contain rich information that is sometimes hidden and difficult to extract. In order to capture the graph properties that are appropriate to the task at hand, it is necessary to look at them with the right lens, *i.e.,* with the right representation.

A very classical representation is the *adjacency matrix* (or *weight matrix* in the case of weighted graphs). For a graph of size n, up to numbering the vertices, we can consider V to be $\{1, \dots, n\}$. Then, the graph can be encoded through its adjacency matrix $A \in \mathbb{R}^{n \times n}$, defined as the binary matrix where for all $i, j \in V$,

$$A_{i,j} = \begin{cases} 1 & \text{if } (i,j) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

In the case of weighted graphs, the weight matrix is the $n \times n$ matrix W such that $W_{i,j} = w_{i,j}$. Adjacency and weight matrices are quite canonical in the sense that they encode graphs in a one-to-one manner, given a numbering of the nodes. They are the most basic matrix representations and thus, they are often used to describe graphs in a simple and faithful manner. Figure 1.5 (right) provides an example of an adjacency matrix. Remark that the adjacency or weight matrix of an undirected graph is symmetric.

For a weighted graph with vertex set $V = \{1, \dots, n\}$, the *degree* of a vertex $i \in V$ is the total weight of incident edges of i. It is denoted by d_i such that $d_i = \sum_j W_{i,j}$. The degree matrix D is the diagonal matrix whose diagonal coefficients are the degrees: $D_{i,i} = d_i$ for all vertices i. From there, we can define other well-used matrix representations: the Laplacian matrices. The combinatorial (or unnormalized) Laplacian matrix is defined as $L = D - W$. The symmetric normalized Laplacian matrix is defined as $\tilde{L} = I - D^{-1/2} W D^{-1/2}$ where I is the identity matrix. While Laplacian matrices also encode graphs in a one-to-one manner, similarly to adjacency and weight matrices, their definition is less intuitive. One way to better understand them is to take the point of view of signal processing on graphs. In this context, Laplacian matrices act as differential operators. Similarly to a Fourier basis for periodic functions, Laplacian eigenvectors allow to decompose and regularize signals on graphs.

From a mathematical and theoretical point of view, matrices are a convenient way to represent graphs, as they are well known mathematical objects, that we learned to manipulate and study. Furthermore, seen as operators, adjacency and Laplacian matrices carry relevant information about the graphs they encode, especially in their spectra. On the other hand, from a computational point of view, matrix representations might not be the most memory-efficient, especially for sparse graphs[17].

[17]Sparse graphs are graphs with a significantly smaller number of edges than $|V|^2$, the order of

Instead of providing an exhaustive description of graphs, as matrix representations do, we can use structural summaries to represent graphs. They focus on specific characteristics of the structures. They can be simple statistics like the number of edges, the distribution of the degrees, or counts of small given subgraphs. It can also be more complex indices, like clustering coefficients or the algebraic connectivity. These representations are practical to classify graphs in categories[18]. However, they might miss some critical information when it comes to capturing subtle variations in graph structures.

Another way to represent graphs is to embed them into other spaces that are more common and practical. The idea is to define a mapping from a graph space to another space. Such mapping can be explicit, for example, by concatenating numerical structural summaries into a vector. It can also be implicit. For example, when considering kernel methods, data are implicitly embedded into an abstract Hilbert space. Explicit embeddings[19] can be used for visualization and data exploration, while implicit ones are often used for more complex statistical tasks.

For complex data, it is crucial to design meaningful notions of distance. When working with data sets of graphs, some issues related to identifying nodes can arise. Consider two graphs of the same size. In order to compare them, most methods require knowing the correspondence between the nodes of the two graphs. When it is the case, we can number the nodes of both graphs in a coherent manner. We say the the graphs are *aligned*. Otherwise, when the correspondence is unknown, even for an apparently simple task such as deciding whether the graphs are the same, the problem is known to be difficult. This is the *graph isomorphism* problem or *exact graph matching* problem. In the general case, it is unknown whether an algorithm with polynomial complexity can solve it. In the same spirit, comparing graphs of different sizes can be difficult, because there exists no node correspondence.

When doing statistics on graph data, there are two types of asymptotics. First, we can consider a generative model of graphs and study the properties of the graphs it generates when the number of vertices tends to infinity. We can see in this approach a parallel with asymptotic studies of point processes, where the number of points tends to infinity. This type of asymptotic is helpful when analyzing single big graphs. On the other hand, assume that we are given a sample of graphs of reasonable sizes. For example, we can think about *i.i.d.* copies of a random graph following a given distribution. In that case, the asymptotic concerns the number of graphs. We can therefore study the statistical properties of the sample and therefore the ones of the underlying probability distribution. It is then the sample size that tends to infinity and not the graph size anymore.

These two settings are very different. In the first one, nodes are the objects of interest, and the graph structure comes as complementary information. In the second one, the samples of graphs are the main interest, and we try to learn more

magnitude of the number of possible edges.

[18]For example, graphs whose degree distribution follows a power law are called *scale-free*.

[19]Eventually coupled with dimension reduction techniques.

about them and the probability distributions that generated them.

1.2.2 An overview of statistical methods on graphs

This section proposes a non-exhaustive overview of existing works targeting some of the above challenges. We first mention some contributions to the study of large graphs before detailing different approaches to the study of graph samples. We will focus more particularly on the notions of distances and on methods taking into account the topology of graphs. For the interested reader, each paragraph will provide complementary references.

One big graph. One subject that has received a lot of attention is the study of data sets composed of a single large graph. For large graphs, it can be really difficult to visualize or interpret their structure. In order to reduce the complexity of such data and to extract workable information, one of the classical tasks is the problem of *community detection*. The objective is to cluster the vertices, *i.e.* separate them into groups such that vertices inside each group are similar in terms of their connectivity properties. A classical example is a graph representing a social networks where people are linked according to their friendship. The problem of community detection aims at identifying groups of friends, so that people inside each group are well connected, and people between groups are less connected.

Several solutions have been proposed to this problem. Some proposed to used the spectra and eigenvectors of matrix representations of the graph [Fie73, DH03, VL07], this approach in known as *spectral clustering*. It made the object of theoretical analyses [VLBB08, RCY+11]. Others proposed model-based approaches. Parametric models are assumed[20], and procedures to estimate the parameters, often using maximum likelihood estimation, are developed [NS01, DPR08, LBA12].

In this one-large-graph setting, most theoretical results are asymptotic with respect to the graph size.

Data set of graphs. Rather than a single large graph, data can sometimes be in the form of data sets of multiple graphs. For example, we can think of graphs representing the cerebral activities of a cohort of patients [RVM+16, FKL19].

Another example, which we will exploit later, is that of sets of activation graphs of neural networks. Neural networks are machine learning tools that have received a lot of attention lately, mainly due to their state-of-the-art performances on numerous problems. They come with an inherent graph structure that is used a lot for representation (see Figure 1.7). Some works also propose to use this graph structure to study neural networks. [GS17] and [GSH19] introduced activation graphs. These are weighted graphs associated to the input data. Each activation graph represents how the neural network processed a certain input.

In this framework with data sets of several graphs, the tasks are often standard statistical problems, like statistical testing, clustering, classification or regression. To tackle these questions, various solutions have been proposed.

[20] *e.g.,* the stochastic block model.

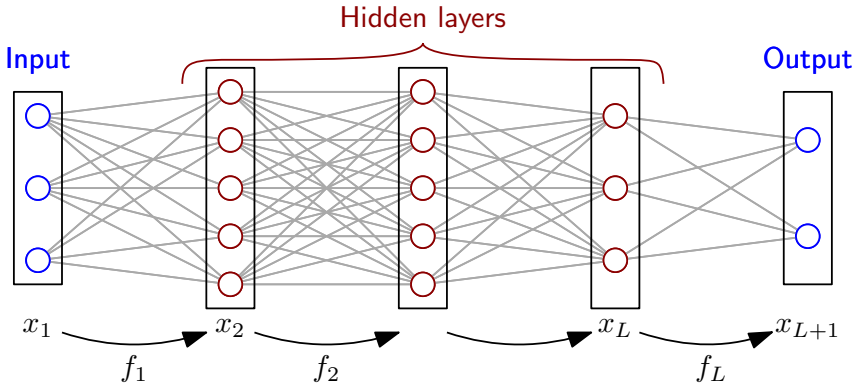


Figure 1.7: Graph representation of a dense neural network.

Essentially, these methods use a linear classifier on non-linear problems. The trick is to define a kernel on the data space, i.e., a function that measures the similarity between data and corresponds to a scalar product in an abstract Hilbert space. Many kernel methods exist for various statistical problems. To apply them to graph data, we only need to provide them with a relevant kernel. However, this step is far from trivial and is actually the main challenge of kernel methods. Most graph kernels focus either only on local substructures (counting small subgraphs [SVP+09, KM12], subtrees [GFW03, MV09], etc...) or on global structures (random walks [GFW03, VSKB10], shortest paths [BK05]). As a result, they usually miss the multi-scale graph characteristics, although more recent works have introduced multi-scale kernels [SSVL+11, KP16]. Another problem with this approach is that graph kernels are usually restricted to certain types of graphs. Some are only defined when nodes have labels or attributes, most of them do not apply to weighted graphs, etc...

Following the success of neural networks on other types of data (e.g., images), graph neural networks (GNNs) have also been proposed to process graph data sets. The first GNNs were variations of the *message passing* algorithm were node representations are learned by propagating information along the edges [GMS05, SGT+08, GM10]. Then, motivated by the success of convolutional neural networks for images, variations for graphs started to appear. Instead of doing convolutions on grids, as it is done for images, people tried to redefine convolution by exploiting graph structures [BZSL14, AT16, NAK16, GSR+17]. These methods are usually designed to perform specific learning tasks but also provide high-level representations of the nodes. These approaches are essentially supported by numerical evidences. For more details on GNNs, we refer the reader to the survey of [WPC+20].

A need to compare graphs. Particular attention has been devoted to designing notions of distance between graphs. Indeed, in the works we mentioned above, many techniques (like kernel methods) revolve around comparing graphs. The working framework highly constrains the design of these distances. In particular, different types of information can be used depending on whether the graphs are directed or undirected, weighted or unweighted, or have the same size or not. Another crit-

ical factor in defining distances is whether a node correspondence (NC) is known or not. In the case of a known NC, we can consider the graphs to be defined on the same vertex set and comparisons can be made at the edge scale. In this context, people have applied various metrics to compare adjacency matrices, Laplacian matrices, heat kernels [HGJ13] and other matrices [KVF13] whose entries represent quantities associated with pairs of nodes. On the other hand, when no NC is available, graphs are often compared at a mesoscopic or macroscopic level using structural summaries. People have used global statistics on graphs like degree distributions, network diameters, or clustering coefficients [PCJ04]. Another well studied approach is to consider graphlets [PCJ04], *i.e.*, small given subgraphs that are counted in graphs. Then, various methods have been developed to compare the graphlet counts [Pržo7, YMDD+14, ARR+14, FNC+17]. Some work has also been pursued to exploit the structural information carried by spectra of operators [WZ08, GAC+18].

The same comments on graph kernels can be made for comparison methods. They are often not multi-scale or are designed for specific types of graphs. Moreover, graph alignment is an essential requirement for some methods.

For more references on comparisons of graphs, we refer the reader to [SERG14], [ESDS16], [TITP19] and references therein.

Topological data analysis for graphs. Topological data analysis (TDA) is a recent fields that aims at extracting topological information out of complex data. It is based on the theory of *persistent homology*. Basically it records the birth and death of topological features (connected components, cycles, cavities and higher-dimensional holes) along a family of nested topological spaces. This tool is classically applied to point clouds. For a given point cloud, we build a filtration. That is a nested family of simplicial complexes (set of vertices, edges, triangles, and higher-dimensional simplices) supported on the point cloud[21]. And for each topological feature, we encode its birth b and death d as a point of coordinates $(b, d) \in \mathbb{R}^2$. The resulting multi-set of points is called a *persistent diagram* and plays the role of a topological summary of the point cloud.

Similar approaches have been proposed to apply this framework to graphs. The idea is similar: building a family of simplicial complexes on top of the graph. This time, the simplicial are supported not only by the vertices but also by the edges of the graph. The *flag complex* [ABM05] is a classic example of such a construction. Empirical works have used this approach to perform topological analysis of graphs [RNS+17, KGB19, LGSL20].

Other approaches are based on the theory of *extended persistent homology*. The idea is to look at two families of increasing subgraphs : the sublevel subgraphs, and the superlevel subgraph given by a real-valued function on the vertices. Once again, we record various levels of appearances and disappearances of connected components and cycles in theses families and encode them in a persistent diagram. This approach has been followed by [ZW19, CCI+20, ZYCW20].

[21] The vertices of the simplicial complexes correspond to the point of the point clouds.

The theory of persistent homology has been proved powerful in the analysis of complex data[22] by extracting topological features in a multi-scale manner. Recent works support the idea that TDA should be able to capture relevant information in graph data and could help perform meaningful analysis. We would like to build on these existing works to provide TDA-flavored graph data analysis methods that come with statistical guarantees.

1.2.3 Contributions

From the works presented above, we recall two essential lines of work. On the one hand, data composed of a single large graph have been well studied, with graph-specific solutions and theoretical results. On the other hand, promising tools have been developed for data sets containing several graphs: from various distances, to graph neural networks, to ideas from topological data analysis. Most of them are supported by empirical results and could be a basis for more advanced and statistically founded methods.

Motivated by the above overview, the main objective of this work can be summarized as follows:

> Providing new graph-specific and statistically founded analysis methods for data sets of graphs.

More precisely, we explore new tools to compare graphs and prove theoretical results that allow us to design consistent two-sample tests for graph data. As an application, we investigate the problem of distribution shift detection in the context of neural network learning.

A summary of the contributions. This thesis aims to provide statistically founded tools to analyze graph data sets. We focus on the case of undirected weighted graphs. We introduce and study new graph comparison methods based on heat diffusion. The novelty of our approach does not especially come from the definition of new heat-diffusion-based metrics or semi-metrics d_t, but rather in the introduction of the concept of *distance process*, where for a pair of graphs (G, G') we consider the family of distances over a range of diffusion times $\{d_t(G, G'),\ t \in [0, T]\}$, for some $T > 0$. Moreover, by representing graphs by *persistence diagrams* (tools from topological data analysis), we are able to compare graphs of different sizes or without node correspondence.

The study of these distance processes brings us to the theory of empirical processes. For distributions of graphs with bounded sizes and bounded positive weights, we prove a functional central limit theorem for our processes and refine it by proving that the speed of convergence is actually independent of the graph size. Moreover,

[22] *e.g.*, 3D shapes [TMB14], images [QTT+19, RYB+20], time series [SDB16, Ume17].

even though the proofs build on standard results on empirical processes, our formulations of the Donsker theorem and Gaussian approximation result for continuous processes index by a real parameter are very general and could easily be applied to other distance processes or even to other processes in general.

This work on distance processes yields to theoretically founded analysis of graph data sets. Most importantly, we construct consistent two-sample tests for graph data. We study their performances on simulated data sets and show that their behavior matches with the asymptotic theoretical results, even in the finite samples setting. Furthermore, we consider the problem of distribution shift detection for NNs. When a NN makes predictions on a new data set with shifted distribution (compared to the training and test set), we want to be able to detect whenever it performs poorly. We propose to use our two-sample tests on activation graphs. Our approach has the advantage of being applicable, regardless of the nature of the input data. Furthermore, we show that activation graphs and our two-sample tests together accurately capture the behavior of neural networks and are able to detect whenever their behavior changes. Compared to other methods, our approach is not the most sensitive to distribution shifts, but it is more aware of the NN's robustness and sensitivity, and would avoid alerting the user for changes that are well handled by the NN.

Heat diffusion distances. We analyze distance processes defined from two heat-diffusion-based distances, namely the heat kernel distance (HKD) and the heat persistence distance (HPD).

For an undirected weighted graph G of size n, let L be its unnormalized Laplacian matrix. The *heat kernel* at time $t \geq 0$ of G is the symmetric matrix e^{-tL}. It provides all possible solutions to the heat equation $\partial_t u_t = -Lu_t$, where $u_t \in \mathbb{R}^n$ is the vector representing the heat distribution in the graph. Hence, the heat kernel encodes how heat diffuses in the graph for a diffusion time t.

We consider two graphs G and G' for which the node correspondence is known, so we number the nodes to align the graphs. Hence, we can compare their heat kernels using any matrix norm. We choose the Frobenius norm $\|\cdot\|_F$ and define the Heat Kernel Distance (HKD) by

$$D_t((G, G')) = \|e^{-tL} - e^{-tL'}\|_F, \tag{1.4}$$

where L and L' are the Laplacian matrices of G and G' respectively[23]. In an attempt to conserve as much multi-scale information as possible, we introduce the notion of distance processes. As a result, for some $T > 0$, we consider the family $\{D_t((G, G')),\ t \in [0, T]\}$ of all the heat kernel distances, and call it the HKD process.

Let G and G' be two graphs, eventually of different sizes or not aligned. We follow the approach of [CCI+20] of comparing G and G' via their persistence diagrams. These encode the topological structure of the graphs, given a heat-diffusion-based

[23]Double brackets are inserted in (1.4) to match with the framework of empirical processes: the pair of graphs (G, G') is evaluated by the function $D_t(\cdot)$.

filtering function. Consider the *heat kernel signature* (HKS) defined for a graph $G = (V, E)$ with Laplacian matrix L as the function $h_t(G) : V \ni i \mapsto (e^{-tL})_{i,i}$. This function is used to compute four types of persistence diagrams, characterizing four types of topological features that we can interpret as downward branches, upward branches, connected components, and loops[24]. Each persistence diagram is generically denoted by $Dg(G, h_t(G))$. These diagrams can be compared using the *bottleneck distance* d_B, a sort of optimal transport distance in the space of persistence diagrams. As a result, we define the *heat persistence distance* (HPD) by

$$H_t((G, G')) = \max_{Dg} d_B(Dg(G, h_t(G)), Dg(G', h_t(G'))),$$

where the maximum is taken over all four types of diagrams[25]. Again, to consider all possible HKS filters, we study the HPD process defined as $\{H_t((G, G')),\ t \in [0, T]\}$.

For random graphs G and G', we can see the HKD and HPD processes as the evaluation of the random pair (G, G') by the families of functions $\mathcal{F}_{HKD} = \{D_t,\ t \in [0, T]\}$ and $\mathcal{F}_{HPD} = \{H_t,\ t \in [0, T]\}$ respectively. This corresponds exactly to the notion of empirical process. It is through this framework that we analyze the HKD and HPD processes.

Results for empirical processes. To analyze the heat distance processes, we take advantage of the already established theory of empirical processes. The first step in our analysis is to derive statistical results about general Lipschitz-continuous empirical processes. As we prove, both the HKD and HPD processes are Lipschitz-continuous. Consider a general probability space $\mathcal{X}$, equipped with a probability measure P and a family $\mathcal{F}$ indexed by $[0, T]$ of measurable functions $f_t : \mathcal{X} \to \mathbb{R}$. We show the following result.

Theorem 1.1. *Assume there exists $k > 0$ and $M > 0$ such that for all $x \in X$, $t \mapsto f_t(x)$ is k-Lipschitz continuous and bounded by M. Then for a given sample $X_1, \dots, X_N$ drawn under P, the following process*

$$\{G_N f_t,\ t \in [0, T]\} := \left\{ \sqrt{N} \left(\frac{1}{N} \sum_{i=1}^{N} f_t(X_i) - \mathbb{E}_P \left[f_t(X) \right] \right),\ t \in [0, T] \right\}$$

converges in some sense to a Gaussian process $\mathbb{G} = \{G_t,\ t \in [0, T]\}$.

Complements about the type of convergence and the definition of Gaussian processes can be found in Section 2.3. This theorem is a functional version of the central limit theorem. Later, it will be fundamental to provide asymptotic guarantees to resampling methods.

To get information about the speed of convergence in Theorem 1.1, we derive a Gaussian approximation result.

[24] Up/down orientation is taken with respect to $h_t(G)$, as if the nodes of G were plotted in a space where the vertical axis represents the values of $h_t(G)$.

[25] Same remark about the double brackets of $H_t((G, G'))$, as in (1.4).

Theorem 1.2. *Let $r_N = N^{-1/7}\log(N)^{9/14}$ for any integer N. Then, under the same hypotheses of the previous theorem, for all $\lambda > 1$, there exist a constant ρ and a rank N_0 such that for all $N \geq N_0$, one can construct on the same probability space both the sample $X_1, \ldots, X_N$ and a version $\mathbb{G}^{(N)}$ of the Gaussian process $\mathbb{G}$ such that they verify*

$$\mathbb{P}\left(\sup_t \left|G_N f_t - \mathbb{G}_t^{(N)}\right| > \rho . r_N\right) \leq N^{-\lambda}.$$

The important consequence of this theorem is that the rate of convergence r_N does not depend on the parameters of the problem: the Lipschitz constant k and the uniform bound M. This is encouraging because when applied to HKD and HPD processes, this result means that the speed of convergence to the Gaussian processes will be independent of the graph sizes.

Theorem 1.1 and Theorem 1.2 come from already established and more general results. However, working in the specific framework of continuous processes indexed by a real parameter allows our results to have simple statements. The hypotheses required to apply them are quite elementary, as the processes essentially need to be uniformly Lipschitz-continuous. By taking $\mathcal{X}$ to be specific spaces of pairs of weighted graphs, we show that we can apply Theorem 1.1 and Theorem 1.2 to $\mathcal{F}_{HKD}$ and $\mathcal{F}_{HPD}$. Moreover, we believe that these theorems could easily be applied to other processes.

Furthermore, we refine the result of Theorem 1.2 by making explicit the dependency of ρ and N_0 with respect to k and M and show that it is polynomial. This result, coupled with the fact that the rate r_N is independent of k and M, offers good hints that methods based on the Gaussian approximation will work well in practice, even in the finite sample setting.

Statistical applications to samples of pairs of graphs. To apply Theorem 1.1 to the HKD and HPD processes, the most natural way is to consider samples of pairs of graphs. By doing so, we obtain asymptotic guarantees on the construction of confidence bands and two-sample tests using resampling methods (also called *bootstrap* methods). Consider a sample of pairs of graphs $(G_1, G_1'), \ldots, (G_N, G_N')$. We want to determine the width of the confidence band around the empirical mean process $\{P_N d_t, t \in [0,T]\} := \{N^{-1}\sum_i d_t(G_i, G_i'), t \in [0,T]\}$, where d_t is either the HKD or the HPD. This can be done by drawing several resamplings with replacement $((\hat{G}_1^{(b)}, \hat{G}_1'^{(b)}), \ldots, (\hat{G}_N^{(b)}, \hat{G}_N'^{(b)}))_{1\leq b\leq B}$. From Theorem 1.1, we can show that the bootstrap processes

$$\left(\left\{\sqrt{N}\left(\frac{1}{N}\sum_i d_t(\hat{G}_i^{(b)}, \hat{G}_i'^{(b)}) - \frac{1}{N}\sum_i d_t(G_i, G_i')\right),\ t \in [0,T]\right\}\right)_{1\leq b\leq B} \tag{1.5}$$

are almost distributed as the Gaussian limit process. Hence, by computing $\hat{c}$, an empirical upper quantile of the supremum over t of processes in (1.5), $N^{-1/2}\hat{c}$ provides a consistent width for the confidence band.

Based on similar idea, we can construct a two-sample test for samples of pairs of graphs $((G_{1,1}, G'_{1,1}), \dots, (G_{1,N}, G'_{1,N}))$ and $((G_{2,1}, G'_{2,1}), \dots, (G_{2,M}, G'_{2,M}))$. Denote by P_1 and P_2 the probability distributions generating the samples. We want to perform a test that distinguishes between the null hypothesis $H_0 \ : \ P_1 = P_2$ and the alternative hypothesis $H_1 \ : \ P_1 \neq P_2$. To do so, we propose a test based on the comparison of the empirical means of the distance processes, by computing the test statistic T defined as

$$T = \sqrt{\frac{NM}{N+M}} \sup_{t\in[0,T]} \left| \frac{1}{N} \sum_i d_t(G_{1,i}, G'_{1,i}) - \frac{1}{M} \sum_i d_t(G_{2,i}, G'_{2,i}) \right|. \tag{1.6}$$

Similar to the computation of the width of the confidence band, we can compute bootstrap equivalents $T^{(1)}, \dots, T^{(B)}$ of the test statistics. For any b, $T^{(b)}$ is computed using formula (1.6) but on bootstrap samples $((\hat{G}^{(b)}_{1,1}, \hat{G}'^{(b)}_{1,1}), \dots, (\hat{G}^{(b)}_{1,N}, \hat{G}'^{(b)}_{1,N}))$ and $((\hat{G}^{(b)}_{2,1}, \hat{G}'^{(b)}_{2,1}), \dots, (\hat{G}^{(b)}_{2,M}, \hat{G}'^{(b)}_{2,M}))$ drawn with replacement from the pooled sample $((G_{1,1}, G'_{1,1}), \dots, (G_{1,N}, G'_{1,N}), (G_{2,1}, G'_{2,1}), \dots, (G_{2,M}, G'_{2,M}))$. The values $(T^{(b)})_{1\leq b\leq B}$ provide a good estimate of the distribution of the test statistic under the null hypothesis. Hence, their empirical upper α-quantile $\tilde{c}_\alpha$ can be used as a decision threshold. If $T > \tilde{c}_\alpha$, we reject H_0, while if $T \leq \tilde{c}_\alpha$ we conserve H_0. Thanks to Theorem 1.1, this procedure ensures an asymptotic control of the level and power of the test. Indeed, denote by $\hat{c}_\alpha$ the (non-empirical) upper α-quantile of the random variable $T^{(1)}$, and recall that $\tilde{c}_\alpha$ is a good approximation of $\hat{c}_\alpha$ for large enough B. Then, we have the following result.

Theorem 1.3. *Assume that P_1 and P_2 are distributions of pairs of weighted graphs, with non-negative and bounded weights. Assume that the graph sizes are bounded, and that the sample sizes M and N goes to infinity with similar speeds*[26]*. Then the test that rejects H_0 whenever $T > \hat{c}_\alpha$ is consistent. This means that the asymptotic level is α :*

$$\mathbb{P}_{H_0}\left(T > \hat{c}_\alpha\right) \underset{M,N\to\infty}{\longrightarrow} \alpha.$$

Moreover, under any alternative verifying $\mathbb{E}_{P_1}\left[d_t(G, G')\right] \neq \mathbb{E}_{P_2}\left[d_t(G, G')\right]$ for some $t \in [0, T]$,

$$\mathbb{P}\left(T > \hat{c}_\alpha\right) \underset{M,N\to\infty}{\longrightarrow} 1.$$

Statistical applications to samples of graphs. Samples of pairs of graphs are not the most frequent type of graph data that one encounters. As a result, we propose a modification of the above two-sample test procedure in order to be able to perform two-sample tests for samples of graphs and not pairs of graphs. The main idea is to start with a random pairing step, where we create three sets of

[26]Here, "similar speeds" means that $N/(N+M) \to \lambda \in (0, 1)$.

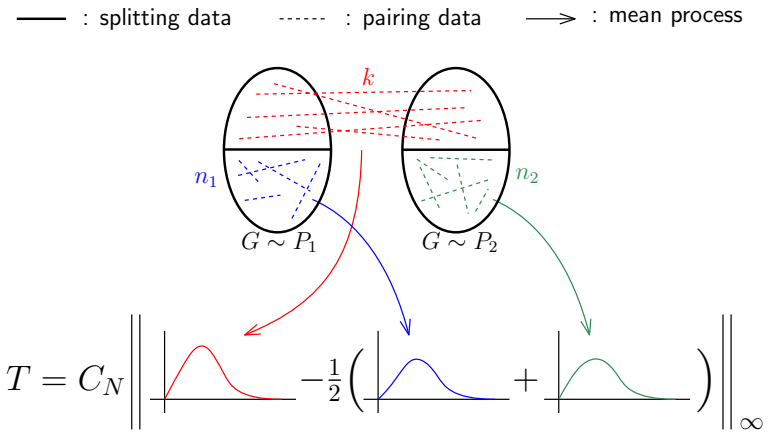


Figure 1.8: Illustration of the pairing step for the two-sample test on samples of graphs.

pairs: pairs made of a graph from each sample (inter-samples pairs), and pairs from the same sample, either from the first one or the second one (intra-sample pairs). We ensure that each graph appears in at most one pair to preserve independence properties. Then, we compare the mean distance process computed on the inter-samples pairs to the mean distance processes computed on the intra-sample pairs (see Figure 1.8). A rejection threshold is computed by a bootstrap method, similar to the previous procedure. We also propose two variations of this two-sample test for samples of graphs. The first one is to include additional information while pairing graphs. We consider the situation where graphs are labeled: either because they come from classification data or because a classifier is available. In that case, instead of pairing graphs at random, we can pair them so that their labels match. Thus, distance processes now only compare pairs of graphs with the same label, meaning more similar graphs. This often results in a reduction of the variance of the distance processes and ultimately in an increase of the test's power. The second variation follows the observation that, as each graph appears in at most one pair, not all pairs can be considered. Hence, the pairing step does not capture all possible information. We propose a multiple testing approach where we repeat the pairing step several times, compute the p-values associated with each test, and apply a Benjamini-Hochberg procedure to make the final decision. In our case, although the tests are not independent, we obtain in our simulations a good control of the test level. The Benjamini-Yekutieli procedure can also be used. Although more conservative, it ensures a theoretical control of the test level, without any assumption on the dependence of the test statistics.

Simulations. All these methods designed to analyze and compare samples of graphs or pairs of graphs are analyzed in different contexts in Part II. We first evaluate the construction of confidence bands and two-sample tests for pairs of graphs under various random graph models (Erdős-Rényi (ER) model, stochastic block model, random geometric graph model). Experiments confirm the theoretical result about the asymptotic control of the level and power even with samples of

a few hundred pairs of graphs. Under ER models with different parameters, analytic computations are tractable. Therefore, we compute the threshold determining when the optimal Neymann-Pearson test can distinguish between two ER models with different parameters that depend on the sample size. Experiments show that our HKD-based two-sample test can distinguish between the two models even when working close to the phase transition, indicating satisfactory power performances of the test.

Applications to distribution shift detection. The rest of Part II is dedicated to the contribution of the HKD-based two-sample tests to the problem of distribution shift detection in the context of neural network learning. We propose to use activation graphs as the input data of our tests. By testing on activation graphs, we test whether the way the neural network processes the inputs has changed.

We evaluate our HKD-based two-sample test and its variations (pairing by labels and multiple testing) on several data sets and NN architectures. We first conduct experiments on the MNIST data set [LCB10] of hand-written digit images with two low-complexity NN architectures: a dense neural network and a convolutional neural network. Then, we considered more complex data and architectures by studying point clouds data processed by a neural network called *Ripsnet* [dSHC+22]. The network is designed to return robust estimations of topological descriptors of points clouds called *persistence images*. In all settings, we compare our methods to BBSD [LWS18], a shift detection method based on a multiple testing procedure on the output values of the network. We also compare our methods to a test similar to ours, that uses activation graphs and the *diffusion distance* [HGJ13] defined as $\max_{t\in[0,T]} D_t(G, G')$ instead of the HKD distance process. Compared to ours, this method computes the maximum of each distance process before computing a test statistic based on these maxima.

In the pure context of two-sample testing, our methods are outperformed in terms of power by BBSD. That means that even for slight changes, BBSD has a better detection rate than our methods. However, in the context of distribution shift detection, BBSD is actually too sensitive and would return warning signals even for changes in the distribution that do not impact the network performances. On the other hand, it arises from the experiments that the method based on the *diffusion distance* has not enough power to detect changes before the network accuracy significantly drops, which is not the case for our methods. We explain this phenomenon by observing that the most discriminating diffusion time to compare data when using distance processes is not necessarily close to the most discriminating diffusion times used to compare graphs in each pair. In the end, the HKD-process-based methods are performing well in the context of distribution shift detection.

The conclusions of this series of experiments are twofold. Firstly, it demonstrates the utility of considering distance processes instead of the diffusion distance. Secondly, it shows that the activation graphs are good features for monitoring the behavior of neural networks and that the HKD-process-based method we introduced, and its variations, are well suited to analyze these features.

1.2.4 Outline

Chapter 2: Background. This thesis takes advantage of several fields, ranging from graph theory, to topological data analysis, to more classical statistics, to neural network learning. This chapter introduces these notions and favors the reader's understanding instead of exhaustiveness. The chapter starts with an introduction to graphs theory and gives particular attention to spectral considerations. It is followed by a presentation of topological data analysis, the classical framework, and the alternatives for graphs. Then, we present classical definitions and results of empirical processes. Finally, we introduce neural networks, more precisely dense and convolutional neural networks. We also define descriptors of the neural networks' behavior called activation graphs.

Part I – Theory –

Chapter 3: Multi-scale comparisons of graphs. In this chapter, we introduce new comparison tools for graphs. We first define the *heat kernel distance* (HKD) and the *heat persistence distance* (HPD), building on the work of [HGJ13] and [CCI+20]. We relate these choices to spectral arguments and graph-specific issues before presenting the main contribution of this thesis. We introduce the new concept of *distance processes*: a family of distances (*e.g.*, HKD or HPD) indexed by a range of scale parameters.

Chapter 4: Statistical properties of real-valued continuous empirical processes indexed by a real parameter. This chapter develops the theoretical results needed to analyze our distance processes. The classical theory of empirical processes supports it. Nonetheless, we derive specific results for continuous empirical processes indexed by a real parameter that are simpler and easily generalizable to other processes. Our results provide asymptotic guarantees that underpin statistical methods. Moreover, fine studies of the convergence rate in the functional central limit theorem supports the empirical validity of these methods in the finite sample setting.

Chapter 5: Application to confidence bands and two-sample tests for graphs. Building on the results of the previous chapter, this one proposes methods to build consistent confidence bands and consistent two-sample tests for graph data. Alternatives to the initial two-sample test are proposed to be able to treat samples of graphs and not pairs of graphs, and to increase the test power. They involve pairing schemes and multiple testing.

Chapter 6: Appendix. The proofs of results from Chapter 4 and Chapter 5 are deferred to this chapter.

Part II – Numerical Illustrations –

Chapter 7: From generative random graph models. This chapter is dedicated to empirically studies of the HKD and HPD-based two-sample tests on pairs of graphs. We conduct experiments on generative random graph models in which we evaluate the tests' performances by evaluating their levels and powers.

Chapter 8: Distribution shift detection from activation graphs in neural network learning. In this last chapter, we tackle the problem of detecting bad performances in neural networks when they are presented with new data from shifted distributions. We propose to apply our two-samples test to activation graphs. We compare our approach to other shift detection methods and confirm that activation graphs are relevant features to capture changes in the neural network behavior. Moreover, HKD-process-based two-sample tests detect changes with a better correlation with bad neural network performances, compared to the other methods that are either too sensitive or not powerful enough.

The work from Chapters 3-7 has been submitted [Las21].

Chapter 2

Background

This chapter provides background information on notions involved in Part I and Part II. The objective is not to provide an exhaustive presentation of the various subjects but rather to give a general understanding on the following topics: graphs (Section 2.1), topological data analysis (Section 2.2), empirical processes (Section 2.3) and neural networks (Section 2.4). More precisely, we present definitions, standard results, and general remarks, to provide an adequate description of the notions that will serve the rest of the manuscript.

2.1 Graphs

This section presents the basics of graph theory. It introduces the concept of graphs, their different matrix representations, and some spectral considerations. For references on graph theory, we refer the reader to [BLW86, TT01]. For more emphasis on spectral graph theory, see [CDGT88, CG97].

2.1.1 Definitions and representations

A graph is defined through a set of vertices (also called nodes) V and a set of edges $E \subset V \times V$. We note $G = (V, E)$. For vertices $u, v \in V$, if $(u, v) \in E$ this means that u and v are connected in the graph. We only consider graphs with a finite set of vertices in the following. The number of vertices $|V|$ is called the *size* of the graph. Graphs will be undirected so that if $(u, v) \in E$ then $(v, u) \in E$ also. Moreover, we also consider weighted graphs, such that for any pair of vertices (u, v), we provide a weight $w_{u,v} \in \mathbb{R}$ representing the strength of the connection between u and v. $w_{u,v} = 0$ means no connection. Unweighted graphs are a special case of weighted graphs with weights in $\{0, 1\}$.

Matrix representation. Consider an undirected weighted graph G with vertex set $\{1, \dots, n\}$ and with nonnegative weights encoded by the weight matrix W. So $W_{i,j}$ is the weight of edge (i, j). Denote by D the diagonal matrix whose entry $D_{i,i}$ is $d_i = \sum_j W_{i,j}$, the degree of node i. Then the *unnormalized Laplacian* matrix L is

defined by $L = D - W$. This can also be written as

$$L_{i,j} = \begin{cases} -w_{i,j} & \text{if } i \neq j \\ d_i & \text{if } i = j \end{cases} . \tag{2.1}$$

From a signal processing point of view, we can see the Laplacian matrix as a differential operator. Consider a vector f in $\mathbb{R}^n$ and see it as a signal on the vertices of G, that is, vertex i is mapped to the value f_i. Then, simple computations show that

$$Lf = \left(\sum_{j=1}^{n} w_{i,j}(f_i - f_j) \right)_{1 \leq i \leq n},$$

and

$$f^T L f = \sum_{1 \leq i,j \leq n} w_{i,j}(f_i - f_j)^2. \tag{2.2}$$

These computations show that L captures the variations of f along the edges. For example, small values of $f^T L f$ are obtained for signals close to being constant.

The same remarks can be made for $\tilde{L} := I - D^{-1/2} W D^{-1/2}$, the symmetric normalized Laplacian matrix. In that case, (2.2) rewrites as

$$f^T \tilde{L} f = \sum_{1 \leq i,j \leq n} \frac{w_{i,j}}{\sqrt{d_i d_j}} (f_i - f_j)^2.$$

2.1.2 Spectral analysis

For an undirected graph G, its Laplacian matrix L is a symmetric matrix. Having nonnegative weights in W ensures that L is a positive-semidefinite matrix (see equation (2.2)). Hence L admits a spectral decomposition with nonnegative eigenvalues. Let n be the size of G. Then let $\lambda_1 \leq \cdots \leq \lambda_n$ be the eigenvalues of L and let $(\phi_1, \ldots, \phi_n)$ be a family of orthonormal eigenvectors. We denote by Λ the diagonal matrix containing the eigenvalues on the diagonal and ϕ the matrix whose columns are the ϕ_i's so that L admits the following decomposition

$$L = \phi \Lambda \phi^T = \sum_{k=1}^{n} \lambda_k \phi_k \phi_k^T. \tag{2.3}$$

Remark 2.1. Note that $\lambda_1 = 0$ and that ϕ_1 can always be chosen to be the vector whose entries are equal to $1/\sqrt{n}$. In the following, we always make this choice.

Connectivity. We now present results that link the smallest eigenvalues of the Laplacian matrix and their eigenvectors to connectivity properties of the graph. For simplicity, all the results are presented for unweighted graphs, but similar results exist for weighted graphs. Let us start with some definitions.

Definition 2.1 (Connected graph)**.** A graph $G = (V, E)$ is said to be connected, if for every pair of vertices (u, v), there is a path from u to v. That is, one can find $u = u_0, \ldots, u_k = v$ in V such that for all $l < k$, $(u_l, u_{l+1}) \in E$.

Definition 2.2 (Subgraph)**.** We say that $G' = (V', E')$ is a subgraph of $G = (V, E)$ if $V' \subseteq V$ and $E' \subseteq (V' \times V') \cap E$. For a graph $G = (V, E)$ and a subset of vertices $V' \subseteq V$, the subgraph of G induced by V' is defined as $G' = (V', E_{|V'})$ where $E_{|V'} = (V' \times V') \cap E$.

Definition 2.3 (Connected component)**.** A connected component of a graph G is a maximal connected subgraph of G. This means that it is a connected subgraph $G_c = (V_c, E_{|V_c})$ such that for any $u \notin V_c$ and for $V_c' = V_c \cup \{u\}$, the subgraph $(V_c', E_{|V_c'})$ is not connected.

Given all these definitions, we can now state a classical result that links the multiplicity of the zero eigenvalue of the Laplacian matrix with the connectivity of the graph.

Theorem 2.1. *The multiplicity of the zero eigenvalue of a graph Laplacian matrix is equal to the number of connected components in the graphs.*

Proof. Consider a graph G and its connected components $G_1 = (V_1, E_1), \ldots, G_k = (V_k, E_k)$. For any i such that $1 \leq i \leq k$, the indicator vector ψ_i of V_i, that is the vector such that $\psi_i(v) = 1$ if $v \in V_i$ and 0 otherwise, verifies $L\psi_i = 0$ (see equation (2.1)). Hence the multiplicity of the zero eigenvalue is at least k. Then take any eigenvector ψ associated with the zero eigenvalue. Then from equation (2.2), it is clear that ψ must be constant on each connected component of G. So ψ can be written as a linear combination of $\psi_1, \ldots \psi_k$. ■

This result indicates that by looking at λ_2 we can deduce whether the graph is connected or not. Actually, the *algebraic connectivity* of G, denoted by $a(G) := \lambda_2$, contains even more information. This has extensively been studied by Fiedler [Fie73]. Basically, when $a(G)$ is positive, it provides information on how close to being disconnected the graph is. In the following, we make this statement more rigorous.

Definition 2.4 (Edge connectivity)**.** For a graph $G = (V, E)$, the edge connectivity is defined as the minimal number of edges that need to be removed to disconnect the graph. It is denoted by $e(G)$.

Fiedler showed in [Fie73] the following result.

Theorem 2.2. *Consider a graph $G = (V, E)$ that is not the complete graph. Then*

$$a(G) \leq e(G).$$

This show that for graphs that are almost disconnected, the algebraic connectivity is small. This inequality can be tight, for example is G is a star graph (one node that connects to all the other; see Figure 2.1 (left)), then $a(G) = e(G) = 1$. On the other hand, the algebraic connectivity could be considered as a finer measure of connectivity than the edge connectivity. Consider for example the graph G of size

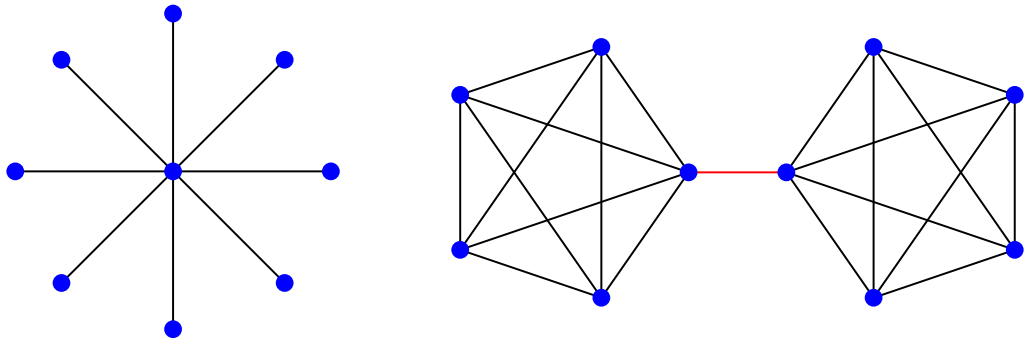


Figure 2.1: Left: an example of a star graph with nine vertices. Right: an example of two complete graphs of size 5, connected by one edge (in red). This red edge plays a special role in terms of connectivity.

$2n$ made by binding two complete graphs of size n with one edge; see Figure 2.1 (right). Then, $e(G) = 1$ and we can show that

$$a(G) = \frac{n + 2 - \sqrt{n^2 - 4(n-1)}}{2} \sim \frac{3}{2n}.$$

In that case, the algebraic connectivity translates to the fact that the graph "looks" almost like a disconnected graph. In some sense, the algebraic connectivity captures here multi-scale information. The fact that the graph is connected is local information: the edge connecting the two complete graphs. However, the fact that the rest of the graph is made of highly connected components appears in the fact that $a(G)$ is close to 0 for large n.

These examples show that the Laplacian spectrum carries some fine information about the graph. In the following, we show how one can explore this information and the one contained in the eigenvectors of the Laplacian matrix.

Spectral clustering. The eigenvector ϕ_2 associated with the eigenvalue λ_2, often called the Fiedler vector, also contains some information. While λ_2 tells us if the graph is almost connected, ϕ_2 can tell which are the two almost connected components. In the example with the two connected complete graphs, if we assume that vertices from 1 to n are in a complete graph, the ones from $n+1$ to $2n$ are in the other complete graph, and that the additional edge is $(n, n+1)$, we have that ϕ_2 is proportional to

$$(\underbrace{1, \dots, 1}_{n-1}, \alpha, -\alpha, \underbrace{-1, \dots, -1}_{n-1})^T,$$

where $\alpha = 1 - \lambda_2 = 1 - (2/3)n^{-1} + o(n^{-1})$. In this instance, we see that we can recover the two complete graphs according to the sign of the coordinates in the Fiedler vector.

This use of the Fiedler vector is a basic example of spectral clustering. Spectral clustering has been introduced by [DH03] and [Fie73] and then applied in various fields. A survey about the history of spectral clustering can be found in [ST96]. The

central idea is to consider the embedding $\Phi : V \mapsto \mathbb{R}^K$ for some K, given by

$$\Phi(i) = (\phi_1(i), \dots, \phi_K(i))^T, \ \forall i \in V.$$

This embedding distributes the nodes in $\mathbb{R}^K$ so that it is simple to find groups of nodes consistent with the graph structure. For example, one can apply some classical clustering algorithm such as k-means [Llo82] on the points $\Phi(1), \dots, \Phi(n)$. The previous example with the two complete graphs was using $K = 1$. Note that spectral clustering may use the eigenvectors of the adjacency matrix or the Laplacian matrix.

2.1.3 Heat diffusion on graphs

Let G be an undirected weighted graph of size n with nonnegative weights and let L be its Laplacian matrix. Recall that L can be interpreted as a differential operator for signals on the vertices of G. It is used to define the heat equation on graphs.

For $t \geq 0$, let $u_t \in \mathbb{R}^n$ be the vector whose i-th coefficient represents the amount of heat of node i at time t. Then, u_t follows the heat equation :

$$\forall t \geq 0, \quad \frac{d}{dt} u_t = -L u_t$$

The solution is given by $u_t = e^{-tL} u_0$. The matrix e^{-tL} is called the *heat kernel* and describes how heat diffuses in the graph. It encodes all the solutions of the heat equation. The i-th column of e^{-tL} contains the amount of heat of each node at time t when a single unit of heat was placed at node i at time $t = 0$. From (2.3), the heat kernel decomposes in

$$e^{-tL} = \phi e^{-t\Lambda} \phi^T = \sum_{k=1}^{n} e^{-t\lambda_k} \phi_k \phi_k^T.$$

Note that the diffusion time t acts as a scale parameter, where diffusion for small values of t considers only direct neighborhoods of the nodes. In contrast, for larger values, it takes more global structures into account.

Link with spectral clustering. We have seen above that spectral clustering uses the embedding $\Phi : j \to (\phi_1(j), \dots, \phi_K(j))^T$ for some K. Remark that the heat kernel can be written as

$$e^{-tL} = \left(e^{-\frac{t}{2}\Lambda} \phi^T\right)^T . \left(e^{-\frac{t}{2}\Lambda} \phi^T\right)$$

and consider another embedding $\Phi_t : V \to \mathbb{R}^n$ defined as

$$\Phi_t(j) = (e^{-t\lambda_1/2} \phi_1(j), \dots, e^{-t\lambda_n/2} \phi_n(j))^T.$$

Then, the heat kernel can be seen as the matrix of all scalar products between the vectors $\Phi_t(1), \dots, \Phi_t(n)$. Note that the embedding Φ and Φ_t are quite similar. The former performs hard thresholding by considering only the K first eigenvectors. The latter performs soft thresholding by multiplying the i-th eigenvector by $e^{-t\lambda_i/2}$.

Heat diffusion with the normalized Laplacian matrix. In some work, heat diffusion is defined using the normalized Laplacian matrix $\tilde{L} = I - D^{-1/2}WD^{-1/2}$ instead of the standard one. The time derivative of $u_t(i)$ for some vertex i is then given by

$$\frac{d}{dt}u_t(i) = \frac{1}{\sqrt{d_i}}\sum_j w_{i,j}\left(\frac{u_t(j)}{\sqrt{d_j}} - \frac{u_t(i)}{\sqrt{d_i}}\right).$$

This means that if j is hotter that i but d_j is sufficiently larger than d_i, then j tends to decrease the temperature at i. For example, consider the unweighted star graph with 10 vertices. That is the graphs with vertices from 1 to 10 such that the only edges are between vertex 1 and all other vertices. So $d_1 = 9$ and for all $j \neq 1$, $d_j = 1$. Then if $u_t(2) = 0.5$ and $u_t(1) = 1$,

$$\frac{d}{dt}u_t(2) = \frac{1}{\sqrt{9}} - \frac{0.5}{\sqrt{1}} < 0.$$

So the temperature of $u_t(2)$ will decrease, even though the only neighbor of 2 is hotter.

Based on the remarks of the paragraph and in order to keep the intuition on what we mean by heat diffusion on graphs, in the following we will only use the standard Laplacian matrix to define heat diffusion.

2.2 Topological Data Analysis

Topological Data Analysis (TDA) is a relatively recent field in data analysis aiming to capture information relative to the shape of potentially complex or high-dimensional data. This topological information is believed to allow for a better understanding of the data and better performances of machine learning methods that integrate it into their pipeline.

We present here the basics of persistent homology, which is the theory at the heart of TDA. We introduce ordinary persistence and especially the tools used to analyze point clouds. These notions are classical in the fields and are involved in Section 8.4. We also present extended persistence and explain how it can be applied to graphs. It will be a key element in the methods developed in Chapter 3. We refer the reader to [CSEH09], [EH10] and [Oud15] for a complete description of these theories.

2.2.1 Persistence homology

The theory of *persistent homology* (or *persistence* for short) allows to study the topology of topological spaces in a multi-scale manner. Usually, given a topological space X and a continuous real-valued function $f : X \to \mathbb{R}$, one considers the family of sublevel sets $X_\alpha := \{x \in X,\ f(x) \leq \alpha\}$, for α varying from $-\infty$ to $+\infty$. The sequence of increasing sublevel sets $(X_\alpha)_{\alpha\in\mathbb{R}}$ is called the *filtration* induced by f. Ordinary persistence records the levels at which topological features like connected

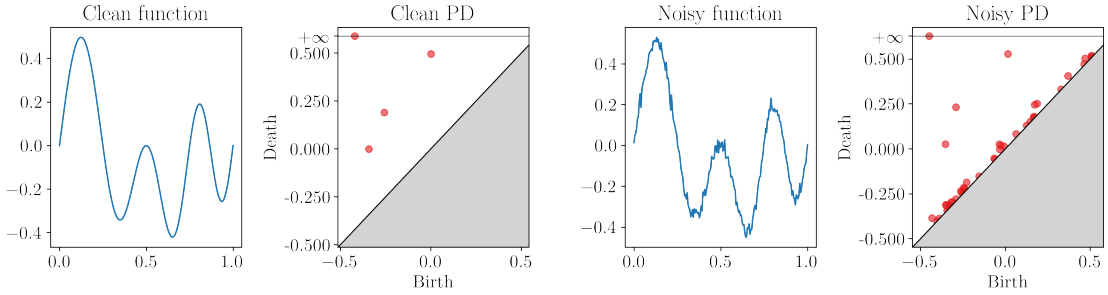


Figure 2.2: Two examples of persistence diagrams (PDs) associated to the sublevel set filtration of two functions: a clean one and a noisy one.

components, loops, cavities, or higher dimensional holes appear and disappear along the filtration. For each feature, its birth and death levels are stored as the coordinates (b, d) of a point in $(\mathbb{R} \cup \{-\infty, +\infty\})^2$. The multiset of these points is called a persistent diagram (PD). Hence, in a PD points are located above the diagonal as topological features appear before they disappear. Moreover, points far from the diagonal correspond to topological features that *persist* on a wide range of levels. Hence, they represent topological signal. On the other hand, points close to the diagonal are considered as topological noise (see Figure 2.2).

The space of PD can be equipped with the *Bottleneck distance* d_B. We recall its definition. Let μ and ν be two diagrams, *i.e.*, two multisets of points in $\mathbb{R}^2$, and let $\Delta := \{(a, a), a \in \mathbb{R}\}$ be the diagonal. Denote by $\Pi(\mu, \nu)$ the set of bijections from $\mu \cup \Delta$ to $\nu \cup \Delta$. d_B is defined by

$$d_B(\mu, \nu) := \inf_{\pi \in \Pi(\mu,\nu)} \sup_{x \in \mu \cup \Delta} \|x - \pi(x)\|_\infty,$$

where $\|y\|_\infty := \max\{|y_1|, |y_2|\}$ for $y = (y_1, y_2) \in \mathbb{R}^2$. Considering the bijections between the multisets augmented by the diagonal allows us to compare PDs with different numbers of points, as extra points can be matched with points on the diagonal.

One of this theory's fundamental results concerns the stability of persistence diagrams. It is due to [CDSGO16] and states that slightly modifying the function f (with respect to the uniform norm) can only slightly change the persistence diagram.

Theorem 2.3. *(Stability (continuous function)) Let X be a topological space, then for all continuous functions $f, f' : X \to \mathbb{R}$, satisfying an additional regularity assumption specific to persistence theory (see [CDSGO16] for more details), the persistence diagram $Dg(f)$ and $Dg(f')$ verify*

$$d_B(Dg(f), Dg(f')) \leq \|f - f'\|_\infty,$$

with $\|f\|_\infty = \max\{|f(x)|, \ x \in X\}$.

Figure 2.2 illustrates this theorem. All points close to the diagonal in the noisy

PD will be matched to the diagonal and the main points will be match to their equivalent in the clean PD, resulting in a small cost matching.

2.2.2 Persistence for point clouds

A classical framework to apply TDA methods is the study of point clouds in $\mathbb{R}^D$. In essence, point clouds' topology is trivial and not informative. Indeed, a point cloud is composed of several connected components (the points) without any cycle, loop, or cavity. However, points may be arranged in ways that remind us of continuous objects with a richer topology, e.g., think about points sampled from a circle. Persistence homology aims at capturing that.

Essentially, the problem of point clouds is that points are 0-dimensional objects. To circumvent that problem, we classically built high-dimensional structures on top of point clouds called simplicial complexes.

Definition 2.5 (Simplicial complex)**.** Let $X = \{x_1, \dots, x_n\}$ be a finite set. A simplicial complex K with vertex set X is a set of subsets of X verifying the following two conditions :

1. $V \subset K$
2. For all $\tau \in K$ and $\sigma \subset \tau$, $\sigma \in K$.

Elements of K are called *simplices*. If $\tau \in K$ such that its cardinal $|\tau| = k + 1$, it is called a k-simplex, and one should interpret it as a k-dimensional object. For any $\sigma \subset \tau$, σ is called a face of τ.

Usually, to study the topology of a point cloud X, we build K the simplicial complex containing all subsets of X of cardinal at most $k+2$, where k is chosen by the user. Basically, k correspond to the dimension of the topological holes we are interested in the point cloud, hence we need object up to dimension $k+1$ which are $(k+1)$-simplices, *i.e.,* subsets of cardinal $k+2$. K is a much richer topological space to study than X. In order to apply the persistence approach developed above, K needs to be equipped with a continuous function[1]. The choice of this filter function will impact the resulting filtration and thus the persistence diagram. If we denote by d the Euclidean metric on $\mathbb{R}^D$, a classical choice for the filter is the function f defined on K by

$$f(\tau) = \max_{x,x' \in \tau} d(x, x'), \ \forall \tau \in K.$$

This choice leads to the so-called Vietoris-Rips filtration $Rips(X) := (K_r)_{r \geq 0}$. From the definition of f, remark that for all $r \geq 0$, K_r is still a simplicial complex. Furthermore, τ is in K_r if and only if $\tau \in K$ and for all pairs of vertices $x, x' \in \tau$, $d(x, x') \leq r$.

PDs computed from Vietoris-Rips filtration also have stability properties, but with respect to the Hausdorff distance.

[1] Note that all functions defined on a finite set are continuous.

Definition 2.6 (Hausdorff distance)**.** For two compact sets C and C' in $\mathbb{R}^D$, the Hausdorff distance between C and C' is defined as

$$d_H(C, C') := \max \left\{ \sup_{x \in C} d(x, C'), \ \sup_{x' \in C'} d(x', C) \right\},$$

where $d(x, C') = \inf_{x' \in C'} d(x, x')$ and $d(x', C) = \inf_{x \in C} d(x, x')$.

Then we have the following results from [CDSO14].

Theorem 2.4 (Stability (point clouds))**.** *Given two points clouds X and X' in $\mathbb{R}^D$, their persistence diagrams computed on the Vietoris-Rips filtrations verify*

$$d_B(Dg(Rips(X)), Dg(Rips(X'))) \leq 2 d_H(X, X').$$

2.2.3 Persistence for graphs

The classical framework of persistent homology, with a filtration given by the sublevel sets of a filter function, can be adapted for graphs. To do so, we use the theory of extended persistence; see [CSEH09] for more details. Let $G = (V, E)$ be a graph with vertex set V and edge set E, and f be a real-valued function on V. Consider the family of sublevel subgraphs $(G_\alpha)_{\alpha \in \mathbb{R}}$, where $G_\alpha = (V_\alpha, E_\alpha)$ with $V_\alpha = \{v \in V, \ f(v) \leq \alpha\}$ and $E_\alpha = \{\{v, v'\} \in E, \ v, v' \in V_\alpha\}$. Across the family of sublevel graphs, as α increases, we can record birth and death levels of connected components and birth levels of loops. However, as a connected component only dies when connected to an older connected component, remark that the connected components of G will never die. Similarly, as a loop dies when it gets "filled-in" by a 2-dimensional object, loops appearing in G_α (an object of maximal dimension 1) will never die. To prevent topological features from having no death levels (or infinite death levels), the theory of extended persistence suggests also considering the family of superlevel subgraphs. Define $G^\alpha = (V^\alpha, E^\alpha)$ similarly to G_α with $V^\alpha = \{v \in V, \ f(v) \geq \alpha\}$. We assign a death level to connected components of G and loops. It is defined as the level at which they appear in the family of superlevel subgraphs when α decreases. Additionally, we record the birth and death of connected components in the family of superlevel subgraphs. Hence, extended persistence detects four types of topological features and extracts their birth and death levels (b, d) corresponding to four types of points :

- Ord_0 : birth and death of a connected component in (G_α).
- Rel_1 : birth and death of a connected component in (G^α).
- Ext_0 : birth and death of a connected component of G when using (G_α) and (G^α).
- Ext_1 : birth and death of a loop when using both (G_α) and (G^α).

These four types of topological features can be seen as downward branches, upward branches, connected components, and loops, respectively, with the orientation being taken with respect to f. For a graph G and a function f on its vertices, we will denote by $Ord_0(G, f)$, $Rel_1(G, f)$, $Ext_0(G, f)$ and $Ext_1(G, f)$ the persistence diagrams containing the corresponding points. In the following, $Dg(G, f)$ will generically denote any of these four diagrams. We refer the reader to [CCI+20, Section 2.1] for a more precise and illustrative presentation of extended persistence diagrams on graphs.

We state a stability result for extended persistence diagrams computed on graphs. It is a consequence of the more general stability result for persistence diagrams, and it has been established by [CDSGO16, CSEH09].

Theorem 2.5. *For all graphs $G = (V, E)$, for all $f, f' : V \to \mathbb{R}$, and for all diagram constructions Dg among Ord_0, Rel_1, Ext_0 and Ext_1,*

$$d_B(Dg(G, f), Dg(G, f')) \leq \|f - f'\|_\infty,$$

with $\|f\|_\infty = \max\{|f(v)|\,,\ v \in V\}$.

2.2.4 Vectorization

A known difficulty when using persistence diagrams (PDs) in data analysis and machine learning is the non-linearity of the space of PDs. As multisets of points in $(\mathbb{R} \cup \{-\infty, +\infty\})^2$, diagrams are not vectors; thus, they can not be summed or multiplied scalars. Unfortunately, most machine learning pipelines require finite-dimensional vectors as inputs. The most classical approach to solve this problem is to compute *vectorizations* of PDs that are stable and informative. While various approach have been proposed [CFL+14, B+15, COO15, Kal19], we choose to only present *persistent images* [AEK+17], as they will be used in Section 8.4.

Consider a persistent diagram D, computed from the filtration induced by a continuous filter on some topological space. Hence, points are above the diagonal. We start by rotating the diagram by applying the transformation $T : (b, d) \mapsto (b, d - b)$. Then, we compute a *persistent surface*, by summing Gaussian bumps $g_u(\cdot)$ of fixed variance σ^2 centered on each point u of $T(D)$, weighted by $w(u)$. Hence, the persistent surface is defined by

$$\rho_D(z) = \sum_{u \in T(D)} w(u) g_u(z),$$

where $g_u(z) = (2\pi\sigma^2)^{-1} \exp(-\|z - u\|^2 / (2\sigma^2))$ and $w(u)$ verifies $w(b, 0) = 0$. Usually, w is a smooth function of the distance of u to the x-axis. We discretize the persistent surface on a grid domain to obtain the persistent image. Take a rectangular domain $A := [a, b] \times [c, d] \subset \mathbb{R}^2$ and consider a subdivision on a grid $(P_{i,j})_{1 \leq i \leq n_r,\ 1 \leq j \leq n_c}$ such that $A = \cup_{i,j} P_{i,j}$. Finally, the $n_r \times n_c$ persistent image I_D is defined such that its (i, j)-pixel equals $I_D(i, j) = \int_{P_{i,j}} \rho_D(z) dz$.

2.3 Empirical Processes

2.3.1 Introduction and notations

A stochastic process is a family of random variables $\{X_t,\ t \in T\}$ indexed by a set T defined on the same probability space. For example, a Brownian motion $B = \{B_t,\ t \geq 0\}$ is a stochastic process. In all generality, the index set is arbitrary. A realization of all these random variables $\{X_t,\ t \in T\}$ is called a *sample path*.

Inside this family of random objects, we distinguish a specific class of stochastic processes, those defined from a sample of random variables. Consider a probability measure P on some arbitrary space $\mathcal{X}$, and take an *i.i.d.* sample under P of size $N : X_1, \dots X_N$. Denote by P_N the *empirical probability measure* $P_N = N^{-1} \sum_i \delta_{X_i}$, where δ_x is the Dirac mass in $x \in \mathcal{X}$. For a measurable function $f : \mathcal{X} \to \mathbb{R}$, we denote by $P_N f$ the integration of f with respect to the measure P_N that is $P_N f = N^{-1} \sum_i f(X_i)$. Then for any family $\mathcal{F}$ of measurable functions from $\mathcal{X}$ to $\mathbb{R}$, we can define an *empirical process* $\{P_n f,\ f \in \mathcal{F}\}$.

Example 2.1 (Empirical distribution function)**.** Take $\mathcal{X} = \mathbb{R}$ and define f_t as $f_t(x) = \mathbb{1}_{x \leq t}$, where $\mathbb{1}$ is the indicator function such that $f_t(x) = 1$ if $x \leq t$ and 0 otherwise. Then by setting $\mathcal{F} = \{f_t,\ t \in \mathbb{R}\}$, the empirical process defined above is just the empirical distribution function $P_N f_t = N^{-1} \sum_i \mathbb{1}_{X_i \leq t} =: F_N(t)$.

Statistical properties of the empirical distribution function are well known. From the law of large number, we know that for a given t, $F_N(t)$ converges almost surely towards $F(t)$ where F is the distribution function of P. The convergence is actually stronger as it is uniform in t. Glivenko and Cantelli showed that $\sup_t |F_N(t) - F(t)| \to 0$. Similarly, the central limit theorem ensure that for each t, $\sqrt{N}(F_N(t) - F(t))$ converges in distribution to a centered Gaussian distribution. Once again, this result can be strengthened, as Donsker proved that the process $\{\sqrt{N}(F_N(t) - F(t)),\ t \in \mathbb{R}\}$ converges in some sense to a centered Gaussian process, which general definition is given below.

Definition 2.7 (Gaussian process)**.** A Gaussian process indexed on a set T is a stochastic process $\{Z_t,\ t \in T\}$ such that for every finite subset $T_k \subset T$, $\{Z_t,\ t \in T_k\}$ follows a multivariate Gaussian distribution, and where the sample paths are almost surely continuous in some sense. See [Kos08, Section 2.2] for a more precise definition.

Remark 2.2. A Gaussian process $Z = \{Z_t,\ t \in T\}$ is characterized by its expectations $\mathbb{E}[Z_t] = m_t$, for all $t \in T$ and by its covariance function $Cov(Z_t, Z_s) = \kappa(t, s)$, for all $t, s \in T$.

All the above results are actually not specific to the empirical distribution function. A lot of work has been done to understand under which conditions a general empirical process $\{P_N f,\ f \in \mathcal{F}\}$ converges uniformly to $\{Pf,\ f \in \mathcal{F}\}$, and under which ones we have a uniform central limit theorem, such that $\{\sqrt{N}(P_N - P)f,\ f \in \mathcal{F}\}$ converges to some Gaussian limit process. For extended details on this work see [VDVW96] and for a more recent and easy to read reference, we advise [Kos08].

In the following, we detail how we can obtain a Donsker theorem (*i.e.* a uniform central limit theorem) for general empirical processes and go further by introducing Gaussian approximation results. Then we will discuss applications of these results to resampling methods (or bootstrap methods) for constructing confidence regions and two-sample tests.

2.3.2 Donsker and Gaussian approximation

In this section, we properly introduce two concepts that will play an important role in Chapter 4. These concepts concern the convergence of the centered and renormalized process $G_N = \{\sqrt{N}(P_N - P)f,\ f \in \mathcal{F}\}$. To introduce them, we need a proper definition of convergence, that is the *weak convergence*. It is define on the space $l^\infty(\mathcal{F})$ of all bounded functions $\phi : \mathcal{F} \to \mathbb{R}$, equipped with the uniform distance d_∞ defined as $d_\infty(\phi, \psi) = \sup_{f\in\mathcal{F}} |\phi(f) - \psi(f)|$, for all $\phi, \psi \in l^\infty(\mathcal{F})$.

Definition 2.8 (Weak convergence)**.** For X_N and X Borel measurable $l^\infty(\mathcal{F})$-valued stochastic processes, we say that X_N converges weakly to X when $\mathbb{E}[h(X_N)] \to \mathbb{E}[h(X)]$, for all bounded continuous function $h : l^\infty(\mathcal{F}) \to \mathbb{R}$, where continuity for h is in term of the d_∞ metric. We note $X_N \rightsquigarrow X$.

From the classical multivariate central limit theorem, for every finite sub-family $\mathcal{F}_k \subset \mathcal{F}$, the finite dimensional marginal $\{\mathbb{G}_N(f),\ f \in \mathcal{F}_k\}$ weakly converges to the a Gaussian distribution. To go from the convergence of processes indexed by all finite $\mathcal{F}_k$ to the one indexed by $\mathcal{F}$ is not trivial. A family of function $\mathcal{F}$ that verifies this is said to be P-Donsker. Here is the proper definition.

Definition 2.9 (Donsker family)**.** A family $\mathcal{F}$ of functions $f : \mathcal{X} \to \mathbb{R}$ is called P-Donsker if $\mathcal{G}_N = \{\sqrt{N}(P_N - P)f,\ f \in \mathcal{F}\}$ converges weakly to a centered Gaussian process $\mathbb{G}$ with covariance function $\kappa(f, g) = \mathbb{E}_P[f(X)g(X)] - \mathbb{E}_P[f(X)]\,\mathbb{E}_P[g(X)]$. We may drop the P notation in P-Donsker whenever there is no ambiguity.

Not that this result informs on the asymptotic behavior of the empirical process $\{P_N f,\ f \in \mathcal{F}\}$. In order for this Donsker property to be more meaningful and informative in practice, it is important to question the speed of convergence.

In the case of real-valued random variables, the way we can have information about the speed of convergence in the standard central limit theorem is for example through the Berry-Essen theorem [Ber41, Ess42]. The latter that the difference between the distribution function of the centered and renormalized variable and the distribution function of the Gaussian limit distribution is uniformly bounded by a term in $O(N^{-1/2})$.

In the case of empirical processes however, distribution functions do not exist. In order to provide information about the speed of convergence, one can prove Gaussian approximation result.

Definition 2.10 (Gaussian Approximation)**.** Let $(r_N)_{N\geq 1}$ be a vanishing sequence of positive real numbers. We say that the process $\{G_N f,\ f \in \mathcal{F}\}$ admits a *Gaussian approximation with rate* r_N, if for all $\lambda > 1$, there exists a constant ρ such that for all

$N \geq 1$ one can construct on the same probability space both the sample $X_1, \ldots, X_N$ in $\mathcal{X}$ and a version $\mathbb{G}^{(N)}$ of the Gaussian process $\mathbb{G}$ verifying

$$\mathbb{P}\left(\sup_{f\in\mathcal{F}} \left|G_N(f) - \mathbb{G}^{(N)}(f)\right| > \rho.r_N\right) \leq N^{-\lambda}.$$

Remark 2.3. If $\{G_N f,\ f \in \mathcal{F}\}$ admits a Gaussian approximation with rate r_N, applying the Borel-Cantelli Lemma would yield to $\left\|G_N - \mathbb{G}^{(N)}\right\|_\infty = O(r_N)$ *almost surely*, where $\|\cdot\|_\infty$ denotes the supremum over $\mathcal{F}$.

The idea of the Gaussian approximation is to measure the similarity of the distribution of G_N with the distribution of the Gaussian process by looking at a coupling that ensure the realisation of both processes on the same probability space will be close with respect to the uniform metric with high probability. And how close both processes can be is given by the rate r_N.

2.3.3 Confidence bands and two-sample tests

Similarly to the case of real-valued random variables, the functional central limit theorem allows to compute asymptotically consistent confidence bands around the empirical mean process. Based on the same ideas, given two samples, we can also perform a consistent two-sample test by comparing the empirical mean processes associated to each sample. We recall the principles of these two constructions.

Confidence Band. Consider a P-Donsker family $\mathcal{F}$, so that $\{\sqrt{N}(P_N - P)f,\ f \in \mathcal{F}\} \rightsquigarrow \mathbb{G}$. Let $c_\alpha := \inf\{u,\ \mathbb{P}\left(\|\mathbb{G}\|_\infty > u\right) \leq \alpha\}$ be the upper α-quantile of the maximum of the Gaussian limit process. As a consequence of Definition 4.1, we have

$$\lim_{N\to\infty} \mathbb{P}\left(\forall f \in \mathcal{F},\ Pf \in \left[P_N f - \frac{c_\alpha}{\sqrt{N}}, P_N f + \frac{c_\alpha}{\sqrt{N}}\right]\right) \geq 1 - \alpha. \tag{2.4}$$

Unfortunately, as the distribution of $\mathbb{G}$ is unknown, c_α can not be directly computed. Instead, consider a bootstrap sample $\hat{X}_1, \ldots, \hat{X}_N$ drawn under P_N and let $\hat{P}_N$ be its empirical probability measure. Consider the process $\{\hat{G}_N f_t, t \in I\}$ where $\hat{G}_N$ is the measure $\sqrt{N}(\hat{P}_N - P_N)$. Theorem 2.6 in [Kos08] ensures that in the case where $\mathcal{F}$ is P-Donsker, $\{\hat{G}_N f_t, t \in I\}$ converges weakly to $\mathbb{G}$, given the data. Let $\hat{c}_\alpha$ be the upper α-quantile of $\|\hat{G}_N f\|_\infty$ given the data. We can use $\hat{c}_\alpha$ to design a consistent confidence band around $P_N f$:

$$\lim_{N\to\infty} \mathbb{P}\left(\forall t, Pf_t \in \left[P_N f_t - \frac{\hat{c}_\alpha}{\sqrt{N}}, P_N f_t + \frac{\hat{c}_\alpha}{\sqrt{N}}\right]\right) \geq 1 - \alpha. \tag{2.5}$$

Note that $\hat{c}_\alpha$ can be estimated with Monte-Carlo simulations, by drawing as many bootstrap samples as we want.

Two Sample Tests. Consider the following setup. Let P and Q be two probability distributions on $\mathbb{X}$, and assume we are given two independent iid samples $(X_1, \dots, X_M)$ and $(Y_1, \dots Y_N)$, drawn under P and Q, respectively. We denote by P_M and Q_N the empirical measures. We would like to test the null hypothesis $H_0 : P = Q$ against the alternative $H_1 : P \neq Q$, by using the family $\mathcal{F}$, assuming that it is Donsker with respect to both P and Q. We follow the approach described in Section 3.7 of [VDVW96], and consider the following test statistic :

$$D_{M,N} := \sqrt{\frac{MN}{M+N}} \left\| P_M f - Q_N f \right\|_\infty . \tag{2.6}$$

The strategy is to define a data-dependent threshold $\hat{c}_{M,N}(\alpha)$ and reject the null hypothesis whenever $D_{M,N} > \hat{c}_{M,N}(\alpha)$. Consider the pooled data $(Z_1, \dots Z_{M+N})$ $= (X_1, \dots, X_M, Y_1, \dots, Y_N)$, and its empirical measure H_{N+M}. Let $(\hat{Z}_1, \dots \hat{Z}_{M+N})$ be a bootstrap sample drawn from H_{M+N} and consider the bootstrap empirical measures

$$\hat{P}_M = \frac{1}{M} \sum_{i=1}^{M} \delta_{\hat{Z}_i} \qquad \text{and} \qquad \hat{Q}_N = \frac{1}{N} \sum_{i=1}^{N} \delta_{\hat{Z}_{M+i}}. \tag{2.7}$$

We can define

$$\hat{D}_{M,N} := \sqrt{\frac{MN}{M+N}} \left\| \hat{P}_M f - \hat{Q}_N f \right\|_\infty , \tag{2.8}$$

as well as

$$\hat{c}_{M,N}(\alpha) = \inf \left\{ t,\ \mathbb{P}\left(\left\| \hat{D}_{M,N} f \right\|_\infty > t \ \middle|\ Z_1, \dots, Z_{N+M} \right) \leq \alpha \right\}, \tag{2.9}$$

for $\alpha \in (0, 1)$. Note that $\hat{c}_{M,N}(\alpha)$ can be estimated with Monte-Carlo simulations. Using $\hat{c}_{M,N}(\alpha)$ as the threshold to accept or reject H_0 leads to a consistent test.

Theorem 2.6 (Section 3.7.2, [VDVW96])**.** *Assume that $\mathcal{F}$ is Donsker with respect to both P and Q and that $\|Pf\|_\infty$ and $\|Qf\|_\infty$ are finite. Furthermore assume that $M/(M+N) \to \lambda \in (0, 1)$. Then the test that rejects H_0 whenever $D_{M,N} > \hat{c}_{M,N}(\alpha)$ is consistent, in the sense that the asymptotic level is α and under any alternative verifying $\|Pf - Qf\|_\infty > 0$, $\mathbb{P}\left(D_{M,N} > \hat{c}_{M,N}(\alpha)\right) \to 1$.*

Intuitively, under H_0, as both the process defining $D_{M,N}$ and the one defining $\hat{D}_{M,N}$ given $Z_1, \dots, Z_{N+M}$ converge to the same Gaussian process, we can sample from the distribution of the latter (randomness coming only from resampling) to estimate the quantile of the former. This theorem ensures the appropriate asymptotic behavior of the two-sample test, both in terms of level and power.

2.4 Neural Networks

Neural Networks are now playing an essential role in machine learning. They are used in various applications and are often state-of-the-art on numerous problems. In this section, we introduce the concept of neural networks in general and specify two major architectures used in the following chapters: dense neural networks and convolutional neural networks.

2.4.1 Dense neural networks

Put in mathematical terms, a neural network is a function $f : R^d \to R^K$ defined as a composition of specific parametric functions $(f_l)_{l\in\{1,\dots,L\}}$:

$$f = f_L \circ f_{L-1} \circ \cdots \circ f_2 \circ f_1.$$

For each $l < L$, f_l is the composition of an affine function with matrix $W_l \in \mathbb{R}^{d_l \times d_{l+1}}$ and vector $b_l \in R^{d_{l+1}}$ and a non-linear function σ_l, called activation function, such that

$$x_{l+1} = f_l(x_l) = \sigma_l(W_l \cdot x_l + b_l). \tag{2.10}$$

Each vector x_l for $1 \leq l \leq L+1$ is called a layer, and each coordinate is called a neuron. We see from (2.10) that each neuron depends on all neurons of the previous layer, hence the denomination: *dense neural network* (DNN). See Figure 2.3 for a illustration of a DNN.

We often choose σ_l to be the ReLU function, that is, for $z = (z_i)_{i\in\{1,\dots d_{l+1}\}} \in \mathbb{R}^{d_{l+1}}$, $\sigma_l(z) = (\max(0, z_i))_{i\in\{1,\dots d_{l+1}\}}$. For classification tasks, the final function f_L is usually taken as the *softmax* function :

$$softmax(x) = \left(x_i / \sum_{j=1}^{K} x_j \right)_{i\in\{1,\dots,K\}}.$$

This way, the output of the neural network can be understood as a probability distribution over the K labels, and the predicted label is the label with maximal probability :

$$y = \underset{k\in\{1,\dots,K\}}{\operatorname{argmax}} f(x)_k.$$

For regression tasks, the final activation function is often either the linear activation (*i.e.* the identity) or, for normalized data in $[0,1]^K$ (as images), the logistic function $x \to 1/(1+e^{-x})$.

Remark 2.4. In the context of classification, the maximal value $\max_k f(x)_k$ is called the *confidence* of the network. For a given dataset $(X_1, \dots, X_N)$ the mean confidence of the network is measured by $N^{-1}\sum_i \max_k f(X_i)_k$. It is supposed to provide some information about the quality of the network predictions. However, as we will discuss in Chapter 8, networks tend to be over-confident even when performing poorly [NYC15, AM18, BR18].

The success of this structure for f is that the weights W_l and biases b_l can be updated in order to optimize a certain criterion during the training phase. Indeed, the chain rule allows to compute gradients easily and optimize the parameters of the network. This optimization step is what we call *learning*. This optimization algorithm is called the *back-propagation* as we update the parameters in backward order starting from function f_{L-1} to f_{L-2} until f_1, leading to a fast and efficient optimization procedure. For a training data set $((X_1, Y_1), \dots, (X_N, Y_N))$ where the X_i are the data to process and the Y_i are the targets (*e.g.* the Y_i could be labels

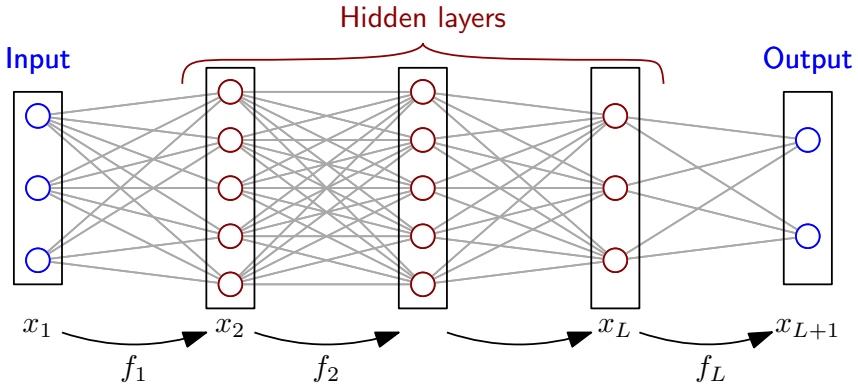


Figure 2.3: Example of a dense neural network structure.

in the context of classification), a global loss is defined as $\sum_i \mathcal{L}(f(X_i), Y_i)$ where $\mathcal{L}$ is a loss function measure the error of the prediction $f(X_i)$ with respect to the true value Y_i. In classification, a recurrent choice for $\mathcal{L}$ is the cross-entropy defined as follows. If $f(X_i)$ is a probability vector (for example, the image of the softmax function) such that $f(X_i)_k$ can be interpreted as the likelihood that X_i has label k, and if Y_i is the true label of X_i then

$$\mathcal{L}(f(X_i), Y_i) = -\log\left(f(X_i)_{Y_i}\right).$$

So it is minimal when $f(X_i)$ has the Y_i coordinate equals to 1 and all the others equal 0. For regression tasks, a classical choice of loss is the *mean square error* where $L(y, y') = K^{-1} \sum_{1 \leq k \leq K} (y_k - y'_k)^2$ for $y, y' \in \mathbb{R}^K$.

2.4.2 Activation graphs

Introduced by [GS17, GSH19], activation graphs are a way to encode how a neural network processes data. We will focus on activation graphs computed on forward architecture with dense layers. While this may seem reductive, note that most complex forward architectures usually include fully connected layers before the output layer.

For a given sequence of consecutive dense layers and a given input, the activation graph is a weighted undirected graph with a vertex set corresponding to the neurons of these layers. So the activation graphs are supported by the same vertex set, only the edge weights vary with the data. We now explain how to construct these activation graphs on a full DNN for simplicity. However, it is also possible to define them just from a part of the network.

Consider a fixed DNN with L layers and an input $x \in \mathbb{R}^d$. As the NN is fixed, it comes with weight matrices W_l connecting the l-th layer to the $(l+1)$-th layer. The activation graph associated to x is defined by $\overline{G}(x) = (V, \overline{W}_x)$ where V is the set of nodes and $\overline{W}_x$ is the weight matrix. The node set is $V = V_1 \sqcup V_2 \sqcup \cdots \sqcup V_{L+1}$ where V_l is the set of neurons of the l-th layer. The activation graphs have the structure of the NN: the nodes from a given layer are only connected to the nodes of the previous and following layers. For a neuron $u \in V_l$ with activation $x_l(u)$ and

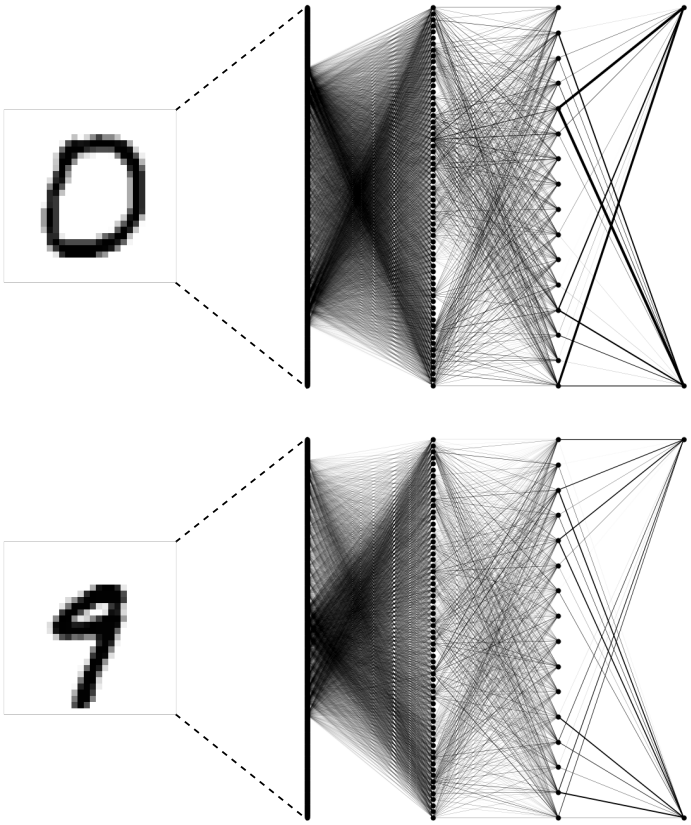


Figure 2.4: Two examples of activation graphs associated with two inputs from the MNIST dataset [LCB10]. The edge width is proportional to the edge weight.

a neuron $v \in V_{l+1}$, the weight between u and v in the activation graph is

$$\overline{W}_x(u, v) = W_l(u, v)\ x_l(u). \tag{2.11}$$

Note that $\overline{W}_x(u, v) = \overline{W}_x(v, u)$ as in equation (2.11), we multiply by the activation of the neuron in the previous layer, not by the activation of the first entry of $\overline{W}_x(\cdot, \cdot)$. So activation graphs are indeed undirected graphs. Figure 2.4 illustrates two activation graphs computed from the same DNN with two hidden layers on two different inputs.

2.4.3 Convolutional neural networks

Convolutional neural networks (CNNs) are a type of neural network that has successfully solved numerous image-driven learning tasks. Their definition differs slightly from the DNNs' one, but these differences significantly impact the NN behavior. In CNNs, a neuron from layer l also has its activation determined by neurons of layer $l-1$ through the composition of a linear operation and an activation function. However, it only uses a small proportion of these neurons. Moreover, while the source neurons change with the target neurons, the weights used in the linear combination are kept constants. Figure 2.5 illustrates how we pass from one layer

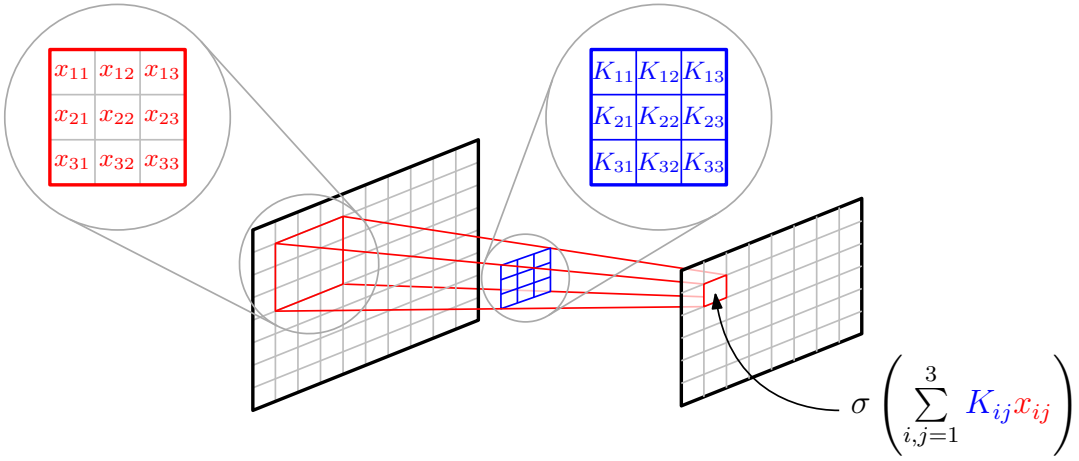


Figure 2.5: An illustration of a convolutional layer with a 3×3 kernel. The red beam scans the whole images. Each 3×3 sub-image is filtered by the fixed blue kernel to produce the image on the right.

to the next. As input data for CNNs are often images, we represent each layer as an array. The convolution of the previous layer by a fixed kernel, composed by a pixel-wise activation function, yields to the neuron activations of the next layer. The kernel weights are optimized during the training phase.

Keeping the kernel weights constant with respect to the target neurons drastically reduces the number of parameters to learn. It results in a more robust neural network, less prone to overfitting. Moreover, reducing the number of neurons involved in each linear combination considerably speeds up computations.

Classical CNN architectures. Each kernel, from its set of trained weights, is sensitive to a specific feature in the images. To build powerful CNNs, people classically learn several kernels and stack the corresponding convolutional layers together (see part (a) in Figure 2.6). In order to reduce dimension and avoid having too large layers, pooling layers are often introduced. The most common is the max-pooling layer. Each image is subdivided into small arrays (often 2×2-arrays), and only the largest value is conserved. For 2×2-arrays, this operation divide by four the number of neurons (see part (b) in Figure 2.6). Finally, after extracting features from convolutional and pooling layers, the learned representations of the input data pass through several dense layers (see part (c) in Figure 2.6).

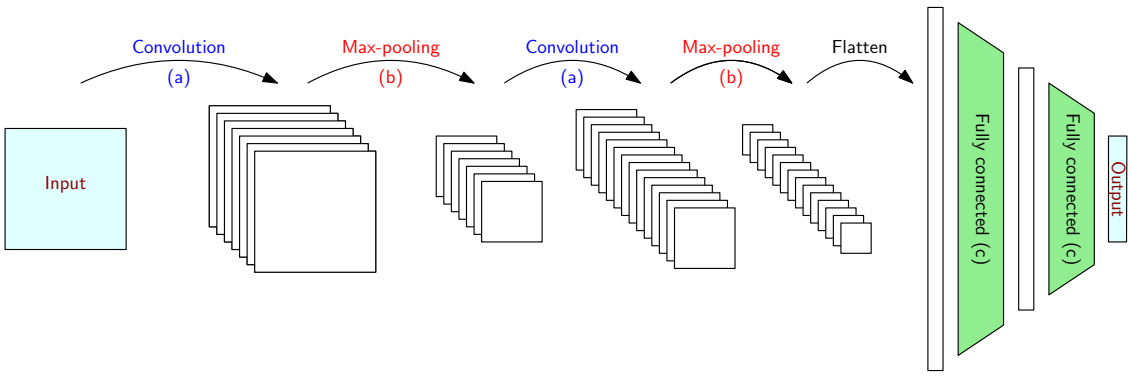


Figure 2.6: An example of a CNN architecture: a series of 7 convolution layers, a max-pooling layer (2×2), 2 convolution layers, a max-pooling layer, a flattening layer, and 2 dense layers.

Part I

Theory

Chapter 3

Multi-scale comparisons of graphs

In this chapter, we introduce two ways of comparing graphs based on heat diffusion. Moreover, to fully exploit the multi-scale properties of heat diffusion, we propose to study a new object that we call *distance process*.

First, we present the *heat kernel distance* (HKD), which compares heat kernels through the Frobenius norm. It can be computed for graphs of the same size and is meaningful whenever graphs can be aligned through to a known node correspondence (NC). Secondly, we introduce the *heat persistent distance* (HPD) that allows us to compare graphs without any assumption on size or NC. For that, we represent graphs by topological descriptors from topological data analysis and compare them through the so-called *Bottleneck distance*.

3.1 Heat diffusion

A powerful way to encode structural information about graphs is to use diffusion processes, like heat diffusion. When working with weighted graphs with non-negative weights, one can interpret weights as the thermal conductivity of edges, meaning that heat diffuses faster along edges with higher weights. Given initial conditions, the way heat is distributed over the nodes after a given time can be used to characterize and compare graphs [CL06, CH14, HGJ13, TMK+18]. Before recalling the basics of heat diffusion, we define some notations about sets of weighted graphs used in the following.

Notations. We consider two sets of weighted graphs. For $0 \leq w_{\min} \leq w_{\max}$, consider the following sets:

- $\mathcal{G}_n(w_{\min}, w_{\max})$ is the set of undirected weighted graphs of size n, without self-loop and whose weights are in $\{0\} \cup [w_{\min}, w_{\max}]$.
- $\mathcal{G}^n(w_{\min}, w_{\max})$ (with n as an exponent) is the set of graphs of size at most n, without self-loop and whose weights are in $\{0\} \cup [w_{\min}, w_{\max}]$.
  So $\mathcal{G}^n(w_{\min}, w_{\max}) = \cup_{1 \leq i \leq n} \mathcal{G}_i(w_{\min}, w_{\max})$.

The special case of unweighted graphs corresponds to $w_{\min} = w_{\max} = 1$. For clarity in the notation, we will remove $w_{\min}$ and $w_{\max}$ whenever there is no ambiguity.

Let G be a graph in $\mathcal{G}_n$. We assume that its vertex set is $V = \{1, \dots, n\}$ and that the graph structure is encoded in the weight matrix W. The degree matrix D is the diagonal matrix whose entries are the node degree: $D_{i,i} = \sum_j W_{i,j}$, for all $i \in V$. Recall that $L := D - W$ is the unnormalized Laplacian matrix of G. For $t \geq 0$, let $u_t \in \mathbb{R}^n$ be the vector whose i-th coefficient represents the amount of heat of node i at time t. Then, u_t follows the heat equation :

$$\forall t \geq 0, \quad \frac{d}{dt} u_t = -L u_t.$$

The matrix e^{-tL} is called the *heat kernel* at time t. It describes how heat diffuses in the graph as it generates the solution for any given initial condition u_0: $u_t = e^{-tL} u_0$. Hence, we can see that the j-th column of e^{-tL} contains the amount of heat of each node at time t, when a single unit of heat was placed at node j at time $t = 0$.

Note that the diffusion time t acts as a scaling parameter. For t close to 0, the Taylor expansion $e^{-tL} = I - tL + o(t)$ indicates that the heat diffusion is only determined by the direct neighborhood of nodes. However, for larger values of t, larger and possibly more complex structures are involved, taking into account topological properties of the graph.

As a result, comparing graphs using heat diffusion is appealing as it allows to analyze graphs at different scales by looking at different diffusion times. However, the choice of relevant and informative diffusion times is essential. This is why, instead of arbitrarily choosing t or developing data-driven procedures to select t, we introduce the *distance processes*. The idea is to record all heat-diffusion-based distances for all t in a continuous range of diffusion times.

In the following sections, we present the two distances that we consider and introduce their respective distance processes.

3.2 Heat Kernel Distance

Assume that we know the NC for graphs in $\mathcal{G}_n$. Hence we can align graphs by numbering the nodes such that the identity mapping gives the correspondences. This is equivalent to assume that all graphs in $\mathcal{G}_n$ are defined on the vertex set $\{1, \dots, n\}$. With this assumption, we can meaningfully compare adjacency matrices, Laplacian matrices, or heat kernels entry-wise. Here we compare graphs through their heat kernels. For two graphs G and G', define their Heat Kernel Distance (HKD) at time t by

$$D_t((G, G')) = \|e^{-tL} - e^{-tL'}\|_F, \tag{3.1}$$

where L and L' are the Laplacian matrices of G and G' respectively, and $\|\cdot\|_F$ denotes the Frobenius norm.

Considering the heat kernels instead of the adjacency or Laplacian matrices allows for a more robust notion of distance. It is also more aware of relevant structures

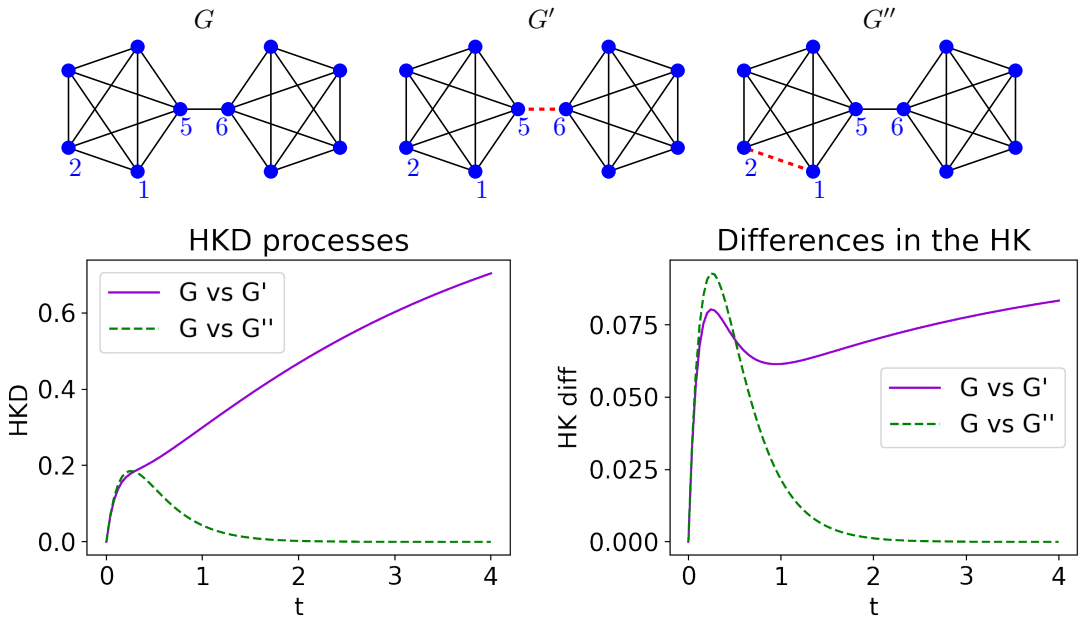


Figure 3.1: Top row: representation of three unweighted graphs G, G', G''. Removing the edge $\{5,6\}$ to G gives G'. Removing $\{1,2\}$ to G gives G''.
Bottom row: (left) we plot $t \mapsto D_t((G,G'))$ and $t \mapsto D_t((G,G''))$. (right) We plot of $t \mapsto \left|(e^{-tL})_{5,6} - (e^{-tL'})_{5,6}\right|$ and $t \mapsto \left|(e^{-tL})_{1,2} - (e^{-tL''})_{1,2}\right|$.

of graphs. The example displayed in Figure 3.1 shows that, for large enough values of t, removing an edge joining two dense components and removing an edge inside one of the components is seen differently by the HKD. This would not be the case if graphs were compared through their adjacency or Laplacian matrices. The resulting distances would be the same, regardless of which edge is removed. Moreover, this example illustrates the multi-scale property of HKDs. Indeed, at the local scale (small t), both scenarios are interpreted similarly: the HKD only detects that an edge has been removed. However, at a larger scale (say t larger than 0.5), the functional role of the removed edge for the graph connectivity is captured by the HKD. The loss of connectedness results in a much larger distance than removing an edge inside an already densely connected component.

The HKD was introduced by [HGJ13]. Although the authors chose to turn the HKD into a parameter-free notion of distance and rather defined the *Graph Diffusion Distance* by $\max_t D_t((G,G'))$. This approach has the drawback of comparing different pairs of graphs at different diffusion times. To avoid this issue and take advantage of the multi-scale benefits of heat kernels, we choose to use the whole function $t \to D_t((G,G'))$. More precisely, considering a probability distribution P on $\mathcal{G}_n \times \mathcal{G}_n$ and a random pair of graphs $(G,G') \sim P$, we are interested in the stochastic process $\{D_t((G,G')),\ t \in [0,T]\}$ for some $T > 0$. That is, the process obtained by evaluating the random pair of graphs (G,G') by the functions of the family $\mathcal{F}_{HKD} := \{D_t,\ t \in [0,T]\}$. This framework corresponds to the general framework of *empirical processes*.

3.3 Heat Persistence Distance

In practice, graphs can not always be aligned, for NCs are not always known. Additionally, one may be interested in comparing graphs of different sizes. In these cases, HKD is either not relevant (as coordinate-wise matrix comparisons are meaningless) or can not be defined. To circumvent these issues and following ideas from [CCI+20], we define the Heat Persistence Distance (HPD) by using extended persistence diagrams computed with the Heat Kernel Signature (HKS). These persistence diagrams can be compared with the Bottleneck distance d_B without any assumption on graph sizes and node identification.

The HKS was first introduced by [SOG09] for the study of shapes. Here we restrict ourselves to the definition of the HKS on graphs of [HRG14]. For a graph G in $\mathcal{G}_n$, with vertex set $V = \{1, \dots, n\}$ and Laplacian matrix L, the HKS at time t is the function $h_t(G) : V \to \mathbb{R}$ such that

$$h_t(G)(i) = (e^{-tL})_{i,i}, \quad 1 \leq i \leq n.$$

Intuitively, the image of $h_t(G)$ corresponds to the diagonal of the heat kernel e^{-tL}. Hence, $h_t(G)(i)$ represents the remaining amount of heat at node i after a diffusion time t, when a single unit of heat was placed at node i at time $t = 0$. For each value of t, the HKS provides a function on the vertices of a graph.

From G equipped with $h_t(G)$ and using the theory of extended persistence, we can compute topological descriptors of G called persistence diagrams (PDs). By looking at the increasing families of sublevel and superlevel subgraphs of G given by h_t, we can record levels at which topological features, like connected components or cycle, appear or disappear. Then we construct four types PDs. They are multi-sets of points in $\mathbb{R}^2$ where each point corresponds to a topological feature and its coordinates to specific levels associated with this feature. Each type of persistence diagram encodes a type of topological feature that can be seen as downward branches, upward branches, connected components, and loops, where the up/down orientation is taken with respect to $h_t(G)$. Generically, we denote any of these four diagrams by $Dg(G, h_t(G))$. For more details about the construction of PDs for graphs, we refer the interested reader to Section2.2.3 or [CCI+20].

Recall that the space of PDs can be equipped with the *Bottleneck distance* d_B. For μ and ν two PDs, and $\Delta := \{(a, a), a \in \mathbb{R}\}$ the diagonal line in $\mathbb{R}^2$, we denote by $\Pi(\mu, \nu)$ the set of bijections from $\mu \cup \Delta$ to $\nu \cup \Delta$. Then, d_B is defined by

$$d_B(\mu, \nu) := \inf_{\pi \in \Pi(\mu,\nu)} \sup_{x \in \mu \cup \Delta} \|x - \pi(x)\|_\infty,$$

where $\|y\|_\infty := \max\{|y_1|, |y_2|\}$ for $y = (y_1, y_2) \in \mathbb{R}^2$.

Finally, the HPD at time t between two graphs G, G' in $\mathcal{G}^n$ is defined by

$$H_t((G, G')) = \max_{Dg} d_B(Dg(G, h_t(G)), Dg(G', h_t(G'))), \tag{3.2}$$

where the maximum is taken over the four diagram constructions. By using PDs and the Bottleneck distance, we switch from node-based representations of graphs

to comparable topological summaries, requiring no assumption on graph sizes and NC anymore. Moreover, the HPD still benefits from the multi-scale properties of heat diffusion as we choose to filter graphs with the HKS.

In all simulations of Part II, extended persistence diagrams are computed by following the approach of [CCI+20] and using the Gudhi library [MBGY14].

Similarly to $\mathcal{F}_{HKD}$, we define the family $\mathcal{F}_{HPD} := \{H_t,\ t \in [0,T]\}$ in order to study the induced stochastic process : $\{H_t((G,G')),\ t \in [0,T]\}$, for some random pair of graphs $(G,G') \in \mathcal{G}^n \times \mathcal{G}^n$. Note that for HPD processes we take graphs in $\mathcal{G}^n$, to allow them to be of different sizes.

Analyses. Instead of directly analyzing the statistical properties of the processes associated with $\mathcal{F}_{HKD}$ and $\mathcal{F}_{HPD}$, we propose a more general study in Chapter 4 where we consider general Lipschitz-continuous empirical processes indexed by a real-parameter.

The results derived in this chapter are then specified in the case of the HKD and HPD processes in Chapter 5. Moreover, from these results, we propose statistical applications for graph data.

Chapter 4

Statistical properties of real-valued continuous empirical processes indexed by a real parameter

In this chapter, we properly introduce the general framework for continuous empirical processes, then show that uniform boundedness and Lipschitz-continuity imply a functional central limit theorem, as well as a Gaussian approximation. Finally, we derive consequences for the construction of confidence bands and two-sample tests.

4.1 Introduction

Let I be a compact interval of $\mathbb{R}$ and $\mathcal{C}(I)$ the space of continuous real-valued functions on I endowed with the metric induced by the uniform norm : $\|h\|_\infty = \sup_{t\in I} |h(t)|$. Consider a measurable space $(\mathbb{X}, \mathcal{X})$. For all measures Q on $(\mathbb{X}, \mathcal{X})$ and all measurable functions $g : \mathbb{X} \to \mathbb{R}$, we denote the integral of g with respect to Q by $Qg := \int_{\mathbb{X}} g(x)dQ(x)$. Consider a probability measure P on $(\mathbb{X}, \mathcal{X})$ and $\mathcal{F} := \{f_t,\ t \in I\}$, a family of measurable real-valued functions on $\mathbb{X}$ indexed by I. For all $x \in \mathbb{X}$, define $f(x)$ as the function $t \to f_t(x)$, and assume that $f(x) \in \mathcal{C}(I)$. Therefore, given a random variable X with distribution P, one can equivalently see $\{f_t(X),\ t \in I\}$ either as a random process or as $f(X)$ a random variable in $\mathcal{C}(I)$.

Given an i.i.d sample $X_1, \ldots, X_N$ drawn under P, we are interested in the statistical properties of the mean function $N^{-1}\sum_i f(X_i)$ and its centered and scaled version $N^{-1/2}\left(\sum_i f(X_i) - Pf\right)$. Equivalently, one can study the empirical processes $\{P_N f_t,\ t \in I\}$ and $\{G_N f_t,\ t \in I\}$, where $P_N = N^{-1}\sum_i \delta_{X_i}$, and $G_N = \sqrt{N}(P_N - P)$. In the following, we see random processes and random functions as the same objects.

When studying the statistical properties of $\mathcal{F}$, one might be interested in a functional version of the central limit theorem. This corresponds to the concept of Donsker families.

Definition 4.1 (P-Donsker)**.** For P a probability measure on $(\mathbb{X}, \mathcal{X})$, the family $\mathcal{F}$ is called P-Donsker if the process $\{G_N f_t,\ t \in I\}$ converges in distribution to

the centered Gaussian process $\{\mathbb{G}_t,\ t \in I\}$ with covariance function κ defined as $\kappa_{s,t} := Pf_tf_s - Pf_tPf_s$ for all $t, s \in I$..

Here convergence in distribution means weak convergence in the space $\mathcal{C}(I)$. That is, for all continuous bounded functions $h : \mathcal{C}(I) \to \mathbb{R}$, $\lim_{N\to\infty} \mathbb{E}\left[h\left(G_N f\right)\right] = \mathbb{E}\left[h\left(\mathbb{G}\right)\right]$, where the expectation on the left-hand side is taken over the distribution of the sample $X_1, \ldots, X_N$, and the one on the right-hand side is taken over the distribution of the Gaussian process $\mathbb{G}$.

Going further into the statistical analysis of $\mathcal{F}$, one might want to assess the speed at which $\{G_N f_t,\ t \in I\}$ converges in distribution to $\{\mathbb{G}_t,\ t \in I\}$. This can be done by proving Gaussian approximation results.

Definition 4.2 (Gaussian Approximation)**.** Let $(r_N)_{N\geq 1}$ be a vanishing sequence of positive real numbers. We say that the process $\{G_N f_t,\ t \in I\}$ admits a *Gaussian approximation with rate* r_N, if for all $\lambda > 1$, there exist a constant ρ and a rank N_0, such that for all $N \geq N_0$ one can construct on the same probability space both the sample $X_1, \ldots, X_N$ and a version $\mathbb{G}^{(N)}$ of the Gaussian limit process $\mathbb{G}$ verifying

$$\mathbb{P}\left(\left\|G_N f - \mathbb{G}^{(N)}\right\|_\infty > \rho.r_N\right) \leq N^{-\lambda}.$$

Note that if $\{G_N f_t,\ t \in I\}$ admits a Gaussian approximation with rate r_N, applying the Borel-Cantelli Lemma would yield to $\left\|G_N f - \mathbb{G}^{(N)}\right\|_\infty = O(r_N)$ *almost surely.*

4.2 Donsker theorem and Gaussian approximation

Before stating the central theorem of this chapter, we introduce two assumptions on $\mathcal{F}$.

(L) - There exists $k > 0$ such that for all $x \in \mathbb{X}$ the function $t \to f_t(x)$ is k-Lipschitz continuous on I, meaning that for all $t, s \in I$

$$|f_t(x) - f_s(x)| \leq k\,|t - s|\,.$$

(B) - $\mathcal{F}$ is uniformly bounded. That is, there exists a constant $M > 0$ such that for all $x \in \mathbb{X}$ and for all $t \in I$,

$$|f_t(x)| \leq M.$$

Remark 4.1. Assumptions **(L)** and **(B)** are quite simple and can be easily checked for a given process. Thus the following results based on these assumptions will be general and could be easily applied.

We show that the above assumptions are sufficient to obtain a Donsker theorem and a Gaussian approximation result.

Theorem 4.1. *Assume that $\mathcal{F}$ verifies assumptions* ***(L)*** *and* ***(B)*** *. Then for any probability measure P on $(\mathbb{X}, \mathcal{X})$, $\mathcal{F}$ is P-Donsker and $\{G_N f_t,\ t \in I\}$ admits a Gaussian approximation with rate $r_N = N^{-1/7} \log N^{9/14}$.*

This theorem provides a functional central limit theorem as well as some information about the rate of convergence. It allows us to derive, in the next chapter, more practical consequences. Namely, it validates the construction of consistent confidence bands and consistent two-sample tests using bootstrap methods.

The proof of Theorem 4.1 can be found in Appendix 6.1. The Donsker property is proved by standard arguments of tightness, allowing to pass from a multivariate central limit theorem to a functional one. The proof of Gaussian approximation is based on a result from [BM06] requiring technical work and a more complex formalism. Essentially, by using Lipschitz-continuity, we control the covering number of $\mathcal{F}$, a quantity that indicates the complexity of the family. Even if Theorem 4.1 is a consequence of [BM06], working in the special case of continuous processes indexed by a real parameter allows us to present a more accessible result. Moreover, although the main interest of this work is its application to the HKD and HPD processes, we believe that the framework presented here is simpler and that this theorem could easily be applied to other processes.

Remark 4.2. Note that the rate in the Gaussian approximation is independent of the parameters k and M appearing in assumption **(L)** and **(B)** . This will be interesting later, as in our applications k and M will be dependent on the graph sizes. However, in the Gaussian approximation result, the constant ρ multiplying the rate in Definition 4.2 depends on those parameters k and M. The next result shows that this dependency is polynomial.

Theorem 4.2. *Assume that $\mathcal{F}$ verifies assumptions* ***(L)*** *and* ***(B)*** *. For all $\lambda > 1$, and for all $N > N_0 := 3 \vee N_1 \vee N_2 \vee N_3 \vee N_4 \vee N_5$, we can construct on the same probability space both the sample $X_1, \dots, X_N$ and a version $\mathbb{G}^{(N)}$ of the Gaussian limiting process $\mathbb{G}$ verifying*

$$\mathbb{P}\left(\left\| G_N f - \mathbb{G}^{(N)} \right\|_\infty > \rho N^{-\frac{1}{7}} \log N^{\frac{9}{14}} \right) \leq 7N^{-\lambda},$$

where the constant ρ is given by

$$\rho = B_1 + B_2\sqrt{\lambda + 1} + B_3 M (kT + M)^{5/2} (\lambda + 2/7),$$

with B_1, B_2, B_3 absolute constants, T the length of the interval I and the ranks

N_1, N_2, N_3, N_4 and N_5 are given by

$$
\begin{aligned}
N_1 &= 2(M \vee e)^7 \left(\log\left(2(M \vee e)^7\right) - 1\right) \\
N_2 &= \left(B_4 M^2 \left(\log\left(B_4 M^2\right) - 1\right)\right)^{7/5} \\
N_3 &= C_1(kT + M)^2 \left(2(\lambda+1)\left(\log\left(2(\lambda+1)(C_1(kT+M)^2)^{1/(\lambda+1)}\right) - 1\right)\right)^{\lambda+1} \\
N_4 &= 2\left(B_5 \frac{\lambda}{\sqrt{\lambda+1}} M\right)^{14/5} \left(\log\left(2\left(B_5 \frac{\lambda}{\sqrt{\lambda+1}} M\right)^{14/5}\right) - 1\right) \\
N_5 &= \left(2(\lambda+1)\left(\log(2(\lambda+1)) - 1\right)\right)^{\lambda+1},
\end{aligned}
$$

where B_4 and B_5 are absolute constants.

Remark 4.3. The above result is valid for

$$N > N_0 = 3 \vee N_1 \vee N_2 \vee N_3 \vee N_4 \vee N_5 = \mathcal{O}\left(M^{7+u} + (kT+M)^{2+u}\right),$$

for any $u > 0$. And the constant ρ is polynomial in kT and M.

These results, combined with the rate independent of k and M, will provide hints on the validity of the asymptotic methods that we will derive in the next chapters.

Chapter 5

Application to confidence bands and two-sample tests for graphs

Using the results of the previous chapter, we prove in Section 5.1 that the HKD and HPD processes are Donsker and admit a Gaussian approximation. Direct applications of these results allow us to propose consistent construction of confidence bands around empirical means and two-sample tests for samples of pairs of graphs. In Section 5.2, we then refine the two-sample test construction to present a two-sample test for samples of graphs. We do so by introducing a pre-processing pairing phase. Finally, in Section 5.3, we propose a multiple testing approach by repeating this pairing phase several times to increase the two-sample test's power.

5.1 Applying the Donsker property

Using the results of the previous section, we are able to show that the HKD and HPD processes admit a functional version of the central limit theorem, as well as a Gaussian approximation. This is essentially based on the fact that they are uniformly bounded Lipschitz-continuous processes. From these results, we propose a procedure to construct consistent confidence bands. We also detail the construction of a two-sample test for samples of pairs of graphs based on either the HPD processes or the HKD processes when the graph sizes and NC allow their computation. We explicit these constructions and state consistency results.

5.1.1 Statistical properties

Let P be a probability distribution on $\mathcal{G}_n \times \mathcal{G}_n$, where $\mathcal{G}_n = \mathcal{G}_n(w_{\min}, w_{\max})$ is the set of undirected weighted graphs of size n, whose weights are in $\{0\} \cup [w_{\min}, w_{\max}]$, for $0 \leq w_{\min} \leq w_{\max}$. Let T be a positive real number and recall that $\mathcal{F}_{HKD} := \{D_t,\ t \in [0,T]\}$, where D_t denotes the HKD and is defined in (3.1). We will study the centered and rescaled empirical process $\{G_N D_t,\ t \in [0,T]\} = \{\sqrt{N}(P_N - P)D_t,\ t \in [0,T]\}$, where P_N is the empirical measure $N^{-1}\sum_i \delta_{(G_i,G'_i)}$ associated to a N-sample $((G_1, G'_1), \ldots, (G_N, G'_N))$ drawn under P. We now state the statistical results concerning the HKD process.

Theorem 5.1. *For all probability distributions P on $\mathcal{G}_n \times \mathcal{G}_n$, the family $\mathcal{F}_{HKD}$ is P-Donsker. That is, the process $\{\sqrt{N}(P_N - P)D_t,\ t \in [0,T]\}$ converges weakly to $\mathbb{G}$ in $\mathcal{C}([0,T])$, where $\mathbb{G} = \{\mathbb{G}_t,\ t \in [0,T]\}$ is a zero mean Gaussian process with covariance function $\kappa(t,s) = P(D_tD_s) - PD_t.PD_s$.*
Moreover, it admits a Gaussian approximation with rate $r_N = N^{-1/7}\log N^{9/14}$.

We have a similar result for HPD processes. Recall that for HPD processes we work in $\mathcal{G}^n = \mathcal{G}^n(w_{\min}, w_{\max}) = \cup_{1\leq i\leq n}\mathcal{G}_i(w_{\min}, w_{\max})$, which is the set of graphs of size at most n whose weights are in $\{0\} \cup [w_{\min}, w_{\max}]$. Let P be a probability distribution on $\mathcal{G}^n \times \mathcal{G}^n$. Let T be a positive real number and recall that $\mathcal{F}_{HPD} := \{H_t,\ t \in [0,T]\}$, where H_t denote the HPD and is defined in (3.2). Again, we study the empirical process $\{G_NH_t,\ t \in [0,T]\} = \{\sqrt{N}(P_N - P)H_t,\ t \in [0,T]\}$, where P_N is the empirical measure $N^{-1}\sum_i \delta_{(G_i,G'_i)}$ associated to a N-sample $((G_1,G'_1),\ldots,(G_N,G'_N))$ drawn under P.

Theorem 5.2. *For all probability distributions P on $\mathcal{G}^n \times \mathcal{G}^n$, the family $\mathcal{F}_{HPD}$ is P-Donsker. Thus, the process $\{\sqrt{N}(P_N - P)H_t,\ t \in [0,T]\}$ converges weakly to $\mathbb{G}$ in $\mathcal{C}([0,T])$, where $\mathbb{G} = \{\mathbb{G}_t,\ t \in [0,T]\}$ is a zero mean Gaussian process with covariance function $\kappa(t,s) = P(H_tH_s) - PH_t.PH_s$.*
Moreover, it admits a Gaussian approximation with rate $r_N = N^{-1/7}\log N^{9/14}$.

These theorems can be seen as functional central limit theorems for the HKD and HPD processes. They validate bootstrap methods for the construction of consistent confidence bands and consistent two-sample tests. Gaussian approximations strengthen these results and provide information about the speed of convergence to Gaussian processes. Note that this rate is independent of the graph size or maximal graph size n. This could indicate that even in the case of finite samples with graphs of reasonable sizes, methods based on the Donsker property could work well.

The proof of Theorem 5.1 and Theorem 5.2 can be found in Appendix 6.3. We start by showing that under the hypothesis of bounded weights on the graph edges, the HKD and HPD process are uniformly bounded and Lipschitz-continuous, then the proofs are direct applications of Theorem 4.1.

5.1.2 Confidence band and consistency

Donsker results allow for the validity of the construction of consistent confidence bands using bootstrap methods. We first recall the general ideas for such constructions in the particular case of the HKD and HPD processes on a sample of pairs of graphs. Then, in Algorithm 1, we provide the pseudo-code to compute a confidence band given a sample of pairs of graphs. As the construction is similar in the HKD or HPD case, we only focus on the HKD case.

Consider an *i.i.d* sample of pairs of graphs $((G_1,G'_1),\ldots(G_N,G'_N))$ following a distribution P supported on $\mathcal{G}_n \times \mathcal{G}_n$ and let P_N be its empirical probability measure. For a given α in $(0,1)$, we want to find a band around the empirical mean process

$\{P_N D_t,\ t \in [0,T]\}$ such that asymptotically, with probability greater than $1-\alpha$, it contains the expected process $\{PD_t,\ t \in [0,T]\}$.

Recall that from the Donsker property of $\mathcal{F}_{HKD}$, the process $\{\sqrt{N}(P_N - P)D_t,\ t \in [0,T]\}$ converges weakly to a Gaussian process $\mathbb{G}$. Hence, a first idea would be to consider $c_\alpha := \inf\{u,\ \mathbb{P}\left(\|\mathbb{G}\|_\infty > u\right) \leq \alpha\}$ the upper α-quantile of the maximum of the Gaussian limit process. As a consequence of Theorem 4.1, we have

$$\lim_{N\to\infty} \mathbb{P}\left(\forall t, PD_t \in \left[P_N D_t - \frac{c_\alpha}{\sqrt{N}}, P_N D_t + \frac{c_\alpha}{\sqrt{N}}\right]\right) \geq 1-\alpha.$$

Unfortunately, as the distribution of $\mathbb{G}$ is unknown, c_α can not be directly computed. Instead, consider a bootstrap sample $((\hat{G}_1, \hat{G}'_1), \dots (\hat{G}_N, \hat{G}'_N))$, that is a sample drawn under P_N, and let $\hat{P}_N$ be its empirical probability measure. Consider the process $\{\hat{G}_N D_t, t \in [0,T]\}$ where $\hat{G}_N$ is the measure $\sqrt{N}(\hat{P}_N - P_N)$. As $\mathcal{F}_{HKD}$ is P-Donsker, Theorem 2.6 in [Kos08] ensures that $\{\hat{G}_N D_t, t \in [0,T]\}$ converges weakly to the same Gaussian process $\mathbb{G}$, given the data. Hence the upper α-quantile of $\left\|\hat{G}_N D\right\|_\infty$ denoted by $\hat{c}_\alpha$ provides an approximation of c_α. Thus, we can use $\hat{c}_\alpha$ to design a consistent confidence band around $P_N D_\cdot$:

$$\lim_{N\to\infty} \mathbb{P}\left(\forall t, PD_t \in \left[P_N D_t - \frac{\hat{c}_\alpha}{\sqrt{N}}, P_N D_t + \frac{\hat{c}_\alpha}{\sqrt{N}}\right]\right)) \geq 1-\alpha. \tag{5.1}$$

Remark 5.1. Recall that the problem with c_α was that it could not be computed. Here, $\hat{c}_\alpha$ is the quantile of $\left\|\hat{G}_N D\right\|_\infty$ given the data, so randomness only comes from drawing the bootstrap samples. Hence, $\hat{c}_\alpha$ can be estimated with Monte-Carlo simulations by drawing as many bootstrap samples as we want.

We now summarize in Algorithm 1 how to compute the confidence band.

Algorithm 1: Confidence Band

Data: a sample of pairs of graphs $((G_1, G'_1), \dots (G_N, G'_N))$
Input: an integer B and a level α.

for $i \in \{1, \dots, N\}$ **do**
  compute $d_i : t \mapsto D_t((G_i, G'_i))$;
end
compute $\overline{d}_N = N^{-1}\sum_i d_i$;
for $b \in \{1, \dots, B\}$ **do**
  draw with replacement a bootstrap sample $(\hat{d}_1^{(b)}, \dots \hat{d}_N^{(b)})$ out of $(d_1, \dots d_N)$;
  compute $\tilde{d}_N^{(b)} = N^{-1}\sum_i \hat{d}_i^{(b)}$;
  compute $t^{(b)} = \sqrt{N}\|\tilde{d}_N^{(b)} - \overline{d}_N\|_\infty$;
end
compute $\tilde{c}_\alpha$ the empirical upper α-quantile of $t^{(1)}, \dots, t^{(B)}$;

Result: Return the confidence band $\left[\overline{d}_N(t) - \frac{\tilde{c}_\alpha}{\sqrt{N}}, \overline{d}_N(t) + \frac{\tilde{c}_\alpha}{\sqrt{N}}\right]$ for all t.

By choosing B large enough, $\tilde{c}_\alpha$ can approximate, as closely as we want, the upper α-quantile $\hat{c}_\alpha$. According to (5.1), this means that for large enough B, Algorithm 1 produces a consistent confidence band.

5.1.3 Two-sample test and consistency

We now detail how to construct a two-sample test for sample of pairs of graphs in Algorithm 2. As previously, we present the HKD case as the HPD case is similar. This test follows a classical construction from Donsker processes. More insights can be found in the background chapter, Section 2.3.3.

Algorithm 2: Two-sample Test

Data: two samples of pairs of graphs
$((G_{1,1}, G'_{1,1}), \dots (G_{1,N}, G'_{1,N}))$ and $((G_{2,1}, G'_{2,1}), \dots (G_{2,M}, G'_{2,M}))$
Input: an integer B and a level α.

for $i \in \{1, \dots, N\}$ **do**
  compute $d_{1,i} : t \mapsto D_t((G_{1,i}, G'_{1,i}))$;
end
for $i \in \{1, \dots, M\}$ **do**
  compute $d_{2,i} : t \mapsto D_t((G_{2,i}, G'_{2,i}))$;
end
compute the empirical means $\overline{d}_1 = N^{-1} \sum_i d_{1,i}$ and $\overline{d}_2 = M^{-1} \sum_i d_{2,i}$;
compute the test statistic $T = \sqrt{NM(N+M)^{-1}} \|d_1 - d_2\|_\infty$;
for $b \in \{1, \dots, B\}$ **do**
  draw with replacement $(\hat{d}^{(b)}_{1,1}, \dots \hat{d}^{(b)}_{1,N})$ and $(\hat{d}^{(b)}_{2,1}, \dots \hat{d}^{(b)}_{2,M})$ out of $(d_{1,1}, \dots d_{1,N}, d_{2,1}, \dots d_{2,M})$;
  compute $\tilde{d}^{(b)}_1 = N^{-1} \sum_i \hat{d}^{(b)}_{1,i}$ and $\tilde{d}^{(b)}_2 = M^{-1} \sum_i \hat{d}^{(b)}_{2,i}$;
  compute $T^{(b)} = \sqrt{NM(N+M)^{-1}} \|\tilde{d}^{(b)}_1 - \tilde{d}^{(b)}_2\|_\infty$;
end
compute $\tilde{c}_\alpha$ the empirical upper α-quantile of $T^{(1)}, \dots, T^{(B)}$;

Result: Reject the null hypothesis if $T > \tilde{c}_\alpha$.

Once again, for a large enough B, $\tilde{c}_\alpha$ closely approximates $\hat{c}_\alpha$ the upper α-quantile of the random variable $T^{(1)}$ given the data. Recall that from Theorem 2.6, using $\hat{c}_\alpha$ as the rejection threshold yields to a control of the asymptotic level and power. That is

$$\lim \mathbb{P}_{H_0} \left(T > \hat{c}_\alpha \right) = \alpha,$$

and under any alternative verifying $\|P_1 D_\cdot - P_2 D_\cdot\|_\infty > 0$ where P_1 and P_2 are the distributions that generated the two samples,

$$\lim \mathbb{P}_{H_1} \left(T > \hat{c}_\alpha \right) = 1.$$

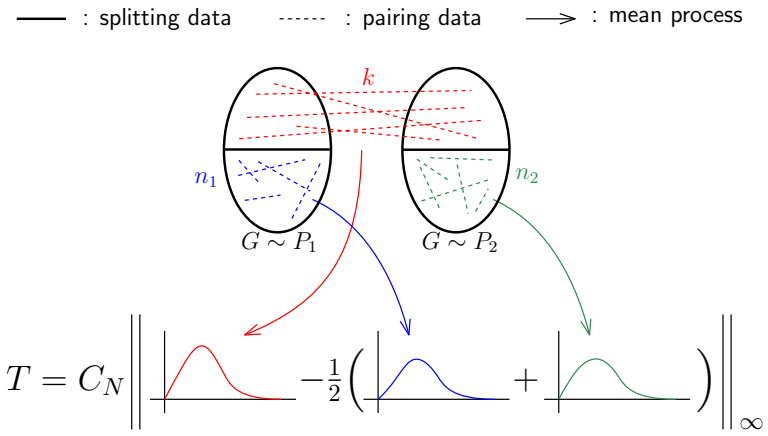


Figure 5.1: Pairing scheme for the two-sample test for samples of graphs.

The limits are for both N and M going to infinity under the condition that $N/(N+M) \to \lambda$, for a λ in $(0,1)$. Hence, for large enough B, Algorithm 2 produces an asymptotically valid two-sample test for samples of pairs of graphs.

5.2 Pairing procedure for two-sample test for samples of graphs

Let us consider two graph samples $D_1 = (G_1^1, \dots G_{N_1}^1)$ and $D_2 = (G_1^2, \dots G_{N_2}^2)$ drawn from the distributions P_1 and P_2, respectively. We want to use the HPD or HKD to construct a two-sample test. To do so, we need to construct pairs of graphs.

One way to do so is to split each data set into two groups at random and construct inter-sample pairs of graphs and intra-sample pairs of graphs while ensuring that each graph appears in at most one pair to guarantee independence properties.

Let us fix the integer k (growing with N_1 and N_2, *e.g.*, $k = \min(N_1, N_2)/2$), and assume that we shuffle D_1 and D_2. Now consider the following samples of pairs of graphs

$$\begin{aligned} D_{12} &= ((G_1^1, G_1^2), (G_2^1, G_2^2) \dots, (G_k^1, G_k^2)) \\ D_{11} &= ((G_{k+1}^1, G_{k+2}^1), (G_{k+3}^1, G_{k+4}^1) \dots, (G_{N_1-1}^1, G_{N_1}^1)) \\ D_{22} &= ((G_{k+1}^2, G_{k+2}^2), (G_{k+3}^2, G_{k+4}^2) \dots, (G_{N_2-1}^2, G_{N_2}^2)). \end{aligned} \tag{5.2}$$

Note that D_{12}, D_{11} and D_{22} are samples of independent pairs of graphs, although they are not independent conditionally on D_1 and D_2. Their respective sizes are k, $n_1 := (N_1 - k)/2$ and $n_2 := (N_2 - k)/2$. Without loss of generality, we will assume that $N_1 - k$ and $N_2 - k$ are even. We denote by Q_{12}^k, $Q_{11}^{n_1}$ and $Q_{22}^{n_2}$ the empirical distribution associated to D_{12}, D_{11} and D_{22}.

In the following, we assume that k, n_1, n_2 tend to infinity such that

$$\frac{4n_1n_2}{kn_1 + kn_2 + 4n_1n_2} \to \mu \; ; \quad \frac{kn_2}{kn_1 + kn_2 + 4n_1n_2} \to \nu \; ; \quad \frac{kn_1}{kn_1 + kn_2 + 4n_1n_2} \to \lambda,$$

hence verifying $\mu + \nu + \lambda = 1$.

Consider the test statistic T_{k,n_1,n_2} defined by

$$T_{k,n_1,n_2} = \sqrt{\frac{4kn_1n_2}{kn_1 + kn_2 + 4n_1n_2}} \quad \left\| Q_{12}^k D_\cdot - \frac{1}{2}\left(Q_{11}^{n_1} + Q_{22}^{n_2}\right) D_\cdot \right\|_\infty \tag{5.3}$$

Applying Theorem 5.2, one can check that under $H_0 : P_1 = P_2$, T_{k,n_1,n_2} converges in distribution to

$$\left\| \sqrt{\mu}\mathbb{G}_{12} + \sqrt{\nu}\mathbb{G}_{11} + \sqrt{\lambda}\mathbb{G}_{22} \right\|_\infty \overset{\mathcal{D}}{=} \|\mathbb{G}\|_\infty ,$$

where $\mathbb{G}$, $\mathbb{G}_{12}$, $\mathbb{G}_{11}$ and $\mathbb{G}_{22}$ are i.i.d Gaussian process with covariance function $\kappa(s,t) = P_1(D_sD_t) - P_1D_s.P_1D_t$. Hence using bootstrap methods, one can determine a rejection threshold for T_{k,n_1,n_2} that yields to a two-sample test with an asymptotic control of the type I error. Alternatively, bootstrap methods can provide consistent estimations of the p-values, see Remark 5.2. Details about this pairing procedure can be found in Algorithm 3.

Remark 5.2. In Algorithm 3, once the $T^{(1)}, \ldots, T^{(B)}$ are computed, instead of deciding to reject or not the null hypothesis, as presented in the algorithm, we can return an estimation of the p-value given by the proportion of $T^{(b)}$ greater than T : $p = \left|\{b \in \{1, \ldots B\},\ T^{(b)} > T\}\right| / B$.

Pairing by label. We propose a refinement of Algorithm 3 in the case where graphs come with labels (*e.g.* classification problem). Remark that the pairing procedure might sometimes pair graphs that are similar (with the same label) and sometimes pair graphs that can be very different (different labels). Intuitively, this increases the variance in our samples of distance processes and, as a result, could hide the fact that the graph distributions are different. To prevent this, we propose to be more careful when pairing graphs by pairing only graphs with the same label[1]. In practice, we consider the subsamples D_i^1 and D_i^2 of graphs that have label i, from D_1 and D_2 respectively. For each i, apply the pairing procedure described by equation (5.2) to the sample D_i^1 and D_i^2, to obtain samples of pairs of graphs D_i^{12}, D_i^{11} and D_i^{22}. In the end, D_{12}, D_{11}, and D_{22} are obtained by concatenating for all i, the samples D_i^{12}, D_i^{11} and D_i^{22}, respectively.

Limitations of the pairing procedure. While the procedure defined by Algorithm 3 yields a consistent two-sample test, forcing each graph to appear in at most one pair might lose part of the available information. Indeed, we can not consider all possible pairs. To circumvent this phenomenon, we propose to re-shuffle each data set and repeat the procedure several times while adopting a multiple testing approach. This approach is presented in the next section.

[1]In the case of a classification problem, we will use true labels if available and predicted labels otherwise.

Algorithm 3: Two-sample Test with a pairing scheme

Data: two samples of graphs
$((G_1^1, \dots G_{N_1}^1))$ and $(G_1^2, \dots G_{N_2}^2)$
Input: an integer B and a level α.

shuffle each data set (we keep the same indices for clarity);
set $k = \min(N_1, N_2)/2$;
set $n_1 = (N_1 - k)/2$ and $n_2 = (N_2 - k)/2$ (assuming n_1 and n_2 are integer);
apply the pairing scheme detailed in equations (5.2);
for $i \in \{1, \dots, k\}$ **do**
  compute $d_{12,i} : t \mapsto D_t((G_i^1, G_i^2))$;
end
for $i \in \{1 \dots, n_1\}$ **do**
  compute $d_{1,i} : t \mapsto D_t((G_{k+2i-1}^1, G_{k+2i}^1))$;
end
for $i \in \{1 \dots, n_2\}$ **do**
  compute $d_{2,i} : t \mapsto D_t((G_{k+2i-1}^2, G_{k+2i}^2))$;
end
compute the empirical means:
$\overline{d}_{12} = k^{-1} \sum_i d_{12,i}$, $\overline{d}_1 = n_1^{-1} \sum_i d_{2,i}$ and $\overline{d}_2 = n_2^{-1} \sum_i d_{2,i}$;
compute the test statistic T given by equation (5.3);
for $b \in \{1, \dots, B\}$ **do**
  draw with replacement $(\hat{d}_{12,1}^{(b)}, \dots \hat{d}_{12,k}^{(b)})$, $(\hat{d}_{1,1}^{(b)}, \dots \hat{d}_{1,n_1}^{(b)})$ and $(\hat{d}_{2,1}^{(b)}, \dots \hat{d}_{2,n_2}^{(b)})$ out of $(d_{12,1}, \dots d_{12,k}, d_{1,1}, \dots d_{1,n_1}, d_{2,1}, \dots d_{2,n_2})$;
  compute $\tilde{d}_{12}^{(b)} = k^{-1} \sum_i \hat{d}_{12,i}^{(b)}$, $\tilde{d}_1^{(b)} = n_1^{-1} \sum_i \hat{d}_{1,i}^{(b)}$ and $\tilde{d}_2^{(b)} = n_2^{-1} \sum_i \hat{d}_{2,i}^{(b)}$;
  compute $T^{(b)}$ from equation (5.3) using the bootstrap means $\tilde{d}_{12}^{(b)}$, $\tilde{d}_1^{(b)}$ and $\tilde{d}_2^{(b)}$.
end
compute $\tilde{c}_\alpha$ the empirical upper α-quantile of $T^{(1)}, \dots, T^{(B)}$;

Result: Reject the null hypothesis if $T > \tilde{c}_\alpha$.

5.3 Multiple testing approach

Assume that for data $D = (D_1, D_2)$, we can compute a test statistic $X = f(R(D))$ where R is a pairing phase done by re-sampling (or shuffling) and f is a deterministic function. Note that the previous section follows this framework, where R is the pairing defined by equations (5.2) and f is defined in equation (5.3). As this procedure might only capture a part of the available information in D, we want to repeat the pairing phase several times. Let denote by $R_1, \dots, R_m$ the independent pairing phases, leading to m test statistics $X_1, \dots X_m$ defined by

$$X_i = f(R_i(D)) \tag{5.4}$$

We apply standard multiple testing techniques to conclude whether we accept or reject $H_0 : P_1 = P_2$. Let us denote by $P_1, \dots, P_m$ the p-values associated to

the test statistics, and consider their ordering $P_{(1)} \leq \cdots \leq P_{(m)}$. If the test statistics $X_1, \dots X_m$ were independent, the Simes' test [See68, Sim86] would ensure that rejecting H_0 as soon as there exists an integer i such that

$$P_{(i)} \leq \frac{i \cdot \alpha}{m} \tag{5.5}$$

yields to a test of level at most α. Here $X_1, \dots X_m$ are *a priori* not independent, as they are all defined from D.

According to Theorem 1.2 of [BY01], under some condition on the dependency of the tests, the level of the Simes' test is still upper-bounded by α. We recall the proper dependency condition in Definition 5.1. First, consider the total order on $\mathbb{R}^m$, such that for $x, y \in \mathbb{R}^m$, $x \leq y$ if $x_i \leq y_i$, for all i. We say that a subset S of $\mathbb{R}^m$ is increasing if for $x \leq y$, $x \in S$ implies $y \in S$.

Definition 5.1. (PRDS) Let us consider a subset $I \subset \{1, \dots, m\}$. A random vector $X = (X_1, \dots X_m)$ is said to verify the *positive regression dependency on each one from the subset* I (PRDS) if for all increasing subset S and for each $i \in I$, $\mathbb{P}(X \in S | X_i = x)$ is nondecreasing in x.

However, we have not been able to prove that the PRDS condition is verified for the tests defined by (5.4) and (5.5). Nonetheless, we apply this multiple testing procedure to perform two-sample tests on graph samples. We repeat Algorithm 3 several times and compute the p-values as explained in Remark 5.2. Then deciding to reject or conserve the null hypothesis is made according to Sime's procedure in equation (5.5). Even without theoretical guarantee on the resulting test level, experiments suggest that it is still controlled, as desired, by α (see Figure 8.3 and Figure 8.4).

If one wishes to theoretically control the test level, we can modify the rejection rule in equation (5.5) and adopt the more conservative Benjamini-Yekutieli correction : reject H_0 as soon as there exists an integer i such that

$$P_{(i)} \leq \frac{i \cdot \alpha}{mH(m)},$$

where $H(m) := \sum_{1 \leq k \leq m} 1/k$ is the m-th harmonic number. [BY01] proved in their Theorem 1.3 that this procedure yields to a test with a level upper-bounded by α, without any assumption on the tests' dependency.

Chapter 6

Appendix

6.1 Proof of Theorem 4.1

We start by proving the Donsker property.

Proof. We need to prove the weak convergence of $\{G_N f_t,\ t \in I\}$ to the centered gaussian process $\mathbb{G}$. To do so, remark that assumption **(B)** ensures that second moments of $f(X)$ are finite. This allows to apply the multidimensional version of the central limit theorem to all finite-dimensional marginals of the random function $G_N f$. Hence, these finite-dimensional marginals converge in distribution to those of $\mathbb{G}$. To conclude to the weak convergence of the whole process, the sequence of random functions $\{G_N f,\ N \geq 1\}$ needs to be tight. According to [Bil13, Theorem 12.3], tightness of $\{G_N f,\ N \geq 1\}$ is implied by the following two conditions :

1. There exists $t_0 \in I$, such that $\{G_N f_{t_0},\ N \geq 1\}$ is tight.
2. There exist $\gamma \geq 0$, $\alpha > 1$ and a non-decreasing function $\psi : I \to \mathbb{R}$ such that $\forall t, s \in I$, $\forall N \geq 1$,

$$\mathbb{E}\left[|G_N f_t - G_N f_s|^\gamma\right] \leq |\psi(t) - \psi(s)|^\alpha .$$

Let us start by proving point 1. Let t_0 be any point in I. We have to prove that for all $\eta > 0$, there exists $\alpha > 0$ such that for all $N \geq 1$

$$P\left(|G_N f_{t_0}| \leq \alpha\right) > 1 - \eta. \tag{6.1}$$

If $Var(f_{t_0}(X)) = 0$, $\{G_N f_{t_0},\ N \geq 1\}$ is tight, since for all N, $G_N f_{t_0} = 0$ P-*a.s.*. Otherwise, fix $\eta > 0$. As the left hand side in (6.1) is non-decreasing with respect to α, we may as well show that there exists α such that for all N, $P\left(-\alpha < G_N f_{t_0} \leq \alpha\right) > 1 - \eta$. Following from **(B)** , $f_{t_0}(X)$ admits a third moment. Combined with the positive variance, we can apply the Berry-Essen Theorem : if F_N and ϕ denote the cumulative distribution functions of, respectively, $G_N f_{t_0}$ and a centered Gaussian variable of variance $Var(f_{t_0}(X))$, then

there exists a constant C such that for all $\alpha \in \mathbb{R}$ and all $N \geq 1$

$$|F_N(\alpha) - \phi(\alpha)| \leq \frac{C}{\sqrt{N}}.$$

Now we take N_η such that for all $N > N_\eta$, $C/\sqrt{N} < \eta/4$, and we choose α_η such that $\phi(\alpha_\eta) > 1 - \eta/4$. Then, for all $N > N_\eta$

$$\begin{aligned}
&P\left(-\alpha_\eta < G_N f_{t_0} \leq \alpha_\eta\right) \\
=&F_N(\alpha_\eta) - F_N(-\alpha_\eta) \\
=&\phi(\alpha_\eta) - \phi(-\alpha_\eta) + (F_N(\alpha_\eta) - \phi(\alpha_\eta)) - (F_N(-\alpha_\eta) - \phi(-\alpha_\eta)) \\
\geq&\phi(\alpha_\eta) - \phi(-\alpha_\eta) - 2\frac{C}{\sqrt{N}} \\
=&2\phi(\alpha_\eta) - 1 - 2\frac{C}{\sqrt{N}} \\
>&2\left(1 - \frac{\eta}{4}\right) - 1 - 2\frac{\eta}{4} = 1 - \eta
\end{aligned}$$

One can easily choose $\alpha > \alpha_\eta$ to extend the inequality $P\left(-\alpha_\eta < G_N f_{t_0} \leq \alpha_\eta\right) > 1 - \eta$ to all $N \geq 1$. This finishes the proof of point 1.

The proof of point 2, is a consequence of assumption **(L)** . For all $t, s \in I$, one has the following inequality

$$\begin{aligned}
&\mathbb{E}\left[|G_N f_t - G_N f_s|^2\right] \\
=&\mathbb{E}\left[|(f_t(X) - Pf_t) - (f_s(X) - Pf_s)|^2\right] \\
\leq&(2k\,|t - s|)^2 \\
=&(2kt - 2ks)^2.
\end{aligned}$$

This proves point 2. with $\gamma = 2$, $\alpha = 2$ and $\psi(u) = 2ku$, and finishes the proof of Theorem 4.1. ■

The proof of Gaussian approximation is based on a result from [BM06]. They derive rates for the Gaussian approximation of more general processes. They do so by approaching the process by finite-dimensional marginals and applying a multidimensional Gaussian approximation result. Controlling the covering number of the family (or equivalently its metric entropy) allows them to derive a good trade-off between a good approximation by the marginals and keeping the dimension small enough to obtain good rates in the multidimensional Gaussian approximation. For the sake of completeness, let us recall their result.

Let $\mathcal{M}$ be the set of all measurable real-valued functions on $(\mathbb{X}, \mathcal{X})$. The authors work with a centered and scaled empirical process $\{G_N f,\ \tilde{f} \in \tilde{\mathcal{F}}\}$ indexed by a general family $\tilde{\mathcal{F}} \subset \mathcal{M}$, not necessarily indexed by I. Their result shows that, under mild assumptions, this process can be approached by a centered Gaussian process $\tilde{\mathbb{G}}$ indexed by $\tilde{\mathcal{F}}$ with covariance $\mathbb{E}\left[\tilde{\mathbb{G}}(\tilde{f})\tilde{\mathbb{G}}(\tilde{g})\right] = P\tilde{f}\tilde{g} - P\tilde{f}P\tilde{g}$, for $\tilde{f}, \tilde{g} \in \tilde{\mathcal{F}}$. Let us define the covering number of $\tilde{\mathcal{F}}$. First, assume that there exists $\tilde{F} \in \mathcal{M}$

such that for all $\tilde{f} \in \tilde{\mathcal{F}}$ and all $x \in \mathbb{X}$, $|\tilde{f}(x)| \leq \tilde{F}(x)$. We say that $\tilde{F}$ is an *envelope* of $\tilde{\mathcal{F}}$. For all probability measures Q on $(\mathbb{X}, \mathcal{X})$, we consider the semi-metric $d_Q(g,h)^2 = \int (g-h)^2 dQ$, for $g, h \in \mathcal{M}$. Under d_Q, we define the ball of radius $\delta > 0$ centered in $h \in \mathcal{M}$ by $B_Q(h, \delta) := \{g \in \mathcal{M},\ d_Q(g,h) < \delta\}$. We also define $\tilde{F}_Q^2 := \int \tilde{F}^2 dQ$. For $\delta > 0$, let $N(\tilde{\mathcal{F}}, d_Q, \delta)$ be the size of the smallest finite subset $K \subset \mathcal{M}$ verifying that the union of balls $B_Q(h, \delta)$ for h in K covers $\tilde{\mathcal{F}}$. Finally, we set the covering number of $\tilde{\mathcal{F}}$ to be $N(\tilde{\mathcal{F}}, \delta) := \sup_Q N(\tilde{\mathcal{F}}, d_Q, \delta.\tilde{F}_Q)$, where the supremum is taken over all Q such that $0 < \tilde{F}_Q < \infty$.

Before stating the result of [BM06], consider these two basic assumptions.

(F.i) For some $M > 0$ and for all $\tilde{f} \in \tilde{\mathcal{F}}$, $\sup_{x \in \mathbb{X}} |\tilde{f}(x)| \leq M/2$.

(F.ii) The class $\tilde{\mathcal{F}}$ is point-wise measurable, i.e., there exists a countable subclass $\tilde{\mathcal{F}}_\infty$ of $\tilde{\mathcal{F}}$ such that we can find for any function $\tilde{f} \in \tilde{\mathcal{F}}$ a sequence of functions $\{\tilde{f}_m\}$ in $\tilde{\mathcal{F}}_\infty$ for which $\lim_{m \to \infty} \tilde{f}_m(x) = \tilde{f}(x)$ for all $x \in \mathbb{X}$.

Proposition 6.1. *[BM06, Proposition 1.] Assume that $\tilde{\mathcal{F}}$ verifies (F.i) and (F.ii). Take $\tilde{F} := M/2$ as the envelope of $\tilde{\mathcal{F}}$. Moreover, assume that there exist positive constants c_0 and v_0 such that $N(\tilde{\mathcal{F}}, \delta) \leq c_0 \delta^{-v_0}$. Then, for each $\lambda > 1$ there is a constant $\rho(\lambda)$ such that for each N, one can construct $X_1, \ldots, X_N$ and $\tilde{\mathbb{G}}^{(N)}$ such that*

$$\mathbb{P}\left(\sup_{\tilde{f} \in \tilde{\mathcal{F}}} \left| G_N \tilde{f} - \tilde{\mathbb{G}}^{(N)}(\tilde{f}) \right| > \rho(\lambda) N^{-\frac{1}{2+5v_0}} \log N^{\frac{4+5v_0}{4+10v_0}} \right) \leq N^{-\lambda}.$$

Let us now prove the Gaussian approximation in Theorem 4.1 by applying the previous proposition.

Proof. The proof consists in applying Proposition 6.1 to $\mathcal{F}$ with $v_0 = 1$. Clearly, assumption **(B)** implies (F.i). Since the paths $t \to f_t(x)$ are continuous for all $x \in \mathbb{X}$, taking $\mathcal{F}_\infty = \{f_t,\ t \in I \cap \mathbb{Q}\}$ gives (F.ii), where $\mathbb{Q}$ is the set of rational numbers.

Let us prove the upper-bound on the covering number. Let L be the length of I. From **(L)** , we know that there exists a constant k, such that $t \to f_t(x)$ is k-lipschitz, for all x. Considering a regular grid on I , $\min I = t_0 < t_1 < \cdots < t_q = \max I$. For all $t \in I$ there exists a integer j such that for all $x \in \mathbb{X}$, $\left| f_t(x) - f_{t_j}(x) \right| \leq k\,|t - t_j| \leq kL/q$. Hence, for all probability measures Q, $d_Q(f_t, f_{t_j}) \leq kL/q$, meaning that $N(\mathcal{F}, d_Q, kT/q) \leq q+1$. As this last inequality stands for all Q, we can find a constant $c_0 > 0$ such that $N(\mathcal{F}, \delta) \leq c_0 \delta^{-1}$, for all $\delta > 0$. This finishes the proof. ∎

6.2 Proof of Theorem 4.2

To prove Theorem 4.2, we need to refine the analysis made by [BM06] in the proof of their Proposition 1. More precisely, we first need to explicit the dependency of

the constant ρ and the rank N_0 from which the Gaussian approximation inequality holds, with respect to the parameters c_0, ν_0 and M. Then the proof will only boil down to studying these parameters and their dependency with k and M.

We first start by proving a tool lemma that allows to properly bound logarithmic terms by polynomial terms, in order to simplify our expressions.

Lemma 6.1. *Let α, β and C be positive constants.*

- *If $\beta C^{1/\beta}/\alpha \leq e$, then $x^\alpha \geq C \log^\beta x$, for all $x \geq 1$,*
- *If $\beta C^{1/\beta}/\alpha > e$, then $x^\alpha \geq C \log^\beta x$, for all x such that*

$$x \geq \left(\frac{2\beta C^{1/\beta}}{\alpha} \left(\log\left(2\beta C^{1/\beta}/\alpha\right) - 1 \right) \right)^{\beta/\alpha}.$$

Proof. We first start by proving the result for $\alpha = \beta = 1$. We study the function $h(x) = x - C\log(x)$ for $x \geq 1$. Note that this function is convex and admit a minimum in $x = C$ with value $C(1 - \log(C))$. Hence, if $C \leq e$, $h(x) \geq 0$ for all $x \geq 1$. Otherwise, from convexity the curve of h stays above the tangent of h in $x = 2C$. So if x_0 is the value at which the tangent crosses the y-axis, then for all $x \geq x_0$, $h(x) \geq 0$. Simple computations show that

$$x_0 = 2C\left(\log(2C) - 1\right).$$

This conclude the proof for the special case of $\alpha = \beta = 1$.

In general, for any positive α and β, and for $x \geq 1$, we have the following equivalence :

$$\begin{aligned} & x^\alpha \geq C \log(x)^\beta \\ \iff & x^{\alpha/\beta} \geq C^{1/\beta} \log(x) \\ \iff & X \geq \frac{\beta C^{1/\beta}}{\alpha} \log(X) \quad (\text{for } X = x^{\alpha/\beta}). \end{aligned}$$

So we can solve our problem in X, with a constant equals to $\beta C^{1/\beta}/\alpha$ and deduce a bound for x. This finishes the proof. ■

We now take as a starting point, a result from the proof of [BM06, Proposition 1.]. The authors show that for all N and for all $\varepsilon < 1/(M \vee e)$, verifying

$$\frac{\sqrt{N}\varepsilon}{2\sqrt{1 + 2\nu_0}\sqrt{\log(1/\varepsilon)}} > M, \tag{6.2}$$

we have the following inequality

$$\begin{aligned}
&\mathbb{P}\left(\left\|G_N f - \mathbb{G}^{(N)}\right\|_\infty > (D+\gamma_1+(1+A)\gamma_2)\varepsilon\sqrt{\log(1/\varepsilon)}\right) \\
\leq &\frac{C_1 c_1^2}{\varepsilon^{2\nu_0}} \exp\left(-\frac{\gamma_1 C_2\sqrt{N}}{c_1^{5/2}M}\varepsilon^{1+5\nu_0/2}\sqrt{\log(1/\varepsilon)}\right) \\
&+ 2\exp\left(-\frac{A_1\gamma_2\sqrt{N}}{M}\varepsilon\sqrt{\log(1/\varepsilon)}\right) + 4\exp(-A_5\gamma_2^2\log(1/\varepsilon))
\end{aligned} \tag{6.3}$$

where A, A_1, A_5, C_1 and C_2 are universal constants, γ_1 and γ_2 can be any positive constants, $c_1 = c_0 M^{\nu_0}$ and $D = (A_2 + 2A_4)\sqrt{2\nu_0}$ (with A_2 and A_4 universal constants).

The authors use this result by taking $\varepsilon = \varepsilon_N = ((\log N)/N)^{1/(2+5\nu_0)}$. Through Lemma 6.2 and Lemma 6.3, we derive explicit conditions on N to ensure that $\varepsilon_N < 1/(M \vee e)$ and that the condition in equation (6.2) are verified.

Lemma 6.2. *We have $\varepsilon_N < 1/(M \vee e)$, for*

$$N > 2(M \vee e)^{2+5\nu_0}\left(\log\left(2(M \vee e)^{2+5\nu_0}\right) - 1\right) =: N_1.$$

Proof of Lemma 6.2. By taking $\varepsilon_N = ((\log N)/N)^{1/(2+5\nu_0)}$, the condition $\varepsilon_N < 1/(M \vee e)$ can be written as

$$N > (M \vee e)^{2+5\nu_0}\log N.$$

Then applying Lemma 6.1 with $\alpha = \beta = 1$ and $C = (M \vee e)^{2+5\nu_0}$ gives the result. ■

Similarly, we need to understand the conditions needed on N to get equation (6.2) when $\varepsilon = \varepsilon_N$. This is done in the following Lemma.

Lemma 6.3. *Equation (6.2) is verified for $N \geq 3$ and*

$$N \geq \left(\frac{8+16\nu_0}{5\nu_0}M^2\left(\log\left(\frac{8+16\nu_0}{5\nu_0}M^2\right) - 1\right)\right)^{\frac{2+5\nu_0}{5\nu_0}} =: N_2.$$

Proof of Lemma 6.3. For $\varepsilon = \varepsilon_N$, equation (6.2) rewrites as

$$N^{\frac{5\nu_0}{4+10\nu_0}} > 2M\sqrt{\frac{1+2\nu_0}{2+5\nu_0}}\sqrt{\log\left(\frac{N}{\log N}\right)}.$$

For $N \geq 3$, $\log N \geq 1$ hence it is enough to find N_2 such that for $N \geq N_2$

$$N^{\frac{5\nu_0}{4+10\nu_0}} > 2M\sqrt{\frac{1+2\nu_0}{2+5\nu_0}}\sqrt{\log(N)}.$$

Then, N_2 is given by applying Lemma 6.1 with

$$\alpha = \frac{5\nu_0}{4+10\nu_0}\ , \quad \beta = \frac{1}{2} \text{ and } \quad C = 2M\sqrt{\frac{1+2\nu_0}{2+5\nu_0}}.$$

■

At this stage we can rewrite inequality (6.3) with $\varepsilon = \varepsilon_N$, and from the two previous lemmas we get that for $N > 3 \vee N_1 \vee N_2$,

$$\begin{aligned}
&\mathbb{P}\left(\left\|G_N f - \mathbb{G}^{(N)}\right\|_\infty > \frac{(D+\gamma_1+(1+A)\gamma_2)}{\sqrt{2+5\nu_0}} N^{-(2+5\nu_0)} (\log N)^{\frac{4+5\nu_0}{4+10\nu_0}}\right) \\
\leq & C_1 c_1^2 \left(\frac{N}{\log N}\right)^{\frac{2\nu_0}{2+5\nu_0}} \exp\left(-\frac{\gamma_1 C_2}{c_1^{5/2} M\sqrt{2+5\nu_0}} \sqrt{\log N \log\left(\frac{N}{\log N}\right)}\right) \\
&+ 2\exp\left(-\frac{A_1\gamma_2}{M\sqrt{2+5\nu_0}} N^{\frac{5\nu_0}{4+10\nu_0}} (\log N)^{\frac{4+5\nu_0}{4+10\nu_0}}\right) + 4\exp\left((-\frac{A_5\gamma_2^2}{2+5\nu_0} \log\left(\frac{N}{\log N}\right)\right) \\
&=: S_1 + S_2 + S_3.
\end{aligned}$$

Fix $\lambda > 1$ and take

$$\gamma_1 = \frac{c_1^{5/2} M\sqrt{2+5\nu_0}}{C_2}\left(\lambda + 1 + \frac{2\nu_0}{2+5\nu_0}\right) \quad \text{and} \quad \gamma_2 = \sqrt{\frac{(\lambda+1)(2+5\nu_0)}{A_5}}.$$

We now need conditions on N to ensure upper-bound or order $N^{-\lambda}$ for each S_1, S_2 and S_3.

Study of S_1. From the choice of γ_1 we have that

$$\begin{aligned}
S_1 &= C_1 c_1^2 \left(\frac{N}{\log N}\right)^{\frac{2\nu_0}{2+5\nu_0}} \exp\left(-\left(\lambda + 1 + \frac{2\nu_0}{2+5\nu_0}\right)\sqrt{\log(N)\log\left(\frac{N}{\log N}\right)}\right) \\
&\leq C_1 c_1^2 \left(\frac{N}{\log N}\right)^{-(\lambda+1)} \\
&= N^{-\lambda}\ C_1 c_1^2 N^{-1} (\log N)^{\lambda+1}.
\end{aligned}$$

Hence, we need to know how big N needs to be, so that $N \geq C_1 c_1^2 (\log N)^{\lambda+1}$. Once again, applying Lemma 6.1 with $\alpha = 1$, $\beta = \lambda + 1$ and $C = C_1 c_1^2$ gives that $S_1 \leq N^{-\lambda}$, for

$$\begin{aligned}
N &\geq C_1 c_1^2 \left(2(\lambda+1)\left(\log\left(2(\lambda+1)(C_1 c_1^2)^{1/(\lambda+1)}\right) - 1\right)\right)^{\lambda+1} \\
&=: N_3.
\end{aligned}$$

Study of S_2. Recall the expression of S_2 :

$$S_2 = 2\exp\left(-\frac{A_1\sqrt{\lambda+1}}{\sqrt{A_5}M} N^{\frac{5\nu_0}{4+10\nu_0}} (\log N)^{\frac{4+5\nu_0}{4+10\nu_0}}\right).$$

We want to have

$$\frac{A_1\sqrt{\lambda+1}}{\sqrt{A_5}M}N^{\frac{5\nu_0}{4+10\nu_0}}(\log N)^{\frac{4+5\nu_0}{4+10\nu_0}} \geq \lambda \log N,$$

meaning that we need

$$N^{\frac{5\nu_0}{4+10\nu_0}} \geq M\frac{\sqrt{A_5}}{A_1}\frac{\lambda}{\sqrt{\lambda+1}}(\log N)^{\frac{5\nu_0}{4+10\nu_0}}.$$

Thus, using Lemma 6.1 with $\alpha = \beta = 1$ and

$$C = \left(M\frac{\sqrt{A_5}}{A_1}\frac{\lambda}{\sqrt{\lambda+1}}\right)^{\frac{4+10\nu_0}{5\nu_0}},$$

we have $S_2 \leq 2N^{-\lambda}$ for

$$\begin{aligned} N &\geq 2\left(M\frac{\sqrt{A_5}}{A_1}\frac{\lambda}{\sqrt{\lambda+1}}\right)^{\frac{4+10\nu_0}{5\nu_0}}\left(\log\left(2\left(M\frac{\sqrt{A_5}}{A_1}\frac{\lambda}{\sqrt{\lambda+1}}\right)^{\frac{4+10\nu_0}{5\nu_0}}\right)-1\right) \\ &=: N_4. \end{aligned}$$

Study of S_3. From the choice of γ_2 we have

$$\begin{aligned} S_3 &= 4\exp\left(-(\lambda+1)\log\left(\frac{N}{\log N}\right)\right) \\ &= 4N^{-\lambda}\frac{(\log N)^{\lambda+1}}{N} \end{aligned}$$

So we want $N \geq (\log N)^{\lambda+1}$. Applying once again Lemma 6.1, we have $S_3 \leq 4N^{-\lambda}$, for

$$N \geq (2(\lambda+1)(\log(2(\lambda+1))-1))^{\lambda+1} =: N_5.$$

If we now combine all the previous conditions on N we get the following result.

Lemma 6.4. *For all $\lambda > 1$, and for all $N > 3 \vee N_1 \vee N_2 \vee N_3 \vee N_4 \vee N_5$, we have*

$$\mathbb{P}\left(\sup_{\tilde{f}\in\tilde{\mathcal{F}}}\left|G_N\tilde{f} - \tilde{\mathbb{G}}^{(N)}(\tilde{f})\right| > \rho N^{-\frac{1}{2+5\nu_0}}\log N^{\frac{4+5\nu_0}{4+10\nu_0}}\right) \leq 7N^{-\lambda},$$

where the constant ρ is given by

$$\rho = \frac{(A_2+2A)\sqrt{2\nu_0}}{\sqrt{2+5\nu_0}} + \frac{c_0^{5/2}M^{\frac{2+5\nu_0}{2}}}{C_2}(\lambda+1+\frac{2\nu_0}{2+5\nu_0}) + (1+A)\sqrt{\frac{\lambda+1}{A_5}}$$

and the ranks N_1, N_2, N_3, N_4 and N_5 are given by

$$
\begin{aligned}
N_1 &= 2(M \vee e)^{2+5\nu_0} \left(\log\left(2(M \vee e)^{2+5\nu_0}\right) - 1\right) \\
N_2 &= \left(\frac{8+16\nu_0}{5\nu_0} M^2 \left(\log\left(\frac{8+16\nu_0}{5\nu_0} M^2\right) - 1\right)\right)^{\frac{2+5\nu_0}{5\nu_0}} \\
N_3 &= C_1 c_1^2 \left(2(\lambda+1)\left(\log\left(2(\lambda+1)(C_1 c_1^2)^{1/(\lambda+1)}\right) - 1\right)\right)^{\lambda+1} \\
N_4 &= 2\left(M\frac{\sqrt{A_5}}{A_1}\frac{\lambda}{\sqrt{\lambda+1}}\right)^{\frac{4+10\nu_0}{5\nu_0}} \left(\log\left(2\left(M\frac{\sqrt{A_5}}{A_1}\frac{\lambda}{\sqrt{\lambda+1}}\right)^{\frac{4+10\nu_0}{5\nu_0}}\right) - 1\right) \\
N_5 &= \left(2(\lambda+1)\left(\log(2(\lambda+1)) - 1\right)\right)^{\lambda+1}.
\end{aligned}
$$

The last step to obtain Theorem 4.2 is to derive the dependency of c_0 and ν_0 with respect to k and M, the parameters from assumptions **(L)** and **(B)** . We showed at the end of Section 6.1 that

$$N(\mathcal{F}, d_Q, kT/q) \leq q + 1.$$

Hence for all $0 < \delta < 1$, we have the following control on the covering number

$$N(\mathcal{F}, \delta) := \sup_Q N(\mathcal{F}, d_Q, \delta M) \leq \frac{kT}{M}\delta^{-1} + 1.$$

Choosing $c_0 = kT/M + 1$ and $\nu_0 = 1$ yields to a bound on the covering number by $c_0\delta^{-\nu_0}$, as desired. So rewriting Lemma 6.4 with these values for c_0 and ν_0 finishes the proof of Theorem 4.2.

6.3 Application to the Heat distance processes.

6.3.1 Bounds for the laplacian eigenvalues.

Let us start by proving a lemma on the extremal laplacian eigenvalues of graphs in $\mathcal{G}_n$. We define $\Lambda_{\min}$ and $\Lambda_{\max}$ as, respectively, the minimal and maximal positive laplacian eigenvalues of graphs in $\mathcal{G}_n$:

$$
\begin{aligned}
\Lambda_{\min} &:= \inf\{\lambda > 0, \text{ s.t. } \lambda \text{ is an eigenvalue of } L(G), G \in \mathcal{G}_n\} \\
\Lambda_{\max} &:= \sup\{\lambda > 0, \text{ s.t. } \lambda \text{ is an eigenvalue of } L(G), G \in \mathcal{G}_n\}.
\end{aligned}
$$

Lemma 6.5. *$\Lambda_{\min}$ and $\Lambda_{\max}$ satisfy the following bounds :*

$$\Lambda_{\min} \geq \frac{8w_{\min}}{n^2}, \tag{6.4}$$

$$\Lambda_{\max} \leq n w_{\max}. \tag{6.5}$$

Note that (6.4) will not be used in the rest of the paper. Still, we choose to present it as we believe it could be of future use.

Proof. Take $G \in \mathcal{G}_n$. We want to prove that λ_{min}, the smallest positive eigenvalue of $L(G)$, verifies $\lambda_{min} \geq 8w_{\min}n^{-2}$. To do so, we apply a Cheeger-type inequality. First assume that G is connected, hence $\lambda_{min} = \lambda_2(G)$. Let V be the set of vertices of G. Following [Fie95, Section 3] we define the *average minimal cut* of G by

$$\gamma(G) = \min_{\emptyset \neq U \subsetneq V} \sum_{i \in U,\ j \in V \setminus U} \frac{w_{i,j}}{|U|\,(n - |U|)}.$$

As G is connected, there exists at least one edge with a weight greater than $w_{\min}$ joining U and $V \setminus U$. Hence,

$$\sum_{i \in U,\ j \in V \setminus U} w_{i,j} \geq w_{\min}.$$

Moreover, for all U, $|U|\,(n - |U|) \leq n^2/4$. This yields to $\gamma(G) \geq 4w_{\min}/n^2$. From [Fie95, Theorem 2], we have that $\lambda_2(G) \geq 2\gamma(G)$, hence $\lambda_2(G) \geq 8w_{\min}n^{-2}$. If now G is not connected, one can check that there exists G_{sub} a connected subgraph of G of size n_{sub}, such that $\lambda_{min} = \lambda_2(G_{sub})$. This gives

$$\lambda_{min} = \lambda_2(G_{sub}) \geq \frac{8w_{\min}}{n_{sub}^2} \geq \frac{8w_{\min}}{n^2}.$$

This finishes the proof of the first bound.

We now prove the second bound. The largest eigenvalue of L(G) is denoted by λ_n. Letting x be an eigenvector associated to λ_n verifying $\|x\|_2 = 1$, we have

$$\lambda_n = x^T L(G) x = \sum_{i<j} w_{i,j}(x_i - x_j)^2 \leq w_{\max} \sum_{i<j} (x_i - x_j)^2 = w_{\max}\ x^T L(G_{comp}) x$$

where G_{comp} is the complete undirected graph, hence

$$L(G_{comp}) = \begin{pmatrix} n-1 & -1 & \cdots & -1 \\ -1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & -1 \\ -1 & \cdots & -1 & n-1 \end{pmatrix}.$$

One can check that $x^T L(G_{comp}) x \leq \lambda_n(L(G_{comp})) = n$, so $\lambda_n \leq n w_{\max}$. ■

6.3.2 Result on HKD processes.

To apply Theorem 4.1 to the HKD processes, we first prove that they are uniformly bounded and Lipschitz-continuous.

Proposition 6.2. *For all G, G' in $\mathcal{G}_n$, denote by $(\lambda_k)_{1\leq k\leq n}$ and $(\phi_k)_{1\leq k\leq n}$ (resp. $(\lambda'_l)_{1\leq l\leq n}$ and $(\phi'_l)_{1\leq l\leq n}$) the eigenvalues and orthonormal eigenvectors of $L(G)$ (resp. $L(G')$). Remember that we always choose ϕ_1 and ϕ'_1 equal to the vector with all entries equal to $1/\sqrt{n}$. We denote by $\langle\cdot,\cdot\rangle$ the standard scalar product in $\mathbb{R}^n$. Then, the application $t \to D_t((G, G'))$ verifies :*

1. *For all $t \in [0, T]$, $D_t((G, G'))$ can be written in terms of the eigen-elements of $L(G)$ and $L(G')$:*

$$D_t((G,G')) = \left(\sum_{k,l=2}^{n} (e^{-t\lambda_k} - e^{-t\lambda'_l})^2 \langle\phi_k, \phi'_l\rangle^2\right)^{1/2}. \tag{6.6}$$

2. *For all $t \in [0, T]$, $D_t((G, G')) \leq \sqrt{n}$.*

3. *$t \mapsto D_t((G, G'))$ is $(n^{3/2} w_{\max})$-Lipschitz continuous on $[0, T]$.*

Proof. Let G, G' be in $\mathcal{G}_n$. We start by proving (6.6). This is done through the following computation :

$$\begin{aligned}
&\|e^{-tL} - e^{-tL'}\|_F^2 \\
=&\|e^{-tL}\|_F^2 + \|e^{-tL'}\|_F^2 - 2Tr\left(e^{-tL}e^{-tL'}\right) \\
=&\sum_{k=1}^{n} e^{-2t\lambda_k}\|\phi_k\phi_k^T\|_F^2 + \sum_{l=1}^{n} e^{-2t\lambda'_l}\|\phi'_l\phi'^T_l\|_F^2 - 2\sum_{k,l=1}^{n} e^{-t\lambda_k}e^{-t\lambda'_l} Tr\left(\phi_k\phi_k^T\phi'_l\phi'^T_l\right)
\end{aligned}$$

One can prove that $Tr\left(\phi_k\phi_k^T\phi'_l\phi'^T_l\right) = \langle\phi_k, \phi'_l\rangle^2$.

Similarly $\|\phi_k\phi_k^T\|_F^2 = \|\phi_k\|_2^2 = \sum_{l=1}^{n}\langle\phi_k, \phi'_l\rangle^2$ and $\|\phi'_l\phi'^T_l\|_F^2 = \|\phi'_l\|_2^2 = \sum_{k=1}^{n}\langle\phi_k, \phi'_l\rangle^2$.

So

$$\|e^{-tL} - e^{-tL'}\|_F^2 = \sum_{k,l=1}^{n} (e^{-t\lambda_k} - e^{-t\lambda'_l})^2\langle\phi_k, \phi'_l\rangle^2.$$

The sums can start at $k = 2$ and $l = 2$ thanks to the facts that $\lambda_1 = \lambda'_1 = 0$ and $\phi_1 = \phi'_1$ combined with the orthogonality of the eigenvectors families. This finishes the proof of (6.6).

Bounding all terms $(e^{-t\lambda_k} - e^{-t\lambda'_l})^2$ by 1 in (6.6), and using the orthonormality of the eigenvectors families yields to $D_t((G, G')) \leq \sqrt{n}$.

We now prove the Lipschitz result. One can check that $t \to D_t((G, G'))$ is $\mathcal{C}^1$ on $[0, T]$. The case $G = G'$ is easily dealt with. Assume now that $G \neq G'$.

For all $t \in (0, T]$,

$$\left|\frac{d}{dt} D_t((G, G'))\right| = \left| \frac{-\sum_{k,l=2}^{n} (e^{-t\lambda_k} - e^{-t\lambda'_l})(\lambda_k e^{-t\lambda_k} - \lambda'_l e^{-t\lambda'_l}) \langle \phi_k, \phi'_l \rangle^2}{\left(\sum_{k,l=2}^{n} (e^{-t\lambda_k} - e^{-t\lambda'_l})^2 \langle \phi_k, \phi'_l \rangle^2 \right)^{1/2}} \right|$$

$$\leq \sum_{k,l=2}^{n} \frac{\left|e^{-t\lambda_k} - e^{-t\lambda'_l}\right| \left|\lambda_k e^{-t\lambda_k} - \lambda'_l e^{-t\lambda'_l}\right| \left|\langle \phi_k, \phi'_l \rangle\right|^2}{\left(\sum_{i,j=2}^{n} (e^{-t\lambda_i} - e^{-t\lambda'_j})^2 \langle \phi_i, \phi'_j \rangle^2 \right)^{1/2}}$$

$$\leq \sqrt{\sum_{k,l=2}^{n} \left|\lambda_k e^{-t\lambda_k} - \lambda'_l e^{-t\lambda'_l}\right|^2 \langle \phi_k, \phi'_l \rangle^2} \sqrt{\frac{\sum_{k,l=2}^{n} (e^{-t\lambda_k} - e^{-t\lambda'_l})^2 \langle \phi_k, \phi'_l \rangle^2}{\sum_{i,j=2}^{n} (e^{-t\lambda_i} - e^{-t\lambda'_j})^2 \langle \phi_i, \phi'_j \rangle^2}}$$

$$= \sqrt{\sum_{k,l=2}^{n} \left|\lambda_k e^{-t\lambda_k} - \lambda'_l e^{-t\lambda'_l}\right|^2 \langle \phi_k, \phi'_l \rangle^2},$$

where the last inequality comes from the Cauchy-Schwarz inequality.

According to the mean value theorem, for all k and l

$$\left|\lambda_k e^{-t\lambda_k} - \lambda'_l e^{-t\lambda'_l}\right| \leq |\lambda_k - \lambda'_l| \leq \Lambda_{\max}.$$

Hence, for all $t \in (0, T]$,

$$\left|\frac{d}{dt} D_t((G, G'))\right| \leq \Lambda_{\max} \sqrt{\sum_{k,l=2}^{n} \langle \phi_k, \phi'_l \rangle^2} \leq n w_{\max} \sqrt{n} = n^{3/2} w_{\max}.$$

This finishes the proof. ■

The proof of Theorem 5.1 is a direct application of Theorem 4.1, as Proposition 6.2 ensures that the hypotheses of Theorem 4.1 are verified by the HKD process.

6.3.3 Result on HPD processes.

To prove Theorem 5.2, we start by showing that the HPD processes are uniformly bounded and Lipschitz-continuous under the theorem's hypotheses.

Proposition 6.3. *For all graphs G, G' in $\mathcal{G}^n$, the application $t \to H_t((G, G'))$ verifies :*

1. *For all $t \in [0, T]$, $0 \leq H_t((G, G')) \leq 1$.*

2. $t \mapsto H_t((G, G'))$ *is* $(2nw_{\max})$*-Lipschitz-continuous on* $[0, T]$.

Proof. Recall that for all $G \in \mathcal{G}^n$, and for a vertex i,

$$h_t(G)(i) = \sum_{k=1}^{n(G)} e^{-t\lambda_k} \phi_k(i)^2,$$

where $(\lambda_k)_{1 \leq k \leq n(G)}$ and $(\phi_k)_{1 \leq k \leq n(G)}$ are the eigenvalues and orthonormal eigenvectors of $L(G)$ and $n(G)$ is the size of G. Hence $0 \leq h_t(G)(i) \leq 1$ for all i, meaning that all points in the diagram $Dg(G, h_t(G))$ are contained in $[0, 1]^2$. So from the definition of the Bottleneck distance, $0 \leq H_t((G, G')) \leq 1$, for all G, G' in $\mathcal{G}^n$ and for all $t \in [0, T]$.

Let us now compute the first derivative of $h_t(G)(i)$:

$$\frac{d}{dt} h_t(G)(i) = -\sum_{k=1}^{n(G)} \lambda_k e^{-t\lambda_k} \phi_k(i)^2.$$

Its absolute value is upper-bounded by $\lambda_{n(G)}$, the largest eigenvalue of $L(G)$. From Lemma 6.5, we have $\lambda_{n(G)} \leq n(G) w_{\max}$. Hence, $t \to h_t(G)$ is $(nw_{\max})$-Lipschitz continuous on $[0, T]$. To conclude, we come back to the definition of the HPD in terms of distance between persistence diagrams. Applying the triangular inequality to the Bottleneck distance gives for all $G, G' \in \mathcal{G}^n$, for all $t, t' \in [0, T]$, and for all diagram construction Dg,

$$\begin{aligned} &|d_B(Dg(G, h_t(G)), Dg(G', h_t(G'))) - d_B(Dg(G, h_{t'}(G)), Dg(G', h_{t'}(G')))| \\ \leq &d_B(Dg(G, h_t(G)), Dg(G, h_{t'}(G))) + d_B(Dg(G', h_t(G')), Dg(G', h_{t'}(G'))). \end{aligned}$$

Similarly, the same inequality but with maxima over the diagram constructions holds :

$$\begin{aligned} &|H_t((G, G')) - H_{t'}((G, G'))| \\ \leq &\max d_B(Dg(G, h_t(G)), Dg(G, h_{t'}(G))) + \max d_B(Dg(G', h_t(G')), Dg(G', h_{t'}(G'))). \end{aligned}$$

Applying Theorem 2.3 and using the Lispchitz continuity of the HKS yields

$$\begin{aligned} &|H_t((G, G')) - H_{t'}((G, G'))| \\ \leq &\|h_t(G) - h_{t'}(G)\|_\infty + \|h_t(G') - h_{t'}(G')\|_\infty \\ \leq &2nw_{\max} |t - t'| . \end{aligned}$$

■

Theorem 5.2 follows from Proposition 6.3 and Theorem 4.1.

Part II

Numerical Illustrations

Chapter 7

From generative random graph models

We illustrate the construction of confidence bands and two-sample tests on synthetic data sets of pairs of graphs based on the HKD and HPD processes. We consider different random graph models and combine them to create pairs of independent graphs.

7.1 Methods and graph models

7.1.1 Methods

Recall that for a pair of aligned (weighted) graphs G and G', we can compute the heat kernel distance (HKD) at time t : $D_t((G,G')) = \|e^{-tL} - e^{-tL'}\|_F$, where L and L' are the Laplacian matrices of G and G' respectively. If G and G' are not aligned or of different size, recall that we can compute the heat persistent distance (HPD) at time t: $H_t((G,G')) = \max_{Dg} d_B(Dg(G,h_t(G)), Dg(G',h_t(G')))$. It compares persistence diagrams $Dg(G,h_t(G))$ and $Dg(G',h_t(G'))$ with the bottleneck distance. For more details on these distances, see Chapter 3.

We are interested in using distance processes. Hence, we consider HKD processes $\{D_t((G,G')),\ t \in [0,T]\}$ and HPD processes $\{H_t((G,G')),\ t \in [0,T]\}$. As a first step, we illustrate the construction of confidence bands around the empirical mean distance process, using the bootstrap method detailed in Algorithm 1. Then, we evaluate the performance of the two-sample test presented in Algorithm 2. Recall that for two samples of pairs of graphs $((G_{1,1},G'_{1,1}),\dots,(G_{1,N},G'_{1,N}))$ and $((G_{2,1},G'_{2,1}),\dots,(G_{2,M},G'_{2,M}))$, if we denote by P_N^1 and P_M^2 their empirical probability measures, the HKD-based test statistic[1] is defined by

$$T = \sqrt{NM(N+M)^{-1}} \sup_{t\in[0,T]} \left|P_N^1 D_t - P_M^2 D_t\right|.$$

[1]The formula for the HPD-based test statistic is similar, only D_t is replaced by H_t

7.1.2 Random graph models

We present the models generating the random graphs, namely the Erdős-Rényi model [ER+60], the stochastic block model [HLL83] and the geometric model [P+03]. For each, we specify the parameters used for the simulations.

Erdős-Rényi model (ER) This model generates random graphs where each edge appears with probability p, independently from all the others. It requires two parameters: the graph size n and the edge probability p. Because of the edges' independence and their homogeneity, ER graphs are considered to have no specific structure.

In our simulations, we take $n = 50$ and $p = 0.5$. Weights may be added by assigning a uniform weight between 0 and 2 to each existing edge, independently from all the others.

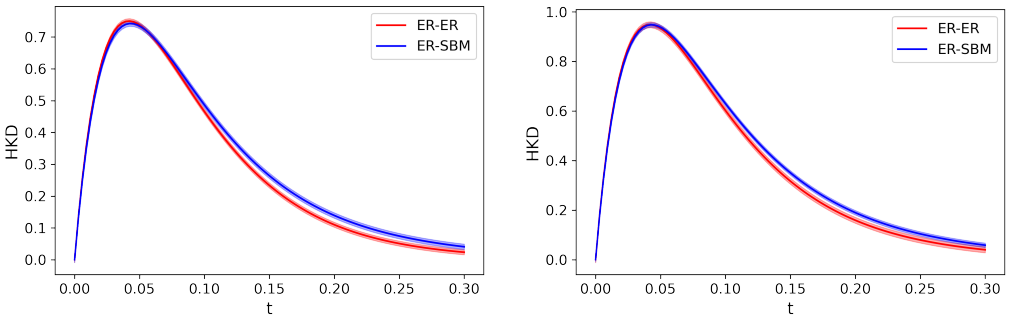
Stochastic block model (SBM) This model is a generalization of the ER model that introduces a block structure. The n nodes are clustered in K groups $C_1, \dots, C_K$, of respective sizes $n_1, \dots, n_K$. Edges appear independently from the others, but with a probability depending on the groups : edge $\{i, j\}$ appears with probability $p_{k,l}$ when $i \in C_k$ and $j \in C_l$.

In our simulations, we take $K = 2$, $n_1 = n_2 = 25$, and $p_{1,1} = p_{2,2} = 0.75$, $p_{1,2} = p_{2,1} = 0.25$. So graphs are composed of two dense clusters, with few edges between them. Similarly to the ER model, we may add random weights following the uniform distribution between 0 and 2.

Geometric model (GM) Given a compact domain U of $\mathbb{R}^d$ for some d, a graph is generated from the GM by drawing n points uniformly on U and creating an edge between two points if their Euclidean distance is smaller than a given threshold. Here we choose a slight variation of this model by considering a number $p \in [0, 1]$ and creating the edges corresponding to the $\lfloor p\binom{n}{2} \rfloor$ pairs of points with the smallest euclidean distances.

In our simulations, the compact domain U is either A_ε the annulus in $\mathbb{R}^2$ with outer radius 1 and inner radius $\varepsilon > 0$ or A_0 the unit disk. We either fix $n = 50$ or for each graph, n is drawn from a Poisson distribution of parameter 50. We take $p = 0.5$. To obtain weighted graphs, we may assign the weight e^{-2d} to an edge, where d is the distance between the two points forming the edge.

We combine these models to generate pairs of independent graphs on which we can compute HKD and HPD processes. We consider the pairs of independent ER graphs (ER-ER) and the pairs containing one ER graph and one SBM graph (ER-SBM). For these distributions, the groups' composition is known and nodes are treated independently among groups. Thus, we can consider that we know an NC between graphs. As a result, we can compute both HKD and HPD processes. Similarly, we consider pairs of independent geometric random graphs: Disk-Disk and Disk-Annulus. However, as nodes in these models correspond to random points, there is no default NC. Hence, only HPD processes are computed.



(a) Unweighted.

(b) Weighted.

Figure 7.1: Confidence band around the mean HKD processes with ER-ER (red) and ER-SBM (blue) distributions.

7.2 Simulation results

In this section, we present the simulation results concerning the construction of confidence bands and the performances of the two-sample tests.

7.2.1 Confidence bands

In this section, we compute confidence bands under the different models of pairs of graphs defined above. For each model, we draw a sample $(G_1, G_1'), \dots, (G_N, G_N')$ with $N = 100$. We compute the mean process, that is $t \to N^{-1} \sum_i D_t((G_i, G_i'))$ or $t \to N^{-1} \sum_i H_t((G_i, G_i'))$ and compute a confidence band of level 99% around this empirical mean using the bootstrap method detailed in Algorithm 1. Computations are done with 1000 bootstrap samples. Results are shown in Figure 7.1, 7.2 and 7.3, where solid lines represent empirical means and transparent areas represent confidence bands.

Remark that confidence bands around HKD processes (Figure 7.1) seem to be narrower than those around HPD processes (Figure 7.2). Therefore, users should rather use HKD processes whenever NCs are available. Nonetheless, the versatility of HPD processes does not totally reduce their efficiency. As Figure 7.3 indicates, HPD empirical means seem to be able to discriminate between the different distributions. These observations will be confirmed in the next section, where the performances of the two-sample tests are investigated.

7.2.2 Two-sample tests

Levels and Powers. Simulations are run to evaluate the performances of the two-sample tests using HKD and HPD processes as detailed in Algorithm 2, see Figure 7.4 and Figure 7.5. In all tests, the desired level is set to 0.05, and computations are done with 1000 bootstrap samples. Figures 7.4a and 7.5a illustrate that the asymptotic levels of the tests correspond to the set level. On the other hand, Figures 7.4b and

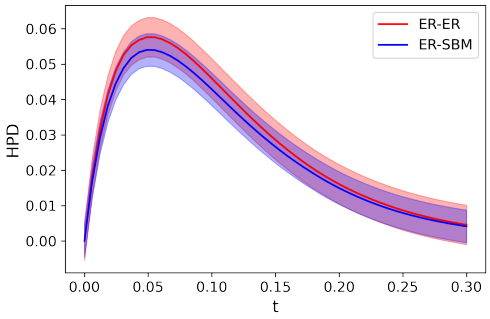
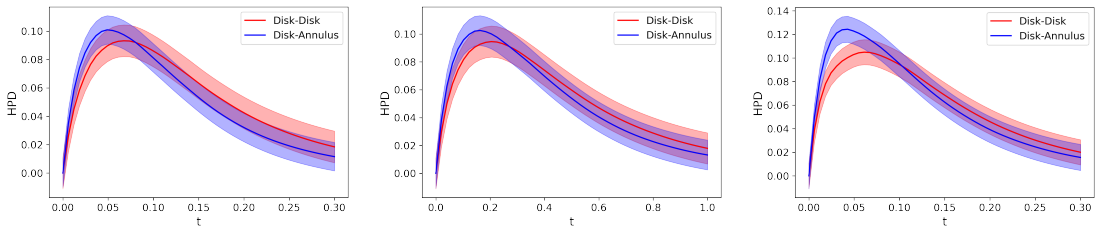


Figure 7.2: Confidence band around the mean HPD processes with unweighted ER-ER (red) and ER-SBM (blue) distributions.

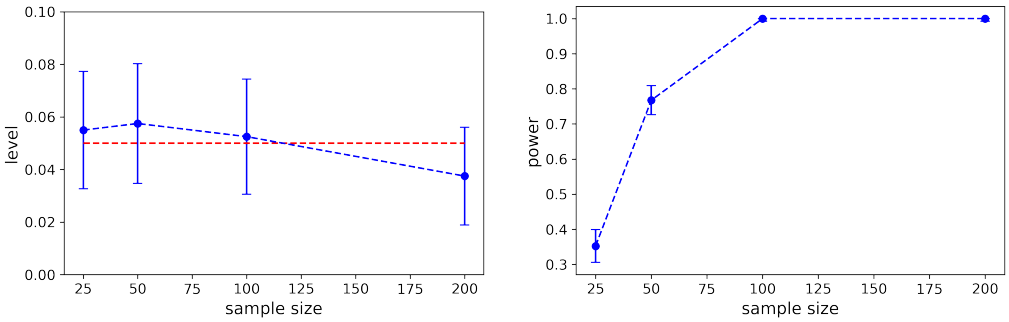


(a) Unweighted, fixed size.

(b) Weighted, fixed size.

(c) Unweighted, random size.

Figure 7.3: Confidence band around the mean HPD processes with Disk-Disk (red) and Disk-Annulus (blue) distributions.



(a) Levels estimations.

(b) Powers estimations.

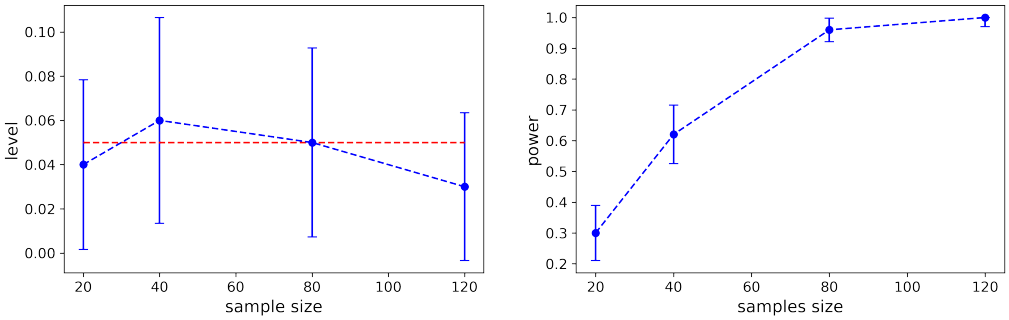
Figure 7.4: Performances of the two-sample test using HKD processes. The level estimations are made with weighted ER-ER distributions, while the power estimations are made with weighted ER-ER and ER-SBM distributions. The samples size varies in [25,50,100,200]. Estimations are done by repeating 400 independent tests. Vertical lines represent 95%-confidence intervals of the estimations.

7.5b illustrate that the powers tend to 1 when sample sizes increase, indicating that the tests manage to distinguish between the different distributions.

Comparison with the Neyman-Pearson Test. The Neyman-Pearson test [NP33] is an optimal testing procedure in the sense that it is the test with the highest power for a given level. But being based on likelihood ratios, it rarely is computable. Its performances are determined by the total variation distance (TV) between the two distributions. If we consider the case of distributions that depend on the sample size, the speed at which their TV tends to 0 determines if the Neyman-Pearson test will asymptotically be able to distinguish between the two distributions. For ER models, the independence of the edges allows us to theoretically compute bounds of the TV and determine the phase transition. Let us take a real number p such that $0 < p < 1$. And for each sample size $N \geq 1$ consider the parameters $p_0(N)$ and $p_1(N)$, such that both converge to p. Following [LR06, Section 13.1.], we can show that as long as $|p_0(N) - p_1(N)| \gg N^{-1/2}$, the Neyman-Pearson test asymptotically distinguishes between the distributions of independent pairs of $\mathcal{G}(n, p_0(N))$ and independent pairs of $\mathcal{G}(n, p_1(N))$ when using N-samples. Figure 7.6 shows that for large enough sample sizes, our two-sample test based on HKD processes distinguishes between ER models up to $|p_0(N) - p_1(N)| = C \log N/\sqrt{N}$, with $C = 0.01$. The tests are computed with a set level of 0.05 and with 1000 bootstrap samples.

7.3 Discussion

We illustrated the performances of our methods by simulations on synthetic data sets of pairs of graphs. We showed that the two-sample tests were able to distinguish between Erdős-Rényi and SBM graphs, as well as between geometric graphs sampled on different domains. On Erdős-Rényi models with parameters



(a) Levels estimations. (b) Powers estimations.

Figure 7.5: Performances of the two-sample test using HPD processes. The level estimations are made with Disk-Disk distributions, while the power estimations are made with Disk-Disk and Disk-Annulus distributions. Graphs are unweighted and of fixed size. The samples size varies in [20,40,80,120]. Estimations are done by repeating 100 independent tests. Vertical lines represent 95%-confidence intervals of the estimations.

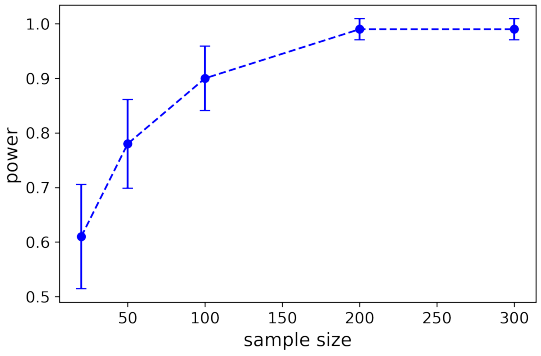


Figure 7.6: Evolution of the power of the HKD two-sample test when $|p_0(N) - p_1(N)| = C \log N/\sqrt{N}$. The samples size N varies in [20,50,100,200,300]. Estimations are done by repeating 100 independent tests. Vertical lines represent 95%-confidence power intervals of the estimations.

depending on the sample size, the tests seemed to be asymptotically distinguishing between the different distributions, even when working close to the phase transition of the Neyman-Pearson test.

However, this empirical study was performed on elementary synthetic data. While it corroborates with the theoretical asymptotic results, it acts essentially as a sanity check. Further investigations are needed to understand more deeply how these new tools behave.

Chapter 8

Distribution shift detection from activation graphs in neural network learning

We tackle the problem of distribution shift detection (DSD) in the context of neural network learning. That is, we want to detect if a given neural network, presented with a new data set, is performing poorly due to a change in distribution. For that, we propose to use activation graphs as inputs to the two-sample tests we developed for graphs.

8.1 Introduction to distribution shift detection

To motivate the DSD problem, assume that we train a neural network (NN) for a classification task and that we evaluate the performance of the neural network classifier on some test data. If performances are good[1], we can propose this classifier as a solution to the classification task. Now, one can wonder how this classifier would perform on some new data set. Of course, if the data distribution stays unchanged, it should still perform well. However, what can we do in the case where the distribution changes? There are two approaches to this problem. The first one is to build a better and more robust classifier. This is usually done by taking action before or during the training phase. We can mention several techniques: *data augmentation* [SK19], *pre-training* on large auxiliary data sets [HLM19], modifying the network architecture or changing the loss function [AHY+19]. The other approach is the DSD, where we try to detect the degradation of the neural network performances in a post-training phase. This is the subject of this chapter.

DSD is a challenging problem. In essence, it requires to compute the neural network performances. However, the information needed for the computations is not available as the true outputs of the new data set, *e.g.,* the true labels, are unknown. To circumvent this issue, we can try to capture hints that indicate whether

[1]Performances are measured in terms of given metrics, *e.g.,* the classifier's accuracy.

the neural network is performing poorly or not. To do so, people often look at confidence values [HG17]. They are defined as the maximal entry of the probability vector returned by the neural network. The general idea is that a small mean confidence value potentially indicates poor performances of the network on the data set. While this approach has the advantage of taking into account the behavior of the classifier, it might not be able to capture distribution changes. Indeed, neural networks have been shown to be over-confident in some situations where they have very poor performances, especially in the context of *adversarial attacks* [NYC15, AM18, BR18].

Another approach to the DSD problem is to apply a two-sample test directly between the new input data and the test data. It has several drawbacks. First, two-sample test algorithms depend on the nature of the data. While for some classical input spaces, such tests might be available, when working with more infrequent and complex data types, finding relevant and trustworthy tests might be more challenging. Secondly, what we are actually interested in is detecting whether the neural network predictions on a given data set can be trusted or not. This can be different from detecting if the data set has the same distribution as the training and test sets. In fact, if we expect the classifier to have some robustness, it exactly means that it would perform well on data sets coming from slightly different distributions.

In order to take into account how the network works on the data, we propose to perform two-sample tests on activation graphs instead of input data. Activation graphs are weighted graphs that encode how the neural network processes each input data. Introduced by [GS17] and [GSH19], they have been used to monitor training phases [RTB+18] or to measure *topological uncertainty* in the NN predictions [LIC+21]. By comparing activation graph distributions, instead of testing whether the data distribution has changed, we test whether the way the neural network processes the data has changed. To perform these tests, we apply the methods developed in Chapter 5 by using *heat kernel distance* (HKD) *processes* to compare activation graphs data.

8.2 Tools and methods

Activation graphs. We define activation graphs on a sequence of consecutive dense layers. Each input data yields to an activation graph. It is a weighted undirected graph whose vertex set corresponds to the neurons of the considered layers. Hence, activation graphs have the same vertex set; only the edge weights vary.

For two neurons u and$v$ in layers l and $l+1$ respectively, the weight between u and v in the activation graph is given by $\overline{W}_x(u,v) = W_l(u,v)\, x_l(u)$, where $W_l(u,v)$ is the weight of the NN between u and v, and $x_l(u)$ is the activation level of u. For more details on activation graphs see Section 2.4.2. In the following, we consider graphs that are defined as activation graphs except that all weights are composed with the absolute value, ensuring non-negative weights. We call them *absolute activation graphs*. As we only use absolute activation graphs in the rest of this chapter, we will sometimes drop the adjective "*absolute*".

Heat kernel distance. Our two-sample tests are based on distance processes. In this chapter, we can use the heat kernel distance (HKD) process to compare graphs, as absolute activation graphs come with a natural node correspondence given by the neurons. Assume that the neurons considered are numbered from 1 to n. We compare graphs through their heat kernels. Consider two weighted graphs G and G' and their respective Laplacian matrices L and L'. Recall that their heat kernels at time t are the matrices e^{-tL} and $e^{-tL'}$. Then the HKD at time t is defined as $D_t((G, G')) = \|e^{-tL} - e^{-tL'}\|_F$ where $\|\cdot\|_F$ is the Frobenius norm. The associated HKD-process is $\{D_t((G, G')),\ t \in [0, T]\}$. Hence, it is the evaluation of (G, G') by the family of HKDs $\mathcal{F}_{HKD} = \{D_t,\ t \in [0, T]\}$.

8.2.1 Our methods

Basis method. We propose to apply our two-sample test to the samples of absolute activation graphs computed from the test set and the new data set. Consider the test set $D_1 = (X_1^1, \ldots, X_{N_1}^1)$ and the new data set $D_2 = (X_1^2, \ldots, X_{N_2}^2)$. For a given neural network f, consider that we have a function G such that for each input x, it returns $G(f, x)$, the absolute activation graph of x associated with a sequence of dense layers on f. Then to detected whether f will perform poorly or not on D_2, we propose to apply our two-sample test detailed in Algorithm 3, using the HKD and the samples of activation graphs $(G_1^1, \ldots, G_{N_1}^1)$ and $(G_1^2, \ldots, G_{N_2}^2)$, where $G_i^k := G(f, X_i^k)$, for all $k \in \{1, 2\}$ and $i \leq N_k$. Recall that Algorithm 3 contains a pairing phase where we define samples of pairs of graphs:

$$
\begin{aligned}
D_{12} &= ((G_1^1, G_1^2), (G_2^1, G_2^2) \ldots, (G_k^1, G_k^2)) \\
D_{11} &= ((G_{k+1}^1, G_{k+2}^1), (G_{k+3}^1, G_{k+4}^1) \ldots, (G_{N_1-1}^1, G_{N_1}^1)) \\
D_{22} &= ((G_{k+1}^2, G_{k+2}^2), (G_{k+3}^2, G_{k+4}^2) \ldots, (G_{N_2-1}^2, G_{N_2}^2)),
\end{aligned}
$$

where $k = \min(N_1, N_2)/2$. Set $n_1 := (N_1 - k)/2$ and $n_2 := (N_2 - k)/2$. We assume that k, n_1, and n_2 are integers. If we denote by Q_{12}^k, $Q_{11}^{n_1}$ and $Q_{22}^{n_2}$ the empirical distribution associated to D_{12}, D_{11} and D_{22}, recall that the test statistic is given by

$$
T = \sqrt{\frac{4kn_1n_2}{kn_1 + kn_2 + 4n_1n_2}} \quad \left\| Q_{12}^k D_\cdot - \frac{1}{2}\left(Q_{11}^{n_1} + Q_{22}^{n_2}\right) D_\cdot \right\|_\infty .
$$

In the following, this method will be denoted by `HD`. See Chapter 5 for more details.

Variations. We also evaluate the performances of two variations around `HD`. The first one concerns classification tasks. In that case, the pairing in Algorithm 3 is done such that each pair contains two graphs with the same predicted labels (or true labels[2]) as explained in Section 5.2. This method will be denoted by `HDy`.

The second variation is the multiple testing procedure described in Section 5.3. We shuffle the data and repeat the pairing phase several times. We compute the p-values for each pairing scheme and apply a Benjamini-Hochberg procedure. This

[2]When available.

method will be denoted by `HDmt`.

Bootstrap methods are applied to compute rejection thresholds for `HD` and `HDy`, and to estimate p-values for `HDmt`.

8.2.2 Comparison methods

Recall that `HD` is based on HKD processes. The idea is to compute means of such processes for different families of pairs and compare them using the uniform norm. A simple alternative to this method would be to invert the order in which we take the mean and the supremum over all $t \in [0, T]$. For that, we apply the same pairing procedure as in `HD`. Then, for each pair (G, G'), we compute the *diffusion distance* $\max_t D_t((G, G')$. We compute the means for each family of pairs and compute a similar test statistic. As we are working with real-valued variables now, a standard bootstrap method provides a rejection threshold. This method will be denoted by `hammond`. Comparing `HD` and `hammond` will provide a better understanding of the interest in using distance processes instead of directly working with the diffusion distance.

We also compare our methods to the Black Box Shift Detection [LWS18], denoted by `BBSD`. This method was originally designed to detect shifts in the label distribution, assuming that the distribution of the data conditionally on the label stays unchanged. This type of shift is called *label shift*. However, a survey [RGL19] showed that among various techniques and different types of shifts, not only label shifts, `BBSD` was among the best shift detection methods. The test performed by `BBSD` concerns the network outputs: $(f(X_1^1), \dots, f(X_{N_1}^1))$ and $(f(X_1^2), \dots, f(X_{N_1}^2))$. It applies a Bonferonni multiple testing procedure that uses Kolmogorov-Smirnov tests for each dimension. More precisely, for each coordinate $k \leq K$, consider the Kolmogorov-Smirnov test statistic

$$Z_k = \sup_u \left| F_k^1(u) - F_k^2(u) \right|,$$

where F_k^i is the empirical cumulative distribution function associated to the sample $(f(X_1^i)_k, \dots, f(X_{N_i}^i)_k)$, for each $i \in \{1, 2\}$. As the asymptotic distribution of Z_k under the null hypothesis is known, we can compute the associated p-value p_k. To combine the K tests, having no assumption on the independence of the tests, `BBSD` performs a Bonferroni correction [BA95] and rejects the null hypothesis whenever

$$\min_k p_k \leq \alpha / K.$$

8.3 Dense and convolutional neural networks for MNIST

In this section, we consider the DSD problem in the context of neural network learning on the MNIST data set [LCB10]. We study the performances of our two-sample tests on the activation graphs of clean and corrupted data, and discuss

to which extent it could solve the DSD problem. We analyze the impact of the variations of our methods, namely the pairing with similar predicted labels and the multiple testing procedure. We also compare ourselves to other methods: BBSD introduced by [LWS18] and a test based on the *diffusion distance* introduced by [HGJ13] to study the added value of considering distance processes.

8.3.1 The data and neural networks

We work with MNIST, the data set of hand-written digit images. It is a classification data set where the labels correspond to the digits depicted on the image. The data set is split into 60 000 training images and 10 000 test images. Each image is a black and white image of size 28×28 pixels. The grey value of each pixel is normalized between 0 and 1.

We apply four corruptions to the test data to simulate distribution shifts: Gaussian blur, Gaussian noise, random black pixel, and rotation. Each corruption method depends on a parameter, denoted by κ, that sets the corruption intensity. We now present the corruptions. Figure 8.1 and Figure 8.2 show examples of the original data images and their corrupted versions, for different corruption intensities.

Gaussian blur. For a given image, we apply a Gaussian blur with standard deviation κ. That is, the value of pixel (i, j) in the corrupted image is a weighted average of all the pixel values of the original image where the weights are given by the Gaussian distribution centered in (i, j).

Gaussian noise. For a given image, we add to each pixel value some noise following a centered Gaussian distribution with standard deviation κ. Corrupted pixel values are clipped between 0 and 1, if necessary.

Random Black Pixels. For each image, we select p pixels at random and turn them into black pixels. The number of pixels to select is defined by $p = \min(n_{pix}, P)$, where n_{pix} corresponds to the total number of pixels in the image (*i.e.* $n_{pix} = 784$) and P is the realization of a Poisson variable with expected value κn_{pix}.

Rotation. Each image is rotated by an angle of $\kappa\pi$. Remark that this corruption is deterministic.

In this series of experiments, we work with two different neural network architectures. An elementary dense neural network (DNN) and a slightly more complex convolutional neural network (CNN). For technical definitions on NNs, see Section 2.4.

Dense neural network. The DNN architecture that we use is the following. We consider 3 hidden layers of 16 neurons each. The activation functions are ReLU functions and the activation of the output layer is the softmax. It is trained during

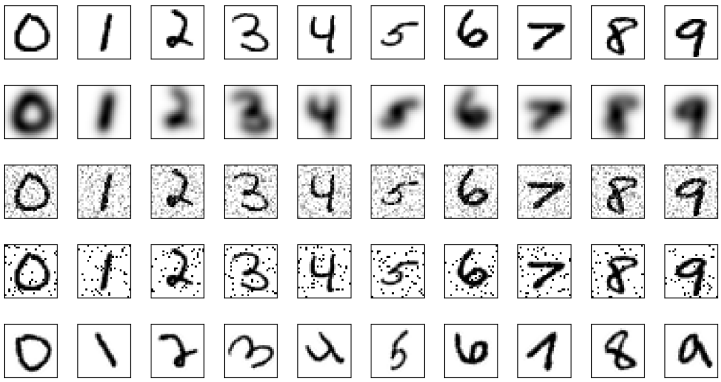


Figure 8.1: Digit images from the MNIST data set and their various corruptions. From top to bottom : original data, Gaussian blur with a standard deviation of 2.7, Gaussian noise with a standard deviation of 0.22, random black pixels with intensity 0.07, and a rotation of 0.23π. The corruption intensities were chosen so that the accuracy of the considered dense neural network is in $[0.49, 0.51]$.

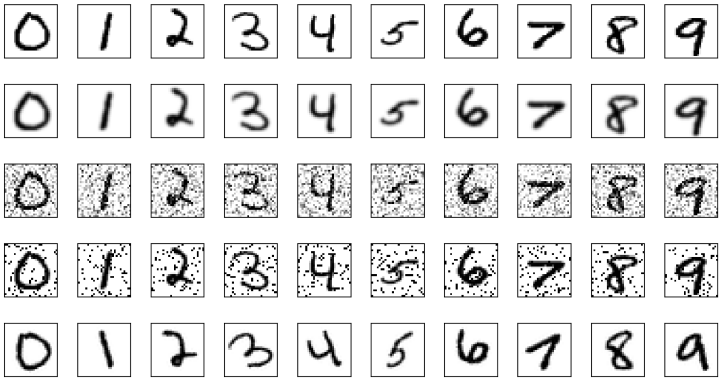


Figure 8.2: Digit images from the MNIST data set and their various corruptions. From top to bottom : original data, Gaussian blur with a standard deviation of 0.7, Gaussian noise with a standard deviation of 0.35, random black pixels with intensity 0.1, and a rotation of 0.2π. The corruption intensities were chosen so that the accuracy of the considered convolutional neural network is in $[0.9, 0.95]$.

50 epochs and the loss function is the categorical cross-entropy. At the end of the training phase, the accuracy on the clean test set is 0.9567.

Convolutional neural network. The CNN architecture is as follows. We apply two series of 8 convolution filters with 3×3-kernels. Both are followed by a *max pooling* of size 2×2. The results are then flattened and go through 3 dense layers of 32 neurons each. All activation functions are ReLU, except for the output layer, which has a softmax activation. It is trained for 50 epochs using categorical cross-entropy. The accuracy on the test set is 0.9899.

8.3.2 Levels and powers study

The first part of this section aims to verify that all methods are well calibrated in terms of level. We fix a target level $\alpha = 5\%$ and we want to confirm that the actual levels are upper-bounded by this prescribed α. The interest of this study is twofold. First, it performs a sanity check to ensure that the methods and their implementations behave as expected. Secondly, it will allow us to interpret differences in power performances in the following sections.

The level experiment. For each neural network architecture, method, and sample size N, we draw two subsamples D_1 and D_2 of size N. D_1 is taken from the training data and D_2 from the clean test data. We apply the methods to perform two-sample tests on D_1 and D_2. Assuming that the training and test data follow the same distribution, the right decision for the tests should be to conserve H_0. We repeat this procedure n_{estim} times and compute the error rate to estimate the level of each method. The results are presented in Figure 8.3 for the DNN and in Figure 8.4 for the CNN. We plot the level estimations with respect to the sample sizes.

Remark 8.1. For a given method, the n_{estim} tests are not independent, as for each iteration, we draw subsamples from the same training and test data. Nonetheless, as N is a few orders of magnitude smaller than the size of the original samples, we perform the analysis as if the tests were independent.

Results. We observe from Figure 8.3 and Figure 8.4 that for both the DNN and CNN, all confidence intervals of the level estimations either contain the prescribed value $\alpha = 5\%$ or are below this value. This indicates that the methods are well calibrated and that we can safely interpret the differences in power in the next section.

However, note that for `HDy` tested on the DNN, the confidence interval stays below α for all sample sizes. And for `BBSD` on the CNN, the confidence interval is below α for $N = 100$. For the latter, the explanation could be that the asymptotic behavior of the test statistics may not have been reached with $N = 100$. For the former, as the phenomenon is present for all sample sizes up to $N = 400$, it is less likely due to the non-asymptotic nature of the experiment. A possible explanation could be that the level estimations are made by computing error rates of *a priori* dependent tests, as mentioned in Remark 8.1.

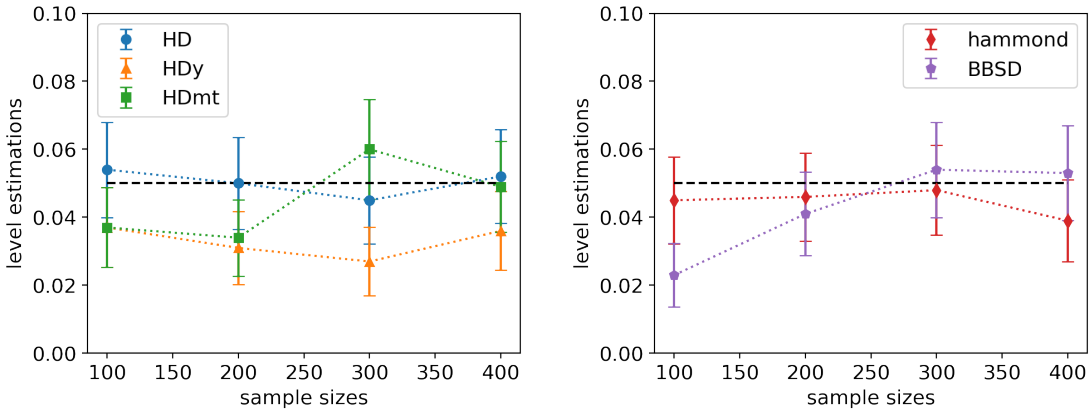


Figure 8.3: For the dense neural network. These plots represent the level estimations with respect to the sample sizes. On the left, we show the levels of our methods (`HD`, `HDy`, `HDmt`). On the right we show the comparison methods (`hammond`, `BBSD`). Level estimations are computed from $n_{estim} = 1000$ iterations for each method and sample size. Vertical segments represent the confidence intervals at 95% of the level estimations. The horizontal black dashed lines represent the level $\alpha = 5\%$ initially prescribed.

For all methods, the level calibration is based on asymptotic considerations, either from bootstrap methods or from the knowledge of the asymptotic distribution of the test statistics. Nonetheless, despite sample sizes being of the order of a few hundred, the level calibration seems to perform well. For the heat distance processes based methods `HD`, `HDy` and `HDmt`, this assessment correlates with the theoretical results about the speed of convergence toward Gaussian processes derived in Part I. Recall that we proved a Gaussian approximation result with a rate independent of the sample size. Moreover, we refined this result by proving that the constants involved have a polynomial dependency in the sample size.

The calibration of the methods being validated, in the rest of this section we study the capacity of the various methods to detect an actual distribution shift. This is measured by the *power* of the tests, defined as the probability that the test rejects H_0, knowing that H_1 is verified.

The power experiment. For each neural network architecture, corruption type, method, and sample size N, we draw two subsamples of size N, D_1 and D_2, from the training data and the corrupted data, respectively. Recall that corrupted data are the test data to which we apply the various corruptions. We apply the methods to perform two-sample tests on D_1 and D_2. Here, as D_1 and D_2 do not follow the same distribution, the right decision for the tests should be to reject H_0. We repeat this procedure n_{estim} times and compute the proportion of right calls for each method, which estimates the power. Remark 8.1 also apply in this set of experiments, as the tests are *a priori* dependent. The corruption intensities are fixed and correspond to the values presented in Figure 8.1 and Figure 8.2. They are chosen so that the

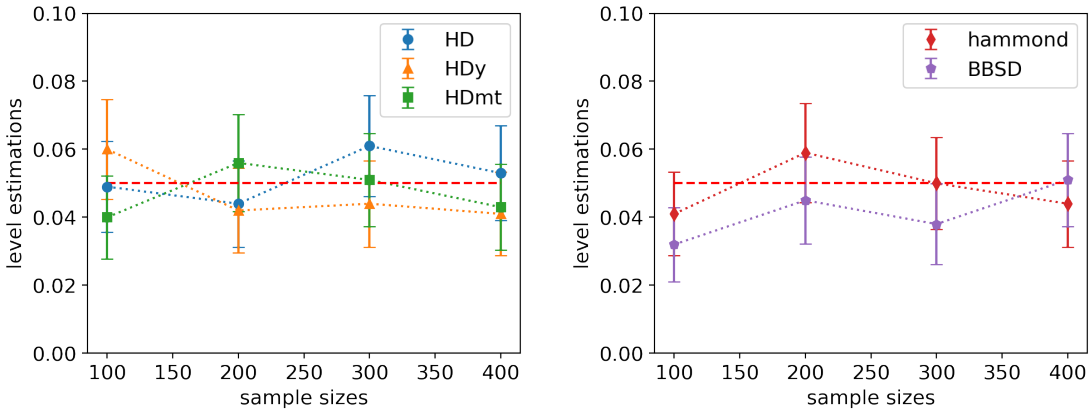


Figure 8.4: For the convolutional neural network. These plots represent the level estimations with respect to the sample sizes. On the left, we show the levels of our methods (`HD`, `HDy`, `HDmt`). On the right we show the comparison methods (`hammond`, `BBSD`). Level estimations are computed from $n_{estim} = 1000$ iterations for each method and sample size. Vertical segments represent the confidence intervals at 95% of the level estimations. The horizontal black dashed lines represent the level $\alpha = 5\%$ initially prescribed.

accuracies of the DNN and CNN on the corrupted data sets stay in $[0.49, 0.51]$ and $[0.9, 0.95]$, respectively. The results are presented in Figure 8.5 and Figure 8.6. They display the power estimations with respect to sample sizes. Each plot corresponds to a corruption type and we plot the power estimations of all methods.

Results. We remark that the heat-distance-processes-based methods seem to have their powers converging to 1 when the sample size increases. This is at least what we can read from all plots on Figure 8.5 and the top plots, associated with Gaussian blur and Gaussian noise, on Figure 8.6. This observation agrees with the theoretical results from Part I (see Section 5.1.3). Moreover, the variations `HDy` and `HDmt` seem to generally perform better than `HD`. This increase in power is particularly visible for `HDy` for all corruptions with the CNN architecture (Figure 8.6). It is also significant for pixel and rotation corruptions with DNN architecture. Concerning `HDmt`, it is significantly better for Gaussian noise, pixel, and rotation corruptions both on DNN and CNN architectures.

Remark 8.2. It is difficult to explain the performance variations with respect to the different corruption types. Even though we tried to balance the various corruptions by controlling the accuracy drop (accuracy around 0.5 for the DNN and 0.9-0.95 for the CNN), all methods depend on networks' variables such as values in the output layer (for `BBSD`) or activation graphs (for all other methods). As a result, the modifications in how neural networks process the corrupted data compared to the clean data are a determining factor when it comes to understanding the methods' performance. Unfortunately, we do not have an understanding of the impact of the corruptions on the neural networks' working.

The power performances of `hammond` are mainly worse than `HD`, except for the rotation corruption with DNN and CNN where differences between the two methods are not significant. In every other situation, considering the heat distance processes and comparing them with the uniform norm is more informative and powerful than directly applying the norm to each distance process and comparing the norms afterward. This can be understood from the fact that the best diffusion time to discriminate between the two samples is not the same as the ones that discriminate pairs of graphs. See Figure 8.7, where the suprema of the individual processes are essentially located at the end of the time range, while the discriminative time between the samples is located roughly at the first third of the time range. Hence, this set of experiments demonstrates the added value of considering the HKD processes instead of the diffusion distances in the context of two-sample tests.

However, note that `BBSD` outperforms all other methods. Even though for big enough sample sizes our heat-distance-processes-based methods can have comparable powers (*e.g.*, see Gaussian blur corruption with DNN and CNN, or Gaussian noise corruption with CNN), power estimations of `BBSD` reach 1 for smaller sample sizes on all plots.

8.3.3 Application to the distribution shift detection problem

The analyses made in the two previous sections were meant to demonstrate the ability of the proposed methods to solve the two-sample tests problem applied in the context of DSD. To that regard, levels and powers were investigated, and the conclusion was that variations `HDy` and `HDmt` were better than `HD`, but all methods were outperformed by `BBSD`. In this section, we put these two-sample test performances into perspective with the shift detection problem. We want to compare the ability of the methods to detect distribution shifts with the accuracy drops of the neural networks. As explain in the beginning of this chapter, if neural networks have some robustness, a shift detection systems should not send warnings to the user as soon as the distribution starts to shift, if the network is still performing well. Rather, we want to be able to detect when the shift starts to have a significant impact on the network performances.

Experiments. For this set of experiments, we work with the whole test set and the whole corrupted set. We incrementally increase the corruption intensity. For each intensity, apply the two-sample testing methods on the test set and the corrupted set. As we use whole data sets, only one iteration is computed. We study the results of these tests. In particular, we are interested in the intensity thresholds from which methods start to detect the shift. These thresholds are to be compared with the accuracy drops of the network. We also compute mean confidences on corrupted data for each corruption intensity, as it is a classical baseline [HG17]. Moreover, we record the intensity from which the error rate of the network increases by more than 50% compared to the error rate on the test set. Recall that the error rate is the proportion of wrong predictions made by the neural network, *i.e.* it is 1 minus the accuracy. The choice of the value 50% is arbitrary, but this intensity threshold

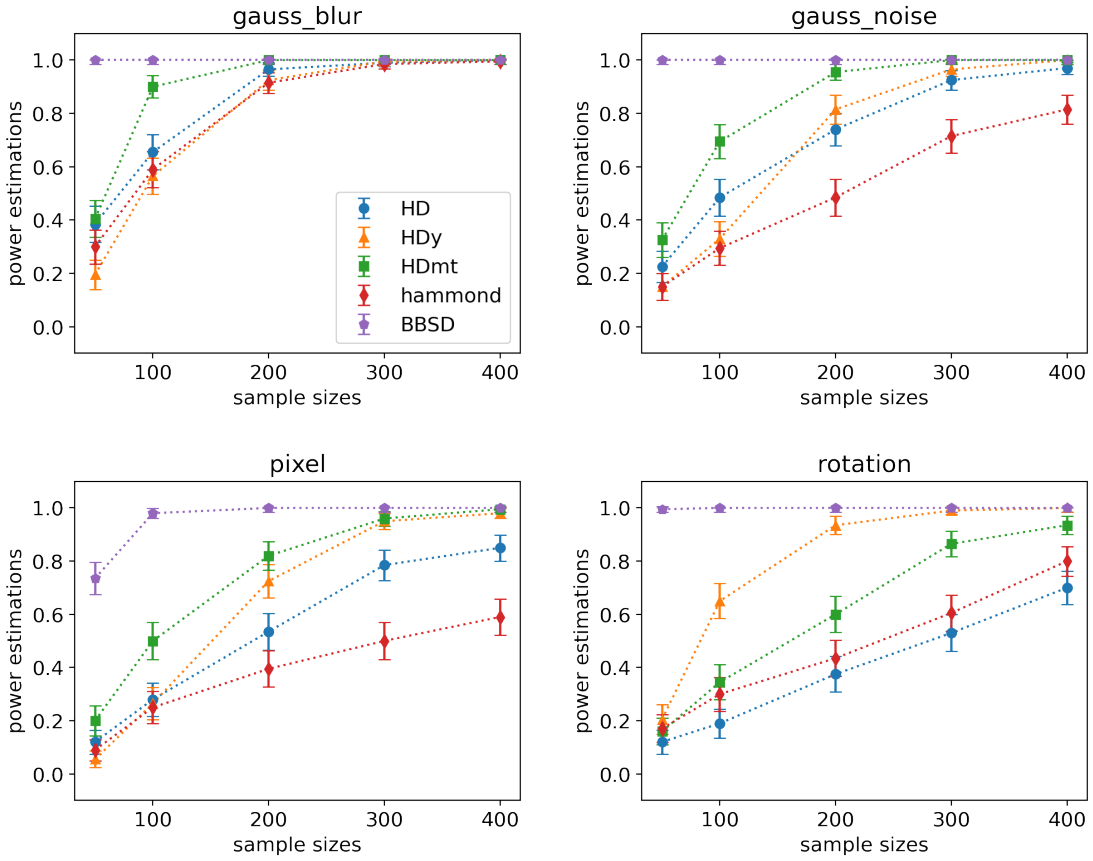


Figure 8.5: For the dense neural network. These plots represent the power estimations with respect to the sample sizes, for the different corruptions types (Gaussian blur, Gaussian noise, random black pixels, rotation), and for the different methods as indicated in the legend on the top left plot (`HD`, `HDy`, `HDmt`, `hammond`, `BBSD`). Power estimations are computed from $n_{estim} = 200$ iterations for each method, sample size, and corruption type. Vertical segments represent the confidence intervals at 95% of the power estimations.

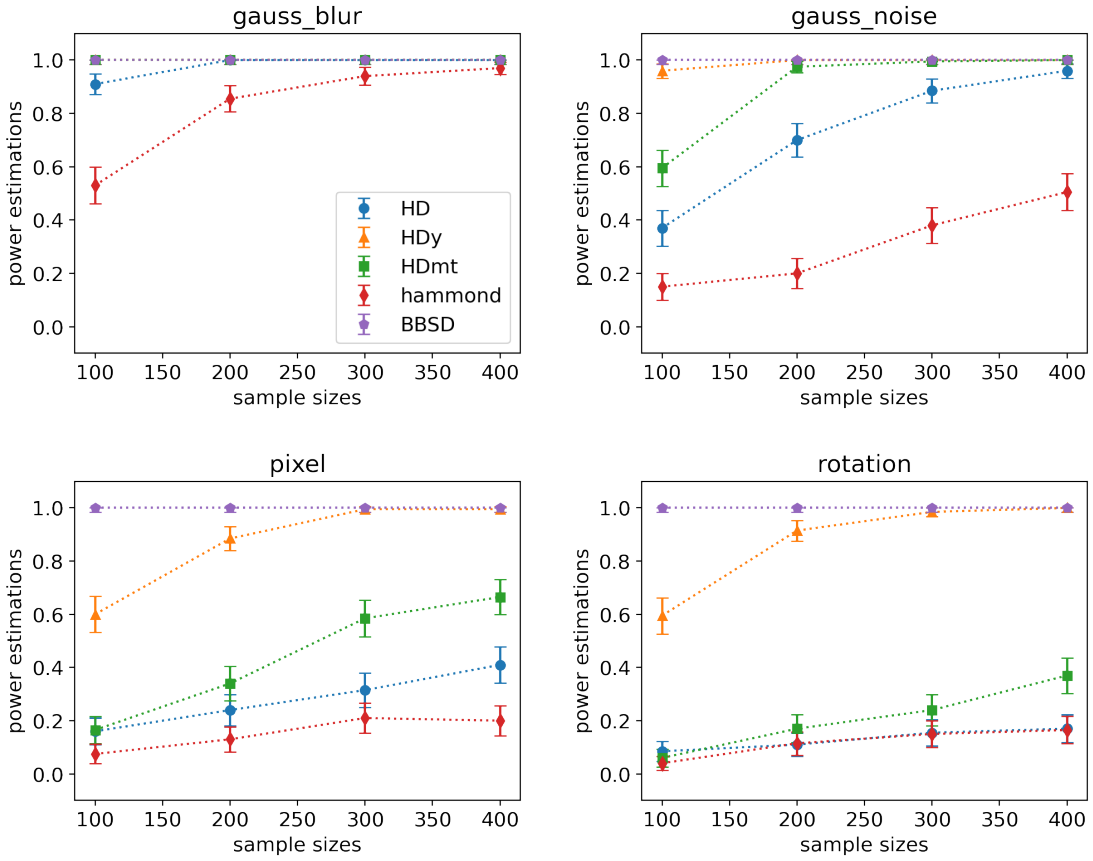


Figure 8.6: For the convolutional neural network. These plots represent the power estimations with respect to the sample sizes, for the different corruptions types (Gaussian blur, Gaussian noise, random black pixels, rotation), and for the different methods as indicated in the legend on the top left plot (`HD`, `HDy`, `HDmt`, `hammond`, `BBSD`). Power estimations are computed from $n_{estim} = 200$ iterations for each method, sample size, and corruption type. Vertical segments represent the confidence intervals at 95% of the power estimations.

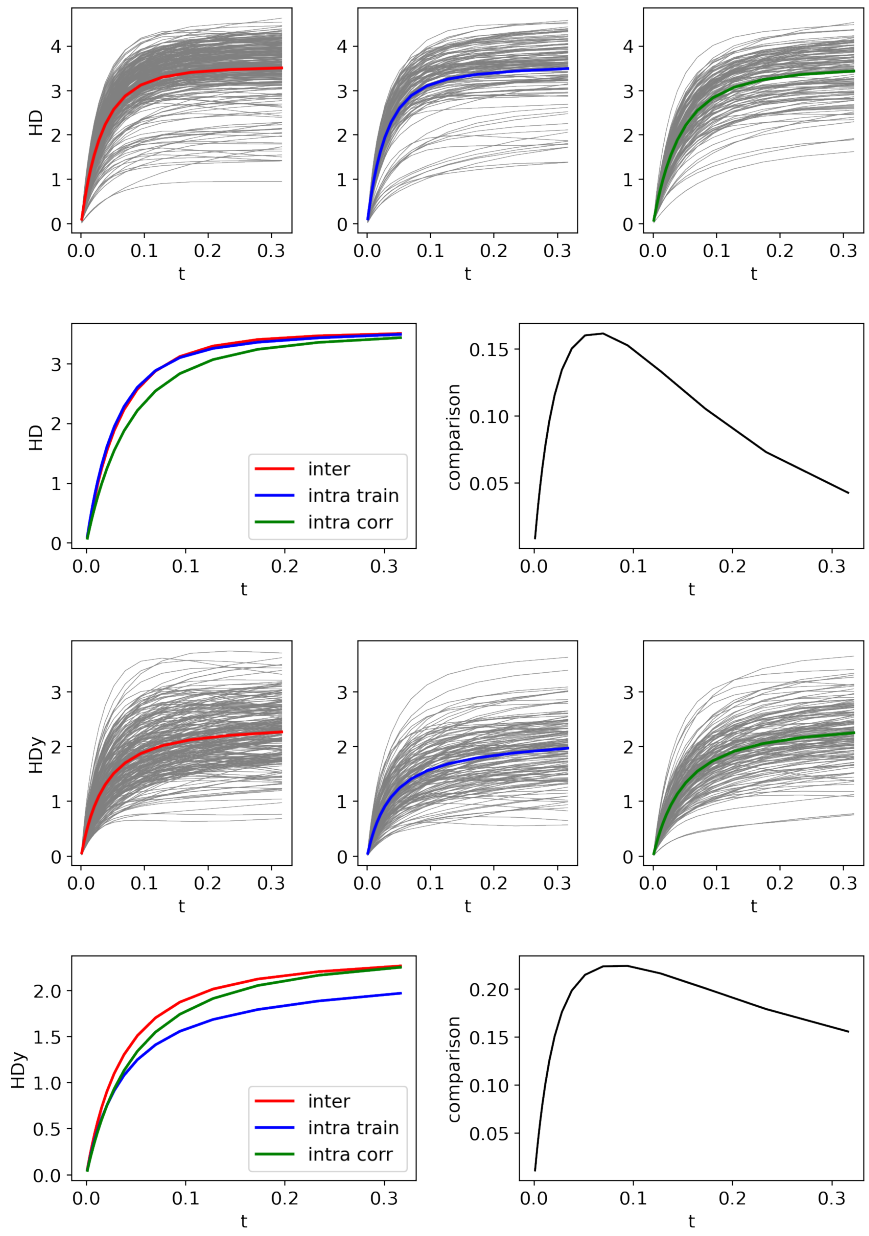


Figure 8.7: Heat distance processes after the pairing procedure between subsamples of size 500 from the train set and the pixel corrupted set. The first two rows correspond to the pairing procedure in `HD`, while the last two are for the one in `HDy` where only data with the same predicted label are paired. On rows 1 and 3, from left to right, we plot the HKD processes of the inter-samples pairs, intra-train-sample pairs, and intra-corrupted-sample pairs. The thin gray lines are the individual processes of each pair. The thicker colored lines are their empirical means. On rows 2 and 4, we display the three empirical means on the left. On the right, we plot the comparison of the means by computing the absolute value of the difference between the inter mean and the average of the inside means. This quantity is the one used to compute the test statistic.

provides a standard way of deciding if the network performances have dropped too much. Figure 8.8 and Figure 8.9 present the results of this set of experiments.

Results. Firstly, we observe that the `BBSD` method, which was outperforming heat-distance-processes-based methods in terms of power of the two-sample test, happens to be too sensitive to changes in the distribution. Except for the Gaussian blur corruption with CNN, in all other scenarios, it rejects H_0 at very low corruption intensities where the accuracy is still comparable to the one on the test set. This indicates that `BBSD` does not capture the behavior of the neural network with enough precision to distinguish between harmful and unharmful shifts. On the other hand, `hammond` detects the distribution shifts when the performances of the network have already significantly dropped. This fits with the findings of the previous section, where the power was essentially smaller than all other methods.

In the end, in our set of experiments, the methods based on absolute activation graphs and heat distance processes detect distribution shifts for reasonable accuracy drops. Except for the pixel corruption with DNN, where the detection happens for too big intensities, they usually detect a distribution shift close to the 50% increase of the error rate.

Remark 8.3. The mean confidence, frequently used to monitor the performances of neural networks on new unlabeled data, appears to be unreliable. While it follows the accuracy curve for the CNN architecture (Figure 8.9), it does not drop as much as the accuracy for the DNN architecture (Figure 8.8). This coincides with similar behaviors in the context of adversarial attacks.

Remark 8.4. Due to randomness and to the fact that we use whole data sets without repetition, some plots may not display a clear transition from no detection to detection as the corruption intensity increases.

8.3.4 Discussion

The results of this section show that the heat-distance-processes-based methods applied to absolute activation graphs for detecting distribution shifts perform well under various considerations. First, their levels are controlled by the prescribed value. It is interesting to notice, as theory only supports a control of the asymptotic level. The experiments of this section demonstrated that even for a sample of a few hundred, levels were controlled. Secondly, the experiments also proved the advantage of using heat distance processes compared to using the *Diffusion distance* that computes the maxima of the distance processes for each pair of graphs. Power performances for the heat-distance-processes-based methods were generally better than `hammond`. Moreover, the variations of `HD`, namely `HDy` and `HDmt`, did enhanced the test performances. Finally, regarding the test performances, `BBSD` outperformed all other methods. However, when applied in the context of distribution shift detection, `BBSD` turned out to be too sensitive, returning warning signals while the accuracy of the neural networks was still high. On the contrary, `hammond` was not

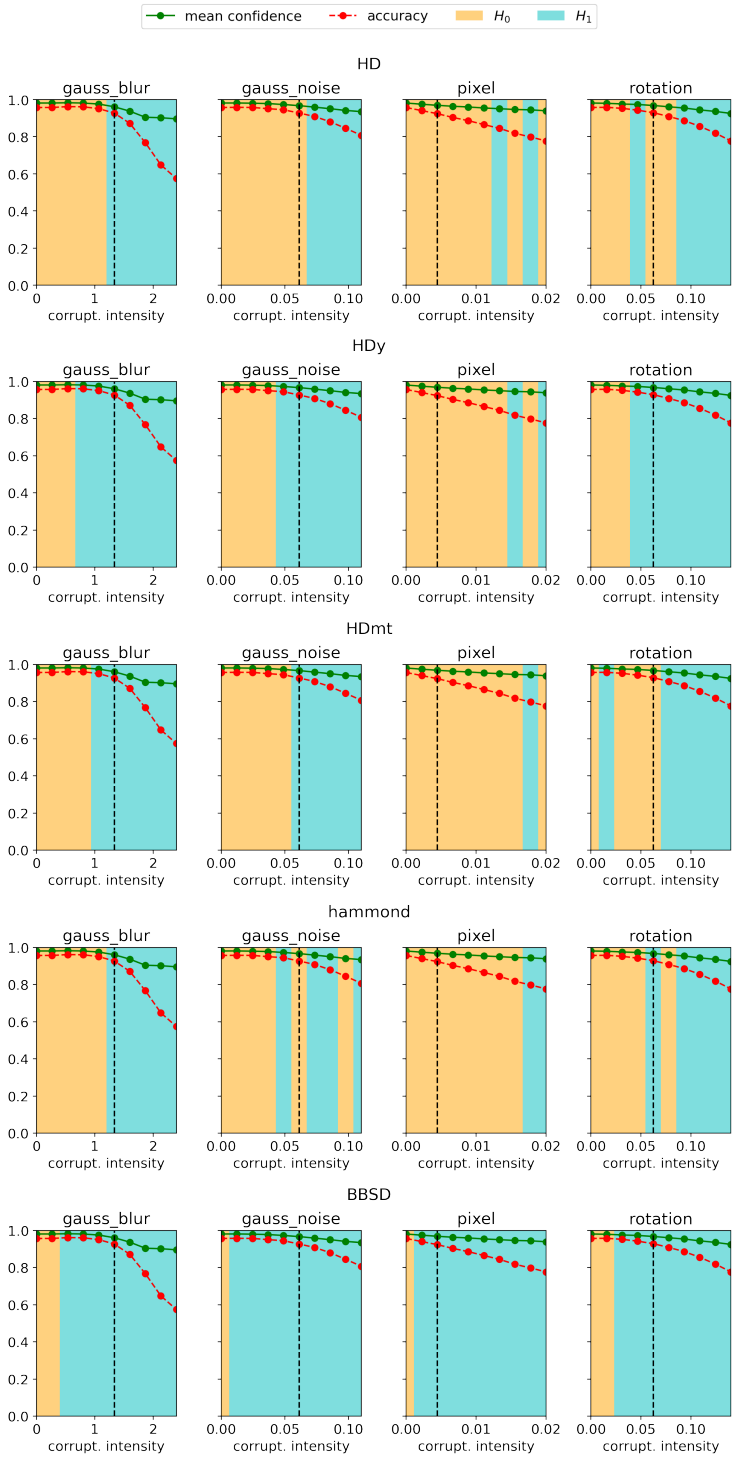


Figure 8.8: For the dense neural network. Accuracy and mean confidence are plotted with respect to corruption intensities. Each row corresponds to a method and each column to a corruption type. Colored areas represent the methods' decisions to accept or reject H_0, based on the whole test set and corrupted set. The vertical dashed lines represent the first intensity at which the error rate increases by more than 50%.

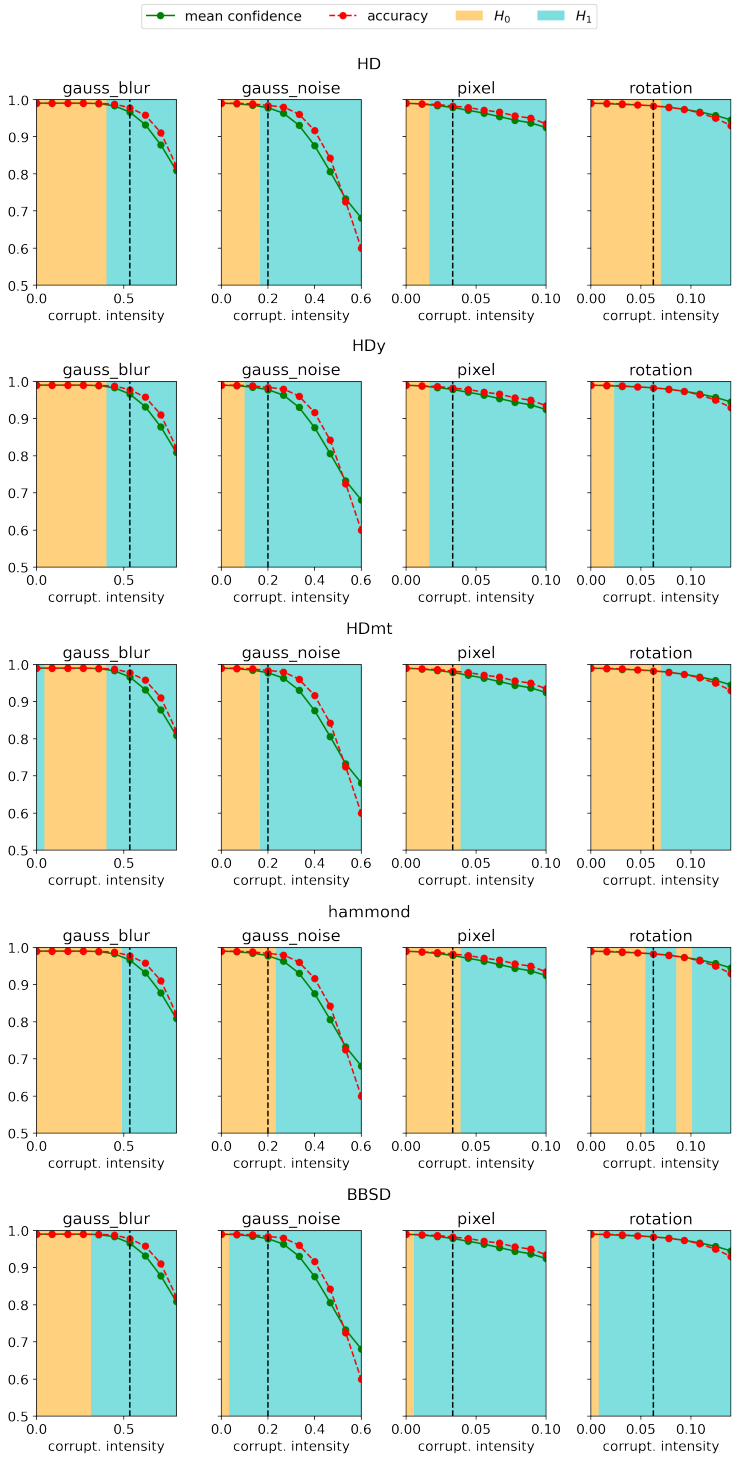


Figure 8.9: For the convolutional neural network. Accuracy and mean confidence are plotted with respect to corruption intensities. Each row corresponds to a method and each column to a corruption type. Colored areas represent the methods' decisions to accept or reject H_0, based on the whole test set and corrupted set. The vertical dashed lines represent the first intensity at which the error rate increases by more than 50%.

powerful enough to detect changes before significant drops in accuracy. All three heat-distance-processes-based methods detected shifts at a reasonable stage, close to the point where the error rate increased by 50%. In the end, these conclusions might indicate that activation graphs are accurate descriptors of the behavior of neural networks and that our proposed methods `HD`, `HDy` and `HDmt` are adapted tools to analyze these descriptors in the context of distribution shift detection.

We should point out that these experiments were performed on elementary neural networks, compared to state-of-the-art architectures, and on the MNIST data set that is not as complex as some other real-world data sets. In the next section, we investigate the performance of our methods on a more complex neural network architecture developed to infer topological descriptors of point clouds, namely *persistence images*.

8.4 Ripsnet

In this section, we investigate the use of our heat-distance-processes-based methods to detect distribution shifts on more atypical data than MNIST, namely point cloud data, processed by a more complex neural network called *Ripsnet*. Ripsnet has been introduced by [dSHC+22] in order to estimate topological descriptors of point clouds called persistence images (PIs) (see Section 2.2.4). Their exact computation being costly and noise sensitive, the authors propose a faster and more robust alternative: train a specific neural network to estimate these topological descriptors.

8.4.1 Introduction to Ripsnet

The authors' objective in [dSHC+22] was to propose a network architecture able to estimate PIs of test point clouds after being trained on training point clouds. The function, that associates to a point cloud $X = \{x_1, \dots, x_k\} \subset \mathbb{R}^d$ its PI, is permutation invariant. Hence, the authors wanted to ensure that the proposed neural network verifies this property. They based their architecture on the one of *DeepSets* [ZKR+17] such that point clouds are processed by

$$\textbf{Ripsnet} : X = \{x_1, \dots, x_k\} \mapsto \phi_2 \left(\textbf{op} \left(\{\phi_1(x_i)\}_{1 \leq i \leq k} \right) \right),$$

where $\phi_1 : \mathbb{R}^d \to \mathbb{R}^{d'}$ and $\phi_2 : \mathbb{R}^{d'} \to \mathbb{R}^K$ are parametrized function (here dense neural networks) and **op** is a permutation invariant operator (here the mean). ϕ_1 and ϕ_2 optimized during the training phase. ϕ_1 is applied to each point of the point cloud. It provides a d'-dimensional representation of each point. Then these representations are gathered through the permutation invariant operator **op**, hence, making sure that the resulting function is itself permutation invariant. Finally, the representation of the point cloud obtained from **op** is processed by ϕ_2 to return the estimation of the PI. Classical topological descriptors of point clouds such as PIs are stable with respect to the *Hausdorff* distance (see Definition 2.6) which is quite sensitive to noise (*e.g.,* introduction of outliers). Here, the authors proved that the estimation of the PIs with Ripsnet is stable with respect to the 1-Wasserstein distance, which is best suited for working with noisy data.

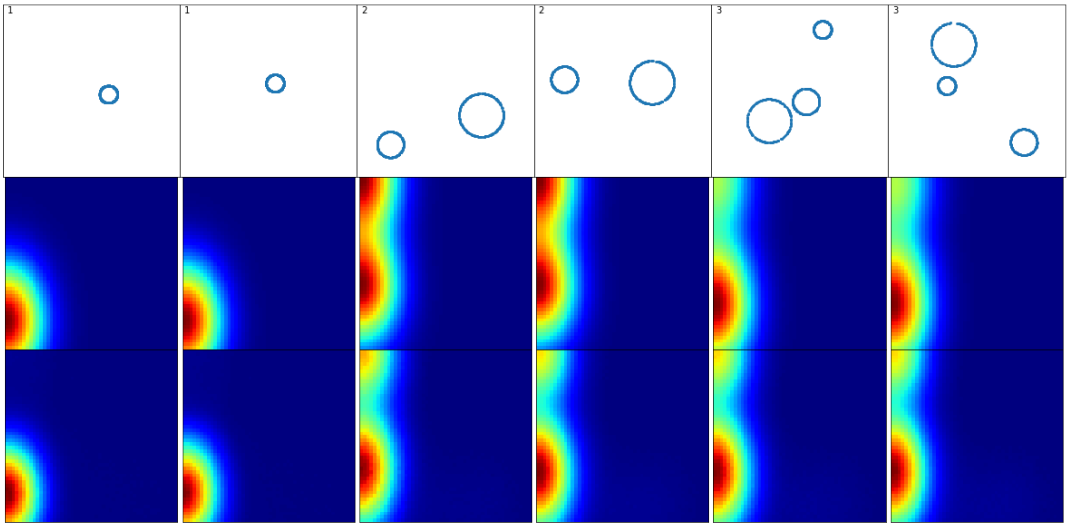


Figure 8.10: Examples of clean data and PIs. From top to bottom, the point cloud data, their PIs computed with the exact method, and their PIs estimated by Ripsnet. Two examples of each of the 3 classes (1 circle, 2 circles, 3 circles) are displayed. Each point cloud has 600 points.

8.4.2 The data

We experiment with the same synthetic data as [dSHC+22]. We consider random point clouds in $\mathbb{R}^2$ of 600 points arranged on either 1, 2, or 3 circles. The number of circles determines point cloud labels. For point clouds with label 1, the circle has a radius of 2 and the center is chosen uniformly at random in the square $[-5, 5]^2$. For point clouds with label 2 or label 3, the radii of the circles are 3 and 5, or 2, 3, and 5, respectively. The centers are chosen uniformly at random in $[-15, 15]^2$. All point clouds are sampled conditionally on having non-intersecting circles.

In the following experiments, we consider two types of shift: a data distribution shift and a label shift. For the former, we shift the data by adding noise to the models. A fraction λ of the 600 points are outliers distributed uniformly on the square $[-20, 20]^2$. Note that this domain contains all point clouds. The latter consists in sampling data sets of point clouds with an unbalanced number of classes. For $\mu \geq 0$, the proportion of label 1, 2 and 3 are proportional to 1, $1 + \mu$, $1 + 2\mu$.

Figure 8.10 represents examples of clean data and their corresponding PIs, while Figure 8.11 display noisy examples with $\lambda = 0.1$. The exact PIs on noisy data are distorted compared to the ones on clean data. This demonstrates the high sensitivity to noise of the exact computation of PIs. On the other hand, the estimated PIs on noisy data are quite similar to the exact PIs on clean data. This shows the robustness of the Ripsnet estimations. However, note that the estimation of the PIs on clean data is not perfect, as it seems difficult to distinguish between class 2 and class 3 from the estimated PIs.

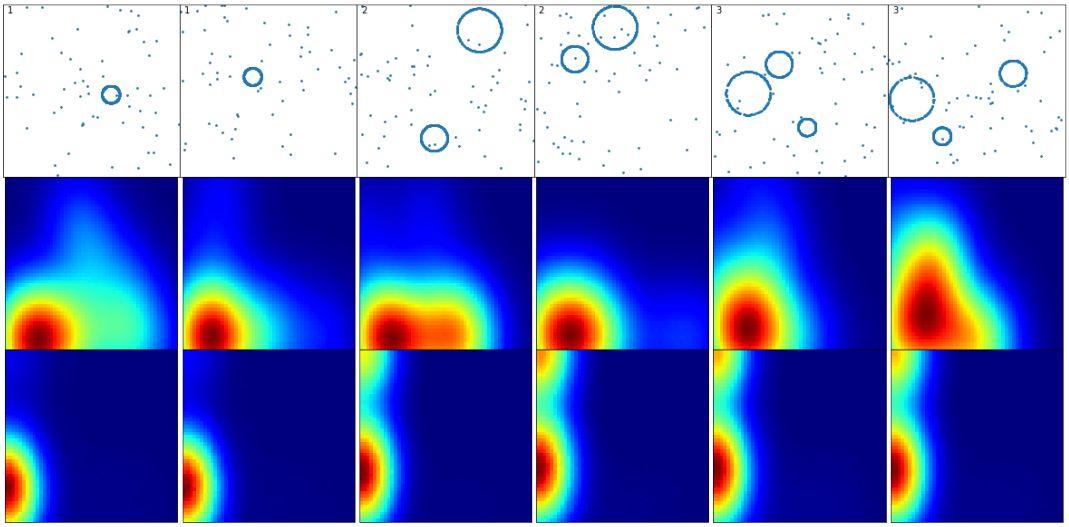


Figure 8.11: Examples of noisy data and PIs. From top to bottom, the point cloud data, their PIs computed with the exact method, and their PIs estimated by Ripsnet. Two examples of each of the 3 classes (1 circle, 2 circles, 3 circles) are displayed. Each point cloud has 600 points. For the noisy data, 60 of the 600 points ($\lambda = 0.1$) contribute to noise and are distributed uniformly on the square.

8.4.3 The experiments

In this series of experiments, we compare the performances of `HDy` and `BBSD`. The setting is as follows. First, a version of Ripsnet is optimized on some training data containing clean point clouds and exact PIs. Then, on another training data set containing clean point clouds and labels, we estimate the PIs using Ripsnet and train an XGboost classifier [CG16] on the predicted PIs. The resulting predicting function is denoted by **XGboost**. Both training samples contain clean point clouds. Note that **XGboost** predicts labels only based on the predicted PIs, not considering the original point clouds. In the end, we obtain a classifier f such that for a point cloud X, $f(X) := \mathbf{XGboost}(\mathbf{Ripsnet}(X))$ is the predicted label of X; see Figure 8.12.

Recall that $\mathbf{Ripsnet} := \phi_2 \circ \mathbf{op} \circ \phi_1$, where **op** is the mean operator and ϕ_1 and ϕ_2 are dense neural networks chosen as follows. ϕ_1 has one hidden layer of 10 neurons and an output layer of 5 neurons. ϕ_2 has hidden layers of size 50, 100, and 200 neurons and an output layer of 2500 neurons to match the 50×50 pixels PIs. Similarly to the original experiment on Ripsnet, activation functions are $GELU : x \mapsto x\Phi(x)$ (where Φ is the cumulative distribution function of the standard Gaussian distribution), except for the output of ϕ_2 where the activation is the sigmoid function. The training of **Ripsnet** is done with a training set of 900 clean point clouds on 250 epochs. The training of **XGboost** is done on another training set of 900 clean point clouds.

Results of the classifier. We test f on clean and noisy test data. Each sample is of size 1000. The results are shown in the confusion matrices in Figure 8.13 and the accuracy on the clean data set is 0.79 and 0.62 on the noisy ($\lambda = 0.1$) data set.

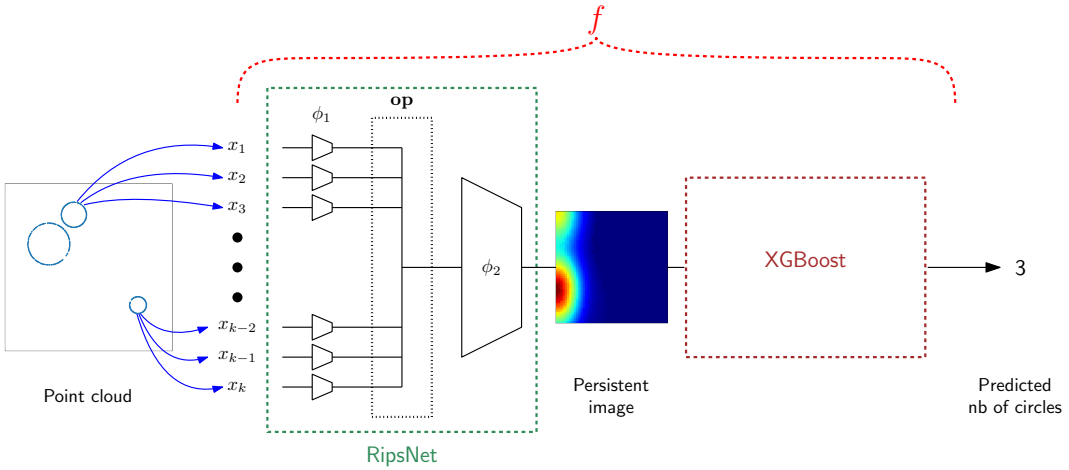


Figure 8.12: An illustration of the classification pipeline $f = \mathbf{XGboost} \circ \mathbf{Ripsnet}$ designed to predict the number of circles in point cloud data.

From the confusion matrix, we observe that on clean data, the classifier has trouble distinguishing between label 2 and 3. However, the proportions of the predicted labels stay balanced, around one-third for each label. This is no longer the case on noisy data, where the proportions of label 1, 2, and 3 are respectively close to 20%, 33%, and 47%. The classifier seems to be biased towards predicting more circles on noisy data.

Data distribution shift detection. We now study how distribution shift detection methods `HDy` and BBSD perform with Ripsnet. To clarify, we apply `HDy` by using the absolute activation graphs corresponding to ϕ_2 and the predicted labels given by f. `BBSD` is applied using the output values of ϕ_2. We sample 1000 clean and 1000 noisy point clouds and apply the methods. We repeat this procedure 20

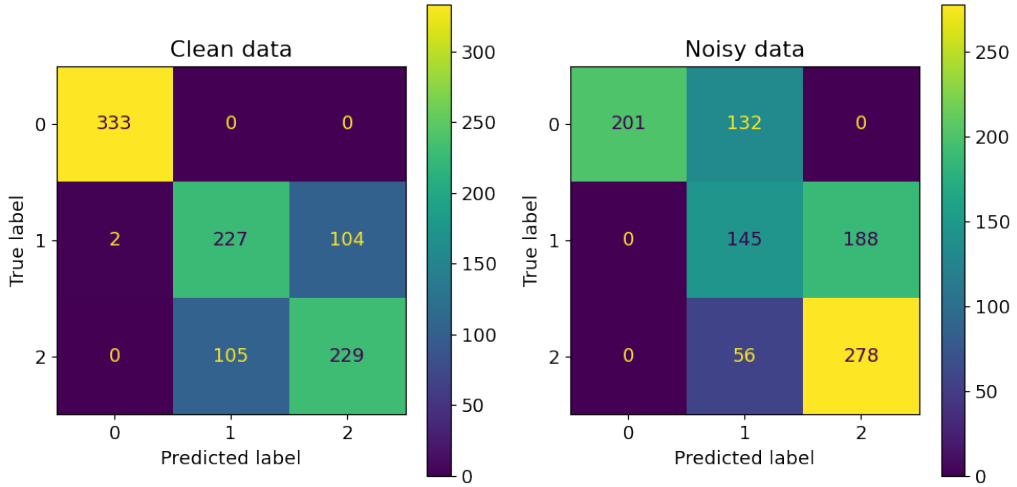


Figure 8.13: Confusion matrices for the classifier f on clean test data (left) and noisy test data (right) with an outlier proportion of $\lambda = 0.1$.

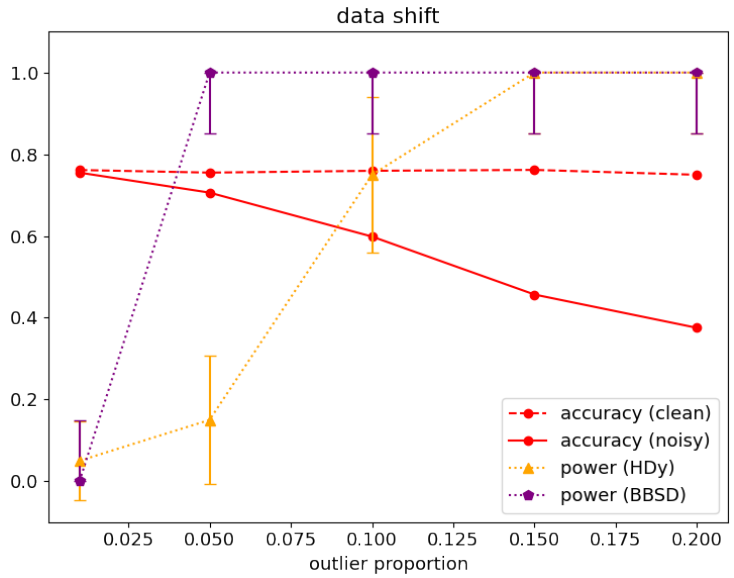


Figure 8.14: Power estimations of `HDy` and `BBSD` two-sample tests under a data distribution shift and mean accuracy of the classifier $f = \textbf{XGboost} \circ \textbf{Ripsnet}$ on clean and noisy data, with respect to the outlier proportion (*i.e., shift intensity)*. Power estimations and mean accuracy are computed over 20 iterations. Data sets contain 1000 point clouds. Vertical segments represent the confidence intervals at 95% of the power estimations.

times to estimate the power of the tests. Powers are estimated for various outlier proportion $\lambda \in \{0.01,\ 0.05,\ 0.1,\ 0.15,\ 0.2\}$. The results of the tests are in Figure 8.14. We also display the mean accuracy of f on the clean and noisy data sets to provide an idea of the perturbations produced by the shift on the classifier.

The results show that `BBSD` outperform `HDy` in the shift detection. Even for only 5% of outliers, it detects the shift on all iterations. `HDy`, on the other hand, detects the shift on all iterations for at least 15% of outliers. However, it is not clear what each method is detecting. Remember that in the classification results in Figure 8.13, we observed that the proportions of the predicted classes were shifted toward more circles. Moreover, from Figure 8.10 and Figure 8.11, the distribution of the estimated PIs is probably three-modal (one mode for each class). As the classification is made by **XGboost** from the estimated PIs, it is possible that the unbalance in the predicted label proportion of f comes from unbalance in the mode distribution of predicted PIs. If this was the case, this could explain the great performance of `BBSD`, as the method was originally designed to detect shifts in the label proportion and not in the distribution conditionally on the labels.

Label shift detection. The previous remark motivates the following experiment. Instead of a data distribution shift, we study the performance of `BBSD` and `HDy` under a label shift. We propose to draw clean data with balanced label proportions and shifted data where label 1, 2, and 3 have proportions proportional to 1, $1+\mu$, $1+2\mu$, respectively, for $\mu \geq 0$. This way, when $\mu = 0$ labels are balanced, and for $\mu = 1$

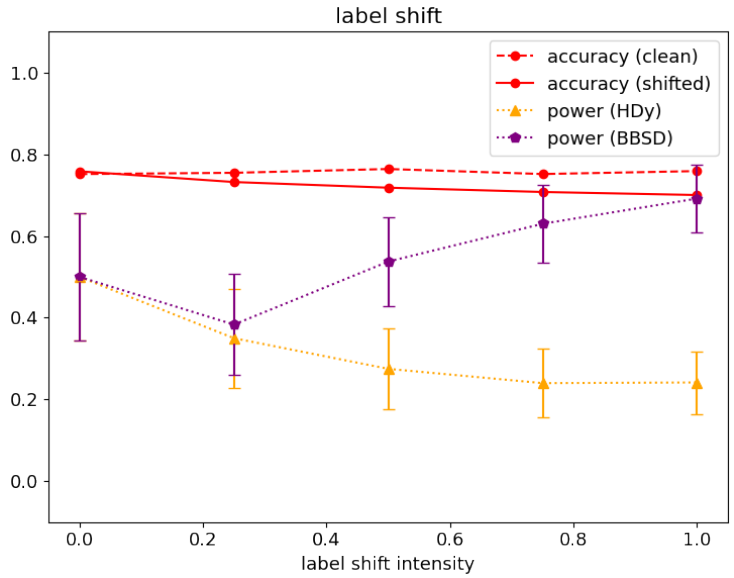


Figure 8.15: Power estimations of `HDy` and `BBSD` two-sample tests under a label shift and mean accuracy of the classifier $f = \textbf{XGboost} \circ \textbf{Ripsnet}$ on clean and shifted data, to the label shift intensity μ. Data sets contain 1000 point clouds. Power estimations and mean accuracy are computed over 20 iterations. Vertical segments represent the confidence intervals at 95% of the power estimations.

we are close to the unbalance observed on the predicted labels in Figure 8.13. We perform 20 tests for each $\mu \in \{0,\ 0.25,\ 0.5,\ 0.75,\ 1\}$, with data sets of size 1000. We present the results in Figure 8.15.

In this situation, we observe that the label shift has much less impact on the classification than in the case of a data distribution shift. The error rate goes from around 0.25 for clean data to about 0.3 on shifted data with $\mu = 1$. However, even for a slight increase of the error rate, `BBSD` seems to have relatively high power, above 0.6 for $\mu = 1$. On the other hand, `HDy` seems to have a more reasonable behavior, with power around 0.25 for $\mu = 1$.

8.4.4 Discussion

Similarly to the previous section on MNIST, the results of the experiments with Ripsnet indicate that `BBSD` has a greater sensitivity as a pure two-sample test method. In the experiments, its power was always greater than that of `HDy`. However, it seems to be more oblivious to the network's performance. As a result, it tends to be oversensitive, even in situations where the network is still performing well. On the contrary, `HDy` is more aware of the network's behavior. And although its power is not as good as `BBSD` in situations where there is an apparent loss of performance of the network, it is more cautious and avoids false alarms when the network is still performing reasonably well.

This series of experiments tends to demonstrate that activation graphs capture well neural networks' behaviors, even in the case of more complex neural networks

such as Ripsnet. Moreover, the HKD processes and the tools to analyze them allow us to compare these activation graphs faithfully and retrieve meaningful information. In the end, we are able to capture changes in the behavior of the neural networks.

Conclusion

In this thesis, we wanted to develop new tools to analyze data sets of graphs. The goal was to propose statistically founded methods that account for graph data's specific features and behaviors.

We started by exploring new ways of comparing graphs. For that, we studied two comparisons of graphs based on heat diffusion, namely the *heat kernel distance* (HKD) and the *heat persistence distance* (HPD). The first one requires that the graphs are of the same sizes and aligned, while the second one is free of these assumptions. The novelty of our work comes from the multi-scale approach we developed. Instead of choosing somehow an informative diffusion time for the HKD and HPD, we introduced the concept of *distance process*, which is the family of all the distances over a continuous range of scaling parameters[3]. This concept defines the so-called HKD and HPD processes.

To analyze the statistical properties of these processes, we relied on the theory of empirical processes. We proved general results on continuous processes indexed by a real parameter. Simple conditions were derived to ensure that such processes verify a functional central limit theorem. Essentially, the processes are required to be uniformly bounded and Lipschitz-continuous. Moreover, we finely controlled the convergence rate to the Gaussian limit process. As a consequence, we showed that the HKD and HPD families are Donsker[4] and that their associated processes admit a Gaussian approximation with a rate that does not depend on the graph sizes.

From these theoretical results, we derived statistical applications. We constructed consistent confidence bands and consistent two-sample tests for samples of pairs of graphs. We refined these two-sample tests to be able to compare samples of graphs (and not pairs of graphs) by incorporating a pairing procedure.

The performances of our methods were illustrated by simulations on synthetic data sets. We showed that the two-sample tests could distinguish between Erdős-Rényi and SBM graphs, as well as between geometric graphs sampled on different domains. On Erdős-Rényi models with parameters depending on the sample size, the tests were still distinguishing between the different distributions, even when working close to the phase transition of the Neyman-Pearson test.

We confronted our two-sample tests with the problem of distribution shift detection in the context of neural network learning. Our approach consists in

[3]Here, the scaling parameters are diffusion times.

[4]Their associated processes verify a functional central limit theorem.

computing the activation graphs of a new data set and the ones of the test set and applying our two-sample tests to these samples of graphs. Experiments on the MNIST data set and on Ripsnet were carried out. Results showed that our methods were performing reasonably well in terms of power. Moreover, they were able to capture the neural network's robustness or sensitivity to distribution changes. As a result, our methods were less prone to sending false alarms to the neural network's user.

We believe that the introduction of the HKD and HPD processes has the potential to bring statistically founded and innovative ways to analyze data sets of graphs. In this work, we tackled the two-sample testing problem. However, it would be interesting to see if these methods could extend to classical learning tasks on graph data, *e.g.,* clustering, classification. Another line of research, motivated by numerous real-life applications, is the study of time series of graphs. It would be interesting to see how the ideas we developed in this thesis could transpose to the problem of change point detection, for example.

On the theoretical side, our approach considered asymptotic results when sample sizes go to infinity. Studying the interplay between graph sizes and sample sizes could be a first step toward developing non-asymptotic analyses of graph data sets.

Bibliography

[ABM05] Ron Aharoni, Eli Berger, and Roy Meshulam. Eigenvalues and homology of flag complexes and vector representations of graphs. *Geometric & Functional Analysis GAFA*, 15(3):555–566, 2005.

[AEK+17] Henry Adams, Tegan Emerson, Michael Kirby, Rachel Neville, Chris Peterson, Patrick Shipman, Sofya Chepushtanova, Eric Hanson, Francis Motta, and Lori Ziegelmeier. Persistence images: A stable vector representation of persistent homology. *Journal of Machine Learning Research*, 18, 2017.

[AHY+19] Matan Atzmon, Niv Haim, Lior Yariv, Ofer Israelov, Haggai Maron, and Yaron Lipman. Controlling neural level sets. *Advances in Neural Information Processing Systems*, 32, 2019.

[AM18] Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *Ieee Access*, 6:14410–14430, 2018.

[ARR+14] Waqar Ali, Tiago Rito, Gesine Reinert, Fengzhu Sun, and Charlotte M Deane. Alignment-free protein interaction network comparison. *Bioinformatics*, 30(17):i430–i437, 2014.

[AT16] James Atwood and Don Towsley. Diffusion-convolutional neural networks. *Advances in neural information processing systems*, 29, 2016.

[B+15] Peter Bubenik et al. Statistical topological data analysis using persistence landscapes. *J. Mach. Learn. Res.*, 16(1):77–102, 2015.

[BA95] J Martin Bland and Douglas G Altman. Multiple significance tests: the bonferroni method. *Bmj*, 310(6973):170, 1995.

[Ber41] Andrew C Berry. The accuracy of the gaussian approximation to the sum of independent variates. *Transactions of the american mathematical society*, 49(1):122–136, 1941.

[Bil13] Patrick Billingsley. *Convergence of probability measures*. John Wiley & Sons, 2013.

[BK05] Karsten M Borgwardt and Hans-Peter Kriegel. Shortest-path kernels on graphs. In *Fifth IEEE international conference on data mining (ICDM'05)*, pages 8–pp. IEEE, 2005.

[BLW86] Norman Biggs, E Keith Lloyd, and Robin J Wilson. *Graph Theory, 1736-1936*. Oxford University Press, 1986.

[BM06] Philippe Berthet and David M Mason. Revisiting two strong approximation results of dudley and philipp. In *High dimensional probability*, pages 155–172. Institute of Mathematical Statistics, 2006.

[BM18] Andrew B Bernard and Andreas Moxnes. Networks and trade. 2018.

[BO04] Albert-Laszlo Barabasi and Zoltan N Oltvai. Network biology: understanding the cell's functional organization. *Nature reviews genetics*, 5(2):101–113, 2004.

[BR18] Battista Biggio and Fabio Roli. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, 84:317–331, 2018.

[BY01] Yoav Benjamini and Daniel Yekutieli. The control of the false discovery rate in multiple testing under dependency. *Annals of statistics*, pages 1165–1188, 2001.

[BZSL14] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann Lecun. Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations (ICLR2014), CBLS, April 2014*, pages http–openreview, 2014.

[CCI+20] Mathieu Carrière, Frédéric Chazal, Yuichi Ike, Théo Lacombe, Martin Royer, and Yuhei Umeda. Perslay: a neural network layer for persistence diagrams and new graph topological signatures. In *International Conference on Artificial Intelligence and Statistics*, pages 2786–2796. PMLR, 2020.

[CDGT88] Dragos M Cvetkovic, Michael Doob, Ivan Gutman, and Aleksandar Torgašev. *Recent results in the theory of graph spectra*. Elsevier, 1988.

[CDSGO16] Frédéric Chazal, Vin De Silva, Marc Glisse, and Steve Oudot. *The structure and stability of persistence modules*. Springer, 2016.

[CDSO14] Frédéric Chazal, Vin De Silva, and Steve Oudot. Persistence stability for geometric complexes. *Geometriae Dedicata*, 173(1):193–214, 2014.

[CFL+14] Frédéric Chazal, Brittany Terese Fasy, Fabrizio Lecci, Alessandro Rinaldo, and Larry Wasserman. Stochastic convergence of persistence landscapes and silhouettes. In *Proceedings of the thirtieth annual symposium on Computational geometry*, pages 474–483, 2014.

[CG97] Fan RK Chung and Fan Chung Graham. *Spectral graph theory*, volume 92. American Mathematical Soc., 1997.

[CG16] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[CH14] Ronald R Coifman and Matthew J Hirn. Diffusion maps for changing data. *Applied and computational harmonic analysis*, 36(1):79–107, 2014.

[CL06] Ronald R Coifman and Stéphane Lafon. Diffusion maps. *Applied and computational harmonic analysis*, 21(1):5–30, 2006.

[COO15] Mathieu Carrière, Steve Y Oudot, and Maks Ovsjanikov. Stable topological signatures for points on 3d shapes. In *Computer graphics forum*, volume 34, pages 1–12. Wiley Online Library, 2015.

[CSEH09] David Cohen-Steiner, Herbert Edelsbrunner, and John Harer. Extending persistence using poincaré and lefschetz duality. *Foundations of Computational Mathematics*, 9(1):79–103, 2009.

[DH03] William E Donath and Alan J Hoffman. Lower bounds for the partitioning of graphs. In *Selected Papers Of Alan J Hoffman: With Commentary*, pages 437–442. World Scientific, 2003.

[DPR08] J-J Daudin, Franck Picard, and Stéphane Robin. A mixture model for random graphs. *Statistics and computing*, 18(2):173–183, 2008.

[dSHC+22] Thibault de Surrel, Felix Hensel, Mathieu Carrière, Théo Lacombe, Yuichi Ike, Hiroaki Kurihara, Marc Glisse, and Frederic Chazal. Ripsnet: a general architecture for fast and robust estimation of the persistent homology of point clouds. In *ICLR 2022 Workshop on Geometrical and Topological Representation Learning*, 2022.

[EH10] Herbert Edelsbrunner and John Harer. *Computational topology: an introduction*. American Mathematical Soc., 2010.

[ER+60] Paul Erdos, Alfréd Rényi, et al. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5(1):17–60, 1960.

[ESDS16] Frank Emmert-Streib, Matthias Dehmer, and Yongtang Shi. Fifty years of graph matching, network alignment and network comparison. *Information sciences*, 346:180–197, 2016.

[Ess42] Carl-Gustav Esseen. On the liapunov limit error in the theory of probability. *Ark. Mat. Astr. Fys.*, 28:1–19, 1942.

[FBA17] Kathyrn R Fair, Chris T Bauch, and Madhur Anand. Dynamics of the global wheat trade network and resilience to shocks. *Scientific reports*, 7(1):1–14, 2017.

[Fie73] Miroslav Fiedler. Algebraic connectivity of graphs. *Czechoslovak mathematical journal*, 23(2):298–305, 1973.

[Fie95] Miroslav Fiedler. An estimate for the nonstochastic eigenvalues of doubly stochastic matrices. *Linear algebra and its applications*, 214:133–143, 1995.

[FKL19] Farzad V Farahani, Waldemar Karwowski, and Nichole R Lighthall. Application of graph theory for identifying connectivity patterns in human brain networks: a systematic review. *frontiers in Neuroscience*, 13:585, 2019.

[FNC+17] Fazle E Faisal, Khalique Newaz, Julie L Chaney, Jun Li, Scott J Emrich, Patricia L Clark, and Tijana Milenković. Grafene: Graphlet-based alignment-free network approach integrates 3d structural and sequence (residue order) data to improve protein structural comparison. *Scientific reports*, 7(1):1–15, 2017.

[FZB16] Alex Fornito, Andrew Zalesky, and Edward Bullmore. *Fundamentals of brain network analysis*. Academic Press, 2016.

[GAC+18] Ralucca Gera, Lázaro Alonso, Brian Crawford, Jeffrey House, JA Mendez-Bermudez, Thomas Knuth, and Ryan Miller. Identifying network structure similarity using spectral graph theory. *Applied network science*, 3(1):1–15, 2018.

[GFW03] Thomas Gärtner, Peter Flach, and Stefan Wrobel. On graph kernels: Hardness results and efficient alternatives. In *Learning theory and kernel machines*, pages 129–143. Springer, 2003.

[GM10] Claudio Gallicchio and Alessio Micheli. Graph echo state networks. In *The 2010 international joint conference on neural networks (IJCNN)*, pages 1–8. IEEE, 2010.

[GMS05] Marco Gori, Gabriele Monfardini, and Franco Scarselli. A new model for learning in graph domains. In *Proceedings. 2005 IEEE international joint conference on neural networks*, volume 2, pages 729–734, 2005.

[GS17] Thomas Gebhart and Paul Schrater. Adversary detection in neural networks via persistent homology. *arXiv preprint arXiv:1711.10056*, 2017.

[GSH19] Thomas Gebhart, Paul Schrater, and Alan Hylton. Characterizing the shape of activation space in deep neural networks. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 1537–1542. IEEE, 2019.

[GSR+17] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.

[GTH+16] Madhavi K Ganapathiraju, Mohamed Thahir, Adam Handen, Saumendra N Sarkar, Robert A Sweet, Vishwajit L Nimgaonkar, Christine E Loscher, Eileen M Bauer, and Srilakshmi Chaparala. Schizophrenia interactome with 504 novel protein–protein interactions. *NPJ schizophrenia*, 2(1):1–10, 2016.

[HCG+08] Patric Hagmann, Leila Cammoun, Xavier Gigandet, Reto Meuli, Christopher J Honey, Van J Wedeen, and Olaf Sporns. Mapping the structural core of human cerebral cortex. *PLoS biology*, 6(7):e159, 2008.

[HG17] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *International Conference on Learning Representations*, 2017.

[HGJ13] David K Hammond, Yaniv Gur, and Chris R Johnson. Graph diffusion distance: A difference measure for weighted graphs based on the graph laplacian exponential kernel. In *2013 IEEE Global Conference on Signal and Information Processing*, pages 419–422. IEEE, 2013.

[HLL83] Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, 1983.

[HLM19] Dan Hendrycks, Kimin Lee, and Mantas Mazeika. Using pre-training can improve model robustness and uncertainty. In *International Conference on Machine Learning*, pages 2712–2721. PMLR, 2019.

[HRG14] Nan Hu, Raif M Rustamov, and Leonidas Guibas. Stable and informative spectral signatures for graph matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2305–2312, 2014.

[IFM09] Marios Iliofotou, Michalis Faloutsos, and Michael Mitzenmacher. Exploiting dynamicity in graph-based traffic analysis: Techniques and applications. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, pages 241–252, 2009.

[Kal19] Sara Kališnik. Tropical coordinates on the space of persistence barcodes. *Foundations of Computational Mathematics*, 19(1):101–129, 2019.

[KGB19] Alexander P Kartun-Giles and Ginestra Bianconi. Beyond the clustering coefficient: A topological analysis of node neighbourhoods in complex networks. *Chaos, Solitons & Fractals: X*, 1:100004, 2019.

[KM12] Nils Kriege and Petra Mutzel. Subgraph matching kernels for attributed graphs. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pages 291–298, 2012.

[Kos08] Michael R Kosorok. Introduction to empirical processes. *Introduction to Empirical Processes and Semiparametric Inference*, 2008.

[KP16] Risi Kondor and Horace Pan. The multiscale laplacian graph kernel. *Advances in neural information processing systems*, 29, 2016.

[KVF13] Danai Koutra, Joshua T Vogelstein, and Christos Faloutsos. Deltacon: A principled massive-graph similarity function. In *Proceedings of the 2013 SIAM International Conference on Data Mining*, pages 162–170. SIAM, 2013.

[Las21] Etienne Lasalle. Heat diffusion distance processes: a statistically founded method to analyze graph data sets. *arXiv preprint arXiv:2109.13213*, 2021.

[LBA12] Pierre Latouche, Etienne Birmele, and Christophe Ambroise. Variational bayesian inference and complexity control for stochastic block models. *Statistical Modelling*, 12(1):93–115, 2012.

[LCB10] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

[LGSL20] Daniel Lütgehetmann, Dejan Govc, Jason P Smith, and Ran Levi. Computing persistent homology of directed flag complexes. *Algorithms*, 13(1):19, 2020.

[LIC+21] Théo Lacombe, Yuichi Ike, Mathieu Carriere, Frédéric Chazal, Marc Glisse, and Yuhei Umeda. Topological uncertainty: Monitoring trained neural networks through persistence of activation graphs. In *IJCAI 2021-30th International Joint Conference on Artificial Intelligence*, 2021.

[Llo82] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

[LR06] Erich L Lehmann and Joseph P Romano. *Testing statistical hypotheses*. Springer Science & Business Media, 2006.

[LWS18] Zachary Lipton, Yu-Xiang Wang, and Alexander Smola. Detecting and correcting for label shift with black box predictors. In *International conference on machine learning*, pages 3122–3130. PMLR, 2018.

[MBGY14] Clément Maria, Jean-Daniel Boissonnat, Marc Glisse, and Mariette Yvinec. The gudhi library: Simplicial complexes and persistent homology. In *International congress on mathematical software*, pages 167–174. Springer, 2014.

[MV09] Pierre Mahé and Jean-Philippe Vert. Graph kernels based on tree patterns for molecules. *Machine learning*, 75(1):3–35, 2009.

[NAK16] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. Learning convolutional neural networks for graphs. In *International conference on machine learning*, pages 2014–2023. PMLR, 2016.

[NP33] Jerzy Neyman and Egon Sharpe Pearson. Ix. on the problem of the most efficient tests of statistical hypotheses. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 231(694-706):289–337, 1933.

[NS01] Krzysztof Nowicki and Tom AB Snijders. Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, 96(455):1077–1087, 2001.

[NYC15] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 427–436, 2015.

[Oud15] Steve Y Oudot. *Persistence theory: from quiver representations to data analysis*, volume 209. American Mathematical Society Providence, 2015.

[P+03] Mathew Penrose et al. *Random geometric graphs*, volume 5. Oxford university press, 2003.

[PBV07] Gergely Palla, Albert-László Barabási, and Tamás Vicsek. Quantifying social group evolution. *Nature*, 446(7136):664–667, 2007.

[PCJ04] Natasa Pržulj, Derek G Corneil, and Igor Jurisica. Modeling interactome: scale-free or geometric? *Bioinformatics*, 20(18):3508–3515, 2004.

[Pož07] Nataša Pržulj. Biological network comparison using graphlet degree distribution. *Bioinformatics*, 23(2):e177–e183, 2007.

[QTT+19] Talha Qaiser, Yee-Wah Tsang, Daiki Taniyama, Naoya Sakamoto, Kazuaki Nakane, David Epstein, and Nasir Rajpoot. Fast and accurate tumor segmentation of histology images using persistent homology and deep convolutional features. *Medical image analysis*, 55:1–14, 2019.

[RCY+11] Karl Rohe, Sourav Chatterjee, Bin Yu, et al. Spectral clustering and the high-dimensional stochastic blockmodel. *The Annals of Statistics*, 39(4):1878–1915, 2011.

[RGL19] Stephan Rabanser, Stephan Günnemann, and Zachary Lipton. Failing loudly: An empirical study of methods for detecting dataset shift. *Advances in Neural Information Processing Systems*, 32, 2019.

[RNS+17] Michael W Reimann, Max Nolte, Martina Scolamiero, Katharine Turner, Rodrigo Perin, Giuseppe Chindemi, Paweł Dłotko, Ran Levi, Kathryn Hess, and Henry Markram. Cliques of neurons bound into cavities provide a missing link between structure and function. *Frontiers in computational neuroscience*, page 48, 2017.

[RTB+18] Bastian Rieck, Matteo Togninalli, Christian Bock, Michael Moor, Max Horn, Thomas Gumbsch, and Karsten Borgwardt. Neural persistence: A complexity measure for deep neural networks using algebraic topology. In *International Conference on Learning Representations*, 2018.

[RVM+16] Maria A Rocca, Paola Valsasina, Alessandro Meani, Andrea Falini, Giancarlo Comi, and Massimo Filippi. Impaired functional integration in multiple sclerosis: a graph theory study. *Brain Structure and Function*, 221(1):115–131, 2016.

[RYB+20] Bastian Rieck, Tristan Yates, Christian Bock, Karsten Borgwardt, Guy Wolf, Nicholas Turk-Browne, and Smita Krishnaswamy. Uncovering the topology of time-varying fmri data using cubical persistence. *Advances in neural information processing systems*, 33:6900–6912, 2020.

[SDB16] Lee M Seversky, Shelby Davis, and Matthew Berger. On time-series topological data analysis: New data and opportunities. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 59–67, 2016.

[See68] Paul Seeger. A note on a method for the analysis of significances en masse. *Technometrics*, 10(3):586–593, 1968.

[SERG14] Sucheta Soundarajan, Tina Eliassi-Rad, and Brian Gallagher. A guide to selecting a network similarity method. In *Proceedings of the 2014 Siam international conference on data mining*, pages 1037–1045. SIAM, 2014.

[SGT+08] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

[Sim86] R John Simes. An improved bonferroni procedure for multiple tests of significance. *Biometrika*, 73(3):751–754, 1986.

[SK19] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

[SOG09] Jian Sun, Maks Ovsjanikov, and Leonidas Guibas. A concise and provably informative multi-scale signature based on heat diffusion. In *Computer graphics forum*, volume 28, pages 1383–1392. Wiley Online Library, 2009.

[SSVL+11] Nino Shervashidze, Pascal Schweitzer, Erik Jan Van Leeuwen, Kurt Mehlhorn, and Karsten M Borgwardt. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(9), 2011.

[ST96] Daniel A Spielman and Shang-Hua Teng. Spectral partitioning works: Planar graphs and finite element meshes. In *Proceedings of 37th conference on foundations of computer science*, pages 96–105. IEEE, 1996.

[SVP+09] Nino Shervashidze, SVN Vishwanathan, Tobias Petri, Kurt Mehlhorn, and Karsten Borgwardt. Efficient graphlet kernels for large graph comparison. In *Artificial intelligence and statistics*, pages 488–495. PMLR, 2009.

[TITP19] Mattia Tantardini, Francesca Ieva, Lucia Tajoli, and Carlo Piccardi. Comparing methods for comparing networks. *Scientific reports*, 9(1):1–19, 2019.

[TMB14] Katharine Turner, Sayan Mukherjee, and Doug M Boyer. Persistent homology transform for modeling shapes and surfaces. *Information and Inference: A Journal of the IMA*, 3(4):310–344, 2014.

[TMK+18] Anton Tsitsulin, Davide Mottin, Panagiotis Karras, Alexander Bronstein, and Emmanuel Müller. Netlsd: hearing the shape of a graph. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2347–2356, 2018.

[TT01] William Thomas Tutte and William Thomas Tutte. *Graph theory*, volume 21. Cambridge university press, 2001.

[Ume17] Yuhei Umeda. Time series classification via topological data analysis. *Information and Media Technologies*, 12:228–239, 2017.

[VDVW96] Aad W Van Der Vaart and Jon A Wellner. Weak convergence. In *Weak convergence and empirical processes*. Springer, 1996.

[VL07] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

[VLBB08] Ulrike Von Luxburg, Mikhail Belkin, and Olivier Bousquet. Consistency of spectral clustering. *The Annals of Statistics*, pages 555–586, 2008.

[VSKB10] S Vichy N Vishwanathan, Nicol N Schraudolph, Risi Kondor, and Karsten M Borgwardt. Graph kernels. *Journal of Machine Learning Research*, 11:1201–1242, 2010.

[WPC+20] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

[WZ08] Richard C Wilson and Ping Zhu. A study of graph spectra for comparing graphs and trees. *Pattern Recognition*, 41(9):2833–2841, 2008.

[YMDD+14] Ömer Nebil Yaveroğlu, Noël Malod-Dognin, Darren Davis, Zoran Levnajic, Vuk Janjic, Rasa Karapandza, Aleksandar Stojmirovic, and Nataša Pržulj. Revealing the hidden language of complex networks. *Scientific reports*, 4(1):1–9, 2014.

[ZAM08] Hugo Zanghi, Christophe Ambroise, and Vincent Miele. Fast online graph clustering via erdős–rényi mixture. *Pattern recognition*, 41(12):3592–3599, 2008.

[ZKR+17] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. *Advances in neural information processing systems*, 30, 2017.

[ZW19] Qi Zhao and Yusu Wang. Learning metrics for persistence-based summaries and applications for graph classification. *Advances in Neural Information Processing Systems*, 32, 2019.

[ZYCW20] Qi Zhao, Ze Ye, Chao Chen, and Yusu Wang. Persistence enhanced graph neural network. In *International Conference on Artificial Intelligence and Statistics*, pages 2896–2906. PMLR, 2020.