

## Representing graphs through data with learning and optimal transport

Présentée le 17 septembre 2021

Faculté des sciences et techniques de l'ingénieur  
Laboratoire de traitement des signaux 4  
Programme doctoral en informatique et communications

pour l'obtention du grade de Docteur ès Sciences

par

**Hermina PETRIC MARETIC**

Acceptée sur proposition du jury

Prof. N. Kiyavash, présidente du jury  
Prof. P. Frossard, directeur de thèse  
Dr E. Kokiopoulou, rapporteuse  
Prof. R. Gribonval, rapporteur  
Prof. D. Van De Ville, rapporteur



# Abstract

Graphs offer a simple yet meaningful representation of relationships between data. This representation is often used in machine learning algorithms in order to incorporate structural or geometric information about data. However, it can also be used in an inverted fashion: instead of modelling data through graphs, we model graphs through data distributions. In this thesis, we explore several applications of this new modelling framework.

Starting with the graph learning problem, we exploit the probabilistic model of data given through graphs to propose a multi-graph learning method for structured data mixtures. We explore various relations that data can have with the underlying graphs through the notion of graph filters. The structured data mixture assumes that the mapping of data to graphs is not known and we propose an algorithm to jointly cluster a set of data and learn a graph for each of the clusters. Experiments demonstrate promising performance in data clustering and multiple graph inference, and show desirable properties in terms of interpretability and proper handling of high dimensionality on synthetic and real data. The model has further been applied to fMRI data, where the method is used to successfully identify different functional brain networks and their activation times.

The probabilistic model of data defined through graphs can also be very meaningful even when no data is available. Thus, in the second part of this thesis, we use such models to represent each graph through the probabilistic distribution of data, which varies smoothly on the graph. Optimal transport allows for a comparison of two such distributions, which in turn gives a structurally meaningful measure for graph comparison. This novel measure is able to take into account the global behaviour of a graph, while most other comparison measures merely observe local changes independently.

We follow by using this distance to formulate a new graph alignment problem based on the optimal transport framework, whose objective is to estimate the permutation that minimizes the distance between two graphs. We propose an efficient stochastic algorithm based on Bayesian exploration to accommodate for the nonconvexity of the graph alignment problem. We demonstrate the performance of our novel framework on different tasks like graph alignment, graph classification and graph signal prediction, and we show that our method leads to significant improvement with respect to the state-of-art algorithms.

Furthermore, we cast a new formulation for the one-to-many graph alignment problem, allowing for comparison of graphs of different sizes. The resulting alignment problem is solved with stochastic gradient descent, where a novel Dykstra operator ensures that the solution is a one-to-many (soft) assignment matrix. Experiments on graph alignment and

## Abstract

---

graph classification problems show that our method for one-to-many alignment leads to meaningful improvements with respect to the state-of-the-art algorithms for each of these tasks.

Finally, we explore a family of probabilistic distributions for data based on graph filters. Distances defined through a graph filter give a high level of flexibility in choosing which graph properties we want to emphasize. In addition, in order to make the above graph alignment problem more scalable, we formulate an approximation to our filter Wasserstein graph distance that allows for the exploitation of faster algorithms, without grossly sacrificing the performance. We propose two algorithms, a simple one based on mirror gradient descent and another one built on its stochastic version, which offers a trade-off between speed and accuracy. Our experiments show the performance benefits of our novel stochastic algorithm, as well as the strong value of flexibility offered by filter-based distances.

In summary, we explore the idea of representing graphs through data distributions in several important scenarios. The works presented in this thesis finds applications in timely research problems, such as inference and comparison of biological networks and graph generation methods.

**Keywords:** Graph learning, network inference, optimal transport, graph signal processing, graph comparison, graph distance



# Résumé

Les graphes offrent une représentation simple de relations entre les données. Cette représentation est souvent utilisée dans les algorithmes d'apprentissage automatique afin d'incorporer des informations de structure ou de géométrie dans les données. Toutefois, elle peut également être utilisée de manière inversée : au lieu de modéliser les données par des graphes, nous proposons de modéliser les graphes par des distributions de données. Dans cette thèse, nous explorons plusieurs applications de cette nouvelle perspective.

En commençant par le problème d'apprentissage de graphes, nous exploitons le modèle probabiliste de données fourni par les graphes et proposons une méthode d'apprentissage multi-graphes pour les mélanges de données structurées. Nous explorons différentes relations que les données peuvent avoir avec les graphes sous-jacents à travers la notion de filtres définis sur des graphes. Le mélange de données structurées suppose que la relation entre données et graphes n'est pas connue et nous proposons un algorithme afin de conjointement regrouper un ensemble de données et apprendre un graphe pour chacun des groupes. Des expériences démontrent des performances prometteuses dans le partitionnement de données et l'inférence multi-graphes, et montrent des propriétés souhaitables en termes d'interprétabilité et de gestion de la dimensionnalité élevée sur des données synthétiques et réelles. Le modèle a en outre été appliqué aux données IRMf, où la méthode est utilisée afin d'identifier différents réseaux cérébraux fonctionnels et leurs temps d'activation.

Le modèle probabiliste de données défini par le biais de graphes peut également être très significatif même lorsqu'aucune donnée n'est disponible. Ainsi, dans la deuxième proposition de cette thèse, nous utilisons de tels modèles afin de représenter chaque graphe à travers la distribution probabiliste des données, qui varie de façon continue sur le graphe. Le transport optimal permet une comparaison de telles distributions, ce qui à son tour donne une mesure structurellement significative pour la comparaison de graphes. Cette nouvelle mesure est capable de prendre en compte le comportement global d'un graphe, tandis que la plupart des autres mesures de comparaison observent uniquement des changements locaux de manière indépendante.

Nous utilisons ensuite cette distance pour formuler un nouveau problème d'alignement de graphes basé sur le framework de transport optimal, dont l'objectif est d'estimer la permutation qui minimise la distance entre deux graphes. Nous proposons un algorithme stochastique efficace basé sur l'exploration bayésienne pour tenir compte de la non-convexité du problème d'alignement de graphes. Nous démontrons la performance de notre nouveau framework sur différentes tâches, telles que l'alignement de graphes, la classification de graphes et la prédic-

tion de signaux sur graphes, et nous montrons que notre méthode conduit à une amélioration significative par rapport aux algorithmes de l'état de l'art.

En outre, nous proposons une nouvelle formulation pour le problème d'alignement de graphes de différentes tailles. Le problème d'alignement qui en résulte est résolu par descente de gradient stochastique, où un nouvel opérateur Dykstra garantit que la solution est une matrice d'affectation souple. Des expériences sur les problèmes d'alignement et de classification de graphes montrent que notre méthode d'alignement conduit à des améliorations significatives par rapport à l'état de l'art pour chacune de ces tâches.

Enfin, nous explorons une famille de distributions probabilistes pour les données basée sur des filtres de graphes. De plus, afin de rendre le problème d'alignement de graphes ci-dessus plus scalable, nous formulons une approximation de notre coût, permettant l'exploitation d'algorithmes plus rapides, sans sacrifier significativement la performance. Nous proposons deux algorithmes : un algorithme simple basé sur la descente de gradient en miroir et un autre basé sur sa version stochastique, qui offre un compromis entre vitesse et précision. Nos expériences montrent les avantages en termes de performance de notre nouvel algorithme stochastique, ainsi que la grande valeur qu'apporte la flexibilité offerte par les distances basées sur les filtres.

En résumé, nous explorons les applications de la représentation de graphes par des distributions de données. Les travaux présentés dans cette thèse trouvent des applications dans les problèmes de recherche actuels, tels que l'inférence et la comparaison des réseaux biologiques et les méthodes de génération de graphes.

**Mots clés :** Apprentissage de graphes, inférence de réseau, transport optimal, traitement du signal sur graphe, comparaison de graphes, distance de graphes

# Contents

<b>Abstract (English/Français)</b>	<b>i</b>
<b>1 Introduction</b>	<b>1</b>
1.1 The perspective of data distributions in graph analysis . . . . .	1
1.2 Preliminaries . . . . .	3
1.2.1 Graph signal processing . . . . .	3
1.2.2 Optimal transport . . . . .	5
1.3 Thesis outline . . . . .	7
1.4 Summary of contributions . . . . .	8
<b>2 Learning with the Graph Laplacian mixture model</b>	<b>11</b>
2.1 Introduction . . . . .	11
2.2 Related work . . . . .	12
2.3 Graph Laplacian mixture model . . . . .	13
2.3.1 Multigraph signal representation . . . . .	14
2.3.2 Problem formulation . . . . .	16
2.4 Algorithm . . . . .	16
2.4.1 Expectation (E step) . . . . .	16
2.4.2 Maximisation (M step) . . . . .	18
2.5 Simulations . . . . .	20
2.5.1 Synthetic data . . . . .	21
2.5.2 Real data . . . . .	24
2.5.3 Inference of functional Brain Networks . . . . .	31
2.6 Conclusion . . . . .	34
<b>3 GOT: An Optimal Transport framework for Graph comparison</b>	<b>35</b>
3.1 Introduction . . . . .	35
3.2 Related work . . . . .	36
3.3 Graph Alignment with Optimal Transport . . . . .	37
3.3.1 Wasserstein distance between graphs . . . . .	37
3.3.2 Graph alignment . . . . .	39
3.4 GOT Algorithm . . . . .	40
3.4.1 Optimization . . . . .	41
3.4.2 Stochastic exploration . . . . .	42

## Contents

---

3.5	Experimental results . . . . .	43
3.5.1	Alignment of structured graphs . . . . .	43
3.5.2	Graph classification . . . . .	44
3.5.3	Graph signal transportation . . . . .	45
3.6	Conclusion . . . . .	46
<b>4</b>	<b>One-to-Many alignment based on the Wasserstein graph distance</b>	<b>49</b>
4.1	Introduction . . . . .	49
4.2	One-to-many assignment problem . . . . .	50
4.3	Optimization algorithm . . . . .	51
4.3.1	Relaxation . . . . .	51
4.3.2	Dijkstra operator . . . . .	52
4.3.3	Stochastic formulation . . . . .	54
4.4	Experiments . . . . .	55
4.4.1	Graph alignment and community detection . . . . .	56
4.4.2	Graph classification . . . . .	58
4.5	Conclusion . . . . .	61
<b>5</b>	<b>fGOT: filter Graph distances using Optimal Transport</b>	<b>63</b>
5.1	Introduction . . . . .	63
5.2	Filter Graph Alignment with Optimal Transport . . . . .	64
5.2.1	Filter graph distance . . . . .	64
5.2.2	Scalable alignment approximation . . . . .	66
5.3	FGOT algorithm . . . . .	68
5.4	Experimental results . . . . .	71
5.4.1	Speed comparison on a graph alignment task . . . . .	71
5.4.2	Community detection in structured graphs . . . . .	73
5.5	Conclusion . . . . .	75
<b>6</b>	<b>Conclusion</b>	<b>77</b>
6.1	Summary . . . . .	77
6.2	Future directions . . . . .	78
<b>A</b>	<b>Appendix: Inference of Brain Networks</b>	<b>81</b>
A.1	Introduction . . . . .	81
A.2	Materials and Methods . . . . .	82
A.2.1	Data and Preprocessing . . . . .	82
A.2.2	Experimental details . . . . .	82
A.2.3	Comparison of learned functional graphs to brain structure . . . . .	83
A.3	Results . . . . .	84
A.3.1	Estimated network time-courses are consistent with the timing of the task paradigms . . . . .	84

A.3.2	GLMM captures spatial networks corresponding to each task consistent with their known established neurophysiological descriptions . . . . .	84
A.3.3	Learned graphs bear similarity to the underlying brain structure and their relation reveals behaviorally-relevant organization . . . . .	86
A.4	General findings . . . . .	88
	<b>Bibliography</b>	<b>89</b>
	<b>Curriculum Vitae</b>	<b>99</b>



# 1 Introduction

## 1.1 The perspective of data distributions in graph analysis

With the rapid development of digitisation in various domains, the volume of data increases very rapidly, and a large amount of it takes the form of structured data. Often, this structure in data can be represented through graphs, with nodes representing entities and edges modelling connections between them. Examples of such structured data include online social networks, as well as transportation networks with traffic information on top of them. The underlying graphs provide valuable context and can be crucial in performing successful analysis of such data. Interesting examples of such analysis can be found in the field of graph signal processing [81], where data comes in the form of signals on an irregular domain, and tools are developed to extend classical signal processing to such applications. From here onwards, we use the term *data* and *signal* interchangeably.

However, the structure of data is not always readily available in the form of a graph. For instance, brain activities can be measured independently in different brain regions, but the connections between those regions are not accessible without further analysis. Furthermore, even when the graph is available, the analysis of structured data can be very challenging. In order to use the additional information provided by the graph structure, certain assumptions on signal behaviour need to be made. Probably the most common assumption is that of smoothness, stating that the signal values vary slowly across the edges of the graph. It has been used extensively, from applications in machine learning, such as label propagation [127] or robust principal component analysis [101], to signal processing, such as graph signal denoising or image super-resolution [92]. In the recent years, a more general model of graph filtered signals has gained on popularity, and has been extensively developed with the rise of graph signal processing research. Both of these representations, though mostly in an implicit manner, impose a probability distribution on data through the graph. While they have proven to be very useful in processing and analysing data, they can also be used to analyse graphs themselves.

In this work, we explore the idea of inverting this notion and modelling graphs through

distributions of signals. By doing so, we present a graph as a distribution of all data which can appear on it. The fact that this representation can actually be written in terms of a probabilistic distribution is not only intuitively interesting, but also enables the utilisation of bayesian statistics, such as graphical models, as well as the use of tools of optimal transport theory, allowing us to find a meaningful distance and a transportation map between graphs. In particular, in this thesis we study two very important problems for structured data and graph analysis: graph learning and graph comparison.

**Graph learning** methods aim at recovering a graph from available data observations [25]. They provide very valuable insights in the behaviour of data, by properly modelling their connections and creating a structure between them. Even more, they can be subsequently used in order to perform further analysis on the data itself. However, most graph learning works consider only scenarios in which all data can be described with one graph. This is often not the case, as data is often available in a form of a mixture and needs to be separated into meaningful groups. One such example are signals of brain activity, where fMRI data contains mixed measurements when the brain goes through various processes described by different functional networks. We will focus on the problem of learning graphs from mixed signals, where the probabilistic representation of graphs through signals will allow us to create a generative model for such data.

**Graph comparison** is a vital tool in graph analysis, providing the basis for working with several graphs, processing data on them and generalising findings to different graph structures. It also has vast direct applications, for example in biology for protein classification, and in drug discovery for molecule comparisons. However, it remains a challenging problem and there is no universally accepted solution in the research community. The reason for that is twofold: 1) The nodes of the two graphs are generally not aligned in the best possible way. This prevents a direct quantitative analysis of graph matrices and requires a generally very costly procedure of recovering a good alignment between them. 2) Even when the nodes are aligned, it is not entirely clear how two graphs can be compared in a meaningful way. Namely, direct comparison of graph adjacency matrices will only capture local differences between two graphs independently. Edges in the graph, however, are not all equally important, and have different structural values to the global shape of the graph. We will again resort to graph representation through their signal distributions, taking their global structure into account through its influence on signal behaviour. This time, the probabilistic representation allows us to use tools of optimal transport, a natural choice in distribution comparison. The resulting Wasserstein distance between graphs therefore has an interesting interpretation of comparing graphs based on the support they provide for data living on them. Equipped with a structurally meaningful distance, we consider the challenging problem of graph alignment and provide algorithms for alignment with both the exact Wasserstein distance, and a faster approximation. Finally, the distribution of filtered signals allows a consideration of structural differences with different priorities on signal behaviour, extending the definition to a more flexible framework.



## 1.2 Preliminaries

### 1.2.1 Graph signal processing

In this section, we introduce some basic notions in graph signal processing that we will be using throughout the thesis. Let  $\mathcal{G} = (\mathcal{V}, \mathcal{E}, W)$  be an undirected, weighted graph with a set of  $N$  vertices  $\mathcal{V}$ , edges  $\mathcal{E}$  and a weighted adjacency matrix  $W$ . The value  $W_{ij}$  is equal to 0 if there is no edge between  $i$  and  $j$ , and denotes the weight of that edge otherwise. The degree of a vertex  $i \in V$ , denoted by  $d(i)$ , is the sum of weights of all the edges incident to  $i$  in the graph  $\mathcal{G}$ . The degree matrix  $D \in \mathbb{R}^{N \times N}$  is then defined as:

$$D_{i,j} = \begin{cases} d(i) & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases} \quad (1.1)$$

Based on  $W$  and  $D$ , the combinatorial graph Laplacian  $L$  is defined as

$$L = D - W. \quad (1.2)$$

As the graph Laplacian is a real symmetric matrix, it has a complete set of orthonormal eigenvectors  $U = \{U_0, U_1, \dots, U_{N-1}\}$  with a corresponding set of non-negative eigenvalues  $\lambda_i$ . Furthermore, zero appears as an eigenvalue with a multiplicity equal to the number of connected components of the graph. The spectrum of the combinatorial graph Laplacian thus satisfies

$$\sigma(L) = \{0 = \lambda_0 \leq \lambda_1 \leq \dots \leq \lambda_{N-1}\} \quad (1.3)$$

We then define a signal on a graph as a function  $x : \mathcal{V} \rightarrow \mathbb{R}$ , where  $x_n$  denotes the value of a signal on a vertex  $n$ . We can now observe these graph signals in the graph spectral domain (as opposed to the vertex domain). Similarly to the classical Fourier transform, we can define the graph Fourier transform  $\hat{x}$  of a signal  $x$  at frequency  $\lambda_l$  as the expansion:

$$\hat{x}(\lambda_l) = \langle x, U_l \rangle = \sum_{n=1}^N x_n U_{ln}, \quad (1.4)$$

and the inverse graph Fourier transform as

$$x_n = \sum_{l=0}^{N-1} \hat{x}(\lambda_l) U_{ln}. \quad (1.5)$$

Here, the normalised Laplacian eigenvectors form a Fourier basis and it is not difficult to see that the corresponding eigenvalues carry a notion of frequency [102].

A graph signal is considered smooth if most of its energy is concentrated in the low frequencies (first eigenvalues of the underlying graph), which can be measured with a quadratic form of

## Chapter 1. Introduction

---

the graph Laplacian:

$$x^T L x = \frac{1}{2} \sum_{i,j} W_{ij} (x_i - x_j)^2. \quad (1.6)$$

Indeed, it is clear from Equation (1.6) that the signal difference will get more penalised for two vertices linked by a strong edge. It might be less apparent that there is also a strong spectral interpretation. Namely, using the graph Fourier decomposition, we can see that the above relation

$$x^T L x = x^T U \Lambda U^T x \quad (1.7)$$

penalises signals according to their frequency support. Therefore, eigenvectors corresponding to low eigenvalues will not be penalised very much, whereas high frequency components of the signal will be penalised with the corresponding high eigenvalues. This again leads to the notion of smoothness.

We further define filtering in graph signal processing [81] as

$$\hat{x}^f(\lambda_l) = \hat{x}(\lambda_l) \hat{g}(\lambda_l), \quad (1.8)$$

where  $\hat{x}^f$  is the outcome of filtering a graph signal  $x$  with a graph filter  $g$  defined in the spectral domain. Using the inverse graph Fourier transform,  $x^f$  can be written as

$$x_n^f = \sum_{l=0}^{N-1} \hat{x}(\lambda_l) \hat{g}(\lambda_l) U_{ln}. \quad (1.9)$$

A graph filter can also be represented in a matrix form

$$g(L) = U \hat{G} U^T \quad (1.10)$$

with a diagonal matrix

$$\hat{G}_{i,j} = \begin{cases} \hat{g}(\lambda_i) & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases} \quad (1.11)$$

The filtered graph signal can now be written simply in the matrix form as  $x^f = g(L)x$ .

This can be used to generate graph signals with specific properties. Given a graph filter  $g(L)$  and a white noise signal  $w \sim \mathcal{N}(0, I)$ , in this work we will consider a kernel graph signal as

$$x = \mu + g(L)w. \quad (1.12)$$

This kernel signal follows a Gaussian distribution:

$$x = \mu + g(L)w \sim \mathcal{N}(\mu, g(L)Ig(L)^T) = \mathcal{N}(\mu, g^2(L)), \quad (1.13)$$

As the special case of smooth signals is of large importance and brings specific challenges, we will explore both the general case, and this special case, in detail throughout the thesis.

Notice that a smooth signal can be seen as a special case of the kernel signal, with the filter equal  $g(L) = \sqrt{L^\dagger}$ ,  $L^\dagger$  being the pseudo-inverse of the graph Laplacian matrix. Namely, under such an assumption, the signal  $x$  minimising (1.6) is equivalent to the maximiser of the log likelihood of  $x$  given through Eq. (1.13). Furthermore, it is equivalent to modelling  $x$  through a latent variable  $h \sim \mathcal{N}(0, \Lambda^\dagger)$ , such that  $x = \mu + Uh$ . As  $\Lambda^\dagger$  represents the pseudo-inverse of the eigenvalue matrix, this model assumes that signal support is inversely proportional to frequency, describing smooth signals. This gives us a direct relationship between smooth signals and the graph Laplacian matrix  $L$ :

$$x = \mu + Uh \sim \mathcal{N}(\mu, U\Lambda^\dagger U^T) = \mathcal{N}(\mu, L^\dagger). \quad (1.14)$$

Note that  $x$  can also be seen as a special degenerate Gaussian Markov Random Field (GMRF) [93], since  $L_{ij} = 0 \Leftrightarrow (i, j) \notin E$ . Namely,  $L$  is the precision matrix (inverse covariance matrix) of the distribution defined in (1.14). It is also usually assumed to be sparse, and the particular properties include that  $L$  has at least one zero eigenvalue, as well as a special structure ensuring:

$$L_{i,j} = L_{j,i} \leq 0, \forall i \neq j, \quad (1.15)$$

$$\sum_{j=1}^N L_{i,j} = 0, \forall i. \quad (1.16)$$

When these conditions are satisfied, we write  $L \in \mathcal{L}$ , where  $\mathcal{L}$  is a set of valid Laplacian matrices [102].

### 1.2.2 Optimal transport

Here we present a quick overview of optimal transport and the tools that we will be using throughout the thesis. Optimal transport (OT) is a theory that enables comparison of probability distributions and defines a distance between them (known as the Wasserstein distance). It was introduced by Monge [77], and reformulated in a more tractable way by Kantorovich [58]. It has been a topic of great interest both theoretically and practically [113], and has recently been largely revisited with new applications in image processing, data analysis, and machine learning [88].

Intuitively, we can imagine two probability distributions as two piles of sand. The piles are taller wherever observations are more likely. Assuming distributions have equal amounts of

## Chapter 1. Introduction

---

sand in them both (probability distributions sum up to 1), we can imagine many ways for transforming one distribution in to the other one. The idea of optimal transport is to find the best way to do this transformation, such that the cumulative cost of transporting all sand grains is minimal.

Formally, let  $(\nu_1, \nu_2)$  be the set of two arbitrary probability measures on two spaces  $(\mathcal{X}, \mathcal{Y})$ , and  $f : \mathcal{X} \rightarrow \mathcal{Y}$  a measurable mapping. A push forward measure of  $\nu_1$ , denoted by  $f_{\#}\nu_1$ , is defined on  $\mathcal{Y}$  with

$$f_{\#}\nu_1(B) = \nu_1(f^{-1}(B)) = \nu_1(\{x \in \mathcal{X} : f(x) \in B\}) \quad (1.17)$$

for every measurable set  $B \subset \mathcal{Y}$ . In other words, a push forward operator  $\#$  defines a measure on  $\mathcal{Y}$  through two things: a predefined measure  $\nu_1$  on  $\mathcal{X}$  and a mapping  $f$  connecting  $\mathcal{X}$  and  $\mathcal{Y}$ . For each set  $B \subset \mathcal{Y}$ , its push forward measure is equal to  $\nu_1(A)$ , where  $A \subset \mathcal{X}$  is a set containing all elements  $x \in \mathcal{X}$  such that  $f(x) \in B$ .

In terms of our example with two piles of sand, we can see a set  $A$  as an area of the ground  $\mathcal{X}$  under the sand. In that case, the measure  $\nu_1(A)$  denotes the total amount of sand in area  $A$ . With  $B$  denoting an area of the ground  $\mathcal{Y}$ , the push forward measure  $f_{\#}\nu_1$  is defined such that the total amount of sand in area  $B$ ,  $f_{\#}\nu_1(B)$ , corresponds to the amount of sand that function  $f$  transferred from  $\mathcal{X}$  to  $B$ ,  $\nu_1(\{x \in \mathcal{X} : f(x) \in B\})$ .

Let  $c : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}^+$  be a cost function, with  $c(x, y)$  denoting the cost of transporting a unit of mass from  $x \in \mathcal{X}$  to  $y \in \mathcal{Y}$ . The Monge formulation [77] of the optimal transport problem then reads

$$\underset{T}{\text{minimize}} \int_{\mathcal{X}} c(x, T(x)) d\nu_1(x), \quad \text{s.t.} \quad T_{\#}\nu_1 = \nu_2, \quad (1.18)$$

where  $T : \mathcal{X} \rightarrow \mathcal{Y}$  is a  $\nu_1$  - measurable map and  $\#$  is the pushforward operator. Intuitively,  $T$  can be seen as function that preserves positivity and total mass, i.e., moving an entire probability mass on  $\mathcal{X}$  to an entire probability on  $\mathcal{Y}$ . Equation (1.18) can be seen as the minimal cost needed to transport one probability measure to another with respect to the cost  $c : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}^+$ . This minimal cost of transporting one measure to another is called the Wasserstein distance between two measures (also referred to as Kantorovich-Monge-Rubinstein distance).

Going back to the sand pile example,  $c(x, y)$  is the cost of transferring a sand grain from  $x$  to  $y$ , and  $T$  is the optimal mapping that defines where each sand grain should be transferred. The Wasserstein distance presents the minimal cumulative cost with which it is possible to transfer one pile of sand into the other.

An example of a Wasserstein distance that has a closed expression is that of Gaussian distributions. Namely, taking the  $l_2$  distance as the cost  $c$ , the Wasserstein distance between Gaussian distributions has an explicit expression in terms of the mean vectors and covariance matrices.

Let  $\mathcal{X} = \mathcal{Y} = \mathbb{R}^N$ , and let  $\nu_1$  and  $\nu_2$  be two Gaussian distributions on  $\mathbb{R}^N$ ,  $\nu_1 = \mathcal{N}(m, \Sigma_1)$  and  $\nu_2 = \mathcal{N}(n, \Sigma_2)$ . With  $c(x, y) = \|x - y\|^2$  the Wasserstein distance  $\mathcal{W}_2^2(\nu_1, \nu_2)$  can be written as [106]:

$$\mathcal{W}_2^2(\nu_1, \nu_2) = \inf_{T_{\#}\nu_1 = \nu_2} \int_{\mathbb{R}^N} \|x - T(x)\|^2 d\nu_1(x) \quad (1.19)$$

$$= \|m - n\|^2 + \text{Tr}(\Sigma_1 + \Sigma_2) - 2\text{Tr}\left(\sqrt{\Sigma_1^{\frac{1}{2}}\Sigma_2\Sigma_1^{\frac{1}{2}}}\right) \quad (1.20)$$

and the optimal mapping  $T$  that takes  $\nu_1$  to  $\nu_2$  is

$$T(x) = n + \Sigma_1^{\frac{1}{2}}\left(\Sigma_1^{\frac{1}{2}}\Sigma_2\Sigma_1^{\frac{1}{2}}\right)^{\frac{1}{2}}\Sigma_1^{-\frac{1}{2}}(x - m). \quad (1.21)$$

The Wasserstein distance gives a very meaningful comparison of two distributions and has seen a recent surge in applications [88]. We will use the Wasserstein distance between Gaussian distributions in order to define a structurally meaningful distance between graphs.

### 1.3 Thesis outline

The goal of this thesis is to explore the applications of the new framework of modelling graph through data distributions. The thesis is organised as follows:

In Chapter 2 we explore the problem of multiple graph learning from mixed signals. We propose a generative model for mixed graph signals, formulated for a generic class of signals following a graph filter distribution. We solve the problem with the expectation maximisation algorithm, and explore two specific graph learning methods in details, based on two signal models of interest. Finally, we apply the method to the inference of multiple functional brain networks, where we use fMRI data to simultaneously separate time frames based on the functional brain network that they activate, and infer these brain networks and their structure. Apart from recovering meaningful clusters and recognising brain networks which are known in the literature, we get interesting insights into the correlation of distinct functional brain graphs with the brain structure.

Chapter 3 further explores the representation of graphs through signal distributions, in a scenario in which the signals are not necessarily available. We use the probabilistic model of smooth graph signals and build on the notions of optimal transport, creating a framework that permits to compare graphs through their respective signal distributions. This optimal transport framework for graph comparison allows to define a structurally meaningful distance between graphs, but also to adapt signals from one graph to another. We follow by formulating a new graph alignment problem using this optimal transport distance. We propose to solve this problem with a novel stochastic algorithm based on Bayesian exploration, in order to accommodate for the non-convexity of the graph alignment problem. Finally, experiments on

different tasks, such as graph alignment, graph classification and graph signal prediction, show that our method successfully captures the global structure of graphs and provides considerable improvements with respect to the state-of-art algorithms.

In Chapter 4 we extend the framework introduced in the previous chapter to graphs of different sizes and formulate graph alignment as a one-to-many assignment problem, allowing us to easily interpret an alignment of different-sized graphs. We propose a stochastic algorithm using a novel Dykstra operator to implicitly impose the one-to-many alignment structure. We demonstrate the performance of our novel framework on graph alignment and graph classification, and show that our method leads to significant improvements with respect to the state-of-the-art algorithms for each of these tasks.

In Chapter 5 we introduce the filter graph distance, that brings more flexibility into the graph optimal transport framework. It results in graph distances that can take different spectral information into account. We next formulate an approximation to the filter graph distance cost, removing some of the largest computational challenges. We propose a simple and fast solution using mirror gradient descent (MGD), as well as a more accurate solution using a novel stochastic algorithm based on MGD, offering a tradeoff between speed and accuracy. Our experiments confirm the computational gains of our approximated formulation, the efficiency of our proposed stochastic algorithm, as well as the benefits of flexibility introduced with filter graph distances.

Finally, Chapter 6 offers concluding remarks for the dissertation and discusses possibilities for future work in this area.

### 1.4 Summary of contributions

The main contributions of this thesis are summarised below. In particular, we propose:

- A multi-graph learning method for structured signal mixtures, which simultaneously clusters a set of signals and learns a graph for each of the clusters. The method has an interesting application in neuroscience for multiple functional brain networks inference
- A framework based on optimal transport for graph comparison, which shapes a structurally meaningful graph distance, as well as a signal transportation function between graphs
- A novel formulation for the graph alignment problem defined through the graph optimal transport distance and a stochastic algorithm based on Bayesian exploration, which leads to efficient solutions to this non-convex problem
- An extension of the optimal transport framework to graphs of different sizes with a one-to-many assignment, and an effective stochastic algorithm using a Dykstra operator that computes a (soft) one-to-many assignment

- A new filter graph distance, which is a flexible distance that enables prioritising specific spectral properties in graph comparison, together with an approximation to the filter optimal transport cost function, and a novel stochastic mirror descent algorithm for efficient graph alignment with filter graph distance





## 2 Learning with the Graph Laplacian mixture model

### 2.1 Introduction

Relationships between data can often be well described with a graph structure. Although many datasets, including social and traffic networks, come with a pre-existing graph that helps in interpreting them, there is still a large number of datasets (e.g., brain activity information) where a graph is not readily available. Many graph learning techniques have been proposed in the past years [25] [72] to help in analysing such datasets. More specifically, a lot of interest in the field of graph learning is currently focused on designing graph learning methods that can take into account prior information on the graph structure [29], or different relationships between data and the underlying graph [98]. However, most of these works only consider simple data, where all datapoints follow the same model defined with only one graph. While there are still many topics of interest in those settings, we argue that natural data often comes in more complicated forms. In fact, such data opens an entire field of unsupervised learning methods. A natural example of such a dataset can be found in brain fMRI data, where signals usually measure the brain activity through different brain processes. Each of these processes can be explained with a different brain functional network, with regions of interest as shared network nodes. However, it is not clear which network is activated at what time, causing the need to separate signals corresponding to different networks.

In this chapter, we precisely consider data that naturally forms several clusters, where signals in a cluster live on the same graph<sup>1</sup>. This allows analysis of more complex datasets where simple graph learning methods would suffer from intertwined data and thus lose the ability to capture a meaningful graph structure. In particular, we study the problem of multiple graph inference from a general group of signals that are an unknown combination of data with different structures. Namely, we propose a novel generative model for data represented as a

---

<sup>1</sup>Parts of this chapter have been published in:  
H. P. Maretic and P. Frossard. Graph laplacian mixture model. *IEEE Transactions on Signal and Information Processing over Networks*, 6:261–270, 2020  
H. Petric Maretic, M. El Gheche, and P. Frossard. Graph heat mixture model learning. In *Asilomar Conference on Signals, Systems, and Computers*, pages 1003–1007. IEEE, 2018

mixture of signals naturally living on a collection of different graphs. As it is often the case with actual data, the separation of these signals into clusters is assumed to be unknown. We thus propose an algorithm that jointly clusters the signals and infers multiple graph structures, one for each of the clusters. Our method assumes a general signal model, and offers a framework for multiple graph inference that can be directly used with a number of state-of-the-art network inference algorithms, harnessing their particular benefits. Numerical experiments show promising performance in terms of both data clustering and multiple graph inference, while coping well with the dimensionality of the problem. Simulations show that our method is effective in finding interesting patterns in a traffic network of New York, while it demonstrates high interpretability on a weather dataset, as well as MNIST data. Furthermore, the model permits to successfully cluster brain fMRI signals into groups determined by well-known functional brain networks, providing understanding on the timing of network activations, as well as on the similarity of functional and structural connectivity.

As we will deal with clustering in large dimensionalities in these settings, it is worth noting that inherently high dimensional clustering problems often suffer from the curse of dimensionality [13] and poor interpretability. While imposing that data lives on a graph implicitly reduces the dimensionality of the problem, graphs also offer a natural representation for connections between data. Therefore, they provide interpretability both in terms of direct inspection of graph structure, as well as the ability to further deploy various data analysis algorithms.

## 2.2 Related work

In the literature, the graph learning problem has been first considered as sparse precision matrix inference. Data is modelled as a multivariate Gaussian distribution, whose inverse covariance matrix reveals direct pairwise connections between nodes [22] [36]. Based on these methods, several models have been proposed to infer Gaussian mixture models with sparse precision matrices [21] [67] [48]. All these works actually focus on inferring Gaussian Markov Random Fields (GMRFs) [93], and do not constrain values in the precision matrix in any special way. Therefore, the resulting graphs can have both positive and negative weights, as well as self-loops, which can be difficult to interpret in practice. On the other hand, graph representation constrained to a valid Laplacian matrix circumvents this problem, while opening the door to numerous data analysis methods [102]. For these reasons, an increasing amount of work has recently been focusing on inferring (generalised) graph Laplacians. Among the first researchers to focus on graph Laplacian inference, Dong et al. [24] adopt a graph signal processing perspective to enforce data smoothness on the inferred graph. The proposed model results in assumptions similar to those in GMRFs, but with added constraints that ensure that the graph is given by a valid Laplacian matrix. Kalofolias [55] uses a similar framework and proposes a computationally more efficient solution by inferring a weight matrix, which can eventually be easily transformed into a Laplacian. An even more efficient, approximate solution has been proposed for large scale graph learning [57], scaling to graphs with millions of nodes. Other recent works in this vein include inference of graph shift operators, with the

assumption that data is the result of a graph diffusion process. A popular approach to this problem consists in exploiting the fact that the eigenvectors of the graph will be shared with those of any graph filter. Therefore, they can be estimated from the data sample covariance matrix, and the optimisation can be done only over the eigenvalues [98] [82]. Dictionary based methods try to model signal heterogeneity by taking into account sparse combinations or dictionary atoms. In [108], signals are represented as linear combinations of atoms from a heat kernel dictionary. As those are still bound to be smooth, the authors in [86] model signals as sparse linear combinations of atoms in a predefined polynomial graph dictionary. Several methods exploit possible additional priors in the graph inference problem, such as a bandlimited representation of signals [96], or properties on graph topologies [68] [83]. Online graph learning from streaming signals has been considered in [114] and [100]. All aforementioned methods assume the whole set of signals can be well explained with one single graph model. Naturally, this is unlikely to be the case in many applications, where different signals might be generated in different ways or exist as a combination of distinct causes.

Finally, there have been some works focused on signals in one dataset that naturally live on different graphs. Kalofolias et al. [56] infer the structure of a time varying graph, effectively capturing multiple graphs in different periods of time. A similar approach based on sparseness of variation between graphs in different periods has been proposed by Yamada et al. [119]. Sardellitti et al. [95] propose a method for multi-layer graph inference for prediction of dynamic graph signals. Segarra et al. [99] infer multiple networks under the assumption of signal stationarity. Recent approaches include multi-graph data which would benefit from being explained with one graph. Such methods infer one global graph which tries to retain important properties from available graphs and signals [11] [76]. Multi-graph inference works inspired by graphical lasso [36] include a method by Kao et al. [59] that promotes differences in inferred graphs, minimizing the correlation between each graph and the sample covariance of signals that do not live on it. Recent work of Gan et al. [37] imposes that the sparsity pattern on all inferred graphs should be similar through a Bayesian prior. However, differently from our work, all of these methods assume that signal clusters are given a priori, i.e., it is clear which signals correspond to which graph. The joint network inference then becomes a problem of imposing similarities on different graph structures, rather than decoupling the signals into groups and learning a graph to explain each of the groups, which is the focus of our method. To the best of our knowledge, our work presents the first framework to deal with inference of multiple graph Laplacians from a mixture of not *a priori* clustered signals.

## 2.3 Graph Laplacian mixture model

We propose a probabilistic model for a set of graph signals with distinguishable subsets, where the behaviour of signals in each of the subsets (groups) is well explained with a graph. A toy example of our data model is given in Figure 2.1.

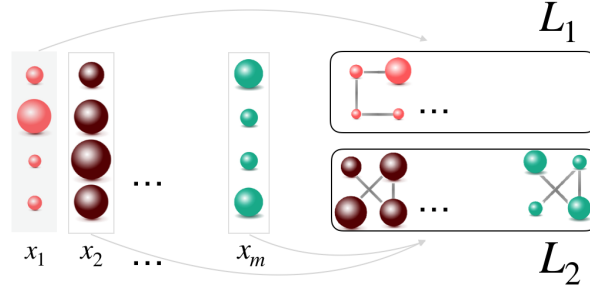


Figure 2.1 – Illustration of our model. Data is given as signals  $x_1, \dots, x_m$ . In this example, each signal is smooth on its respective graph. Our model groups signals naturally belonging to different clusters and learns a graph for each of the clusters.

Our goal is to distinguish these groups of signals, and eventually infer a graph that will model the structure of signals in each cluster. The clusters of signals are unknown and the graph structures largely influence behaviours inside clusters. We thus argue that identifying the clusters and learning the associated graphs are intertwined problems. Therefore, we propose a generative model that jointly explains both signals and clusters, under an assumption of signals following the graph kernel model, for each cluster.

Specifically, as the graph kernel model we consider a distribution of filtered graph signals, as described in (1.13). For a graph filter  $g(L)$  and a white noise signal  $w \sim \mathcal{N}(0, I)$ , we consider a distribution of signals  $x$ , such that  $x = \mu + g(L)w$ . The graph kernel model is then given as:

$$x = \mu + g(L)w \sim \mathcal{N}(\mu, g(L)Ig(L)^T) = \mathcal{N}(\mu, g^2(L)), \quad (2.1)$$

Furthermore, along with the general model for filtered graph signals, we will consider in detail a special case of the graph kernel model focusing on signals which are smooth on  $L$ . As shown in (1.14), smooth signals can be represented through a Gaussian distribution with a filter  $g(L) = \sqrt{L^\dagger}$ , resulting in:

$$x \sim \mathcal{N}(\mu, L^\dagger). \quad (2.2)$$

### 2.3.1 Multigraph signal representation

The graphical representation for our model is given in Figure 2.2. Let us assume that there are  $K$  undirected, weighted graphs  $G^k = (V, E^k, W^k)$  with a set of  $N$  shared vertices  $V$ . Each graph has a specific set of edges  $E^k$  and a weighted adjacency matrix  $W^k$ . From each of these weight matrices  $W^k$ , we can define a graph Laplacian matrix  $L_k$ , as in (1.2).

We further assume there are  $K$  clusters, and each of the  $M$  observed signals  $x_m \in \mathbb{R}^N$  on the nodes  $V$ , belongs to exactly one of the clusters. Cluster participation is modelled through

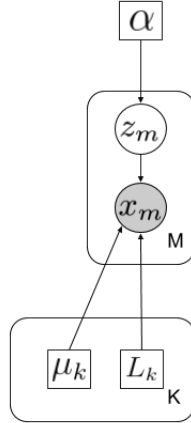


Figure 2.2 – Plate notation for our generative model. Filled in circles are observed variables, small empty squares are unknown parameters, and the empty circles represent latent variables. Large plates indicate repeated variables.

a binary latent variable  $\mathbf{z}_m \in \mathbb{R}^K$ , with  $z_{m,j} = \delta_{j,k}, \forall j$ , if signal  $x_m$  belongs to cluster  $k$ . The mixing coefficients  $\boldsymbol{\alpha} \in \mathbb{R}^K$  model the size of each cluster, and define a prior distribution on variables  $\mathbf{z}_m$ , with  $p(z_{m,j} = \delta_{j,k}, \forall j) = p(z_{m,k} = 1) = \alpha_k, \forall m$ .

Finally, we model data in each cluster  $k$  with a mean  $\mu_k$  and a graph Laplacian  $L_k$ , assuming associated signals will be close to  $\mu_k$  and follow a kernel model with a filter  $g$  on graph  $L_k$ , as defined in Eq. (2.1):

$$p(x_m | z_{m,k} = 1) = p(x_m | \mu_k, g_k(L_k)) = \mathcal{N}(\mu_k, g_k^2(L_k)) \quad (2.3)$$

Marginalising over latent variables  $\mathbf{z}$ , we have:

$$p(x_m) = \sum_{\mathbf{z}_m} p(\mathbf{z}_m) p(x_m | \mathbf{z}_m) \quad (2.4)$$

$$= \sum_{k=1}^K p(z_{m,k} = 1) p(x_m | z_{m,k} = 1) \quad (2.5)$$

$$= \sum_{k=1}^K \alpha_k \mathcal{N}(\mu_k, g_k^2(L_k)), \quad (2.6)$$

$$\text{with } L_k \in \mathcal{L}, \forall k \quad (2.7)$$

$$\sum_{k=1}^K \alpha_k = 1, \quad (2.8)$$

$$\alpha_k > 0, \forall k \quad (2.9)$$

This fully describes our generative model, with (2.7) ensuring that all  $L_k$ s are valid Laplacians, while (2.8) and (2.9) ensure that  $\boldsymbol{\alpha}$  defines a valid probability measure.

### 2.3.2 Problem formulation

Given a set of  $M$   $N$ -dimensional graph signals  $\mathbf{X} \in \mathbb{R}^{N \times M}$  with some intrinsic grouping into  $K$  clusters associated to it, we look at the maximum a posteriori (MAP) estimate for our parameters: mixing coefficients  $\boldsymbol{\alpha} = \alpha_1 \dots \alpha_K$ , means  $\boldsymbol{\mu} = \mu_1 \dots \mu_K$  and graph Laplacians  $\mathbf{L} = L_1 \dots L_K$ . Namely, assuming the data has been sampled independently from the distribution in (2.6), and allowing for a prior on the graph structure, we want to maximise over the a posteriori distribution of our model:

$$\arg \max_{\boldsymbol{\alpha}, \boldsymbol{\mu}, \mathbf{L}} \ln p(\boldsymbol{\alpha}, \boldsymbol{\mu}, \mathbf{L} | \mathbf{X}) \quad (2.10)$$

$$\propto \arg \max_{\boldsymbol{\alpha}, \boldsymbol{\mu}, \mathbf{L}} \ln p(\mathbf{X} | \boldsymbol{\alpha}, \boldsymbol{\mu}, \mathbf{L}) p(\mathbf{L}) \quad (2.11)$$

$$= \arg \max_{\boldsymbol{\alpha}, \boldsymbol{\mu}, \mathbf{L}} \ln \prod_{m=1}^M p(x_m | \boldsymbol{\alpha}, \boldsymbol{\mu}, \mathbf{L}) p(\mathbf{L}) \quad (2.12)$$

$$= \arg \max_{\boldsymbol{\alpha}, \boldsymbol{\mu}, \mathbf{L}} \ln \prod_{m=1}^M \sum_{k=1}^K \alpha_k \mathcal{N}(x_m | \mu_k, g_k^2(L_k)) p(L_k) \quad (2.13)$$

$$= \arg \max_{\boldsymbol{\alpha}, \boldsymbol{\mu}, \mathbf{L}} \sum_{m=1}^M \ln \sum_{k=1}^K \alpha_k \mathcal{N}(x_m | \mu_k, g_k^2(L_k)) p(L_k), \quad (2.14)$$

which does not have a closed form solution. We will thus estimate the parameters using an expectation maximisation algorithm (EM), as explained below.

Note that we present our algorithm for the case of general kernel signals as long as that is possible. However, we will see that smooth signals pose specific challenges, and will for that reason be treated separately throughout the chapter.

## 2.4 Algorithm

We propose an expectation maximisation algorithm that alternates between optimising for expected cluster participations  $\boldsymbol{\gamma} := \mathbb{E}(\mathbf{z})$  in the expectation step, and signal means  $\boldsymbol{\mu}$ , class proportions  $\boldsymbol{\alpha}$  and graph topologies  $\mathbf{L}$  in the maximisation step. Therefore, the joint clustering and multi-graph learning problem iterates over two steps: the first one estimates the correct clustering, and the second one describes the clusters by inferring cluster means and proportions, as well as the graphs describing them. Precisely, we first initialise  $\boldsymbol{\alpha}, \boldsymbol{\mu}$  and  $\mathbf{L}$  randomly, noting that we ensure  $\mathbf{L}$  is a set of set of random valid Laplacian matrices, guaranteeing that it truly describes graph structures. The alternating steps follow, as described below.

### 2.4.1 Expectation (E step)

Let us define  $\boldsymbol{\gamma} \in \mathbb{R}^{M \times K}$  as a matrix of posterior probabilities, with  $\gamma_{m,k}$  modelling the probability that the signal  $x_m$  belongs to the group  $k$ . Note that, at the same time, this is the expected

value of the latent indicator variable  $z_{m,k}$ , with  $\boldsymbol{\gamma}_m = \mathbb{E}(\mathbf{z}_m)$ .

$$\gamma_{m,k} = p(z_{m,k} = 1 | x_m, \mu_k, L_k) \quad (2.15)$$

$$= \frac{p(z_{m,k} = 1) p(x_m | z_{m,k} = 1, \mu_k, L_k)}{\sum_{l=1}^K p(z_{m,l} = 1) p(x_m | z_{m,l} = 1, \mu_l, L_l)} \quad (2.16)$$

$$= \frac{\alpha_k \mathcal{N}(x_m | \mu_k, g_k^2(L_k))}{\sum_{l=1}^K \alpha_l \mathcal{N}(x_m | \mu_l, g_l^2(L_l))} \quad (2.17)$$

In practice,  $\boldsymbol{\gamma}$  can take different forms, depending on the choice of kernels  $g_k, k \in \{1, \dots, K\}$ . More precisely, the posterior probabilities  $\boldsymbol{\gamma}$  can be directly computed in the case of most arbitrarily chosen graph kernels using Equation (2.17).

However, the above formulation for  $\boldsymbol{\gamma}$  cannot be computed directly when signals follow the smooth model. Namely, it is well known that a graph Laplacian has at least one eigenvalue that is zero, corresponding to the eigenvector  $\mathbf{1}$ . This makes the distribution in (2.17) degenerate when signals follow (2.2) or a similar filter model. At the same time, from the signal processing point of view, the corresponding eigenvector is completely smooth on all possible graphs, and is therefore non-informative. The disintegration theorem [54] guarantees that we can restrict our problem to the  $N - 1$  dimensional subspace spanned by all remaining eigenvectors. We thus proceed by projecting signals to this subspace, where we can then compute  $\boldsymbol{\gamma}$  and retrieve the probabilities we want to model.

Furthermore, if a graph has disconnected components, it will have as many zero eigenvalues as the number of components in the graph. Even if the final graph is connected, there is no guarantee that the algorithm does not return a disconnected graph in one of the iterations of the optimisation problem. It is easy to see how this can pose large numerical problems, both in each graph separately as their eigenvalues approach zero, but also in terms of comparing the probabilities that these graphs define when trying to infer  $\gamma_{m,k}$  from (2.17). To avoid this problem, we add a small positive regularising constant  $\epsilon$  to every eigenvalue corresponding to the  $N - 1$  non-trivial eigenvectors of the graph Laplacian. Note that  $\epsilon$  serves only for numerical regularization and should be kept as small as possible. Finally, with  $y_{m,k} := x_m - \mu_k$ , the computation of probabilities  $\boldsymbol{\gamma}$  sums up as follows:

$$L_k = U_k \Lambda_k U_k^T \quad (2.18)$$

$$\tilde{\Lambda}_k = (\Lambda_k)_{2:N, 2:N} + \epsilon \mathbf{I}_{N-1} \quad (2.19)$$

$$\tilde{U}_k = (U_k)_{1:N, 2:N} \quad (2.20)$$

$$\tilde{y}_{m,k} = \tilde{U}_k^T y_{m,k} \quad (2.21)$$

$$\gamma_{m,k} = \frac{\alpha_k \mathcal{N}(\tilde{y}_{m,k} | 0, g^2(\tilde{\Lambda}_k))}{\sum_{l=1}^K \alpha_l \mathcal{N}(\tilde{y}_{m,l} | 0, g^2(\tilde{\Lambda}_l))} \quad (2.22)$$

### 2.4.2 Maximisation (M step)

Having estimated probabilities  $\gamma_{m,k}$  in the E-step, we can now maximise the expected posterior distribution given all observed signals, to infer  $\alpha, \mu$  and  $L$ :

$$\arg\max_{\alpha, \mu, L} \sum_Z p(Z|X, \mu, L) \ln p(X, Z|\alpha, \mu, L) p(L) \quad (2.23)$$

$$= \arg\max_{\alpha, \mu, L} \sum_Z p(Z|X, \mu, L) \quad (2.24)$$

$$\ln \prod_{m=1}^M p(x_m, z_m|\alpha, \mu, L) p(L) \quad (2.25)$$

$$= \arg\max_{\alpha, \mu, L} \sum_{m=1}^M \sum_{k=1}^K p(z_{m,k} = 1|x_m, \mu_k, L_k) \quad (2.26)$$

$$\ln p(x_m, z_m|\alpha_k, \mu_k, L_k) p(L_k) \quad (2.27)$$

$$= \arg\max_{\alpha, \mu, L} \sum_{m=1}^M \sum_{k=1}^K \gamma_{m,k} \ln (\alpha_k \mathcal{N}(x_m|\mu_k, g_k^2(L_k)) p(L_k)) \quad (2.28)$$

$$= \arg\max_{\alpha, \mu, L} \sum_{m=1}^M \sum_{k=1}^K \gamma_{m,k} (\ln \alpha_k + \quad (2.29)$$

$$+ \ln \mathcal{N}(x_m|\mu_k, g_k^2(L_k)) + \ln p(L_k)) \quad (2.30)$$

Equation (2.30) is concave over  $\mu$  and  $\alpha$ , and offers closed form solutions for both:

$$\mu_k = \frac{\sum_{m=1}^M \gamma_{m,k} x_m}{\sum_{m=1}^M \gamma_{m,k}} \quad (2.31)$$

$$\alpha_k = \frac{\sum_{m=1}^M \gamma_{m,k}}{N}. \quad (2.32)$$

In order to maximise over  $L$ , we substitute  $x_m$  with variables  $y_{m,k} := x_m - \mu_k$ . Now we can formulate a problem of multiple graph inference:

$$\arg\max_L \sum_{k=1}^K \sum_{m=1}^M \gamma_{m,k} (\ln \mathcal{N}(y_{m,k}|0, g_k^2(L_k)) + \ln p(L_k)). \quad (2.33)$$

It is clear that these are  $K$  independent optimisation problems, and we can maximise each one separately:

$$\arg\max_{L_k \in \mathcal{L}} \sum_{m=1}^M \gamma_{m,k} (\ln \mathcal{N}(y_{m,k}|0, g_k^2(L_k)) + \ln p(L_k)). \quad (2.34)$$

The generative model gives a general framework and a graph inference algorithm of choice can be used to infer  $L$  in 2.34, depending on the nature of data and the appropriate graph filter. Here, we explore two different methods, one representing the special case of smooth graph signals, while the other investigates one of the most common graph filters, the heat kernel.



### Graph inference from smooth signals

Similarly to the graph inference problem explored in [24], using the kernel for smooth signals  $g(L_k) = L_k^{\dagger/2}$  as discussed in (2.2), each of the problems (2.34) can be efficiently approximated with:

$$\operatorname{argmin}_{L_k \in \mathcal{L}} \sum_{m=1}^M \gamma_{m,k} y_{m,k}^T L_k y_{m,k} + f(L_k), \quad (2.35)$$

with the graph Laplacian prior encoded in  $f(L_k)$ .

To efficiently update  $\mathbf{L}$  through (2.35), it is crucial to choose a good prior on the graph Laplacians. Namely, we want to maximise signal smoothness on the graph, while controlling graph sparsity. Due to its computational efficiency, we will use the algorithm from Kalofolias [55] with the small addition of cluster probabilities  $\gamma_{m,k}$ , as in (2.35). The graph update step (2.34) thus becomes:

$$\operatorname{argmin}_{L_k \in \mathcal{L}} \sum_{m=1}^M \gamma_{m,k} y_{m,k}^T L_k y_{m,k} - \quad (2.36)$$

$$\beta_1 \operatorname{tr}(\mathbf{1}^T \log(\operatorname{diag}(L_k))) + \beta_2 \|L_k\|_{F,\text{off}}^2 \quad (2.37)$$

$$\mathcal{L} = \{L \in \mathbb{R}^{N \times N} : L_{i,j} = L_{j,i} \leq 0, \forall i \neq j, \mathbf{L}\mathbf{1} = \mathbf{0}\} \quad (2.38)$$

in which  $\operatorname{diag}(L_k)$  is a vector with the diagonal values (node degrees) from  $L_k$ , and  $\|L_k\|_{F,\text{off}}^2$  is the Frobenius norm of the off-diagonal values in  $L_k$ . Notice that  $\|L_k\|_{F,\text{off}}^2 = \|W_k\|_F^2$ , where  $W_k$  is the weight matrix of the same graph. Compared to the formulation from (2.35), the function  $f(L_k) = -\beta_1 \operatorname{tr}(\mathbf{1}^T \log(\operatorname{diag}(L_k))) + \beta_2 \|L_k\|_{F,\text{off}}^2$  consists of two parts, such that increasing  $\beta_1$  strengthens graph connectivity, while a large  $\beta_2$  promotes density. With small  $\beta_1$  and  $\beta_2$ , the solution will be very sparse by the nature of the problem. The parameters can be selected with a simple parameter sweep, or automatically as proposed in [57]. Note that our  $f(L_k)$  does not stem from a probabilistic prior, and is used here as an effective heuristic to approximate problem (2.35). As before, the constraint that  $L_k \in \mathcal{L}$  ensures that  $L_k$  is a valid Laplacian matrix. This problem is solved with an iterative algorithm [55] and the computational complexity of this algorithm is  $\mathcal{O}(MN^2)$  once for preprocessing, and then  $\mathcal{O}(N^2)$  per iteration.

### Graph inference from kernel signals

Assuming the graph filter model in (2.34) is known *a priori*, an efficient graph inference method can easily be formulated. As an example, we investigate a widely used filter model, the graph heat kernel,

$$g_k(L_k) = e^{-\tau L_k}. \quad (2.39)$$

Following (2.34), it is clear that data is connected to the graph Laplacian matrix through its

covariance  $g_k^2(L_k) = e^{-2\tau L_k}$ . To infer a graph Laplacian matrices  $L_k$ , we first notice that the estimation of a covariance matrix without graph priors  $p(L_k)$  can be written in closed form as:

$$\Sigma_k := \frac{\sum_m \gamma_m(k) y_{m,k} y_{m,k}^T}{\sum_m \gamma_m(k)} \quad (2.40)$$

In order to efficiently infer graph structures, the information of data probability might however not be sufficient. Namely, without very large amounts of data, the sample covariance matrices  $\{\Sigma_k\}$  are usually noisy (if not low rank), and it can be difficult to recover the exact structure of the graph matrix. We thus formulate a problem that aims at finding a valid Laplacian matrix that would give a covariance matrix similar to the sample covariance one, while at the same time imposing a graph sparsity constraint. Namely, we can estimate the weight matrix  $W_k$  as

$$\arg \min_{W_k} \|\Sigma_k - e^{-2\tau L_k}\|_F^2 + \beta \|W_k\|_1, \quad (2.41)$$

$$\text{s.t.} \begin{cases} L_k = D_k - W_k \\ \mathcal{W} = \{W_k \in \mathbb{R}_+^{N \times N} : W_k = W_k^T, \text{diag}(W_k) = 0\}. \end{cases} \quad (2.42)$$

Similarly, we can avoid the computation of the matrix exponential by searching for heat kernel parameters which are close to the sample log-covariance matrix. Namely, we solve

$$\arg \min_{W_k \in \mathcal{W}} \|\log \Sigma_k + 2\tau L_k\|_F^2 + \beta \|W_k\|_1 \quad (2.43)$$

with the same constraints. It results in a convex problem that can be solved efficiently with FISTA [12] [85].

As our work offers a generic model for multiple graph inference, a very wide range of graph signal models, and most graph learning methods can be used directly with our framework to perform multiple graph inference. Moreover, while performing graph inference already has a clear advantage with respect to covariance estimation in terms of both interpretability and accuracy [22], this effect can be enhanced using graph-related prior information to simplify the problem. Namely, any graph specific additional information, such as a mask of possible or forbidden edges, or a desired level of sparsity, can easily be incorporated in most graph learning methods, including the ones presented in this chapter. Such prior information would render the implicit dimensionality reduction of the problem even stronger, and the search space smaller, thus leading to even better results.

## 2.5 Simulations

In this section, we evaluate our algorithm on both synthetic and real data, followed by a short study on the inference of multiple functional brain networks. To the best of our knowledge, there are no other methods tackling the problem of jointly clustering graph signals and learning multiple graph Laplacians. Thus, we compare our algorithm (namely, GLMM for the smooth

signal model, GHMM for the heat kernel) to the estimation of a Gaussian mixture model (GMM) and respectively to a signal clustering by a simple K-means algorithm, followed by graph learning [55] in each inferred cluster separately (denoted as K-means + GL). In order to compare the inferred graphs, we threshold all graphs obtained with any graph learning method to remove very small values. Furthermore, the Gaussian mixture model does not contain any sparsity regularization, and will thus not naturally provide sparse precision matrices. We therefore obtain graphs from fitted GMMs by choosing only the largest absolute values in the inferred precision matrices, in such a way that the number of edges is the same as the one in the original graph.

### 2.5.1 Synthetic data

#### Smooth signals on a graph mixture

We consider randomly generated connected graphs of size  $N = 15$  following an Erdos-Renyi model [31] with  $p = 0.7$ . Each graph  $L_k$  is given a randomly generated 15-dimensional mean signal  $\mu_k$  that lives on its vertices,  $\mu_k \sim \mathcal{N}(0, \sigma^2 I)$ , with  $\sigma = 0.5$ . We fix the total number of signals to 150 and consider a case with 2 and 3 classes, given by the probability vectors  $\alpha^1 = \{0.5, 0.5\}$ ,  $\alpha^2 = \{0.2, 0.8\}$  and  $\alpha^3 = \{0.33, 0.33, 0.33\}$ . For each signal  $x_m$  we randomly generate  $z_m$  through probabilities  $\alpha$ . Then  $x_m$  is randomly generated through a distribution  $x_m \sim \mathcal{N}(\mu_k, L_k^\dagger)$ , if  $z_{m,k} = 1$ , which gives us the full synthetic dataset for each experiment. The whole experiment has been repeated 100 times, each time with different graphs, means and randomly generated data.

Table 2.1 – Synthetic data clustering results for graphs with 15 nodes. The error is presented in terms of signal clustering NMSE percentage (values between 0 and 100, smaller is better). The first row shows results for 2 balanced clusters with  $\alpha = [0.5, 0.5]$ , the second for 3 balanced clusters  $\alpha = [0.33, 0.33, 0.33]$ , while the last row presents the error for clustering unbalanced clusters with  $\alpha = [0.2, 0.8]$ .

$\alpha$	GLMM	GMM	K-Means
[0.5, 0.5]	<b>2.49</b>	22.6	7.3
[0.33, 0.33, 0.33]	<b>5.98</b>	27.7	11.86
[0.2, 0.8]	<b>2.84</b>	23.02	21.03

We examine the performance of GLMM, GMM and K-means + GL in recovering the original clusters of signals (given by  $z_m$ ) and the corresponding graph topologies. The hyperparameters have been fixed with a grid search before running the experiment. Note that, to avoid numerical issues and render the comparison more fair, we train the GMM in  $n - 1$  dimensions, ignoring the direction of the first eigenvector, as the corresponding eigenvalue is known to be zero (see Section 1.2). We present the error in terms of class identification NMSE ( $\frac{1}{2M} \|\mathbf{z} - \mathbf{y}\|_F^2$ , presented in %) in Table 2.1. The performance of graph inference for each of the methods is presented in Table 2.2 in terms of mean edge recovery F measure.

Table 2.2 – Synthetic data graph learning results for graphs with 15 nodes. The performance is evaluated in terms of edge recovery F-measure (values between 0 and 1, larger is better). The first row shows results for balanced clusters with  $\alpha = [0.5, 0.5]$ , the second for 3 balanced clusters  $\alpha = [0.33, 0.33, 0.33]$ , while the last two rows present the F-measure for unbalanced clusters with  $\alpha = 0.2$ , and  $\alpha = 0.8$ , respectively.

$\alpha$	GLMM	GMM	K-Means + GL
[0.5, 0.5]	<b>0.81</b>	0.59	0.77
[0.33, 0.33, 0.33]	<b>0.71</b>	0.44	0.64
0.2	<b>0.66</b>	0.39	0.52
0.8	<b>0.86</b>	0.7	0.81

As expected, we can see that all methods are affected by unbalanced clusters, and give significantly poorer results in terms of clustering performance, as well as edge recovery for the graph in less represented cluster. As the graph in the more represented cluster (with  $\alpha = 0.8$ ) will always be inferred from the bigger set of relevant signals, all three methods outperform their own F measure score compared to the case of balanced clusters.

### Signals on a mixture of graph heat kernels

We next investigate the performance of our algorithm for signals generated from a graph filter that describes a graph heat kernel.

We generate two connected Erdos-Renyi graphs  $L_1$  and  $L_2$  of size  $N = 20$ , with edge probability  $p = 0.7$ . The means for each cluster are randomly drawn from  $\mu_k \sim \mathcal{N}(0, 0.1I)$ , and the membership probabilities for each cluster are fixed to  $\alpha_k = 0.5$ . The signals are random instances of Gaussian distributions  $x_m \sim \mathcal{N}(\mu_k, e^{-2\tau L_k})$ , where the mean  $\mu_k$ ,  $\tau$  and the graph Laplacian  $L_k$  drive the heat diffusion processes. The number of signals is fixed to  $M = 200$  and each experiment has been repeated 100 times.

We test the performance of all four inference algorithms as a function of the heat parameter  $\tau$  that varies between 0.1 and 0.8, as shown in Figure 2.3. For very small values of  $\tau$ , all algorithms have difficulties in recovering the structure as the sample covariance matrix is close to identity, and does not contain a lot of graph related information. For large values of  $\tau$ , the signals that we observe are very smooth, with very small variations. For this reason, the simple smoothness assumption used in GLMM is too weak to successfully separate signals, while the heat kernel model achieves very good performance. This shows that adapting the signal model to data can be very important, and emphasizes the importance of the flexibility that our framework offers in incorporating various graph learning methods.

Note that no prior information on  $\tau$  was given to any of the algorithms (as it is just a scaling factor in the heat model). Therefore, the only influence that  $\tau$  has on results comes from data structure.

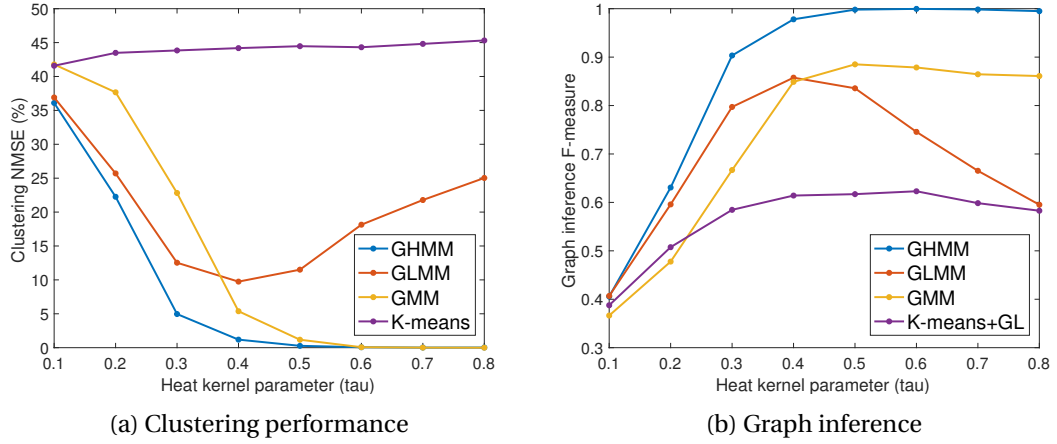


Figure 2.3 – Performance of clustering and graph inference with respect to the heat parameter  $\tau$ .

### Gaussian mixture model signals

Next, we test the performance of our algorithm on data generated from random Gaussian mixture models. The objective is to investigate if the added structure in GLMM creates implicit dimensionality reduction that can help in inference of high dimensional GMMs. We generate random covariance matrices of size  $N = 20$  from the Wishart distribution  $\Sigma_k \sim \frac{1}{n} W(I_n, n)$ , and means as  $\mu_k \sim \frac{1}{2} N(0, I_n)$ ,  $\forall k$ . With  $\alpha = [0.5, 0.5]$ , we sample the Gaussian mixture model 200 times. We then add white noise to each sampled signal  $w \sim \mathcal{N}(0, \sigma^2 I)$ , with  $\sigma$  varying from 0 to 0.6, as in Figure 2.4. Each point is averaged over 100 experiments. As expected, all methods are affected by increasing noise. However, it is interesting to see that, while the error of GMM and GLMM is very similar in the no-noise scenario, the GMM error increases drastically already for small noise ( $\sigma = 0.1$ ). This is probably due to the high sensitivity to noise that Gaussian mixture models exhibit in high dimensions, while GLMM's implicit dimensionality reduction makes it more robust, even in cases when data is not sampled from a graph mixture.

To test this assumption further, we explore changing the dimensionality of the Gaussian mixtures. We vary  $N$  from 15 to 50 as shown in Figure 2.4, keeping the number of signals fixed ( $M = 200$ ). Note that with larger  $N$  the task becomes easier, as means are still generated in the same way ( $\mu_k \sim \frac{1}{2} N(0, I_n)$ ), so in higher dimensions cluster means are more likely to be further away. That explains a significant decrease in error for K-means. However, GMM still suffers from the curse of dimensionality, while GLMM manages to achieve lower errors for higher dimensions, implying that an implicit dimensionality reduction has occurred.

Various experiments on synthetic data show that our method achieves promising results in comparison with other inspected methods, introducing higher flexibility in data modelling and reducing the dimensionality of the problem.

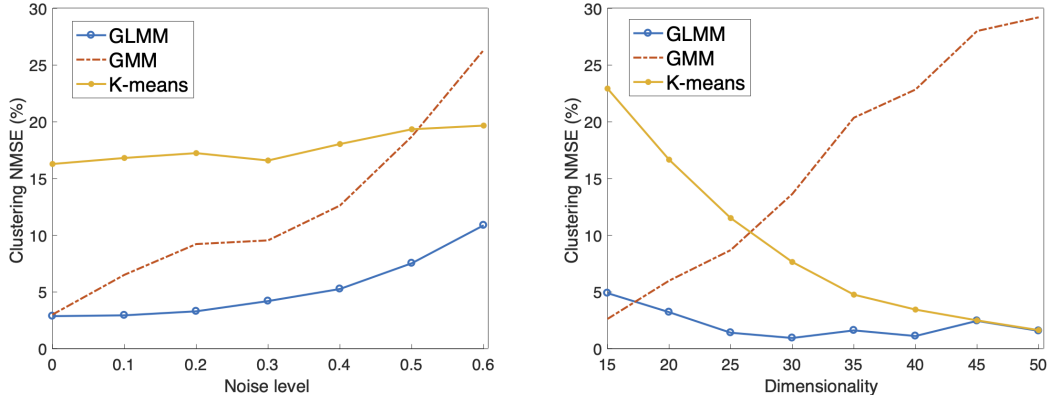


Figure 2.4 – Clustering performance of data generated through random Gaussian mixture models. The performance is shown as a function of noise level ( $\sigma$ ) and dimensionality ( $N$ ), respectively.

### Signals with noisy labels

Cluster labels are often not entirely unknown, and the available labels may be unreliable due to some errors or noise. In this final experiment on synthetic data, we examine this problem and evaluate our algorithm given noisy cluster labels. For the purpose of this experiment, we slightly modify our model, allowing for mixing coefficients to come in the form of group priors  $\alpha$ . Namely, each signal  $x_m$  belongs to a predefined group  $S_i$ , and  $p(z_{m,k} = 1) = \alpha_{S_i,k}$ , if  $x_m \in S_i$ . In this experiment, we use the same settings as in experiment 2.5.1, with 3 balanced clusters. In addition, each signal has a noisy cluster label available. However, using the labels directly with a hard assignment might not be optimal due to the noise. We therefore group all the signals with the same noisy label, and vary the group prior initialisation: from 1 corresponding to the hard assignment of noisy labels, to 0.33 corresponding to a random prior.

Figure 2.5 shows results in terms of clustering and graph inference F-measure for different percentages of noise among available labels (from 0 to 100). Note that, apart from the scenario in which the prior is fixed as 1 (hard assignment of labels), all other priors manage to adapt reasonably well, with 0.8 and 0.9 performing optimally for small noise levels, and 0.33 giving the best results when noise is larger than 50%. In addition, even small group priors benefit from scenarios with no noise, probably due to the fact that mere separation of signals into (almost) correct groups is enough to help the algorithm eventually converge to better clustering.

### 2.5.2 Real data

We further evaluate our algorithm on real data. In applications where the real network is not known, we evaluate the performance by inspecting the quality of recovered clusters. We further compare the inferred graphs to a constructed ground-truth graph, and use visual

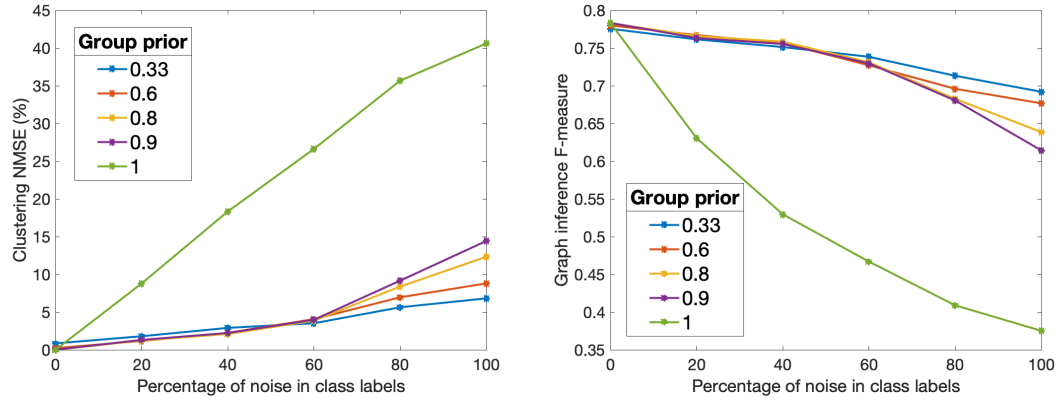


Figure 2.5 – Performance of clustering and graph inference with regard to the amount of noise present in data labels.

inspection to assess graph quality.

### Molene weather dataset

The Molene weather dataset [40] provides measurements of temperature and wind strength in 28 stations across Bretagne, France. There are in total 744 measurements of each type, in each of the 28 stations. Note that we refer to signals as measurements, while measure represents a whole class of temperature or wind strength signals. We preprocess the data by subtracting the mean of each signal and by normalising both measures, to ensure the data is in the same range. We compare the results of GLMM, GMM, K-means + GL and GHMM in terms of clustering accuracy, graph inference, as well as model generalisability.

We first look at clustering accuracy and graph inference performance. We randomly select a subset of 300 preprocessed measurements describing temperature and 300 describing wind speed to create a dataset for evaluation. The whole experiment has been repeated 100 times, each time with a different randomly selected subset of measurements. Clustering performance is presented in Table 2.3 in terms of NMSE (%).

Table 2.3 – Clustering of 600 randomly selected signals from Molene weather dataset, of which 300 are temperature measurements and 300 represent wind speed. C stands for the clustering error in terms of NMSE percentage (values between 0 and 100, smaller is better), while G stands for graph inference error in terms of edge recovery F measure (values between 0 and 1, larger is better).

	GLMM	GMM	K-means + GL	GHMM
C	<b>7.66</b>	24.51	21.61	18.68
G	<b>0.82</b>	0.49	0.73	0.52

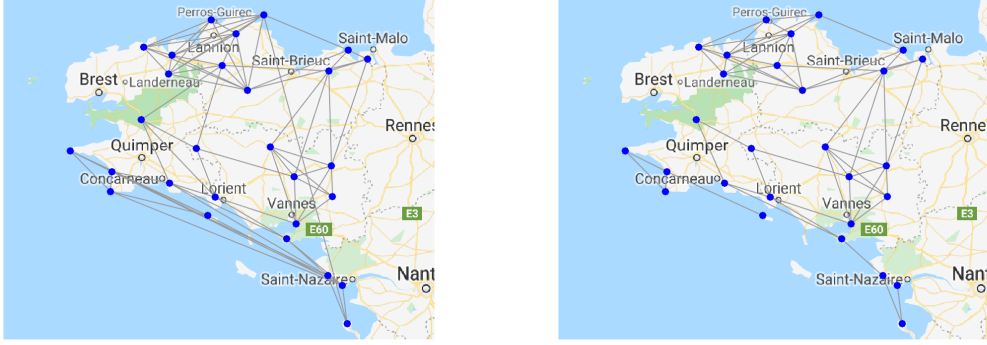


Figure 2.6 – GLMM inferred temperature and wind graphs, respectively.

We can see that GLMM outperforms other methods in terms of clustering accuracy. This is especially interesting as the examined dataset does not have an inherent ground truth graph structure supporting it. We argue that this suggests that our data can be well modelled with graphs. Therefore, additional constraints posed by Laplacian inference reduce the scope of the problem when compared to GMM, while they still allow for a lot more adaptivity when compared to simple K-means. The results obtained by GHMM suggest that, while there is still added benefit compared to the other two methods, the choice of kernel in our framework is very important. Namely, the smooth kernel is more general and therefore more flexible to possible outside influences, while the heat kernel is a more strict model, and as such should be used more carefully with noisy data.

We also investigate graph inference performance on this data. As there are no ground-truth graphs for this data, it is difficult to compare inference performance in a fair way. We thus construct pseudo-ground-truth graphs using the knowledge of original clusters: temperature and wind. For each cluster separately, we use all available data, preprocess them by subtracting the mean, and use a graph learning algorithm [55] to infer the graph. We present the results in Table 2.3. While this is clearly not a conclusive comparison due to the lack of real ground truth graphs, it is interesting to see that there is a significant gain of GLMM compared to K-means + GL, both of which use the same graph inference method.

To complement numerical findings, we further investigate graph quality by visual inspection. Figure 2.6 shows inferred temperature and wind graph topologies for GLMM. Figure 2.7 presents the same for GMM. We note that these are geographical graphs, plotted with node position corresponding to true measuring station coordinates and that none of the algorithms were given any prior information on node positions. We can see that graphs inferred by GLMM offer much more structure, mostly connecting neighbouring nodes, a desirable property in a weather-related geographical graph. We also note that the temperature and wind graph inferred by GLMM are fairly similar, with the largest difference appearing along the southern coast. One possible explanation could be that the temperature values are all highly correlated



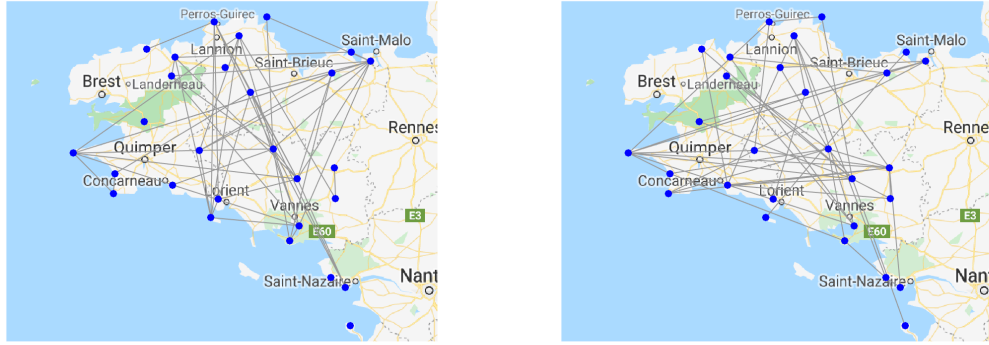


Figure 2.7 – GMM inferred temperature and wind graphs, respectively.

as they are regulated by the ocean, while the wind values in these areas are less stable, often change direction and strength along the coast.

Next, we evaluate how trained models generalise to new data. Namely, we separate both temperature and wind data into 600 training and 100 testing signals. We then randomly choose subsets of training data of different sizes to fit generative models. We run each algorithm 5 times with different random initialisations, and select the best run in terms of training data clustering performance. The unseen (test) data is then clustered using trained models. Namely, we determine the cluster for each new datapoint by estimating which of the pre-trained clusters it fits into the best. The whole experiment has been repeated 100 times, each time with different randomly selected measurements. Figure 2.8 shows the test data clustering NMSE (%) for all three methods, given different training set sizes.

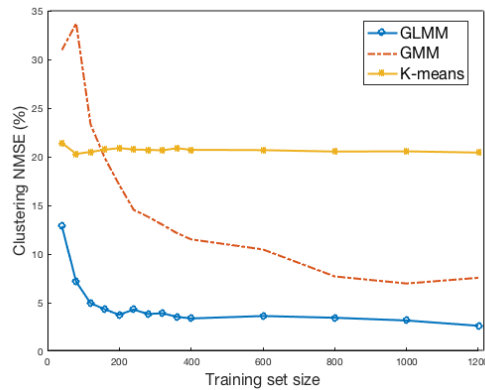


Figure 2.8 – Test data clustering performance for different sizes of training data. Each training dataset contains 50% temperature and 50% wind strength signals. The x-axis represents the training set size. The y-axis show signal clustering NMSE(%).

We can see a significantly better performance in our method compared to the others in

terms of generalisability to new data. We reiterate that this is especially significant as the results are demonstrated on data that does not inherently live on a graph. We argue that this shows the flexibility of our model, as it is able to generalise well to unseen data on signals that do not necessarily live on a mixture of graphs (but might be well representable with them). Furthermore, we note that, while increasing the size of our training set improves the performance of more adaptive algorithms, like GMM and GLMM, K-means does not show the possibility to adapt due to its very simple nature.

### Uber data

We next search for patterns in Uber data representing hourly pickups in New York City during the working days of September 2014 <sup>2</sup>. We divide the city into 29 taxi zones, and treat each zone as a node in our graphs. The signal on these nodes corresponds to the number of Uber pickups in the corresponding zone, per hour. We fix the number of clusters to  $K = 5$ , following the reasoning in [108].

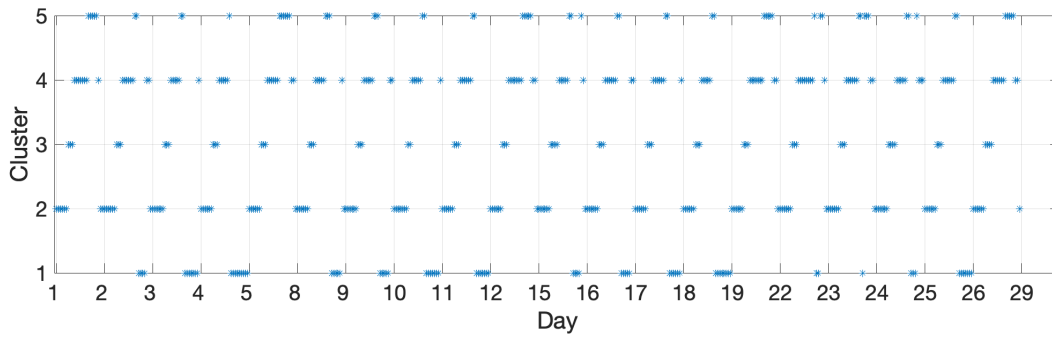


Figure 2.9 – Cluster indexes for Uber hourly signals, obtained with GLMM. Each dot represents one hour in the day, and thin vertical lines represent the beginning of each working day.

Figure 2.9 shows the clustering of hourly Uber signals into 5 different clusters, obtained with GLMM. We can see a slightly noisy periodic pattern, recurring daily. Note that no temporal information was given to the algorithm.

As there are no ground truth clusters, we inspect the results in terms of clustering consistency. Namely, we compare clusters obtained in each day to clusters obtained in all other days and average the result, in order to approximate the expected difference between daily patterns. If the periodic pattern occurs in the same way every day, the clustering is consistent, and the average difference will be zero. Note that without ground truth clusters, consistency does not give sufficient information in evaluating clustering performance. Namely, a clustering that groups all data into only one cluster will be deemed perfectly consistent. For that reason, we complement our findings with qualitative analysis of clusters. Table 2.4 shows the average NMSE obtained by comparing daily clusters. While GLMM has the best consistency, the result

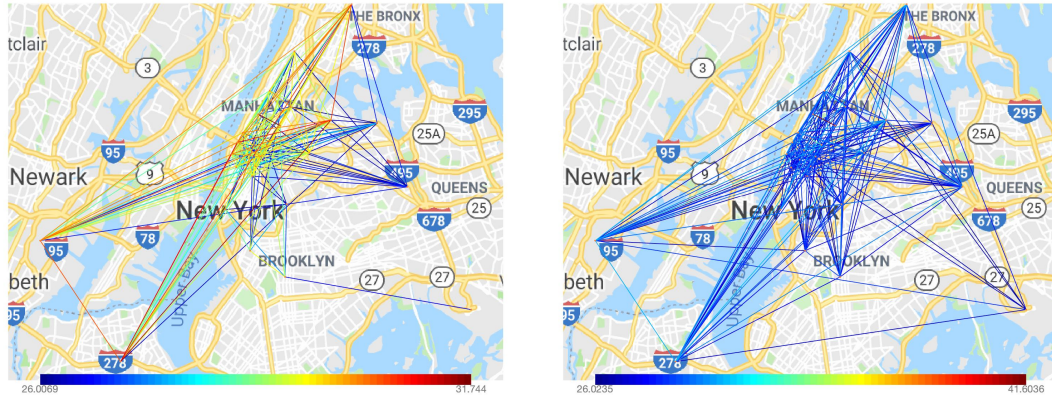
<sup>2</sup><https://github.com/fivethirtyeight/ubertlc-foil-response>

for GMM is not far in terms of consistency. However, GLMM on average separates the day into very sensible periods: 23h-6h, 6h-9h, 9h-15h + 22h-23h, 15h-20h and 20h-22h, separating night time, rush hours, day time and evening. At the same time, GMM on average separates the day into only 3 clusters, filling the remaining two clusters only with outliers, and putting most of the data into one large cluster: 6h-9h, 16h-20h and all the rest (20h-16h without 6h-9h). This explains a good score in consistency (as most data falls into the same cluster), however these clusters are not very meaningful. Note that GHMM and K-means produce clusters similar to those of GLMM, with the important difference of K-means not detecting morning rush hours (6h-9h and 9h-15h + 22h-23h are grouped into one cluster).

Table 2.4 – Clustering consistency error in terms of NMSE percentage (values between 0 and 100, smaller is better).

GHMM	GMM	GLMM	K-means + GL
11.45	10.75	<b>10.07</b>	13.55

Finally, Figure 2.10 presents two of the graphs inferred with GLMM. Each graph shows patterns of a different period in the day. We can see that the traffic during nights and early mornings is restricted to the city center and communications with the airports, while direct communication among non-central locations becomes more active later in the day. These different mobility patterns look reasonable with respect to daily people routine in NYC.



(a) 23h - 6h

(b) 15h - 20h

Figure 2.10 – GLMM inferred graphs corresponding to Uber patterns in different times of day.

### MNIST digits

Finally, we comment on the advantage that our method brings over standard GMMs in terms of interpretability in high dimensional settings. We tackle a well known classification problem in which there is no reason to believe that signals live on a graph. Each signal is one MNIST digit representing "0" or "1". Since each digit is given as a 20x20 pixel grayscale image, the di-

mension of our signal is 400. We gather signals by randomly choosing 1000 digits from class '0' and 1000 digits from class '1'. We test the performance of GLMM, GMM and K-means + GL, as well as a modified GLMM that was restricted to allow edges only between 2-hop neighbouring pixels (mGLMM). We show clustering performance in Table 2.5, but note that MNIST digit clustering is inherently a much lower dimensional problem, and should be treated as such. The experiments confirm this, giving better results for simpler methods: K-means performs very well as the simplest model, while GLMM still performs better than GMM due to its more focused nature and lower sensitivity to noise. The imposed structure in mGLMM simplifies the problem significantly and yields the best results, suggesting additional interpretative knowledge of the problem can be easily incorporated and reduce the dimensionality of the problem. Note that a mask of "allowed edges" was very easy to construct in this example, which is not necessarily always the case. However, these results imply that investigating contextual information might be a worthwhile task even when the masks are not as simple to construct. An alternative in such cases might be to include a soft mask as a matrix of costs weighing the value of each edge. In this way, prior knowledge can be incorporated in a way to promote larger values only in certain edges, without constraining the connectivity pattern in a definitive manner. Furthermore, we discuss the interpretability advantage offered by GLMM in settings

Table 2.5 – Clustering of 2000 randomly selected MNIST digits, of which 1000 represent digit '0' and 1000 represent digit '1'. The error is presented in terms of NMSE percentage (values between 0 and 100, smaller is better).

GLMM	mGLMM	GMM	K-means
1.76	<b>0.64</b>	7.18	0.89

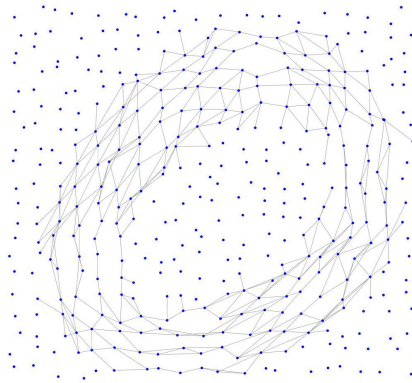


Figure 2.11 – Recovered graph with GLMM for the cluster of MNIST digit '0', no prior structure imposed

where this knowledge might not be available a priori, and can therefore not be imposed to the

learning problem. Figure 2.11 shows the graph topology inferred with GLMM corresponding to the cluster of digit zero. Note that no prior information or restrictions on edges were imposed in this version of the algorithm. To visualise the graph, we only considered edges adjacent to at least one vertex with a mean value larger than a threshold.

As expected, we can see that neighbouring pixels that form the number zero are strongly correlated. We can also see that pixels are rarely connected if they are not close in the image, even though no pixel position information was given to the graph inference algorithm. It is clear that such insights can be very valuable in terms of interpretability in these high dimensional settings.

We finish with noting that the cost of using this model when data does not live on a graph comes, of course, in terms of restrictions imposed by a valid graph Laplacian, as well as the sparsity constraint in the graph learning method. These restrictions reduce model flexibility and could therefore lead to less accurate results. However, in large dimensional settings where this model is meant to be used, these restrictions are not too constraining. Even more, as they implicitly reduce the dimensionality of the problem, they seem to even improve the performance on some datasets (as seen in Figure 2.4 and Table 2.3).

### 2.5.3 Inference of functional Brain Networks

Our final application is related to neuroscience, where GLMM is applied to analyse functional magnetic resonance imaging (fMRI) temporal data. This is a natural application for our algorithm, as brain regions can be synchronised in activity despite the absence of task or external stimuli [15]. Naturally, there is a need to both separate these activations of different brain networks and identify their structure. We give a short overview of the analysis here, and explain one experiment in detail. Further details on the whole project, including materials and methods used, some background information and additional analysis, can be found in the Appendix A.<sup>3</sup>

The general workflow of this work is summarized in Fig. 2.12. We began by extracting the mean blood-oxygen-level-dependent (BOLD) signals within regions of the Automated Anatomical Labeling (AAL) atlas for each session of each subject. We built a data matrix  $\mathbf{X}$  that contains the timecourses of all AAL 90 regions. The timecourses of all sessions and all subjects are then concatenated together to form a huge data matrix  $\mathbf{Y}$ . This is then fed to the GLMM algorithm which estimates the means or *metastates* ( $\mu_k$ ), learned graphs represented in the output as the graph Laplacians ( $L_k$ ), and signal clustering probabilities ( $\gamma$ ).

We first validate the performance of the proposed framework using task fMRI by demonstrating that the timing of the task paradigms are captured by the averaged  $\gamma$ -values across all subjects.

<sup>3</sup>The experiments presented here are part of a joint project with the Medical Image Processing Laboratory of EPFL:

A. Tarun, I. Ricchi, H. Petric Maretic, P. Frossard, and D. Van De Ville. Inference of Multiple functional Brain Networks using Graph Laplacian Mixture Model. *under preparation*

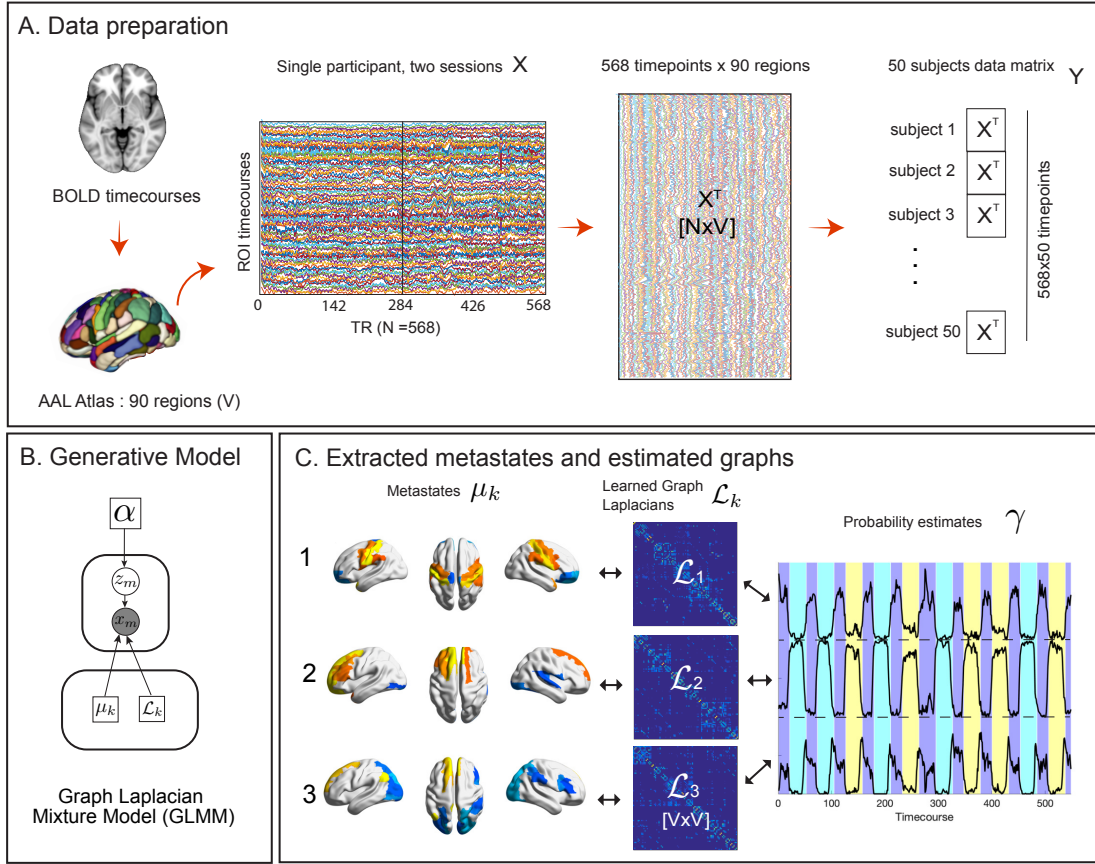


Figure 2.12 – General workflow: from fMRI signals to the extraction of metastates. (a) Mean BOLD signals from four sessions of fMRI recordings are computed within each region of the AAL90 atlas and are concatenated together to form the subject data matrix  $X$ . The final data ( $Y$ ) is obtained by concatenating 50 HCP subjects. (b) Plate notation of the generative model for extracting functional metastates and their corresponding estimated graphs represented through the Laplacians. (c) The output of the algorithm are  $K$ -number of metastates, their corresponding graph Laplacians, and the probability that each metastate would occur at a particular time-point. In the above example, we show  $K = 3$ .

We show that the extracted metastates consist of brain areas that are consistent with previously observed regions that are implicated in the corresponding task. Details about this experiment can be found in the Appendix (A.3.1 and A.3.2). Next, we apply GLMM to RS fMRI data and obtain the most prevalent networks of spontaneously interacting brain areas, as shown in detail below. Finally, an important benefit of the GLMM algorithm compared to other clustering methods is the estimation of graph Laplacians ( $L$ ). Not only this does help in obtaining more meaningful metastates (means), but it also conveys much more information and details about our clusters. We reveal indicative similarities of the learned functional graphs to the structural connectome derived from diffusion-weighted MRI (DW-MRI), and find a behaviorally-relevant organisation of functional brain graphs. Further details about this experiment are available in A.3.3.



### Extracted metastates during resting state fMRI

After validating the outcome of the proposed framework with task fMRI using the known experimental task paradigm as the ground truth, we applied the method to resting state (RS) fMRI. Fig. 2.13 displays the estimated metastates together with the approximated likelihood to occur at each time-point for one representative subject. This likelihood of occurrence gives a proxy of the metastate's activity profiles. As expected, we observe the default mode network (DMN) to be highly occurring. We found a metastate that clearly covers areas of the visual cortex, another metastate that shows regions corresponding to the auditory network and some portions of the frontoparietal cortex, and another one that contains the bilateral temporal cortices and the insula, which is analogous to the salience network. Additionally, we also found separately clustered activations in the left and right portion of the brain.

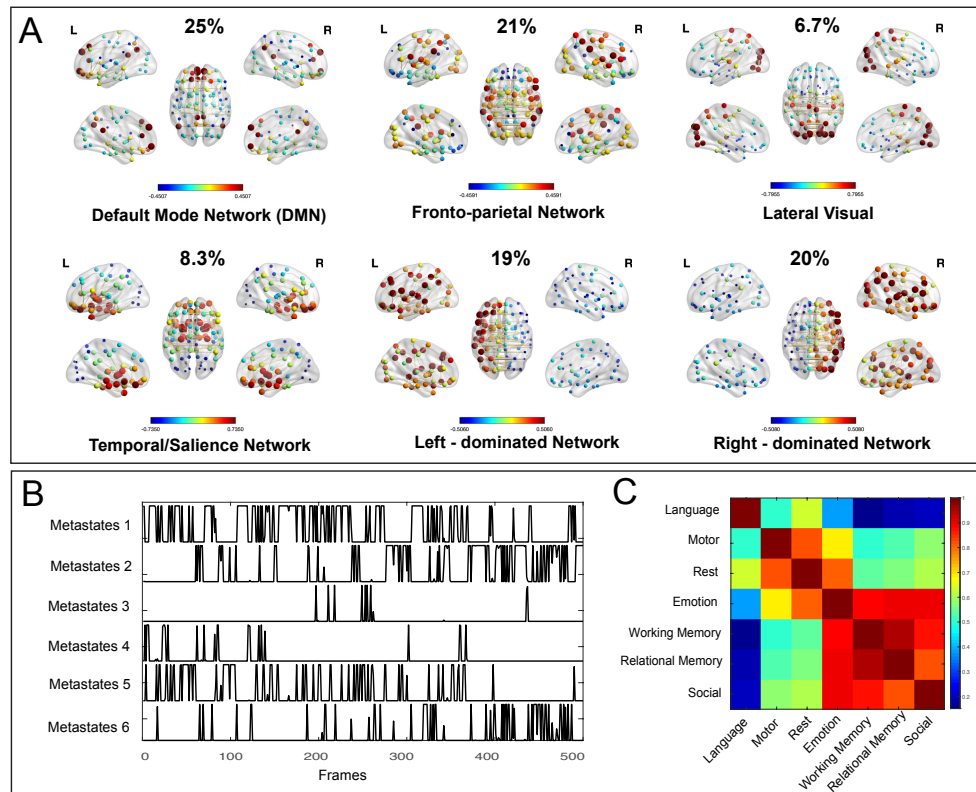


Figure 2.13 – **Extracted metastates from resting-state** (A) Six metastates corresponding to the RS data. Values in percentage show the occurrences of each metastate across all 50 subjects considered. (B) Example activity profile of each metastate for an example subject. (C) Correlation of estimated means corresponding to DMN-networks extracted from rest and all task fMRI data.

We found that metastates corresponding to the rest epochs of the tasks always corresponded to the DMN, which is expected. To understand the nature of the recovered DMN-related metastates, we computed the correlation across all pair-wise networks corresponding to

each task and RS data. Fig. 2.13(C) displays the correlation between the extracted DMN-related metastates from all tasks and RS fMRI. The  $x$ - and  $y$ -axes were sorted from lowest to highest correlation, showing a distinction between tasks that require low-level sensory-motor functions (*e.g.*, language, motor, rest) versus tasks that associated to high-level cognition (*e.g.*, emotion, working memory, relational memory, and social). More general findings for all experiments can be found in A.4.

## 2.6 Conclusion

We propose a generative model for mixtures of signals living on a collection of different graphs. We assume that both the mapping of signals to graphs, as well as the topologies of the graphs are unknown. We therefore jointly decouple the signals into clusters and infer multiple graph structures, one for each cluster. To do so, we formulate a Maximum a posteriori problem and solve it through Expectation minimisation. Our method offers high flexibility in terms of incorporation of different signal models, as well as simple inclusion of additional priors on the graph structure. We further argue that our model can be used for high dimensional clustering even when data is not assumed to have inherent graph structures, bringing additional interpretability to data. Simulations on real data achieve good clustering and meaningful graphs on weather data, infer interesting patterns in New York traffic, and find highly interpretable results on MNIST digits. Finally, the analysis of fMRI data using our method successfully identifies periods of time corresponding to different brain activities, reconstructs brain networks which are found in the literature and provides interesting insights into the organizational principles of human brain function and its relation to structure.



## 3 GOT: An Optimal Transport framework for Graph comparison

### 3.1 Introduction

Structured information is often represented by graphs that capture potentially complex structures. It stays however pretty challenging to analyse, classify or predict graph data, due to the lack of efficient measures for comparing graphs. In particular, the mere comparison of graph matrices is not necessarily a meaningful distance, as different edges can have a diverse importance in the graph. Spectral distances have also been proposed [53, 39], but they usually do not take into account all the information provided by the graphs, focusing only on the Laplacian matrix eigenvalues and ignoring a large portion of the structure encoded in eigenvectors. In addition to the lack of effective distances, a major difficulty with graph representations is that their nodes may not be aligned, which further complicates graph comparisons.

In this chapter, we propose a new framework for graph comparison, which permits to compute both the distance between two graphs under unknown permutations, and the transportation plan for data from one graph to another, under the assumption that the graphs have the same number of vertices<sup>1</sup>. Instead of comparing graph matrices directly, we propose to look at the smooth graph signal distributions associated to each graph, and to relate the distance between graphs to the distance between the graph signal distributions. We resort to optimal transport for computing the Wasserstein distance between distributions, as well as the corresponding transportation plan. Interestingly, the Wasserstein distance takes a closed-form expression in our settings, which essentially depends on the Laplacian matrices of the graphs under comparison. We further show that our novel Wasserstein graph distance has the important advantage of capturing the main structural information of the graphs.

Equipped with this distance, we formulate a new graph alignment problem for finding the permutation that minimises the mass transportation between a "fixed" distribution and a "permuted" distribution. This yields a nonconvex optimization problem that we solve

---

<sup>1</sup>This chapter contains work which has been published in:  
H. Petric Maretic, M. El Gheche, G. Chierchia, and P. Frossard. GOT: An Optimal Transport framework for Graph comparison. In *Advances in Neural Information Processing Systems 32*, pages 13876–13887. 2019

efficiently with a novel stochastic gradient descent algorithm. It permits to efficiently align and compare graphs, and it outputs a structurally meaningful distance and transport map. These are important elements in graph analysis, comparison, or graph signal prediction tasks. We finally illustrate the benefits of our new graph comparison framework in representative tasks such as noisy graph alignment, graph classification, and graph signal transfer. Our results show that the proposed distance outperforms both Gromov-Wasserstein and Euclidean distance for what concerns the graph alignment and graph clustering. In addition, we show the use of transport maps to predict graph signals. To the best of our knowledge, this is the only framework for graph comparison that includes the possibility to transport and adapt graph signals to another graph.

## 3.2 Related work

In the literature, many methods have formulated the graph matching problem as a quadratic assignment problem [120, 52], under the constraint that the solution is a permutation matrix. As this is an NP-hard problem, different relaxations have been proposed to find approximate solutions [79]. In this context, spectral clustering [18, 104] emerged as a simple relaxation, which consists of finding the orthogonal matrix whose squared entries sum to one, but the drawback is that the matching accuracy is suboptimal. To improve on this behavior, the semi-definite programming relaxation was adopted to tackle the graph matching problem by relaxing the non-convex constraint into a semi-definite one [97]. Spectral properties have also been used to inspect graphs and define different classes of graphs for which the convex relaxation is equivalent to the original graph matching problem [1] [34]. Other works focus on the general problem and propose provably tight convex relaxations for all graph classes [28]. Based on the assumption that the space of doubly-stochastic matrices is a convex hull, the graph matching problem was relaxed into a non-convex quadratic problem in [19, 126]. A related approach was recently proposed to approximate discrete graph matching in the continuous domain asymptotically by using separable functions [123]. Along similar lines, a Gumbel-sinkhorn network was proposed to infer permutations from data [75, 30]. The approach consists of producing a discrete permutation from a continuous doubly-stochastic matrix obtained with the Sinkhorn operator [103].

Closer to our framework, some recent works have studied the graph alignment problem from an optimal transport perspective. For example, Flamary *et al.* [35] propose a method based on optimal transport for empirical distributions with a graph-based regularization. The objective of this work is to compute an optimal transportation plan by controlling the displacement of a pair of points. Graph-based regularization encodes neighborhood similarity between samples on either the final position of the transported samples, or their displacement [32]. Gu *et al.* [45] define a spectral distance by assigning a probability measure to the nodes via the spectrum representation of each graph, and by using Wasserstein distances between probability measures. This approach however does not take into account the full graph structure in the alignment problem. Nikolentzos *et al.* [80] propose instead to match the graph

embeddings, where the latter are represented as bags of vectors, and the Wasserstein distance is computed between them. The authors also propose a heuristic to take into account possible node labels or signals.

Another line of works have looked at more specific graphs. Memoli [74] investigates the Gromov-Wasserstein distance for object matching, and Peyré *et al.* [87] propose an efficient algorithm to compute the Gromov-Wasserstein distance and the barycenter of pairwise dissimilarity matrices. The algorithm uses entropic regularization and Sinkhorn projections, as proposed by [20]. The work has many interesting applications, including multimedia with point-cloud averaging and matching, but also natural language processing with alignment of word embedding spaces [2]. Vayer *et al.* [112] build on this work and propose a distance for graphs and signals living on them. The problem is given as a combination between the Gromov-Wasserstein of graph distance matrices and the Wasserstein distance of graph signals. Barbe *et al.* [8] extend this idea and apply the Wasserstein distance between signals only after a graph filtering step, encoding the structural information of the graph into the filtered signals. Xu *et al.* [118] proposed a method based on Gromov-Wasserstein which simultaneously learns the graph alignment and the embeddings of graph nodes. The node embeddings are derived using optimal transport, which in turn helps in the graph matching task. The authors follow up in [117], devising a scalable version of Gromov-Wasserstein distance for graph partitioning and matching. However, while the above methods solve the alignment problem using optimal transport, the simple distances between aligned graphs do not take into account its global structure and the methods do not consider the transportation of signals between graphs.

Finally, the graph optimal transport framework presented below has recently been extended to an interesting coordinated optimal transport approach, where the structural properties of our distance are successfully used for graph sketching [26].

### 3.3 Graph Alignment with Optimal Transport

Despite recent advances in the analysis of graph data, it stays pretty challenging to define a meaningful distance between graphs. Even more, a major difficulty with graph representations is the lack of node alignment, which prevents from performing direct quantitative comparisons between graphs. In this section, we propose a new distance based on Optimal Transport (OT) to compare graphs through smooth graph signal distributions. Then, we use this distance to formulate a new graph alignment problem, which aims at finding the permutation matrix that minimizes the distance between graphs.

#### 3.3.1 Wasserstein distance between graphs

We interpret graphs as the structure that drives the probability distributions of signals, and as such, represent graphs through these distributions. Such a graph representation captures the properties and particularities of the graph, which are important in shaping data that lives on

it. Specifically, we consider two graphs  $\mathcal{G}_1$  and  $\mathcal{G}_2$  with Laplacian matrices  $L_1$  and  $L_2$ , and we consider smooth signals on them, which follow a Gaussian distribution as described in (1.14):

$$v^{\mathcal{G}_1}(x) = \mathcal{N}(0, L_1^\dagger) \quad (3.1)$$

$$v^{\mathcal{G}_2}(x) = \mathcal{N}(0, L_2^\dagger). \quad (3.2)$$

Recall that the smooth signal assumption is verified for most common graph and network datasets, and used as regularization in many graph applications, such as robust principal component analysis [101] and label propagation [127].

Instead of comparing graphs directly, we propose to compare their representations as smooth signal distributions. Specifically, we measure the dissimilarity between two aligned graphs  $\mathcal{G}_1$  and  $\mathcal{G}_2$  through the Wasserstein distance of the respective distributions  $v^{\mathcal{G}_1}$  and  $v^{\mathcal{G}_2}$ . More precisely, the 2-Wasserstein distance corresponds to the minimal “effort” required to transport one probability measure to another with respect to the Euclidean norm [77], that is

$$W_2^2(v^{\mathcal{G}_1}, v^{\mathcal{G}_2}) = \inf_{T_\# v^{\mathcal{G}_1} = v^{\mathcal{G}_2}} \int_{\mathcal{X}} \|x - T(x)\|^2 dv^{\mathcal{G}_1}(x), \quad (3.3)$$

where  $T_\# v^{\mathcal{G}_1}$  denotes the push forward of  $v^{\mathcal{G}_1}$  by the transport map  $T: \mathcal{X} \rightarrow \mathcal{X}$  defined on a metric space  $\mathcal{X}$ . For normal distributions such as  $v^{\mathcal{G}_1}$  and  $v^{\mathcal{G}_2}$ , the 2-Wasserstein distance can be explicitly written in terms of their covariance matrices, as seen in (1.19):

$$W_2^2(v^{\mathcal{G}_1}, v^{\mathcal{G}_2}) = \text{Tr}(L_1^\dagger + L_2^\dagger) - 2 \text{Tr} \left( \sqrt{L_1^{\frac{1}{2}} L_2^\dagger L_1^{\frac{1}{2}}} \right), \quad (3.4)$$

and the optimal transportation map is

$$T(x) = L_1^{\frac{1}{2}} \left( L_1^{\frac{1}{2}} L_2^\dagger L_1^{\frac{1}{2}} \right)^{\frac{1}{2}} L_1^{\frac{1}{2}} x. \quad (3.5)$$

The Wasserstein distance captures the structural information of the graphs under comparison. It is sensitive to differences that cause a global change in the connection between graph components, while it gives less importance to differences that have a small impact on the whole graph structure. Indeed, as graphs are represented through the distribution of smooth signals, the Wasserstein distance essentially measures the discrepancy in lower graph frequencies, known to capture the global graph structure. This behaviour is illustrated in Figure 3.1 by a comparison with a simple distance that is the Euclidean norm between the Laplacian matrices of the graphs.<sup>2</sup>

Moreover, the optimal transportation map enables the movement of signals from one graph to

---

<sup>2</sup>Note that in our setting a possible alternative to the Wasserstein distance could be the Kullback-Leibler (KL) divergence, whose expression is explicit for normal distributions.

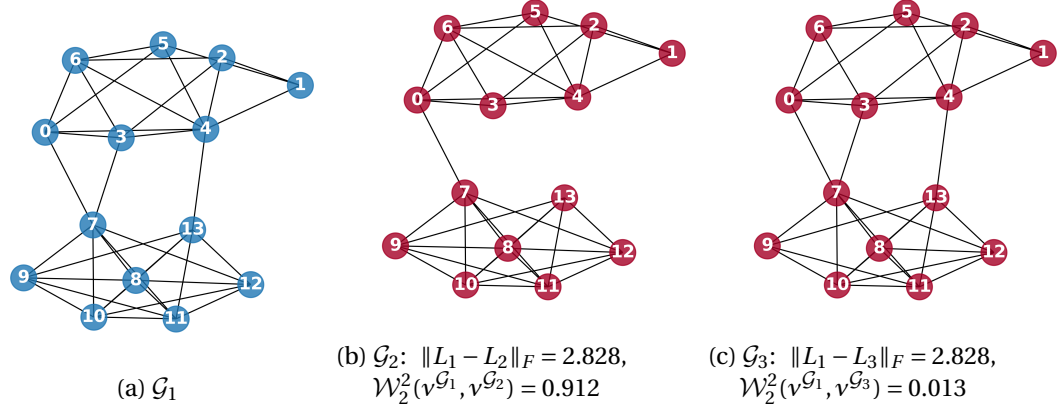


Figure 3.1 – Illustration of the structural differences captured with Wasserstein distance between graphs defined in (1.18). The graphs  $\mathcal{G}_2$  and  $\mathcal{G}_3$  are both copies of  $\mathcal{G}_1$ , with 2 edges removed. The modification in  $\mathcal{G}_2$  is very influential, as the two communities are almost disconnected; here, both Frobenius norm and Wasserstein distance measure a significant difference w.r.t.  $\mathcal{G}_1$ . Conversely, the modification in  $\mathcal{G}_3$  is hardly noticeable; here, the Frobenius norm still measures a significant difference, whereas the Wasserstein distance does not. The latter is a desirable property in the context of graph comparison.

another. This is a continuous Lipschitz mapping that adapts a graph signal to the distribution of another graph, while keeping similarity. This results in a simple and efficient prediction of the signal on another graph.

### 3.3.2 Graph alignment

Equipped with a measure to compare aligned graphs of the same size through signal distributions, we now propose a new formulation of the graph alignment problem. It is important to note that the graph signal distributions depend on the enumeration of nodes chosen to build  $L_1$  and  $L_2$ . While in some cases (e.g., dynamically changing graphs, multilayer graphs, etc...) a consistent enumeration can be trivially chosen for all graphs, it is not always the case. This generally leads to the challenging problem of first estimating an *a priori* unknown permutation between graphs before they could be compared. In our approach, we are given two connected graphs  $\mathcal{G}_1$  and  $\mathcal{G}_2$ , each with  $N$  distinct vertices and with different sets of edges. We aim at finding the optimal transportation map  $T$  from  $\mathcal{G}_1$  to  $\mathcal{G}_2$ . However, the vertices of these graphs are not necessarily aligned. In order to take all possible enumerations into account, we define the probability measure of a permuted representation for the graph  $\mathcal{G}_2$  as

$$v_P^{\mathcal{G}_2} = \mathcal{N}(0, (P^\top L_2 P)^\dagger) = \mathcal{N}(0, P^\top L_2^\dagger P), \quad (3.6)$$

where  $P \in \mathbb{R}^{N \times N}$  is a permutation matrix. Consequently, our graph alignment problem consists in finding the permutation that minimizes the mass transportation between  $v^{\mathcal{G}_1}$  and  $v_P^{\mathcal{G}_2}$ , which

**Algorithm 1** Approximate solution to the graph alignment problem defined in (3.7).

---

**Require:** Graphs  $\mathcal{G}_1$  and  $\mathcal{G}_2$

**Require:** Sampling size  $S \in \mathbb{N}$ , learning rate  $\gamma > 0$ , and constant  $\tau > 0$

**Require:** Random initialization of matrices  $\eta_0$  and  $\sigma_0$

- 1: **for**  $t = 0, 1, \dots$  **do**
- 2:   Draw samples  $\{\epsilon_t^{(s)}\}_{1 \leq s \leq S}$  from the distribution  $q_{\text{unit}}$
- 3:   Define the stochastic approximation of the cost function as

$$J_t(\eta_t, \sigma_t) = \frac{1}{S} \sum_{s=1}^S \mathcal{W}_2^2(v^{\mathcal{G}_1}, v_{\mathcal{S}_\tau(\eta_t + \sigma_t \odot \epsilon_t^{(s)})}^{\mathcal{G}_2})$$

- 4:    $g_t \leftarrow$  gradient of  $J_t$  evaluated at  $(\eta_t, \sigma_t)$
  - 5:    $(\eta_{t+1}, \sigma_{t+1}) \leftarrow$  update of  $(\eta_t, \sigma_t)$  using  $g_t$
  - 6: **end for**
  - 7: **return**  $P = \mathcal{S}_\tau(\eta_*)$
- 

reads

$$\underset{P \in \mathbb{R}^{N \times N}}{\text{minimize}} \mathcal{W}_2^2(v^{\mathcal{G}_1}, v_P^{\mathcal{G}_2}) \quad \text{s.t.} \quad \begin{cases} P \in [0, 1]^N \\ P \mathbb{1}_N = \mathbb{1}_N \\ \mathbb{1}_N^\top P = \mathbb{1}_N \\ P^\top P = I_{N \times N}, \end{cases} \quad (3.7)$$

where  $\mathbb{1}_N = [1 \dots 1]^\top \in \mathbb{R}^N$  and  $I_{N \times N}$  is the  $N \times N$  identity matrix. The constraints in 3.7 ensure that  $P$  is a valid permutation matrix. According to (3.1), (3.4), (3.6), the above distance boils down to

$$\mathcal{W}_2^2(v^{\mathcal{G}_1}, v_P^{\mathcal{G}_2}) = \text{Tr}(L_1^\dagger + P^\top L_2^\dagger P) - 2 \text{Tr} \left( \sqrt{L_1^{\frac{\dagger}{2}} P^\top L_2^\dagger P L_1^{\frac{\dagger}{2}}} \right). \quad (3.8)$$

The optimal permutation allows us to compare  $\mathcal{G}_1$  and  $\mathcal{G}_2$  when the consistent enumeration of nodes is not available. This is however a non-convex optimization problem that cannot be easily solved with standard tools. In the next section, we present an efficient algorithm to tackle this problem.

### 3.4 GOT Algorithm

We propose to solve the OT-based graph alignment problem described in the previous section via stochastic gradient descent. The latter is summarized in Algorithm 1, and its derivation is presented in details below.

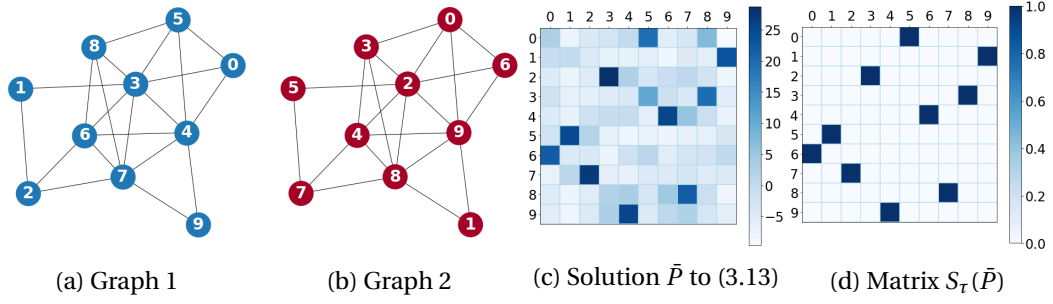


Figure 3.2 – Illustrative example of the graph alignment problem. The solution to (3.13) is a matrix  $\tilde{P}$  whose rows may be interpreted as assignment log-likelihoods. Applying the Sinkhorn operator to  $\tilde{P}$  yields a matrix whose rows are assignment probabilities from Graph 1 (columns) to Graph 2 (rows).

### 3.4.1 Optimization

The main difficulty in solving Problem (3.7) arises from the constraint that  $P$  is a permutation matrix, which leads to a discrete optimization problem with a factorial number of feasible solutions. We propose to circumvent this issue through an implicit constraint reformulation. The idea is that the constraints in (3.7) can be enforced implicitly by using the Sinkhorn operator [103, 20, 38, 75]. Given a square matrix  $P \in \mathbb{R}^{N \times N}$  (not necessarily a permutation) and a small constant  $\tau > 0$ , the Sinkhorn operator  $\mathcal{S}_\tau$  normalizes the rows and columns of  $\exp(P/\tau)$  via the multiplication by two diagonal matrices  $A$  and  $B$ , yielding<sup>3</sup>

$$\mathcal{S}_\tau(P) = A \exp(P/\tau) B. \quad (3.9)$$

The diagonal matrices  $A$  and  $B$  are computed iteratively as follows:

$$A^{[k]} = \text{diag}(P^{[k]} \mathbb{1}_N)^{-1} \quad (3.10)$$

$$B^{[k]} = \text{diag}(\mathbb{1}_N^\top A^{[k]} P^{[k]})^{-1} \quad (3.11)$$

$$P^{[k+1]} = A^{[k]} P^{[k]} B^{[k]}, \quad (3.12)$$

with  $P^{[0]} = \exp(P/\tau)$ . It can be shown [75] that the operator  $\mathcal{S}_\tau$  yields a permutation matrix in the limit  $\tau \rightarrow 0$ . Consequently, with a slight abuse of notation (as  $P$  no longer denotes a permutation), we can rewrite Problem (3.7) as follows

$$\underset{P \in \mathbb{R}^{N \times N}}{\text{minimize}} \quad \mathcal{W}_2^2(v^{\mathcal{G}_1}, v_{\mathcal{S}_\tau(P)}^{\mathcal{G}_2}). \quad (3.13)$$

The above cost function is differentiable [69], and can be thus optimized by gradient descent. An illustrative example of a solution of the proposed approach is presented in Fig. 3.2.

<sup>3</sup>Note that  $\exp$  is applied element-wise to ensure the positivity of the matrix entries.

### 3.4.2 Stochastic exploration

Problem (3.13) is highly nonconvex, which may cause gradient descent to converge towards a local minimum. Hence, instead of directly optimizing the cost function in (3.13), we can optimize its expectation w.r.t. the parameters  $\theta$  of some distribution  $q_\theta$ , yielding

$$\underset{\theta}{\text{minimize}} \mathbb{E}_{P \sim q_\theta} \left\{ \mathcal{W}_2^2(v^{\mathcal{G}_1}, v_{S_r(P)}^{\mathcal{G}_2}) \right\}. \quad (3.14)$$

The optimization of the expectation w.r.t. the parameters  $\theta$  aims at shaping the distribution  $q_\theta$  so as to put all its mass on a minimizer of the original cost function, thus integrating the use of Bayesian exploration in the optimization process.

A standard choice for  $q_\theta$  in continuous optimization is the multivariate normal distribution, thus leading to  $\theta = (\eta, \sigma) \in \mathbb{R}^{N \times N} \times \mathbb{R}^{N \times N}$  and  $q_\theta = \prod_{i,j} \mathcal{N}(\eta_{ij}, \sigma_{ij}^2)$ . By leveraging the reparameterization trick [62, 33], which boils down to the equivalence

$$(\forall (i, j) \in \{1, \dots, N\}^2) \quad P_{ij} \sim \mathcal{N}(\eta_{ij}, \sigma_{ij}^2) \quad \Leftrightarrow \quad \begin{cases} \epsilon_{ij} \sim \mathcal{N}(0, 1) \\ P_{ij} = \eta_{ij} + \sigma_{ij} \epsilon_{ij}, \end{cases} \quad (3.15)$$

the above problem can be reformulated as<sup>4</sup>

$$\underset{\eta, \sigma}{\text{minimize}} \underbrace{\mathbb{E}_{\epsilon \sim q_{\text{unit}}} \left\{ \mathcal{W}_2^2(v^{\mathcal{G}_1}, v_{S_r(\eta + \sigma \odot \epsilon)}^{\mathcal{G}_2}) \right\}}_{J(\eta, \sigma)}, \quad (3.16)$$

where  $q_{\text{unit}} = \prod_{i,j} \mathcal{N}(0, 1)$  denotes the multivariate normal distribution with zero mean and unitary variance. The advantage of this reformulation is that the gradient of the above stochastic function can be approximated by sampling from the parameterless distribution  $q_{\text{unit}}$ , yielding

$$\nabla J(\eta, \sigma) \approx \sum_{\epsilon \sim q_{\text{unit}}} \nabla \mathcal{W}_2^2(v^{\mathcal{G}_1}, v_{S_r(\eta + \sigma \odot \epsilon)}^{\mathcal{G}_2}). \quad (3.17)$$

The problem can be thus solved by stochastic gradient descent [61]. An illustrative application of this approach on a simple one-dimensional nonconvex function is presented in Fig. 3.3. Under mild assumptions, the algorithm converges almost surely to a critical point, which is not guaranteed to be the global minimum, as the problem is nonconvex.

The computational complexity of our naive implementation is  $O(N^3)$  per iteration, due to the matrix square-root operation based on a singular value decomposition (SVD). A better option consists of approximating the matrix square-root with the Newton's method [66]. These iterations only involve matrix multiplications, which can take advantage of the matrix sparsity, thus resulting in a faster implementation than SVD. Moreover, the computation of pseudo-inverses can be avoided by adding a small diagonal shift to the Laplacian matrices and directly

<sup>4</sup>Note that  $\odot$  is the entry-wise (Hadamard) product between matrices.



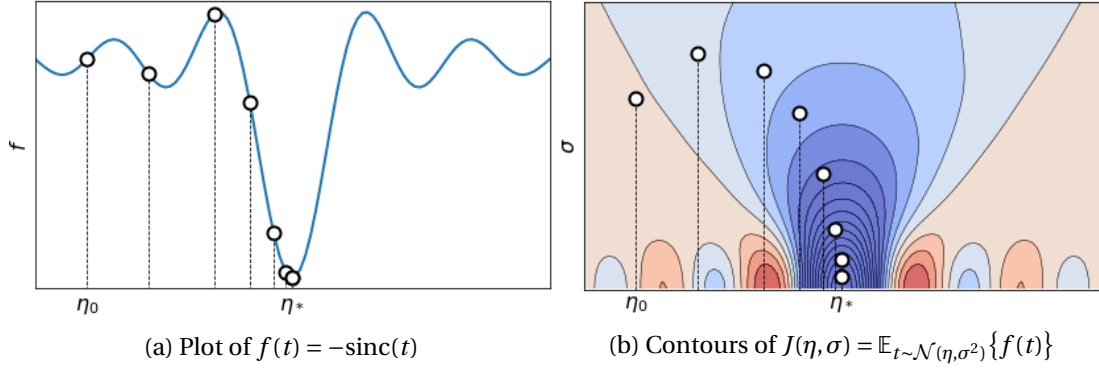


Figure 3.3 – Illustrative example of stochastic exploration. The white circles mark the iterates  $(\eta_0, \sigma_0), \dots, (\eta_*, \sigma_*)$  produced by optimizing  $J$  (the expectation of  $f$ ) via stochastic gradient descent. As this optimization is performed in the space of parameters  $\eta$  and  $\sigma$  (see the right panel), the algorithm avoids local minima and successfully converges to the global minimum of both  $J$  and  $f$ .

computing the inverse matrices, which is orders of magnitude faster. This is not a large concern though, as it can be done in preprocessing and only needs to be done once.

### 3.5 Experimental results

We illustrate the behaviour of our approach, named GOT, in terms of both distance metric computation and transportation map inference. We show how the distance can be beneficial in computing alignment between structured graphs even when they are very different, due to its ability to strongly capture structural properties. For similar reasons, the metric is able to properly separate instances of random graphs according to their original model. Finally, we show illustrations of the use of transportation maps for signal prediction in simple image classes.

Prior to running experiments, we chose the parameters  $\tau$  (Sinkhorn) and  $\gamma$  (learning rate) with grid search, while  $S$  (sampling size) was fixed empirically. In all experiments, we set  $\tau = 5, \gamma = 0.2$ , and  $S = 30$ . We set the maximal number of Sinkhorn iterations to 10, and we run stochastic gradient descent for 3000 iterations (even though the algorithm converges long before, after around 1000 iterations, typically). As our algorithm seems robust to different initialisations, we used random initialisation in all our experiments. The code is available at <https://github.com/Hermine/GOT>. Finally, the algorithm was implemented using automatic differentiation (in PyTorch with AMSGrad [91]).

#### 3.5.1 Alignment of structured graphs

We generate a stochastic block model graph  $\mathcal{G}_1$  with 40 nodes and 4 communities. A noisy version of this graph,  $\mathcal{G}_2$ , is created by randomly removing edges within communities with

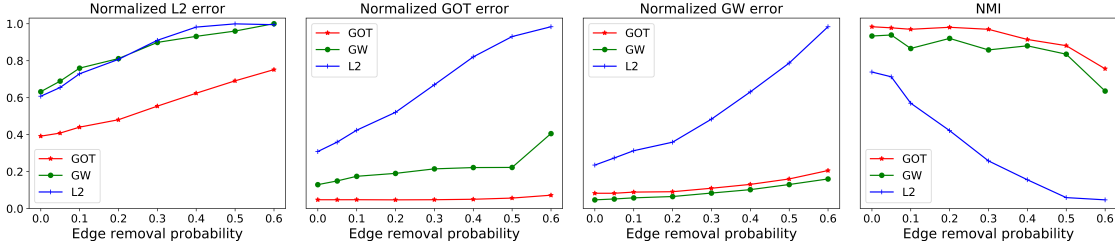


Figure 3.4 – Alignment and community detection performance for distorted stochastic block model graphs as a function of the edge removal probability. The first three plots show different error measures (closer to 0 the better); the last one shows the community detection performance in terms of Normalized Mutual Information (NMI closer to 1 the better).

probability  $p = 0.5$ , and edges between communities with increasing probabilities  $p \in [0, 0.6]$ . We then generate a random permutation to change the order of nodes in the noisy graph  $\mathcal{G}_2$ . We investigate the influence of a distance metric on alignment recovery. We compare three different methods for graph alignment, namely the proposed method based on the suggested Wasserstein distance between graphs (GOT), the proposed stochastic algorithm with the Euclidean distance (L2), and the state-of-the-art Gromov-Wasserstein distance [87] [112] for graphs (GW), based on the Euclidean distance between shortest path matrices, as proposed in [112]. After adjusting parameters for all compared methods, we repeat each experiment 50 times, with different  $\mathcal{G}_\infty$ s, and show the results in Figure 3.4.

Apart from analysing the distance between aligned graphs with all three error measures, we also evaluate the structural recovery of these community-based models by inspecting the normalized mutual information (NMI) for community detection. While GW slightly outperforms GOT in terms of its own error measure, GOT clearly performs better in terms of all other inspected metrics. In particular, the last plot shows that the structural information is well captured in GOT, and communities are successfully recovered even when the graphs contain a large amount of introduced perturbations.

### 3.5.2 Graph classification

We next tackle the task of graph classification. In order to successfully classify random instances of different graph models, a method needs to both find a good alignment and capture structural differences in graphs. Such a method can compare graphs in a structurally meaningful way, and can therefore be used to classify graphs based on their global properties.

We create 100 graphs following five different models (20 per model), namely Stochastic Block Model [51] with 2 blocks (SBM2), Stochastic Block Model with 3 blocks (SBM3), random regular graph (RG) [105], Barabasy-Albert model (BA) [7], and Watts-Strogatz model (WS) [115]. All graphs have 20 nodes and a similar number of edges to make the classification task more meaningful. All graphs are randomly permuted.

We use GOT to align graphs and compute the Wasserstein distance between them, and then use these distances with a simple non-parametric 1-NN classification algorithm to eventually classify graphs. We compare to several methods for graph alignment and comparison, followed by the same non-parametric 1-NN classification algorithm: GW [87, 112], FGM [125], IPFP [65], RRWM [19] and NetLSD[110].

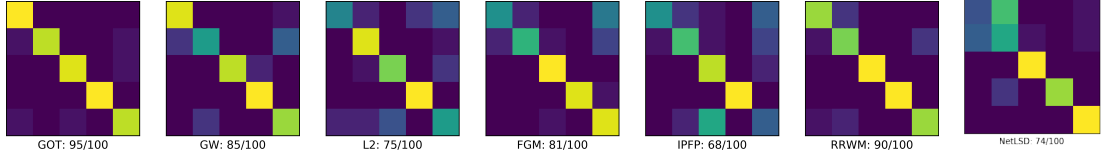


Figure 3.5 – Confusion matrices for 1-NN classification results on random graph models: SBM2, SBM3, RG, BA and WS respectively. Rows represent indices of actual classes, while columns are the ones of the predicted classes. Yellow squares correspond to values close to 100%, and blue to values close to 0%. A perfect results would be a diagonal matrix of yellow squares.

We present the results in terms of confusion matrices in Figure 3.5, accompanied with their accuracy scores. GOT clearly outperforms the other methods in terms of general accuracy, with GW and RRWM also performing well, even if they have more difficulties with SBMs and the WS model. This once again suggests that GOT is able to capture well the structural information of graphs.

### 3.5.3 Graph signal transportation

Finally, we look at the relevance of the transportation plans produced by GOT in illustrative experiments with simple images. We use the MNIST dataset, which contains around 60000 images of size  $28 \times 28$  displaying handwritten digits from 0 to 9, with 6000 per class. For each class  $c \in \{0, \dots, 9\}$ , we stack all the available images into a feature matrix of size  $6000 \times 784$ , and we build a graph over the resulting 784 feature vectors. To construct a graph, we first create a 20-nearest-neighbour binary graph, which we then square (multiply with itself) to obtain the final graph, capturing 2-hop distances and creating more meaningful weights. Hence, each class of digits is represented by a graph of 784 nodes (i.e., image pixels), yielding 9 aligned graphs  $\mathcal{G}_{\text{zero}}, \mathcal{G}_{\text{one}}, \dots, \mathcal{G}_{\text{nine}}$ .

Each image of a given class can be seen as a smooth signal  $x \in \mathbb{R}^{784}$  that lives on the corresponding graph. A transportation plan  $T$  is then constructed as written in (3.5), between the source graph (e.g.,  $\mathcal{G}_{\text{zero}}$ ) and all other graphs (e.g.,  $\mathcal{G}_{\text{one}}, \mathcal{G}_{\text{two}}, \dots, \mathcal{G}_{\text{nine}}$ ).

Figure 3.6 shows two original “zero” digits with different inclinations, transported to the graphs of all other digits. We can see that the predicted digits are recognisable, because they are adapted to their corresponding graphs, and they further keep the similarity with the original digit in terms of inclination. This shows that transported signals successfully adapt to the structure of a new graph, while keeping properties from the original signal. Note that images of

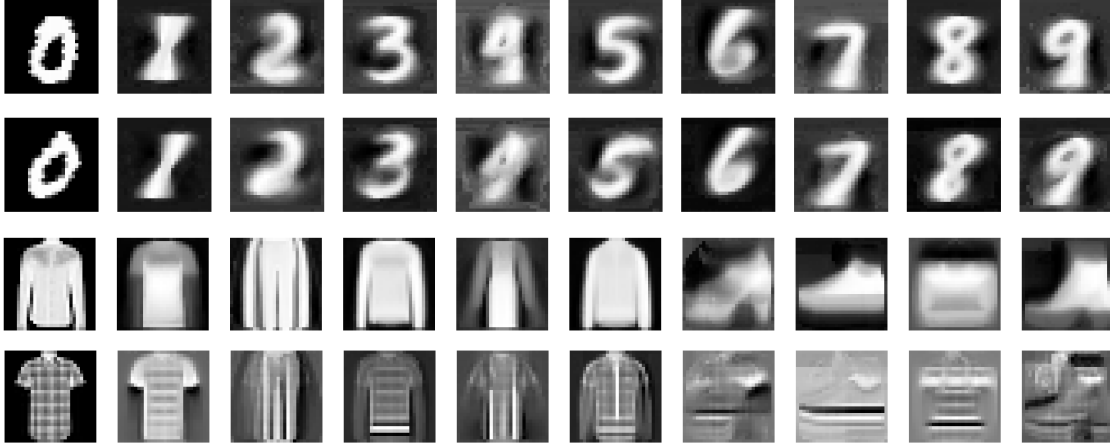


Figure 3.6 – *First two rows*: Original “zero” digits in MNIST dataset, and their images transported to graphs of different digits. The transported digits in each row follow the inclination of the original zero digit. *Last two rows*: Original “Shirt” images in Fashion MNIST dataset, and their images transported to the graphs of other classes (“T-shirt”, “Trouser”, “Pullover”, “Dress”, “Coat”, “Sandal”, “Sneaker”, “Bag”, “Ankle boot”).

digits 1 to 9 are fully generated from the original “zero” digit and their corresponding transport plans  $T$ . This is particularly remarkable if we note that these newly generated images are created simply as  $Tx$ , where the transportation plan  $T$  is a linear operator and  $x$  is the “zero” digit.

We repeated the same experiment on Fashion MNIST, and reported the results in Figure 3.6. By transporting a “Shirt” image to the graphs of classes “T-shirt”, “Trouser”, “Pullover”, “Dress”, “Coat”, “Sandal”, “Sneaker”, “Bag”, “Ankle boot”, we can remark that the predicted images are still recognisable with a good degree of fidelity. Furthermore, it is easy to see that the white shirt translates to white clothing items, while the textured shirt leads to textured items, reinforcing the observation that our signal transport function adapts a signal to the new graph structure, while keeping some of its original properties. Note that this graph signal transportation plan is unique to GOT as it uses the structure of graphs to adapt the signal, even when the graphs are already aligned. Such signal adaptation methods have been largely unexplored. This experiment confirms the unique potential that the GOT framework offers in graph signal prediction through adaptation of a graph signal to another graph.

### 3.6 Conclusion

We presented an optimal transport based approach for computing the distance between two graphs and the associated transportation plan. Equipped with this distance, we formulated the problem of finding the permutation between two unaligned graphs, and we proposed to solve it with a novel effective stochastic gradient descent algorithm. We evaluated the proposed approach in the context of graph alignment, graph classification, and graph signal

transportation. Our experiments confirmed that GOT can efficiently capture the structural information of graphs, while our algorithm can effectively find corresponding alignments between graphs, resulting in a structurally meaningful graph distance. Furthermore, the proposed transportation plan leads to promising results for the transfer of signals from one graph to another. Finally, despite the efficiency of our approach, there are several improvements and generalisation that can be explored. The main limitation of our approach is the assumption that the graphs are of the same size. To address this issue, we extend this framework to one-to-many graph alignment and propose an efficient algorithm in Chapter 4. After that, we explore the generalisation of GOT to filter graph distances, which are not limited to distributions of smooth signals and offer a more flexible comparison of graphs. In addition to that, we formulate an approximation to the Wasserstein distance cost, leading to more efficient algorithms, and present this work in details in Chapter 5.



## 4 One-to-Many alignment based on the Wasserstein graph distance

### 4.1 Introduction

Comparing graphs of different sizes can be a particularly challenging problem. As we have seen in the last chapter, when two graphs are not aligned *a priori*, graph alignment must be performed prior to any comparison. This leads to the challenging problem of estimating an unknown assignment between their vertices. Furthermore, when the graphs are not of the same size, the assignment cannot be a permutation matrix anymore, and the question of a meaningful assignment structure can be posed. Several relaxations to this problem exist, often minimizing a suitable distance between graphs under the quadratic assignment model, such as the  $\ell_2$ -norm between the graph adjacency matrices [123], or the Gromov-Wasserstein distance [117]. However, these approaches may yield solutions that are unable to capture the importance of edges with respect to the overall structure of the graph. An alternative that seems more appropriate for graph comparison is based on the Wasserstein distance between the graph signal distributions presented in the last chapter, but it was limited to graphs of the same size.

In this chapter, we build on the optimal transport framework for graph comparison, and propose a novel method for comparing graphs of different sizes. Specifically, we first cast a new formulation for the one-to-many graph alignment problem, which aims at matching a node in the smaller graph with one or more nodes in the larger graph. To accommodate for the nonconvexity of the problem, we propose a stochastic formulation based on a novel Dykstra operator to implicitly ensure that the solution is a one-to-many soft-assignment matrix. This allows us to devise an efficient algorithm based on stochastic gradient descent, which naturally integrates Bayesian exploration in the optimization process, so as to help finding better local minima. We illustrate the benefits of our new graph comparison framework in representative tasks such as graph alignment and graph classification on synthetic and real datasets. Our results show that the Wasserstein distance combined with the one-to-many graph assignment permits to outperform both Gromov-Wasserstein and Euclidean distance in these tasks, suggesting that our approach outputs a structurally meaningful distance to

efficiently align and compare graphs of different sizes. These are important elements in graph analysis, comparison, or graph signal prediction tasks.

## 4.2 One-to-many assignment problem

Following the previous chapter, instead of comparing graphs directly, we look at their signal distributions, which are governed by the graphs. Specifically, we measure the dissimilarity between two aligned graphs  $\mathcal{G}_1$  and  $\mathcal{G}_2$  through the Wasserstein distance of the respective distributions  $\nu^{\mathcal{G}_1} = \mathcal{N}(0, L_1^\dagger)$  and  $\nu^{\mathcal{G}_2} = \mathcal{N}(0, L_2^\dagger)$ , which can be calculated explicitly as

$$\mathcal{W}_2^2(\nu^{\mathcal{G}_1}, \nu^{\mathcal{G}_2}) = \text{Tr}(L_1^\dagger + L_2^\dagger) - 2 \text{Tr}\left(\sqrt{L_1^{\frac{\dagger}{2}} L_2^\dagger L_1^{\frac{\dagger}{2}}}\right). \quad (4.1)$$

The Wasserstein distance  $\mathcal{W}_2^2$  presented in the last chapter requires the two graphs to be of the same size. However, we now want to compare graphs of different sizes as well, which represents a common setting in practice. Throughout the rest of this work, we will consider two graphs  $\mathcal{G}_1 = (V_1, E_1)$  and  $\mathcal{G}_2 = (V_2, E_2)$ , and we arbitrarily denote by  $\mathcal{G}_1$  the graph with the smaller number of nodes.

We now compare graphs of different sizes by looking for the one-to-many assignment between their vertices, similarly to [124]. This is illustrated in the toy example of Figure 4.1, where every vertex of the smaller graph  $\mathcal{G}_1$  is assigned to one or more vertices in the larger graph  $\mathcal{G}_2$ , and every vertex of  $\mathcal{G}_2$  is assigned to exactly one vertex in  $\mathcal{G}_1$ . Let  $k_{\max} \geq 1$  be the maximum number of nodes in  $\mathcal{G}_2$  matched to a single node in  $\mathcal{G}_1$ . Such a one-to-many assignment can be described by a matrix  $P \in \mathbb{R}^{|V_1| \times |V_2|}$  satisfying the constraints

$$\mathcal{C}_{\text{hard}} = \left\{ P \in \mathbb{R}^{|V_1| \times |V_2|} : \begin{array}{l} (\forall i, \forall j) P_{ij} \in \{0, 1\} \\ (\forall i) \sum_j P_{ij} \in [1, k_{\max}] \\ (\forall j) \sum_i P_{ij} = 1 \end{array} \right\}. \quad (4.2)$$

In words, the matrix  $P$  only takes values zero or one, which corresponds to a hard assignment. Moreover, the sum of each matrix row has to be between 1 and  $k_{\max}$ , ensuring that every vertex of  $\mathcal{G}_1$  is matched to at least one and at most  $k_{\max}$  vertices of  $\mathcal{G}_2$ . Finally, the sum of each matrix column has to be exactly one, so that every vertex of  $\mathcal{G}_2$  is matched to exactly one vertex of  $\mathcal{G}_1$ . To ensure that  $\mathcal{C}_{\text{hard}}$  is a nonempty constraint set, we require that

$$1 \leq k_{\max} \leq 1 + |V_2| - |V_1|. \quad (4.3)$$

Given the true assignment matrix  $P_* \in \mathcal{C}_{\text{hard}}$ , the larger graph  $\mathcal{G}_2$  can be aligned to the smaller graph  $\mathcal{G}_1$  by transforming its Laplacian matrix as  $P_* L_2 P_*^\top$  [124]. It further yields an associated



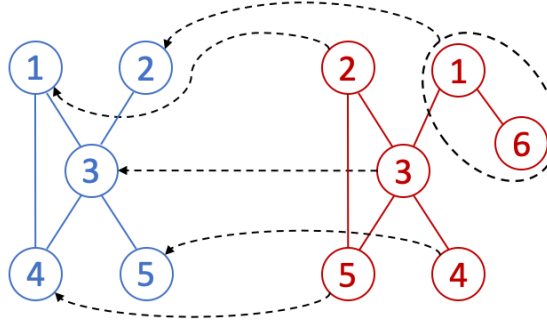


Figure 4.1 – One-to-many assignment between different-size graphs.

distribution of signals given as:

$$v_{P_*}^{\mathcal{G}_2} = \mathcal{N}(0, (P_* L_2 P_*^\top)^\dagger). \quad (4.4)$$

The graph alignment with the one-to-many assignment solution thus naturally leads to the use of  $\mathcal{W}_2^2(v^{\mathcal{G}_1}, v_{P_*}^{\mathcal{G}_2})$  of Equation (4.1) for evaluating the distance<sup>1</sup> between graphs that originally have different sizes.

Of course, the true assignment matrix  $P_*$  is often unknown beforehand. We are thus interested in estimating the best alignment, or equivalently in finding the assignment matrix  $P$  that minimizes the distance between two graphs  $\mathcal{G}_1$  and  $\mathcal{G}_2$ , leading to the optimization problem

$$\underset{P \in \mathcal{C}_{\text{hard}}}{\text{minimize}} \quad \mathcal{W}_2^2(v^{\mathcal{G}_1}, v_P^{\mathcal{G}_2}). \quad (4.5)$$

The main difficulty in solving Problem (4.5) arises from the constraint  $\mathcal{C}_{\text{hard}}$  defined in (4.2), since it leads to a discrete optimization problem with a factorial number of feasible solutions. To circumvent this issue, we propose a relaxation of the one-to-many assignment problem in the next section.

## 4.3 Optimization algorithm

### 4.3.1 Relaxation

To deal with the nonconvexity of the alignment problem in Equation (4.5), we rely on two main ideas. Firstly, we relax the binary constraint into the unitary interval, so that  $P$  becomes

<sup>1</sup>It is not a distance in the theoretical sense. For brevity, we will use the term “distance” with an abuse of terminology.

a *soft*-assignment matrix belonging to the set

$$\mathcal{C}_{\text{soft}} = \left\{ P \in \mathbb{R}^{|V_1| \times |V_2|} : \begin{array}{l} (\forall i, \forall j) P_{ij} \in [0, 1] \\ (\forall i) \sum_j P_{ij} \in [1, k_{\max}] \\ (\forall j) \sum_i P_{ij} = 1 \end{array} \right\}. \quad (4.6)$$

Secondly, we enforce the relaxed constraints implicitly using the Dykstra operator

$$\mathcal{A}_\tau : \mathbb{R}^{|V_1| \times |V_2|} \rightarrow \mathcal{C}_{\text{soft}}, \quad (4.7)$$

which transforms a rectangular matrix into a soft-assignment matrix, as explained in Section 4.3.2. This operator can be injected into the cost function to remove all the constraints, thus yielding the new unconstrained optimization problem

$$\underset{\tilde{P} \in \mathbb{R}^{|V_1| \times |V_2|}}{\text{minimize}} \quad \mathcal{W}_2^2(v^{\mathcal{G}_1}, v^{\mathcal{G}_2}_{\mathcal{A}_\tau(\tilde{P})}). \quad (4.8)$$

However, problem (4.8) remains highly nonconvex, and this may cause gradient descent to converge towards a local minimum with high probability. Hence, we define below the Dykstra operator  $\mathcal{A}_\tau(\tilde{P})$  that will allow us to devise a stochastic formulation which can be efficiently solved with a variant of gradient descent integrating Bayesian exploration in the optimization process. This will help finding better local minima than solving 4.8 with gradient descent.

### 4.3.2 Dykstra operator

The Dykstra operator uses an iterative algorithm to find a least square projection of a point to any finite intersection of convex sets [16]. In our case, given a rectangular matrix  $\tilde{P}$  and a small constant  $\tau > 0$ , the Dykstra operator will normalise the rows and columns of  $\exp(\tilde{P}/\tau)$  to obtain a one-to-many assignment matrix, where a node in the smaller graph is matched to one or more (but at most  $k_{\max}$ ) nodes in the larger graph. It is defined as

$$\mathcal{A}_\tau(\tilde{P}) = \underset{P \in \mathcal{C}_{\text{soft}}}{\text{argmax}} \left[ \langle P, \tilde{P} \rangle - \tau \sum_{ij} P_{ij} \log(P_{ij}) \right]. \quad (4.9)$$

This operator can be efficiently computed by the Dykstra algorithm [27] with Bregman projections [10]. Indeed, Problem (4.9) can be written as a Kullback-Leibler (KL) projection [14]

$$\mathcal{A}_\tau(\tilde{P}) = \underset{P \in \mathcal{C}^{(0)} \cap \mathcal{C}^{(1)}}{\text{argmin}} \quad \text{KL}(P | \exp(\tilde{P}/\tau)), \quad (4.10)$$

with

$$\begin{aligned} \mathcal{C}^{(0)} &= \{ \Xi \in \mathbb{R}_+^{|V_1| \times |V_2|} \mid \Xi \mathbb{1}_{|V_2|} \in [1, k_{\max}]^{|V_1|} \}, \\ \mathcal{C}^{(1)} &= \{ \Xi \in \mathbb{R}_+^{|V_1| \times |V_2|} \mid \Xi^\top \mathbb{1}_{|V_1|} = \mathbb{1}_{|V_2|} \}. \end{aligned} \quad (4.11)$$

The Dykstra algorithm [27] starts by initializing

$$P^{[0]} = \exp(\tilde{P}/\tau) \quad \text{and} \quad Q^{[0]} = Q^{[-1]} = \mathbb{1}_{|V_1| \times |V_2|}, \quad (4.12)$$

and then iterates for every  $t = 0, 1, \dots$

$$P^{[t+1]} = \mathcal{P}_{\mathcal{C}^{(t \bmod 2)}}^{\text{KL}}(P^{[t]} \odot Q^{[t-1]}), \quad (4.13)$$

$$Q^{[t+1]} = \frac{Q^{[t-1]} \odot P^{[t]}}{P^{[t+1]}}, \quad (4.14)$$

where all operations are entry-wise.<sup>2</sup> The KL projections are defined, for every  $\Xi \in \mathbb{R}_+^{|V_1| \times |V_2|}$ , as follows

$$\mathcal{P}_{\mathcal{C}^{(0)}}^{\text{KL}}(\Xi) = \text{diag} \left( \left[ \frac{\max\{1, \min\{\sum_j \Xi_{ij}, k_{\max}\}\}}{\sum_j \Xi_{ij}} \right]_i \right) \Xi \quad (4.15)$$

$$\mathcal{P}_{\mathcal{C}^{(1)}}^{\text{KL}}(\Xi) = \Xi \text{diag} \left( \left[ \frac{1}{\sum_i \Xi_{ij}} \right]_j \right). \quad (4.16)$$

Intuitively, projections (4.15) and (4.16) project the solution to sets  $\mathcal{C}^{(0)}$  and  $\mathcal{C}^{(1)}$ , respectively. The Dykstra algorithm alternates between these projections, ensuring the final solution is found in the intersection of the two sets. In the limit  $\tau \rightarrow 0$ , the operator  $\mathcal{A}_\tau$  yields a one-to-many assignment matrix. It is also differentiable [69], and can be thus used in a cost function optimized by gradient descent, as we will see in Section 4.3.3.

### Connection to the Sinkhorn operator

In the special case where the two graphs have the same size  $|V_1| = |V_2| = |V|$ , the condition in (4.3) leads to  $k_{\max} = 1$ , and thus  $\mathcal{C}_{\text{soft}}$  reduces to the space of doubly-stochastic matrices. This describes the scenario that was studied in detail in the last Chapter. We now see that the relaxed constraints introduced in this Chapter are consistent with the constraints used for the original GOT formulation. Furthermore, in such a case, the Dykstra operator reverts to a Sinkhorn operator [20, 38, 75, 84]. Recall that given a square matrix  $\tilde{P}$  and a small constant  $\tau > 0$ , the Sinkhorn operator [103] normalizes the rows and columns of  $\exp(\tilde{P}/\tau)$  so as to obtain a doubly stochastic matrix. Formally, it is defined as

$$\mathcal{S}_\tau(\tilde{P}) = \underset{P \in \mathcal{C}_{\text{doubly}}}{\text{argmax}} \left[ \langle P, \tilde{P} \rangle - \tau \sum_{ij} P_{ij} \log(P_{ij}) \right], \quad (4.17)$$

<sup>2</sup> $\odot$  denotes the entry-wise (Hadamard) product of matrices.

where  $\mathcal{C}_{\text{doubly}}$  is the set of doubly stochastic matrices

$$\mathcal{C}_{\text{doubly}} = \left\{ P \in \mathbb{R}^{|V| \times |V|} : \begin{array}{l} (\forall i, \forall j) P_{ij} \in [0, 1] \\ (\forall i) \sum_j P_{ij} = 1 \\ (\forall j) \sum_i P_{ij} = 1 \end{array} \right\}. \quad (4.18)$$

It is well known that the above operator can be computed with the following iterations

$$\begin{aligned} P^{[0]} &= \exp(\tilde{P}/\tau) \\ L^{[t]} &= \text{diag}(P^{[t]} \mathbb{1}_{|V|})^{-1} \\ R^{[t]} &= \text{diag}(\mathbb{1}_{|V|}^\top L^{[t]} P^{[t]})^{-1} \\ P^{[t+1]} &= L^{[t]} P^{[t]} R^{[t]}. \end{aligned} \quad (4.19)$$

In the limit  $\tau \rightarrow 0$ , the operator  $S_\tau$  yields a permutation matrix [75]. It is also differentiable [69], and can be thus used in a cost function optimized by gradient descent, as we will see in Section 4.3.3.

### 4.3.3 Stochastic formulation

With help of the Dykstra operator, the cost function in Problem (4.8) is differentiable, and can thus be optimized by gradient descent. However, the nonconvex nature of the problem may cause gradient descent to converge towards a local minimum. Instead of directly solving Problem (4.8), we propose to optimize the expectation w.r.t. the parameters  $\theta$  of some distribution  $q_\theta$ , yielding

$$\underset{\theta}{\text{minimize}} \mathbb{E}_{\tilde{P} \sim q_\theta} \left\{ \mathcal{W}_2^2(v^{\mathcal{G}_1}, v_{\mathcal{A}_\tau(\tilde{P})}^{\mathcal{G}_2}) \right\}. \quad (4.20)$$

The optimization of the expectation w.r.t. the parameters  $\theta$  aims at shaping the distribution  $q_\theta$  so as to put all its mass on a minimizer of the original cost function, thus integrating the use of Bayesian exploration in the optimization process, possibly helping the algorithm to find better local minima.

A standard choice for  $q_\theta$  in continuous optimization is the multivariate normal distribution, leading to  $\theta = (\eta, \sigma)$  with  $\eta$  and  $\sigma$  being  $|V_1| \times |V_2|$  matrices. By leveraging the reparameterization trick [62, 33], which boils down to setting

$$\tilde{P}_{ij} = \eta_{ij} + \sigma_{ij} \epsilon_{ij} \quad \text{with} \quad \epsilon_{ij} \sim \mathcal{N}(0, 1). \quad (4.21)$$

The problem of Equation (4.20) can thus be reformulated as

$$\underset{\eta, \sigma}{\text{minimize}} \mathcal{J}(\eta, \sigma) := \mathbb{E}_{\epsilon \sim q_{\text{unit}}} \left\{ \mathcal{W}_2^2(v^{\mathcal{G}_1}, v_{\mathcal{A}_\tau(\eta + \sigma \odot \epsilon)}^{\mathcal{G}_2}) \right\}, \quad (4.22)$$

where  $q_{\text{unit}} = \prod_{i,j} \mathcal{N}(0, 1)$  denotes the multivariate normal distribution with zero mean and unitary variance. The advantage of this reformulation is that the gradient of the above stochas-

---

**Algorithm 2** Approximate solution to Problem (4.5).

---

- 1: **Input:** Graphs  $\mathcal{G}_1$  and  $\mathcal{G}_2$
- 2: **Input:** Sampling  $S \in \mathbb{N}$ , step size  $\gamma > 0$ , and  $\tau > 0$
- 3: **Input:** Random initialization of matrices  $\eta_0$  and  $\sigma_0$
- 4: **for**  $t = 0, 1, \dots$  **do**
- 5:     Draw samples  $\{\epsilon_t^{(s)}\}_{1 \leq s \leq S}$  from  $q_{\text{unit}}$
- 6:     Approximate the cost function with

$$\mathcal{J}_t(\eta_t, \sigma_t) = \frac{1}{S} \sum_{s=1}^S \mathcal{W}_2^2(v^{\mathcal{G}_1}, v_{\mathcal{A}_\tau(\eta_t + \sigma_t \odot \epsilon_s)}^{\mathcal{G}_2})$$

- 7:      $g_t \leftarrow$  gradient of  $\mathcal{J}_t$  evaluated at  $(\eta_t, \sigma_t)$
  - 8:      $(\eta_{t+1}, \sigma_{t+1}) \leftarrow$  update of  $(\eta_t, \sigma_t)$  using  $g_t$
  - 9: **end for**
  - 10: **Output:**  $P = \mathcal{A}_\tau(\eta_*)$
- 

tic function can be approximated by sampling from the parameterless distribution  $q_{\text{unit}}$ , yielding

$$\nabla \mathcal{J}(\eta, \sigma) \approx \sum_{\epsilon \sim q_{\text{unit}}} \nabla \mathcal{W}_2^2(v^{\mathcal{G}_1}, v_{\mathcal{A}_\tau(\eta + \sigma \odot \epsilon)}^{\mathcal{G}_2}). \quad (4.23)$$

The problem can be thus solved by stochastic gradient descent [61]. Our approach is summarized in Algorithm 2.

Under mild assumptions, the algorithm converges almost surely to a critical point, which is not guaranteed to be the global minimum, as the problem is nonconvex. The computational complexity of our naive implementation is  $O(N^3)$  per iteration, due to the matrix square-root operation, but faster options exist to approximate this operation [66]. Moreover, the computation of pseudo-inverses can be avoided by adding a small diagonal shift to the Laplacian matrices and directly computing the inverse matrices, which is orders of magnitude faster.

## 4.4 Experiments

We now analyse the performance of our new algorithm in two parts. Firstly, we assess the performance achieved by our approach for graph alignment and community detection in structured graphs of different sizes, testing the preservation of both local and global graph properties. We investigate the influence of distance on alignment recovery and compare to methods using different definitions of graph distance for graph alignment. Secondly, we extend our analysis to graph classification, where we compare our approach with several state-of-the-art methods.

Prior to running experiments, we determine the algorithmic parameters  $\tau$  (in the Dykstra operator) and  $\gamma$  (step size in SGD) with grid search, while  $S$  (sampling size) is fixed empirically.

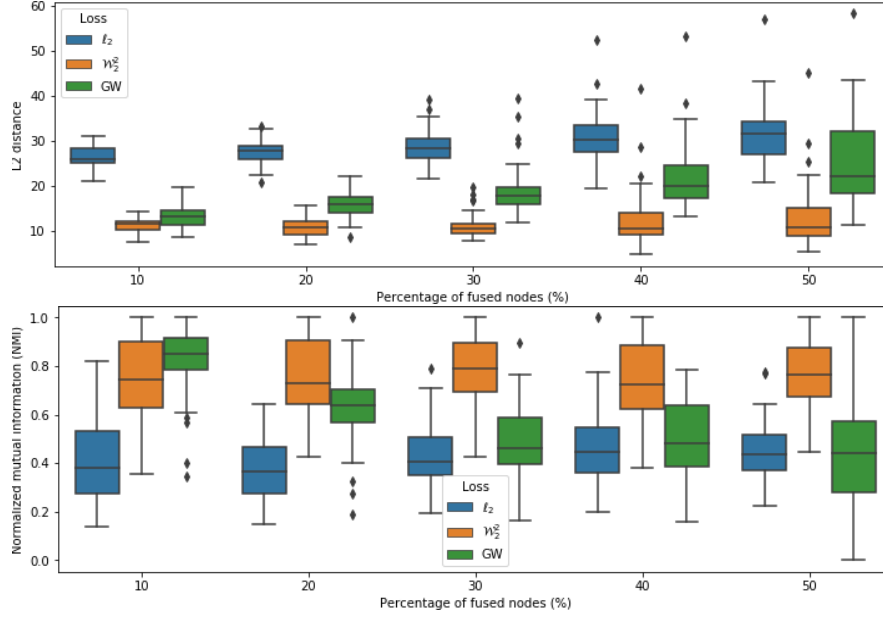


Figure 4.2 – Alignment and detection of communities in structured graphs, performed on distorted stochastic block model graphs as a function of the percentage of fused nodes. The first plot shows the  $\ell_2$  distance between aligned graphs (closer to 0 the better), while the second one shows the community detection performance using spectral clustering in terms of Normalized Mutual Information (NMI closer to 1 the better).

In all experiments, we set  $\tau = 3$ ,  $\gamma = 1$  and  $S = 10$ . We set the maximal number of Dykstra iterations to 20, and we run stochastic gradient descent for 1000 iterations. As our algorithm seems to be robust to different initialisations, we used random initialization in all our experiments. The algorithm was implemented in PyTorch with AMSGrad method [91].

#### 4.4.1 Graph alignment and community detection

In this section, we test our proposed approach for graph alignment and recovery of communities in structured graphs. Namely, apart from the direct comparison of two graphs matrices that shows the recovery of local changes, we evaluate the preservation of global properties by comparing the clustering of nodes into communities. We consider two experimental settings. In the first one (Figure 4.2), we generate a stochastic block model graph  $\mathcal{G}_2$  with 24 nodes and 4 communities. The graph  $\mathcal{G}_1$  is a noisy version of  $\mathcal{G}_2$  constructed by repeatedly fusing two randomly selected connected nodes into one, until a target percentage of nodes is merged. We then generate a random permutation to change the order of the nodes in graph  $\mathcal{G}_1$ .

In the second experimental setting (Figure 4.3), the graph  $\mathcal{G}_2$  is again generated as a stochastic block model with four communities. For each  $\mathcal{G}_2$ , six graphs  $\mathcal{G}_1$  are created as random instances of stochastic block model graphs with the same number of communities, but with a different number of vertices and edges. Apart from the number of communities, there is no direct

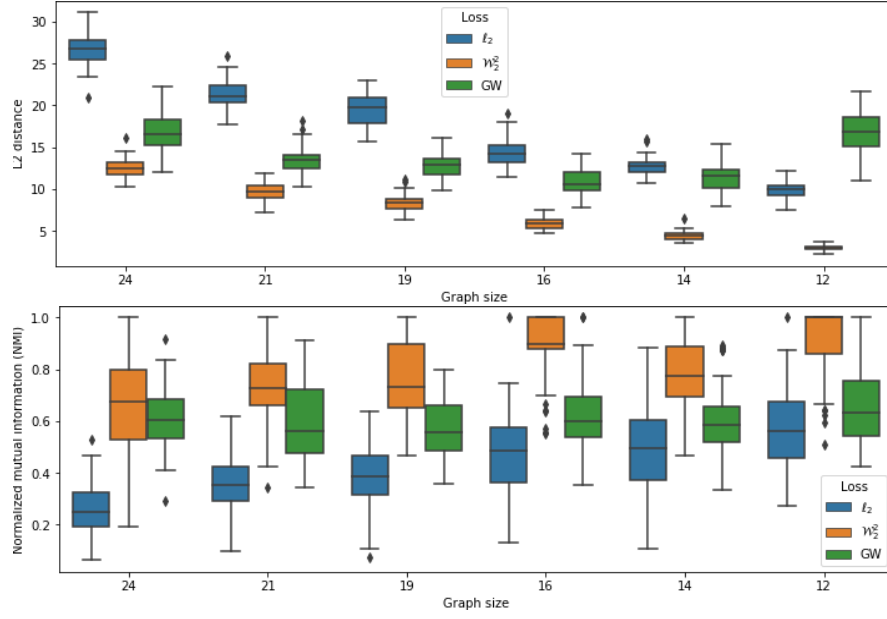


Figure 4.3 – Alignment and detection of communities in structured graphs, performed on random instances of the stochastic block model as a function of the graph size. The first plot shows the  $\ell_2$  distance between aligned graphs (closer to 0 the better), while the second one shows the community detection performance using spectral clustering in terms of Normalized Mutual Information (NMI closer to 1 the better).

connection between  $\mathcal{G}_1$  and  $\mathcal{G}_2$ .

We investigate the influence of a distance metric on alignment recovery. We compare three different methods for graph alignment, namely the proposed method ( $\mathcal{W}_2^2$ ) based on the Wasserstein distance between graphs, the proposed stochastic algorithm with the Euclidean distance ( $\ell_2$ ) defined as  $\|L_1 - PL_2P^\top\|^2$ , and the state-of-the-art Gromov-Wasserstein distance [87] for graphs (GW), using the Euclidean distance between shortest path matrices, as proposed in [112]. We repeat each experiment 50 times, after adjusting parameters for all compared methods, and show the results in Figures 4.2 and 4.3.

In both experiments, we evaluate the alignment quality by computing the  $\ell_2$  distance between aligned graphs. We further evaluate the structure recovery of the community-based models with the community recovery normalised mutual information (NMI). Namely, the  $\ell_2$  distance considers changes in the local structure independently, while community recovery measures the preservation of global structure. For that reason, they can be seen as complementary measures, taking into account both independent and local, as well as structural and global properties. To measure the NMI, we use spectral clustering to cluster the nodes in each graph after the alignment estimation. A good alignment should detect and preserve communities, keeping the nodes in their original clusters and close to their original neighbours, even when the exact neighbours are not recovered. We evaluate the quality of community recovery with

Table 4.1 – Accuracy scores for 1-NN classification results on graph dataset.

Dataset	GA	IPFP	RRWM	GW	NetLSD	$\ell_2$	$\mathcal{W}_2^2$
IMDB-B	56.72	55.22	61.19	54.54	53.73	54.54	<b>63.63</b>
PTC	50.75	52.24	49.25	56.71	52.23	47.76	<b>61.19</b>

NMI between the clusters in the original graph and the recovered clusters.

As shown in Figure 4.2, the proposed approach manages to capture the structural information and outperform methods based on different distance metrics. This suggests that our proposed Wasserstein distance between graphs has a strong positive significance in determining a good graph alignment. Furthermore, while all observed methods seem to be slightly negatively affected by a larger percentage of fused nodes in terms of the  $\ell_2$  norm, our method seems have consistently good performance in terms of community recovery NMI, suggesting that the global structural properties are well captured even under large perturbations.

In Figure 4.3, we can see a similar trend in comparison of our method with methods based on other observed distances, reinforcing once again the positive effect of our Wasserstein graph distance in recovering an appropriate alignment. Furthermore, we observe an increase in performance in terms of NMI for both  $\ell_2$  and  $\mathcal{W}_2^2$ . The emergence of this phenomenon, despite the growing size difference between compared graphs, suggests that our assignment matrix has the ability to fuse nodes into meaningful groups, forming well defined clusters.

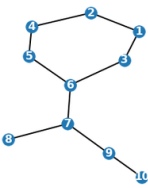
### 4.4.2 Graph classification

We now tackle the task of graph classification on two different datasets: PTC [63] and IMDB-B [121]. We randomly sample 100 graphs from each dataset. The graphs have a different number of nodes and edges. We use  $\mathcal{W}_2^2$  to align graphs and compute pairwise graph distances, followed by a simple non-parametric 1-NN classification algorithm to classify graphs with help of the computed graph distances. We compare the classification performance with methods where the same 1-NN classifier is used with different state-of-the-art methods for graph alignment: GW [87, 112], GA [42], IPFP [65], RRWM [19], NetLSD [110], and the proposed stochastic algorithm with the Euclidean distance ( $\ell_2$ ) instead of the Wasserstein distance in Eq. (4.20). We present the accuracy scores in Table 4.1, where the classification with the proposed  $\mathcal{W}_2^2$  clearly outperforms the other methods in terms of general accuracy. Furthermore, we analyse the performance of  $\mathcal{W}_2^2$ , GW and  $\ell_2$  on several examples from the two datasets.

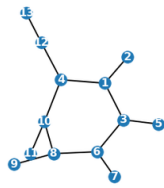
#### PTC dataset

PTC dataset contains the molecular structure of the NTP dataset. Figure 4.4 presents a set of graph examples from two different classes, namely graphs of class 0 and 1 for respectively the

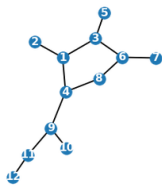




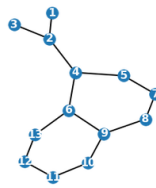
$\mathcal{G}_1$



$\mathcal{G}_2$




$\mathcal{G}_3$

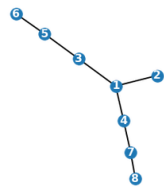


$\mathcal{G}_4$

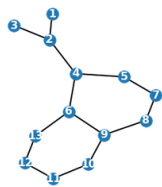
	Class 0		Class 1	
	$\mathcal{D}(\mathcal{G}_1, \mathcal{G}_2)$	<	$\mathcal{D}(\mathcal{G}_1, \mathcal{G}_3)$	> $\mathcal{D}(\mathcal{G}_3, \mathcal{G}_4)$
$\ell_2$ -norm	0.0058		0.0096	0.0093
GW	1.2417		0.7866	2.2204
$\mathcal{W}_2^2$	<b>0.9301</b>		<b>0.9465</b>	<b>0.5457</b>



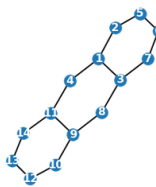
$\mathcal{G}_1$



$\mathcal{G}_2$

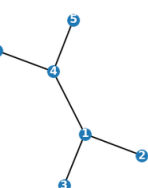


$\mathcal{G}_3$

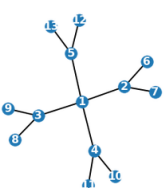


$\mathcal{G}_4$

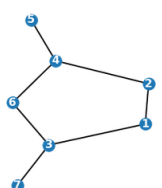
	Class 0		Class 1	
	$\mathcal{D}(\mathcal{G}_1, \mathcal{G}_2)$	<	$\mathcal{D}(\mathcal{G}_1, \mathcal{G}_3)$	> $\mathcal{D}(\mathcal{G}_3, \mathcal{G}_4)$
$\ell_2$ -norm	0.0476		0.0002	0.0067
GW	<b>2.7187</b>		<b>3.5081</b>	<b>0.9897</b>
$\mathcal{W}_2^2$	<b>1.0891</b>		<b>2.0754</b>	<b>0.8202</b>



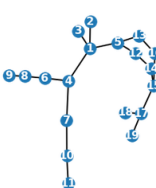
$\mathcal{G}_1$



$\mathcal{G}_2$



$\mathcal{G}_3$



$\mathcal{G}_4$

	Class 0		Class 1	
	$\mathcal{D}(\mathcal{G}_1, \mathcal{G}_2)$	<	$\mathcal{D}(\mathcal{G}_1, \mathcal{G}_3)$	> $\mathcal{D}(\mathcal{G}_3, \mathcal{G}_4)$
$\ell_2$ -norm	0.1580		0.0023	0.0050
GW	1.4493		0.9217	8.5444
$\mathcal{W}_2^2$	1.2069		0.2332	1.7364

Figure 4.4 – PTC dataset with two classes. Each row presents a set of graph examples, from the left to the right:  $\mathcal{G}_1$ ,  $\mathcal{G}_2$ ,  $\mathcal{G}_3$  and  $\mathcal{G}_4$ .  $\mathcal{G}_1$  and  $\mathcal{G}_2$  belong to class 0.  $\mathcal{G}_3$  and  $\mathcal{G}_4$  belong to class 1. Each table provides two kinds of distances: an intra ( $\mathcal{D}(\mathcal{G}_1, \mathcal{G}_2)$  and  $\mathcal{D}(\mathcal{G}_3, \mathcal{G}_4)$ ) and an inter class distance ( $\mathcal{D}(\mathcal{G}_1, \mathcal{G}_3)$ ). We evaluate three different methods in terms of distances in order to classify the graphs (e.g.,  $\mathcal{D}(\mathcal{G}_1, \mathcal{G}_2) \leq \mathcal{D}(\mathcal{G}_1, \mathcal{G}_3)$  or  $\mathcal{D}(\mathcal{G}_3, \mathcal{G}_4) \leq \mathcal{D}(\mathcal{G}_1, \mathcal{G}_3)$ ).

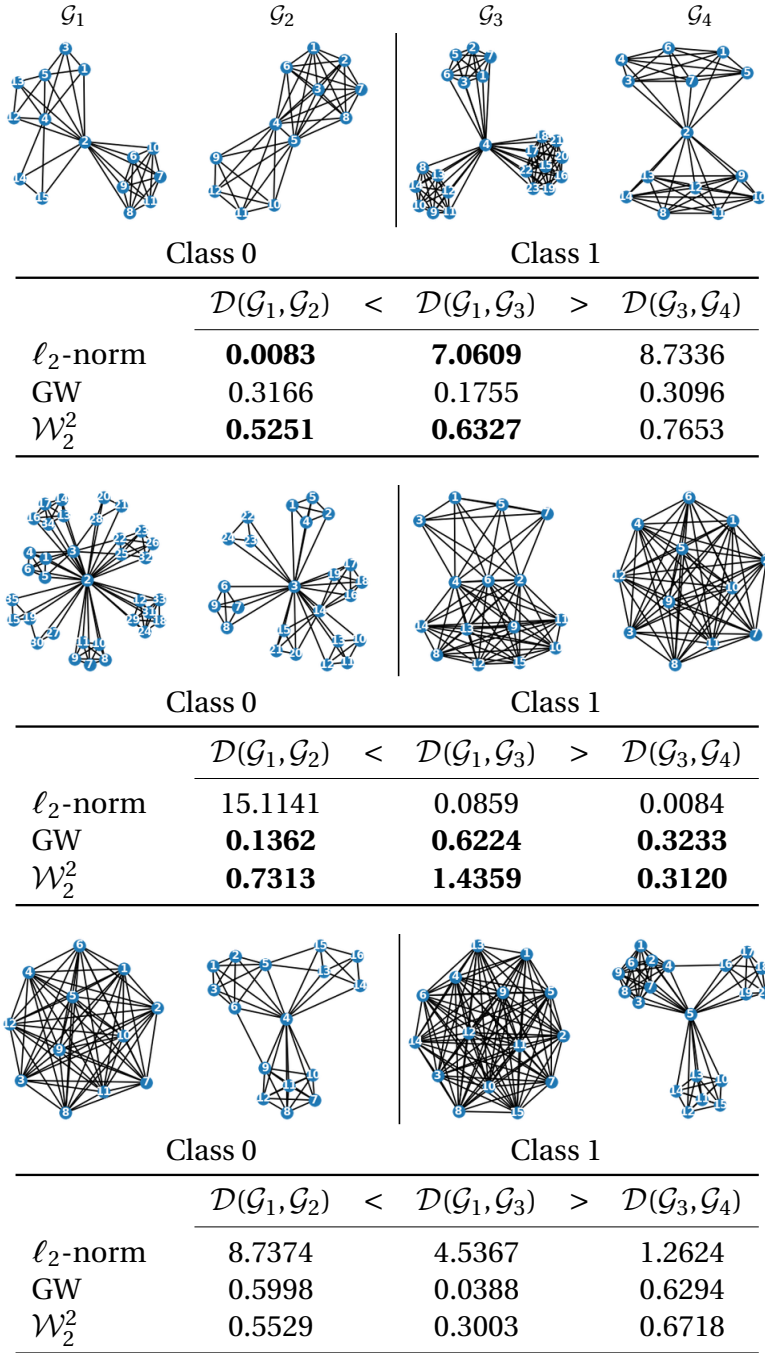


Figure 4.5 – IMDB-B dataset with two classes. Each row presents a set of graph examples, from the left to the right:  $\mathcal{G}_1$ ,  $\mathcal{G}_2$ ,  $\mathcal{G}_3$  and  $\mathcal{G}_4$ .  $\mathcal{G}_1$  and  $\mathcal{G}_2$  belong to class 0.  $\mathcal{G}_3$  and  $\mathcal{G}_4$  belong to class 1. Each table provides two kinds of distances: an intra ( $\mathcal{D}(\mathcal{G}_1, \mathcal{G}_2)$  and  $\mathcal{D}(\mathcal{G}_3, \mathcal{G}_4)$ ) and an inter class distance ( $\mathcal{D}(\mathcal{G}_1, \mathcal{G}_3)$ ). We evaluate three different methods in terms of distances in order to classify the graphs (e.g.,  $\mathcal{D}(\mathcal{G}_1, \mathcal{G}_2) \leq \mathcal{D}(\mathcal{G}_1, \mathcal{G}_3)$  or  $\mathcal{D}(\mathcal{G}_3, \mathcal{G}_4) \leq \mathcal{D}(\mathcal{G}_1, \mathcal{G}_3)$ ).

first, and last two columns. In the first example (first row),  $\mathcal{W}_2^2$  outperforms both  $\ell_2$  and  $GW$  in separating the two classes. The distinguishing feature between  $\mathcal{G}_1$  and  $\mathcal{G}_3$  is the number of nodes that forms the ring, which has been captured by  $\mathcal{W}_2^2$ , thanks to the soft permutation applied to the larger graph  $\mathcal{G}_3$  ( $|V_3| > |V_2|$ ).

The second example (second row) shows in a very intuitive way how  $\mathcal{W}_2^2$  and  $GW$  are able to capture structural similarities in graphs, even when those largely vary in size. This is especially clear when comparing the almost two times larger  $\mathcal{W}_2^2(\mathcal{G}_1, \mathcal{G}_3)$  and  $\mathcal{W}_2^2(\mathcal{G}_1, \mathcal{G}_2)$ , with structurally very similar  $\mathcal{G}_1$  and  $\mathcal{G}_2$ , and an easy-to-imagine assignment of one node in the graph  $\mathcal{G}_1$  to several nodes in the graph  $\mathcal{G}_2$ . However, it is not always as simple to understand the similarities. The third row shows an example in which all the three methods fail to find structural similarities between graphs in the same class.

### IMDB-B dataset

IMDB-B dataset contains two classes: Comedy and science-fiction movies, with several examples shown in Figure 4.5. The striking difference between example 2 (second row) and example 3 (third row) shows that, while taking into account the global graph structure can be crucial in distinguishing some samples (second row), it remains a challenging dataset with very similar graphs often belonging to different clusters (third row). Namely, graphs in the second row which belong to the same clusters share clear structural similarities, and both the  $GW$  distance and our proposed  $\mathcal{W}_2^2$  distance manage to capture these shared properties. On the other hand, graphs in the third row which belong to the same clusters do not have very clear shared properties, and even share many more structural similarities with some of the graphs in the other cluster. This possibly explains the low accuracy across all examined methods.

Finally, example 1 (first row) shows the high flexibility of the assignment matrix proposed in our algorithm, where the one-to-many assignment is able to detect that graph  $\mathcal{G}_1$  is very close to a graph with 2 communities, even if it has 3 communities. This combination of putting emphasis on structural information, and allowing for flexibility with the one-to-many assignment might be the reason why  $\mathcal{W}_2^2$  still manages to outperform the other investigated methods.

## 4.5 Conclusion

In this chapter, we have proposed a new method to align graphs of different sizes. Equipped with an optimal transport approach to compute the distance between two smooth graph distributions associated to each graph, we have formulated a new one-to-many alignment problem to find a soft assignment matrix that minimizes the “mass” transportation from a fixed distribution to a permuted and merged distribution. The resulting nonconvex optimization problem is solved efficiently with a novel stochastic gradient descent algorithm. It allows

us to align and compare graphs of different size, and it outputs a structurally meaningful distance. We have shown the performance of the proposed method in the context of graph alignment and graph classification. Our results show that the proposed algorithm outperforms state-of-the-art alignment methods for structured graphs of different sizes.

# 5 fGOT: filter Graph distances using Optimal Transport

## 5.1 Introduction

The graph optimal transport distance introduced in the last chapters has shown to capture well the global structure of graphs. While that is a vital property for a graph distance in a large number of applications, it might not always be the optimal way to compare graphs. For example, if we are interested in how certain phenomena will spread in our graphs, the global graph structure is important, but a more natural comparison would be based directly on the graphs' diffusion properties. In fact, several graph distances based on heat diffusion have been proposed. The authors in [46] propose a direct comparison of graph heat diffusion matrices, but stay limited to graphs of the same size. A spectral method proposed by [110] circumvents this problem by comparing heat kernel traces, but it still does not address the graph alignment problem. Closer to our work, a fast heat kernel distance based on optimal transport has recently been proposed in [8]. However, this distance compares graph through available signals, whereas our method uses the representation of graphs through signal distributions and therefore does not assume any signal availability. The above methods are limited only to heat diffusion models, while more general distances could often be of interest.

In this chapter, we propose the filter graph distance, a graph comparison method based on optimal transport, which compares graphs through signals generated with graph filters. Filtered signals present a generic model for graph signals, capturing the structural properties of the underlying graph through the definition of a graph filter. They offer a high level of flexibility in modelling the relationship between data and the underlying graph, and are capable of capturing a wide range of structural graph properties, including local characteristics, global structure, and any combination of spectral graph properties. We use probabilistic distributions of such signals to formulate a graph comparison problem using optimal transport, generalising the graph Wasserstein distance proposed in Chapter 3. In order to render the method computationally competitive, we formulate an efficient approximation to the alignment problem defined through filter graph distances. The approximation removes the largest computational difficulties present in the original formulation of the alignment problem and permits the

utilisation of much faster algorithms.

We then propose a simple and efficient solution to the approximated alignment problem using mirror gradient descent. As the problem remains non-convex, we propose a novel stochastic algorithm based on mirror gradient descent which can be used efficiently for any filter graph distance. We finish by showing the benefits of our method in experimental settings. Filter graph distances show better performance than the standard optimal transport distances on simple graph alignment tasks, confirming the benefits of the flexibility introduced with our method. Finally, the proposed stochastic algorithm achieves significantly better results in community detection in structured graphs, when compared to the simple mirror gradient descent. This suggests that our algorithm successfully takes into account the non-convexity of the alignment problem.

## 5.2 Filter Graph Alignment with Optimal Transport

### 5.2.1 Filter graph distance

In this section we define the filter graph distance, a generalisation of the graph optimal transport (GOT) distance [84] which has the ability to emphasize specific spectral properties of the graph, such as high or low frequencies, local or global graph phenomena. We prioritise these properties through filtered graph signals, exploiting the specific graph information and finally comparing graphs through filtered signal distributions. Clearly, the choice of a graph filter is crucial in driving the resulting distance.

Specifically, given two aligned graphs  $\mathcal{G}_1$  and  $\mathcal{G}_2$  with Laplacian matrices  $L_1$  and  $L_2$ , we consider the shape of their respective filtered signals. As described in 1.13, these graph signals follow a Gaussian distribution defined through the filter:

$$\mathbf{v}^{\mathcal{G}_1, g} = \mathcal{N}(0, g^2(L_1)) \quad (5.1)$$

$$\mathbf{v}^{\mathcal{G}_2, g} = \mathcal{N}(0, g^2(L_2)), \quad (5.2)$$

where  $g(\cdot)$  is a graph filter function as defined in 1.10. The choice of the filter  $g(\cdot)$  will drive different spectral characteristics of the graph signals. For example, a heat kernel  $g(L) = e^{-\tau L}$  will model a graph based on the nature of the spread of heat through the graph, usually emphasizing global graph properties. On the other hand, a high pass filter like  $g(L) = L^2$  will take more local phenomena into account, prioritising high graph frequencies.

Our graph distance will compare graphs through their respective signal distributions. The choice of filter is therefore critical in driving the resulting distance. In the example of the heat kernel, the distance will model differences in the spread of heat through the graphs, emphasizing global dissimilarities with the additional interpretation of comparing graphs based on how they propagate heat through their nodes.

Following the GOT framework introduced in Chapter 3, we can now define the filter graph distance (fGOT) as the Wasserstein distance between the distributions  $\nu^{\mathcal{G}_1, g}$  and  $\nu^{\mathcal{G}_2, g}$  of graphs  $\mathcal{G}_1$  and  $\mathcal{G}_2$ :

$$\mathcal{W}_2^2(\nu^{\mathcal{G}_1, g}, \nu^{\mathcal{G}_2, g}) = \text{Tr}(g^2(L_1) + g^2(L_2)) - 2 \text{Tr}\left(\sqrt{g(L_1) g^2(L_2) g(L_1)}\right). \quad (5.3)$$

The fGOT distance builds on the idea of graph comparison through smooth signal distributions, and extends it to the probabilistic distributions of signals filtered through graphs. This directly compares the nature of filter responses on those graphs, putting an emphasis on the specific properties of filtered signals. In particular, the smooth graph optimal transport (GOT) distance can be seen as a special case of fGOT, with the low pass graph filter equal to  $g(L) = \sqrt{L^\dagger}$ .

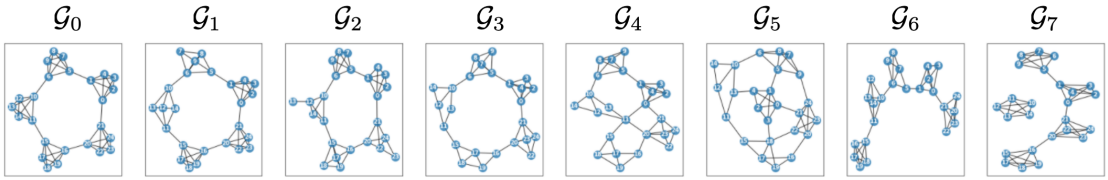


Figure 5.1 – The enumerated list of random graphs which are compared in Figure 5.2. Graphs are generated by randomly perturbing the original graph  $\mathcal{G}_0$  with adding and removing random edges. The number of nodes has not been changed.

Figure 5.2 shows the flexibility of fGOT in prioritising different phenomena in the definition of our distance. We compare the set of graphs presented in Figure 5.1 based on different filter distances, and sort them based on their distance to  $\mathcal{G}_0$ . There are several differences in the ordering of graphs with different filter distances. For instance,  $\mathcal{G}_2$  and  $\mathcal{G}_3$  exchange places with  $\mathcal{G}_6$  as the filter becomes high pass, from rows 1 and 2 to rows 4 and 5. The reason for this is that smooth filters capture the rupture in the global ring structure of  $\mathcal{G}_6$ , while the higher pass filters focus on local changes which are more present in  $\mathcal{G}_2$  and  $\mathcal{G}_3$ . The same example shows the strong impact of temperature on the behaviour of the heat kernel filter. Namely,  $g_1(L) = e^{-5L}$  has a very large reach, which makes it the smoothest filter we observed, while the very limited reach of  $g_3(L) = e^{-0.5L}$  makes it more focused on local changes, and positions it between the low and high pass filters in this example.

Finally, we note that a filter graph distance can be especially useful in systematic comparison of graphs with disconnected components. Namely, traditional distances comparing Laplacian or Adjacency matrices directly will not take any connectedness information into account. More meaningful distances like GOT [84] or FGW [112] are not designed for disconnected graphs, and need to use heuristic solutions in order to compare them. On the other hand, a filter graph distance can easily control the importance of connectedness by adjusting its spectral properties. An example of this can be seen in row 3 of Figure 5.2 with  $g_3(L) = e^{-0.5L}$  and graph  $\mathcal{G}_7$ , where the connectedness information is taken into account without being given too much importance.

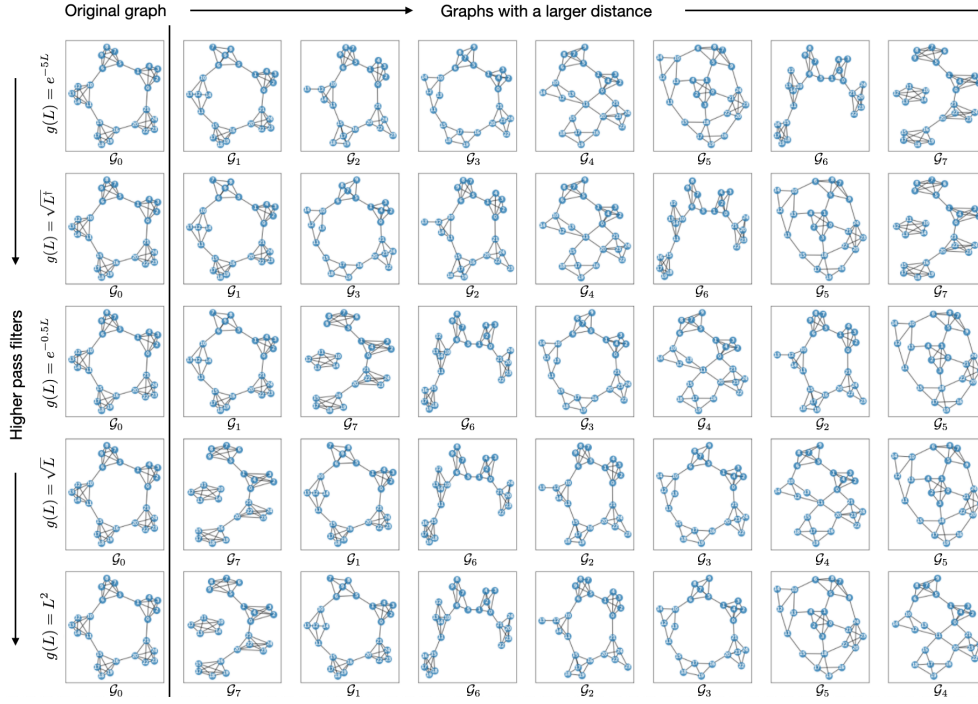


Figure 5.2 – Random graphs of Figure 5.1 sorted based on increasing distance to  $\mathcal{G}_0$ , for different graph filter distances. Each row is a different filter distance, starting with very smooth filters and going towards more high pass filter distances. Namely, the filters are given by  $g_1(L) = e^{-0.5L}$ ,  $g_2(L) = \sqrt{L}$ ,  $g_3(L) = e^{-0.5L}$ ,  $g_4(L) = \sqrt{L}$ ,  $g_5(L) = L^2$ .

### 5.2.2 Scalable alignment approximation

The distance introduced in (5.3) requires to know a correspondence between vertices of the two graphs. This requirement is often not realistic, and a proper alignment between graph vertices needs to be recovered in order to make graphs comparable. In order to take different alignments into account, we define a probability distribution for the permuted version of  $\mathcal{G}_2$  as

$$v_P^{\mathcal{G}_2, g} = \mathcal{N}(0, g^2(PL_2P^T)). \quad (5.4)$$

We now search for the optimal alignment  $P$  by minimising the graph Wasserstein distance between  $\mathcal{G}_1$  and different permutations of  $\mathcal{G}_2$ :

$$\underset{P \in \mathcal{C}_{\text{perm}}}{\text{minimize}} \quad \text{Tr}(g^2(L_1) + g^2(PL_2P^T)) - 2\text{Tr}\left(\sqrt{g(L_1)g^2(PL_2P^T)g(L_1)}\right), \quad (5.5)$$

where  $\mathcal{C}_{\text{perm}}$  denotes the set of permutation matrices:

$$\mathcal{C}_{\text{perm}} = \left\{ P \in \mathbb{R}^{N \times N} : \begin{array}{l} (\forall i, \forall j) P_{ij} \in \{0, 1\} \\ (\forall i) \sum_j P_{ij} = 1 \\ (\forall j) \sum_i P_{ij} = 1 \end{array} \right\}. \quad (5.6)$$



While the alignment algorithms presented in the last two chapters could successfully be used to solve (5.5), the time complexity of such algorithms is prohibitive for large graphs or large sets of graphs. For this reason, we present an alternative cost in order to enable faster optimisation algorithms.

**Lemma 1.** *Let  $P \in \mathcal{C}_{\text{perm}}$  be a permutation matrix and  $L \in \mathbb{R}^{N \times N}$  an arbitrary graph Laplacian matrix. Then  $g(PLP^T) = Pg(L)P^T$  for any graph filter  $g(\cdot)$  defined as in 1.10.*

*Proof.* We write the eigenvalue decomposition of  $L$  as  $L = U\Lambda U^T$ . Notice that the eigenvalue decomposition of  $PLP^T$  is then  $(PU)\Lambda(PU)^T$ . Namely, with  $u_i$  an eigenvector of  $L$  corresponding to the eigenvalue  $\lambda_i$ , we have:

$$PLP^T Pu_i = PLu_i = \lambda_i Pu_i. \quad (5.7)$$

Therefore,

$$g(PLP^T) = g((PU)\Lambda(PU)^T) = (PU)\hat{G}(PU)^T = PU\hat{G}U^T P^T = Pg(L)P^T, \quad (5.8)$$

where  $\hat{G}$  is given by 1.11. □

**Lemma 2.** *Let  $\mathcal{G}_1$  and  $\mathcal{G}_2$  be two graphs with their respective Laplacian matrices  $L_1 \in \mathbb{R}^{N \times N}$  and  $L_2 \in \mathbb{R}^{N \times N}$ . Then, for any graph filter  $g(\cdot)$  defined as in 1.10, and  $P \in \mathcal{C}_{\text{perm}}$ :*

$$\mathcal{W}_2^2(v^{\mathcal{G}_1, g}, v_{P^T}^{\mathcal{G}_2, g}) \leq \text{Tr}(g^2(L_1) + g^2(L_2)) - 2\langle g(L_1)Pg(L_2), P \rangle. \quad (5.9)$$

If  $L_1$  and  $L_2$  further represent permuted versions of the same graph:

$$\min_{P \in \mathcal{C}_{\text{perm}}} \mathcal{W}_2^2(v^{\mathcal{G}_1, g}, v_{P^T}^{\mathcal{G}_2, g}) = \min_{P \in \mathcal{C}_{\text{perm}}} \text{Tr}(g^2(L_1) + g^2(L_2)) - 2\langle g(L_1)Pg(L_2), P \rangle = 0 \quad (5.10)$$

*Proof.*

$$\underbrace{\mathcal{W}_2^2(v^{\mathcal{G}_1, g}, v_{P^T}^{\mathcal{G}_2, g})}_{\mathcal{J}} = \text{Tr}(g^2(L_1)) + \text{Tr}(Pg^2(L_2)P^T) - 2\text{Tr}\left(\sqrt{g(L_1)Pg^2(L_2)P^T g(L_1)}\right) \quad (5.11)$$

$$= \text{Tr}(g^2(L_1)) + \text{Tr}(g^2(L_2)P^T P) - 2\sum_i \lambda_i \left( \sqrt{g(L_1)Pg^2(L_2)P^T g(L_1)} \right) \quad (5.12)$$

$$= \text{Tr}(g^2(L_1)) + \text{Tr}(g^2(L_2)P^T P) - 2\sum_i \sqrt{\lambda_i (g(L_1)Pg^2(L_2)P^T g(L_1))} \quad (5.13)$$

$$= \text{Tr}(g^2(L_1)) + \text{Tr}(g^2(L_2)) - 2\sum_i \sqrt{\lambda_i (g(L_1)Pg(L_2)P^T Pg(L_2)P^T g(L_1))} \quad (5.14)$$

with  $\lambda_i(A)$  denoting the eigenvalues of a matrix  $A$ . Here, (5.12) follows because for a symmetric positive-semidefinite matrix  $\text{Tr}(A) = \sum_i \lambda_i(A)$  and (5.14) because  $PP^T = P^T P = I_N$ . Now with

$\sigma_i(A)$  denoting the singular values of a matrix  $A$ , the equality becomes:

$$\mathcal{J} = \text{Tr}(g^2(L_1)) + \text{Tr}(g^2(L_2)) - 2 \sum_i \sqrt{\lambda_i \left( (g(L_1)Pg(L_2)P^T)(g(L_1)Pg(L_2)P^T)^T \right)} \quad (5.15)$$

$$= \text{Tr}(g^2(L_1)) + \text{Tr}(g^2(L_2)) - \sum_i \left( \sigma_i(g(L_1)Pg(L_2)P^T) + \sigma_i \left( (g(L_1)Pg(L_2)P^T)^T \right) \right), \quad (5.16)$$

because  $\lambda_i(AA^T) = \sigma_i^2(A) = \sigma_i^2(A^T)$ . Finally, (5.17) follows from  $\sum_i \sigma_i(A+B) \leq \sum_i (\sigma_i(A) + \sigma_i(B))$ , and (5.18) because for a symmetric positive-definite matrix  $C$ ,  $\sigma_i(C) = \lambda_i(C)$ :

$$\mathcal{J} \leq \text{Tr}(g^2(L_1)) + \text{Tr}(g^2(L_2)) - \sum_i \left( \sigma_i \left( g(L_1)Pg(L_2)P^T + (g(L_1)Pg(L_2)P^T)^T \right) \right) \quad (5.17)$$

$$= \text{Tr}(g^2(L_1)) + \text{Tr}(g^2(L_2)) - \sum_i \lambda_i \left( g(L_1)Pg(L_2)P^T + (g(L_1)Pg(L_2)P^T)^T \right) \quad (5.18)$$

$$= \text{Tr}(g^2(L_1) + g^2(L_2)) - \text{Tr} \left( g(L_1)Pg(L_2)P^T + (g(L_1)Pg(L_2)P^T)^T \right) \quad (5.19)$$

$$= \text{Tr}(g^2(L_1) + g^2(L_2)) - 2 \text{Tr}(g(L_1)Pg(L_2)P^T) \quad (5.20)$$

$$= \text{Tr}(g^2(L_1) + g^2(L_2)) - 2 \langle g(L_1)Pg(L_2), P \rangle \quad (5.21)$$

□

The result of Lemma 2 opens the door to optimising over a surrogate cost function, which we denote as

$$\widetilde{\mathcal{W}}_2^2(v^{\mathcal{G}_1, g}, v_P^{\mathcal{G}_2, g}) = \text{Tr}(g^2(L_1) + g^2(L_2)) - 2 \langle g(L_1)Pg(L_2), P \rangle, \quad (5.22)$$

and which will allow for a faster algorithm to solve the resulting problem, removing the need to find a matrix square root in every iteration. We will now propose an algorithm to efficiently optimise over 5.22, searching for the optimal alignment  $P$ .

### 5.3 FGOT algorithm

As we have seen in the previous Chapters, the main difficulty in solving an alignment problem arises from the constraint that  $P$  is a permutation matrix, since it leads to a discrete optimisation problem with a factorial number of feasible solutions. We relax the constraints and propose to circumvent this issue through an implicit constraint reformulation.

We look for a soft assignment between the graphs, namely, a matrix  $P \in \mathbb{R}^{|V_1| \times |V_2|}$  satisfying the constraints

$$\mathcal{C}_{\text{fuzzy}} = \left\{ P \in \mathbb{R}^{|V_1| \times |V_2|} : \begin{array}{l} (\forall i, \forall j) P_{ij} \geq 0 \\ (\forall i) \sum_j P_{ij} = \frac{1}{|V_1|} \\ (\forall j) \sum_i P_{ij} = \frac{1}{|V_2|} \end{array} \right\}. \quad (5.23)$$

Note that  $P \in \mathcal{C}_{\text{perm}}$  implies  $P \in \mathcal{C}_{\text{fuzzy}}$  up to a rescaling factor of  $\sqrt{|V_1||V_2|}$ . Using the surrogate

cost function (5.22), we can now write the optimisation problem (5.5) as

$$\underset{P \in \mathcal{C}_{\text{fuzzy}}}{\text{minimize}} \quad \widetilde{\mathcal{W}}_2^2(v^{\mathcal{G}_1, g}, v_P^{\mathcal{G}_2, g}). \quad (5.24)$$

To solve the non-convex optimisation problem in Equation (5.24), one can use the projected mirror gradient descent, where the projection is computed according to the Kullback-Leibler (KL) metric. Adding the entropic regularisation  $\epsilon H(P)$  to the problem yields the following iterative algorithm:

$$P_{t+1} = \mathcal{P}_{\mathcal{C}_{\text{fuzzy}}}^{\text{KL}} \left( P_t \odot \exp \left( -\alpha \left( \nabla \widetilde{\mathcal{W}}_2^2(v^{\mathcal{G}_1, g}, v_{P_t}^{\mathcal{G}_2, g}) - \epsilon \nabla H(P_t) \right) \right) \right) \quad (5.25)$$

where the Kullback-Leibler projection  $\mathcal{P}_{\mathcal{C}_{\text{fuzzy}}}^{\text{KL}}(\cdot)$  can be computed in function of the Sinkhorn operator  $\mathcal{S}_\tau$ :

$$\mathcal{P}_{\mathcal{C}_{\text{fuzzy}}}^{\text{KL}}(P) = \mathcal{S}_\tau(-\tau \log P). \quad (5.26)$$

However, the optimisation problem in Equation (5.24) remains non-convex, and thus very susceptible to converging to locally optimal solutions. To address this issue, we propose a novel stochastic mirror gradient optimisation algorithm. We formulate the optimisation problem (5.24) with an implicit constraint, taking into account the KL projection of the (non-stochastic) mirror gradient descent algorithm. For simplicity, we denote the KL projection to  $\mathcal{C}_{\text{fuzzy}}$  with  $\mathcal{B}(\cdot)$ , and the optimisation problem becomes:

$$\mathcal{B}(P) := \mathcal{P}_{\mathcal{C}_{\text{fuzzy}}}^{\text{KL}}(P) \quad (5.27)$$

$$\underset{P \in \mathbb{R}^{|V_1| \times |V_2|}}{\text{minimize}} \quad \widetilde{\mathcal{W}}_2^2(v^{\mathcal{G}_1, g}, v_{\mathcal{B}(P)}^{\mathcal{G}_2, g}). \quad (5.28)$$

The above cost function is differentiable [69], and can thus be optimised with a gradient based optimisation algorithm.

To avoid converging towards a local minimum, we aim at optimising the expectation of  $\widetilde{\mathcal{W}}_2^2(v^{\mathcal{G}_1, g}, v_{\mathcal{B}(P)}^{\mathcal{G}_2, g})$  w.r.t. the parameters  $\theta$  of some distribution  $p_\theta$ , that is

$$\underset{\theta}{\text{minimize}} \quad \underbrace{\mathbb{E}_{P \sim p_\theta} \{ \widetilde{\mathcal{W}}_2^2(v^{\mathcal{G}_1, g}, v_{\mathcal{B}(P)}^{\mathcal{G}_2, g}) \}}_{U(\theta)}. \quad (5.29)$$

The optimization of  $U$  aims at shaping the distribution  $p_\theta$  so as to put all its mass on a minimizer  $P^*$  of  $\widetilde{\mathcal{W}}_2^2$ , thus integrating the use of Bayesian exploration in the optimization process. This approach is commonly used in many areas of stochastic search, such as evolution strategies [47, 116, 94].

**Algorithm 3** Approximate solution to Problem (5.5)

---

- 1: **Input:** Graphs  $\mathcal{G}_1$  and  $\mathcal{G}_2$
- 2: **Input:** Sampling  $S \in \mathbb{N}$ , step size  $\alpha > 0$  and  $\tau > 0$
- 3: **Input:** Constant initialization of matrix  $\eta_0$  and random initialization of  $\sigma_0$
- 4: **for**  $t = 0, 1, \dots$  **do**
- 5:     Draw the samples  $\epsilon_t^{(n)}$  from  $\mathcal{N}(0, I)$ .
- 6:     Estimate the gradient  $(g_t^\eta, g_t^\sigma)$  with the first-order stochastic approximations:

$$g_t^\eta \approx \frac{1}{N} \sum_{n=1}^N \nabla \widetilde{\mathcal{W}}_2^2(v^{\mathcal{G}_1, g}, v_{\mathcal{B}(\eta_t + \sigma_t \circ \epsilon_t^{(n)})}^{\mathcal{G}_2, g})$$

$$g_t^\sigma \approx \frac{1}{N} \sum_{n=1}^N \epsilon_t^{(n)} \circ \nabla \widetilde{\mathcal{W}}_2^2(v^{\mathcal{G}_1, g}, v_{\mathcal{B}(\eta_t + \sigma_t \circ \epsilon_t^{(n)})}^{\mathcal{G}_2, g})$$

- 7:     Update  $(\eta_t, \sigma_t)$  using  $(g_t^\eta, g_t^\sigma)$ :

$$\eta_{t+1} = \eta_t - \alpha_t \sigma_t^2 \circ g_t^\eta$$

$$d_t = \frac{\alpha_t}{2} \sigma_t^2 \circ g_t^\sigma$$

$$\sigma_{t+1} = \sqrt{\sigma_t^2 + d_t^2} - d_t$$

- 8: **end for**
  - 9: **Output:**  $P = \mathcal{B}(\eta_*)$
- 

The updates of variational gradient descent can be equivalently rewritten as

$$\theta_{t+1} = \operatorname{argmin}_{\theta} \theta^\top \nabla U(\theta_t) + \frac{1}{2\alpha_t} \|\theta - \theta_t\|^2. \quad (5.30)$$

The Euclidean distance can be replaced by a Bregman divergence  $\mathbb{D}$ , resulting in the mirror-descent update:

$$\theta_{t+1} = \operatorname{argmin}_{\theta} \theta^\top \nabla U(\theta_t) + \frac{1}{\alpha_t} \mathbb{D}(\theta, \theta_t). \quad (5.31)$$

For exponential-family distributions, the Kullback-Leibler (KL) divergence corresponds to a Bregman divergence (Raskutti and Mukherjee, 2015):

$$\mathbb{D}(\theta, \theta_t) = \mathbb{D}_{\text{KL}}(p_\theta \| p_{\theta_t}) = \mathbb{E}_{P \sim p_\theta} \left\{ \log \frac{p_\theta(P)}{p_{\theta_t}(P)} \right\}. \quad (5.32)$$

The use of KL divergence results in better steps when optimising the parameter of a probability distribution [3].

When  $p_\theta$  is the diagonal Gaussian distribution  $\mathcal{N}(\eta, \text{diag}(\sigma^2))$  with  $\theta = (\eta, \sigma)$ , the KL divergence is given by:

$$\mathbb{D}_{\text{KL}}(p_{\eta, \sigma} \parallel p_{\eta_t, \sigma_t}) = \frac{1}{2} \sum_{d=1}^D \left( \frac{\sigma_d^2}{\sigma_{t,d}^2} + \frac{(\eta_{t,d} - \eta_d)^2}{\sigma_{t,d}^2} - 1 + \log \frac{\sigma_{t,d}^2}{\sigma_d^2} \right). \quad (5.33)$$

Then, the gradients of  $\mathbb{D}_{\text{KL}}$  w.r.t.  $\eta$  and  $\sigma$  are given by

$$\begin{aligned} \frac{\partial}{\partial \eta_d} \mathbb{D}_{\text{KL}}(p_{\eta, \sigma} \parallel p_{\eta_t, \sigma_t}) &= \frac{\eta_d - \eta_{t,d}}{\sigma_{t,d}^2} \\ \frac{\partial}{\partial \sigma_d} \mathbb{D}_{\text{KL}}(p_{\eta, \sigma} \parallel p_{\eta_t, \sigma_t}) &= \frac{\sigma_d}{\sigma_{t,d}^2} - \frac{1}{\sigma_d}, \end{aligned} \quad (5.34)$$

and the mirror-descent update boils down to

$$\begin{aligned} \eta_{t+1,d} &= \eta_{t,d} - \alpha_t \sigma_{t,d}^2 \frac{\partial U(\theta_t)}{\partial \eta_d} \\ \sigma_{t+1,d} &= \sqrt{\sigma_{t,d}^2 + \left( \frac{\alpha_t \sigma_{t,d}^2}{2} \frac{\partial U(\theta_t)}{\partial \sigma_d} \right)^2} - \frac{\alpha_t \sigma_{t,d}^2}{2} \frac{\partial U(\theta_t)}{\partial \sigma_d}. \end{aligned} \quad (5.35)$$

The stochastic mirror gradient descent algorithm for fGOT is summarised in Algorithm 3. Finally, we note that the computational complexity of our algorithms boils down to the computational complexity of matrix multiplications for each iteration of the algorithm. In the worst case, this complexity is  $O(|V_1|^2 |V_2| + |V_2|^2 |V_1|)$ . However, much more efficient algorithms exist for square matrix multiplication ( $O(|V|^{2.38})$ ), and for a variety of special cases, such as sparse matrices. The gain in computational complexity compared to the algorithms presented in previous chapters is significant, as algorithms proposed in Chapter 3 and 4 both perform SVD decomposition in each iteration in order to compute the matrix square root present in the original cost function (5.5). In our work, the stochastic version of Algorithm 5.5 was implemented in PyTorch, using the AMSGrad method [91].

## 5.4 Experimental results

### 5.4.1 Speed comparison on a graph alignment task

In this experiment, we tackle the problem of alignment of unstructured random graphs across different graph sizes. We compare the execution time, as well as the alignment quality of different algorithms and filters. In particular, for each predefined graph size between 10 and 100, we perform 50 repetitions of aligning two random Erdos-Renyi graphs using the following algorithms: Gromov-Wasserstein (GW) [87], with graphs compared based on their shortest path matrices, as proposed in [112]; the GOT algorithm, as proposed in [84]; the mirror

gradient descent (MGD) algorithm for fGOT with the original GOT low pass filter  $g(L) = \sqrt{L^\dagger}$ ; as well as the stochastic MGD and MGD with a high pass filter  $g(L) = L^2$ .

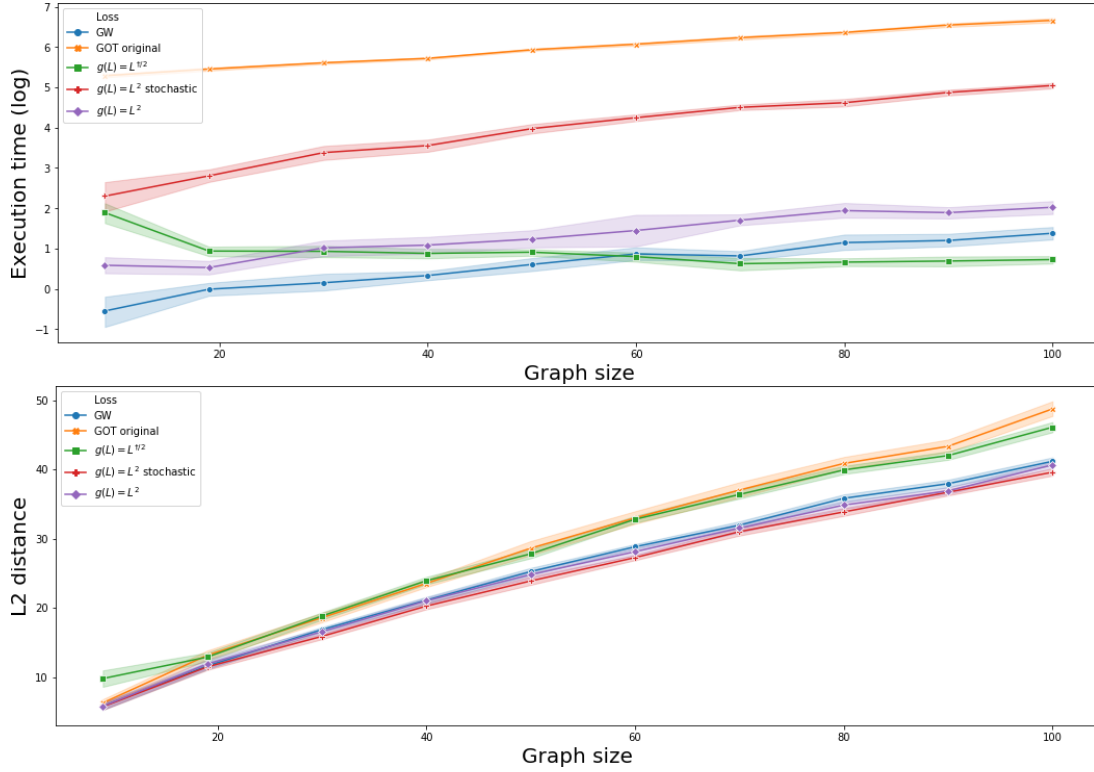


Figure 5.3 – Speed comparison of different algorithms and filters on a task of random graph alignment. The first plot shows the log execution time for each algorithm, while the second one shows the  $\ell_2$  distance between aligned graphs, across different graph sizes.

As the graphs in question have no particular structure, we expect high-pass filters to perform better than low-pass ones. Namely, the  $\ell_2$  norm compares local differences between graphs, which are better captured by high pass filters. Furthermore, as the graphs do not have a particularly strong global structure, the smooth filters do not have the advantage of capturing the global behaviour of graphs. We can see this phenomenon in Figure 5.3, where high pass filters clearly outperform their low-pass counterparts. This reinforces the importance of the right filter choice for each problem, and emphasizes the benefits of the flexibility offered by fGOT.

At the same time, we can observe the significant speed improvement that both of our approximated fGOT algorithms offer compared to the original GOT formulation presented in Chapter 3. The mirror gradient descent algorithm shows competitive performance using all observed filters, with performance differences coming from filter particularities and speed of convergence. In particular, it is worth noting that using the  $g(L) = L^{\dagger/2}$  filter, the proposed approximation renders the computation of the GOT distance comparable to the GW distance speed, circumventing the largest drawback of the GOT framework. On the other hand, the

stochastic mirror gradient descent algorithm still offers a great speed improvement, while keeping the stochastic nature of the algorithm, successfully accounting for the nonconvexity of the alignment problem. This allows the algorithm to capitalise on both speed and accuracy, offering better performance than its non-stochastic counterpart at a faster rate than GOT. In summary, our two proposed algorithms offer a choice between maximising accuracy or speed, while both provide a very competitive trade-off between the two.

#### 5.4.2 Community detection in structured graphs

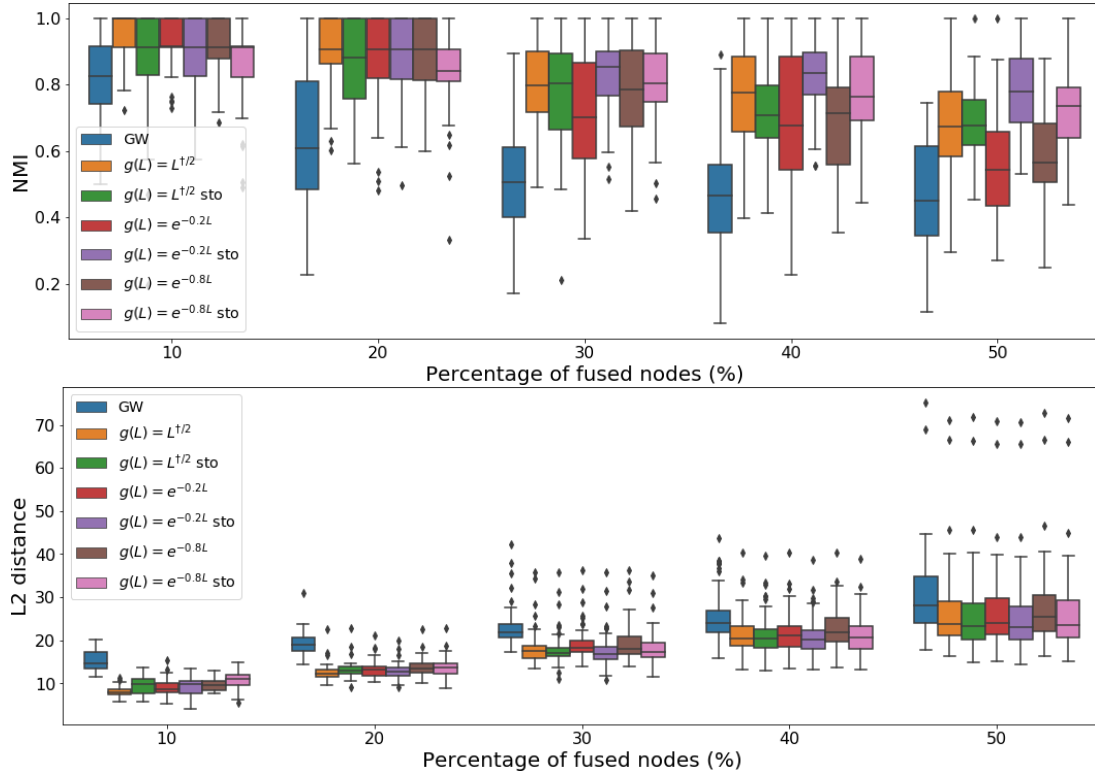


Figure 5.4 – Alignment and community detection performance for distorted stochastic block model graphs as a function of the percentage of fused nodes (Experiment 1). The first plot shows the community detection performance using spectral clustering in terms of Normalized Mutual Information (NMI closer to 1 is better), while the second one shows the  $\ell_2$  distance between aligned graphs (closer to 0 is better).

We test the performance of fGOT on a graph alignment and community detection task in structured graphs. We consider two experimental settings: namely the comparison of a graph with a noisy version of the same graph, and its comparison with a random structured graph. We investigate the influence of the filter model defining the graph distance metric on alignment recovery. Namely, a filter which manages to capture the main properties of these graphs should recognize and align the global graph communities even in these challenging settings, and therefore achieve a better alignment of these structured graphs. In addition to the filter distance, we evaluate the performance of our stochastic mirror algorithm, by showing

two results for each graph filter distance: computed with mirror gradient and stochastic mirror gradient. We use the state-of-the-art Gromov-Wasserstein (GW) distance [87] as baseline, where graphs are compared based on their shortest path matrices, as proposed in [112]. We compare alignment results obtained with three different graph filter models, namely  $g(\sqrt{L^\dagger})$ , which corresponds to the approximated version of the original GOT distance for smooth signals [84], and two heat kernel distances:  $g(L) = e^{-0.2L}$  and  $g(L) = e^{-0.8L}$ .

To evaluate the alignment recovery, we resort to two measures: Normalised mutual information (NMI) of community alignment, and the difference between the aligned graphs in terms of the  $\ell_2$  norm. We estimate the community alignment of these structured graphs by performing spectral clustering in both (aligned) graphs, separating their nodes into 4 communities. The two measures can be considered as complementary, with NMI capturing the recovery of global structure, and  $\ell_2$  considering local differences independently.

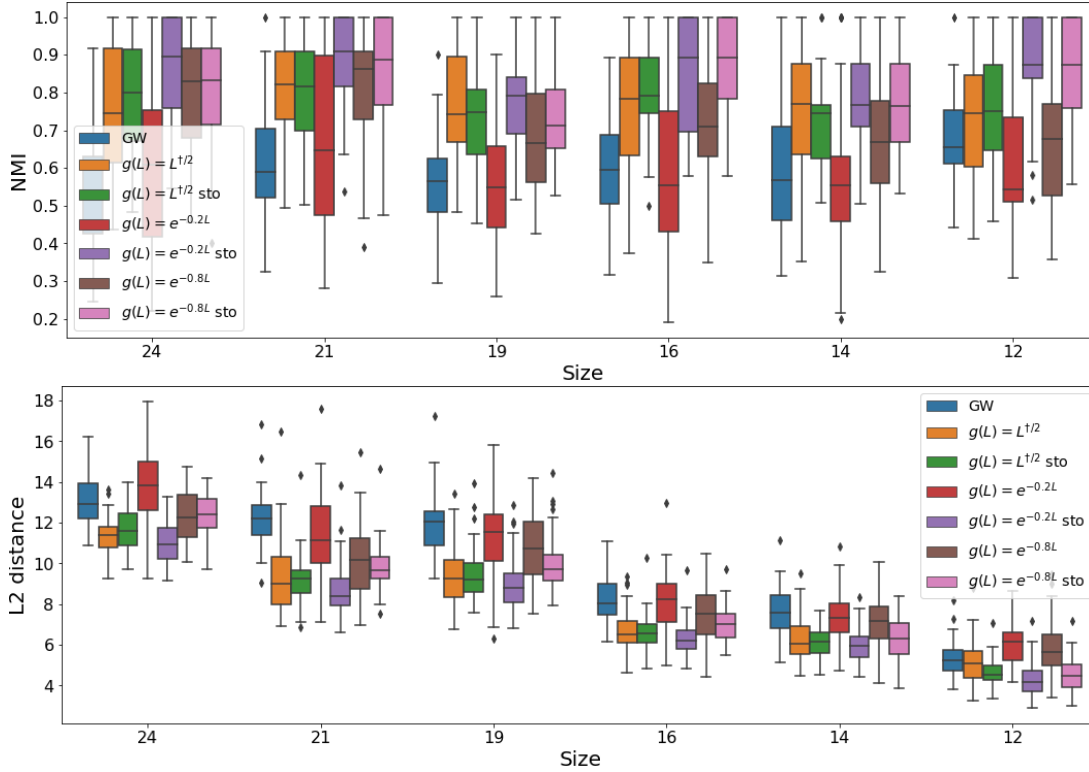


Figure 5.5 – Alignment and community detection performance for random instances of stochastic block model graphs as a function of the graph size (Experiment 2). The first plot shows the community detection performance using spectral clustering in terms of Normalised Mutual Information (NMI closer to 1 is better), while the second one shows the  $\ell_2$  distance between aligned graphs (closer to 0 is better).

In Experiment 1, we generate a stochastic block model graph  $\mathcal{G}_2$  with 24 nodes and 4 communities. The graph  $\mathcal{G}_1$  is constructed as a noisy version of  $\mathcal{G}_2$  by randomly collapsing edges until a target percentage of nodes is fused. Graph  $\mathcal{G}_1$  is then further randomly permuted, in order to



change the order of its nodes. Each experiment has been repeated 50 times with a different  $\mathcal{G}_2$ , after adjusting parameters for all compared methods. The results are given in Figure 5.4.

Then, in Experiment 2, both graphs are generated randomly as instances of stochastic block models with 4 communities. For each  $\mathcal{G}_2$  with size 24, we generate six graphs  $\mathcal{G}_1$  with a different number of edges and vertices (between 12 and 24). While we keep the number of communities equal for a meaningful community detection task, there is no other direct connection between  $\mathcal{G}_1$  and  $\mathcal{G}_2$ . Each experiment has been repeated 50 times with a different  $\mathcal{G}_2$ , after adjusting parameters for all compared methods. The results are given in Figure 5.5.

Both experiments demonstrate the superior performance of our stochastic algorithm compared to the simple mirror descent algorithm, especially in the case of heat kernel models. The improvements are especially visible in the community detection tasks, as the settings become more challenging. Furthermore, the stochastic version of the heat kernel distances outperforms both the original GOT distance, as well as the GW in both experiments. This shows that, even when observing only smooth filters, the flexibility offered by considering different filter models can affect our results substantially and bring considerable benefits. Finally, the results in terms of the  $l_2$  distance are consistent with the NMI, suggesting that the stochastic algorithm improves the alignment significantly both in terms of global structure recovery and more local alignment properties.

## 5.5 Conclusion

We proposed fGOT, a flexible generalisation of the graph optimal transport framework that uses graph filter models to encode specific graph properties. We exploit the generative model of filtered signals, which allows for a representation of graphs through the distribution of filtered signals. It thus permits an efficient comparison through the Wasserstein distance of these distributions. In order to provide a more scalable algorithm, we formulate an approximation to the generic filter optimal transport cost, circumventing the computation of its most expensive parts. We propose an efficient stochastic algorithm based on Bayesian exploration, which adapts mirror gradient descent to this challenging non-convex problem. We show the performance of our method in the context of graph alignment and community detection. Experimental results show that both our proposed algorithms offer a superior performance in terms of speed, when compared to the original GOT algorithm. Furthermore, our novel stochastic algorithm offers superior performance to its non-stochastic counterpart in terms of accuracy and alignment quality, reaching the best trade-off between speed and accuracy. Finally, experiments on both unstructured and structured graphs show that the filter graph distance brings valuable flexibility, with problem-adjusted filters outperforming the state-of-the-art in different simple tasks.



## 6 Conclusion

### 6.1 Summary

In this thesis, we have investigated the problem of representing graphs through data distributions. We have shown how such a new probabilistic representation of a graph is both very successful in capturing important graph properties, and versatile in its applications, enabling the exploitation of various probabilistic tools to address problems in graph analysis.

We first tackled the problem of learning multiple graphs from structured data mixtures. We showed how the probabilistic representation of graphs through filtered signals can be used to create a generative model for a mixture of structured signals. The proposed model extends graph inference to a more general scenario in which the signals are mixed and naturally live on multiple graphs. In our experiments, we demonstrated the advantages of our methods in terms of interpretability and coping with high dimensionality, and evaluate our results in terms of both clustering and graph learning performance. Additionally, we showed that our model offers important findings in a natural application of functional brain network inference. Our model successfully recovered coherent activation patterns and meaningfully structured brain networks from resting state fMRI data, but also provided interesting insights into the organisation of functional brain graphs with respect to the structural connectome.

Next, we employed the graph representation in order to introduce a structurally meaningful distance between graphs. Namely, using a probabilistic representation of graphs through smooth graph signal distributions, we proposed a novel framework based on optimal transport, in order to derive the Wasserstein graph distance, as well as a transportation map for signals between graphs. We have further formulated a new graph alignment problem based on the graph Wasserstein distance that permits to compare graphs while paying attention to their structure, and proposed an efficient stochastic algorithm which accommodates for the nonconvexity of the problem. In our experiments, we have demonstrated the structural benefits that our framework introduces into the graph comparison problem, as well as the efficiency of our graph alignment algorithm. Namely, our graph optimal transport distance outperforms the state-of-the-art methods on both structural graph alignment and graph

classification tasks, and shows interesting results in terms of graph signal prediction.

We then extended our framework to graphs of different sizes, formulating a one-to-many assignment problem, where each node in the smaller graph is matched with one or more nodes in the larger graph. We have proposed an optimisation algorithm using a novel Dykstra operator, which successfully imposes a (soft) one-to-many structure on the alignment matrix. Our experiments on graph alignment and graph clustering tasks confirm the benefits of our optimal transport framework in the challenging settings where graphs have different sizes, and show the efficiency of our novel algorithm in finding a meaningful assignment between graphs.

Finally, we further extended our framework to consider graph representations through probabilistic distributions of filtered graph signals. We showed how such different representations lead to a large flexibility in our graph Wasserstein distance, and can help to prioritise specific graph properties in distance computation. We have then formulated an approximation to the graph filter distance cost, enabling the usage of more scalable algorithms. We proposed a novel stochastic mirror gradient descent algorithm in order to tackle the nonconvexity of the alignment problem. Our proposed stochastic algorithm leads to a great improvement in recovering a meaningful graph alignment, and our experiments in community detection in structured graphs show the strong benefits of the flexibility introduced through the filter graph distance.

### 6.2 Future directions

While the graph representation used in this thesis proved to contain very meaningful information about the graphs, it is surely not unique. Namely, a Gaussian distribution for signals is not always realistic, and representations through different signal distributions could be attractive in many scenarios. In particular, one where signals are modelled as a sparse combination of graph dictionary atoms [109], could be very useful for example in scenarios in which data is expected to originate and spread from only a few graph nodes. It would be very interesting to see the impact of such an intuitively different graph representation on applications similar to those presented in this thesis.

Next, the Graph Laplacian Mixture Model introduced in Chapter 2 is successful in recovering multiple graphs from a mixture of structured signals. However, there are several extensions that could be made in order to make the model more adapted to various applications. Firstly, the graphical model assumes independence between cluster participations of different signals, which is often not the case in applications with temporal dynamics, where a cluster of a signal might depend on the previous history. Incorporating such information by utilizing hidden Markov models is an interesting research avenue for future work. Another interesting extension would be to impose similarities or differences on graphs inferred through GLMM, reducing the difficulty of the learning problem. Such properties could be enforced in a structurally meaningful way by using our graph Wasserstein distance introduced in Chapter 3.

An interesting extension to the graph optimal transport framework introduced in Chapters 3, 4 and 5 would be to incorporate available signal values into the distance in order to further inform the graph alignment recovery. A natural way to do so in our framework would be to use the signal transportation function to measure the distance between signals on different graphs. Namely, transporting signals to the same graph would make them directly comparable and emphasize information meaningful for the graph alignment. In addition, the signal transportation function could be of great interest in domain adaptation, and such a direction could be a promising research avenue.

Finally, in fields such as neuroscience, biology and machine learning, there are several possible applications of our work and algorithms proposed in this thesis. One such application, explored in Appendix A, could be explored with graph comparison with optimal transport. It would be interesting to see whether additional insights on the organisation of functional graph networks could be inferred using the graph Wasserstein distance. Another such application lays in the emerging and active field of graph generation. A very important step in constructing a graph generative model is efficient and meaningful graph comparison. It would be very interesting to see how our structurally meaningful graph distance affects such models, the sort of challenges it would introduce, and the benefits its structure-preserving properties would bring.



# A Appendix: Inference of Brain Networks

Here we give additional information about the inference of Multiple functional Brain Networks using GLMM, part of which has been presented as an experiment in Section 2.5.3. After the introduction, we describe the materials and methods in detail, present our experiments and results, and finish with a discussion of our findings.

## A.1 Introduction

Functional magnetic resonance imaging (fMRI) has provided extensive possibilities for probing the functional architecture of the human brain. Spontaneous fluctuations of blood-oxygenation-level dependent (BOLD) signal has been particularly interesting since the discovery that brain regions can be synchronised in activity despite the absence of task or external stimuli [15]. Several networks of coherent activity between remote areas have long been identified, including the visual area, auditory cortex, language, as well as attention-related areas. One particular network associated with resting-state (RS) studies is the default mode network (DMN), which shows reduced activity when subjects perform an externally oriented task [44], and contrastingly becomes more engaged when a subject undergoes internal mentation [4].

In these experiments, we use our GLMM framework proposed in Chapter 2 to estimate multiple functional brain networks, or *metastates*, by building upon an emerging field of graph learning [25]. From this point onwards, we will use the terms *metastates* and *networks* interchangeably to pertain to repeating functional spatial patterns in the brain. Starting from the whole-brain time-series of parcellated fMRI data, GLMM infers different metastates directly from the data, and at the same time, extracts the corresponding underlying graph that gives rise to these networks.

## **A.2 Materials and Methods**

### **A.2.1 Data and Preprocessing**

The functional data were downloaded from the publicly available Human Connectome Project (HCP) database, WU-Minn Consortium. MRI acquisition protocols of the HCP are extensively described and discussed in [41]. We used 50 subjects consisting of 4 sessions of RS scans (1200 volumes each, a total of 4800 frames), and 2 sessions each of task fMRI data (working memory, relational memory, social, language, emotion, and motor tasks). Functional volumes underwent the standard preprocessing steps [111]. All functional images were realigned to the mean functional volume for each participant. The realigned volumes were registered to the structural T1 data using rigid-body registration (SPM12, <https://www.fil.ion.ucl.ac.uk>), and were de-trended (i.e., constant, linear, quadratic) to remove signal drifts. Then, the images were smoothed using a Gaussian kernel with FWHM equal to 6mm. Finally, we used the Automated Anatomical Labeling (AAL, 90 regions) atlas that was resliced to fMRI resolution to parcellate fMRI volumes and compute regionally averaged fMRI signals. Meanwhile, structural data of each subject were downloaded from the HCP and were processed using MRtrix3 (<http://www.mrtrix.org/>). The SC of each subject was generated based on the total number of fibers connecting two regions divided by the volumes of connecting regions. The normalization is done to ensure that the strength of the connection is not biased towards the size of the bundles. The final SC matrix was obtained by averaging all SC matrices of all subjects.

### **A.2.2 Experimental details**

We first demonstrate the powerful use of the tool by capturing distinct networks corresponding to each task epochs of task fMRI data downloaded from the Human Connectome Project (HCP) database. The performance of the method is validated by comparing the extracted probability of occurrence of each metastate to the timing of the task paradigm. We show that the extracted metastates consist of brain areas that are consistent with previously observed regions that are implicated in the corresponding task. We then apply the GLMM to RS fMRI data and obtain the most prevalent networks of spontaneously interacting brain areas.

Furthermore, in addition to understanding the organizational principles of human brain function, recent studies in neuroimaging are also starting to elucidate the role of structure in cognition and behavior [90, 73]. To extract meaningful interpretation of the learned graphs, we show that the estimated graph Laplacian matrices from fMRI reveal indicative similarities with the structural connectome (SC) derived from diffusion-weighted MRI (DW-MRI). We demonstrate that the degree of their similarity captures behaviorally-relevant gradient that is consistent with the previously observed macro-scale organization of the cortex that ranks low-level sensory processing to high-level cognitive ones [90, 71].

We then validated the performance of the proposed framework using task fMRI by demonstrating that the timing of the task paradigms are captured by the averaged  $\gamma$ -values across all



subjects. The hyper-parameters of the model ( $\Delta, \theta$ ) are optimized using Normalized Mutual Information (NMI) using the task paradigm as the reference ground truth. Meanwhile, the number of clusters (K) is optimized using a feedback step where t cannot expect it to correspond to the number of tasks, since the metastates may be formed as a combination of tasks. For that reason, we evaluated the results by iteratively changing K and performing a visual evaluation of the dynamics of  $\gamma$  with respect to the experimental task paradigm. Doing so, we observe that the meaningful clusters consistently appear in a very similar fashion, regardless of the proposed number of clusters K. Furthermore, fixing K in such a way results in better overall accuracy, suggesting that the number of optimal clusters might be different from the number estimated through more traditional methods. We denote this method as *K- $\gamma$  itero-homogeneity of metastates*.

As for RS, which is lacking a ground truth, the number of clusters K was chosen based on the optimized silhouette measure and the consensus clustering procedure [78], which is a resampling-based method for optimal class discovery. Both metrics suggested 3 as the optimal number of clusters. K has eventually been varied according to the procedure mentioned above, in order to capture multiple networks. In practice, changing the optimal number of clusters does not seem to affect the final estimation, instead, it opens the possibility of inferring more networks.

### A.2.3 Comparison of learned functional graphs to brain structure

An important benefit of the GLMM algorithm compared to other clustering methods is the estimation of graph Laplacians (L). Not only this does help in obtaining more meaningful networks or metastates (means), but it also conveys much more information and details about our clusters. From here onwards, we use the term *graphs* to refer to the learned graph Laplacians and *networks* or *metastates* to refer to the functional means or clusters.

We explored the similarity of the learned functional graphs to the structural connectome derived from DW-MRI by using the spectral euclidean difference as a metric. This is done by first decomposing the adjacency matrices (A) of each graph derived from the Laplacians ( $A = D - L$ ) into their constituent eigenvalues and eigenvectors. The eigenspectrum of A are transformed using the procrustes algorithm [60, 43] to match the ordering of the eigenvectors of SC.

The output of the Procrustes transform is the rotation matrix that is used to retrieve the transformed (functional) eigenvalues. In particular, given the rotation matrix Z generated by the transformation, the rotated eigenvalues are computed as follows:

$$\hat{\lambda} = ZEW, \quad (\text{A.1})$$

where W is the weighted adjacency matrix of the functional network taken into consideration and E is the vector of its original eigenvalues.

Considering all the functional networks inferred, regardless of the task, we computed the euclidean spectral distance between the eigenvalues of the SC and the transformed graphs. This metric reflects the degree of similarity between the learned graphs in each task and the anatomy. Thus, each network will have a certain score of distance that indicates how that function couples with structure: the smaller the metric, the closer the estimated graph to the structure. We performed bootstrap repetitions runs to extract networks that show statistically different values. We then performed a Neurosynth meta-analysis similar to the one implemented by Margulies *et.al.* [71] and Preti *et.al.* [90], where topic terms from the database [<https://neurosynth.org/>] were matched with the derived euclidean spectral difference of the functional graphs and the SC. The active areas of the estimated networks were correlated with the database present in literature, thereby capturing a number of possible behavioral domains. The same 24 topics adopted these two studies were considered. This behavioral gradient has been sorted according to a weighted mean of the resulting z-statistics.

### A.3 Results

#### A.3.1 Estimated network time-courses are consistent with the timing of the task paradigms

The proposed framework extracts consistent patterns of brain activity in each of the considered tasks, and along with it is the estimation of their likelihood to occur at each time-point. Fig. A.1 displays the estimated time-courses for six selected tasks overlaid with the task paradigm which are distinguished in colors. Metastate 2 and 3 of the Language task capture time-points when subjects undergo the *Story* and *Math* epochs, respectively while Metastate 1 corresponds to the resting epoch of the task. Meanwhile, Metastate 3 consistently captures the RS epoch in the *Social* task. However, the algorithm does not distinguish the metastates corresponding to *Mental* and *Random* phases since the likelihood estimation of Metastate 2 corresponded to both task epochs. On the other hand, Metastate 3 captures the transition between the rest and task epochs. The same observation can be made in the other tasks, in particular, the motor, relational, and working memory. In each of these tasks, some metastates capture transitions between epochs and uniquely identifies whether it is a transition between rest and task epoch or a transition between two different task epochs *i.e.*, Metastate 4 in Relational Memory captures transitions between *Relation* and *Match*, while Metastate 2 captures transitions between rest and any of the two tasks.

#### A.3.2 GLMM captures spatial networks corresponding to each task consistent with their known established neurophysiological descriptions

Fig. A.2(A) and (B) displays an example set of results that we obtained when running the GLMM algorithm to the Language and Emotion tasks, respectively. The RS epochs of both tasks reveal metastates that are akin to the DMN network. The regions implicated in Metastate

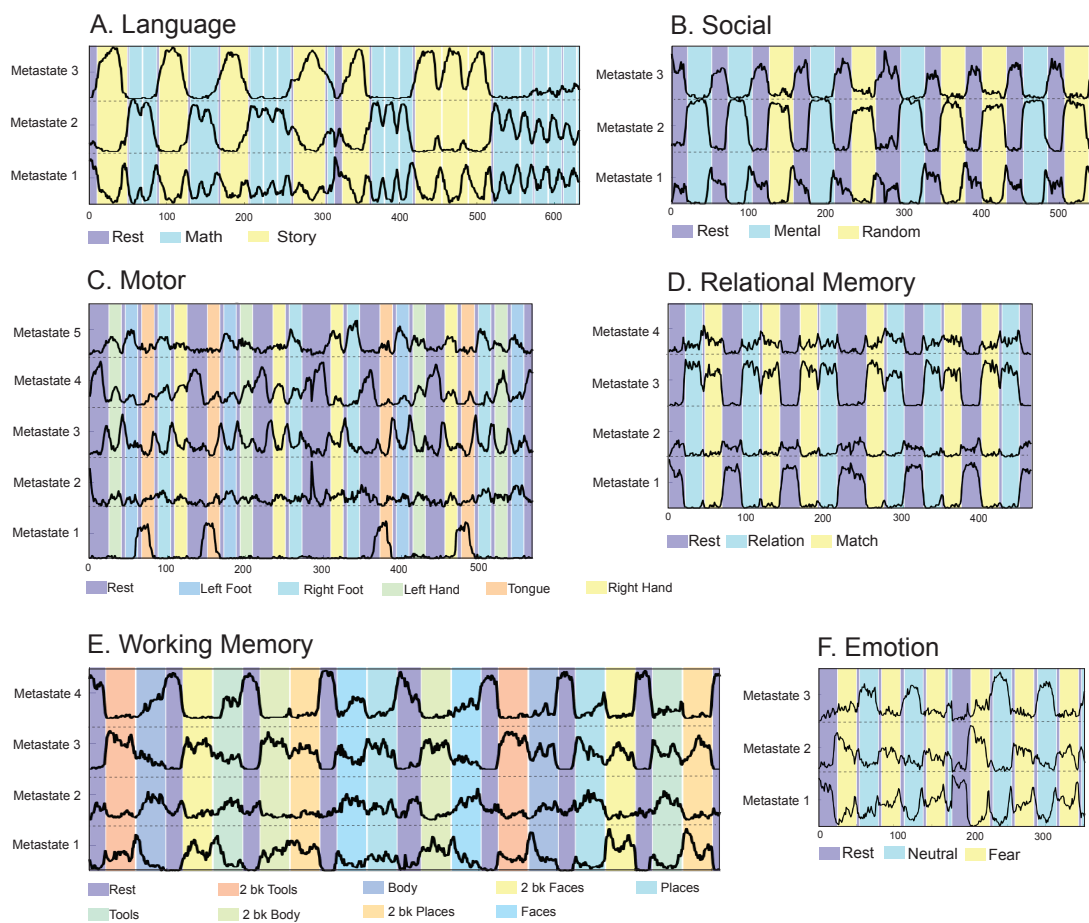


Figure A.1 – **Estimated timecourses with respect to selected task paradigms.** The  $\gamma$  values are plotted over the experimental paradigm of Language (A), Emotion (B), Social (C) and Relational Memory (D) tasks. The black signal corresponds to the probability of belonging to a specific *metastate*.

2 (*Math*) include the parietal areas (superior and inferior) and the frontal region (e.g., middle frontal gyrus, opercular part of the inferior frontal gyrus). These regions are well in-line with the established associated areas corresponding to numbers and calculations [6]. On the other hand, active areas in Metastate 3 (*Story*) are the hippocampus, frontal, and the bilateral superior and anterior temporal cortex, consistent with previously observed regions implicated with story processing tasks [9].

Meanwhile, Metastate 1 corresponding to the *Rest* epoch of Emotion task again captures the DMN pattern. Metastates 2 and 3 corresponding to the *Fear* and *Neutral* phases, respectively both show stark differences in terms of the regions activated. In particular, *Fear* triggers activations in the bilateral central gyrus, superior occipital gyrus, and the parietal cortices. These regions cover the somatosensory cortex, which is a known region responsible not only for processing sensory information from various parts of the body but also for emotional

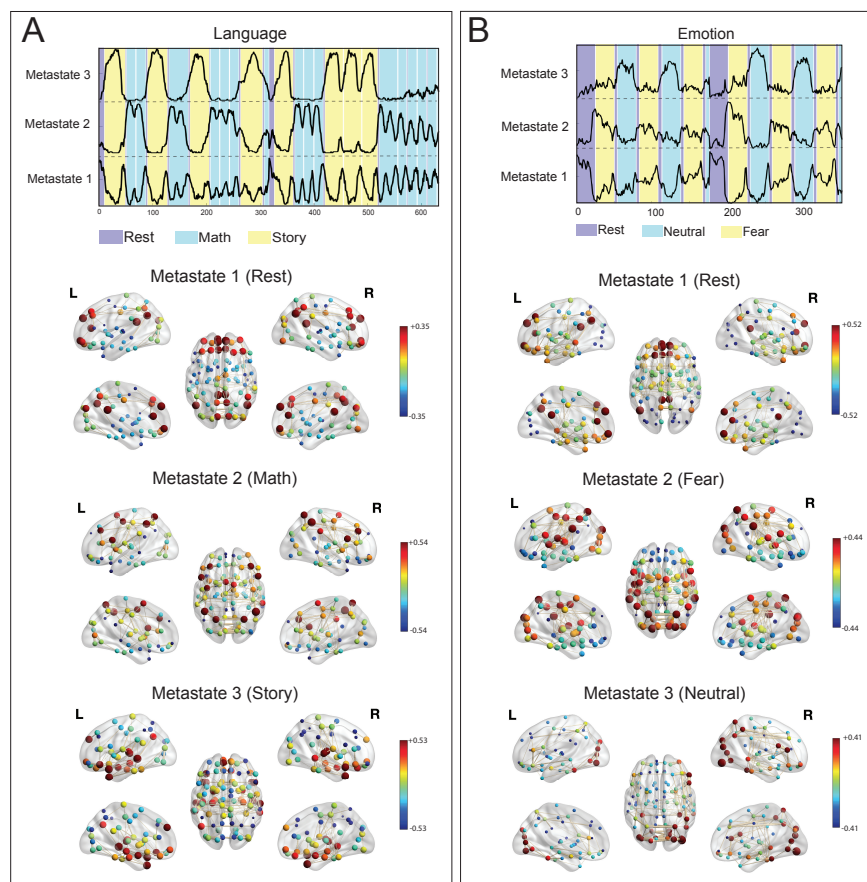
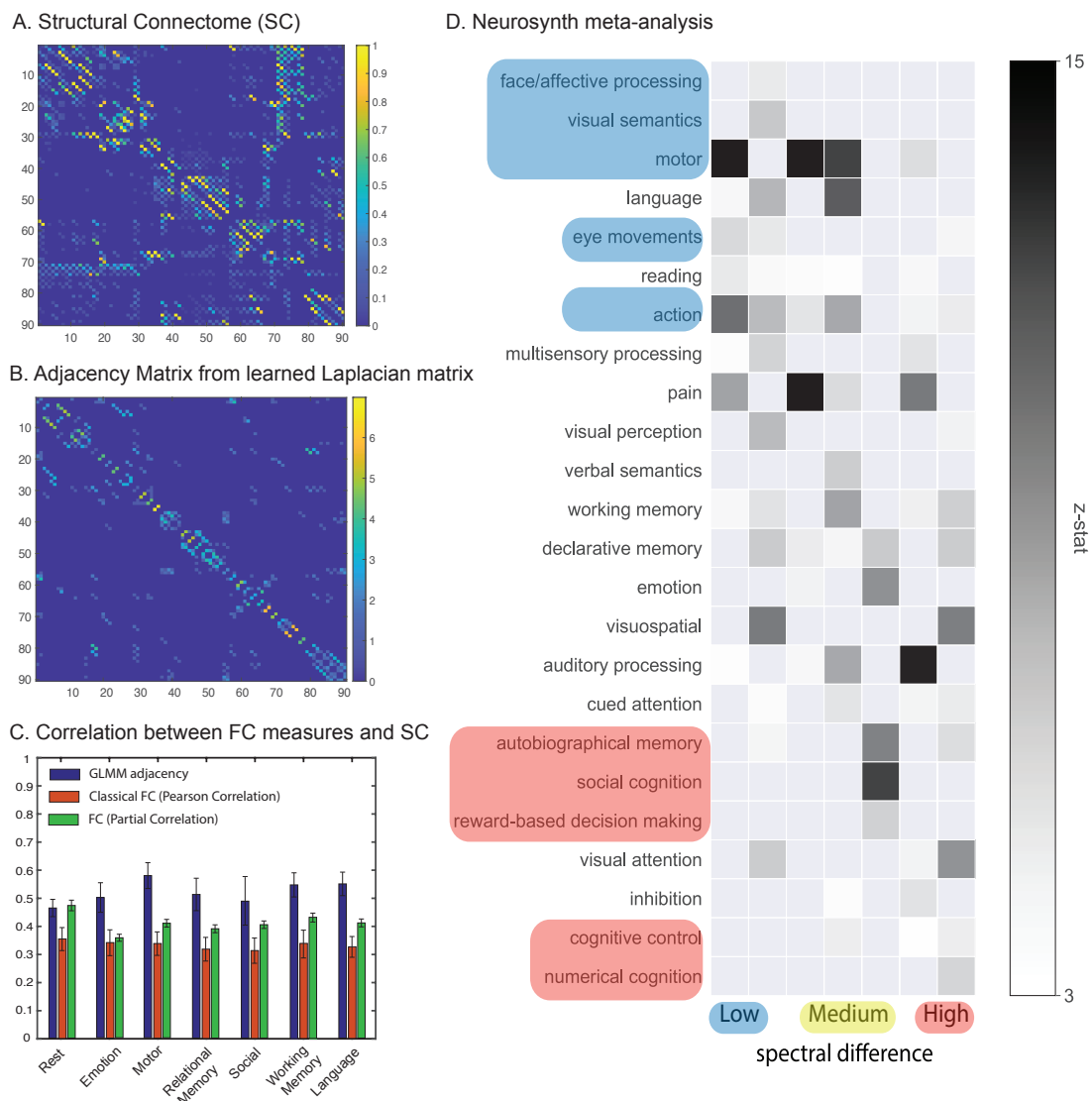


Figure A.2 – **metastates corresponding to each epoch** of the (A) Language task and (B) Emotional task paradigms. Each node corresponds to each brain region in the AAL atlas. The colors denote the signs of the extracted clusters (means), the edges denote the connectivity derived from the estimated graph Laplacians, and the size the nodes indicate the degree of each node's connections. The rest epochs of the two tasks show regions of the default mode network.

processing, including generation of emotional states and emotion regulation [23, 17, 64]. Meanwhile, the metastate for *Neutral* shows activation of visual regions that are strongly related to shape [5, 50]. Unexpectedly, we found relatively low means in the amygdala, a well-known region that is typically activated in emotional matching paradigms [49, 89].

### A.3.3 Learned graphs bear similarity to the underlying brain structure and their relation reveals behaviorally-relevant organization

For each of the tasks evaluated, we obtained different metastates and corresponding graph Laplacian matrices. We, therefore, have a total of 28 metastates across all tasks, and consequently, also 28 graph Laplacian matrices. We evaluated the similarity of these graph Laplacian matrices to the underlying brain structure derived from DW-MRI. Fig. A.3(A) shows the group-averaged structural connectome across all subjects considered in this work, and Fig. A.3(B)



**Figure A.3 – Learned Laplacian matrix and its relation to the structural connectome and behavior.** (A) Group-averaged SC matrix corresponding to all subjects considered in this work. (B) Adjacency matrix extracted from an example Laplacian matrix. (C) Similarity between SC and different FC measures: using (1) GLMM-based adjacency extracted from learned graphs and classical measures of FC using (2) Pearson Correlation and (3) Partial Correlation. (D) Meta-analysis of SC-FC relationship using spectral distance as a metric. The distinction between low level and high-level processing arises thanks to the sorting of the spectral difference and the z-score assigned to a behavioral topic.

displays an example adjacency matrix computed from the estimated graph Laplacian matrix of an example task. The adjacency matrix is visually more sparse than the SC. Moreover, a direct statistical measure (*i.e.*, Pearson correlation) between the estimated GLMM adjacency matrices and SC reveals a similarity values within the range of  $r = 0.48$  to  $r = 0.60$  as is shown in Fig. A.3(C). Compared to the classical FC-SC relationship where FC is obtained by correlating

inter-regional BOLD time-courses using either Pearson correlation or Partial correlation, the strength of FC-SC coherence is much higher using GLMM-based FC.

We then performed a meta-analysis [122] of this SC-FC relationship by using spectral difference as a measure. In essence, this translates to a dissimilarity index between the SC and the GLMM-based FC obtained in all tasks and RS fMRI. Fig. A.3(D) shows the results of the meta-analysis performed using the Neurosynth database, analogous to previous works of Preti, et.al. [90] and Marguiles, et.al., [71]. We displayed seven metastates that showed a statistically significant spectral difference. This is shown in the figure as columns which is sorted in increasing order. The 24 behavioral topics describing the rows are sorted according to their z-score. The ordering of both axes reveals an organized behavioral gradient that naturally arises from this SC-FC relationship. Low-level processes (highlighted in blue) reveal a general low spectral difference while high-level cognitive behavior (in red) show high difference. A diagonal trend can be glimpsed.

### A.4 General findings

We proposed to use our new framework for extracting repeating functional brain patterns based on a generative model that assumes functional data as a collection of signals that live on multiple graphs. Using this technique, we were able to obtain: (1) distinct sets of networks that the brain cycles to, (2) the likelihood of these networks to occur at each time-point, and (3) the underlying graphs that describe the interactions of the regions composing these networks. We validated the approach by demonstrating that the extracted metastates corresponded to spatial patterns that are consistent with their established neurophysiological descriptions. We also showed that the probability of these networks to occur at each time-point generally captured the timing of the task paradigms we evaluated. Then, we applied the approach to RS data to reveal some of the well-known RS networks, such as the DMN, visual, auditory/attention, and salience networks. Finally, we took advantage of the estimated graph Laplacian matrices to understand the interactions of the regions composing these networks, and how they are related to the underlying brain structure obtained from DW-MRI. We showed that the adjacency matrices computed from the graph Laplacians bear closer similarity to SC than conventionally defined FC measures, *e.g.*, Pearson correlation of inter-regional BOLD signals. We then demonstrated that the degree of similarity between the learned graphs and the SC captures behaviorally-relevant gradient which distinguishes low-level and high-level cognitive processes that is consistent with previously observed macro-scale organization in the cortex [90, 71].

# Bibliography

- [1] Y. Aflalo, A. Bronstein, and R. Kimmel. On convex relaxation of graph isomorphism. *Proceedings of the National Academy of Sciences*, 112(10):2942–2947, 2015.
- [2] D. Alvarez-Melis and T. Jaakkola. Gromov-wasserstein alignment of word embedding spaces. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1881–1890, 2018.
- [3] S.-I. Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.
- [4] J. R. Andrews-Hanna. The Brain’s Default Network and Its Adaptive Role in Internal Mentation. *The Neuroscientist*, 18(3):251–270, 2012.
- [5] A. Anzai, X. Peng, and D. C. Van Essen. Neurons in monkey visual area V2 encode combinations of orientations. *Nature Neuroscience*, 10(10):1313–1321, 2007.
- [6] M. Arsalidou, M. Pawliw-Levac, M. Sadeghi, and J. Pascual-Leone. Brain areas associated with numbers and calculations in children: Meta-analyses of fMRI studies. *Developmental Cognitive Neuroscience*, 30:239–250, 2018.
- [7] A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.
- [8] A. Barbe, M. Sebban, P. Gonçalves, P. Borgnat, and R. Gribonval. Graph Diffusion Wasserstein Distances. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2020.
- [9] D. M. Barch, G. C. Burgess, M. P. Harms, S. E. Petersen, B. L. Schlaggar, M. Corbetta, M. E. Glasser, S. Curtiss, S. Dixit, C. Feldt, D. Nolan, E. Bryant, T. Hartley, O. Footer, J. M. Bjork, R. Poldrack, S. Smith, H. Johansen-Berg, A. Z. Snyder, and D. C. Van Essen. Function in the human connectome: Task-fMRI and individual differences in behavior. *NeuroImage*, 80:169–189, 2013.
- [10] H. H. Bauschke and A. S. Lewis. Dykstras algorithm with bregman projections: A convergence proof. *Optimization*, 48(4):409–427, 2000.

## Bibliography

---

- [11] E. Bayram, D. Thanou, E. Vural, and P. Frossard. Mask combination of multi-layer graphs for global structure inference. *IEEE Transactions on Signal and Information Processing over Networks*, 6:394–406, 2020.
- [12] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.
- [13] R. E. Bellman. *Adaptive control processes: a guided tour*, volume 2045. Princeton university press, 2015.
- [14] J.-D. Benamou, G. Carlier, M. Cuturi, L. Nenna, and G. Peyré. Iterative Bregman projections for regularized transportation problems. *SIAM Journal on Scientific Computing*, 37(2):A1111—A1138, 2015.
- [15] B. F. Biswal, Z. Yetkin, V. M. Haughton, and J. S. Hyde. Functional connectivity in the motor cortex of resting human brain using echo-planar MRI. *Magnetic Resonance in Medicine*, 34:537–541, 1995.
- [16] J. P. Boyle and R. L. Dykstra. A method for finding projections onto the intersection of convex sets in Hilbert spaces. In *Advances in order restricted statistical inference*, pages 28–47. 1986.
- [17] I. Bufalari, T. Aprile, A. Avenanti, F. Di Russo, and S. M. Aglioti. Empathy for pain and touch in the human somatosensory cortex. *Cerebral Cortex*, 17(11):2553–2561, 2007.
- [18] T. Caelli and S. Kosinov. An eigenspace projection clustering method for inexact graph matching. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 26(4):515–519, 2004.
- [19] M. Cho, J. Lee, and K. M. Lee. Reweighted random walks for graph matching. In *European conference on Computer vision*, pages 492–505. Springer, 2010.
- [20] M. Cuturi. Sinkhorn Distances: Lightspeed Computation of Optimal Transport. In *Advances in Neural Information Processing Systems*, pages 2292–2300. 2013.
- [21] P. Danaher, P. Wang, and D. M. Witten. The joint graphical lasso for inverse covariance estimation across multiple classes. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(2):373–397, 2014.
- [22] A. P. Dempster. Covariance selection. *Biometrics*, pages 157–175, 1972.
- [23] R. J. Dolan. Emotion, cognition, and behavior. *science*, 298(5596):1191–1194, 2002.
- [24] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst. Learning Laplacian Matrix in Smooth Graph Signal Representations. *IEEE Transactions on Signal Processing*, 64(23):6160–6173, 2016.



- 
- [25] X. Dong, D. Thanou, M. Rabbat, and P. Frossard. Learning graphs from data: A signal representation perspective. *IEEE Signal Processing Magazine*, 36(3):44–63, 2019.
  - [26] Y. Dong and W. Sawin. COPT: Coordinated Optimal Transport for Graph Sketching. *arXiv preprint arXiv:2003.03892*, 2020.
  - [27] R. L. Dykstra. An algorithm for restricted least squares regression. *Journal of the American Statistical Association*, 78(384):837–842, 1983.
  - [28] N. Dym, H. Maron, and Y. Lipman. Ds++ a flexible, scalable and provably tight relaxation for matching problems. *ACM Transactions on Graphics (TOG)*, 36(6):1–14, 2017.
  - [29] H. E. Egilmez, E. Pavez, and A. Ortega. Graph learning from data under laplacian and structural constraints. *IEEE Journal of Selected Topics in Signal Processing*, 11(6):825–841, 2017.
  - [30] P. Emami and S. Ranka. Learning Permutations with Sinkhorn Policy Gradient. *Preprint arXiv:1805.07010*, 2018.
  - [31] P. Erdős and A. Rényi. On random graphs I. *Publicationes Mathematicae (Debrecen)*, 6:290–297, 1959.
  - [32] S. Ferradans, N. Papadakis, J. Rabin, G. Peyré, and J.-F. Aujol. Regularized Discrete Optimal Transport. In *Scale Space and Variational Methods in Computer Vision*, pages 428–439, 2013.
  - [33] M. Figurnov, S. Mohamed, and A. Mnih. Implicit Reparameterization Gradients. In *Advances in Neural Information Processing Systems 31*, pages 441–452. 2018.
  - [34] M. Fiori and G. Sapiro. On spectral properties for graph matching and graph isomorphism problems. *Information and Inference: A Journal of the IMA*, 4(1):63–76, 2015.
  - [35] R. Flamary, N. Courty, A. Rakotomamonjy, and D. Tuia. Optimal transport with Laplacian regularization. In *NIPS 2014, Workshop on Optimal Transport and Machine Learning*, 2014.
  - [36] J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.
  - [37] L. Gan, X. Yang, N. Narisetty, and F. Liang. Bayesian Joint Estimation of Multiple Graphical Models. In *Advances in Neural Information Processing Systems*, pages 9799–9809, 2019.
  - [38] A. Genevay, G. Peyré, and M. Cuturi. Learning Generative Models with Sinkhorn Divergences. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84, pages 1608–1617, 2018.

## Bibliography

---

- [39] R. Gera, L. Alonso, B. Crawford, J. House, J. A. Mendez-Bermudez, T. Knuth, and R. Miller. Identifying network structure similarity using spectral graph theory. *Applied Network Science*, 3(1):2, 2018.
- [40] B. Girault. *Signal processing on graphs-contributions to an emerging field*. PhD thesis, Ecole normale supérieure de lyon-ENS LYON, 2015.
- [41] M. F. Glasser, S. N. Sotiropoulos, J. A. Wilson, T. S. Coalson, B. Fischl, J. L. Andersson, J. Xu, S. Jbabdi, M. Webster, J. R. Polimeni, D. C. Van Essen, and M. Jenkinson. The minimal preprocessing pipelines for the Human Connectome Project. *NeuroImage*, 80:105–124, 2013.
- [42] S. Gold and A. Rangarajan. A graduated assignment algorithm for graph matching. *IEEE Transactions on pattern analysis and machine intelligence*, 18(4):377–388, 1996.
- [43] C. Goodall. Procrustes methods in the statistical analysis of shape. *Journal of the Royal Statistical Society*, 53(2):285–339, 1991.
- [44] M. D. Greicius, B. Krasnow, A. L. Reiss, and V. Menon. Functional connectivity in the resting brain: A network analysis of the default mode hypothesis. *Proceedings of the National Academy of Sciences*, 100(1):253–258, 2003.
- [45] J. Gu, B. Hua, and S. Liu. Spectral distances on graphs. *Discrete Applied Mathematics*, 190-191:56–74, 2015.
- [46] D. K. Hammond, Y. Gur, and C. R. Johnson. Graph diffusion distance: A difference measure for weighted graphs based on the graph Laplacian exponential kernel. In *IEEE Global Conference on Signal and Information Processing*, pages 419–422. IEEE, 2013.
- [47] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary computation*, 9(2):159–195, 2001.
- [48] B. Hao, W. W. Sun, Y. Liu, and G. Cheng. Simultaneous clustering and estimation of heterogeneous graphical models. *The Journal of Machine Learning Research*, 18(1):7981–8038, 2017.
- [49] B. S. Hariri Ahmad and J. Mazziotta. Modulating emotional responses. *NeuroReport*, 11(1):43–48, 2000.
- [50] J. Hegdé and D. C. Van Essen. Selectivity for complex shapes in primate visual area V2. *The Journal of neuroscience : the official journal of the Society for Neuroscience*, 20(5):1–6, 2000.
- [51] P. W. Holland, K. B. Laskey, and S. Leinhardt. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, 1983.

- 
- [52] B. Jiang, J. Tang, C. Ding, Y. Gong, and B. Luo. Graph Matching via Multiplicative Update Algorithm. In *Advances in Neural Information Processing Systems*, pages 3187–3195. 2017.
- [53] I. Jovanović and Z. Stanić. Spectral distances of graphs. *Linear Algebra and its Applications*, 436(5):1425–1435, 2012.
- [54] O. Kallenberg. *Foundations of modern probability*. Springer Science & Business Media, 2006.
- [55] V. Kalofolias. How to learn a graph from smooth signals. In *Artificial Intelligence and Statistics*, pages 920–929, 2016.
- [56] V. Kalofolias, A. Loukas, D. Thanou, and P. Frossard. Learning time varying graphs. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2826–2830. IEEE, 2017.
- [57] V. Kalofolias and N. Perraudin. Large Scale Graph Learning From Smooth Signals. In *International Conference on Learning Representations*, 2019.
- [58] L. Kantorovich. On the transfer of masses: Doklady Akademii Nauk USSR. pages 227–229, 1942.
- [59] J.-Y. Kao, D. Tian, H. Mansour, A. Ortega, and A. Vetro. Disc-glasso: Discriminative graph learning with sparsity regularization. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2956–2960. IEEE, 2017.
- [60] D. G. Kendall. A Survey of the Statistical Theory of Shape. *Statistical Science*, 4(2):87–120, 1989.
- [61] M. E. Khan, W. Lin, V. Tangkaratt, Z. Liu, and D. Nielsen. Variational Adaptive-Newton Method for Explorative Learning. *Preprint arXiv:1711.05560*, 2017.
- [62] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *stat*, 1050:1, 2014.
- [63] N. M. Kriege, P.-L. Giscard, and R. Wilson. On Valid Optimal Assignment Kernels and Applications to Graph Classification. In *Advances in Neural Information Processing Systems 29*, pages 1623–1631. 2016.
- [64] E. Kropf, S. K. Syan, L. Minuzzi, and B. N. Frey. From anatomy to function: the role of the somatosensory cortex in emotional regulation. *Revista brasileira de psiquiatria (Sao Paulo, Brazil : 1999)*, 41(3):261–269, 2019.
- [65] M. Leordeanu, M. Hebert, and R. Sukthankar. An Integer Projected Fixed Point Method for Graph Matching and MAP Inference. In *Advances in Neural Information Processing Systems 22*, pages 1114–1122. 2009.

- [66] T.-Y. Lin and S. Maji. Improved bilinear pooling with {CNN}s. In *British Machine Vision Conference*, 2017.
- [67] A. Lotsi and E. Wit. High dimensional sparse gaussian graphical mixture model. *arXiv preprint arXiv:1308.3381*, 2013.
- [68] K.-S. Lu, E. Pavez, and A. Ortega. On learning laplacians of tree structured graphs. In *IEEE Data Science Workshop (DSW)*, pages 205–209. IEEE, 2018.
- [69] G. Luise, A. Rudi, M. Pontil, and C. Ciliberto. Differential Properties of Sinkhorn Approximation for Learning with Wasserstein Distance. In *Advances in Neural Information Processing Systems*, pages 5859–5870. 2018.
- [70] H. P. Maretic and P. Frossard. Graph laplacian mixture model. *IEEE Transactions on Signal and Information Processing over Networks*, 6:261–270, 2020.
- [71] D. S. Margulies, S. S. Ghosh, A. Goulas, M. Falkiewicz, J. M. Huntenburg, G. Langsf, G. Bezginh, S. B. Eickhoff, F. X. Castellanos, M. Petrides, E. Jefferies, and J. Smallwood. Situating the default-mode network along a principal gradient of macroscale cortical organization. *PNAS*, 113(44):12574–12579, 2016.
- [72] G. Mateos, S. Segarra, A. G. Marques, and A. Ribeiro. Connecting the dots: Identifying network structure via graph signal processing. *IEEE Signal Processing Magazine*, 36(3):16–43, 2019.
- [73] J. D. Medaglia, W. Huang, E. A. Karuza, A. Kelkar, S. L. Thompson-Schill, A. Ribeiro, and D. S. Bassett. Functional alignment with anatomical networks is associated with cognitive flexibility. *Nature Human Behaviour*, 2(2):156–164, 2018.
- [74] F. Mémoli. Gromov–Wasserstein distances and the metric approach to object matching. *Foundations of computational mathematics*, 11(4):417–487, 2011.
- [75] G. Mena, D. Belanger, S. Linderman, and J. Snoek. Learning Latent Permutations with Gumbel-Sinkhorn Networks. In *International Conference on Learning Representations*, 2018.
- [76] P. Mercado, F. Tudisco, and M. Hein. Generalized matrix means for semi-supervised learning with multilayer graphs. In *Advances in Neural Information Processing Systems*, pages 14877–14886, 2019.
- [77] G. Monge. Mémoire sur la théorie des déblais et des remblais. *Histoire de l’Académie Royale des Sciences de Paris*, 1781.
- [78] S. Monti, P. Tamayo, J. Mesirov, and T. Golub. Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data. *Machine Learning*, 52(1-2):91–118, 2003.

- [79] M. Neuhaus, K. Riesen, and H. Bunke. Fast suboptimal algorithms for the computation of graph edit distance. In *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*, pages 163–172. Springer, 2006.
- [80] G. Nikolentzos, P. Meladianos, and M. Vazirgiannis. Matching node embeddings for graph similarity. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.
- [81] A. Ortega, P. Frossard, J. Kovačević, J. M. F. Moura, and P. Vandergheynst. Graph signal processing: Overview, challenges, and applications. *Proceedings of the IEEE*, 106(5):808–828, 2018.
- [82] B. Padeloup, V. Gripon, G. Mercier, D. Pastor, and M. G. Rabbat. Characterization and inference of graph diffusion processes from observations of stationary signals. *IEEE Transactions on Signal and Information Processing over Networks*, 2017.
- [83] E. Pavez, H. E. Egilmez, and A. Ortega. Learning graphs with monotone topology properties and multiple connected components. *IEEE Transactions on Signal Processing*, 66(9):2399–2413, 2018.
- [84] H. Petric Maretic, M. El Gheche, G. Chierchia, and P. Frossard. GOT: An Optimal Transport framework for Graph comparison. In *Advances in Neural Information Processing Systems 32*, pages 13876–13887. 2019.
- [85] H. Petric Maretic, M. El Gheche, and P. Frossard. Graph heat mixture model learning. In *Asilomar Conference on Signals, Systems, and Computers*, pages 1003–1007. IEEE, 2018.
- [86] H. Petric Maretic, D. Thanou, and P. Frossard. Graph learning under sparsity priors. In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, pages 6523–6527, 2017.
- [87] G. Peyré, M. Cuturi, and S. J. Gromov-Wasserstein Averaging of Kernel and Distance Matrices. In *International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 2664–2672, 2016.
- [88] G. Peyré, M. Cuturi, and Others. Computational Optimal Transport: With Applications to Data Science. *Foundations and Trends in Machine Learning*, 11(5-6):355–607, 2019.
- [89] K. Preckel, F. M. Trautwein, F. M. Paulus, P. Kirsch, S. Krach, T. Singer, and P. Kanske. Neural mechanisms of affective matching across faces and scenes. *Scientific Reports*, 9(1):1–10, 2019.
- [90] M. G. Preti and D. Van De Ville. Decoupling of brain function from structure reveals regional behavioral specialization in humans. *Nature Communications*, 10, 2019.
- [91] S. J. Reddi, S. Kale, and S. Kumar. On the Convergence of Adam and Beyond. In *International Conference on Learning Representations*, 2018.

## Bibliography

---

- [92] M. Rossi and P. Frossard. Geometry-consistent light field super-resolution via graph-based regularization. *IEEE Transactions on Image Processing*, 27(9):4207–4218, 2018.
- [93] H. Rue and L. Held. *Gaussian Markov random fields: theory and applications*. Chapman and Hall/CRC, 2005.
- [94] T. Salimans, J. Ho, X. Chen, S. Sidor, and I. Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017.
- [95] S. Sardellitti, S. Barbarossa, and P. Di Lorenzo. Enabling Prediction via Multi-Layer Graph Inference and Sampling. In *13th International Conference on Sampling Theory and Applications*, 2019.
- [96] S. Sardellitti, S. Barbarossa, and P. Di Lorenzo. Graph topology inference based on sparsifying transform learning. *IEEE Transactions on Signal Processing*, 67(7):1712–1727, 2019.
- [97] C. Schellewald and C. Schnörr. Probabilistic Subgraph Matching Based on Convex Relaxation. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 171–186, 2005.
- [98] S. Segarra, A. G. Marques, G. Mateos, and A. Ribeiro. Network Topology Inference from Spectral Templates. *IEEE Transactions on Signal and Information Processing over Networks*, 3(3):467–483, 2017.
- [99] S. Segarra, Y. Wangt, C. Uhler, and A. G. Marques. Joint inference of networks from stationary graph signals. In *Asilomar Conference on Signals, Systems, and Computers*, pages 975–979. IEEE, 2017.
- [100] R. Shafipour and G. Mateos. Online topology inference from streaming stationary graph signals with partial connectivity information. *Algorithms*, 13(9):228, 2020.
- [101] N. Shahid, V. Kalofolias, X. Bresson, M. Bronstein, and P. Vandergheynst. Robust principal component analysis on graphs. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2812–2820, 2015.
- [102] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98, 2013.
- [103] R. Sinkhorn. A relationship between arbitrary positive matrices and doubly stochastic matrices. *The Annals of Mathematical Statistics*, 35(2):876–879, 1964.
- [104] P. Srinivasan, T. Cour, and J. Shi. Balanced graph matching. In *Advances in Neural Information Processing Systems*, pages 313–320. 2007.

- 
- [105] A. Steger and N. C. Wormald. Generating random regular graphs quickly. *Combinatorics, Probability and Computing*, 8(4):377–396, 1999.
  - [106] A. Takatsu. Wasserstein geometry of gaussian measures. *Osaka Journal of Mathematics*, 48(4):1005–1026, 2011.
  - [107] A. Tarun, I. Ricchi, H. Petric Maretic, P. Frossard, and D. Van De Ville. Inference of Multiple functional Brain Networks using Graph Laplacian Mixture Model. *under preparation*.
  - [108] D. Thanou, X. Dong, D. Kressner, and P. Frossard. Learning heat diffusion graphs. *IEEE Transactions on Signal and Information Processing over Networks*, 3(3):484–499, 2017.
  - [109] D. Thanou, D. I. Shuman, and P. Frossard. Learning parametric dictionaries for signals on graphs. *IEEE Transactions on Signal Processing*, 62(15):3849–3862, 2014.
  - [110] A. Tsitsulin, D. Mottin, P. Karras, A. Bronstein, and E. 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. ACM, 2018.
  - [111] K. R. A. Van Dijk, T. Hedden, A. Venkataraman, K. C. Evans, S. W. Lazar, and R. L. Buckner. Intrinsic Functional Connectivity As a Tool For Human Connectomics: Theory, Properties, and Optimization. *Journal of Neurophysiology*, 103(1):297–321, 2010.
  - [112] T. Vayer, L. Chapel, R. Flamary, R. Tavenard, and N. Courty. Optimal Transport for structured data with application on graphs. In *ICML 2019-36th International Conference on Machine Learning*, pages 1–16, 2019.
  - [113] C. Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.
  - [114] S. Vlaski, H. P. Maretić, R. Nassif, P. Frossard, and A. H. Sayed. Online graph learning from sequential data. In *IEEE Data Science Workshop (DSW)*, pages 190–194. IEEE, 2018.
  - [115] D. J. Watts and S. H. Strogatz. Collective dynamics of small-world networks. *Nature*, 393:440–442, 1998.
  - [116] D. Wierstra, T. Schaul, J. Peters, and J. Schmidhuber. Natural Evolution Strategies. In *IEEE World Congress on Computational Intelligence*, pages 3381–3387, 2008.
  - [117] H. Xu, D. Luo, and L. Carin. Scalable Gromov-Wasserstein Learning for Graph Partitioning and Matching. In *Advances in Neural Information Processing Systems 32*, pages 3046–3056. 2019.
  - [118] H. Xu, D. Luo, H. Zha, and L. C. Duke. Gromov-Wasserstein Learning for Graph Matching and Node Embedding. In *International Conference on Machine Learning*, pages 6932–6941, 2019.

## Bibliography

---

- [119] K. Yamada, Y. Tanaka, and A. Ortega. Time-varying Graph Learning Based on Sparseness of Temporal Variation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5411–5415. IEEE, 2019.
- [120] J. Yan, X. Yin, W. Lin, C. Deng, H. Zha, and X. Yang. A short survey of recent advances in graph matching. In *International Conference on Multimedia Retrieval*, pages 167–174, 2016.
- [121] P. Yanardag and S. V. N. Vishwanathan. Deep Graph Kernels. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD ’15, pages 1365–1374, 2015.
- [122] T. Yarkoni, R. A. Poldrack, T. E. Nichols, D. C. Van Essen, and T. D. Wager. Large-scale automated synthesis of human functional neuroimaging data. *Nature Methods*, 8(8):665–670, 2011.
- [123] T. Yu, J. Yan, Y. Wang, W. Liu, and B. Li. Generalizing Graph Matching beyond Quadratic Assignment Model. In *Advances in Neural Information Processing Systems*, pages 853–863. 2018.
- [124] M. Zaslavskiy, F. Bach, and J.-P. Vert. Many-to-many graph matching: a continuous relaxation approach. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 515–530. Springer, 2010.
- [125] F. Zhou and F. De la Torre. Deformable Graph Matching. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2922–2929, 2013.
- [126] F. Zhou and F. D. Torre. Factorized Graph Matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(9):1774–1789, 2016.
- [127] X. Zhu, Z. Ghahramani, and J. D. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *International conference on Machine learning*, pages 912–919, 2003.



## Hermina Petric Maretić

---

PERSONAL INFORMATION	Nationality: Croatian Lives in: Lausanne, Switzerland E-mail: <a href="mailto:hpetricmaretic@gmail.com">hpetricmaretic (at) gmail (dot) com</a>
QUALIFICATIONS AND INTERESTS	<b>Interests:</b> Machine learning interpretability, optimal transport, graphs, network analysis, data science <b>Languages:</b> Croatian (native), English (fluent, C2), French (very good command, B2)
EDUCATION	École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland  <b>Ph.D.</b> Computer science, September 2015 - present <ul style="list-style-type: none"><li>• Graph optimal transport</li><li>• Graph inference</li><li>• Probabilistic graphical models</li><li>• Advisor: <a href="#">Prof. Pascal Frossard</a></li></ul> University of Zagreb, Faculty of Science, Department of Mathematics  <b>Mr.Sc.</b> Mathematical statistics, September 2013 - July 2015 <ul style="list-style-type: none"><li>• <i>Magna cum laude</i> (4.52/5), Thesis: <i>Crowdfunding campaigns analysis using natural language processing</i></li></ul> <b>Mr.Sc.</b> Computer Science, September 2012 - September 2014 <ul style="list-style-type: none"><li>• <i>Summa cum laude</i> (4.88/5), Thesis: <i>Dynamical processes on complex networks</i></li></ul> <b>Bacc.</b> Mathematics (4.73/5), September 2009 - July 2012
WORK EXPERIENCE	<b>Research &amp; Software Development</b>  Amazon Prime Air, Graz, Austria <i>September 2019 – December 2019</i> <ul style="list-style-type: none"><li>• Applied Science intern</li><li>• Machine learning interpretability:<ul style="list-style-type: none"><li>• Which parts of the input is the network focusing on?</li><li>• What training data influenced the decision most?</li></ul></li></ul> <a href="#">Toptal</a> , Freelance <i>August 2014 – November 2014</i> <ul style="list-style-type: none"><li>• Python and PHP developer</li><li>• Data mining and analysis, machine learning, web scraping</li></ul> <a href="#">Poslovna inteligencija</a> , Zagreb, Croatia <i>October 2014</i> <ul style="list-style-type: none"><li>• Intern (developer)</li><li>• Machine learning, data mining and analysis</li></ul> <a href="#">EPFL</a> , Lausanne, Switzerland <i>July 2013 – August 2013</i> <ul style="list-style-type: none"><li>• Intern (researcher) in <a href="#">LCBB</a></li><li>• Bioinformatics and computational biology</li></ul> <a href="#">Ericsson Nikola Tesla</a> , Zagreb, Croatia <i>July 2012 – September 2012</i> <ul style="list-style-type: none"><li>• Intern (developer)</li></ul>

	<b>Teaching</b> EPFL, Lausanne, Switzerland <i>2016 - present</i> <ul style="list-style-type: none"> <li>Teaching assistant in several courses (French and English), notably <i>Network tour of data science</i></li> </ul> University of Zagreb, Zagreb, Croatia <i>2011, 2013</i> <ul style="list-style-type: none"> <li>Student assistant in Linear algebra and measure theory</li> </ul>
SELECTED PUBLICATIONS (FULL LIST: <a href="#">GOOGLE SCHOLAR</a> )	<b>GOT: An Optimal Transport framework for Graph comparison</b> Hermina Petric Maretic, Mireille EL Gheche, Giovanni Chierchia and Pascal Frossard <i>NeurIPS 2019</i> <b>Graph Laplacian mixture model</b> Hermina Petric Maretic and Pascal Frossard <i>IEEE TSIPN 2020</i> <b>A graph learning approach for light field image compression</b> Irene Viola*, Hermina Petric Maretic*, Pascal Frossard and Touradj Ebrahimi <i>SPIE 2018</i> <b>Graph learning under sparsity priors</b> Hermina Petric Maretic, Dorina Thanou and Pascal Frossard <i>ICASSP 2017</i>
INVITED LECTURES, AWARDS AND SCHOLARSHIPS	<b>Lecturer at SDSS</b> <i>September 2020</i> selected as the only PhD student lecturer in the data science summer school talk about graph learning and graph comparison using optimal transport <b>EDIC EPFL Fellowship</b> <i>2015-2016</i> for the first year of PhD studies <b>Rector's award</b> <i>June 2014</i> Petric Maretić, H. <i>Dynamical network analysis - the evolution of a scientific field based on the DESIGN conference 2002-2014</i> <b>Dean's award</b> <i>June 2014</i> for exceptional success during Graduate studies of Computer science and Mathematics <b>2nd place and award at Mozgalo competition</b> <i>May 2014</i> <ul style="list-style-type: none"> <li>Machine learning and data mining</li> <li>Python, Django, PHP, Javascript, Bootstrap</li> </ul> <b>Acknowledgements</b> for exceptional success during studies <i>2012, 2014</i> for undergraduate studies of Mathematics and graduate studies of Computer science <b>Scholarships</b> awarded by the City of Zagreb <i>2008 - 2015</i> awarded to the top 1 % of students with a residence in the city of Zagreb
OTHER EXPERIENCE	<b>Co-founder of Labirintanje</b> , Zagreb, Croatia <i>April 2015 – August 2016</i> <ul style="list-style-type: none"> <li>Creating and running live escape games from scratch</li> </ul> <b>Debate coach and judge</b> (volunteer) <i>September 2011 – September 2014</i> <b>Volunteer teacher at Summer science factory</b> , Samobor, Croatia <i>July 2012</i> <ul style="list-style-type: none"> <li>Motivating and helping children investigate science</li> </ul>