Data Sampling: Database Compression, Graph Reduction, and Compressed Sensing; Three Aspects of Dimensionality Reduction.

Claude Petit

COLLEGE MATHS, TELECOMS
DOCTORAL INFORMATIQUE, SIGNAL
BRETAGNE SYSTEMES, ELECTRONIQUE

Université de Rennes

THÈSE DE DOCTORAT DE

L'UNIVERSITÉ DE RENNES

ÉCOLE DOCTORALE N° 601
Mathématiques, Télécommunications, Informatique, Signal, Systèmes, Électronique
Spécialité : *signal, image, vision*

Par
Claude PETIT

Échantillonnage de données : compression de base de données, réduction de graphe et acquisition comprimée ; trois aspects de la réduction de dimension

Thèse présentée et soutenue à Inria, Rennes, le 26 novembre 2024
Unité de recherche : COMPACT, Inria centre de l'université de Rennes

Rapporteurs avant soutenance :

Elsa DUPRAZ professeure associée, IMT Atlantique.
Charles SOUSSEN professeur, CentraleSupélec.

Composition du Jury :

Président :	Pierre MAUREL	professeur des universités, université de Rennes.
Rapporteurs :	Elsa DUPRAZ	professeure associée, IMT Atlantique.
	Charles SOUSSEN	professeur, CentraleSupélec.
Examinateurs :	Pierre MAUREL	professeur des universités, université de Rennes.
	Adrien SAUMARD	professeur associé, ENSAI.

Invités :

Aline ROUMY directrice de thèse, directrice de recherche, Inria centre de l'université de Rennes.
Thomas MAUGEY directeur de recherche, Inria centre de l'université de Rennes.
Nicolas KERIVEN chargé de recherche CNRS, IRISA Rennes.
François COQUET professeur des universités retraité.

ACKNOWLEDGEMENT

Je remercie tout d'abord Aline Roumy, ma directrice de thèse, pour son encadrement, sa disponibilité et pour les conditions idéales dans lesquelles j'ai pu passer ces trois années et consacrer 100 % de mon temps à la recherche. Je mesure vraiment la chance que j'ai eue de pouvoir travailler dans ce cadre idéal. Je la remercie également de m'avoir suivi sur des sujets qui me passionnaient et pour sa patience. J'ai appris énormément durant ces trois ans, en mathématiques, en théorie du signal, en informatique et cela restera une des périodes les plus enrichissantes de ma carrière professionnelle.

Merci à Thomas Maugey, dont j'ai été le colocataire pendant ces trois ans à Inria, pour nos échanges sur des sujets scientifiques ou non scientifiques. Thomas, ton dynamisme et ton enthousiasme sont contagieux.

Merci à Nicolas Keriven, pour son expertise sur les graphes et les GNN, pour sa disponibilité et ses conseils. Je suis rentré à reculons et avec hésitation dans les réseaux de neurones et *a posteriori* j'y ai découvert un domaine passionnant ; je souhaite maintenant me perfectionner à PyTorch, c'est dire le chemin parcouru.

Je remercie Charles Soussen et Elsa Dupraz d'avoir accepté de rapporter le manuscrit de thèse, de leur lecture attentive, d'avoir pris du temps pour m'apporter leurs conseils et de m'avoir fait bénéficier de leur expertise. Merci aux membres du jury pour le grand honneur qu'ils me font en acceptant de juger mon travail. Je remercie Pierre Maurel de me faire l'honneur de présider ce jury et Adrien Saumard d'examiner mon travail. Merci à François Coquet d'avoir accepté de participer en tant que membre invité, malgré une retraite très active.

Je remercie toute l'équipe Sirocco, maintenant l'équipe Compact, et tout d'abord Christine Guillemot qui m'y a accueilli deux fois en stage et trois ans en détachement. Je remercie les doctorants et les ingénieurs avec lesquels j'ai passé ces trois années. Il était vraiment agréable de venir à Rennes travailler dans une ambiance aussi sympathique. Et merci bien sûr à Caroline Tanguy qui se charge avec gentillesse de toutes les démarches administratives.

Un grand merci à Rémi Gribonval d'avoir accepté de participer à mon CSI, d'avoir gardé un œil attentif sur mon travail, d'y avoir parfois détecté des erreurs et d'avoir pris du temps pour m'aider à les corriger. Merci également à Malcom Egan d'avoir participé à mon CSI et pour ses conseils.

Je tenais à remercier les organisateurs et les participants de l'école d'été du Gretsi à Peyresq, où j'ai passé une semaine motivante et inspirante : Guillaume Ginolhac, Pierre Borgnat et Philippe Ciblat.

Je remercie Christine, Romain et Juliette qui supportent tout ça depuis presque 30 ans.

Je souhaite enfin dédier cette thèse de doctorat à deux de mes anciens professeurs, MM. Hubert Hennion et François Coquet. Il y a de cela très longtemps, lorsque j'étais étudiant à l'Irmar, ils m'ont appris que la théorie des probabilités ne se limitait pas à tirer des boules dans une urne ou à dénombrer des mains au poker et m'ont donné envie de faire de l'enseignement et des mathématiques mon métier.

TABLE OF CONTENTS

RÉSUMÉ EN FRANÇAIS

Contexte

La réduction de dimension vise à transformer des données d'un espace de grande dimension en un espace de dimension inférieure tout en préservant des propriétés jugées essentielles des données d'origine. L'objectif est de rendre possible ou plus rapide le traitement de ces données, de réduire la complexité des processus les impliquant, d'économiser de l'espace de stockage, de l'énergie et de se prémunir contre le fléau de la dimension. Réduire la dimension peut également améliorer l'interprétabilité ou permettre la visualisation des données. On peut enfin considérer la réduction de dimension comme une forme de compression avec perte.

Les méthodes de réduction de dimension sont traditionnellement divisées en approches linéaires et non linéaires, mais d'autres axes de classification existent. On peut par exemple classer les méthodes selon qu'elles sont aléatoires ou déterministes, ou selon qu'elles s'appliquent à des modèles de taille finie ou asymptotiques, lorsque les valeurs des paramètres tendent vers l'infini sous des régimes spécifiques.

La classification qui nous intéresse ici découle du paradigme du big data, où deux paramètres fondamentaux décrivent les dimensions des données : n, la taille de la population (nombre d'éléments de la base de données) et d, la dimension des variables statistiques attachées à ces éléments.

Trois situations sont possibles :

- n grand, d petit : c'est le domaine des statistiques multivariées traditionnelles (l'analyse de données « à la française »). Dans ce scénario, les outils d'inférence statistique classiques fonctionnent bien, en particulier les théorèmes limites classiques, lorsque n tend vers l'infini avec d fixé.
- n petit, d grand : c'est le domaine des statistiques en grande dimension, où les outils d'inférence statistique usuels ne fonctionnent plus, ni dans un cadre non asymptotique ni dans un cadre asymptotique. La matrice de covariance empirique est singulière, les estimateurs des moindres carrés ne sont pas consistants et peuvent donner de mauvais résultats [Wai19],[CD11, Introduction, p.2-4], etc. Des hypothèses supplémentaires sont alors nécessaires pour traiter les données, comme une hypothèse de parcimonie, de structure sous-jacente cachée ayant une petite dimension, etc.
- n et d grands : c'est un autre aspect des statistiques en grande dimension, et typiquement le domaine de la théorie des matrices aléatoires. Dans un cadre asymptotique, aucun théorème limite classique ne s'applique, et des hypothèses sur la limite de n/d doivent être faites lorsque n et d tendent vers l'infini, pour appliquer des théorèmes spécifiques, comme la convergence vers la loi de Marchenko-Pastur [CD11].

À côté des deux paramètres fondamentaux, nous considérons également une troisième possibilité : il peut exister d'autres paramètres pour décrire les données, par exemple si l'espace dans lequel elles vivent n'est pas euclidien. De la taille ou de la complexité de la structure hébergeant les données peuvent alors émerger un ou plusieurs paramètres décrivant cette structure. C'est le cas, par exemple, si les données se trouvent sur un

graphe. Le nombre d'arêtes m est alors un troisième paramètre à prendre en compte. m peut être de l'ordre de n^2 si le graphe est dense, ou de l'ordre de n si le graphe est parcimonieux.

Dans cette thèse, nous abordons chacun des trois aspects du problème de réduction de la dimension et proposons des méthodes pour réduire n, d ou m dans des cadres que nous décrivons maintenant.

La première partie de la thèse traite de compression de bases de données. Nous y proposons une méthode d'échantillonnage d'une base de données $\mathbb{X}$ réduisant n tout en préservant la diversité des informations contenues dans $\mathbb{X}$. La deuxième partie traite de sparsification de graphe. Nous proposons de réduire le nombre d'arêtes m d'un graphe hébergeant des données sur ses nœuds, tout en essayant de préserver la connectivité du graphe. Les première et deuxième parties partagent des outils et méthodes algébriques communs ; nous exploitons ces analogies et proposons un traitement parallèle entre les deux parties. La troisième partie de la thèse traite d'acquisition comprimée et propose une analyse statistique d'un algorithme de reconstruction de signaux parcimonieux. Dans ce cadre, les signaux sont des vecteurs qui appartiennent à un espace de dimension d, mais qui sont parcimonieux et proviennent d'un sous-espace vectoriel inconnu, de dimension beaucoup plus petite. Il faut alors exploiter cette parcimonie dans les algorithmes de reconstruction de signaux. C'est en ce sens que d est réduit.

Notre principal outil à travers les différentes parties est une matrice de travail notée B ou ϕ, dont nous réduisons la taille en extrayant des colonnes ou en s'intéressant à un sous-ensemble de ses colonnes. Selon le contexte, la matrice peut avoir différentes interprétations et différentes dimensions. Dans la première partie, $B \in \mathbb{R}^{d\times n}$ est la matrice de données et ses coefficients représentent les caractéristiques des éléments de la base de données. Dans la deuxième partie, $B \in \mathbb{R}^{n\times m}$ est la matrice d'incidence d'un graphe et ces coefficients $0, 1, -1$ représentent la topologie du graphe. Dans la troisième partie, $\phi \in \mathbb{R}^{m\times d}$ est une matrice de mesure dans un cadre d'acquisition comprimée (CS pour « Compressive Sensing ») et ses coefficients sont des variables aléatoires gaussiennes indépendantes. Dans les deux premières parties, nous travaillons sur les colonnes de B, en construisant une matrice réduite B_k. Dans la dernière partie, nous cherchons les colonnes correspondant au support d'un vecteur parcimonieux, qui varie avec chaque vecteur. Le tableau suivant résume ces informations.

Problème	Matrice initiale	Matrice extraite	Nous travaillons sur...	.. qui représente
I. Échantillonnage	$B \in \mathbb{R}^{d\times n}$	$B_k \in \mathbb{R}^{d\times k}$	n	taille de la pop.
II. Sparsification	$B \in \mathbb{R}^{n\times m}$	$B_k \in \mathbb{R}^{n\times k}$	m	nombre d'arêtes
III. CS: $y = \phi x \in \mathbb{R}^m$	$\phi \in \mathbb{R}^{m\times d}$	-	d	dim. vecteurs

Les paramètres canoniques sont :

- n : taille de la population et/ou nombre d'éléments dans la base de données.
- d : dimension des données et/ou taille du dictionnaire.
- m : paramètre supplémentaire qui peut représenter le nombre d'arêtes (chapitres 2 et 5) ou le nombre de mesures (chapitres 3 et 6).

Plan détaillé

Ce manuscrit est divisé en deux parties : la première partie (chapitres 1 à 3) passe en revue l'état de l'art pour chacun des trois problèmes que nous abordons, et la deuxième partie (chapitres 4 à 6) décrit nos contributions originales dans chacun des trois problèmes.

Le premier chapitre introduit le problème de compression de bases de données. Nous faisons trois hypothèses : $d \ll k \ll n$, le nombre k de colonnes extraites est supérieur au rang d de la matrice des données et les colonnes de la matrice de données sont normées. L'échantillonnage des colonnes se traduit mathématiquement par un problème de sélection d'un sous-ensemble de colonnes (CSSP pour « Column Subset Selection Problem »). Nous proposons une revue de littérature des techniques connues de CSSP pour se focaliser sur celles qui correspondent à notre modèle (un nombre de colonnes supérieur au rang de la matrice des données).

Le second chapitre traite de sparsification et de connectivité dans les graphes. Nous présentons une revue de littérature décrivant l'importance et les très nombreuses incarnations de la notion de connectivité. Nous sélectionnons une définition pertinente et la relions à d'autres notions de connectivité, tout en établissant les propriétés mathématiques qui serviront à présenter nos contributions. Nous effectuons enfin un état de l'art des techniques de sparsification de graphes, en se focalisant sur les plus efficaces connues: les techniques spectrales.

Le chapitre 3 présente brièvement le cadre de l'acquisition comprimée et propose un état de l'art des techniques de reconstruction de signaux parcimonieux. Nous présentons en particulier l'algorithme OMP (Orthogonal Matching Pursuit) dont nous effectuerons, dans le dernier chapitre, une analyse statistique dans un cadre asymptotique.

Le chapitre 4 présente nos contributions dans le problème d'échantillonnage de colonnes, lorsque le nombre de colonnes extraites est supérieur au rang de la matrice de données. Le problème de maximisation de volume que l'on propose de résoudre est NP-difficile et il est nécessaire d'en chercher des solutions approchées. Le lien entre ce problème discret et la théorie des perturbations de rang 1 fait émerger deux principes concernant le spectre de la matrice de covariance associée à la matrice de données. À partir de ces deux principes nous élaborons deux stratégies distinctes: l'une, gloutonne, nous permet de construire deux heuristiques itératives permettant de déterminer des solutions approchées du problème. L'autre, globale, permet d'élaborer trois variantes d'un algorithme s'inspirant des techniques de « Waterfilling ». Nous comparons les performances de ces 5 algorithmes avec un échantillonnage aléatoire uniforme et des processus ponctuels déterminantaux, classiquement utilisés dans ce type de problématique. Nous proposons également deux applications potentielles au conditionnement de matrices et à l'acquisition comprimée.

Le chapitre 5 adapte les algorithmes gloutons du chapitre précédent dans un contexte de théorie des graphes. Il s'agit alors de réduire le nombre d'arêtes d'un graphe sans significativement dégrader sa connectivité. Les deux algorithmes gloutons que nous proposons sont déterministes et ont une complexité quadratique, ce qui en fait des compétiteurs crédibles aux algorithmes de sparsification spectrale les plus efficaces. Nous démontrons que l'un des deux algorithmes est optimal parmi tous les algorithmes itératifs. Nous proposons également une application à la simplification du graphe sous-jacent d'un réseau neuronal sur graphe (« Graph Neural Network »).

Le chapitre 6 présente enfin nos contributions dans l'analyse statistique d'un algorithme de reconstruction de signaux parcimonieux. Nous définissons deux cadres probabilistes dans lesquels la matrice de mesure et le signal à reconstruire sont tous deux aléatoires. Le premier modèle est défini pour des paramètres de

taille d, k, m fixes. Le second modèle est asymptotique : les paramètres tendent vers l'infini sous contrainte de linéarité $k = \gamma d$ et $m = \mu k$, pour des constantes $\gamma \in]0, 1]$ et $\mu > 1$. Ces deux modèles diffèrent des cadres uniformes et non-uniformes généralement proposés en analyse de performances pour le CS [FR13, p. 48; p. 281] et représentent des comportements en moyenne, qui peuvent être moins pessimistes que les études au pire cas. Après avoir démontré quelques lemmes techniques, nous établissons les lois de probabilités des estimateurs statistiques impliqués dans les procesus de reconstruction, en particulier les lois de produits scalaires que l'on retrouve dans de nombreux algorithmes: OMP, OLS, etc. À partir de ces lois nous estimons la probabilité de succès d'OMP à une itération donnée et démontrons que cette probabilité tend vers 1 sans autre condition que celles énoncées plus haut. Nous conjecturons la probabilité de succès sur l'ensemble de l'algorithme. L'ensemble des théorèmes proposés dans ce chapitre peuvent constituer une boite à outils probabiliste utile pour de futurs travaux d'analyse.

INTRODUCTION

Dimensionality reduction aims to transform data from a high-dimensional space into a lower dimensional space while preserving meaningful properties of the original data. The motivation for this process includes saving time when executing algorithms on the database, reducing the complexity of processes acting on the data, circumventing the curse of dimensionality and saving memory space. Lowering the dimension can also improve interpretability or allow for data visualization. Dimensionality reduction may also be viewed as a form of lossy compression.

Methods for reducing dimension are traditionally divided into linear and nonlinear approaches, but other axis of classification exist. For instance, methods can be classified as random versus deterministic, or based on whether they apply to finite size or asymptotic situations where the size of the parameters tend to infinity. The classification we are interested in stems from the Big Data paradigm, where two fundamental parameters describe the dimensions of the data: n, the size of the population (numbers of items) and d, the size of the statistical variables (dimension of the features) [Wai19].

Three possible situations exist:

- n big, d small: this is the domain of traditional multivariate statistics (« analyse de données à la française »). Statistical inference tools work well in this scenario, particularly classical limit theorems, when n tends to infinity with d fixed.
- n small, d big: this is the domain of high dimensional statistics, where usual inference tools do not work well, neither in a non-asymptotic nor in an asymptotic framework. The empirical covariance matrix is singular and least square estimators in regression are not consistent and may perform poorly [Wai19],[CD11, Introduction, p.2-4]. Additional assumptions are necessary to handle the data, such as sparsity, lower dimensional-structure, etc.
- n and d big: is another aspect of high dimensional statistics, typically involving random matrices theory. In an asymptotic framework, no classical limit theorems apply, and assumptions on the limit of n/d has to be made when n and d tends to infinity, to apply specific theorems like convergence toward the Marchenko-Pastur law [CD11].

We also consider a third direction: there may exists other paramaters to describe the data, for example if the space in which they live is not Euclidean. In this case, the size or complexity of the data structure may emerge, for example if the data live on a graph. In such cases, the number of edges m is a third parameter that must be considered. m can be of order n^2 if the graph is dense, n if the graph is sparse.

In this thesis, we will contribute to the three aspects of the dimensionality reduction problem, namely reducing the size of n, d or m.

The first part of this thesis deals with database compression, where we propose sampling a database $\mathbb{X}$ by reducing n while preserving the diversity of information included in $\mathbb{X}$. The second part addresses graph reduction, where we reduce the number of edges m of a graph that hosts data on its nodes, while trying to

preserve the graph's connectivity. The first and the second parts share common algebraic tools and methods; we draw parallels between these analogies and pursue a parallel treatment between the two parts. In the third part of the thesis, we focus on a compressive sensing framework, attempting to reduce the dimension d of the features by exploiting both a sparsity assumption on the data and a random model.

Our main tool across the different parts is a working matrix denoted B or ϕ, which we aim to reduce in size. Depending on the context, the matrix may have different interpretations and dimensions. In the first part, $B \in \mathbb{R}^{d\times n}$ is the datamatrix, whose coefficients represent the features of the elements from the database. In the second part, $B \in \mathbb{R}^{n\times m}$ is the incidence matrix of a graph, whose coefficients $0, 1, -1$ represent the topology of the graph. In the third part, $\phi \in \mathbb{R}^{m\times d}$ is a sensing matrix in a compressive sensing (CS) framework, whose entries are independent Gaussian random variables. In the first two parts, we work on the columns of B, constructing a reduced matrix B_k. In the final part, we search for the columns corresponding to the support of a sparse vector, that may vary with each vector. The folllowing table summarizes this information.

Problem	Initial matrix	Extracted matrix	Working on...	.. which is
I. Data sampling	$B \in \mathbb{R}^{d\times n}$	$B_k \in \mathbb{R}^{d\times k}$	n	pop. size
II. Reduction	$B \in \mathbb{R}^{n\times m}$	$B_k \in \mathbb{R}^{n\times k}$	m	number of edges
III. CS: $y = \phi x \in \mathbb{R}^m$	$\phi \in \mathbb{R}^{m\times d}$	-	d	data dim.

The canonical parameters are:

- n: size of the population and/or number of items in the database.
- d: dimension of data and/or size of the dictionary.
- m: additional parameter, such as number of edges (II) or number of measurements (III).

This manuscript is divided into two parts: the first part reviews the state of the art for each of the three problems we address, and the second part describes our original contributions in each of the three problems.

PART I

State of the art

CHAPTER 1

REDUCING THE SIZE OF THE POPULATION: VOLUME MAXIMIZATION FOR DATA COMPRESSION

1.1 Introduction

The first aspect we address in our problem of dimensionality reduction is the size of the population. The elements of a dataset $\mathbb{X}$ of cardinal n are represented by a real vector of d coordinates, so that $\mathbb{X}$ can be modelized by a data matrix $B \in \mathbb{R}^{d\times n}$. Column vectors represent the elements, called items, rows represent features describing the items via a latent space isomorphic to $\mathbb{R}^d$. Our goal is to sample the data while preserving the diversity of the underlying information.

We make three key assumptions about the data matrix: n is much greater than d (as it is often the case in classical statistics, for example in large surveys or in principal component analysis), the rank of B is equal to d and each column of B is normalized.

The criterion used for extraction is the maximization of the volume of the submatrix. This is motivated by the fact that the volume is closely related to the amount of information contained in a set [CT06b]. For a Gaussian random matrix B, $\ln\det(BB^T)$ is proportional to the differential entropy of the random source and is thus homogeneous to a quantity of information.

In both deterministic and random contexts, the concept of volume (which generalizes the determinant for rectangular or singular matrices) effectively captures the diversity of information. Matrix volume is crucial in fields such as data science, high-dimensional signal processing, numerical linear algebra or scientific computing. It measures the algebraic volume spanned by the columns, quantifies the linear independence of these columns, provides a metric for concepts like diversity or the information contained in the underlying data. It offers a geometric interpretation to the determinant of a matrix. The volume is also used to measure the perceived information of a user regarding data, defined as $\log\det BB^T$, the logarithm of the determinant of BB^T, covariance matrix of B [MT20].

The sampling process boils down to selecting some columns according to a criteria maximizing the volume. This chapter is therefore dedicated to present this specific instance of column subset selection problem (CSSP) and the concept of volume.

The chapter is organized as follows: in section 2, we precise the notations, define volume and some useful properties, outline the mathematical framework of CSSP, and formalize the main optimization problem for

volume maximization. We detail the two possible situations depending on whether the number of columns to sample is below or above the rank of B. Below the rank methods are reviewed in section 2, while section 3 focuses on sampling methods for cases where the number exceeds the rank, aligning with our contributions in Chap. 4.

1.2 CSSP: mathematical framework, problem formulation and related works

The problem is to select k columns ($d \leq k \leq n$) from the initial data matrix B to form a rectangular submatrix $B_k \in \mathbb{R}^{d\times k}$ with maximum volume. This approach acts as a compression method, sampling the population of items while preserving all feature characteristics.

In the following, b_i denotes a column of B, $B_k = (b_1, ..., b_k)$ represents a submatrix with k columns $b_1, ..., b_k$ (possibly reordered). B^T is the transpose of B, $\Sigma = BB^T$ is the covariance matrix and $G = B^TB$ is the Gram matrix of B. The notation $B_k \subset B$ indicates that B_k consists of columns from B, and $b \in B$ means that b is a column of B. (B, b) is the concatenated matrix of B with last column b, and $(B, b)(B, b)^T = BB^T + bb^T$ is the corresponding covariance matrix. Tr denotes the trace operator and Sp the spectrum. $||.||_p$ represents the p-norm for $p = 1, 2$, $||.||_F$ is the Frobenius norm. SPD stands for symmetric positive definite. vol is used for the volume of a matrix; we follow the definition of Ben-Israël [Ben92; ÇM09], which generalizes to a rectangular matrix B of rank d, the usual algebraic volume spanned by the columns of a square matrix:

$$\text{vol}(B) = \prod_{i=1}^{d} \sigma_i = \sqrt{\prod_{i=1}^{d} \lambda_i}, \tag{1.1}$$

where $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_d > 0$ are the d strictly positive singular values of B, $\lambda_i = \sigma_i^2$ are the d strictly positive corresponding eigenvalues of $\Sigma = BB^T$ or $G = B^TB$. Since Σ is a square matrix of rank d,

$$\text{vol}(B) = \sqrt{\det(\Sigma)}. \tag{1.2}$$

The CSSP is our main optimization problem and can be formalized as follows:

Problem 1 (Discrete maximization of the volume)**.** *Given a sample size $k \leq n$, find:*

$$\underset{B_k:\, B_k \subset B}{\text{argmax}}\ \text{vol}(B_k) \tag{1.3}$$

This problem is NP-hard [ÇM09]. To find an approximate solution, two types of methods are available: extracting a submatrix of either lower rank or higher rank than the initial matrix.

The concept of extracting submatrices by maximizing volume originated in the linear algebra and numerical analysis communities. It was initially used for tasks like building low-rank approximations, performing fast matrix multiplication, and optimizing a matrix's condition number [CKM20; GT01; Gor+; Mas22]: high-volume submatrices yield low-rank approximations with good algebraic and numerical properties. With the rise of big data and machine learning, volume has become a key parameter in submatrix extraction applications, including data summarization for search engines [Cel+18], data selection for preserving information [Des+06] and

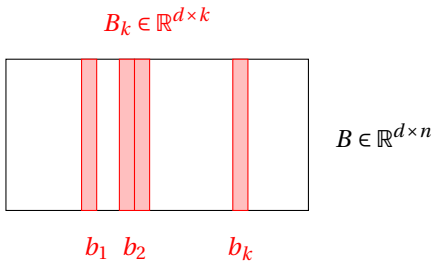

Figure 1.1 – Initial data matrix B and extracted submatrix B_k.

database compression [BBC20; MT20]. Extraction can target rows or columns, reflecting either feature selection or sampling. For simplicity, we assume extraction is performed on columns, though mathematically, row and column extraction are equivalent.

Methods for extracting maximum-volume submatrices fall into two categories:

1. Below-rank extraction: here, the extracted matrix has fewer columns than the rank of the original matrix. This approach is common in low-rank approximation methods, relying on QR, LU, and singular value decompositions [GV96; HJ85], which lead to rank-revealing factorizations [DR07; GE96; HP92; Pan00], as well as cross [CKM20; GT01; Mas22; Tyr00], pseudo-skeleton [GTZ97], or CUR approximations [CK20; DKM06c]. Most of these methods are deterministic. There also exist random techniques, relying on Monte Carlo methods, used for random volume sampling for low-rank approximations, fast matrix multiplication, or projective clustering [Des+06; DKM06a; DKM06b; ÇM07; Git11; Gor+], with complexities that are at best cubic in the size of the matrix's rows and/or columns.

Determinantal point processes (DPPs) also fall into this category. DPPs perform volume sampling on a Gram matrix to extract a random square submatrix with maximum determinant [KT12; BBC20; Cel+18; Liu+21; TBA18; LGD20; Lau20].

2. Above-rank extraction: this category, where the extracted submatrices have more columns than the rank of the original matrix, is less explored and aligns with our framework. The next section details existing methods for this approach.

1.3 CCSP above the rank

Three papers address the CCSP problem where the number of selected columns exceeds the rank of the data matrix.

Avron and Boutsidis [AB13] study the CCSP for $B \in \mathbb{R}^{d\times n}$ with $n > d$ and a sample of k columns where $d \leq k \leq n$. They propose selecting columns based on the Frobenius ($||B||_F^2 = \mathrm{Tr}(BB^T)$) or spectral ($||B||_2 = \sigma_1$) norm of the pseudo-inverse B^+ of B, aiming to maximize each of the singular values of the extracted matrix. Four of their five algorithms perform a volume sampling over the rank of B:

- Algorithms 1 and 2 are deterministic, with a time complexity of $O(dn^2 + nd(n-k))$, greedily removing a column to minimize the trace of B^+. They differ only on the fact that one is adapted for Frobenius norm, the other for spectral norm.

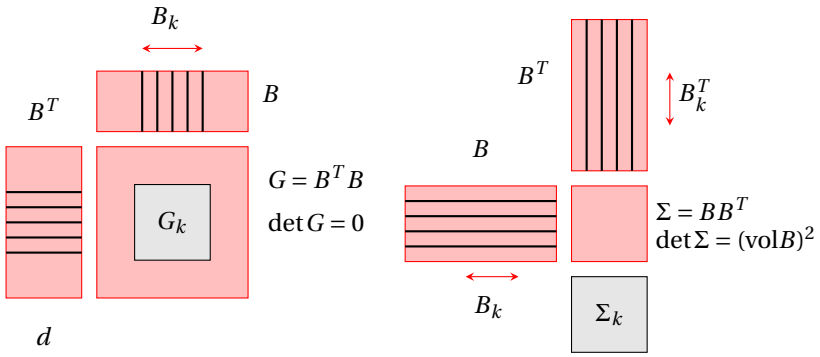

Figure 1.2 – The differences between extraction below (left) and above (right) the rank, in terms of the Gram and covariance matrices. G and Σ are the Gram and covariance matrices, respectively, of the initial data matrix B. G_k and Σ_k are the Gram and covariance matrices, respectively, of the extracted B_k matrix. If $k < d$, $\det G_k > 0$ and $\det \Sigma_k = 0$, while if $k \geq d$, $\det G_k = 0$ and $\det \Sigma_k > 0$.

- Algorithm 3 is deterministic with a fourth-order polynomial complexity $O(nd^2 + kd^2 n))$, inspired by Batson and al.'s sparsification algorithm [Bat+13]. It is not strictly a CCSP algorithm, as it extracts a linear combination of columns and assigns weights to each.
- Algorithm 4 is a one-shot random sampling algorithm with a complexity of $O(nd^2 + k \ln k))$, inspired by Spielman and Srivastava [SS11], which also assigns weights to the selected columns.

Li and al. [LJS17] introduce "dual volume sampling" (DVS), where dual refers to the fact that the sampling is performed above the rank. They propose two algorithms:

- A random one-shot algorithm with a time complexity of $O(kn^4)$ to sample a volume-maximization probability distribution in a manner similar to DPPs. The global criterion to achieve is the same as that of Avron and Boutsidis.
- A deterministic greedy algorithm with a time complexity of $O(kdn^4)$, which is a derandomized version of the previous algorithm.

Mikhalev and Oseledets [MO17] propose a greedy algorithm called RectMaxVol inspired by [Dyk71] that concatenates columns such that the added column is the one maximizing the volume. The time complexity is $O(nk^2)$. The algorithm begins by forming a square matrix using the MaxVol algorithm, a greedy iterative method that randomly swaps rows to maximize the volume of a square submatrix [Gor+]. This initialization makes RectMaxVol a randomized algorithm.

In summary, all deterministic methods have at least fourth-degree polynomial complexity, and the random methods have at least cubic complexity relative to matrix dimension. Thus, the CSSP over the rank lacks a deterministic method of reasonable complexity (e.g., less than cubic) suitable for large databases.

CHAPTER 2

REDUCING THE SIZE OF THE DATA STRUCTURE: GRAPH SPARSIFICATION PRESERVING CONNECTIVITY

2.1 Motivation and objectives

The second aspect of our dimensionality reduction problem does not concern the two fundamental statistical dimensions (population size n and dimension d of each data), but rather the structure of the data itself. When data do not reside in a Euclidean space, a new parameter may appear to describe the structure of the underlying space. In this section, our data are hosted in the nodes of a graph, where the structural paramater is given by the number m of edges, which measures the complexity of describing a non-Euclidean space. This section is therefore dedicated to graph sparsification: approximating a dense graph by a sparse one, by removing edges while keeping all the nodes.

Our interest in graph sparsification is also driven by an intuition regarding strong analogies between graph sparsification and database compression, as developped through the CSSP in the previous section of this thesis. These analogies prompt the following question: is there an equivalent notion of matrix volume in graph theory? We have seen that if B is a data matrix and $\Sigma = BB^T$ its covariance matrix, then $\det\Sigma = (\mathrm{vol}B)^2$ represents the information contained in the database. If we now consider B as the incidence matrix of a graph G, then $BB^T = L$ is the combinatorial Laplacian of the unweighted graph, and its reduced determinant (equal to any cofactor) is a well-known quantity representing the connectivity of the graph.

Our objectives, developed in Chap. 5, is to build graph sparsification algorithms that preserve connectivity. More specifically, we aim to design sparsification algorithms that extract a sparse, unweighted subgraph with maximum connectivity for a given number of edges. This chapter is intended to present related works on connectivity, select a relevant definition of this concept from among numerous indices of connectivity, review state-of-the-art methods of sparsification and establish the necessary mathematical tools.

The chapter is organized as follows: section 2 is dedicated to connectivity and section 3 to sparsification. In section 2, we justify the importance of the notion of connectivity through a literature review. From this review, the need to unify the numerous concepts related to connectivity and to link global and local aspects becomes apparent. We motivate the choice of a specific connectivity index as a fundamental quantity to preserve during sparsification and we link this choice to several other important quantities. In doing so, we establishe a number of properties that will be useful for presenting our contributions. In section 3, we review spectral sparsification

methods to which we compare our contributions in Chap. 5.

2.2 The ubiquitous connectivity of a graph

In the following subsections, we review in detail the different notions of connectivity, explain the choice of one of them and connect various concepts. The proofs provided in this subsection are selected for their relevance to our subject and are consistent with this original presentation of concepts related to connectivity. They can be skipped during a first reading. The section is intended to be a complete self-contained synthesis; therefore, some subsections will not be used in our contributions and can also be skipped in a first reading. This applies to subsubsection 2.2.3 which covers the zeta function, and subsubsection 2.2.5 which discusses random walks.

2.2.1 Related works

Among all the characteristics of a graph, the ubiquitous concept of *connectivity* is one of the most important, as testifed by the numerous indices measuring it [Fre+23; Mar+18; EK13; LB07]; the connectivity controls network robustness [Fre+23; Mar+18; SBG23; AE12; MSN10], information flow [BH09], reliability in telecommunications [CH11] and has applications in fault tolerant systems [LWC23] or social network analysis [Kol09]. The notion is at the heart of clustering algorithms [Lux07; Nd11], centrality and community detection methods [BF05; BF13], online learning [PAS14] or minimum cuts in graph [KS96]. In short, connectivity intervenes in a wide variety of problems related to graphs, making it an important property to preserve during sparsification.

The term *robustness* appears frequently in the literature and is used as a synonym of connectivity.

2.2.2 Notations

In this chapter, and in Chap. 5, we use the following notations: an unweighted and undirected graph G is a set of vertices (or nodes) connected by edges. Let $V = (v_1, ..v_n)$ the set of vertices and $E = (e_1, ..., e_m)$ the set of edges. Any edge is of the form $e = (v_i, v_j)$ (or (i, j) if there is no ambiguity) and can be oriented or not. We assume there are no multi-edge.

$B \in \mathbb{R}^{n\times m}$ is the incidence matrix with $b_{ij} = \pm 1 \iff v_i \sim e_j$. This is the definition of an oriented incidence matrix, but we won't use the orientation. A column b_e of B is denoted by a single index corresponding to the edge e. We denote by $\delta_i \in \mathbb{R}^n$ the vector whose only nonzero coordinate is 1 at vertex v_i. If i and j are the endpoints of an edge, then $b_e = \delta_i - \delta_j$ is the column of B representing the edge $e = (v_i, v_j)$. However, we will also use the notation $(\delta_i - \delta_j)$ for any pair of vertices (v_i, v_j) of the graph, whether it represents an edge or not.

$A = (a_{ij}) \in \mathbb{R}^{n\times n}$ is the adjacency matrix defined by $a_{ij} = 1 \iff v_i \sim v_j$. If any, $W = (w_{ij}) \in \mathbb{R}_+^{m\times m}$ is the diagonal matrix of weights, $D = (d_{ij})$ is the diagonal degree matrix with $d_{ii} = d(i) = \sum_{j=1}^{n} w_{ij}$. For a weighted graph, we can allow the entries of A to be non-binary, in which case the coefficient a_{ij} of A represents the weight corresponding to the edge $e = (v_i, v_j)$ and, if it is nonzero, is one of the diagonal elements of W. $L = D - A$ is the combinatorial Laplacian. $L = BWB^T$ (or BB^T is G is unweighted) but it does not depend on any orientation defined in B. $u_1 \in \mathbb{R}^n$ is the vector with all coordinates equal to 1 and $J \in \mathbb{R}^{n\times n}$ the matrix with all entries equal to 1.

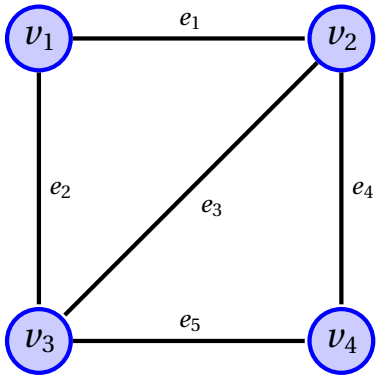

Figure 2.1 – Example of an unoriented, unweighted graph with $n = 4$ and $m = 5$.

Let us illustrate these definitions with a very simple example. Consider an unweighted graph G shown in Fig. 2.1, with vertex set $V = \{v_1, v_2, v_3, v_4\}$ and edges set $E = \{e_1, e_2, e_3, e_4, e_5\}$. Its incidence, adjacency, Laplacian and weight matrices are given respectively by:

$$B = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix} \quad A = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \quad L = \begin{pmatrix} 2 & -1 & -1 & 0 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 3 & 0 \\ 0 & -1 & 0 & 2 \end{pmatrix} \quad W = I_5. \tag{2.1}$$

A spanning tree is a subgraph connecting all nodes without cycles. It possesses $n-1$ edges. The figure 2.2.2 shows all the spanning-tree of the graph from Fig. 2.1.

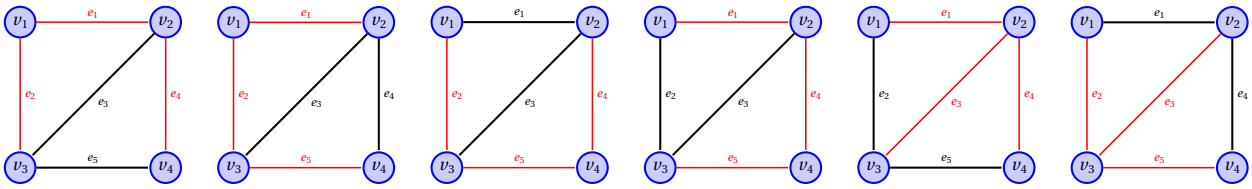

Figure 2.2 – All possible spanning trees (highlighted in red) from graph of Fig. 2.1.

Without loss of generality, we assume G is connected (otherwise, we can work on each connected component separately), so that L and B are of rank $n-1$. This is the main mathematical difference between Σ, as studied in the previous chapter, and L: in the last chapter, we worked with a positive definite matrix, while in this chapter, we deal with a singular matrix. Let $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$ be the spectrum of L and $(u_1, ...u_n)$ a corresponding orthonormal basis. The decomposition of L as a rank-1 operator gives

$$L = \sum_{i=1}^{m} b_i b_i^T = \sum_{i=1}^{n} \lambda_i u_i u_i^T. \tag{2.2}$$

The Moore-Penrose pseudo-inverse of L is givent by

$$L^+ = \sum_{i=2}^{n} \frac{1}{\lambda_i} u_i u_i^T. \tag{2.3}$$

The kernel of L, $\ker L$, is spanned by u_1. The image of B, $\mathrm{Im}B \subset \mathbb{R}^m$, is referred as the cut space.

$LL^+ = L^+L = \sum_{i=2}^{n} u_i u_i^T$ represents the projection onto the span of $u_2, ..., u_n$ and its restriction to $\mathrm{Im}L = (\ker L)^\perp$ is the identity.

We often use the transfer current matrix $\Pi = W^{1/2} B^T L^+ B W^{1/2}$. It is straightforward to verify that $\Pi^2 = \Pi$ and $\mathrm{Im}\Pi = \mathrm{Im}(BW^{1/2}) = W^{1/2}\mathrm{Im}B$. Thus, Π is a projection matrix with two eigenvalues: 0 of multiplicity $m - n + 1$ and 1 with multiplicity $n - 1$.

Following the analogy between the data matrix B from the previous chapter and the incidence matrix B, a natural question arises: are the three assumptions made about B still valid? We have already established that $\mathrm{rank}B = n - 1$, so the first assumption is no longer valid. m is still assumed to be much greater than n (the sparsification process has to reduce m toward n) and the norm of the columns of B are still all equal (since b has only two nonzero elements, $||b||^2 = 2$).

2.2.3 Tree number, Laplacian volume and zeta function of a graph

The matrix tree theorem

Our objective requires a precise definition of the connectivity, but there exists a huge number of indices in graph theory that measure this quantity. Historically, one of the earliest notions of connectivity was defined by the *number of spanning trees* $\tau(G)$ on a finite unweighted graph G, known as the tree number or the complexity of the graph. This quantity will serve as our definition for several reasons discussed below.

In 1847, in the matrix-tree theorem, Kirchhoff [Kir58] demonstrated that the number of spanning trees is equal to the product of nonzero eigenvalues of the combinatorial Laplacian of G (assuming G is unweighted and connected), divided by the number of vertices:

Theorem 1 (Matrix tree theorem)**.**

$$\tau(G) = \frac{1}{n}\lambda_2 \cdots \lambda_n = \frac{1}{n} \det L_0 = \frac{1}{n} \mathrm{vol} L \tag{2.4}$$

where $\det L_0$ is any principal minor of the Laplacian matrix L and $\mathrm{vol}L$ is the volume of the matrix L in the sense of Ben-Israël. Since volume is our main tool for database compression, this justifies our definition for connectivity. As we will see, both the number of spanning trees and the volume appear in other forms, which we propose to list.

Proof. Let J the $n \times n$ size matrix with all coefficients equal to 1 and $u_1 = (1, ..., 1)^T \in \mathbb{R}^n$. Since we assume G is connected, $\mathrm{rank}\, L = \mathrm{rank}\, B = n - 1$ and $\ker L$ is generated by u_1. Let Q the cofactor matrix of L. Then $LQ = \det Q . I_n = 0$, so each column of Q belongs to $\ker L$ and is a multiple of u_1. Since L is symmetric, Q is also symmetric, hence Q is a multiple of J. We now prove that the multiplicative factor is equal to $\tau(G)$. Let $B_\circ$ be obtained from B by deleting the last row. Then $B_\circ B_\circ^T$ is a cofactor of L. The Caucy-Binet formula gives

$$\det(B_\circ B_\circ^T) = \sum_{|S|=n-1} \det B_S \det B_S^T, \tag{2.5}$$

where B_S is the square submatrix of B whose $n-1$ columns corresponds to the edges whose indices belongs to S. $\det B_S$ is nonzero if, and only if the corresponding graph is a spanning tree, in which case $\det B_S = \pm 1$. Hence the conclusion. □

Several important remarks follow from this theorem:

- The Laplacian does not depend on the orientation chosen on B.
- The tree number is equal to any cofactor of L.
- Any minor of L is proportional to $\tau(G)$ and is therefore non-zero. Thus, any submatrix obtained by removing a row of L and the corresponding column is invertible.

We will use the two following results:

Theorem 2 (Temperley 1964)**.**

$$\tau(G) = \frac{1}{n^2} \det(L + J), \tag{2.6}$$

whose proof is given in [Big93].

Theorem 3 (Contraction deletion formula)**.** *For any edge e, let $G\backslash e$ the deletion graph obtained after removing the edge e and G/e the contraction graph obtained by identifying the endpoints of e in G and deleting the loops created. Then,*

$$\tau(G) = \tau(G/e) + \tau(G\backslash e). \tag{2.7}$$

Proof. We denote by $\mathcal{T}$ the set of all the spanning trees T included in G. For any fixed edge e of G, either $e \in T$ either $e \notin T$, so that $\mathcal{T}$ is partitioned into

$$\mathcal{T} = \mathcal{T}_e \cup \overline{\mathcal{T}_e}, \tag{2.8}$$

with $\mathcal{T}_e$ the set of spanning trees from G containing e. Now if $e \notin T$, then T is a spanning tree of $G\backslash e$ and if $e \in T$, T is a spanning tree of G/e. Thus, $\tau(G\backslash e)$ gives the number of trees T with $e \notin T$ and $\tau(G/e)$ gives the number of trees T with $e \in T$, hence the formula. □

We now provide a second proof of the matrix tree theorem using the contraction deletion formula.

Proof. Let E_{ii} be the elementary $n \times n$ matrix with only nonzero coefficient $e_{ii} = 1$, let L_{-i} be the matrix obtained by deleting row and column i and L_{-ij} the matrix obtained by deleting rows and columns i and j. By the multi-linearity of the determinant,

$$\det(L + E_{ii}) = \det L + det L_{-i}. \tag{2.9}$$

The proof proceeds by induction. If G is the empty graph on two vertices, the result is obvious. For the inductive step, we use the contraction-deletion recursive formula for $\tau(G)$. If v_i is an isolated vertex, G admits no spanning tree and there are zeros along the ith row and column. The formula is then valid. Otherwise, if $e = (v_i, v_j)$ is incident to v_i, we have

$$\det L(G)_{-i} = \det\left(L(G\backslash e) + E_{jj}\right) \tag{2.10}$$
$$= \det\left(L(G\backslash e)_{-i}\right) + \det\left(L(G\backslash e)_{-ij}\right) \tag{2.11}$$
$$= \det\left(L(G\backslash e)_{-i}\right) + \det\left(L(G)_{-ij}\right) \tag{2.12}$$

Now, consider G and G/e: if we contract i and j, then $L(G/e)_{-j} = L(G)_{-ij}$ so that

$$\det L(G)_{-i} = \det\left(L(G\backslash e)_{-i}\right) + \det\left(L(G/e)_{-j}\right) \tag{2.13}$$
$$= \tau(G\backslash e) + \tau(G/e) = \tau(G). \tag{2.14}$$

□

The above proofs can be found in [Big93; Spi].

Intuitively, the greater the number of spanning trees, the more difficult it will be to separate a node from the graph. $\tau(G)$ is a global measure of the connectivity of the entire graph. The corresponding local connectivity measure is given by the *probability* $\mathbb{P}[e \in T]$ *of any edge* e *to belong to a spanning tree*, when a random draw is made with a uniform probability on all trees:

$$\mathbb{P}[e \in T] = \frac{|\mathcal{T}_e|}{|\mathcal{T}|} = \frac{\tau(G/e)}{\tau(G)}. \tag{2.15}$$

Zeta function of a graph

The Riemann zeta function is defined for $\mathbb{R}e(s) > 1$ by

$$\zeta(s) = \sum_{n=1}^{+\infty} \frac{1}{n^s} = \prod_{p \text{ prime}} \left(1 - p^{-s}\right)^{-1}. \tag{2.16}$$

Riemann proved in 1859 that ζ is meromorphic on $\mathbb{C}$ with only one pole at $s = 1$. He uses the function in his attempt to prove the prime numbers theorem and conjectures the Riemann hypothesis (RH) on this occasion. The ζ function connects an analytic, algebraic and geometric perspective on the same object. Since then, many kinds of zeta functions have been investigated in various fields of mathematics, displaying striking similarities to the original Riemann function. In graph theory, from the perspectives of Riemannian geometry or algebraic topology, a graph should be studied by its geodesics. For a finite graph, a geodesic is a backtrackless path between vertices of the graph. The *Ihara zeta function* [Iha66] of a finite, connected and oriented graph G is defined by

$$\zeta_G(s) = \prod_{[C]} \left(1 - s^{|C|}\right)^{-1}, \tag{2.17}$$

where s is a complex number with $|s|$ sufficiently close to zero, C is a backtrackless, tailless, closed path of C, $[C]$ is its equivalent class and $|C|$ is its length in terms of number of edges. A path is a sequence of connected and oriented edges $P = e_1 ... e_n$, where the endpoint of e_i is equal to the origin of e_{i+1} for $i = 1, ..., n-1$. We denote e^{-1} as the reverse of e with respect to a given orientation. The path is closed if e_1 and e_n are connected, backtrackless if $e_i^{-1} \neq e_i$, tailless if $e_n \neq e_1^{-1}$ and primitive if $C \neq P^k$ for any path P. The equivalent class $[C]$ of C consists of the circular permutations of the edges from C, and $|C| = n$. The equivalent class of the primitive closed walks plays a role similar to that of prime numbers in this zeta function. The main object of interest is the set of all the (possibly infinite) paths on the graph, where each path can be uniquely decomposed into a concatenation of primitive closed paths.

The Ihara zeta function encapsulates all information about the paths of G and the geometry of the entire graph.

Let $f(s) = \det(I_n - sA + s^2(D - I_n))$. Hashimoto [Has90] and Bass [Bas92] proved that

$$\zeta_G(s)^{-1} = (1-s^2)^{m-n} \det(I_n - sA + s^2(D - I_n)) = (1-s^2)^{m-n} f(s). \tag{2.18}$$

Thus, ζ is the reciprocal of a polynomial of degree $2m$. Northshield [Nor98] proved that

$$f'(1) = 2(m-n)\tau(G), \tag{2.19}$$

so that

$$\tau(G) = \frac{1}{2(m-n)} \frac{d}{ds} \left[(1-s^2)^{m-n} \zeta(s) \right]_{s=1}^{-1}. \tag{2.20}$$

We note that $f(1) = \det L = 0$, and

$$\lim_{s \to 1^-} (1-s)^{m-n+1} \zeta(s) = -\left[2^{m-n+1} (m-n) \tau(G) \right]^{-1}. \tag{2.21}$$

There is a beautiful theory surrounding zeta functions on graphs [Ter05; Ter10] but we will not dvelve further in that direction. The purpose of this section is to emphasize that the number $\tau(G)$ of spanning trees in a graph, in addition to being a graph invariant, is deeply connected to the overall geometry of G and encapsulates information about the paths and the fundamental group of the graph. As we shall see, $\tau(G)$ plays a role in many other aspects of G.

2.2.4 Effective resistance and Kirchhoff electrical laws

Effective resistance is central to our work on graph sparsification. Initially, we define this concept for weighted graph, where the weights represent physical quantities such as conductance, flow capacity, or distances. But our primary algorithms are adapted for unweighted graphs, so in a second time, we will set all the weights to 1 in order to derive the corresponding formulas for unweighted graphs.

Consider G as an idealized electrical resistors network, where each edge e is modeled as a resistor with conductance w_e. Using this analogy, we can apply the Kirchhoff electrical laws for the current and voltage to analyze the behavior of electrical flows and potentials in the network [DS84], [Spi, chap. 12]. This approach

simplifies greatly numerous properties and provides clear interpretations of phenomena related to effective resistance.

If a resistor is not part of an electrical network, its no-load resistance is given by $r = 1/w_e$. We aim to define the resistance of an edge e when this edge, considered as a resistor, is included in a circuit with an electric current flowing through it.

We define a flow f as a real-valued function on the edges (for instance the electric current I) that satisfies Kirchhoff's current law (KCL), and a potential p as a real-valued function on the nodes that satisfies Kirchhoff voltage law (KVL) (for example, the electric potential U at a node).

Let us fix an arbitrary orientation on the graph. For a vertex v, let ∂v^- denote the neighbors of v, where the edge is oriented toward v (inflow nodes) and ∂v^+ denote the neighors of v where the edge is oriented away from v (outflow nodes). For an edge $e = (v_i, v_j)$ we often use the notation $e = (i, j)$ and identify the vertex v_i with its index i.

A unit flow $f : E \mapsto \mathbb{R}$ between a source vertex s and a terminal vertex t (referred as to an $s-t$ flow) assigns a real value to each edge $e = (i, j)$ such that

$$\sum_{e\in E} f_e = \sum_{i\in\partial j^+} f_{(i,j)} - \sum_{i\in\partial j^-} f_{(i,j)} = \delta_s - \delta_t = \begin{cases} 1 & \text{if } j = t \\ -1 & \text{if } j = s \\ 0 & \text{otherwise} \end{cases} \tag{2.22}$$

The KCL states that the algebraic sum of the currents into any junction is zero. The total flow into a vertex equals the total flow out of it, except at the source and terminal vertices where the current is injected and extracted. So the KCL is also called the flow conservating property [Cha+89].

We can rewrite the previous equation in a matrix form by using the incidence matrix B of G and the column vector $b_e = \delta_s - \delta_t$ corresponding to the edge $e = (s, t)$,

$$Bf = b_e. \tag{2.23}$$

If (s, t) are not the endpoints of an edge, but any pair of vertices, the formula remains valid:

$$Bf = \delta_s - \delta_t. \tag{2.24}$$

Any vector f satisfying 2.23 is a unit $s-t$ flow. KVL states that a potential $p : V \to \mathbb{R}$ can be associated with each vertex v such that for all edges $e = (i, j)$,

$$f_{(i,j)} = w_{ij}(p_i - p_j). \tag{2.25}$$

This is the Ohm's law (OL). In matrix form,

$$f = WB^T p. \tag{2.26}$$

where $W = (w_{ij})_{ij} \in \mathbb{R}^{m\times m}$ is the diagonal matrix of conductances. Any vector f satisfying both KCL and OL is an electrical unit flow.

Definition 1. *Let I_{ij} a unit electrical $i-j$ flow (between v_i and v_j). The effective resistance R_{ij} between vertices v_i and v_j is*

$$R_{ij} = p_i - p_j. \tag{2.27}$$

In words, R_{ij} is the potential difference between vertices v_i and v_j if a unit current is injected in v_i and extracted in v_j.

Theorem 4.

$$R_{ij} = (\delta_i - \delta_j)^T L^+ (\delta_i - \delta_j). \tag{2.28}$$

Proof. From KCL and OL,

$$BWB^T p = Lp = Bf = \delta_i - \delta_j \tag{2.29}$$

$$\iff p = L^+(\delta_i - \delta_j), \tag{2.30}$$

since $(\delta_i - \delta_j) \perp u_1$ as G is connected. To find the potential p in a given coordinate, we use the δ functions: $p_i = \delta_i^T p$, so that

$$R_{ij} = p_i - p_j = \delta_i^T p - \delta_j^T p = (\delta_i - \delta_j)^T L^+ (\delta_i - \delta_j). \tag{2.31}$$

□

We note that the diagonal elements of the matrix $B^T L^+ B \in \mathbb{R}^{m\times m}$ are effective resistances R_e of the edges of G. Now if we pre- and post-multiply this matrix by $W^{1/2}$, we obtain the transfer-current matrix

$$\Pi = W^{1/2} B^T L^+ B W^{1/2}, \tag{2.32}$$

whose diagonal elements are

$$\Pi_{ee} = w_e b_e^T L^+ b_e = w_e R_e = \mathbb{P}[e \in T] \tag{2.33}$$

where T is any spanning tree of G.

The electrical energy can be viewed as the Euclidean norm of a flow.

$$E_2(f) = ||f||_2^2 = \sum_{e\in E} \frac{f_e^2}{w_e} = f^T W^{-1} f \tag{2.34}$$

and more generally, we define the L^p energy for a flow as:

$$E_p(f) = ||f||_p^p = \sum_{e\in E} \frac{f_e^p}{w_e} \tag{2.35}$$

Among all unit flows from s to t, the electrical flow minimizes the Euclidean energy. This the Thomson's law (TL):

Theorem 5 (Thomson's Law)**.** *For any pair of vertices i, j, the electrical unit $i-j$ flow minimizes the E_2 energy:*

$$R_{ij} = \min_{f \in F_{ij}} E_2(f) \tag{2.36}$$

where F_{ij} is the set of all unit $i-j$ flows.

Proof. The potential vector p of of the unit $i-j$ electrical flow is a solution of $Lp = \delta_i - \delta_j$. Thus, the flow satisfies

$$f = WB^T L^+ (\delta_i - \delta_j) \tag{2.37}$$

and the energy of f is

$$\begin{aligned} E(f) &= f^T W^{-1} f && (2.38)\\ &= (\delta_i - \delta_j)^T L^+ BWW^{-1} WB^T L^+ (\delta_i - \delta_j) && (2.39)\\ &= (\delta_i - \delta_j)^T L^+ LL^+ (\delta_i - \delta_j) && (2.40)\\ &= (\delta_i - \delta_j)^T L^+ (\delta_i - \delta_j) = R_{ij}. && (2.41) \end{aligned}$$

□

A corollary of TL is Rayleigh's monotonicity principle (RMP):

Theorem 6 (Rayleigh's monotonicity principle)**.** *The effective resistance between any pair of vertices is a decreasing function of the resistance of any edge of G, and a convex function with respect to the conductance of the edges.*

There are different interpretations for the various norms: the shortest path distance corresponds to $E_1(f)$, and $E_\infty(f)$ corresponds to the maximum flow in the graph. The E_2 energy is more relevant for our purposes than E_1 or E_∞, because if $f = \delta_s - \delta_t$, adding an edge increases the connectivity between s and t. For E_1 and E_∞, adding an edge e does not improve the energy unless e belongs to the shortest path or improves the edge connectivity, respectively. The E_2 energy provides a smoother quantitative measure that better captures our intuition of how well s and t are connected in a network [Cha+22].

We now link the effective resistance with the connectivity of the (unweighted) graph and the probability of an edge to belong to a spanning tree.

Theorem 7.

$$\mathbb{P}[e \in T] = \frac{\tau(G/e)}{\tau(G)} = R_e \tag{2.42}$$

Proof. Recall that $\overline{L} = L + J/n$ is invertible and $\overline{L}u_1 = u_1$ where $u_1 = (1,...,1)^T$. Recall that from Thm. 2, $\tau(G) = \det \overline{L}/n$. We use the Sherman-Morrison formula together with the contraction deletion formula in Thm. 3:

$$L(G\backslash e) = L - b_e b_e^T \tag{2.43}$$

$$\tau(G\backslash e) = \frac{1}{n}\det\left(\overline{L} - b_e b_e^T\right) \tag{2.44}$$

$$\mathbb{P}[e \notin T] = \frac{\det\left(\overline{L} - b_e b_e^T\right)}{\det \overline{L}} \tag{2.45}$$

$$\mathbb{P}[e \in T] = 1 - \frac{\det\left(\overline{L} - b_e b_e^T\right)}{\det \overline{L}} = 1 - \det \overline{L}\frac{1 - b_e^T \overline{L}^{-1} b_e}{\det \overline{L}} = b_e^T L^+ b_e = R_e \tag{2.46}$$

Since we assume L connected, $\ker L$ is spanned by u_1. The restriction of L and $\overline{L}$ to the orthogonal of this space is the same. □

If the graph is weighted, the formula becomes

$$\mathbb{P}[e \in T] = w_e R_e. \tag{2.47}$$

Thus, the local mesure of connectivity $\mathbb{P}[e \in T]$ is exactly (for an unweighted graph) the effective resistance of the edge. On a global scale, the connection between connectivity and effective resistance is expressed through the total resistance R_G of the graph, which is the sum of all the effective resistance over all the pairs of vertices:

$$R_G = \frac{1}{2}\sum_{i,j=1}^{n} R_{ij} = n\mathrm{Tr}L^+ = n\sum_{j=2}^{n}\frac{1}{\lambda_j} \tag{2.48}$$

R_G is primarily determined by the magnitude of the small eigenvalues of G.

$$\frac{n}{\lambda_2} \leq R_G \leq \frac{n(n-1)}{\lambda_2} \tag{2.49}$$

(2.49) links the smallest nonzero eigenvalue of L (known as the *algebraic connectivity* of the graph and the corresponding eigenvector u_2 as the Fiedler vector) with the total resistance of G. We can also establish a connection between the local effective resistance R_b of an edge b with the eigenvalues of L; suppose, without loss of generality, that we work in an orthonormal eigenbasis $(u_1,...,u_n)$ of L, where the coordinates of b are expressed in this basis:

$$R_b = b^T L^+ b = \sum_{j=2}^{n}\frac{b_j^2}{\lambda_j} \tag{2.50}$$

The dominant term in (2.48) and (2.49) corresponds to the term related to λ_2. Therefore, the algebraic connectivity can be used to approximate the effective resistance and serve as an alternative measure of robustness. Indeed, λ_2 is related to the Cheeger constant of the graph, which provides a measure of conductance of a subset of nodes [Bat+13; Moh89].

Let ∂S denote the boundary of a subset $S \subset V$ (the number of edges with one endpoint inside S and the other outside S) and let $D(S)$ denote the sum of the degrees of all nodes in S. Define

$$\phi(S) = \frac{|\partial(S)|}{\min(D(S), D(\bar{S}))}, \tag{2.51}$$

and

$$\phi_G = \min_{S \subset V} \phi(S). \tag{2.52}$$

A discrete version of the Cheeger inequality [Bat+13] is given by:

$$2\phi_G \geq \lambda_2(\mathscr{L}) \geq \frac{\phi_G^2}{2}. \tag{2.53}$$

Here, $\lambda_2(\mathscr{L})$ is the smallest nonzero eigenvalue of the normalized Laplacian $\mathscr{L}$ of the graph. While it differs from λ_2 of L, both measure similar structural properties of the graph.

2.2.5 Random walks

The small book of Doyle & Snell [DS84] is the perfect introduction to random walks and electric networks. We briefly summarize some definitions and properties from the book to connect hitting and *commute time* with effective resistance and our definition of connectivity.

The simple random walk on the vertices of G is defined as the random sequence $(X_t)_{t\geq 0}$, where X_t takes his values in the set of vertices V. At each discrete instant t, if $X_t = v_i$, the next vertex is chosen uniformly at random among the adjacent vertices ∂v_i of v_i. The transition probability kernel is thus given by:

$$p_{ij} = \mathbb{P}\left[X_{t+1} = v_j | X_t = v_i\right] = \begin{cases} 1/d_i & \text{if } (v_i, v_j) \in E \\ 0 & \text{otherwise} \end{cases} \tag{2.54}$$

where d_i is the degree of v_i. $(X_t)_{t\geq 0}$ is a Markov chain with state space V and matrix of transition $(p_{ij})_{ij}$. The Markov property holds and since G is connected, the random walk is irreductible. By the Perron-Frobenius theorem, there exists a unique stationary probability distribution given by

$$\pi_i = \frac{d_i}{2m}, \ \forall v_i \in V. \tag{2.55}$$

If G is non-bipartite, the stationary distribution is also the limit distribution:

$$\lim_{t \to +\infty} \mathbb{P}[X_t = v_j | X_0 = v_i] = \pi_j. \tag{2.56}$$

Let $\lambda = \max\{|\lambda_2|, |\lambda_n|\}$, then for a random walk starting at v_i,

$$\left|\mathbb{P}[X_t = j | X_0 = i] - \pi_j\right| \leq \lambda^k \sqrt{d_j / d_i}. \tag{2.57}$$

The mixing rate measures how quickly the random walk converges:

$$\mu = \lim_{t \to +\infty} \sup \max_{i,j} \left| \left(P^k\right)_{ij} - \frac{d_j}{2m} \right|^{1/k}. \tag{2.58}$$

The mixing time for an expander graph is $O(\ln n)$ steps only. If G is non-bipartite, the mixing rate is exactly λ [Cha+89].

The hitting time H_{ij} is the expected first hitting time in v_j, starting from v_i, and the commute time κ_{ij} between vertices v_i and v_j is the expected time for a random walk starting at v_i to reach v_j and return to v_i.

$$\kappa_{ij} = H_{ij} + H_{ji}. \tag{2.59}$$

Theorem 8 (Commute time and resistance distance)**.** *The commute time between vertices v_i and v_j in an unweighted graph G, is equal to:*

$$\kappa_{ij} = 2mR_{ij}. \tag{2.60}$$

In an weighted graph,

$$\kappa_{ij} = \Delta R_{ij}, \tag{2.61}$$

where $\Delta = \sum_{i=1}^{n} d_i$ is the sum of all the degrees of the nodes of G (sometimes called the volume of G, which should not be confused with the volume of the Laplacian matrix).

In particular, if v_i and v_j are the endpoints of an edge e from G:

$$R_e = \frac{\kappa_e}{\Delta}. \tag{2.62}$$

Two different proofs are given in Chandra [Cha+89] and Fouss [Fou+07].

The cover time $\kappa(n)$ is the expected number of steps to visit all the nodes. In 1989, Aldous proved that there exists a constant $c > 0$ such that for every graph with n vertices, $\kappa(n) \geq cn \ln n$, and it is a conjecture that the graph with the smallest cover time is the complete graph [Cha+89].

Let $R_{\max}$ the maximum of all the effective resistance between all pairs of vertices (it is different from the total resistance of the graph).

$$mR_{\max} \leq \kappa(n) \leq O(m \ln n) R_{\max}. \tag{2.63}$$

The proof is given in [Cha+89].

Other interesting interpretations related to effective resistance are described in Appendix A.2.

2.2.6 A step conclusion

We conclude this subsection by summarizing the results of the section up to now: the connectivity of a graph, measured globally as the number of spanning trees, is equal or proportional to the volume of the Lapla-

cian, the total effective resistance and the cover time of the graph. On a local scale, the probability of an edge to belong to a spanning tree is equal or proportional to the effective resistance of the edge and to the commute time between its endpoints. All these notions are one and the same and justify our choice of the number of spanning trees as the definition of the connectivity. The quantity we are going to maximize in this part (volume of the Laplacian) is therefore the same as in the first part (volume of the covariance matrix of the database).

With all these definitions relating to connectivity established, we can now specify how they are used in the related works.

[BH09] characterizes robustness by the number of spanning trees and proposes increasing it by adding a very small number of edges to a graph. They solve a convex optimization program to achieve this for specific families of regular graphs. [SBG23] uses the algebraic connectivity to identify the most robust graph for a given number of nodes and edges, applying this criterion to find such graphs for small values of n and m. [AE12] proposes the second-largest eigenvalue as a measure of robustness and a step-wise algorithm to add edges that enhance robustness. For these first three references, robustness maximization is applied only on specific families or regular graphs. [MSN10] uses the second-largest eigenvalue or algebraic connectivity to increase robustness by adjusting the weights of a small number of edges, making this method suitable only for weighted graphs.

Edges rewiring methods based on different strategies such as spectral radius, algebraic connectivity, effective resistance or nodes degree are proposed in [CA16; Wan+14; Li+18; GBS08]. The objective is to replace the current graph with a more robust one by adding or swapping a few edges, either increasing the number of edges slightly or keeping it the same. The complexity of algorithms based on spectral methods is at least cubic in time, and quadratic when using degrees, but at the price of lower performance. All of these references demonstrate the use of different connectivity criteria to improve robustness by adding or exchanging a small number of edges and do not provide sparsification methods.

2.3 Graph sparsification

2.3.1 Generalities

Sparsification aims to extract a sparse sub-graph from an initial graph in order to spare storage space, reduce complexity and accelerate operations carried out on the graph [LS18; Vis13]. Unlike coarsening, sparsification is a graph reduction method in which all nodes are retained and only edges are removed, with the objective of preserving certain relevant properties of the initial graph, according to specific metrics.

The most efficient class of methods for sparsifying is spectral sparsification, developped over the past two decades by Spielman, Srivastava, Teng and Batson [BSS12; Bat+13; ST11], following the works of Benzcur and Karger [KS96] on minimum cuts. The goal of spectral sparsification is to reduce the number of edges while preserving the spectrum of the Laplacian, leading to the notion of spectral similarity. Spectral similarity preserves both the Laplacian quadratic form and the minimum cuts of the graphs. Our goal is different, because unlike spectral methods, we seek to preserve connectivity, which requires modifying the spectrum.

Even though our approach differs from spectral methods, the algebraic tools involved in both share many similarities. Classical metrics describing the graph structure or geometry include degree distribution, shortest path between any pair of vertices, diameter, global eccentricity, vertex eccentricity, betweenness, closeness

or Katz centrality, local or global clustering coefficients, number of communities, PageRank, min-cut / max-flow, Cheeger constant, number of spanning trees, or spectral related quantities like Laplacian quadratic form, algebraic connectivity or effective resistance [Che+23]. All these metrics represent various aspects of connectivity in the graph, with either global or local measures. The connectivity we defined in Thm. 1 and the spectral similarity-based methods we are about to present capture most of these quantities and establish links between them, justifying our focus on spectral sparsification. For a complete and comprehensive review of other sparsification methods, We refer the reader to [Che+23].

2.3.2 Different notions of similarity

Let $G = (V, E, w)$ an undirected and weighted graph. A subgraph H of G with the same vertex set V satisfies the ϵ-spectral similarity property [Bat+13] relative to G if, and only if for all $x \in \mathbb{R}^n$,

$$(1-\epsilon)x^T L(G)x \leq x^T L(H)x \leq (1+\epsilon)x^T L(G)x. \tag{2.64}$$

A spectral sparsifier is an algorithm that constructs a sparse subgraph from a initial dense graph, whose spectrum is ϵ-similar. In words, a spectral sparsifier preserves the quadratic Laplacian form

$$Q_L(x) = x^T Lx = \sum_{i\sim j} w_{ij}(x_i - x_j)^2, \tag{2.65}$$

up to a factor ϵ. In particular, (2.64) implies the spectrum similarity property; for all $i = 1, .., n$,

$$(1-\epsilon)\lambda_i(G) \leq \lambda_i(H) \leq (1+\epsilon)\lambda_i(G). \tag{2.66}$$

Spectral similarity also implies cut similarity property [Bat+13]; for any subset S of vertices, we define the cut of S in G as the weight of its boundary:

$$\text{cut}_G(S) = \sum_{i\in S, j\notin S} w_{ij}. \tag{2.67}$$

We say that G and H are ϵ-cut similar if, for all $S \subset V$,

$$(1-\epsilon)\text{cut}_G(S) \leq \text{cut}_H(S) \leq (1+\epsilon)\text{cut}_G(S). \tag{2.68}$$

If $S \subset V$ and x is the indicator vector of S, then $Q_L(x) = \text{cut}(S)$ and cut similarity is a special case of spectral similarity for binary coordinates vectors x. This notion of cut similarity was developped by Benczur and Karger to propose fast algorithms for solving the min cut max flow problem, where they also proved that every graph is cut-similar to a sub-graph with average degree of $O(\ln n)$ [Bat+13; BK96].

Spielman and Teng [ST11] presented the first algorithm for constructiong a spectral sparsifier in nearly linear time, with $O\left(n \ln^c n/\epsilon^2\right)$ operations and a number of edges in $O(n \ln^d n/\epsilon^2)$, nearly linear in the number of nodes. Then Spielman and Srivastava [SS11] reduced the number of edges to $O\left(n \ln n/\epsilon^2\right)$ in $O(m)$ time. The algorithm is random and designed for weighted graphs: it modifies weights of the edges based on effective resistance, to preserve the eigenvalues. Batson, Spielman and Srivastava proposed a deterministic algorithm to

produce a linear sparsifier with $O(n/\epsilon^2)$ edges, but quartic time complexity. Eventually, Lee and and Sun [LS18] presentd a random almost-linear time algorithm for constructing a linear-sized spectral sparsifier.

Two graphs on the same vertex set are ϵ-distance similar if the distance $d(u,v)$ between any pair of vertices is not modified by a factor greater than $1\pm\epsilon$. If the graph is unweighted, $d(u,v)$ is the number of edges between u and v along the shortest path and if the graph is weighted, $d(u,v)$ is the weighted sum of the shortest path:

$$(1-\epsilon)d_G(u,v) \leq d_H(u,v) \leq (1+\epsilon)d_G(u,v), \tag{2.69}$$

for all $u,v \in V$. If H is a subgraph of G, the left inequality is always satisfied. A t-spanner is a subgraph for which the right inequality is verified for $t = 1+\epsilon$. If H is a tree, the spanner is a low stretch spanning tree.

2.3.3 sparsification by effective resistance

We now present the random sampling algorithm of Spielman and Srivastava [SS11], which simultaneously solves the sparsification problem formulated by Spielman and Teng and the minimum cut problem addressed by Karger.

A number k of edges are selected in one shot from G, to form the sparse subgraph H, with a probability p_e proportional to the effective resistance. The weight of each selected edge is multiplied by $1/p_e$ so that $\mathbb{E}[L(H)] = L(G)$. The idea is that edges with the highest resistance are more crucial than others and should be selected.

Theorem 9 (Spielman-Srivastava [SS11])**.** *Let $G(V,E,w)$, $0<\epsilon\leq 1$ and $k = \lfloor 8n\ln n/\epsilon^2 \rfloor$.*
Let H be a subgraph of G with k edges selected with probability $p_e = w_e R_e$ and weight $\tilde{w}_e = w_e/(kp_e)$.
Then with probability at least $1/2$, H is a $(1+\epsilon)$-spectral sparsifier of G.

The k samples are chosen independently with replacement, summing the weights if an edge is selected more than once.

Proof. The proof is broken down into two parts: first, we demonstrate that if the transfer current matrices of G and H are close in the spectral norm, then both graphs are spectrally similar. Second, we show that the two matrices are indeed close, using a concentration inequality due to Rudelson and Vershynin. Let

$$L(G) = BW_GB^T \text{ and } L(H) = BW_HB^T, \tag{2.70}$$

where $W_H = W^{1/2}SW^{1/2}$ and $S \in \mathbb{R}^{m\times m}$ is the random diagonal matrix with diagonal entries $S_{ee} = 0$ if e is not sampled in H and $S_{ee} = j/(kp_e)$ if edge e is sampled j times. Therefore, the weight of e in H is $\tilde{w}_e = S_{ee}w_e$ and

$$L(H) = BW^{1/2}SW^{1/2}B^T. \tag{2.71}$$

The scaling of the weights implies successively $\mathbb{E}[\tilde{w}_e] = w_e$ because k independent samples are taken each with probability p_e, $\mathbb{E}[S] = I$ and $\mathbb{E}[L(H)] = L(G)$. Now let Π and Π_H the respective transfer current matrix and suppose that for $\epsilon > 0$,

$$||\Pi_H^2 - \Pi^2||_2 = ||\Pi S\Pi - \Pi^2||_2 < \epsilon. \tag{2.72}$$

For any nonzero vector $y \in \mathrm{Im}(BW^{1/2})$,

$$\left|y^T \Pi(S-I)\Pi y\right| = \left|y^T (S-I) y\right| \tag{2.73}$$

$$= \left|x^T BW^{1/2} SW^{1/2} B^T x - x^T BWB^T x\right| \tag{2.74}$$

$$= \left|x^T L(H) x - x^T L(G) x\right|, \tag{2.75}$$

with $y = BW^{1/2}x$. Likewise, $y^T y = x^T L_G x$, so that

$$\sup_{y \in \mathrm{Im}(BW^{1/2})} \frac{\left|y^T \Pi(S-I)\Pi\right|}{y^T y} = \sup_{x: BW^{1/2}x \neq 0} \frac{\left|x^T L(H) x - x^T L(G) x\right|}{x^T L(G) x}. \tag{2.76}$$

Thus, $||\Pi_H^2 - \Pi^2||_2 < \epsilon$ implies H is ϵ-spectrally similar to G.

The concentration inequality we are going to use is the following:

Theorem 10 (Rudelson & Vershynin [RV07])**.** *Let $\mathbb{P}$ a probability distribution over $\Omega \subset \mathbb{R}^n$, such that $\sup_{y\in\Omega} ||y|| \leq C$ and $\mathbb{E}[yy^T] \leq 1$.*
Let $(y_1, ..., y_k)$ an independent sample from $\mathbb{P}$. Then,

$$\mathbb{E}\left[\left|\left|\frac{1}{k}\sum_{i=1}^{k} y_i y_i^T - \mathbb{E}[yy^T]\right|\right|\right] \leq \min\left(8C\sqrt{\frac{\ln k}{k}}, 1\right). \tag{2.77}$$

The algorithm selects edges independenlty with probability $p_e \propto w_e R_e$. Since $\mathrm{Tr}\,\Pi = \sum_e w_e R_e = n-1$, let's set

$$p_e = \frac{1}{n-1} w_e R_e. \tag{2.78}$$

Let Π_e be the column of Π corresponding to an edge e and $y = \Pi_e/\sqrt{p_e}$. Let $\mathbb{P}$ be the probability distribution selecting y with probability p_e. Let $y_1, ..., y_k$ k be vectors drawn independently with replacement from $\mathbb{P}$. Then

$$\Pi S \Pi = \sum_e S_{ee} \Pi_e \Pi_e^T = \frac{1}{k}\sum_e \frac{j_e}{k p_e} \Pi_e \Pi_e^T = \frac{1}{k}\sum_e j_e \frac{\Pi_e}{\sqrt{p_e}} \frac{\Pi_e^T}{\sqrt{p_e}} = \frac{1}{k}\sum_{i=1}^{k} y_i y_i^T \tag{2.79}$$

and

$$\mathbb{E}[yy^T] = \sum_e p_e \frac{1}{p_e} \Pi_e \Pi_e^T = \Pi^2 = \Pi. \tag{2.80}$$

So $\left|\left|\mathbb{E}[yy^T]\right|\right| = ||\Pi||_2 = 1$. Then,

$$\frac{1}{\sqrt{p_e}} ||\Pi_e||_2 = \frac{1}{\sqrt{p_e}} \sqrt{\Pi_{ee}} = \sqrt{\frac{n-1}{w_e R_e}} \sqrt{w_e R_e} = \sqrt{n-1}, \tag{2.81}$$

taking $k = 4C^2 n \ln n/\epsilon^2$ gives, by the previous theorem,

$$\mathbb{E}\left[||\Pi S \Pi - \Pi^2||\right] = \mathbb{E}\left[\left|\left|\frac{1}{k}\sum_{i=1}^{k} y_i y_i^T - \mathbb{E}[yy^T]\right|\right|_2\right] \leq \min\left(8C\sqrt{\frac{\ln k}{k}}, 1\right) \tag{2.82}$$

and by Markov's inequality, $||\Pi S\Pi - \Pi||_2 \leq \epsilon$ with probability at least $1/2$.

□

The number of edges of the sparsifier is in $O(n \ln n/\epsilon^2)$.

The computation of the approximate effective resistance of the edges uses the Spielman-Teng nearly linear-time solver [ST04; ST06] and the Johnson-Lindenstrauss Lemma [JL84], as stated in the following theorem.

Theorem 11. *Let (G, V, w), $\epsilon > 0$ and $k = \lfloor 24 \ln n/\epsilon^2 \rfloor$. There exists a matrix Z of size $k \times n$ such that, with probability at least $1 - 1/n$,*

$$(1-\epsilon)R_{ij} \leq \left|\left|Z(\delta_i - \delta_j)\right|\right|^2 \leq (1+\epsilon)R_{ij}, \tag{2.83}$$

for every pair of vertices i, j.

The algorithm is run in time $O(m \ln n/\epsilon^2)$. As $Z(\delta_i - \delta_j)$ is the difference of two columns of Z, an approximate effective resistance can be evaluated in $O(\ln n)$ time for any given pair of vertices.

2.3.4 Twice Ramanujan sparsifiers

The previous algorithm is random and requires $O(n \ln n)$ edges. The next algorithm (BSS), developed by Batson, Spielman and Srivastava [BSS12] is deterministic, designed for weighted graphs, and only needs $O(n)$ edges to preserve the spectral similarity. However, its complexity is $O(n^3 m)$ operations.

The first step in the BSS algorithm involves re-scaling the Laplacian to show that the sparsification process can be performed on the identity matrix without loss of generality.

Let $V = L^{+/2} B W^{1/2} \in \mathbb{R}^{n \times m}$. Then,

$$\sum_{i=1}^{m} v_i v_i^T = VV^T = L^{+/2} B W B^T L^{+/2} \tag{2.84}$$

$$= L^{+/2} L L^{+/2} = \mathrm{Id}_F, \tag{2.85}$$

where $F = ImL \simeq \mathbb{R}^{n-1}$. From this point, we work with m vectors $v_i \in \mathbb{R}^n$ that satisfy the isotropy condition:

$$\sum_{i=1}^{m} v_i v_i^T = I_n, \tag{2.86}$$

Let $S \in \mathbb{R}^{m \times m}$ the diagonal matrix where $S_{ii} = s_i$ are the new weights after sparsification. Then, as in the previous section,

$$L(H) = B W^{1/2} S W^{1/2} B^T. \tag{2.87}$$

Theorem 12 (Batson, Srivastava, Spielman [BSS12])**.** *Let $\delta > 1$ and $v_1, \ldots, v_m$ be vectors of $\mathbb{R}^n$ such that $\sum_{i=1}^m v_i v_i^T = I_n$. Then, there exists fewer than δn scalars $s_i \geq 0$ such that*

$$I_n \leq \sum_{i=1}^{m} s_i v_i v_i^T \leq \kappa I_n \tag{2.88}$$

where

$$\kappa = \frac{\delta + 1 + 2\sqrt{\delta}}{\delta + 1 - 2\sqrt{\delta}}. \tag{2.89}$$

This theorem guarantees the existence of a sub-family of δn vectors extracted from the initial m vectors, such that the induced submatrix sprectrum closely approximate the initial spectrum up to a multiplicative constant κ (close to 1). From a graph perspective, this implies the existence of a subgraph with δn edges that is spectrally similar to the original graph, with δ as close to 1 as desired, and with spectral similarity up to a constant κ.

We previously established that if V and VSV^T satisfy the spectral similarity condition (2.88), then L and $L(H)$ are also spectrally similar, due to the Courant-Fisher theorem.

The algorithm that performs sparsification is greedy and utilizes potential barrier functions, akin to discrete Stieljes-Cauchy transforms, to bound the eigenvalues as columns are iteratively concatenated. At each iteration, the algorithm selects a vector v, updates the current matrix $A_{t+1} = A_t + svv^T$ and determines the weight s in such a way that the spectrum of A is altered as little as possible.

Before defining the lower and upper barrier functions, we digress to provide some intuition behind the proof.

Intuition behind the algorithm

This paragraph uses some results from rank-one perturbation theory (c.f. Appendix A.1). At a given iteration, concatenating a column is equivalent to adding a rank-one operator vv^T to the current symmetric matrix, let's say A. This transformation shifts every eigenvalue λ_i of A to a new eigenvalue $\tilde{\lambda}_i = \lambda_i + \mu_i$ of $A + vv^T$, where μ_i is a quantity between 0 and the norm of v (we suppose that $||v||_2 = 1$). We can prove that μ_i is positively correlated to $\left(v^T.u_j\right)^2$. The secular equation links the old and new eigenvalues:

$$1 + \sum_{j=1}^{m} \frac{\left(v_i^T.u_j\right)^2}{\lambda_j - x} = 0. \tag{2.90}$$

where we assume, without loss of generality, that we work in the orthonormal eigenvectors basis $(u_j)_j$. In that case, $\left(v^T.u_j\right)^2 = v_j^2$. The poles of this rational function correspond to the eigenvalues of A and its zeros correspond to the eigenvalues of $A + vv^T$.

Now, imagine that the vectors v are chosen randomly and uniformly from all available vectors. Then,

$$\mathbb{E}\left[\left(v^T.u_j\right)^2\right] = \frac{1}{m}\sum_{i=1}^{m}\left(v_i^T.u_j\right)^2 \tag{2.91}$$

$$= \frac{1}{m}u_j^T\left(\sum_i v_i v_i^T\right)u_j = \frac{||u_j||^2}{m} = \frac{1}{m} \tag{2.92}$$

In words, a random vector increases all eigenvalues by the same average quantity $1/m$. After a certain number of iterations, we expect that this average behavior will lead to eigenvalues that are very close to each other and a condition number close to 1. Under this assumption, note that the v_i's are deterministic and it is the choice among them that is random.

Of course, the algorithm does not select the vectors randomly and nothing proves that a vector is close to the "average" vector $v = (1/\sqrt{m})\sum_j u_j$. Instead, the algorithm selects a sequence of vectors that achieves the desired average behavior by carefully adjusting the weights s_i at each iteration. The control of the eigenvalues is achieved by keeping them within two barriers. The lower barrier pushes the eigenvalues forward and the upper barrier ensures that they do not move too far. As these barriers advance steadily, and by maintaining that the total repulsion at every step is bounded, we can guarantee that there always exists a vector satisfying the constraints at each iteration.

Barrier functions

Definition 2. *Let $A \in \mathbb{R}^{n\times n}$ a symmetric matrix with eigenvalues $\lambda_1, ..., \lambda_n$. The lower and upper barrier functions are respectively defined by*

$$\begin{cases} \phi_-(A,x) = \mathrm{Tr}(A - xI)^{-1} = \sum_{i=1}^{n} \dfrac{1}{\lambda_i - x} \\ \\ \phi_+(A,x) = \mathrm{Tr}(xI - A)^{-1} = \sum_{i=1}^{n} \dfrac{1}{x - \lambda_i} \end{cases} \tag{2.93}$$

where $\phi_+(A,.)$ is defined for $x > \lambda_n$ and $\phi_-(A,.)$ is defined for $x < \lambda_1$.

The function ϕ_+ is decreasing, ϕ_- is increasing and both functions are convex.

After concatenating v, the new lower barrier function becomes:

$$\phi_-(A + vv^T, x) = \mathrm{Tr}(A + vv^T - xI)^{-1} = \sum_{i=1}^{n} \frac{1}{\lambda_i(A + vv^T) - x} = \sum_{i=1}^{n} \frac{1}{\widetilde{\lambda}_i - x}. \tag{2.94}$$

The barrier potential functions measure how close the eigenvalues are to x. They reflect the locations of all the eigenvalues simultaneously and give a bound on the smallest and largest eigenvalue. For example, $\phi_+(A,x) \leq 1$ implies that no λ_i is within a distance 1 from x. ϕ_+ represents the total repulsion of the eigenvalues from the upper barrier at x (c.f. Fig. 2.3.4). Small values of the potential indicate that the eigenvalues of A do not cluster near x.

If $\phi_-(A,x) \leq \epsilon$ and $x < \lambda_n$, then

$$\forall i,\ \lambda_i > x + 1/\epsilon. \tag{2.95}$$

Proof. (of Th. 12). The algorithm selects a value ϵ_L on the x-axis such that, for all iterations t, $\phi_-(A_t,x) \leq \epsilon_L$. An initial value $x = l_0$ is chosen on the x-axis, along with an increment δ_L (these three constants are fixed throughout all iterations). At each iteration, v and x must be chosen so that :

$$\forall t,\ \phi_-(A_{t+1}, x + \delta_L t) \leq \phi_-(A_t, x) \leq \epsilon_L,\ x = l_0 + \delta_L t, \tag{2.96}$$

while ensuring that

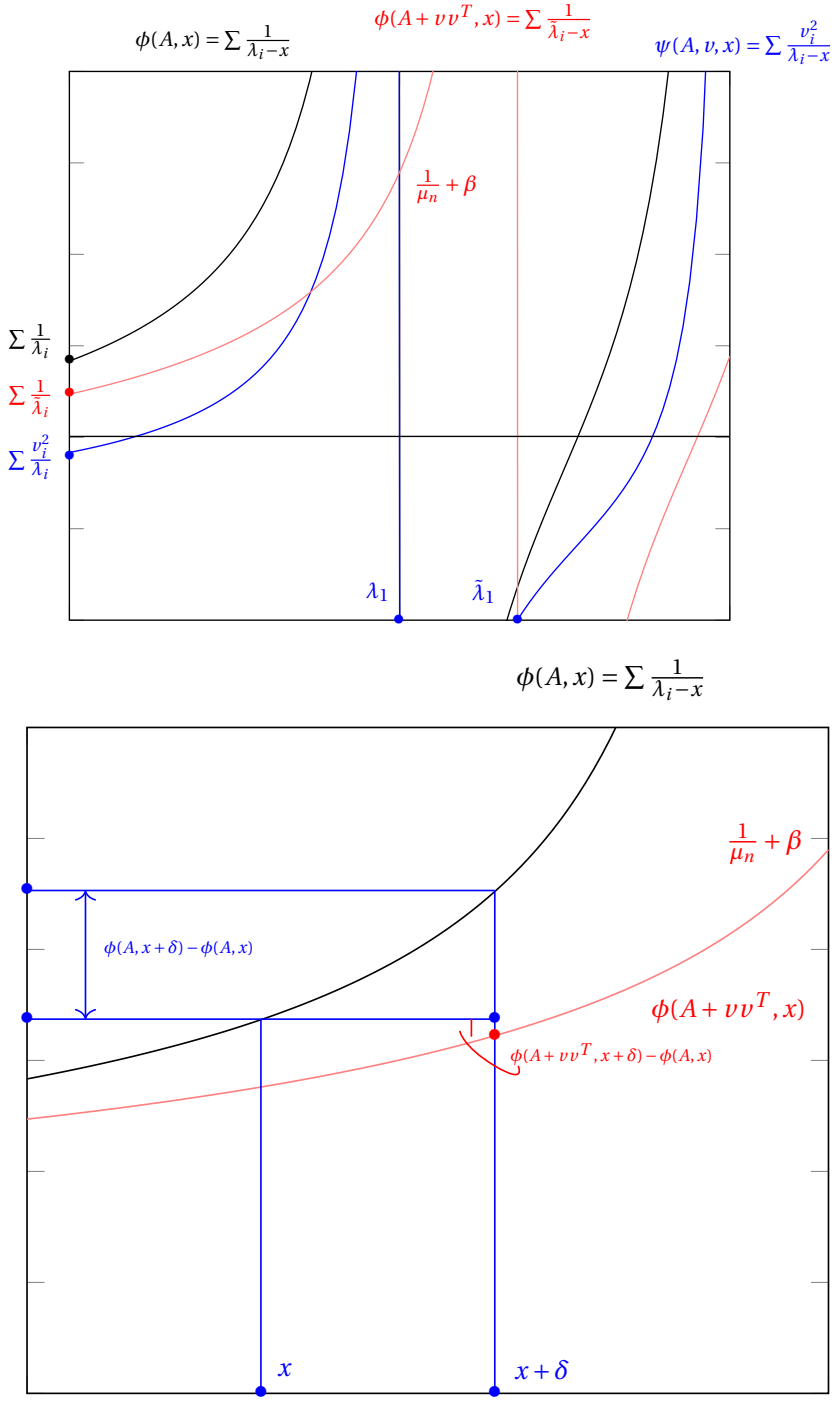

Figure 2.3 – ϕ_- barrier functions and their respective positions for a definite positive matrix A with smallest eigenvalue $\lambda_1 > 0$.

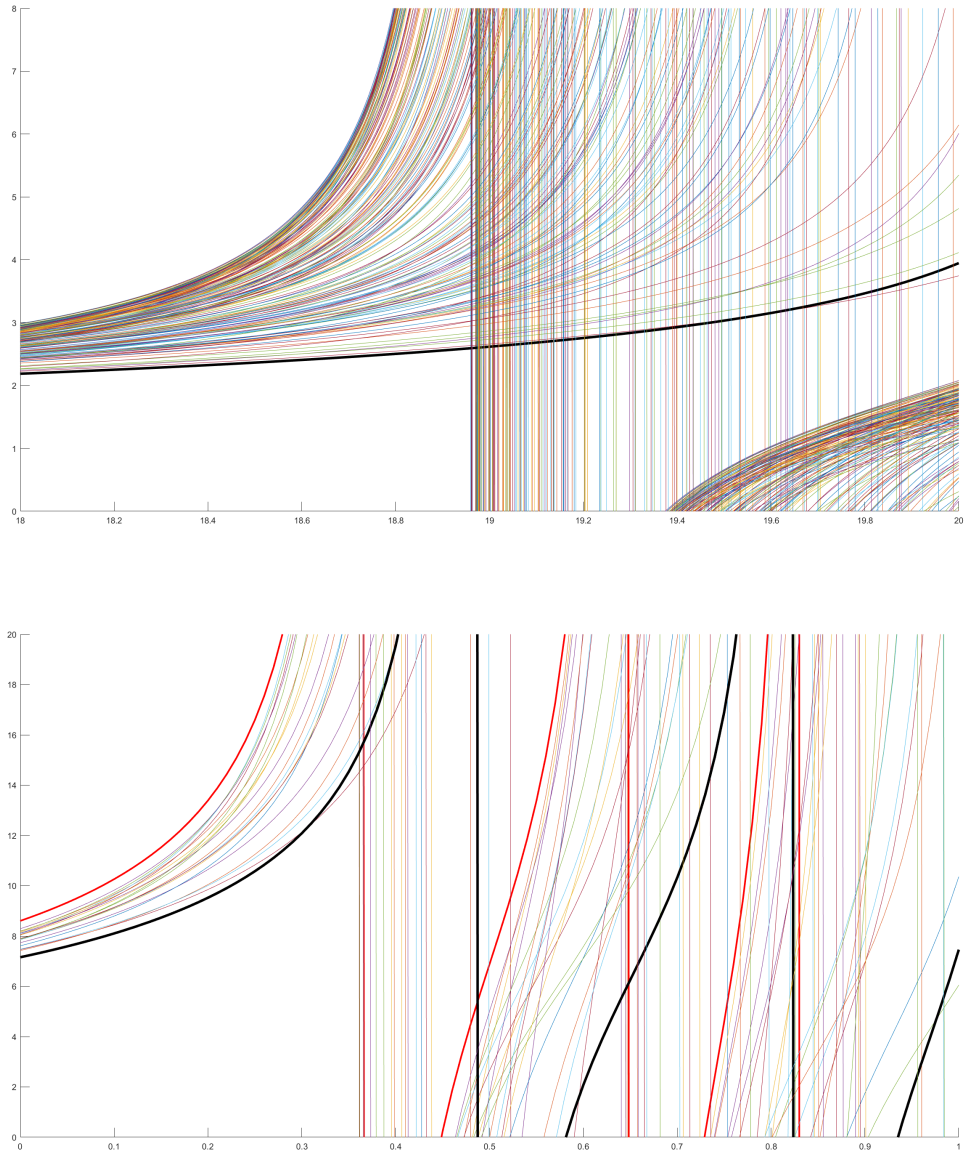

Figure 2.4 – Two examples of ϕ_- and $R(b,.)_-$ functions in the neighborhood of the smallest eigenvalue λ_1. The red curve is the ϕ_- function before selecting an edge b and the black curve is the modified function after addition of the rank-one bb^T operator. Other color curves show the different $R(b,.)_-$ functions for all available edges b. Matrix $A = BB^T$, where B is Gaussian, then its columns are normalized. Top: $k = d = 100$, $m = 400$. Bottom: $k = d = 10$, $m = 40$.

$$\lambda_n(A_t) > l_0 + \delta_L t. \tag{2.97}$$

Similarly, $\epsilon_U, \delta_U, u_0$ are defined such that:

$$\forall t,\ \phi_+(A_{t+1}, x+\delta_U t) \le \phi_+(A_t, x) \le \epsilon_U,\ x = u_0 + \delta_U t, \tag{2.98}$$

$$\lambda_1(A_t) < u_0 + \delta_U t. \tag{2.99}$$

The challenge is to find v such that this condition is satisfied at each iteration, for both the lower and upper barriers. If this is achieved, then after $t = \rho n$ iterations,

$$\frac{\lambda_1(A_t)}{\lambda_n(A_t)} \le \frac{u_0 + \rho n \delta_U}{l_0 + \rho n \delta_L} = \kappa, \tag{2.100}$$

for a well-chosen set of constants.

For the lower barrier, the idea is to push the barrier by a quantity δ_L at each iteration, ensuring that $\phi_-(A + vv^T, x+\delta_L)$ remains below $\phi_-(A,x)$, with δ_L as a lower bound for the increase of the smallest eigenvalue.

We define the following auxiliary functions:

$$R_-(A,v,x) = v^T(A - xI)^{-1}v = \sum_{i=1}^{m} \frac{v_i^2}{\lambda_i - x} \tag{2.101}$$

$$R_+(A,v,x) = v^T(xI - A)^{-1}v = \sum_{i=1}^{m} \frac{v_i^2}{x - \lambda_i} \tag{2.102}$$

$$d(A,x,\delta) = \phi(A, x+\delta) - \phi(A,x), \tag{2.103}$$

and

$$\mathbb{L}(A,v,x,\delta) = \frac{R'_-(A,v,x+\delta)}{d(A,x,\delta)} - R_-(A,v,x+\delta) \tag{2.104}$$

$$\mathbb{U}(A,v,x,\delta) = \frac{R'_+(A,v,x+\delta)}{-d(A,x,\delta)} + R_+(A,v,x+\delta). \tag{2.105}$$

Here, R_+ and R_- are the discrete Cauchy-Stieljes transform of the $\sum_i v_i^2 \delta_{\lambda_i}$ empirical measure, a crucial tool in random matrix theory. They appear in the secular equation (omitting the 1 constant): their poles are the eigenvalues of the initial matrix and their zeros (when the additive constant 1 is included) correspond to the new eigenvalues after the vector v is added. R'_+ and R'_- are their derivates with respect to x.

The next property is central to the algorithm (refer to [BSS12] for details of the proof); if $\phi_-(A,.)$ is small enough in x and $\mathbb{L}$ is greater than $1/s$, then at $x+\delta_L$, $\phi_-(A + svv^T,.)$ remains smaller than $\phi_-(A,x)$ and the lower bound of the smallest eigenvalue increases by δ_L.

Lemma 1. *Suppose $\lambda_n(A) > x$ and $\phi_-(A,x) \le 1/\delta_L$. Then if $\mathbb{L}(A,v,x) \ge 1/s$,*

$$\phi_-(A + svv^T, x+\delta_L) \le \phi_-(A,x), \tag{2.106}$$

$$and\ \lambda_n(A + svv^T) > x + \delta_L. \tag{2.107}$$

Similarly,

Lemma 2. *Suppose* $\lambda_1(A) < x$. *Then if* $\mathbb{U}(A, v, x) \leq 1/s$,

$$\phi_+(A + svv^T, x + \delta_U) \leq \phi_+(A, x), \tag{2.108}$$

$$and\ \lambda_1(A + svv^T) < x + \delta_U. \tag{2.109}$$

To find a vector v satisfying both $\lambda_1(A) < u$, $\lambda_n(A) > l$, $\phi'(A, u) \leq \epsilon_U$, $\phi(A, l) \leq \epsilon_L$, $\mathbb{L}(v) > 1/s$, $\mathbb{U}(v) < 1/s$ it suffices that

$$0 \leq \frac{1}{\delta_U} + \epsilon_U \leq \frac{1}{\delta_L} - \epsilon_L \tag{2.110}$$

Two important equations in the resolution are:

$$\phi_-(A + svv^T, x + \delta_L) - \phi_-(A, x) = d(A, x, \delta_L) - \frac{R'_-(A, v, x + \delta_L)}{1/s + R_-(A, v, x + \delta_L)}, \tag{2.111}$$

$$\tag{2.112}$$

$$\phi_+(A + svv^T, x + \delta_U) - \phi_+(A, x) = d(A, x, \delta_U) + \frac{R'_+(A, v, x + \delta_U)}{1/s - R_+(A, v, x + \delta_U)}. \tag{2.113}$$

These are derived from the Sherman-Morrison formula and the convexity of the ϕ functions. The principal argument of the proof is the following:

$$\sum_i \mathbb{L}(A, v_i) \geq \sum_i \mathbb{U}(A, v_i). \tag{2.114}$$

The sum above is a deterministic average evaluated on all selected vectors. Although it may not be possible to prove the existence of one particular vector satisfying all the conditions, these conditions are always valid on average. If (2.114) is satisfied, then there exists v_i that meets all the constraints.

Condition (2.110) also allows to prove the existence (thanks to the isotropy condition) of a weight s such that

$$\mathbb{U}(v) < 1/s < \mathbb{L}(v). \tag{2.115}$$

Eventually, the constants can be chosen such as:

$$\delta_L = 1, \delta_U = (\sqrt{\rho} + 1)/(\sqrt{\rho} - 1), \tag{2.116}$$

$$\epsilon_L = 1/\sqrt{\rho},\ \epsilon_U = (\sqrt{\rho} - 1)/(\rho + \sqrt{\rho}), \tag{2.117}$$

$$l_0 = -n/\epsilon_L,\ u_0 = n/\epsilon_U. \tag{2.118}$$

The key point is that there must exist, at each iteration, a vector v (and a point x) such that the curve of $\phi(A+vv',.)$ lies sufficiently below that of $\phi(A,.)$ in an neighborhood of x. More precisely, in this neighborhood, the difference $d(A,x)=\phi(A,x+\delta)-\phi(A,x)$ must be smaller than the difference between $\phi(A+vv',x+\delta)$ and $\phi(A,x)$ (see Fig. 2.3.4). The more $\phi(A+vv',.)$ deviates from $\phi(A,.)$, the easier it becomes to satisfy this condition. □

2.3.5 Unweighted sparsifiers and the Kadison-Singer conjecture

The Previous methods require a weighted graph to ensure spectral similarity, as the edges need to be reweighted during the process. Indeed, sparsifying unweighted graphs is a more difficult task related to the Kadinson-Singer (KS) conjecture. In this subsection, we only give a brief introduction to KS conjecture, which was proven by Spielman and his collaborators in 2015. The full proof of KS is clearly beyond the scope of this work, but it is interesting to introduce the conjecture, as the sparsifiers we proposed in our contributions are designed for unweighted graphs.

The KS conjecture dates back from 1959 and deals with infinite Hilbert spaces and Von Neumann algebras in quantum mechanics. In a fascinating article [CT06a], Casazza summarizes a dozen of mathematical problems across various fields of pure, applied mathematics, as well as computer science, all equivalent to the KS conjecture. Weaver proved [Wea04] that the KS conjecture is equivalent to the following result:

Theorem 13 (Marcus, Srivastava, Spielman [MSS13])**.** *There exists universal constants $\epsilon>0, \delta>0, r\in\mathbb{N}$, for which the following statement holds:*
If $v_1,...,v_m\in\mathbb{R}^n$ satisfy $||v_i||\leq\delta$ for all i and

$$\sum_{i\leq m} v_i v_i^T = I_n, \tag{2.119}$$

then there is a partition $I_1,...,I_r$ of $\{1,...,m\}$ such that

$$\left|\left|\sum_{i\in I_j} v_i v_i^T\right|\right| \leq 1-\epsilon,\ \forall j=1,...,r. \tag{2.120}$$

For an unweighted sparsifier, the weights s_i can only take values 0 or 1, and S and its complementary form a partition of $\{1,...,m\}$. So if we suppose $r=2$ (the partition is just composed of two subsets, with only one selected), the conjecture implies the existence of unweighted sparsifiers. Note the condition $||v_i||\leq\delta$, which can be replaced by $||v_i||=1$. For an unweighted sparsifier, we would have

$$\left|\left|\sum_{i\in S} v_i v_i^T\right|\right| \leq \kappa \leq 1-\epsilon. \tag{2.121}$$

CHAPTER 3

REDUCING THE DIMENSION OF THE DATA: SOME ASPECTS OF COMPRESSIVE SENSING

3.1 Motivation and objectives

The third and last aspect we address in the problem of dimensionality reduction concerns the dimension d of the data. The n elements of the dataset $\mathbb{X}$ are still characterized by a real vector of d coordinates, but d is not the real dimension of the data. They belong to an (unknown) hidden subspace of lower dimension and the objective is to find out and exploit their real dimension to save space and execution time.

In this part the fundamental assumption is therefore made on the dimension of the elements of $\mathbb{X}$: there exists a (unknown) basis where any $x \in \mathbb{X}$ is represented by a vector of $\mathbb{R}^k$ with $k \ll d$.

A typical example of this situation is the compressive sensing problematic. In the traditional digital signal processing approach, one starts by acquiring (receiving, measuring, calculating, sampling) data before compressing them. The acquisition step is often done from analog data which are transformed into digital data (via sampling and quantization). It is then necessary to compress data in order to be able to transmit them on a network or to store them in digital memories, for efficiency, time sparing and economy of size. Compression can only be done after the acquisition process. In this approach, a lot of data are sampled and then discarded at the compression step. Is it possible to sample only the data that will be useful ? This is the main objective of compressive sensing which exploits the natural sparsity of most real-world signals by sampling and compressing data at the same time.

The mathematical problem is very simple to model, but hard to solve: we have a vector of measurements

$$y = \phi x \tag{3.1}$$

formed by m linear combinations of a vector $x \in \mathbb{R}^d$ by a known measurement matrix $\phi \in \mathbb{R}^{m \times d}$, traditionnaly callled the sensing matrix. ϕ is rectangular and will often represent a dictionary: the union of several basis or an overcomplete basis. We aim to find x given y and ϕ, by exploiting the fact that x is k-sparse. Here m represents the number of measurements, in accordance to the compressive sensing literarure, which differs from the m in the previous chapter.

The linear model may appear crude and poor, but there are many phenomena and operations in signal processing that can be modelized by a matrix product: all classical transformations on finite signals (Fourier,

wavelets, etc.) are linear. The same goes for a lot of filtering, coding operations, etc.

So the problem boils down to solve an underdetermined linear system under constraint of sparsity. Our objective is to propose a statistical analysis of some of the existing sparse recovery methods. In order to do so, we briefly summarize the tools required to expose and solve the mathematical model and refer the reader to [EK12; FR13] for a comprehensive and rigourous review on compressive sensing.

This chapter is organized as follows: in the next section, we present the definition of an underdetermined linear system and the main properties of sparse signals. In section 3, we expose some properties of the sensing matrix: the spark, the null space property, the coherence and the restricted isometry property. In section 4, we expose the main families of sparse recovery methods and in section 5, we link these methods to the properties of the sensing matrix in order to obtain some guarantees of recovery. These guarantees depend on the type of recovery we consider: uniform or non-uniform.

The appendix contains reminders about (more or less) classical probabilistic distributions we used in our statistical model.

3.2 Sparse solutions of underdetermined linear systems

A vector $x \in \mathbb{R}^d$ is sparse if it has at most k nonzero coordinates (and obviously we assume that k is very small compared to d). The degree of sparsity of x is measured with the function

$$||x||_0 = \text{card}(\text{supp}(x)) = \lim_{p\to 0} ||x||_p = \text{card}(\text{supp}(x)) = \sum_{i=1}^{d} \mathbb{1}_{[x_i \neq 0]} \tag{3.2}$$

$||x||_0$ is equal to the number of nonzero coordinates of x. It is neither a norm, nor even a quasi-norm. The set of k-sparse signals is denoted by Σ_k and is not a vector space, as it is not stable under addition. But if $x, y \in \Sigma_k$ then $x + y \in \Sigma_{2k}$. In fact, Σ_k is the union of all linear subspaces of dimension $s \leq k$, of vectors with less than s nonzero coordinates.

We use

$$\begin{cases} \gamma = k/d < 1 & \text{the } \textit{sparsity rate,} \\ \mu = m/k > 1 & \text{the } \textit{overmeasure rate.} \end{cases} \tag{3.3}$$

When k, d, m vary, γ and μ will be considered as exact values of these parameters, or as limit when n, k, m tends to infinity (with a speed of convergence to be precised) thus defining what we call the *Large System Regime* (LSR) of linear growth of γ and μ with respect to d. In this case, even if we do not always explicitely write k_d and m_d, k and m are function of d.

Sparsity depends on the basis in which the vector is given and this basis is often unknown. Most real-world signals are not exactly sparse, but have coefficients close to zero. We have to consider compressible vectors, whose coordinates tend rapidly towards 0 and which can be approached by sparse vectors. This leads to the following definitions:

The best k-sparse approximation of a vector x relatively to the l_p-norm is the vector $\sigma_K(x)_p$ given by

$$\sigma_K(x)_p = \min_{y \in \Sigma_K} ||x - y||_p. \tag{3.4}$$

A simple way to get this approximation is to threshold to 0 all the $d-k$ smallest coefficients (in absolute value) of x.

A signal is said to be compressible if its coordinates x_i ranged in decreasing order satisfy

$$\exists\, C, q > 0 : \ |x_i| \leq \frac{C}{i^q} \ \forall\, i = 1, ..., d. \tag{3.5}$$

We denote by x_S a vector x whose coordinates indexed elsewhere than in S have been set to 0. We denote by ϕ_S a matrix whose columns indexed elsewhere than in S have been fixed at 0. We have in particular

$$\phi = \phi_S + \phi_{\bar{S}} \tag{3.6}$$

and

$$\phi_S x = \phi_S x_S = \phi x_S, \tag{3.7}$$

for all $x \in \mathbb{R}^d$. In the following, we use the notation $\mathbb{I} = [\![1, d]\!] = \{1, ..., d\}$.

Instead of working in a single basis, compressive sensing often uses union of basis to increase the richness of possible linear combinations and possible sparsity. Such a spanning set is called a dictionary.

The classical mathematical model of compressive sensing is a linear system

$$y = \phi x, \tag{3.8}$$

where $\phi \in \mathbb{R}^{m \times d}$ is the sensing matrix (also called sampling or measuring matrix), with $m \ll d$.

y is the observed vector from which we try to find x, which is only known through ϕ. The measurement is done by projecting x on the column vectors of ϕ, which are linear combinations of the coordinates of x. The previous equation is nothing more than a underdetermined linear system. ϕ is not invertible (and is even rectangular). In the presence of noise, one has

$$y = \phi x + \epsilon \tag{3.9}$$

where ϵ is a random noise.

Solving (3.9) is equivalent to the program $\mathscr{P}_0$:

$$\mathscr{P}_0 : \ \hat{x} = \underset{x \in \mathbb{R}^d : \phi x = y}{\operatorname{argmin}} \ ||x||_0 \tag{3.10}$$

There are three (big) issues:

- The above program is not convex because $||.||_0$ is not a norm. It cannot therefore be solved by conventional optimization techniques.
- It is a NP-hard problem: solving it amounts to testing all the vectors whose support is of cardinality k. There are $\binom{d}{k}$ of them, which grows exponentialy with d.
- The linear system is underdetermined and therefore admits an infinity of solutions.

There are roughly three different philosophies for tackling the problem in a practical way, which results in algorithms of three different types: convex relaxation algorithms, greedy or thresholding algorithms. In this work, we focus on greedy and thresholding algorithms, the performance of which we seek to evaluate based on the parameters d, m, k and the properties of the sensing matrices.

A (false) problem to consider is that of signals which are not sparse in the current basis, but in another basis to be determined. The study of the algorithm shows that the problem does not change and that it is not necessary to know the basis in which the signal is sparse, in order to to find the solution.

Compressive sensing is often presented as part of the sparse representation methods, which is true, but there are key differences between classic sparse methods and compressing sensing. In the former, the encoding is non-linear (it is used to determine a dictionary, depending on the signal, in which the latter is sparse) and the decoding is linear. In compressing sensing, the encoding is linear (ϕx is calculated) while the decoding is non-linear (one of the families of algorithms mentioned above is applied).

Both families seek to minimize the difference between the initial signal and its estimate with equivalent decomposition, but different criteria. In a way, sparsity representations try to determine a local sparse solution, while compressive sensing determines the sparsiest solution.

To circumvent the problem of non convexity, a classical method is to change $\mathscr{P}_0$ in $\mathscr{P}_1$

$$x^\star = \underset{x\in\mathbb{R}^d:\ \phi x=y}{\operatorname{argmin}} ||x||_1. \tag{3.11}$$

Solving $\mathscr{P}_0$ is equivalent to find the unique k-sparse vector $x^\star$ s.t. $\phi x^\star = \phi x$.

Theorem 14 (Theorem 2.13. in [FR13])**.** *Let $\phi \in \mathbb{R}^{m\times d}$. The following points are equivalent:*

- *Every k-sparse x is the unique solution of $\phi x = y$.*
- $\ker\phi \cap \Sigma_{2k} = \{0\}$.
- *Every subset of $2k$ columns from ϕ is linearly independent.*

Proof. If x and z are k-sparse and satisfy $y = \phi x = \phi z$, then $x - z \in \ker\phi$ and $x - z \in \Sigma_{2k}$. If $\ker\phi$ does not contain any $2k$-sparse vector, then $x = z$. Reciprocally, if the only solution is the k-sparse vector x and if $v \in \ker\phi \cap \Sigma_{2k}$, then $v = x - z$, with $x, z \in \Sigma_k$ and $\operatorname{supp}(x) \cap \operatorname{supp}(z) = \emptyset$. Thus $\phi x = \phi z$, so by assumption, $x = z$.
Eventually, every subset of $2k$ columns from ϕ is free if, and only if there does not exist any zero linear combination implying $2k$ coordinates. In words, no $2k$-sparse vector is in $\ker\phi$.

□

If it is possible to recover any k-sparse vector x from $y = \phi x$, then on one hand the theorem says that $\operatorname{rank}\phi = \dim\operatorname{Im}\phi \geq 2k$. On the other hand, $\operatorname{rank}\phi \leq m$ as ϕ does not possess m rows. So the number m of measures must satisfy

$$m \geq 2k. \tag{3.12}$$

3.3 Properties of sensing matrices

3.3.1 Spark

Let us note spark ϕ (contraction of "sparse" and "rank") the smallest number of columns from ϕ linearly dependent. The previous theorem implies that for any $y \in \mathbb{R}^m$, there exists at most one vector $x \in \Sigma_k$ such that $y = \phi x$ if, and only if spark $\phi > 2k$. Since $\phi \in \mathbb{R}^{m \times d}$, spark $\phi \in [2, m+1]$ and

$$\text{spark}\, \phi = \min\{k : \ker \phi \cap \Sigma_k \neq \{0\}\} \tag{3.13}$$

$$= \min_{x \neq 0} ||x||_0 \text{ subject to } \phi x = 0. \tag{3.14}$$

The condition spark $\phi > 2k$ is equivalent to the fact that if a solution of $\mathscr{P}_0$ is k-sparse, then it is unique, or that ϕ is one-to-one when applied to k-sparse vectors. The spark is clearly connected to the rank of the sensing matrix: there exists a set of spark ϕ columns that are dependent, so any set of spark $\phi - 1$ columns are free. The rank is the maximum number of linearly independent columns of ϕ: there exists at least a set of rank ϕ columns that are free, so any set of rank ϕ+1 columns are dependent. In words,

$$\text{spark}\, \phi - 1 \leq \text{rank}\, \phi \tag{3.15}$$

Finding x from y depends on the index of sparsity k of x, on the number of measurements m and on the properties of ϕ. Some deterministic matrices ϕ are known to satisfy the assumptions of the previous theorem (Vandermonde or Fourier transform matrices, etc.). Some random matrices also (Gaussian or sub-Gaussian matrices, for example).

The spark gives a guarantee of unicity for the problem $\mathscr{P}_0$, but is also a necessary and sufficient solution for recovery of any k-sparse vector under problem $\mathscr{P}_0$. But solving this problem, as already said, is NP-hard. We can now complete Th. 14:

Theorem 15. *Let $\phi \in \mathbb{R}^{m \times d}$ and $k \leq m$. The following are equivalent:*

- *Every $x \in \Sigma_k$ is the unique k-sparse solution of $\phi z = \phi x$.*
- $\ker \phi \cap \Sigma_{2k} = \{0\}$.
- $\forall S \subset [\![1, n]\!]$ *with* $|S| \leq 2k$, ϕ_S *is injective.*
- *Every subset of $2k$ columns of ϕ are linearly independent.*
- spark $\phi > 2k$.

The first sentence has to be understood as follows: for all $x \in \Sigma_k$, there is a unique solution to the equation $\phi z = \phi x$ where the unknown is z. In words, x is the only k-sparse solution of $\phi x = \phi z$; note that this is exactly the problem $\mathscr{P}_0$. As a consequence, exact recovery of every k-sparse vector needs

$$m \geq 2k \tag{3.16}$$

and we have already said that in fact, $m = 2k$ is sufficient for perfect recovery, but with unstable methods not usable anymore in high dimension.

There is a strong link between spark and the theory of error correcting codes (which is one possible application for compressive sensing). If ϕ is the generator matrix of a linear error correcting code, then the spark of ϕ is exactly the minimum distance of the code.

So the spark gives a unicity recovery guarantee for $\mathscr{P}_0$. The next guarantee, called null space property (NSP), gives the same guarantee for $\mathscr{P}_1$.

3.3.2 Null space property

Let ϕ be an $m \times n$ matrix. Then ϕ has the null space property (NSP) of order k if, for all $v \neq 0 \in \ker\phi$ and for all index sets S s.t. $|S| \leq k$,

$$||v_S||_1 < ||v_{\overline{S}}||_1 \tag{3.17}$$

Where $\overline{S}$ is the complementary of S; equivalently,

$$||v_S||_1 < \frac{1}{2}||v||_1 \tag{3.18}$$

The algebraic interpretation is that the vectors of the kernel must not be too concentrate on small subsets. The "weight" of the kernel vectors must be diluted within all their coordinates.

Before giving the geometric interpretation of NSP, this is the right place to recall the geometric interpretation of the problem $\mathscr{P}_p$. Let's recall that:

$$\mathscr{P}_p : \ \hat{x} = \underset{z:\phi z=y}{\operatorname{argmin}} ||z||_p \tag{3.19}$$

Solving $\mathscr{P}_p$ is equivalent to find the vector(s) of minimum p-norm, solution of the system $\phi z = y$ (z is the unknown and y is fixed). The solution of $\phi z = y$ formed a linear subspace of $\mathbb{R}^n$. Solving $\mathscr{P}_p$ is finding all vectors in the intersection of the smallest ball $||z||_p$ and the subspace $\phi z = y$. The existence and the unicity of the solution(s) depends on p, because the geometry of the unit balls depends on p:

$\forall v \in \ker\phi$, $x+v$ is solution because $\phi x+\phi v = \phi x$. In fact, the subspace of solutions can be written $x+\ker\phi = \{x+v;\ v \in \ker\phi\}$.

Let's come back to the NSP and let's take an example in dimension $n = 2$. Suppose that $x = (1,0)^T$ is the 1-sparse vector to recover. The support of x is $S = \{1\}$ (x is colinear to the x axis). Let $\phi = (1,a) \in \mathbb{R}^{1\times 2}$ the sensing matrix (i.e. we make one measure to recover the two dimensional vector x). It is easy to see that $||x||_0 = ||x||_1 = 1$ and that $\ker\phi \propto (-a,1)^T$ is the linear space of all vectors colinear to $v = (-a,1)^T$. $\ker\phi$ is of dimension 1 and for all $v \in \ker\phi$, $||v_S||_1 = |a|$ and $||v_{\overline{S}}||_1 = 1$, so that ϕ verifies the NSP of order 1 if, and only if, $|a| < 1$.

To solve $\mathscr{P}_1$, we need to find x in the unit l_1-ball $||x||_1 = 1$, given that $y = \phi x$. If more than one vector z in the ball satisfies the equation $y = \phi z$, then the solution is not unique, and some sparse vectors may not be recovered by $\mathscr{P}_1$. The solutions to $\mathscr{P}_1$ are exactly the vectors in the affine space $x + \ker\phi$ and in the following lines we illustrate a condition on a for the solution to be unique.

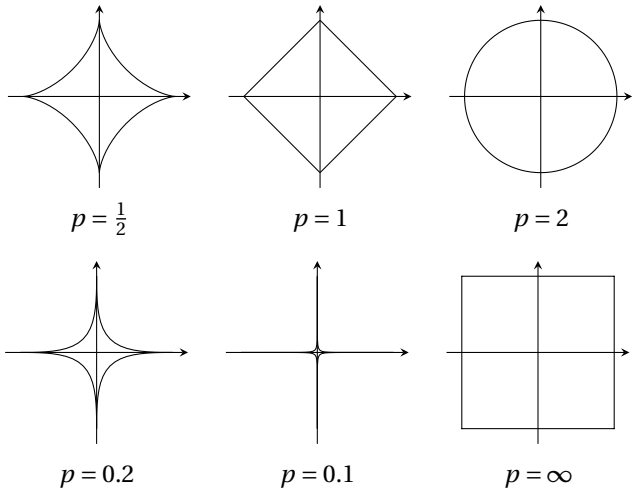

Figure 3.1 – Form of the unit ball of several norms.

As we can see in Fig. 3.2, the number of intersections between the line $x + \ker\phi$ and the ball $||x||_1 = 1$ depends on the slope $-1/a$ of the line. When the NSP is satisfied, i.e. when $|a| < 1$, there is a unique solution. However, when $|a| \geq 1$, there exists more than one solution. Specifically, if $|a| = 1$, the line is parallel to a facet of the ball, resulting in multiple solutions. When $|a| > 1$, the line intersects the ball at two points, demonstrating that the solution is not unique.

Theorem 16. *Let ϕ be an $m \times n$ matrix, and let $k \leq m$. Then, the following are equivalent:*

- *If a solution x of $\mathscr{P}_1$ satisfies $||x||_0 \leq k$, it is the unique solution.*
- *ϕ satisfies NSP of order k.*

NSP is a necessary and sufficient condition that guarantees to find the unique solution of $\mathscr{P}_1$ by using $||.||_1$ minimization algorithms. Let ϕ be a $m \times d$ matrix with $m \leq d$

- $2 \leq \text{spark}\,\phi \leq m + 1$.
- in general $\text{spark}\,\phi \neq \text{rank}\,\phi + 1$.
- if ϕ is random with i.i.d. entries and continuous density, then $\text{spark}\,\phi = \text{rank}\,\phi + 1$ with probability one.
- calculating $\text{spark}\,\phi$ is complex.
- $\text{rank}\,\phi$ can be computed via Gaussian elimination.

Recover x from y is equivalent to define a function (or an algorithm) $\Delta : \mathbb{R}^m \longrightarrow \mathbb{R}^d$ such that $\Delta \circ \phi = \text{Id}_{\mathbb{R}^d}$. This is impossible except if x is sufficiently sparse. Under specific conditions on m and k,

$$x \in \Sigma_k \Rightarrow \Delta \circ \phi(x) = x. \tag{3.20}$$

Let's precise the link between NSP and sparse vectors recovery:

$$\left(\Delta \circ \phi(x) = x,\ \forall x / \text{ supp}(x) \subset \Lambda \right) \iff \left(||u_\Lambda||_1 < ||u_{\overline{\Lambda}}||_1\ \forall u \neq 0 \in \ker\phi \right). \tag{3.21}$$

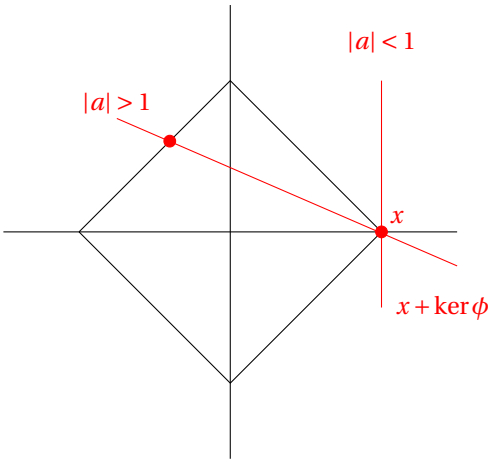

Figure 3.2 – Geometric illustration of the existence and unicity of solutions for the $\mathscr{P}_p$ problem when $p = 1$. The set of solutions to $\phi x = y$ for a given measurement y is the affine space $x + \ker\phi$, whose intersection with the (square) unit ball $||x||_1 = 1$ depends on a. The number of solutions corresponds to the number of intersection points. When the NSP is satisfied, the solution is unique. When the NSP is not satisfied, there exists multiple solutions.

Theorem 17 (1.2. in [EK12]). *Let* (ϕ, Δ) *a couple satisfying* $\forall x \in \mathbb{R}^d$,

$$||\Delta \circ \phi(x) - x||_2 \leq \frac{c}{\sqrt{k}} \sigma_{k,1}(x), \tag{3.22}$$

then ϕ *satisfies NSP with order* $2k$.

Theorem 18 (4.4. in [FR13]). *The next conditions are equivalent:*

- *If a solution* x *of* $\mathscr{P}_1$ *is* k*-sparse,* x *is the unique solution of* $\mathscr{P}_1$.
- ϕ *satisfies NSP of order* k.

x is also the solution of $\mathscr{P}_0$. The proof of this theorem was initially proposed by Cohen, Dahmen et DeVore [CDD09].

3.3.3 Coherence

To get a sufficiant condition of recovery for x and determine in which cases the solutions of $\mathscr{P}_0$ and $\mathscr{P}_1$ coincide, it is necessary to introduce another property of sensing matrices called the coherence. Let's suppose that the columns ϕ_j of ϕ are normalized for $||.||_2$. The coherence measures the correlation between the columns, by evaluating the scalar products between any pair of them. A scalar product close to zero signifies that the columns are "nearly" orthogonal. Coherence measures the lack of orthogonality in the columns of ϕ, or the way ϕ is close to an isometry. If the columns of ϕ form an orthonormal basis, $\mu(\phi) = 0$. When $\mu(\phi)$ is small, we talk of incoherence.

Definition 3 (Coherence). *The coherence* $\mu(\phi)$ *of a matrix* $\phi = (\phi_j)_j \in \mathbb{R}^{m \times d}$ *with normalized columns is*

$$\mu(\phi) = \max_{i \neq j} | < \phi_i, \phi_j > |. \tag{3.23}$$

One can prove the following inequality, called the Welch bound:

$$\sqrt{\frac{d-m}{m(d-1)}} \leq \mu(\phi) \leq 1. \tag{3.24}$$

When $m \ll d$, the Welch bound gives approximatively $\mu(\phi) \geq 1/\sqrt{m}$.

Theorem 19 (3.3. in [Kut12]). *Let x a solution of $\mathscr{P}_0$ such that*

$$||x||_0 < \frac{1}{2}\left(1 + \frac{1}{\mu(\phi)}\right), \tag{3.25}$$

Then x is the unique solution of $\mathscr{P}_0$ and $\mathscr{P}_1$.

3.3.4 Restricted isometry property

When measures are corrupted with noise, the NSP property and coherence are no longer enough to ensure the perfect recovery of x and it is necessary to introduce a more powerful property called RIP for Restricted Isomerty property.

Definition 4 (RIP). *ϕ satisfies RIP of order k if there exists $\delta_k \in]0,1[$ such that for all $x \in \Sigma_k$*

$$(1-\delta_k)||x||_2^2 \leq ||\phi x||_2^2 \leq (1+\delta_k)||x||_2^2. \tag{3.26}$$

This property insures that ϕ preserves approximatively the distances between all pairs of k-sparse vectors and allows a form of robustness against noise. It ensures a sufficient condition of success for some algorithms, in the case of noisy measurements. RIP is linked to the Johnson-Lindenstrauss lemma and the Gelfand width.

If a matrix ϕ satisfies the RIP of order k, it satisfies also RIP of order $k' < k$ with constant $\delta_{k'} \leq \delta_k$.

Theorem 20 (1.4. in [EK12]). *Let $\phi \in \mathbb{R}^{m \times d}$ a sensing matrix satisfying the RIP of order $2k$ for $\delta_{2k} \in]0,1/2]$. Then,*

$$m \geq ck\ln\left(\frac{d}{k}\right), \tag{3.27}$$

with $c = 1/2\ln(\sqrt{24}+1) \simeq 0.28$.

This theorem lower bounds the number of measures m for a sensing matrix to satisfy the RIP. The link between RIP and NSP is givent by the following result:

Theorem 21 (1.5. in [EK12]). *Let $\phi \in \mathbb{R}^{m\times d}$ a sensing matrix satisfying RIP of order $2k$ for $\delta_{2k} < \sqrt{2}-1$. Then ϕ satisfy NSP of order $2k$ with respect to the constant*

$$c = \frac{2}{1-(1+\sqrt{2})\delta_{2k}}. \tag{3.28}$$

We omit the proof of these theorems and refer the reader to [EK12].

The RIP property allows to evaluate the distance between any vector of $\mathbb{R}^d$ and the solution $x^\star$ of $\mathscr{P}_1$:

Theorem 22 (3.4. of [Kut12]). *Let $\phi \in \mathbb{R}^{m\times d}$ a sensing matrix satisfying RIP of order $2k$ for $\delta_{2k} < \sqrt{2}-1$. Let $x \in \mathbb{R}^d$ and $x^\star$ a solution of $\mathscr{P}_1$. Then,*

$$||x-x^\star||_2 \leq \frac{c}{\sqrt{k}}\sigma_{k,1}(x). \tag{3.29}$$

Let's note x_k the vector derived from x while keeping only the k highest coordinates; the previous inequality gives

$$||x-x^\star||_2 \leq \frac{c}{\sqrt{k}}||x-x_k||_1 \tag{3.30}$$

and

$$||x-x^\star||_2 \leq c||x-x_k||_1. \tag{3.31}$$

In particular, if x is k-sparse, the recovery is exact. Indeed, from the RIP point of view,

- If $\delta_{2k} < 1$, $\mathscr{P}_0$ has a unique k-sparse solution.
- If $\delta_{2k} < \sqrt{2}-1$, the solution of $\mathscr{P}_1$ is the same as the solution of $\mathscr{P}_0$.

We are now able to precise the links between all these properties:

Theorem 23 (4.1. in [Kut12]). *Let $\phi \in \mathbb{R}^{m\times d}$ a sensing matrix with normalized columns.*

- $$spark\,\phi \geq 1 + \frac{1}{\mu(\phi)}. \tag{3.32}$$
- *ϕ satisfies RIP of order k for $\delta = \lambda\mu(\phi)$, $\forall \lambda < \mu(\phi)^{-1}$.*
- *If ϕ satisfies RIP of order $2k$ with $\delta_{2k} < \sqrt{2}-1$ and*

$$\frac{\delta_{2k}\sqrt{2}}{1-(1+\sqrt{2})\delta_{2k}} < \sqrt{\frac{k}{n}}, \tag{3.33}$$

then ϕ satisfies NSP of order $2k$.

All these properties are about to be used in the algorithms to ensures necessary and or sufficient guarantees of recovery of a sparse vector with a givent sensing matrix ϕ.

3.4 Sparse recovery methods

3.4.1 Problematic

The linear system $y = \phi x$ posses at least a solution (the real x). But as $\phi \in \mathbb{R}^{m\times d}$, it is underdetermined and thus is likely to have several solutions. We have seen that under conditions on ϕ (RIP, NSP, etc.), the sparsity of x may ensures the uniqueness of the solution. Reconstructing the initial vector amounts to solving the optimization problem

$$\mathscr{P}_0 : \begin{cases} x^\star = \text{argmin}||x||_0 \\ \text{s.c. } \phi x = y \end{cases} \tag{3.34}$$

The problem is non-convex, NP-hard, while

$$\mathscr{P}_1 : \begin{cases} x^\star = \text{argmin}||x||_1 \\ \text{s.c. } \phi x = y \end{cases} \tag{3.35}$$

is convex and may be solveed by classical linear programming algorithms. The complexity of $\mathscr{P}_0$ comes from the exponential number of possible configurations of the support of x.

We have already discussed the links between the two types of problems. Suppose the existence of a unique solution x to the problem $\mathscr{P}_1$. Then the columns ϕ_i where $i \in \Lambda = \text{supp}(x)$ are linearly independent. In particular, $||x||_0 \leq k$. Indeed, if there exists a vector v with support Λ such that $\phi v = 0$, for all $t \neq 0$,

$$||x||_1 < ||x + tv||_1 = \sum_{i\in\Lambda} |x_i + tv_i| = \sum_{i\in\Lambda} \text{sgn}(x_i + tv_i) \times (x_i + tv_i). \tag{3.36}$$

For $|t|$ as small, $\text{sgn}(x_i + tv_i) = \text{sgn}(x_i)$ for all $i \in \Lambda$, and therefore

$$\begin{aligned} ||x||_1 &< \sum_{i\in\Lambda} \text{sgn}(x_i) \times (x_i + tv_i) && (3.37)\\ &= \sum_{i\in\Lambda} \text{sgn}(x_i) \times x_i + t \sum_{i\in\Lambda} \text{sgn}(x_i) \times v_i && (3.38)\\ &= ||x||_1 + t \sum_{i\in\Lambda} \text{sgn}(x_i) \times v_i, && (3.39) \end{aligned}$$

which is impossible because we can always choose t such that the last expression is negative. Therefore, the family $(\phi_i)_{i\in\Lambda}$ is free. The solution of $\mathscr{P}_1$ is therefore necessarily k-sparse: to efficiently solve $\mathscr{P}_0$, we actually need to solve $\mathscr{P}_1$.

If we know Λ, the support of x, an estimator of x using this information is defined as an "oracle". The optimal algorithm solving $y = \phi x$ is in that case to use the Moore-Penrose pseudo-inverse of ϕ_Λ :

$$\phi_\Lambda^+ = (\phi_\Lambda^T \phi_\Lambda)^{-1} \phi_\Lambda^T. \tag{3.40}$$

The unique k-sparse solution is given by

$$\begin{cases} x^\star_\Lambda = \phi^+ y \\ x^\star_{\overline{\Lambda}} = 0 \end{cases} \tag{3.41}$$

The algorithm is optimal in the sense that the solution $x^\star$ minimizes $||\phi x - y||_2$. It is thus also a least square solution.

3.4.2 Convex optimization algorithms

BP algorithm (for "basis pursuit") to solve $\mathscr{P}_1$ becomes "basis pursuit under quadratic constraint" (BPQC) in the presence of noise:

$$BPQC : \begin{cases} x^\star = \text{argmin}||x||_1 \\ \text{s.c. } ||\phi x - y||_2^2 \leq \epsilon \end{cases} \tag{3.42}$$

The problem is linked to BPDN ("basis pursuit denoising")

$$BPDN : x^\star = \underset{x}{\text{argmin}} \left(\lambda ||x||_1 + ||\phi x - y||_2^2\right) \tag{3.43}$$

and to the LASSO estimator:

$$LASSO : \begin{cases} x^\star = \text{argmin}||\phi x - y||_2 \\ \text{s.c. } ||x||_1 \leq \tau \end{cases} \tag{3.44}$$

We admit the following result which specifies the link between all these optimization problems.

Theorem 24 (Prop. 3.2. in [FR13])**.**

- *If x solution of BPDN for $\lambda > 0$, then $\exists\, \epsilon \geq 0$ s.t. x is solution of BPQC.*
- *If x unique solution of BPQC for $\epsilon \geq 0$, then $\exists\, \tau > 0$ s.t. x unique solution of LASSO.*
- *If x solution of LASSO for $\tau > 0$, then $\exists\, \lambda \geq 0$ s.t. x solution of BPDN.*

The Dantzig selector can also be used to solve the $\mathscr{P}_1$ problem.

$$\text{Dantzig} : \begin{cases} x^\star = \text{argmin}||x||_1 \\ \text{s.t. } ||\phi^T(\phi x - y)||_\infty \leq \tau \end{cases} \tag{3.45}$$

Each of these algorithms has been studied in the literature and conditions on the matrix ϕ, in terms of consistency, kernel property or RIP property, makes it possible to ensure the reconstruction, or not, of x.

We won't go further on convex algorithms as our interest is in the performance of greedy algorithms.

3.4.3 Greedy and thresholding algorithms

The two most common greedy algorithms are OMP ("Orthogonal Matching Pursuit") and IHT ("Iterative Hard Tresholding"). There are many variations of these two algorithms, which we will not deal with.

Description of OMP

The sparse signal x observed through the vector y is reconstructed with the OMP algorithm. The inputs to OMP are y, the observation matrix ϕ, and the sparsity level k of the desired solution. Then, at each iteration t, the algorithm selects the index λ_t of the most correlated atom with the residue vector r_{t-1} at the previous iteration ($r_0 = y$ is the observed vector):

$$\lambda_t = \underset{i \in [\![1,d]\!]/\Lambda_{t-1}}{\operatorname{argmax}} \left|\langle r_{t-1}, \phi_i \rangle\right|, = \underset{i \in [\![1,d]\!]/\Lambda_{t-1}}{\operatorname{argmax}} |W_i(t)|, \tag{3.46}$$

where $W_i(t) = \langle r_{t-1}, \phi_i \rangle$. Then, the algorithm adds λ_t to the current support,

$$\Lambda_t = \Lambda_{t-1} \cup \{\lambda_t\}, \tag{3.47}$$

it computes an estimate x_t of the unknown x:

$$x_t = \underset{z \in \mathbb{R}^d \,:\, \operatorname{supp}(z) \subset \Lambda_t}{\operatorname{argmin}} ||y - \phi z||_2 \Leftrightarrow \begin{cases} x_{t\,(\Lambda_t)} &= \left(\phi_{\Lambda_t}^T \phi_{\Lambda_t}\right)^{-1} \phi_{\Lambda_t}^T y \\ x_{t\,(\overline{\Lambda}_t)} &= 0 \end{cases}$$

where $x_{t\,(\Lambda_t)}$ and $x_{t\,(\overline{\Lambda}_t)}$ stand for the vector obtained by extracting the entries of x_t indexed by Λ_t and $\overline{\Lambda}_t = [\![1,d]\!]/\Lambda_t$, respectively. Then, the algorithm computes the residue

$$\begin{aligned} r_t = y - \phi x_t &= y - \phi_{\Lambda_t} x_{t\,(\Lambda_t)} \\ &= y - \phi_{\Lambda_t} \left(\phi_{\Lambda_t}^T \phi_{\Lambda_t}\right)^{-1} \phi_{\Lambda_t}^T y = y - P_t y = Q_t y. \end{aligned} \tag{3.48}$$

and stops when $|\Lambda_t| = k$. The outputs of OMP are the signal estimate x_k and its support Λ_k. Note that $P_t = \phi_{\Lambda_t} \left(\phi_{\Lambda_t}^T \phi_{\Lambda_t}\right)^{-1} \phi_{\Lambda_t}^T$ and $Q_t = I - P_t$ are the projection matrices onto the subspace spanned by the vectors $\{\phi_i\}_{i \in \Lambda_t}$ and onto the orthogonal of this subspace, respectively (see Fig. 3.3). Note also that both matrices only depend on these vectors. This is an obvious point but worth making as it will allow some independency between the iterations.

Theorem 25 (Proposition 3.5. in [FR13]). *Let $\phi \in \mathbb{R}^{m \times d}$, let $x \neq 0$ s.t. $supp(x) \subset \Lambda \subset [\![1..d]\!]$, let v s.t.*

$$v = \underset{z \in \mathbb{R}^d : supp(z) \subset \Lambda}{argmin} ||y - \phi z||_2. \tag{3.49}$$

Then,

$$\left(\phi^T (y - \phi v)\right)_\Lambda = 0, \tag{3.50}$$

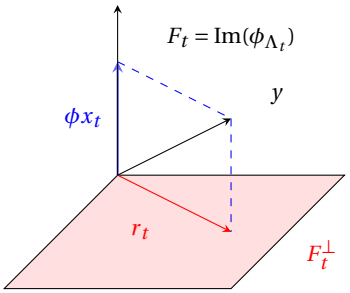

Figure 3.3 – At each iteration of the OMP algorithm, a new atom is selected in Λ_t. x_t is chosen to be the vector from F_t whose image under ϕ is closest to y, and the residual r_t is the projection of y onto the complementary space of F_t.

and if $|\Lambda| = k$, *OMP rebuilds* x *in* k *iterations if, and only if,* ϕ_Λ *is one-to-one and for all* $y \in \{\phi z,\ supp(z) \subset \Lambda\}$,

$$\max_{i\in\Lambda} \left|\phi^T y\right|_i > \max_{i\in\overline{\Lambda}} \left|\phi^T y\right|_i \tag{3.51}$$

Inequality (39) may be written in a simplier way by using Moore-Penrose pseudo-inverse and therefor gives the ERC ("Exact Recovery Condition"):

$$ERC : \max_{i\notin\Lambda} ||\phi_\Lambda^+ \phi_i||_1 < 1 \tag{3.52}$$

Proof. By construction of v, ϕv is the orthogonal projection of y on the space $E = \{\phi z,\ \mathrm{supp}(z) \subset \Lambda\}$, thus $< y - \phi v, \phi z >= 0$ for all z whose support is Λ.
Therefore, $< \phi^T(y - \phi v), z >= 0$.

The proof is by recurrence on the number of indices in the support. If it contains one index, the proposition is obvious. If we assume that OMP finds all vectors x having a support of cardinal k, then necessarily ϕ_Λ is one-to-one, since $\phi x = \phi x' \Rightarrow x = x'$. Furthermore, the index chosen in the first iteration is always in the support Λ, so that

$$\max_{i\in\Lambda} \left|\phi^T y\right|_i > \left|\phi^T y\right|_j \ \forall j \in \overline{\Lambda}. \tag{3.53}$$

Taking the maximum on $j \in \overline{\Lambda}$ gives the second condition.

Reciprocally, let us assume that the support is not yet completely reconstructed at the t-th iteration. We show that $\Lambda_t \subset \Lambda$ for all $t \leq k$. This implies that $r_t = y - \phi x_t^\star$ is in the space E, so that $i_{t+1} \in \Lambda$. Consequently, $\Lambda_{t+1} \subset \Lambda$. As $(\phi^T r_t)_{\Lambda_t} = 0$, $i_{t+1} \notin \Lambda_t$, otherwise we would have $r_t = 0$ and the algorithm would have already reconstructed x. This shows that Λ_k has cardinality k: it is Λ. □

One of the drawbacks of the algorithm (corrected in several variants) is to keep a false index until the end of the algorithm when it is selected at a given iteration. Thus, as soon as an error appears in the estimated support, it ensures that the vector will never be reconstructed.

Description of IHT

The basic thresholding algorithm selects in one shot for the support Λ, the indices of the highest k coordinates of $\phi^T y$ then calculates

$$x^\star = \underset{x\in\mathbb{R}^d:\text{supp(x)}\subset\Lambda}{\text{argmin}} \; ||y-\phi x||_2 \tag{3.54}$$

It finds the right vector if, and only if,

$$\min_{i\in\Lambda} \left|\phi^T y\right|_i > \max_{i\in\overline{\Lambda}} \left|\phi^T y\right|_i . \tag{3.55}$$

The IHT iterative thresholding algorithm initializes and terminates in the same way as OMP and keeps, at each iteration, the highest t coordinates of the vector:

$$\left|x_t^\star + \phi^T(y-\phi x_t^\star)\right| \tag{3.56}$$

It does not require orthogonal projection, the most expensive part of the OMP algorithm. Iterative algorithms are known to be less efficient than convex optimization algorithms [Tro09]. But the performance strongly depends on the choice of the matrix ϕ. If ϕ is a Gaussian matrix and if

$$m = O\left(k\ln\frac{d}{k}\right), \tag{3.57}$$

then ϕ has the RIP property of order k with high probability. We saw that this property was a sufficient condition for the success of l^1 convex optimization algorithms (including in the noisy case). For example, for Gaussian ϕ and $x\in\Sigma_k$,

$$m = O\left(k\ln\frac{d}{k}\right) \tag{3.58}$$

ensures, with high probability, that BP or LASSO finds x.

We also saw that if $\mu(\phi) < 1/(2k-1)$, then OMP reconstructs any k-sparse signal in k iterations.

For a randomly chosen Gaussian matrix ϕ, if $m = O(k\ln(d/k))$, OMP finds any k-sparse signal x with probability $\geq 1-\exp(-c\times m)$. But we can show that for some sensing matrices, when $m\sim k\ln d$, there exist vectors x which are not reconstructed by OMP or IHT with high probability. For more precise considerations, we will refer to [Tro09].

3.5 Recovery guarantees

A guarantee of perfect recovery is a condition on the sensing matrix ϕ and the parameters d,k,m to ensure that the estimated sparse signal $\widehat{x}$, solution of a reconstruction algorithm, is indeed equal to the initial signal x:

$$x = \widehat{x} \tag{3.59}$$

The condition can be deterministic or probably approximatively correct (PAC or one can also say "with overwhelming probability") in which case a parameter ρ measures how far the probability of perfect recovery is from 1.

The guarantee can be uniform if one single matrix ϕ allows the reconstruction of all sparse signal x (ϕ doesn't depend on x) or non-uniform if, for any fixed signal x, there exists a matrix ϕ (depending on x) that allows the reconstruction.

A guarantee can be sufficient, necessary, or both.

Note that a recovery guarantee depends on the nature of the problem, but also on the algorithm used to solve it. Let us summarize:

- Deterministic framework:
  - Uniform: $\exists\, \phi$ s.t. $\forall x, \widehat{x} = x$
    $\Rightarrow m = 2k$ necessary, but in fact, $m = 2k$ sufficient (in theory, without noise or robustness considerations).
  - Non-uniform: $\forall x, \exists\, \phi$ s.t. $\widehat{x} = x$
    $\Rightarrow m = k+1$ necessary, and in fact, $m = k+1$ often sufficient (in theory, without noise).

  The problems are solved with algebraic methods of recovery using interpolation Vandermonde or FFT matrices [FR13, p. 281]. These methods are not stable, not robust to noise or approximate sparsity.
- Random framework
  - Uniform: $\mathbb{P}_\phi[\forall x, \widehat{x} = x] \geq 1 - \rho$
  - Non-uniform:$\forall x, \mathbb{P}_\phi[\widehat{x} = x] \geq 1 - \rho$

  For example, the RIP property obtained with random Gaussian matrix is uniform, necessary, sufficient and probabilist (there exists deterministic matrix with RIP, but difficult to build and less efficient than random ones). If ϕ satisfies the (ϵ, kp)-RIP condition, then with high probability, we can reconstruct every k- sparse vector by using BP (if $p = 2$), IHT (if $p = 3$) or OMP (if $p = 12$).

In the random framework and non-uniform setting, x is deterministic and there is no framework in which x and ϕ could be random, giving rise to an average analysis in which the probability is evaluating with a random couple (x, ϕ). The interest of an average analysis is that the observed performances are better than those predicted by the NSP, Coherence or even RIP property. This average analysis is what we propose in Chap. 6.

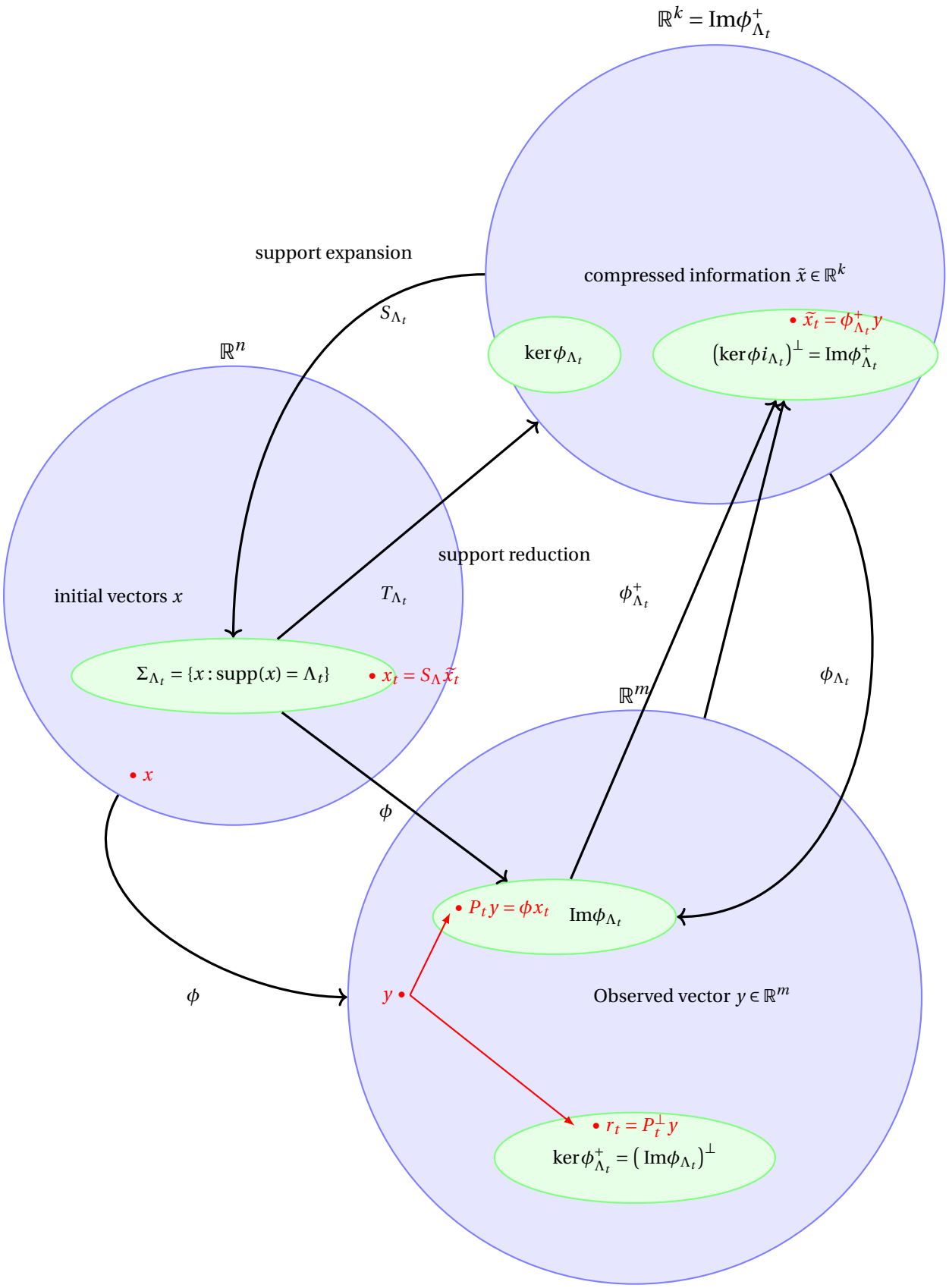

Figure 3.4 – Spaces and subspaces implied in OMP.

PART II

Contributions

CHAPTER 4

VOLUME MAXIMIZATION FOR DATA COMPRESSION

4.1 Introduction and objectives

The goal of this thesis is to reduce the dimensionality of the data. The first aspect we address is the reduction of the database size. This chapter presents our contribution toward this objective.

In Chap. 1, we presented the problem of database compression preserving the volume of the underlying datamatrix $B \in \mathbb{R}^{d\times n}$. We explained how this CSSP "over-the-rank" boils down to a discrete optimization problem given by Pb. 1 whose main equation is

$$\underset{B_k:\ B_k \subset B}{\operatorname{argmax}} \operatorname{vol}(B_k) \tag{4.1}$$

with $\operatorname{rank}(B) = d \leq k \leq n$.

We recall the three key assumptions about the data matrix: n is much greater than d, the rank of B is equal to d and each column of B is normalized. This last assumption is natural because it allows items of the database to be compared, and crucial because without normalization the optimization problems we study would not have any finite solution.

In the following sections, several optimization problems arise from Pb. 1 and provide evidence of two key principles related to the spectrum of the SPD covariance matrix $\Sigma = BB^T$. These principles offer insights on how to maximize the volume, either greedily, either in one-shot approaches.

A fruitful strategy to solve the different discrete problems is to examine a continuous relaxation, solve it and study to what extent the solutions are valid or approximate solutions to the initial discrete problem; we employ this strategy multiple times, with the aim of finding deterministic algorithms with reasonable complexity. All proposed methods are initialized with a square $d \times d$ matrix B_d upon which columns are concatenated - either iteratively or all at once - in order to build the extracted rectangular matrix of the prescribed size. As we shall see, each added column increases the eigenvalues of $\Sigma_d = B_d B_d^T$, the objective being to choose the columns that maximize the volume the most. When columns b_i's are concatenated to B_d, the size of $\Sigma_i = B_i B_i^T$ does not change, and the matrix remains positive definite, so the volume is given by the determinant of Σ_i.

The chapter is organized as follows: in section 2 we present our greedy strategies for solving the initial optimization problem and their connections to low-rank perturbation theory. From this analysis, two key principles and two methods for approximating the solutions emerge, which we describe in detail. Section 3 presents global optimization strategies. A continuous relaxation of the initial problem leads to discrete water-filling

techniques we elaborate on. Section 4 describes the algorithms, their implementations and their complexities. Section 5 is dedicated to simulations and comparisons with other methods, including uniform sampling, DPP and RectMaxVol algorithm, while section 6 discusses two potential applications in matrix conditioning and compressive sensing.

Seven optimization problems are studied in this section. To facilitate their reading and clarify their connections, a paragraph and a summary table are provided at the beginning of section 4.4; it is possible to refer to it at any time during the reading of the chapter.

4.2 Greedy strategies

4.2.1 Greedy strategies modify the spectrum of the covariance matrix by adding low-rank perturbations

Finding the maximum volume submatrix of a given size is a NP-hard problem [ÇM09]. A simple (though sub-optimal) and natural approach to maximize the volume among all choices of n columns is to greedily add one column b to B_i at each iteration, such that the concatenated matrix (B_i, b) has the maximum volume:

Problem 2 (Discrete greedy maximization of the volume)**.** *Given an initial square submatrix $B_d \in \mathbb{R}^{d\times d}$ and $\Sigma_d = B_d B_d^T$, select at each iteration i a column from $B \setminus B_i$ that belongs to*

$$\hat{b} \in \underset{b\in B\setminus B_i}{\operatorname{argmax}} \operatorname{vol}(B_i, b) \tag{4.2}$$

and update $\Sigma_{i+1} = (B_i, \hat{b})(B_i, \hat{b})^T$.

Algebraic interpretation

To understand how the volume evolves in rectangular matrices when the number of columns exceeds the rank, we must interpret the effect of concatenating a column b to a given matrix $B_i \in \mathbb{R}^{d\times i}$, for $i = d+1, ..., k$, from the perspective of the covariance matrix. This involves analyzing both the volume and the spectrum. The decomposition of Σ_{i+1} in rank-one operators is

$$\Sigma_{i+1} = \sum_{j=1}^{i} b_j b_j^T + bb^T = B_i B_i^T + bb^T = \Sigma_i + bb^T, \tag{4.3}$$

where bb^T is a rank-one operator.

The relationship between rank-one perturbations of a symmetric matrix and its spectrum has been extensively studied since the 1960's [GV96; Gol73; Wil88; IN09; BNS78; JS18]. The effects of a rank-one perturbation on the spectrum of a matrix can be summarized by the following family of equalities, which generalize Cauchy's interlacing property [GV96; HJ85]:

$$\lambda_j(\Sigma_i + bb^T) = \lambda_j(\Sigma_i) + \mu_j, \quad j = 1, ..., d \tag{4.4a}$$

$$\sum_{j=1}^{d} \mu_j = ||b||^2,\ 0 \leq \mu_j \leq ||b||^2, \quad j = 1, ..., d \tag{4.4b}$$

The equality $\lambda_j(\Sigma + bb^T) = \lambda_j(\Sigma) + ||b||^2$ in (4.4b) occurs if, and only if b is an eigenvector of Σ corresponding to λ_j. In this case, all other eigenpairs remain unchanged and λ_j is increased by $||b||^2 = 1$ [IN09; BNS78]. If b is orthogonal to an eigenvector of Σ, the corresponding eigenvalue is not modified and remains part of the spectrum of $\Sigma + bb^T$. In all other cases, the contribution of $||b||^2$ to the spectrum of $\Sigma + bb^T$ is distributed among all eigenspaces of BB^T, with a total contribution of $||b||^2 = 1$:

$$\text{tr}(\Sigma + bb^T) = \text{tr}(\Sigma) + \text{tr}(bb^T) = \text{tr}(\Sigma) + \text{tr}(b^T b) = \text{tr}(\Sigma) + ||b||^2 = \text{tr}(\Sigma) + 1. \tag{4.5}$$

More details about the properties of the μ_j's are provided in Appendix A.1. These tools will be central to the one-shot algorithms presented in the section 4.3.

Proposition 1 (Reformulation of Pb. 2)**.**

$$\hat{b} \in \underset{b \in B \setminus B_i}{\text{argmax}}\ \text{vol}(B_i, b) = \underset{b \in B \setminus B_i}{\text{argmax}}\ b^T \Sigma_i^{-1} b \tag{4.6}$$

Proof. Equation (4.3) connects the volume of the augmented and the initial matrices:

$$\begin{aligned} \text{vol}(B_i, b)^2 &= \det(\Sigma_i + bb^T) && (4.7)\\ &= \det \Sigma_i \times (1 + b^T \Sigma_i^{-1} b) && (4.8)\\ &= \text{vol}(B_i)^2 \times (1 + b^T \Sigma_i^{-1} b), && (4.9) \end{aligned}$$

where (4.8) is due to the matrix determinant lemma [JS18]. □

The previous equation shows that maximizing the volume is equivalent to maximizing $b^T \Sigma^{-1} b$ over all choices of b.

Both algorithms require an initial matrix B_d to start the iterations and this matrix B_d must be such that $\Sigma_d = B_d B_d^T$ is invertible. Suppose that, at a given iteration, the current matrix B is the same for both the left-hand side (LHS) and right-hand side (RHS) of (4.6). Then according to (4.9), the columns that maximize the volume are the same for both the LHS and the RHS and (4.9) and also demonstrates that this strategy is optimal within the greedy framework of selecting one column at each iteration. Thus, the two problems are equivalent.

It is important to note that this equivalence does not imply that there is a unique column to select at each iteration. Indeed, different columns might maximize (4.6) and lead, at the next iteration, to different extracted matrices and resulting volumes.

Statistical interpretation of $b^T\Sigma^{-1}b$

Up to the end of this subsection, let us omit the index i of Σ_i and consider a generic Σ at a given iteration. The expression $b^T\Sigma^{-1}b$ admits multiple interpretations.

Σ^{-1} is known as the precision matrix and models the Fisher information (with respect to the parameter of a statistical model and its estimator). $b^T\Sigma^{-1}b$ can be interpreted as the information gain in the direction of b, which should be maximized.

$b^T\Sigma^{-1}b$ represents the norm of b with respect to the Mahalanobis distance associated with Σ; maximizing this expression is equivalent to selecting the item of $\mathbb{X}$ corresponding to the column b that maximizes the contribution of this item to the variance of the sample; it identifies the most extreme point of $\mathbb{X}$.

Eventually, $b^T\Sigma^{-1}b$ can be expressed as the dot product of a fixed vector with b. The next result clarifies this representation.

Proposition 2. *There exists an orthonormal matrix Q and a diagonal matrix D such that*

$$b^T\Sigma^{-1}b = ||D^{-1/2}Q^Tb||_2^2 = \sum_{j=1}^{d} \frac{(Q^Tb)_j^2}{\lambda_j}. \tag{4.10}$$

Proof. Q and D are, respectively, a change of basis matrix and the corresponding diagonal form for the SPD matrix Σ:

$$\Sigma = BB^T = QDQ^T, \tag{4.11}$$

so that

$$\Sigma^{-1} = QD^{-1}Q^T, \tag{4.12}$$

with $D^{-1} = \text{diag}(\lambda_1^{-1}, ..., \lambda_d^{-1})$. Noting that

$$b^T\Sigma^{-1}b = b^TQD^{-1}Q^Tb = (Q^Tb)^T(Q^Tb) = ||D^{-1/2}Q^Tb||_2^2 = \sum_{j=1}^{d} \frac{(Q^Tb)_j^2}{\lambda_j} \tag{4.13}$$

proves that $b^T\Sigma^{-1}b$ is the dot product of the fixed vector $(\lambda_1^{-1}, ..., \lambda_d^{-1})$ with the vector whose coordinates $(Q^Tb)_j^2$ are the square of the coordinates of b, expressed in the eigenbasis given by the columns of Q. □

4.2.2 Continuous relaxation of Pb. 2, solution of Pb. 2 and geometric interpretation of $b^T\Sigma^{-1}b$: increasing the smallest eigenvalues and reducing the spread of the spectrum maximize the volume

The dot product representation (4.10) and a continuous relaxation of Pb. 2, referred to as Pb. 3 provide an important geometric interpretation of $b^T\Sigma^{-1}b$, offering insights into how to solve Pb. 2.

In Pb. 3, we replace the term $(Q^Tb)_j^2$ (the square of the j-th coordinate of the column b_j expressed in the eigenbasis) with a variable that varies continuously between 0 and 1, and is not necessarily derived from a column b of B. Since Q is orthogonal, it preserves the norm, ensuring that $||b||^2 = ||Q^Tb||_2 = 1$.

In this problem, we replace the term $(Q^T b)_i^2$ (the square of the i-th coordinate of the column b_i in the eigenbasis) with the variable x_i, which varies continuously between 0 and 1, and is not necessarily derived from a column b of B. By letting

$$c = (c_1, .., c_d) = \left(\lambda_1^{-1}, ..., \lambda_d^{-1}\right), \tag{4.14}$$

Pb. 3 becomes a classical linear problem of maximizing a dot product:

$$b^T \Sigma^{-1} b = \sum_{i=1}^{d} \frac{(Q^T b)_i^2}{\lambda_i} = c^T x. \tag{4.15}$$

It can be formalized as follows and solved using standard linear programming techniques:

Problem 3 (Continuous relaxation of Pb. 2)**.** *Find*

$$\underset{x \in \mathbb{R}_+^d :\, ||x||=1}{\operatorname{argmax}} \; c^T x \tag{4.16}$$

Proof. (Solution of Pb. 3). The problem is a standard linear maximization over the simplex S_d of $\mathbb{R}^d$. If all the x_i are equal, every point of S_d maximizes the expression. From now on, we assume that the x_i's are not all equal and that $x_1 \leq x_2 \leq ... \leq x_d$. Using Caratheodory and Motzkin theorems, a classical result in linear programming [BT97] shows that the solutions are given by the convex hull of the extreme points of S_d where the maximum is attained. For instance, if x_d is strictly greater than $x_1, ..., x_{d-1}$, the maximum value of the dot product is x_d and is attained at the single point corresponding to $x_j = 0$ for $j = 1, ..., d-1$ et $x_d = 1$. □

In words, the expression is maximized when the coefficient of the smallest eigenvalue is 1 and all the others are 0. We can now discuss the difference between the solutions of Pb. 2 and Pb. 3.

Proof. (Resolution of Pb. 2 and links with Pb. 3). Finding the maximum of a finite number of scalar products with respect to a fixed vector is a well-known task in machine learning called Maximum Inner Product Search (MIPS) [KSR17; Abu+19], and is related to the nearest neighbor search problem. The naïve approach evaluates all nd elements and scales as $O(nd)$ operations (excluding the cost of evaluating the spectrum and eigenbasis). Deterministic methods exist that reduce the complexity to $O(\sqrt{d})$, and randomized Bandits algorithms can achieve $O(1)$ operations for finding an approximate solution [KSR17]. However, the costly part of the algorithm is in the basis change and require to work with the eigenpairs, which has cubic time complexity.

If the eigenbasis is not known, the solution of Pb. 3 provides a possible approach: selecting the column b that maximizes $(u_d^T b)^2$, where u_d is the normalized eigenvector corresponding to λ_d, approximates the goal of maximizing $b^T \Sigma^{-1} b$. □

The previous proofs demonstrate that the greedy maximization scheme involves increasing the smallest positive eigenvalues of Σ. Since the total possible increase is finite and less than 1, this approach penalizes the largest eigenvalues and reduces the spread of the sprectum:

Principle 1. *Increasing the smallest eigenvalues of the covariance matrix maximizes the volume.*

Principle 2. *Narrowing the spread of the spectrum (of the covariance matrix) maximizes the volume.*

4.3 Global optimization strategies

We now study another strategy to maximize the volume, which will be global, unlike the previous case, which was a greedy approach.

4.3.1 Resolving the continuous relaxation of Pb. 1 using principle 2...

We consider a continuous relaxation of Pb. 1, where B_k is now an arbitrary matrix with normalized columns, and not necessarily a submatrix of B.

Problem 4 (Continuous relaxation of Pb. 1)**.** *Find:*

$$\max_{B_k:||b||=1} \text{vol}(B_k)^2 = \max_{B_k:\Sigma_k=B_kB_k^T;\text{Tr}\Sigma_k=k} \det\Sigma_k \tag{4.17}$$

Proof. (Solutions to the two subproblems of Pb. 4) We first solve the RHS of (4.17), before discussing the relationships between the two subproblems and deriving a solution for the LHS. Given that $\det\Sigma = \prod_{j=1}^d \lambda_j$ and $\text{Tr}(\Sigma) = \sum_{j=1}^d \lambda_j$, the maximum of the product of d positive numbers under a sum constraint is the case of equality in the inequality of arithmetic and geometric means [Ber99]. The equality is reached if, and only if all the λ_j are equal and their common value is

$$\lambda = (\sum_{j=1}^d \lambda_j)/d = k/d. \tag{4.18}$$

The maximum volume is thus $(k/d)^d$ and is reached for the scalar matrix λI_d. This result is consistent with principle 2: the spectrum must be as narrow as possible.

The next step is prove the equality between the RHS and LHS of (4.17). Since all column vectors of B_k are normalized, this imposes a constraint on the trace of Σ_k:

$$\text{Tr}(\Sigma_k) = \text{Tr}(B_kB_k^T) = k, \tag{4.19}$$

In words, all matrices B_k whose columns are normalized have a covariance matrix Σ_k with a trace of k. The converse is not true: $\text{Tr}(BB^T) = k \not\Rightarrow ||b_i|| = 1$ for all i. To prove the equality, one must prove that the optimal matrix λI_d can be obtained from a normalized matrix B such that $BB^T = \lambda I_d$. This result can be established using the Schur-Horn theorem [MOA11, Th.B.2. p. 302]. For the remainder of the proof, we consider the initial matrix $B \in \mathbb{R}^{d\times n}$, though the results also hold for any extracted matrix $B_k \in \mathbb{R}^{d\times k}$, where $k = d,...,n$. Suppose that $BB^T = \lambda I_d$. Then the singular values decomposition (SVD) of B is given by

$$B = UDV^T, \tag{4.20}$$

where $U \in \mathbb{R}^{d\times d}$, $V \in \mathbb{R}^{n\times n}$ are orthogonal matrices and $D = \sqrt{\lambda}(I_d, 0) \in \mathbb{R}^{d\times n}$. Let $V = (V_1, V_2)$ where $V_1 \in \mathbb{R}^{n\times d}$ contains the first d columns of V. Since $BB^T = \lambda UDD^TU^T = \lambda I_d$, then

$$B = UDV^T = \sqrt{\lambda}U(I_d, 0)\begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix} = \sqrt{\lambda}UV_1^T, \tag{4.21}$$

and we verify that $BB^T = \lambda UV_1^T V_1 U^T = \lambda I_d$. In words, the solutions of the RHS of (4.17) are precisely the matrices $\sqrt{\lambda}UV_1^T$ where U is orthogonal and the columns of V_1 form an orthonormal set of d vectors. Next, let's prove that we can choose B with normalized columns. The Gram matrix $G = B^T B = \lambda V_1 V_1^T \in \mathbb{R}^{n\times n}$ has two distinct eigenvalues: λ with multiplicity d and 0 with multiplicity $n-d$. The diagonal elements are equal to $||b_i||^2 = 1$ for all i. Now, the Schur-Horn theorem guarantees the existence of a real symmetric matrix G with given spectrum $\lambda_1 \geq ... \geq \lambda_n$ and diagonal elements $c_1 \geq ... \geq c_n$ if, and only if

$$\sum_{i=1}^{s} c_i \leq \sum_{i=1}^{s} \lambda_i, \forall s < n \text{ and } \sum_{i=1}^{n} c_i = \sum_{i=1}^{n} \lambda_i. \tag{4.22}$$

These hypotheses are satisfied for $\lambda_1 = ... = \lambda_d = n/d$, $\lambda_{d+1} = ... = \lambda_n = 0$ and $c_i = 1$ for all i. according to Horn theorem, there exists a real symmetric matrix $G \in \mathbb{R}^{n\times n}$ with spectrum $(\lambda_i)_i$ and diagonal elements $(c_i)_i$. Since G is symmetric, it can be expressed as $G = B^T B$ with $B \in \mathbb{R}^{n\times d}$. B must necessarily have $\sqrt{\lambda} = \sqrt{n/d}$ as its only singular value with multiplicity d and its columns satisfy

$$||b_i||^2 = c_i = 1, \ \forall i = 1, ..., d. \tag{4.23}$$

Furthermore, $BB^T \in \mathbb{R}^{d\times d}$ also has λ as its only eigenvalue with multiplicity d, so that $BB^T = \lambda I_d$, and B is normalized. Thus, the LHS of (4.17) has at least one solution and its maximum is equal to the square root of the maximum of the RHS. We have therefore proven the existence of a normalized matrix B such that $BB^T = \lambda I_d$. Fig. 4.1 illustrates the relationships between the underlying matrix sets. □

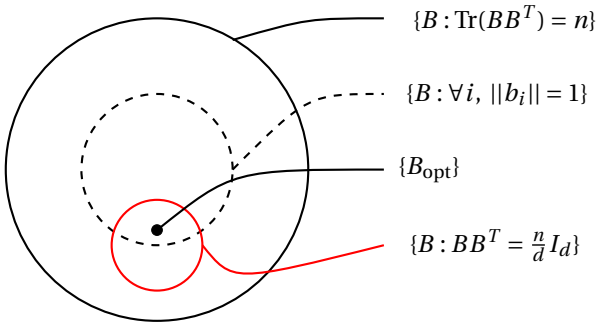

Figure 4.1 – The solutions B_{opt} of the LHS of Pb. 2 form the intersection of two sets: the set (in red) where BB^T is a scalar matrix of the form $(n/d)I_d$ and the set (in dotted lines) where the columns of B are normalized.

Interpretation: the continuous relaxation of Pb. 1, yields matrices with one single eigenvalue, resulting in the narrowest possible spectrum. Interestingly, this is consistent with Principle 2, established in the context of greedy approaches, which we now follow to propose new algorithms and find approximate solutions to the original discrete Pb. 1. Specifically, we seek a submatrix B of A such that the covariance matrix Σ is as "close"

as possible to a scalar matrix. The notion of closeness should be defined in terms of matrix distances, relative to a function of the volume (and thus of the spectrum) that increases as the distance to the set of scalar matrices decreases. Implementing this constraint directly is challenging, so we rely on the following equivalent statement

$$BB^T = \lambda I_d \iff \mathrm{Sp}(BB^T) = \{\lambda\}, \tag{4.24}$$

which leads to the minimization Pb. 5 proposed in the next paragraph and aims to equalize the eigenvalues.

4.3.2 ... leading to a discrete water-filling technique for solving Pb. 1

The traditional water-filling algorithm is designed to optimize power allocation in multiple communication channels [CT06b]. The optimal solution is achieved when the total power is evenly distributed across each channel. Similarly, we aim to equalize the eigenvalues of the extracted matrix, using a discrete water-filling technique that constrains the spectrum of Σ to be as narrow as possible. The problem of equalization can be formalized as follows:

Problem 5 (Discrete minimization of the spread of the spectrum.)**.** *Find:*

$$\underset{B_k \subset B : \Sigma_k = B_k B_k^T}{\operatorname{argmin}} \sum_{j=1}^{d} \left(\lambda_j - \overline{\lambda}\right)^2 \tag{4.25}$$

where $\overline{\lambda}$ is the common value to reach.

As before, we do not solve Pb. 5 directly, but a continuous relaxation, where B_k is no longer a submatrix of B.

Problem 6 (Continuous relaxation of Pb. 5.)**.** *Find:*

$$\underset{B_k \in \mathbb{R}^{d\times k} : \Sigma_k = B_k B_k^T}{\operatorname{argmin}} \sum_{j=1}^{d} \left(\lambda_j - \overline{\lambda}\right)^2. \tag{4.26}$$

Proof. (Resolution of Pb. 6) The initialization step requires to select d columns from B to form two square matrices B_d and $\Sigma_d = B_d B_d^T$ each of size $d \times d$, using any existing volume maximization algorithm or the fast initialization algorithm.

Next, $k-d$ columns are concatenated to B_d in one step to form B_k and $\Sigma_k = B_k B_k^T$. This concatenation changes the initial eigenvalues $\lambda_1(\Sigma_d), ..., \lambda_d(\Sigma_d)$. The goal is to select the $k-d$ columns in such a way as to best equalize the final eigenvalues $\lambda_1(\Sigma_k), ..., \lambda_d(\Sigma_k)$. The effect of concatenating one or more columns to B_d to create the submatrix B_k, can be characterized from the perspective of the covariance matrices Σ_d and Σ_k using the decomposition in terms of rank-one operators:

$$\Sigma_k = B_k B_k^T = \sum_{i=1}^{k} b_i b_i^T = \Sigma_d + \sum_{i=d+1}^{k} b_i b_i^T. \tag{4.27}$$

Concatenating $k-d$ columns to B_d is thus equivalent to adding $k-d$ rank-one operators to Σ_d.

From (4.4a), we know that each column b_i increases every eigenvalue $\lambda_j(\Sigma_d)$ by an amount μ_{ij} where $0 \leq \mu_{ij} \leq 1$ and $\sum_{j=1}^{d} \mu_{ij} = 1$. In Appendix A.1, we detail the relationship between μ_{ij} and the scalar product $(u_j^T b_i)^2$, where u_j is any normalized eigenvector associated with $\lambda_j(\Sigma_d)$: $(u_j^T b_i)^2$ is positively correlated with μ_{ij} and when u_j approaches b_i, μ_{ij} approaches 1 (subject to the constraint of remaining below the gap δ_j between consecutive eigenvalues).

The eigenvalues of Σ_k are obtained from those of Σ_d by adding the $k-d$ contributions μ_{ij} from all the rank-one perturbations $b_i b_i^T$:

$$\begin{cases} \lambda_j(\Sigma_k) = \lambda_j(\Sigma_d) + \epsilon_j, \ j = 1, ..., d \\ \\ \sum\limits_{i=d+1}^{k} \mu_{ij} = \epsilon_j, \ \sum\limits_{j=1}^{d} \epsilon_j = k-d, \ 0 \leq \epsilon_j \leq k-d. \end{cases} \tag{4.28}$$

and the ϵ_j are positively correlated with the scalar products $\left(u_j^T b_i\right)^2$ for all i.

Since the maximum volume is achieved with a scalar matrix λI_d, a strategy to maximize the volume is to select columns that narrow the spread of the sprectum or, equivalently, to constrain the ϵ_j so that $\lambda_j + \epsilon_j$ for all $i = j, .., d$, are as close as possible to a common value $\overline{\lambda}$. Once B_d is fixed, Pb. 6 can thus be expressed as:

$$\begin{cases} \underset{\epsilon_j}{\text{argmin}} \frac{1}{2} \sum_{j=1}^{d} \left(\lambda_j + \epsilon_j - \overline{\lambda}\right)^2 \\ \text{s.t. } \epsilon_j \geq 0 \, ; \sum_{j=1}^{d} \epsilon_j = k-d \, ; \, \sum_{j=1}^{d} \lambda_j = d. \end{cases} \tag{4.29}$$

where the constraints on the ϵ_j and λ_j arise from

$$\sum_{j=1}^{d} \epsilon_j = \text{Tr}(B_k B_k^T) - \text{Tr}(B_d B_d^T) = k-d. \tag{4.30}$$

The Lagrangian $\mathscr{L}(\epsilon, \alpha, \beta)$ is

$$\frac{1}{2} \sum_{j=1}^{d} \left(\lambda_j + \epsilon_j - \overline{\lambda}\right)^2 + \alpha \left(\sum_{j=1}^{d} \epsilon_j - k + d \right) - \sum_{j=1}^{d} (\beta_j \epsilon_j) \tag{4.31}$$

and

$$\frac{\partial \mathscr{L}}{\partial \epsilon_j} = \lambda_j + \epsilon_j - \overline{\lambda} + \alpha - \beta_j \ , \ j = 1, ..., d. \tag{4.32}$$

The problem is convex: the function to optimize is of the form $||M\epsilon - b||_2^2/2$ with $M = I_d$ and $b = (\overline{\lambda} - \lambda_j)_j$. The equality constraint is linear and the inequality constraints are convex. The Karush-Kuhn-Tucker (KKT) conditions are:

$$\bullet\ \epsilon_j \geq 0,\ \beta_j \geq 0,\ \beta_j \epsilon_j = 0,\ \forall j, \tag{4.33}$$

$$\bullet\ \sum \epsilon_j = k - d, \tag{4.34}$$

$$\bullet\ \lambda_j + \epsilon_j - \overline{\lambda} + \alpha - \beta_j = 0,\ \forall j. \tag{4.35}$$

From these equations, if $\beta_j > 0$, then $\epsilon_j = 0$ and (4.35)

$$\overline{\lambda} - \alpha < \lambda_j. \tag{4.36}$$

If $\epsilon_j > 0$, then $\beta_j = 0$ and (4.35) gives

$$\epsilon_j = (\overline{\lambda} - \alpha) - \lambda_j. \tag{4.37}$$

If $\epsilon_j = \beta_j = 0$, then $\overline{\lambda} - \alpha = \lambda_j$.

This solution has a simple water-filling interpretation. The goal is to equalize all the eigenvalues, where each eigenvalue initially has a value λ_j, and we add a certain amount to each, with the constraint that the total amount added must equal $k - d$. This can be intrepreted as pouring water into tubes, where the objective is to level the water across all tubes, but the total amount of water available is limited. We formulated the problem as wanting all λ_j to equal $\bar{\lambda}$, and the optimal solution is: pour water until each tube reaches the level $\bar{\lambda} - \alpha$, if this level is higher than the original value λ_j; for all other tubes, leave them unchanged. There is no closed form solution for $\bar{\lambda} - \alpha$.

The amount of water poured is a function of the water level ν (see Fig. 4.2 for illustration) given by:

$$W(\nu) = \sum_{j=1}^{d} (\nu - \lambda_j(\Sigma_d))_+. \tag{4.38}$$

This function is non-decreasing, continuous and piecewise linear, with changes in slope occuring at the points λ_j. The optimum water level is reached for $\nu = \overline{\lambda} - \alpha$ such that $W(\overline{\lambda} - \alpha) = k - d$. The properties of W ensure the existence and uniqueness of $\overline{\lambda} - \alpha = W^{-1}(k - d)$, though an analytic expression for it cannot be provided and it must be evaluated numerically.

The problem (4.29) is the same when replacing $\overline{\lambda}$ by 0. This simply corresponds to translating the reference level. By setting the new Lagrangian as

$$\mathscr{L} = \frac{1}{2} \sum_{j=1}^{d} \left(\lambda_j + \epsilon_j\right)^2 - \alpha \left(\sum_{j=1}^{d} \epsilon_j - k + d \right) - \sum_{j=1}^{d} (\beta_j \epsilon_j) \tag{4.39}$$

We then obtain in that case an optimal level equal to α instead of $\overline{\lambda} - \alpha$. □

It is interesting to compare $\overline{\lambda} - \alpha$ to the average value k/d of the eigenvalues of Σ_k. By summing the d equations of (4.35), we get:

$$\Sigma_{j=1}^{d} \beta_j = k - \overline{\lambda} d, \tag{4.40}$$

then, after dividing by d, the mean of the β_i is

$$\overline{\beta} = \frac{k}{d} - (\overline{\lambda} - \alpha) \tag{4.41}$$

so that k/d is always greater or equal to $\overline{\lambda} - \alpha$, with equality if all the λ_j are less than k/d. In first approximation, $\overline{\lambda} - \alpha$ can be approximated by the threshold k/d.

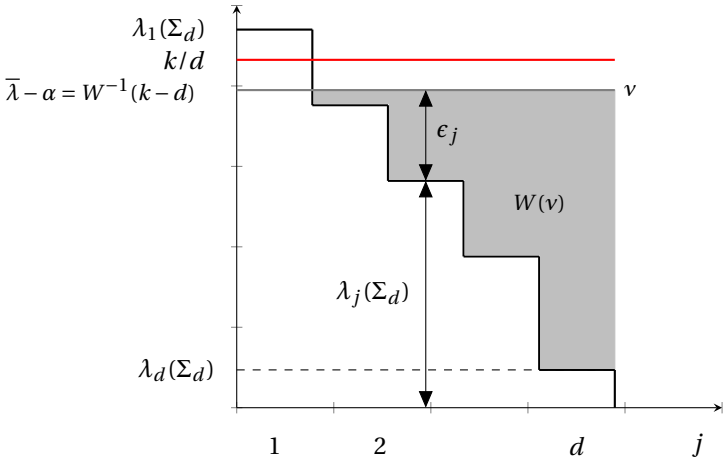

Figure 4.2 – Water-filling technique: each eigenvalue is represented by a column with initial value of $\lambda_j(\Sigma_d)$. Then a quantity of water equal to $k-d$ is poured (in grey) to equalize the eigenvalues. The threshold reached is $\overline{\lambda} - \alpha$, except for eigenvalues larger than this threshold, which therefore remain unchanged. $W(\nu)$ denotes the quantity of poured water required to achieve a water level of ν (in grey). The optimal water level is $\nu = \overline{\lambda} - \alpha = W^{-1}(k-d)$.

Let us return to the discrete Pb. 5. A discrete version of the previous continuous strategy is implemented in the next section, to develop an algorithm capable of equalizing the spectrum of the extracted matrix.

4.4 Algorithms and implementation

In this section, we apply the previous greedy and global strategies and propose several algorithms to solve the optimization problems related to the volume. Up until now, we have presented six optimization problems that are summarized in Fig. 4.3.

4.4.1 Fast initialization algorithm

All proposed algorithms need an initialization step, during which a square $d \times d$ submatrix B_d of maximum volume is created. This submatrix can be constructed using any method for square matrix volume maximization, such as an exhaustive search, the MaxVol algorithm [Gor+], or DDP volume sampling [Lau20].

An exhaustive search has a exponential complexity, while all the other methods generally have a cubic complexity in terms of number of operations. Therefore, it is desirable to find a method with lower complexity.

We propose a simple method with quadratic complexity that is easy to implement. Under the assumption of nomalized columns, the identity matrix I_d is a matrix of maximum volume for size $d \times d$. This matrix has

the maximum possible volume of 1 and its columns form an orthonormal basis (all orthogonal matrices of size $d \times d$ achieve the maximum volume). The idea behind the algorithm is to extract from B a square matrix close to the identity matrix I_d: let $\epsilon > 0$. By browsing the columns of B, select d columns such that the maximum squared coordinate of each column is greater than $1-\epsilon$. Stop when you have selected one column for each coordinate.

The number of columns to examine to find d columns satisfying the ϵ-criteria is at worst n, and each search for the maximum within a column has complexity $O(d)$, making the overall complexity at worst $O(nd)$.

This algorithm can be effective if n is large and ϵ is sufficiently small. In such cases, the volume of the resulting matrix can be close to the actual maximum (see Table 4.1). This method can be particularly useful when n is large and speed is the most critical factor.

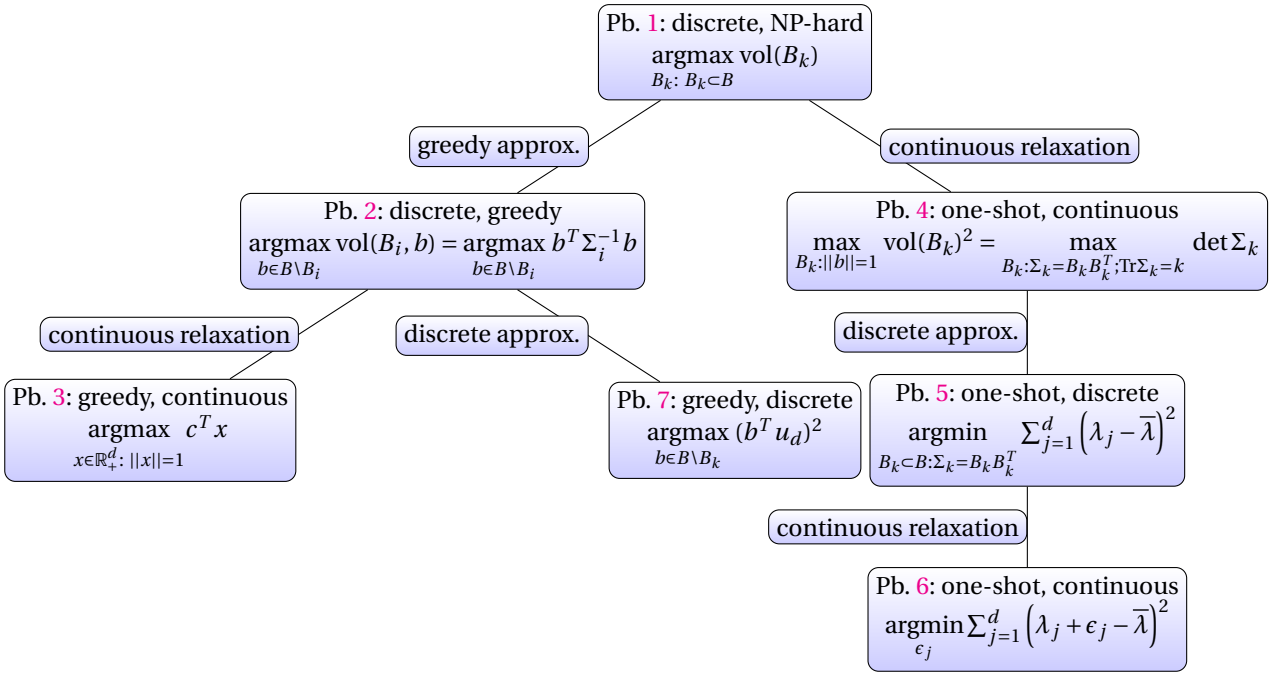

Figure 4.3 – All the optimization problems and their mutual relationships. A child problem is either a continuous relaxation of its parent or an approximation of its objective function. It should be emphasized that there is no guarantee that the solution of a child problem will be a solution to its parent problem.

4.4.2 First greedy algorithm: MaxVolDiv

The MaxVolDiv algorithm (Maximum Volume for Diversity) solves Pb. 2 by selecting the column that maximizes the diversity of information $b^T \Sigma^{-1} b$. Initially, a classical algorithm is used to initialize a maximum volume submatrix of size $d \times d$ upon which the number of columns is greedily increased by selecting, at each iteration, the column that is the solution to (4.2) from among all remaining columns of B.

Table 4.1 – Volume achieved by the MaxVolDiv algorithm when initialized by fast initialization and MAXVOL algorithm as a function of ϵ, n, d and $k = d$ (the real maximum is 1). When $k = 100$, no significant difference in performance is observed between the two initialization methods.

ϵ	n	d	$k = d$		$k = 100$	
			MAXVOL	FASTINIT	MAXVOL	FASTINIT
10^{-2}	10^5	5	0.9924	0.9126	3.20×10^6	3.20×10^6
5.10^{-2}	10^6	10	0.7315	0.4262	9.95×10^9	9.94×10^9
5.10^{-2}	10^7	10	0.8737	0.376	9.95×10^9	9.95×10^9
10^{-3}	10^7	5	0.9987	0.9936	3.20×10^6	3.20×10^6

Algorithm 1 MaxVolDiv

Require: $B \in \mathbb{R}^{d \times n}, k \in [\![d, n]\!]$
Ensure: $B_k \in \mathbb{R}^{d \times k}$
1: Initialize $B = (b_1, ..., b_d) \in \mathbb{R}^{d \times d}$
2: Evaluate $\Sigma^{-1} = (BB^T)^{-1}$
3: **for** $i = d+1, ..., k$ **do**
4: **for** $b_j \neq b_1, ..., b_{i-1}$ **do**
5: Evaluate $b_j^T \Sigma^{-1} b_j$
6: **end for**
7: $b = \text{argmax}(b_j^T \Sigma^{-1} b_j)$
8: Update Σ^{-1}: $\Sigma \leftarrow \Sigma + bb^T$ ▷ With Woodbury formula
9: **end for**

This algorithm is a simpler implementation of the RECTMAXVOL algorithm from [MO17] in terms of complexity ($O(nd) + O(nkd)$ for MaxVolDiv and $O(nk^2) + O(nk^2)$ for RECTMAXVOL) and interpretation.

Complexity: the initialization step of our algorithm has the complexity of the chosen method: $O(nd^2)$ for the MAXVOL algorithm, $O(k^3)$ for DPPs or $O(nd)$ for FastInit.

Before the first iteration, one needs to evalute BB^T and its inverse (line 2), which costs $O(nd^2)$ using naive matrix multiplication and $O(d^3)$ for inverting a $d \times d$ matrix. During the subsequent $k - d$ iterations (lines 3-9), the algorithm evaluates $b_j^T \Sigma^{-1} b_j$ for each candidate columns b_j. The inner loop (lines 4-6) that evaluates the $b_j^T \Sigma^{-1} b_j$ terms should be performed efficiently in matrix form: if $B_{-i} \in \mathbb{R}^{d \times (n-i+1)}$ is formed from all the columns b in B that have not yet been selected at iteration i, then the diagonal terms $b_j^T \Sigma^{-1} b_j$ of $B_{-i}^T \Sigma^{-1} B_{-i}$ can be computed in $O(d^2)$ operations. Σ^{-1} is then updated (line 8), which involves inverting a matrix obtained by adding a rank-one perturbation. The Sherman-Morrison-Woodbury (SMW) formula,

$$\begin{cases} \left(\Sigma + bb^T\right)^{-1} = \Sigma^{-1} - \alpha(\Sigma^{-1} b)(\Sigma^{-1} b)^T \\ \alpha = \left[1 + b^T \Sigma^{-1} b\right]^{-1} \end{cases} \tag{4.42}$$

handles this step with a complexity of $O(d^2)$.

Consequently, the main loop requires $O((n-d)(k-d)d)$ operations, leading to an overall complexity of $O(nkd)$ for the algorithm.

4.4.3 Second greedy algorithm: MaxVolCorr

As observed in Pb. 2, maximizing $b^T\Sigma^{-1}b$ can be equivalent to selecting the column that has the maximum coordinate corresponding, in the eigenbasis, to the smallest eigenvalue (principle 1). This occurs when the gap δ_d between the two smallest eigenvalues is sufficiently large. A simpler cost function that approximately optimizes the same criteria is:

Problem 7 (Greedy maximization of the volume)**.** *At each iteration, select the column*

$$\underset{b\in B\setminus B_k}{\operatorname{argmax}} (b^T u_d)^2 \tag{4.43}$$

where u_d is a normalized eigenvector related to the smallest eigenvalue λ_d.

This new algorithm selects the column most correlated with the eigenspace of the smallest eigenvalue.

Algorithm 2 MaxVolCorr

Require: $B \in \mathbb{R}^{d\times n}, k \in [\![d,n]\!]$
Ensure: $B_k \in \mathbb{R}^{d\times k}$
1: Initialize $B = (b_1,...,b_d) \in \mathbb{R}^{d\times d}$
2: Initialize u_d ▷ Eigenvector related to λ_d
3: Initialize $s(d) = (b_j^T u_d)_j$ and evaluate $((b_j^T u_d)^2)_j$
4: **for** $i = d+1,...,k$ **do**
5: Evaluate b =argmax$(b_j^T u_d)^2$
6: Update u_d ▷ With Mitz and al. method
7: Update $s(d)$ ▷ With Bénasséni formula
8: **end for**

Proof. of the quadratic complexity.

Line 1: the initialization step to produce B_d is the same as for MaxVolDiv. If the fast initialization algorithm is used, its complexity is $O(nd)$.

Line 2: $\Sigma = B_d B_d^T$ need not to be explicitly computed; the Lanczos algorithm, which uses only matrix-vector products, can compute $\Sigma y = B(B^T y)$ with two consecutive matrix-vector products. The Lanczos algorithm (or its variants) first computes the smallest eigenvalue λ_d and its normalized eigenvector u_d in $O(dn)$ operations [Dem97].

Line 3: the initial scalar product vector $s(d)$ with coordinates $s_j(d) = b_j^T u_d$ (where j runs through the $n-d$ remaining columns of B) is evaluated in $O(d(n-d))$ operations; the square of its coordinates are evaluated in $O(n-d)$ operations.

Inside the main loop (lines 4-8), at each iteration i (i runs from $d+1$ to k):

Line 5: finding the maximum of the vector $(b_j^T u_d)^2$ requires $O(n-i)$ operations.

Line 6: a rank-one update of the eigenpair (λ_d, u_d) is required. Mitz and al. proposed an algorithm in $O(d)$ arithmetic operations to perform the rank-one update of this pair by working on a partial sum of the secular equation [MSS19].

Line 7: the scalar product vector is updated using the Bénasséni formula (A.5) and (4.44); it requires the difference $\mu_d(i) = \lambda_d(\Sigma_i) - \lambda_d(\Sigma_{i-1})$ between the updated smallest eigenvalue and the previous iteration's eigen-

value, which is computed in $O(1)$ operations, and the dot product $u_d(\Sigma_i)^T u_d(\Sigma_{i-1})$ of the updated eigenvector with the previous one, in $O(d)$ operations. Then (A.5) gives

$$s_j(i) = \frac{\mu_d(i)}{s_j(i-1)} u_d(\Sigma_i)^T u_d(\Sigma_{i-1}),\ \forall j = d+1,...,k, \tag{4.44}$$

with $s_j(i) = b_j^T u_d(\Sigma_i)$. The coordinates $s_j(i-1) = b_j^T u_d(\Sigma_i)$ were already evaluated in the previous iteration, so updating $s_j(i)$ requires only $O(n-d-i)$ operations.

□

4.4.4 The WaterMaxVol algorithms

These algorithms implement different versions of a discrete water-filling technique to approach the solutions of (4.29). The first version has already been the subject of a publication [PRM23], while the other two are new. The general principle is the following: $k-d$ columns are selected to help maximize the smallest eigenvalues. For each eigenvalue, the columns most correlated with the related eigenvector are selected to specifically increase this eigenvalue up to the common optimal water level.

Three operations are performed sequentially to determine which columns should be assigned to which eigenvalue and how their selection contributes to the increase of this particular eigenvalue and to the equalization of the spectrum:

1. Estimate $\overline{\lambda} - \alpha$ and each ϵ_j for $j = 1,...,d$.
2. Evaluate and rank the scalar products $(u_j^T b_k)^2$, where u_j eigenvector related to $\lambda_j(\Sigma_d)$.
3. Determine the number c_j of selected columns related to $\lambda_j(\Sigma_d)$ and their indices.

1. $\overline{\lambda} - \alpha = W^{-1}(k-d)$ is numerically evaluated, which is straightforward since W is piecewise linear. For each $j = 1,..,d$, $\epsilon_j = (\overline{\lambda} - \alpha - \lambda_j)_+$ is computed.

2. From (4.28) and Appendix A.1, the contribution ϵ_j to $\lambda_j(\Sigma_d)$ is positively correlated with the angle $(u_j^T b_i)^2$ between any column b_i and an eigenvector u_j of $\lambda_j(\Sigma_d)$. We evaluate $S = Q^T B = \left(u_j^T b_i\right)_{ij}$ and for each u_j, we rank the $n-d$ quantities $(u_j^T b_i)^2$ in decreasing order, to identify the indices of the most correlated columns to u_j. The rank matrix R contains, in its j-th row, the indices of $(u_j^T b_i)^2$ sorted in decreasing order.

3. The exact contribution to ϵ_j from the $(u_j^T b_i)^2$, where $i = d+1,...,k$, is not known. Since eigenvalues cannot decrease when adding columns to B_d, a good strategy is to prioritize the smallest eigenvalues first by choosing, in decreasing order, the c_j columns most correlated with u_j. Given the finite total contribution, this strategy penalizes the higher eigenvalues and narrows the spread of the final spectrum of Σ_k. In this third step, three different strategies are possible, giving different variants of the proposed algorithm:

- In the first variant, for each j the number of columns c_j contributing the most to $\lambda_j(\Sigma_d)$ is chosen proportionally to ϵ_j (effectively treating $(u_j^T b_i)^2$ as 1). The columns are selected one by one, starting with those related to the smallest eigenvalue, until $k-d$ columns are selected. In this case,

$$c_j = \left\lceil (k-d)\frac{\epsilon_j}{||\epsilon||_1} \right\rceil,\ j = 1,...,d. \tag{4.45}$$

- The second variant also selects the columns contributing to a given eigenvalue one by one, starting with $\lambda_d(\Sigma_d)$ and moving in increasing order. For each eigenvalue λ_j, the algorithm evaluates the number c_j of

columns b_i related to $\lambda_j(\Sigma_d)$ by adding their contributions $(u_j^T b_i)^2$ until the total contribution exceeds the value ϵ_j.

- The last variant starts with the smallest eigenvalue $\lambda_d(\Sigma_d)$ and selects columns in decreasing order of their correlations $(u_j^T b_i)^2$. For each $j = d,...,2$, the algorithm continues to select columns related to $\lambda_j(\Sigma_d)$ until the sum of $(u_j^T b_i)^2$ exceeds the gap $\delta_j = |\lambda_{j-1}(\Sigma_d) - \lambda_j(\Sigma_d)|$. Once this gap is exceeded, columns related to $\lambda_{j-1}(\Sigma_d)$ are selected along with those related to $\lambda_j(\Sigma_d),...,\lambda_d(\Sigma_d)$, continuing until $k-d$ columns are selected.

The three methods of selection are illustrated in Fig. 4.4.

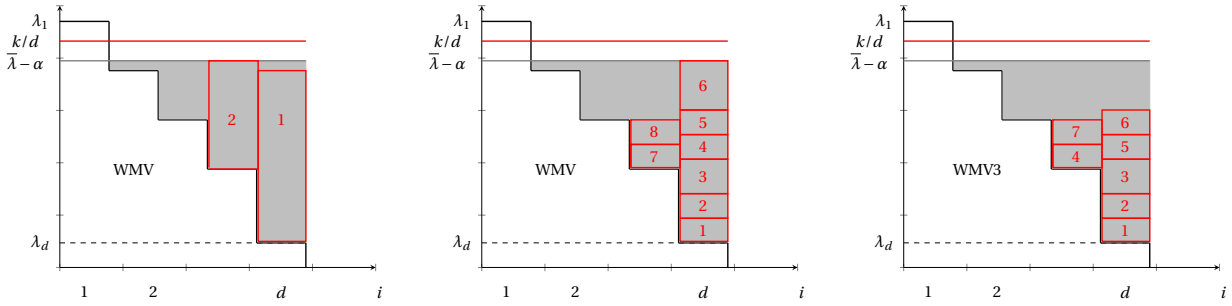

Figure 4.4 – WaterMaxVol algorithms. Red numbers indicate the order of selection of the columns, for the three variants WMV1-3.

Implementation of WaterMaxVol

The initialization step produces a matrix B_d whose columns form the first d selected columns and whose eigenvalues set the initial $(\lambda_j(\Sigma_d))_j$ upon which the water-filling algorithm is run. Then, the vector $c = (c_j)_j \in \mathbb{N}^d$ dictates, through the water-filling technique, how many columns should contribute to each eigenvalue. This number is given by (4.45). To minimize rounding errors and ensure that the smallest eigenvalues are given priority, the update of c_j is performed iteratively: c_d is evaluated first, followed by c_{d-1}, and so on, until the total number of $(k-d)$ columns is assigned.

Complexity:

- the initialization step has the complexity of the chosen method, for example $O(nd^2)$ for the MAXVOL algorithm.
- diagonalization of B_d: $O(d^3)$ operations are required.
- evaluation of $S = Q^T B$: this step requires $O(nd^2)$ operations.
- ranking of the resulting matrix: ordering d rows of n elements takes $O(dn\ln n)$ operations.
- evaluation of ϵ: this can be done in $O(n)$ operations.
- each of the $k-d$ iterations in the while loop takes a constant number of operations, so the complexity of the loop is $O(k-d)$.

The only difference between the three variants of the algorithm is the order by which the columns dedicated to a given eigenvalue are selected.

Algorithm 3 WaterMaxVol (variant 1)

Require: $B \in \mathbb{R}^{d\times n}, k \in [\![d,n]\!]$
Ensure: $B_k \in \mathbb{R}^{d\times k}$
Initialize $B_d \in \mathbb{R}^{d\times d}$ ▷ exhaustive search or MaxVol or FastInit
$J = \emptyset$
Diagonalize $\Sigma_d = B_d B_d^T = Q^T DQ$,
$Q = (u_1,..,u_d), D = \text{diag}(\lambda) = \text{diag}(\lambda_1,..,\lambda_d)$
Evaluate $S = Q^T B$ and $S^{\bullet 2} = \left((u_j^T b_i)^2\right)_{ij}$
R rank matrix of $S^{\bullet 2}$
Evaluate $\overline{\lambda} - \alpha = W^{-1}(k-d)$ and $\epsilon = (\epsilon_1,...,\epsilon_d)$
j=d
while $j > 0$ **do** ▷ evaluate $c = (c_1,...,c_d)$
    $c_j = \text{ceil}\left((k-d)\epsilon_j / ||\epsilon||_1\right)$
    Select c_j first indices in row i of $R : J_j$
    $J = J \cup J_j$
    j=j-1
end while
Return B_J

Algorithm 4 WaterMaxVol (variant 2)

Require: $AB \in \mathbb{R}^{d\times n}, k \in [\![d,n]\!]$
Ensure: $B_k \in \mathbb{R}^{d\times k}$
Initialize $B_d \in \mathbb{R}^{d\times d}$ ▷ exhaustive search or MaxVol or FastInit
$J = \emptyset$
Diagonalize $\Sigma_d = B_d B_d^T = Q^T DQ$,
$Q = (u_1,..,u_d), D = \text{diag}(\lambda) = \text{diag}(\lambda_1,..,\lambda_d)$
Evaluate $S = Q^T A$ and $S^{\bullet 2} = \left((u_j^T b_i)^2\right)_{ij}$
R rank matrix of $S^{\bullet 2}$
Evaluate $\overline{\lambda} = W^{-1}(k-d)$ and $\epsilon = (\epsilon_1,...,\epsilon_d)$
for j from d to 1 **do**
    $\psi = 0;\ i = 1;\ c_j = 0$
    while $\psi < \epsilon_j$ and $\sum c_j \le k-d$ **do**
        $\psi = \psi + S^{\bullet 2}(j, R(j,t))$
        ▷ t next column available, ψ cumulate $(u_j^T b_i)^2$
        $J = J \cup t;\ I = I \setminus t;\ i = i+1;\ c_j = c_j + 1$
    end while
end for
Return B_J

Algorithm 5 WaterMaxVol (variant 3)

Require: $B \in \mathbb{R}^{d\times n}, k \in [\![d,n]\!]$
Ensure: $B_k \in \mathbb{R}^{d\times k}$
Initialize $B_d \in \mathbb{R}^{d\times d}$ ▷ exhaustive search or MaxVol or FastInit
$J = \emptyset$ column indices of B, $I = [\![d,n]\!] \setminus J$
Diagonalize $\Sigma_d = B_d B_d^T = Q^T D Q$,
$Q = (u_1,..,u_d), D = \text{diag}(\lambda) = \text{diag}(\lambda_1,..,\lambda_d)$
Evaluate $S = Q^T B$ and $S^{\bullet 2} = \left((u_j^T b_i)^2\right)_{ij}$
R rank matrix of $S^{\bullet 2}$
total= 0
for j from d to 1 **do**
    $\psi = 0$ ▷ contains cumulative $(u_j^T b_i)^2$
    for i from d to i **do**
        $\psi = \psi + S^{\bullet 2}(i, R(i,t))$
        ▷ t next column available
        $J = J \cup t$; $I = I \setminus t$; total = total + 1
        Stop if $\psi > \lambda_j - \lambda_{j+1}$ or total= $k-d$
    end for
end for
Return B_J

Table 4.2 – The 5 proposed algorithms and algorithms to which we compare them.

Name	Complexity (init.+main part)
MaxVolDiv	$O(nd) + O(nkd)$
MaxVolCorr	$O(nd) + O(n(k-d))$
WaterMaxVol V1	$O(nd) + O(nd^2)$
WaterMaxVol V2	$O(nd) + O(nd^2)$
WaterMaxVol V3	$O(nd) + O(nd^2)$
RectMaxVol [MO17]	$O(nk^2) + O(nk^2)$
DPPs [TBA18]	$O(n^3) + O(nk^2)$

The overall complexity is thereby $O(nd^2)$ operations, with d assumed to be much smaller than n. By using efficient matrix multiplication algorithms, it is possible to reduce the complexity of each cubic power step to $2+\omega$, where $\omega \in]0,1[$.

To conclude the paragraph, we summarize the different proposed and competing algorithms in Table 4.2

4.5 Simulations and performances

In the following figures, we compare the performances of different algorithms with known methods for volume maximization. The initial matrix $B = (b_{ij})$ is constructed from a standard Gaussian matrix, with its columns subsequently normalized.

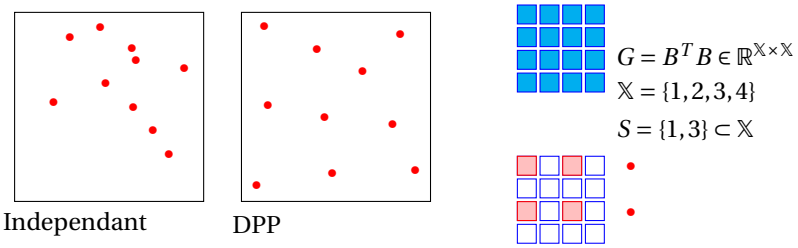

Figure 4.5 – Illustration of determinantal point processes. Left: two samples of random points (red dots) in the square $[0,1]^2$, of size 10, from an uniform independent distribution (left black square) and a DPP distribution (right black square). Right: from a database with 4 items $\{1,2,3,4\}$, the sample $\{1,3\}$ has a probability proportional to $\det G_k$ to be selected.

Presentation of Competing Methods

- Uniform: The extracted columns are selected uniformly at random.
- The RectMaxVol algorithm, proposed by Mikhalev and Oseledets [MO17], and MaxVolDiv optimize the same objective but have different implementations. They achieve the same volume. RectMaxVol includes an initialization step with a complexity of $O(nk^2)$ operations, due to the MaxVol algorithm, followed by a main loop with $O(nk^2)$ operations.
- DPP-based methods: A classical and efficient way to perform CSS with the objective of maximizing volume is to use determinantal point processes (DPPs). For a comprehensive review of DPPs and kernel methods we refer the reader to [KT12; BBC20; Lau20]. We defined two L-ensembles, a subclass of DPPs well suited to our framework, and restrict ourselves to k-DPPs, which are defined by a probability distribution on subsets of columns with cardinality exactly k. The first DPP is defined by its Gram matrix $G = B^T B$, which is modified to $B^T B + \delta I$ to ensure it is invertible (DPPGram). The second is defined by a specific cosine kernel ϕ that promotes diversity (DPPKer), with $G = (G_{ij})$ and $G_{ij} = \langle \phi(b_i), \phi(b_j) \rangle$. The traditional spectral volume sampling algorithm for DPP involves three steps. The initialization step requires $O(n^3)$ operations due to the eigendecomposition of the kernel matrix, while the third step scales as $O(nk^3)$ operations. However, there exists an algorithm that reduces the complexity of the third step to $O(nk^2)$ operations [TBA18]. See Fig. 4.5 for an illustration.

We summarize the complexity of all algorithms in Table 4.2.

Analysis of the simulations

Fig.4.6: The figure illustrates the effect of the algorithm MaxVolDiv on the spectrum of the extracted matrices B_k, $d \le k \le n$, compared to a random column extraction.

Analysis: we observe that by maximizing the volume, the greedy algorithm reduces the spectrum's spread and increases the smallest eigenvalue, which is consistent with the principles stated in Section 4.2. Interestingly, this spectrum crushing is noticeable as long as the number of extracted columns is on the order of $n/2$, where n is the number of available columns. Beyond this threshold, as the possible choices of columns diminish, the spectrum widens until it matches the spectrum of the original matrix, which occurs when $k = n$ (i.e. when no columns are removed).

It is important to note that adding columns selected uniformly at random also reduces the spread of the spectrum, due to the strong law of large numbers: if we fix d and let k, n tend to infinity, due to normalization, the columns of B and B_k are independent uniform random vectors on the d-dimensional sphere. The covariance matrix of any column b is then $\mathbb{E}[bb^T] = I_d/d$, and by the strong law of large numbers:

$$\frac{1}{n}\Sigma = \frac{1}{n}\sum_{i=1}^{n} b_i b_i^T \xrightarrow[\text{a.s.}]{n\to+\infty} = \mathbb{E}[bb^T] = \frac{1}{d} I_d. \tag{4.46}$$

For sufficiently large n, Σ is almost surely close to $(n/d)I_d$. However, Fig.4.6 shows that MaxVolDiv has a far stronger effect on the spectrum than uniform sampling .

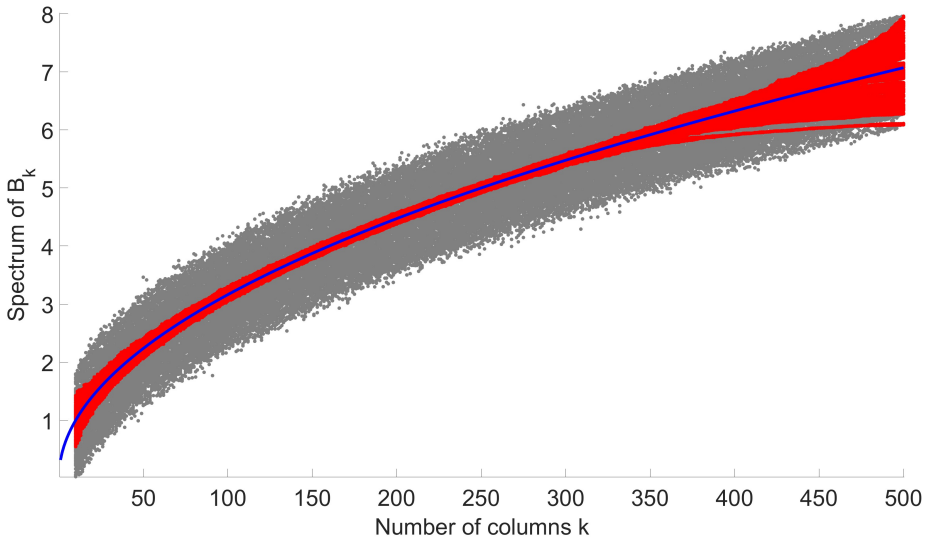

Figure 4.6 – The spectrum of the extracted submatrices $B_k \in \mathbb{R}^{d\times k}$ from the matrix $B = (b_{ij}) \in \mathbb{R}^{d\times n}$ as a function of the number of columns k. B is a standard Gaussian matrix with its columns normalized. The spectrum of submatrices B_k extracted randomly with a uniform distribution is shown in grey, while the sprectum of submatrices extracted using the MaxVolDiv algorithm is shown in red. The mean curve $\sqrt{k/d}$ is indicated in blue. Parameters are set to $d = 10$, $n = 500$, and 10 independent samples are superimposed.

Fig 4.7: we study a case where it is possible to perform an exhaustive search to evaluate the gap between sub-optimal algorithms and the optimal one. Due to the complexity of exhaustive search, this study is conducted with small values of n and k.

Analysis: we observe that greedy algorithms achieve much larger volumes than DDP-based methods, which, it should be noted, are not specifically designed for this context. They are also outperform uniform sampling. Most importantly, greedy methods are very close to the optimal solution, even in its fast version (quadratic rather than cubic). Finally, when k is close to n, all the methods yield similar volumes since the possible choices of columns is limited by n, resulting in almost identical sets of selected columns across all methods. It should also be noted that the scale is logarithmic.

Fig 4.8 : we compare the performance of all algorithms, for $d = 10$, $n = 200$.

Analysis: the proposed Waterfilling algorithms for both variants 1 and 2 yield an extracted matrix with a volume similar to that of MaxVolDiv and MaxVolCorr, achieving performance close to these algorithms. Thus,

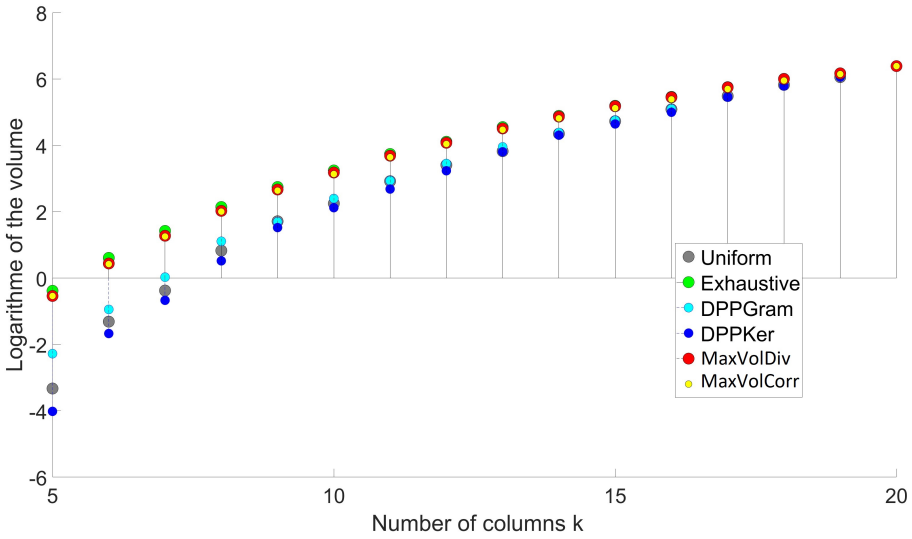

Figure 4.7 – Maximum volume comparison of 6 algorithms, as a function of k varying from 5 to 20 ($d = 5$, $n = 20$). The average of 10 independent samples of the logarithm of volume are given in ordinate. Blue dots: two DPPs algorithms (Gram L-ensemble and cosine kernel). Green: real maximum by exhaustive search. Grey: uniform random choice. Red: MaxVolDiv and yellow: MaxVolCorr. $B = (b_{ij}) \in \mathbb{R}^{d\times n}$ Gaussian with $b_{ij} \sim \mathcal{N}(0,1)$ i.i.d. coefficients, with columns subsequently normalized.

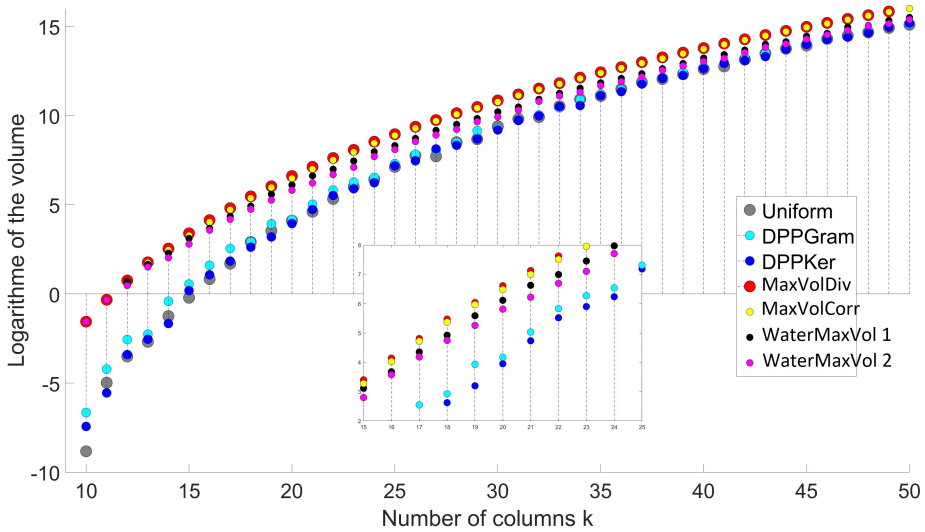

Figure 4.8 – Maximum volume comparison of 7 algorithms, as a function of k varying from 10 to 50 ($d = 10$, $n = 200$). The average of 10 independent samples of the logarithm of volume are given in y-axis. Blue curves: two DPPs algorithms (Gram L-ensemble and cosine kernel). Grey: uniform random choice. Greedy algorithms MaxVolDiv and MaxVolCorr are respectively in red and yellow. One-shot algorithms WaterMaxVol V1 and V2 are respectively in black and magenta. $B = (b_{ij}) \in \mathbb{R}^{d\times n}$ Gaussian with $b_{ij} \sim \mathcal{N}(0,1)$ i.i.d. coefficients, with columns subsequently normalized.

the performance gap remains significant compared to DPP-based methods, particularly for the lower half of the k values.

Fig 4.9: we examine the behavior of the algorithms in scenarios involving a strongly inhomogeneous database. In this case, a mixture of a random number of Gaussians is generated. For each Gaussian in the mixture, a proportion, mean and variance are either fixed or randomly selected. The means are used as angles to create clusters of vectors on the surface on the unit sphere in $\mathbb{R}^d$. These vectors constitute the columns of B. The simulations reveals some irregularities in performance; however, the inhomogeneity does not significantly impact the relative performances of the algorithms.

The implementation of DPPs was done using MATLAB leveraging the code provided by Alex Kulesza. Our implementation utilizes the MAXVOL algorithm during the initialization step, as described in the works of Goreinov and al. [Gor+; MO17]. The MAXVOL algorithm is available in MATLAB as part of the TT-Toolbox of Ivan Oseledets.

4.6 Examples of applications

We present two applications related to volume maximization: matrix conditioning and sparse support recovery in compressive sensing.

4.6.1 Matrix conditioning

In numerous applications, ensuring well-conditioned matrices is crucial, making matrix conditioning a subject in its own right. The condition number of a square non-singular matrix B relative to the norm $||.||$ is defined by

$$\kappa(B) = ||B|| \times ||B^{-1}||. \tag{4.47}$$

It can be extended to rectangular matrices for the Euclidean induced norm by letting $\kappa(B) = \sigma_n/\sigma_d$ for a $d \times n$ matrix of rank d. By construction, our algorithms reduce the spread of the spectrum, as illustrated in Fig.4.10.a, and produce extracted matrices whose covariance matrix closely approximate a scalar matrix. This results in a condition number close to 1, the optimal value for $\kappa(B)$. Figure 4.10.b shows the performance of the deterministic MaxVolDiv and MaxVolCorr algorithms for different sizes of columns selections from the initial matrix.

Thus, to construct a well-conditioned random matrix, one solution is to generate a large matrix, from which a maximum volume submatrix is extracted. This construction is what we study in the following application.

4.6.2 Sparse support recovery in compressive sensing

The goal of this problem is to recover the support Λ of a sparse signal $x \in \mathbb{R}^n$ observed only through k linear measurements of its coordinates. This problem can be modeled as a linear regression

$$y = Bx \tag{4.48}$$

where $y \in \mathbb{R}^d$ is the observed vector and $B \in \mathbb{R}^{d\times n}$ a sensing matrix. The support of x has cardinality $k \ll n$. Solving this underdetermined linear system (4.48) with the sparsity constraint can be formulated as the

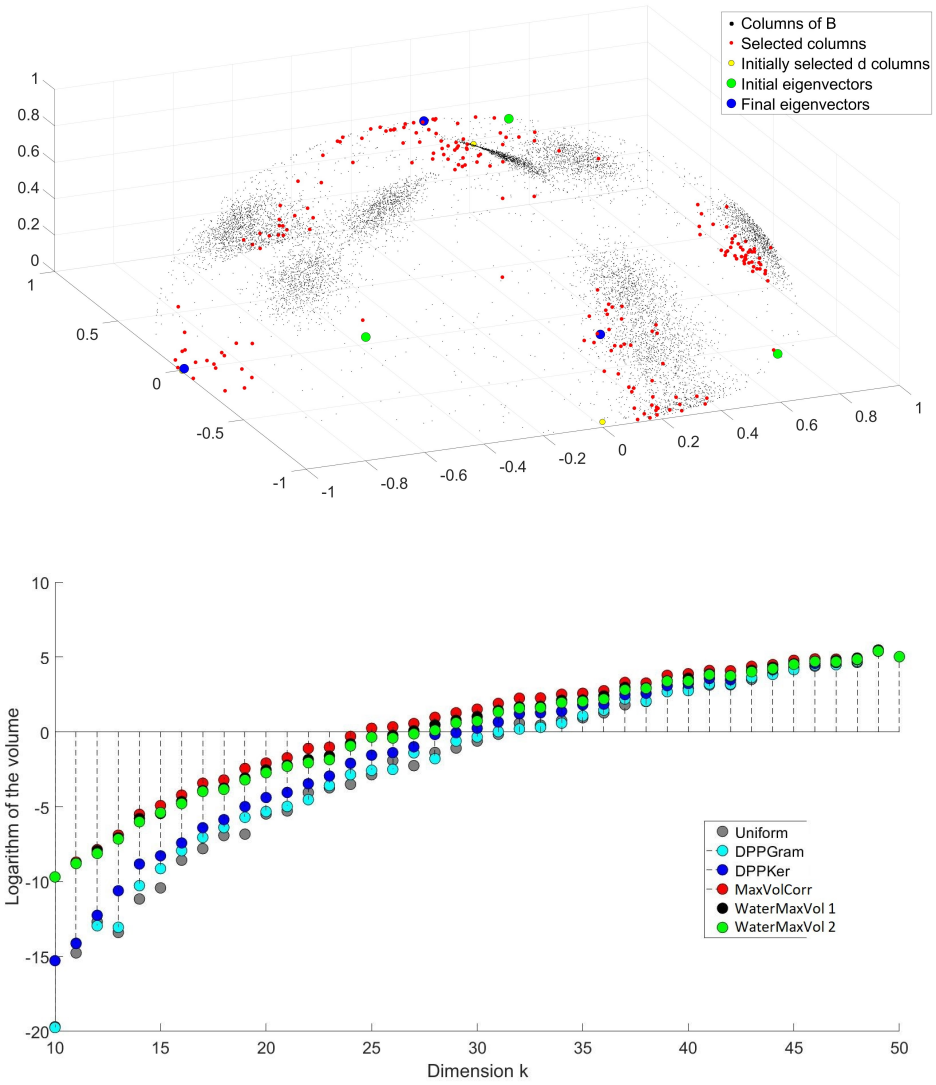

Figure 4.9 – Behavior of the algorithm MaxVolCorr in the case of an inhomogeneous database. A mixture of a random number of Gaussians with different directions (angles in $\mathbb{R}^d$), variances and sizes results in an inhomogeneous distribution, forming irregular clusters on the surface of the unit sphere. Top: visualization in a low-dimensional space ($d = 3$, $k = 200$, $n = 10^4$); black dots represent the feature vectors of the datamatrix. Yellow dots indicate the initially selected columns formnig B_d. Red dots are the selected columns for B_k after running MaxVolCorr algorithm. The initial and final eigenvectors of Σ_d and Σ_m are shown in green and blue, respectively. Bottom: the performances of several algorithms for varying number of columns k, considering an inhomogeneous distribution ($d = 10$, $n = 50$).

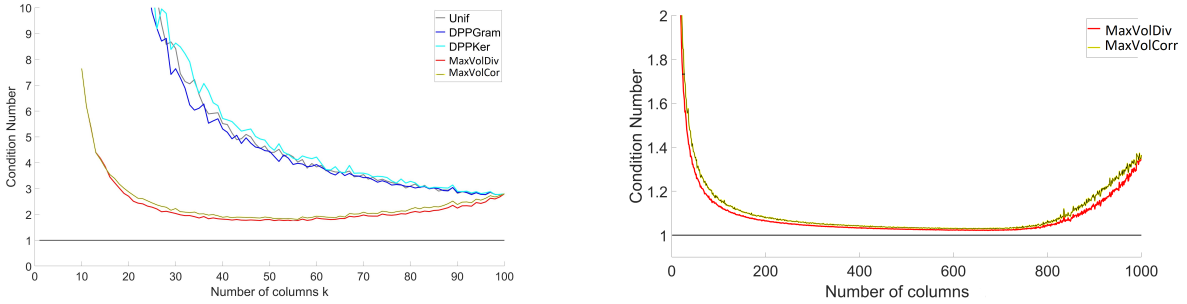

Figure 4.10 – Condition number of extracted submatrices with MaxVolDiv and MaxVolCorr algorithms. Left: comparison with DPP and uniform sampling in dimension $d = 10$, $n = 100$. Right: performance in dimension $d = 10$, $n = 1000$, $d \leq k \leq n$. The horizontal line indicates the threshold $\kappa = 1$, the lowest possible condition number. Each point of the curves represents the average of 100 independent samples of standard Gaussian matrices.

following $\mathscr{P}_0$ optimization problem

$$\mathscr{P}_0 : \begin{cases} x^\star = \underset{x}{\arg\min} ||x||_0 \\ \text{s.t. } Bx = y \end{cases} \tag{4.49}$$

where $||x||_0 = |\Lambda| = k$. $\mathscr{P}_0$ is an NP-hard problem, but several algorithms can solve it with reasonable complexity.

The existence and uniqueness of the solution depends deeply on the algebraic properties of the sensing matrix B, which can be designed by the user. One way to quantify the quality of the sensing matrix is through the restricted isometry property (RIP), defined by Candès and Tao in their pioneering work [CT05]. The RIP measures how close B is to an isometry when restricted to k-sparse vectors. Formally, this is equivalent to proving the existence of a constant $\delta_s \in]0,1[$ such that, for every submatrix B_s of size $d \times s$ from B, all the eigenvalues of $B_s B_s^T$ lie in the interval $[1-\delta_s, 1+\delta_s]$. The RIC constant δ_k is the smallest constant δ_s for $|s| \leq k$. Recovery performance is directly linked to the RIC, which should be as small as possible [DW10], as this provides a sufficient condition for OMP to recover any k-sparse signal. Translating this definition in terms of the condition number κ of $B_s B_s^T$ gives:

$$\delta_k \geq \frac{\kappa - 1}{\kappa + 1}, \tag{4.50}$$

which grows with κ, for $\kappa \geq 1$. Therefore, a well-conditioned matrix has $\kappa \sim 1$.

Our algorithms allow the possibility for the design of sensing matrices by choosing the size of the matrix. Starting with a initial $d \times n$ matrix with n carefullty chosen columns, the resulting sensing matrix will be an extracted submatrix of size $d \times k$, with a condition number close to 1. Figure 4.11 illustrates the performance of a recovery algorithm by comparing random (left) and optimized (right) sensing matrices. The performance gain by using a matrix optimizing the condition number is significant. We will revisit compressive sensing in Chap. 3 and Chap. 6 (with slightly different notations).

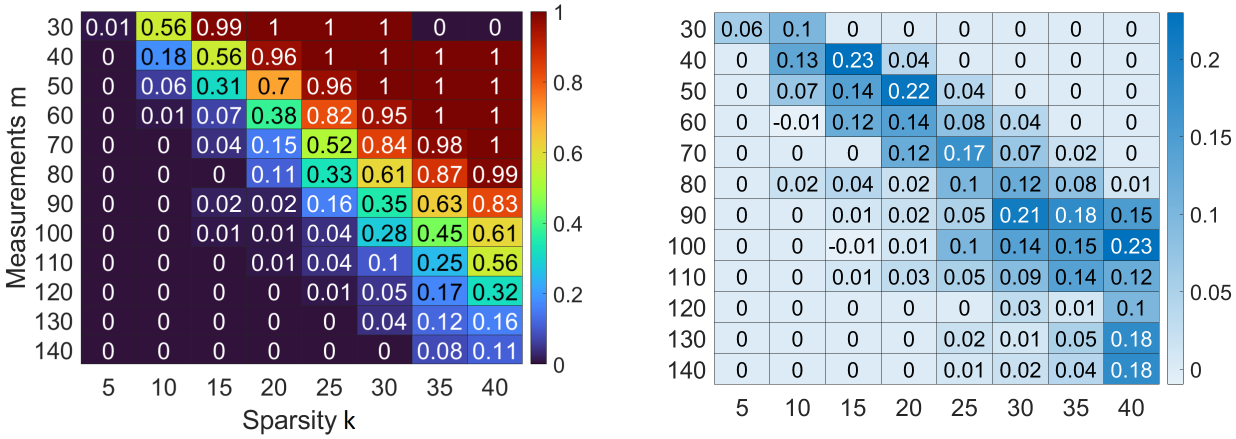

Figure 4.11 – Performances of the orthogonal matching pursuit algorithm (OMP). Left: the color boxes indicate the probability of failure (i.e., not recovering the correct indices of the support from a signal x). Red signifies a high probability of failure, while blue indicates success. The x-axis represents increasing sparsity parameter k, and the y-axis represents *decreasing* overmeasure parameter m. The more sparse a vector is and the more measurements are made, the easier it is to recover. The algorithm is run with optimized matrices (extracted using MaxVolCorr) with a condition number close to one. Right: the algorithm is run with random non-optimized matrices and the *difference* in the probability of failure compared to the previous case is shown. A positive number indicates improved performance. Parameters are set to $d = 200$ and $n = 400$, with k varying from 5 to 40 and k from 30 to 140. Probabilities are averages over 100 independent samples of standard Gaussians matrices (known for their good condition numbers).

4.7 Conclusion of this chapter

We studied the submatrix extraction problem through volume maximization. By constraining the matrix to have normalized columns, we have formally shown that the continuous relaxation of this problem is equivalent to spectrum equalization and the minimization of the smallest eigenvalue.

Furthermore, we proposed several algorithms to solve the discrete volume extraction problem:

- A new initialization method with quadratic complexity, which, although suboptimal, minimally impacts the final volumes of the algorithms.
- A new implementation of the recursions in the greedy algorithm RectMaxVol, making it slightly less complex than the original greedy approach ($O(nd)$ instead of $O(nk^2)$ for the initialization step and $O(nkd)$ instead of $O(nk^2)$ for the main loop).
- A new greedy algorithm with quadratic complexity, achieving performance close to that of the cubic algorithm.
- Finally, algorithms that solve the problem in a single step, providing performance equivalent to that of greedy algorithms.

We proposed two direct applications of these algorithms: matrix conditioning and compressive sensing.

CHAPTER 5

GRAPH SPARSIFICATION PRESERVING CONNECTIVITY

5.1 Objectives

In this chapter, we propose solutions for sparsifying a graph while preserving its connectivity, using tools and results discussed in Chap. 4 on database compression. The two chapters will therefore have a very similar structure. The problem, related works and notations used here are described in Chap. 2.

We present two greedy algorithms for sparsification to extract a sparse unweighted subgraph with maximum connectivity for a given number of edges. These algorithms rely on maximizing the volume of the Laplacian submatrix through an iterative process, selecting edges based on their effective resistance. The optimization problem boils down to maximizing the volume of a matrix, where the greedy approach involves maximizing quantities related to the effective resistance of an edge. This principle forms the basis of the first proposed algorithm. Its algebraic interpretation suggests that the spectrum should be as narrow as possible. A continuous relaxation of this principle leads to the second greedy algorithm, which maximizes the smallest positive eigenvalue of the Laplacian, with a quadratic complexity in terms of number of operations. Our algorithms are deterministic, both having quadratic time complexity, designed for unweighted graphs. One (GSMVDIV) is optimal among all greedy algorithms that maximizes connectivity by selecting one edge at a time.

We propose an application of our algorithms in the context of learning on graphs, specifically in Graph Neural Networks (GNNs). We explore the possibility of learning on a sparsified graph to reduce the complexity of both learning and inference. Our results show that our algorithms can effectively sparsify the graph while maintaining the performance of the GNN.

The chapter is organized as follows: section 2 briefly reviews the notations, the framework and formalizes the main optimization problem related to sparsification. Section 3 presents our two solutions: an optimal greedy process linked to the effective resistance of edges, and an approximate solution obtained through a continuous relaxation of the initial problem. Section 4 describes the algorithms, their implementations and complexity, while section 5 and 6 are dedicated to simulations and applications.

5.2 Problem formulation

Let's recall that $G = (V, E)$ is an undirected and unweighted graph, where V is the set of vertices and E is the set of edges. $B \in \mathbb{R}^{n \times m}$ represents the incidence matrix of the graph and $L = L(G) = BB^t \in \mathbb{R}^{n \times n}$ is the combinatorial Laplacian of G.

The importance of the notion of connectivity in a graph [Fre+23; Mar+18; EK13; LB07; SBG23; AE12; MSN10] leads us to seek a sparse subgraph that preserves connectivity. From Thm. 1, this is equivalent to maximizing the volume of the graph Laplacian. Thus, the problem can be formulated as follows:

Problem 8 (Maximization of the connectivity)**.** *Let k be the number of edges to be kept, $n \leq k \leq m$. Find:*

$$\underset{B_k : B_k \subset B}{\operatorname{argmax}} \operatorname{vol}(B_k). \tag{5.1}$$

Maximizing the volume of the incidence matrix B_k of the sparsified graph with k edges is equivalent to maximizing the volume of the corresponding Laplacian matrix $L_k = B_k B_k^T$: the mapping from B to $L = BB^T$ is onto by definition of a symmetric matrix, while this map is not necessarily one-to-one. But if $BB^T = B'B'^T$, then $\operatorname{vol}(B) = \operatorname{vol}(B')$.

5.3 Problem analysis and resolution

5.3.1 Optimal greedy solution

Finding the maximum volume submatrix of a given size is a NP-hard problem [ÇM09]. A straightforward and natural, though sub-optimal, approach to maximize the volume among all possible choices of m columns is to greedily concatenate to B_k one column b at each iteration, such that (B_k, b) is of maximum volume:

Problem 9 (Discrete greedy maximization of the volume)**.** *Select at each iteration a column from $B \setminus B_k$ belonging to*

$$\hat{b} = \underset{b \in B \setminus B_k}{\operatorname{argmax}} \, b^T L_k^+ b \tag{5.2}$$

and update $L_{k+1} = (B_k, \hat{b})(B_k, \hat{b})^T$.

Proof. (Links between Pb. 8 and Pb. 9). The decomposition of L_{k+1} in rank-one operators is

$$L_{k+1} = \sum_{i=1}^{k} b_i b_i^T + bb^T = L_k + bb^T, \tag{5.3}$$

where bb^T is a rank-one operator. Equation (5.3) connects the volume of the augmented matrix to that of the initial matrix:

$$\begin{aligned} \operatorname{vol}(B_k, b)^2 &= \operatorname{vol}(L_k + bb^T) && (5.4)\\ &= \operatorname{vol}(L_k) \times (1 + b^T L_k^+ b) && (5.5)\\ &= \operatorname{vol}(B_k)^2 \times (1 + b^T L_k^+ b), && (5.6) \end{aligned}$$

where (5.6) follows from a generalized matrix determinant lemma, which we prove below. □

The previous equation shows that maximizing the volume is equivalent to maximizing $b^T L_k^+ b$ over all choices of b. This also demonstrates the optimality of the method when restricted to a greedy process that selects one column at each iteration.

The classical matrix determinant lemma can't be applied here because the matrix L is not of full rank. We have to use a generalized version of this lemma:

Lemma 3.

$$\text{vol}(L + bb^T) = \text{vol}(L) \times (1 + b^T L^+ b). \tag{5.7}$$

Proof. (Generalized matrix determinant lemma) Without loss of generality, we can work in an eigenbasis of L, assuming u_1 is the eigenvector associated to $\lambda_1 = 0$. Let $L_\bullet$ denote the reduced Laplacian matrix obtained by deleting the first row and column of L. $L_\bullet$ is invertible and $\text{vol}(L) = \det L_\bullet$. Let $b_\bullet$ be the vector obtained from b by deleting its first coordinate. According from the matrix determinant lemma,

$$\det(L_\bullet + b_\bullet b_\bullet^t) = \det L_\bullet \times (1 + b_\bullet^T L_\bullet^{-1} b_\bullet). \tag{5.8}$$

Moreover,

$$L^+ = \text{diag}(0, \lambda_2^{-1}, ...\lambda_n^{-1}) \text{ and } L_\bullet^{-1} = \text{diag}(\lambda_2^{-1}, ...\lambda_n^{-1}), \tag{5.9}$$

so that $b^T L^+ b = b_\bullet^T L_\bullet^{-1} b_\bullet$, which proves the proposed result. □

Different scenarios will be proposed in the next section: either starting with a single edge and iteratively adding $k-1$ edges, or starting from a suitably chosen spanning tree with $n-1$ edges and adding iteratively $k-n+1$ edges. Another natural approach for sparsifying a graph is to start from the complete graph and delete one edge at each iteration. In this case, the edge to be removed should be the one that minimizes $b^T L_k^+ b$

Interpretation

Let G be a generic graph of Laplacian L. For any pair of vertices (v_s, v_t), the quantity

$$R_{st} = (\delta_s - \delta_t)^T L^+ (\delta_s - \delta_t) \tag{5.10}$$

represents the effective resistance between v_s and v_t, measuring the connectivity and robustness of the graph between these vertices (see Th. 4). In particular, if v_s and v_t are the endpoints of an edge in G (corresponding to the column $b_e \in B$, where $e = (s,t)$), $R_e = b_e^T L^+ b_e$ measures the criticality of this edge with respect to the graph's connectivity.

At each iteration, the algorithm in Pb. 9 evaluates all available edges from G and selects those with the highest effective resistance. A high effective resistance between two vertices indicates a bottleneck in the graph, where information is poorly transmitted due to the limited number of paths or low conductance. Adding such an edge to G reduces the effective resistance between the two vertices (and the total resistance of the graph), thanks to this new connection, thereby increasing the graph's robustness. In a sense, this algorithm can be viewed as a deterministic version of Spielman and Srivastava's algorithm (see Th. 9), with the key differences

being that it is applied to an unweighted graph and to preserve the connectivity instead of the Laplacian spectrum.

Let us examine the way each edge modifies the spectrum of the graph. Without loss of generality, assume we are working in an eigenbasis of L. We have:

$$R_b = b^T L^+ b = \sum_{j=2}^{n} \frac{b_j^2}{\lambda_j}. \tag{5.11}$$

For the sake of simplicity, we keep the notation b without a change of basis. R_b is a function of b and L and depends on whether b represents an edge of G or not. Let s,t be the indices of the nonzero coordinates of b and let $e=(s,t)$. Define $\tilde{G}$ as the graph with edge set $\tilde{E}=E\cup\{e\}$. Let $(\lambda_i)_i$ and $(\tilde{\lambda}_i)_i$ be the eigenvalues of L and $L+bb^T$, respectively. The effect of adding to L the rank-one operator bb^T can be summarized by the following sets of equalities [Gol73; IN09; BNS78] (cf. Appendix A.1):

$$\begin{cases} \tilde{\lambda}_j = \lambda_j + \mu_j, & j=2,...,n \\ \\ \sum_{j=2}^{n} \mu_j = ||b||^2,\ 0 \le \mu_j \le ||b||^2, & j=2,...,n \end{cases} \tag{5.12}$$

The equality $\tilde{\lambda}_j = \lambda_j + ||b||^2$ occurs if, and only if b is an eigenvector of L corresponding to λ_j. In this case, all other eigenpairs remain unchanged and λ_j is increased by $||b||^2 = 2$ [IN09; BNS78]. If b is orthogonal to an eigenvector of L, the corresponding eigenvalue is not modified and belongs to the spectrum of $L+bb^T$. In all other cases, the contribution of $||b||^2$ to the spectrum of $L+bb^T$ is distributed among all eigenspaces of L, with a total contribution of $||b||^2 = 2$:

$$\mathrm{tr}(L+bb^T) = \mathrm{tr}(L) + \mathrm{tr}(b^T b) = \mathrm{tr}(L) + 2. \tag{5.13}$$

5.3.2 Approximate greedy solution by continuous relaxation

The expression of R_b in 5.11 shows that the smallest eigenvalues contribute the most to R_b. The continuous relaxation of Pb. 9, where $x=(x_j)_j=(b_j^2)_j$ is any vector of $[0,2]^{n-1}$ instead of a column b of B, can be easily solved using the classical linear programming techniques [BT97]. If we define

$$c = (c_2,..,c_n) = \left(\lambda_2^{-1},...,\lambda_n^{-1}\right), \tag{5.14}$$

Pb. 9 boils down to maximizing the dot product on a compact set:

$$b^T L^+ b = \sum_{j=2}^{n} \frac{x_j}{\lambda_j} = x^T c. \tag{5.15}$$

The continuous relaxation of Pb. 8 can also be solved easily. Each column of $b\in B$ verifies $||b||^2=2$, so $\mathrm{Tr}L_k = 2k$. The problem then reduces to finding the maximum determinant under the constraint of a constant trace: $\mathrm{vol}(L)=\prod_{j=2}^{n}\lambda_i$ and $\mathrm{Tr}(L)=\sum_{j=1}^{n}\lambda_j = 2k$. The maximum of the product of $n-1$ positive numbers under a constraint sum is the case of equality in the inequality of arithmetic and geometric means [Ber99]. The equality

is reached if, and only if all the nonzero λ_j are equal and their common value is

$$\lambda = \frac{1}{n-1}\sum_{j=2}^{n}\lambda_i = \frac{2k}{n-1}. \tag{5.16}$$

The maximum volume is therefore λ^{n-1}, achieved for the scalar matrix $\mathrm{diag}(\lambda,...,\lambda,0) \in \mathbb{R}^{n\times n}$, whose spectrum reduces to only one nonzero (eigen)value.

This result is consistent with the previous greedy maximization scheme: as the total increase of eigenvalues is 2 at each iteration, increasing the smallest positive eigenvalues also penalizes the largest ones, narrowing the spread of the spectrum. We can summarize both strategies into two principles:

- Increasing the smallest eigenvalues of the Laplacian matrix maximizes the volume.
- Narrowing the spread of the spectrum of the Laplacian matrix maximizes the volume.

Following the first principle, we propose to approximates R_b by the most significant term b_2^2/λ_2 of the sum. In Appendix A.1 we provide tight bounds for the smallest nonzero eignevalue in terms of the coefficient b_2 and the gap $\delta_2 = \lambda_3 - \lambda_2$, upper bounding the increase of λ_2. In the canonical basis, the coefficient b_2^2 is replaced by the scalar product $(b^T u_2)^2$, where u_2 is the normalized eigenvector associated with λ_2 (the Fiedler vector). Thus, the method selects at each iteration the column most correlated to the Fiedler vector. This is formalized in the following problem:

Problem 10 (Approximate greedy maximization of the volume)**.** *At each iteration, select the column*

$$\underset{b\in B\setminus B_k}{\operatorname{argmax}} (b^T u_2)^2 \tag{5.17}$$

where u_2 is a normalized eigenvector related to the smallest eigenvalue λ_2.

Using the Fiedler vector to select, partition or classify data on graphs is clearly not new and this vector is implicated in many algorithms [CA16; SBG23; Wan+14; GBS08; Lux07], as discussed in Section 2.2.6. However, the originality in our algorithm is the way we use it: the greedy process used here updates at each iteration the entire spectrum of the current Laplacian. Consequently, the Fiedler vector of the next iteration integrates information from all eigenpairs and reflects the geometry of the entire graph, as the increase of the smallest eigenvalue modifies all the spectrum.

5.4 Algorithms and implementation

In this section, we apply the previous greedy strategies and propose several algorithms to solve the optimization problems Pb. 9 and Pb. 10 related to connectivity.

5.4.1 Optimal greedy algorithm: GSMVDIV

The Graph Sparsification Maximum Volume for Diversity algorithm (GSMVDIV) implements the solution to Pb. 9 by selecting the column maximizing $R_b = b^T L^+ b$ at each iteration.

This algorithm is an adaptation of MaxVolDiv, but whose principle dates back to the early 70s [Dyk71; MO17]. The original algorithm is run on a definite positive covariance matrix of rank n, while the present algorithm is run on the Laplacian of rank $n-1$ and possess a lower complexity.

Three variants are proposed. The first two variants start from an empty graph and add edges during two steps, differing only in the first step. The third variant deletes edges iteratively from the entire initial graph.

In the first variant $n-1$ edges are selected in one time: there exists algorithms that find spanning trees in nearly linear time [Sch18; Cha00]. A spanning tree has $n-1$ edges, a volume of exactly n, and this is the maximum possible volume for a connected graph of $n-1$ edges. Therefore, it is possible to start the algorithm with such a spanning tree and then selects $k-n+1$ additional edges to complete the sparsifier iteratively. Complexity: finding the highest resistance edge b requires $O(m)$ operations. Then, one has to evaluate $BB^T = bb^T$ and its pseudo inverse. Since b is sparse with 2 nonzero coordinates, the product bb^T requires $O(1)$ operations, and since $(bb^T)^+ = (bb^T)/4$, the pseudo-inversion takes $O(1)$ operations. In this case, the main loop has to be run $k-1$ times.

In the second variant, we select one edge after another; during the first $n-1$ iterations, $\dim\ker L_k = n-k$ and the algorithm selects an edge that reduces the dimension of the kernel by one, until $\dim\ker L_k = 1$ at iteration $n-1$ and thereafter. Complexity: if a spanning tree is initialized and replaces the first $n-1$ iterations, $O(m\ln n)$ operations are necessary. The pseudo-inverse L^+_{n-1} of the Laplacian has to be evaluated. This is possible with a nearly linear time solver for any Laplacian [ST06], but for the Laplacian of a spanning tree, the complexity is exactly linear in time [Spi10]. Then the main loop has to be run $k-n+1$ times.

Algorithm 6 GSMVDIV

Require: $B \in \mathbb{R}^{n\times m}, k \in [\![n, m]\!]$
Ensure: $B_k \in \mathbb{R}^{n\times k}$
Initialize B_k with $B_1 \in \mathbb{R}^n$ or $B_{n-1} \in \mathbb{R}^{n\times n-1}$
Evaluate $L_k^+ = (B_k B_k^T)^{-1}$
while $i \leq k$ **do**
  for $b_j \neq b_1, ..., b_{i-1}$ **do**
    Evaluate $b_j^T L_k^+ b_j$
  end for
  $b =$argmax$(b_j^T L_k^+ b_j)$
  Update L^+_{i+1}: $L_{i+1} \leftarrow L_i + bb^T$ ▷ Woodbury formula
  $i = i+1$
end while

The third variant of the algorithm starts with the initial graph and deletes iteratively the least resistant edge. It has the same complexity as the first greedy selection version. It could be faster if the size k of the sparsifier is closer to m than n, but requires the initial evaluation of L_m^+ with a time cost of $O(m\ln^c n)$, where c is a constant. For the sake of clarity, Alg. 6 presents only the first two variants whose pseudo-code is nearly identical.

At each iteration, the algorithm needs to evaluate $b^T L^+ b$ for each candidate columns b, and then update L^+, the pseudo-inverse of a matrix obtained by adding (for variants 1 and 2) or subtracting (for variant 3) a rank-one perturbation. Let us take the case of adding a rank-one perturbation; the Sherman-Morrison-Woodbury (SMW) formula operates this step, but requires the matrix to be non singular. Nevertheless, the formula can be generalized for a Laplacian matrix and gives:

$$\begin{cases} \left(L + bb^T\right)^+ = L^+ - \alpha (L^+ b)(L^+ b)^T \\ \alpha = \left[1 + b^T L^+ b\right]^{-1} \end{cases} \tag{5.18}$$

Proof. (Generalized Sherman–Morrison formula) Let $u_1 \in \mathbb{R}^n$ be the vector with all coordinates equal to 1 in the canonical basis. Let $J = u_1 u_1^T / n$. It is well known (Thm. 2 and [GBS08; Bla+23]) that $L + J$ is invertible and

$$L^+ = (L + J)^{-1} - J. \tag{5.19}$$

Let $\alpha = \left[1 + b^T (L + J)^{-1} b\right]^{-1}$. Applying the Sherman-Morrison formula to $L + J$, we have:

$$(L + J + bb^T)^{-1} = (L + J)^{-1} - \alpha \left[(L + J)^{-1} b\right] \left[(L + J)^{-1} b\right]^T. \tag{5.20}$$

Then using (5.19) and the fact that $Jb = 0$,

$$(L + bb^T)^+ = L^+ + J - J - \alpha (L^+ b)(L^+ b)^T, \tag{5.21}$$

so that

$$\begin{cases} \left(L + bb^T\right)^+ = L^+ - \alpha (L^+ b)(L^+ b)^T \\ \alpha = \left[1 + b^T L^+ b\right]^{-1} \end{cases} \tag{5.22}$$

□

The inner loop evaluating the $b^T L^+ b$ terms should be done at once and in matrix form: if $B_{-i} \in \mathbb{R}^{n \times (m-i+1)}$ is formed of all the columns b from B not yet selected at the iteration i, the diagonal terms $b^T L^+ b$ of $B_{-i}^T L^+ B_{-i}$ require only $O(n)$ operations, by taking advantage of the SMW formula and the sparsity of B. The main loop therefore requires $O(n(k-1))$ or $O(n(k-n+1))$ operations and the overall complexity of the algorithm is quadratic in time.

5.4.2 Approximate greedy algorithm: GSMVCORR

This algorithm selects the column most correlated to the eigenspace of the smallest nonzero eigenvalue.

As with the previous algorithm, three variants are proposed. The first two variants start from an empty graph and add edges during two steps, differing only in the first step. The third variant deletes edges iteratively from the entire initial graph.

In the first variant a maximum spanning tree provides a graph of maximum volume with $n-1$ edges, for which the first eigenpair (u_2, λ_2) must to be evaluated (first step). Then $k-n+1$ edges are added iteratively (second step).

In the second variant, the algorithm starts with the highest resistant edge and evaluates $L_1 = bb^T$. The eigenpair is clearly $(b, 2)$ (first step), after which $k-1$ edges are added iteratively (second step).

Complexity: in the initialization step of the first variant, the eigenpair (u_2, λ_2) is evaluated using the Lanczos algorithm, which requires only matrix-vector products of the form $Ly = B(B^T y)$, fully exploiting the sparsity of

Algorithm 7 GSMVCORR

Require: $B \in \mathbb{R}^{n\times m}, k \in [\![n,m]\!]$
Ensure: $B_k \in \mathbb{R}^{n\times k}$
Initialize B_k with B_1 or $B_k \in \mathbb{R}^{n\times n-1}$
Initialize eigenpair (u_2,λ_2) ▷ $(b,2)$ or Lanczos algorithm
while $i \leq k$ **do**
  for $b_j \neq b_1,...,b_{i-1}$ **do**
    Evaluate $(b_j^T u_2)^2$
  end for
  b =argmax$(b_j^T u_2)^2$
  Update $L_{i+1} \leftarrow L_i + bb^T$
  Update (u_2,λ_2) ▷ Mitz or Lanczos algorithm
  $i = i+1$
end while

B [Dem97; Saa11]. The complexity of this step is thus $O(n)$ operations. For the second variant, the eigenpair is directly given by $(b,2)$ and requires only $O(1)$ operations.

Within the main loop, at each iteration i (i runs from 2 to k or n to k depending on the variant), the evaluation of the $k-i+1$ scalar products $b_j^T u_2$ costs $O(k-i+1)$ operations, thanks to the sparsity of b_k. A rank-one update of the eigenpair (λ_2,u_2) is needed. Mitz and al. proposed an algorithm requiring $O(n)$ arithmetic operations, by working on a partial sum of the secular equation [MSS19]. It is also possible to use again the Lanczos algorithm for the same complexity. The overall complexity of the algorithm is therefore $O(k(n+k))$.

5.5 Simulations and performance

In this section, we propose several simulations on generic and real-world graphs to visualize the effects of the sparsification process on connectivity and the spectra of the underlying Laplacians.

Figure 5.1 shows a random geometric graph with n nodes. The nodes are random points sampled uniformly on the unit square and are connected if, and only if, their distance is less than a paramater p. In Fig. 5.1 the graph has parameters $n = 100$, $p = 0.2$ and $m = 755$. The initial graph G and three stages of sparsification by GSMVDIV algorithm are shown for different numbers of edges. As the algorithm maximizes the number of spanning trees and every graph contains at least one, when the sparsification is run for $m = n-1$ edges, it always produces a spanning tree. For the sake of comparison, we also consider the complete graph. Figure 5.2 shows this graph and three stages of sparsification by GSMVDIV.

The two lines in Fig. 5.3 compare the connectivity of the complete and a random geometric graphs and subgraphs (similar to the one of Fig. 5.3), sparsified using GSMVDIV, GSMVCORR, or at random, as the number of deleted edges grows from 1 to $m-n+1$. When the graph is sparsified by deleting random edges, it may split into two connected components, reducing the number of spanning trees to zero; this explains the vertical blue lines in the figures. The sparsification using GSMVDIV algorithm iteratively maximizes the number of spanning trees, while maintaining the graph connected, so the corresponding curve cannot drop to zero except when $k = n-1$.

The two lines in Fig. 5.4 illustrate the effect of the sparsifying algorithm on the evolution of the Laplacian spectrum, as a function of the number k of edges. When sparsification is performed randomly, the width of the

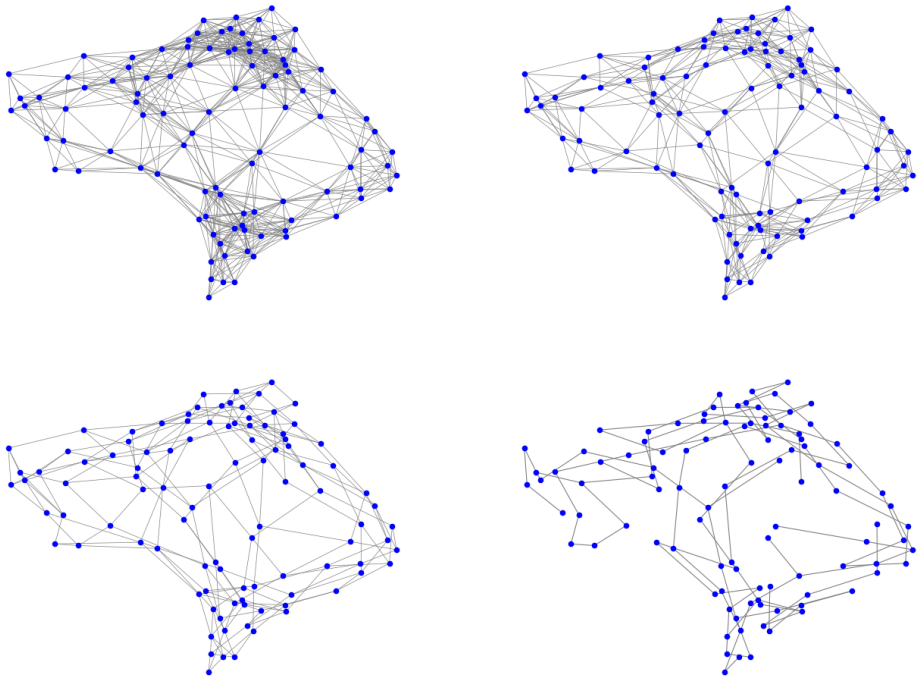

Figure 5.1 – A random geometric graph with $n = 100$, $m = 755$ and three subgraphs sparsified by GSMVDIV: $k = 390$, $k = 195$ and a final spanning tree with $k = 99$.

spectrum does not change as the number of edges increases, and no effect is observed on the smallest eigenvalues. With GSMVDIV or GSMVCORR, the eigenvalues (except 0) increase with the number of edges but the reduction in spectrum width is clearly less pronounced than in Fig. 4.6, where the algorithms are applied to a Gaussian matrix. The last column corresponding to $k = m$ shows the spectrum of the initial incidence matrix: only the two eigenvalues 0 and m for the complete graph, and values ranging from 0 to 30 for the random geometric graph. $\lambda_1 = 0$ is always an eigenvalue and corresponds to the horizontal line coinciding with the x-axis. λ_2, the algebraic connectivity, is the smallest nonzero eigenvalue. It increases steadily with k in the sparsified (with GSMVDIV) complete graph, reaching the maximum eigenvalue of 20 when $k = 175$. For the geometric graph, since the algebraic connectivity of the initial graph is low, the increase of λ_2 in the sparsified graph is less pronounced. However, the curve corresponding to λ_2 quickly becomes nearly horizontal, indicating that the sparsified graph maintains an algebraic connectivity close to the maximum, even with a relatively low number of edges. No such phenomenon is observed with random sparsification, where the smallest eigenvalues remain close to zero until the maximum k is reached.

Fig. 5.5 compares the 50 smallest eigenvalues of the Laplacian of sparsified subgraphs of the Cora Planetoid dataset, when sparsification is performed using GSMVDIV (left) or random edge deletion (right), as a function of the number of edges in the sparsified graph. The ideal outcome for the sparsified graphs is a plot with nearly horizontal lines, indicating that the eigenvalues are not decreasing. This is observed in the left figure, with GSMVDIV. In contrast, when sparsified randomly, the eigenvalues drop quickly, degrading the connectivity.

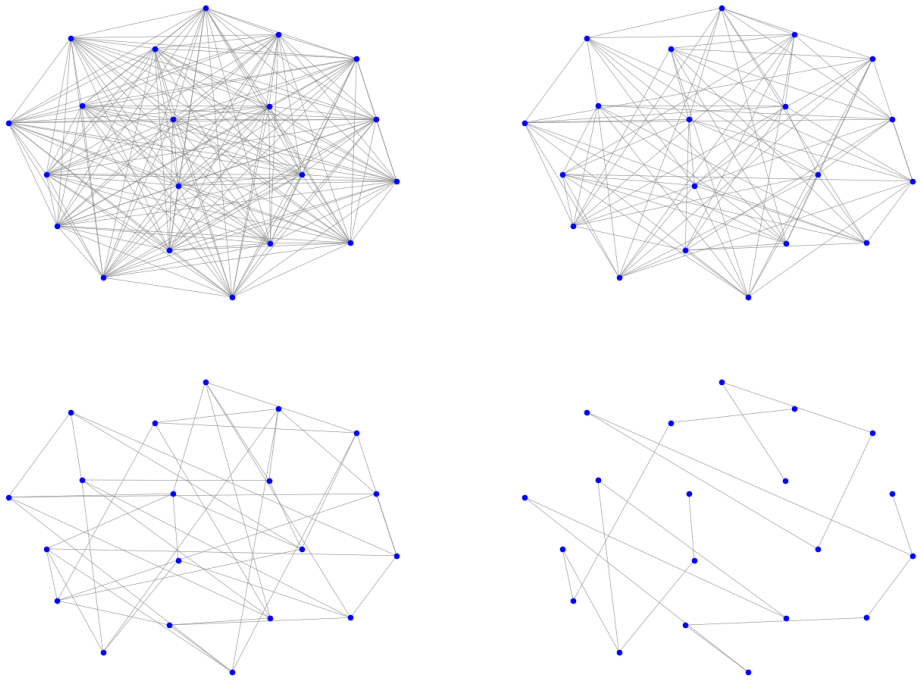

Figure 5.2 – The complete graph with $n = 20$, $m = 190$ and three subgraphs sparsified by GSMVDIV: $k = 95$, $k = 47$ and a final spanning tree $k = 19$.

5.6 Application to Graph Neural Networks

5.6.1 A brief introduction to GNN

A Graph Neural Network (GNN) is a class of neural network and geometric deep learning model designed for graph-structured data [Sca+09; KW17; Vel+17; HYL17; Wu+22]. GNNs achieve state-of-the-art performance in many graph-related tasks, such as node or graph classification, link prediction, graph clustering, semi-supervised learning and dimensionality reduction.

GNNs use a message-passing algorithm as layer-wise propagation rule: every node receives information only from its direct neighbors, and iteratively updates its node feature vectors, in a manner reminiscent of Pearl's belief propagation algorithm [Pea82]. By aggregating information from the local neighborhood, GNNs integrate node feature data, edge weights and the graph topology into the learning process.

A layer in a GNN corresponds, for a given node, to its one-hop neighborhood. Stacking the layers therefore results in the iterative updating of the message-passing process: a l-layer GNN performs l iterations on the same underlying graph, where the l-th iteration brings information to any node from its l-hop neighborhood (cf. Fig. 5.6). The necessity of l iterations for two nodes at distance l to communicate, is referred to as the problem radius [AY20].

The mathematical formulation of a GNN is defined by its underlying graph $G = (V, E)$, the node features matrix $X \in \mathbb{R}^{n \times d}$, where $x_v \in \mathbb{R}^d$ is the row of X corresponding to the node $v \in V$, and the recurrent propagation

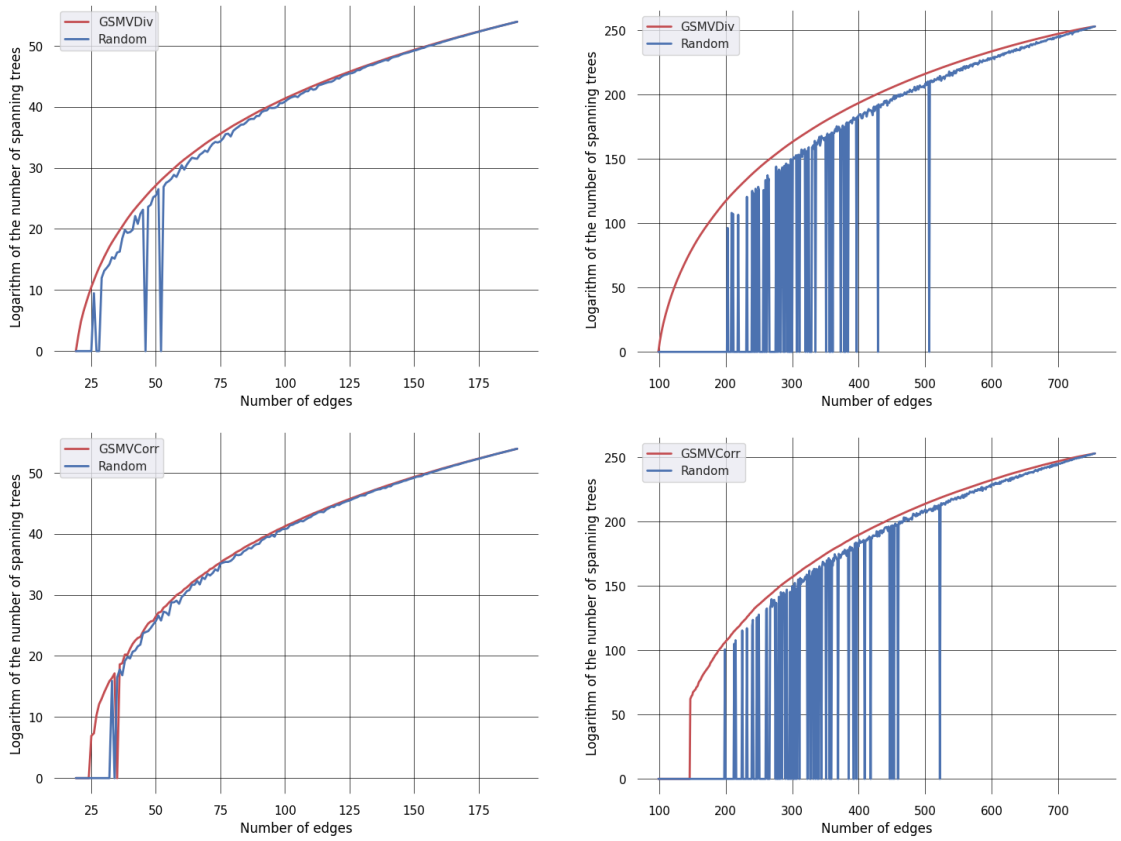

Figure 5.3 – Connectivity with GSMVDIV and GSMVCORR versus random. Top: connectivity (measured as the logarithm of the number of spanning trees) of sparsified subgraphs using the GSMVDIV algorithm, as a function of the number of edges, for two example graphs (red curve); left: the complete graph with $n = 20$, $m = 190$; right: a random geometric graph with $n = 100$ and $m = 755$. The connectivity of the extracted subgraphs (in red) is compared with that of sparsified subgraphs whose edges are selected at random (in blue). Bottom: same as the previous line, but with GSMVCORR algorithm. Note that, for each value of k, a new random subgraph is generated, which explains why a zero value in the blue curves can be followed by a higher connectivity value.

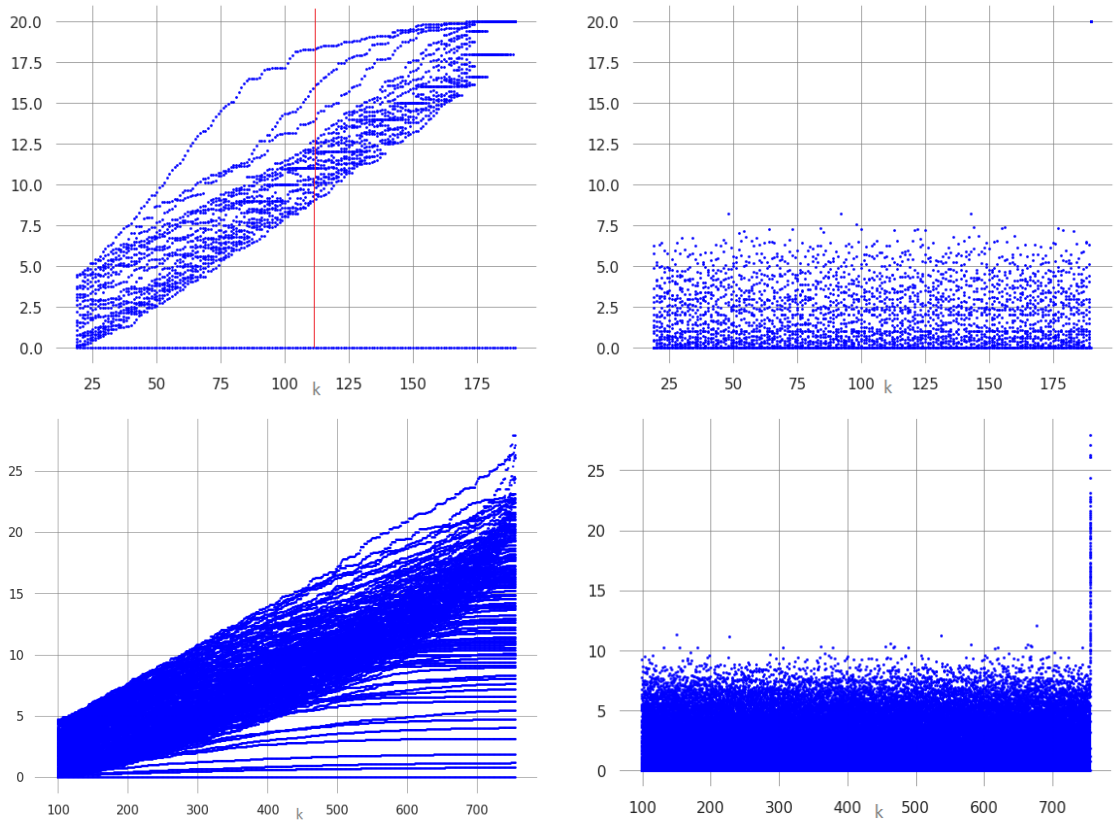

Figure 5.4 – Connectivity and spectrum: GSMVDIV versus random sparsification. Top: the spectrum of sparsified subgraphs from the complete graph with parameters $n = 20$, $m = 190$, as a function of the number of edges k. Each vertical line at position k on the x-axis represents the set of eigenvalues from the sparsified subgraph with k edges. Left: the sparsification is performed with GSMVDIV; right: the sparsification is performed randomly. Bottom: same as above, but with the initial graph being a random geometric graph with parameters $n = 100$ and $m = 755$.

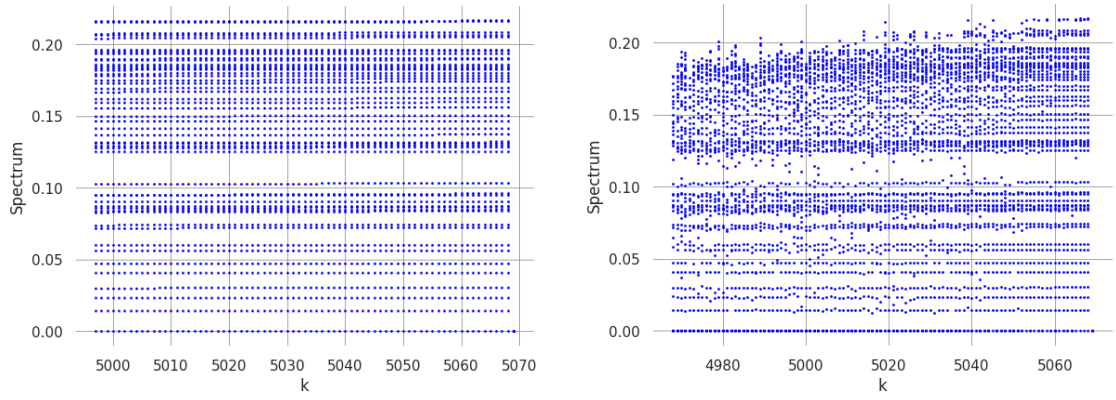

Figure 5.5 – The 50 smallest eigenvalues of the largest connected component of Cora Planetoid graph, when sparsified by GSMVDIV (left) or at random (right), as a function of the number of edges in the sparsified graph.

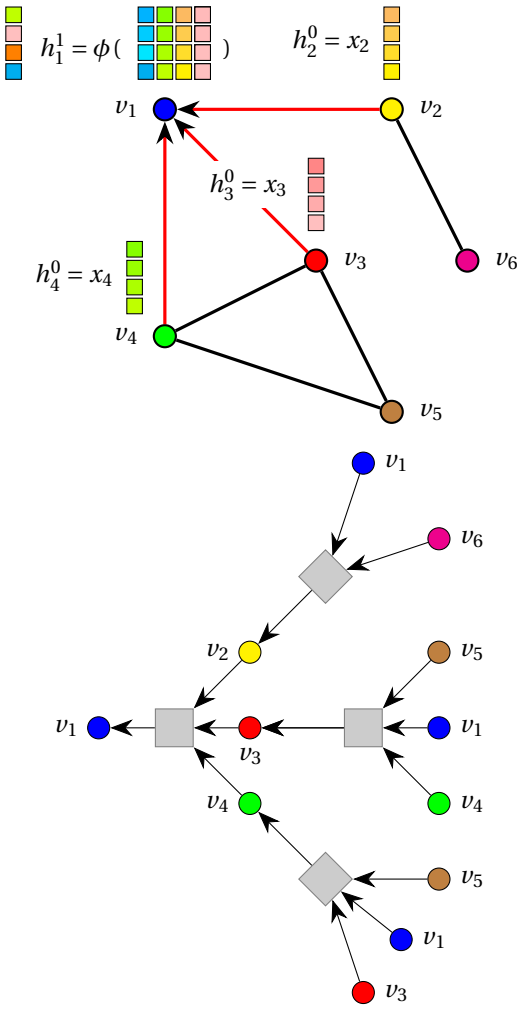

Figure 5.6 – Structure of a typical GCN. Above: the target node is v_1, we begin just before the first iteration. The 1-hop neighborhood of v_1 (consisting in v_2, v_3, v_4) sends their initial node features vector ($h_2^0 = x_2$, $h_3^0 = x_3$, $h_4^0 = x_4$) to v_1 during the first iteration, where these vectors are aggregated with $h_1^0 = x_1$ using the ϕ function. The result is the updated feature vector h_1^1, which replaces x_1 and will be sent to all the neighbors of v_1 in the iteration 2. Below: the tree of the 2-hop neighborhood of v_1. The first layer aggregates messages from the 1-hop neighborhood, corresponding to the operation described in the left diagram, where the features vector of v_2, v_3, v_4 are sent to and aggregated at v_1. The same process is performed on all nodes, particularly for v_2, v_3, v_4. The second layer aggregates messages from the 2-hop neighborhood of v_1, corresponding to the nodes at distance 2 from v_1. The messages updated in these nodes reach the initial node in the second iteration. Gray squares represent the aggregation process modeled by the functions ϕ_l. The nodes of the underlying graph coincide with the nodes of the neural network: 1 GNN layer = 1 iteration of the message passing algorithm.

rule:

$$\begin{cases} h_v^0 = x_v, \\ h_v^{l+1} = \phi_l\left(h_v^l, \sum\limits_{u \sim v} \hat{A}_{uv} \psi_l\left(h_u^l\right)\right). \end{cases} \tag{5.23}$$

For each node v, $h_v^l \in \mathbb{R}^d$ is the updated feature vector of v at iteration l, ϕ_l and ψ_l are differentiable, learnable functions and $\hat{A} = A + I$ is the modified adjacency matrix of G, which includes self loops. Self loops allow a node to incorporate its own current information along with the information from its neighbors and are known to prevent the phenomenon of over-smoothing described below.

For the sake of simplicity, we present our results on a typical Graph Convolutional Network (GCN) [KW17] whose simple formal expression is given in matrix form:

$$H_{l+1} = \sigma\left(D^{-1/2}\hat{A}D^{-1/2}H_l W_l\right), \tag{5.24}$$

with $H_l = \left(h_v^l\right)_v \in \mathbb{R}^{n\times d}$, $D = \mathrm{diag}(d_{ii})$, $d_{ii} = \sum_j \hat{A}_{ij}$, W_l is the trainable weight matrix. $\sigma = \phi_l$ is an activation function (ReLU, sigmoid, etc.) and $\psi_l = \mathrm{Id}$.

Over-smoothing, over-squashing and under-reaching are three issues that impair the performance of a GNN when using message passing algorithms. Over-smoothing occurs when node features quickly converge toward a common average and become indistinguishable [CW20]. Under-reaching happens when the network is not deep enough to convey information from distant nodes due to the problem of radius [Bar+20]. To prevent over-smoothing, the depth of the network should not be too great, while the opposite is necessary to prevent under-reaching. Although these two phenomena are now well understood, less is known about over-squashing, which measures the difficulty of propagating information between distant nodes, often due to bottlenecks between nodes in certain parts of the graph [Top+22]. While under-reaching is solely due to the neighborhood radius, over-squashing is linked to the topology and connectivity of the entire graph and seems to occur mainly in tasks that depend on long-range interactions [AY20].

To prevent over-squashing, the most common technique is local rewiring, which involves adding, removing or changing weights of edges selected to optimize certain properties related to the graph topology, thereby reducing bottlenecks. The property to optimize can be graph curvature [Top+22], algebraic connectivity [KBM22], commute time distance [Di +23], effective resistance [Ban+22; Arn+22; Bla+23] or the use of specific graphs such as expanders [DLV22] or complete graphs [AY20].

5.6.2 Application to GNNs: learning on the sparsified graph

As an application of our sparsification methods, we propose to learn on the sparsified graph of a GNN before any training or learning process. The objective is to reduce the computational resources required to store the graph and to speed up the training and processing of the data, while preserving connectivity. We aim to study the extent to which performance is affected by the sparsification process and how connectivity optimization limits this effect and the occurrence of over-squashing.

Our approach differs from previous work in several key aspects:

- We do not improve the connectivity of the graph by locally adding edges, but at the contrary we aim to preserve it in a simplified version of the graph.
- We adopt a global strategy, optimal within greedy methods, to preserve the graph's connectivity while significantly simplifying it by deleting a substantial number of edges (typically up to 30% or 50%).
- Our method is completely independent of the GNN architecture, which does not need to be modified; we simply replace the initial graph with its sparsifier before any operation are performed.

Our application benefits from the low complexity of our algorithms: to the best of our knowledge, a quadratic complexity in the number of operations is the lowest existing complexity for a deterministic sparsification algorithm optimizing connectivity in GNN. The greedy total resistance (GTR) algorithm [Bla+23] has a complexity of $O(n^3)$ and the first-order spectral rewiring (FoSR) algoritm [KBM22] has a complexity of $O(kn^2)$ operations; the stochastic discrete Ricci flow (SDRF) [Top+22] and the Random local edge flip (RLEF) algorithms [Ban+22] also have higher complexity. [Bla+23] proposed an implementation in $O(nm\ln n)$ operations, similar to the Spielman-Teng random algorithm, making it suitable only for weighted graphs.

5.6.3 Experiments

The following simulations demonstrate the performance of several GCNs on classic graph datasets run on a node classification task: the three Planetoid citation networks Citeseer, Pubmed and Cora, whose size is given in Table. 5.1. The nodes represent documents with bag-of-words feature vectors, and the edges are citation links between research papers. The different classes correspond to research fields and the model is trained to predict missing labels.

Table 5.1 – Citation network datasets.

Name	#nodes	#edges	#features	#classes
Cora	2708	5278	1433	7
CiteSeer	3327	4552	3703	6
PubMed	19717	44324	500	3

Given a target number k of edges ($n \leq k \leq m$), for each dataset, a GCN architecture is built around five different graphs: the original dataset's underlying graph and four other graphs sparsified to k edges. The sparsified graphs include two generated by our algorithms GSMVDIV and GSMVCORR, one with edges sampled uniformly at random and one sparsified using the Spielman-Teng algorithm [ST11]. All GCNs are trained and run on a node classification task. The accuracy curves for training and testing data are then assessed for different values of the parameters, such as the number and dimension of the hidden layers.

Table 5.2 – Models of the simulations.

Name	Underlying graph
Gi	Base: initial graph
GSMVDIV	GSMVDIV
GSMVCORR	GSMVCORR
Spielman-Teng [SS11]	Spielman-Teng
Random	Random

Name of the different models of GNN and their underlying graph.

Table 5.3 – Performance for the Cora dataset.

#layers	dim HL	Gi	GSMVCORR	Spielman-Teng	Random
4	64	82,1±0,5	81,2±0,6	80,2±0,4	78,5±0,5
	16	80,0±0,9	79,5±0,6	77,1±0,7	76,4±0,9
	8	74,2±2,1	74,4±0,8	71,3±2,0	71,9±1,5
8	64	82,9±0,5	80,6±0,4	78,5±0,4	77,6±0,3
	16	78,6±0,8	78,5±0,6	74,4±0,8	73,5±0,7
	8	77,6±1,0	75,9±2,0	72,7±1,5	73,1±1,5
12	64	82,6±0,8	80,6±0,4	79,5±0,5	77,8±0,6
	16	80,7±1,2	79,3±0,8	78,7±1,0	75,4±1,5
	8	76,1±2,1	75,9±1,7	73,9±1,9	69,1±2,4
24	64	82,5±0,8	78,7±2,4	74,7±3,5	71,7±3,3
	16	60,3±6,5	58,4±4,7	47,3±8,1	44,8±8,1
	8	48,9±8,1	48,6±6,4	40,4±6,2	39,7±7,5

Accuary of the test set in node classification task using the (largest connected composant of) Cora dataset, for a GCN model with 4 different underlying graphs, as a function of the number and dimension of hidden layers. The best performance among the 3 sparsified graphs is highlighted in red. Each accuracy value represents the average of 10 independent samples, with the standard deviation indicated as ±.

Regarding Cora and Citeseer datasets and in most cases, PubMed, the model GSMVCORR outperforms the Spielman-Teng algorithm and always outperforms the randomly sparsified graph. The performance of GSMVCORR remains close to that of the initial model Gi, thus demonstrating that the sparsification does not significantly affect the classification task.

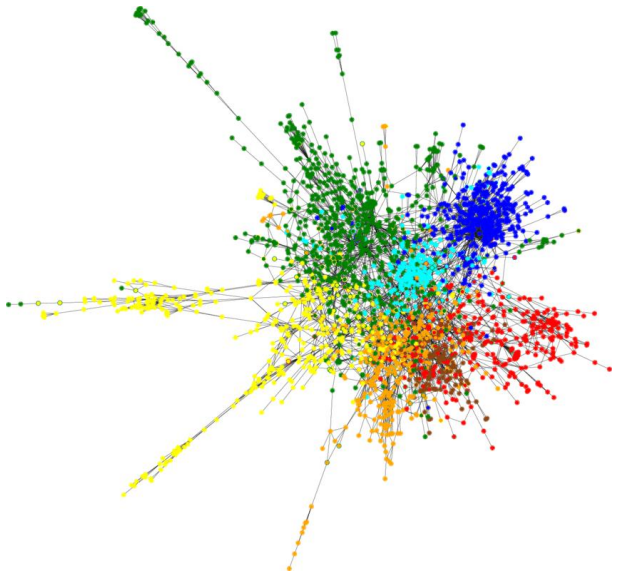

Figure 5.7 – Biggest connected component of Cora dataset. Nodes color correspond to the different features value.

Table 5.4 – Performance for the CiteSeer dataset.

#layers	dim HL	Gi	GSMVCORR	Spielman-Teng	Random
4	64	74,6±0,5	74,6±0,4	73,1±0,5	72,5±0,6
	16	75,5±0,7	75,9±0,3	74,7±0,7	73,4±0,5
	8	75,9±1,2	75,8±1,1	75,1±0,4	74,4±0,6
8	64	73,9±0,5	73,3±0,2	72,9±0,5	71,8±0,4
	16	74,3±0,7	74,3±0,5	73,2±0,6	72,5±0,6
	8	73,4±0,9	74,1±1,1	72,9±0,7	72,1±0,6
12	64	73,5±0,7	73,2±0,4	73,5±0,5	72,5±0,6
	16	73,3±1,0	73,4±0,8	72,7±0,8	72,4±0,8
	8	71,4±1,3	71,2±0,8	71,1±1,1	69,5±1,0
24	64	72,9±1,0	72,4±0,8	72,1±0,7	69,9±1,7
	16	48,8±1,2	46,5±7,0	47,7±8,0	38,0±8,7
	8	36,9±9,3	32,3±3,7	29,9±6,3	28,3±3,4

Accuary of the test set in node classification task using the (largest connected composant of) CiteSeer dataset, for a GCN model with 4 different underlying graphs, as a function of the number and dimension of hidden layers. The best performance among the 3 sparsified graphs is highlighted in red. Each accuracy value represents the average of 10 independent samples, with the standard deviation indicated as ±.

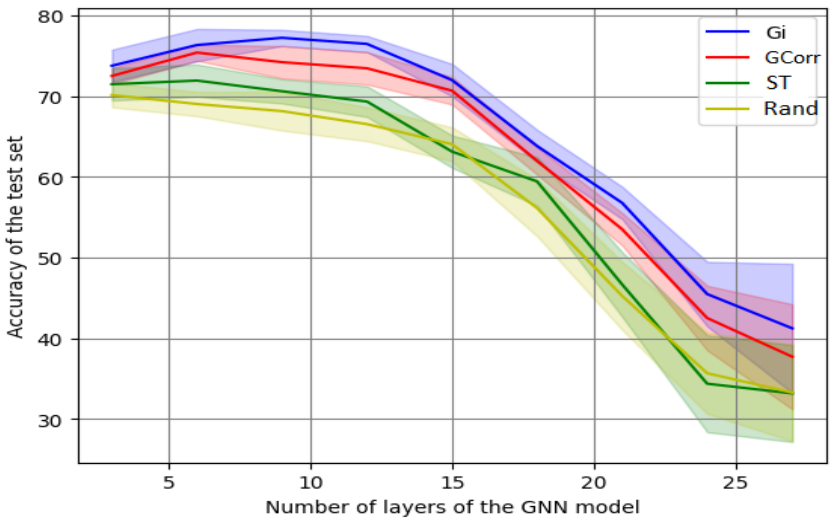

Figure 5.8 – Accuary of the test set in a node classification task, using the (largest connected component of the) Cora dataset, with a GCN architecture, for different underlying graphs, as a function of the number of layers. The initial graph has $n = 2485$ nodes and 5069 edges. Blue: Gi, red: GSMVCORR (GCorr), green: Spielman-Teng (ST), yellow: Random. The shaded confidence interval represents ± standard deviation over 10 independent samples. Each hidden layer has a dimension of only 8 and the number of edges in the sparsified graphs is 2500 (50% of the initial graph). Note that the sparsified graph is nearly a spanning tree.

Tables 5.3, 5.4 and 5.5 present the performance metrics for respectively the Cora, CiteSeer and PubMed

Table 5.5 – Performance for the PubMed dataset.

#layers	dim HL	Gi	GSMVCORR	Spielman-Teng	Random
4	64	88,9±0,2	88,7±0,1	88,3±0,3	88,1±0,1
	16	88,5±0,1	88,4±0,2	88,3±0,1	87,7±0,2
	8	87,3±0,1	87,7±0,3	87,4±0,1	86,7±0,2
8	64	88,4±0,1	88,3±0,3	87,8±0,6	87,0±0,2
	16	88,0±0,3	87,5±0,1	87,7±0,1	87,3±0,2
	8	87,5±0,1	86,7±0,5	87,3±0,2	86,9±0,3
12	64	88,2±0,3	88,0±0,2	87,8±0,1	86,8±0,2
	16	87,6±0,1	87,4±0,1	87,4±0,2	86,8±0,1
	8	86,9±0,6	86,8±0,7	87,0±0,1	86,8±0,4
24	64	86,2±0,3	85,0±0,5	85,0±0,5	83,9±0,1
	16	83,1±0,8	82,9±1,0	82,6±2,0	77,6±0,8
	8	69,1±3,9	73,5±4,0	59,7±9,8	56,8±6,5

Accuary of test set in the node classification task using the (biggest connected composant of) PubMed dataset, for a GCN model with 4 different underlying graphs, as a function of the number and dimension of hidden layers. The best performance among the 3 sparsified graphs is highlighted in red. Each accuracy value represents the average of 10 independent samples, with the standard deviation indicated as ±.

datasets. The difference in performance between Gi and GSMVCORR remains small, regardless of the number and the dimension of the layers, even for a significant percentage of edges deleted. As the dimension of the hidden layers decreases and the number of layers increases, the performance gap between the group Gi, GSMVCORR and the two models Spielman-Teng and Random widens. Over-squashing, known to appear with a high number of layers of small dimension, might be mitigated by the maximization of connectivity, potentially explaining the strong performance of our algorithms. Figure 5.8 illustrates these points for the Cora dataset by showing the accuracy curves for all graphs as a function of the number of layers.

The performance improvements are significant only for homophilic graphs, where the neighbors of a node provide relevant information about its state. In the case of heterophilic graphs, the topology does not contribute as effectively.

Our algorithms have better performance for almost all parameter configurations, along with lower complexity. The connectivity is thus a crucial property to preserve and optimizing connectivity proves to be an effective solution to prevent over-squashing phenomenon.

Datasets configuration

For each dataset, the training set is extended to half of the data, the other half being the test set (using a random state parameter of 42). If the graph is not connected, the largest connected component is selected and the simulations are run on this subgraph.

The GNN architecture is a classical GCN as defined in [KW17], with some minor modifications: a linear classifier is defined as the first layer, a sequence of GCNConv forms the hidden layers, which end with another linear classifier. The output of each layer, after applying the non-linear activation function, is added with to

the input vector (or the residual after the first GCNConv layer) to serve as a residual connection preventing over-smoothing.

Some parameters are fixed throughout all simulations and their values are provided in Tab. 5.6.

Table 5.6 – Fixed parameters.

Parameter	Value
Learning rate	5×10^{-3}
Weight decay	5×10^{-4}
Dropout	0,5
Epochs	200

The dimension of each hidden layer varies from 8 to 64, and the number of layers ranges from 3 to 27.

5.7 Conclusion

In this work, we proposed two greedy algorithms to sparsify a graph while preserving its connectivity. These algorithms emerge from a geometric interpretation of the volume of the graph Laplacian matrix and modify the sprectum of the initial graph to optimize its robustness.

Both algorithms are deterministic, adapted to unweighted graphs and of reasonable quadratic complexity in time. We provided a detailed implementation of several variants and empirically demonstrated that they perform better than state of the art sparsification algorithms.

We proposed an application to GNNs that simplifies the architecture and accelerates the processes executed on the network, without significantly degrading performance. Our method seems to be an effective way to limit the over-squashing phenomenon, but only for homophilic graphs where the neighbors of a node bring meaningful information.

CHAPTER 6

ASYMPTOTIC ANALYSIS IN COMPRESSIVE SENSING

6.1 Introduction and mathematical framework

In this chapter, we lay the foundations for a statistical analysis of some sparsity pattern recovery algorithms, including the Orthogonal Matching Pursuit (OMP) algorithm. Our goal is to analyze the performance of these algorithms within an information-theoretic framework, where both the signal to be recovered and the observation matrix are random, and the sizes of the signal and the matrix tend to infinity.

As discussed in Ch. 3, the two traditional frameworks for performance analysis are the uniform or non-uniform recovery [FR13, p. 48; p. 281]. In uniform recovery, the measurement scheme enables the recovery of all k-sparse vectors for a given sensing matrix (random or not). In non-uniform recovery, for a given sparse vector x, there must exists a sensing matrix (depending on x) capable of recovering the sparsity pattern of x.

Non-uniform recovery requires x to be deterministic. Our framework differs by providing an average-case analysis, allowing both x and ϕ to be random.

The first model considers fixed d, m, k. The sensing matrix ϕ has random i.i.d Gaussian entries and the initial vector x have nonzero random i.i.d Gaussian entries. x and ϕ are independent. The second model is asymptotic, where d, m, k tend to infinity with the following linear constraints

$$k = \gamma d \text{ and } m = \mu k \tag{6.1}$$

$$\gamma \in]0,1[\text{ and } \mu > 1. \tag{6.2}$$

These constraints define the large system regime (LSR). Here, γ is the sparsity rate and μ is the overmeasure rate. Both models provide an average statistical analysis, differing from the traditional uniform and non-uniform recovery frameworks.

We derive the joint densities of all statistics involved in the recovery process. This allows us to establish an asymptotic lower bound on the probability of recovery at a given iteration and prove the convergence toward 1 of the asymptotic probability of recovering a correct atom from the initial signal, at that specific iteration, under the sole condition of linear sparsity and overmeasure rate.

The objective was to determine the probability distribution of the scalar products involved in sparse recovery methods and to use these distributions to evaluate the probability of success of some of these methods, either at finite lengths or in the LSR. As we shall see, this is possible for some certain simple one-shot methods or for a given iteration of greedy algorithms. Although we do not succeed in analyzing the entire OMP process

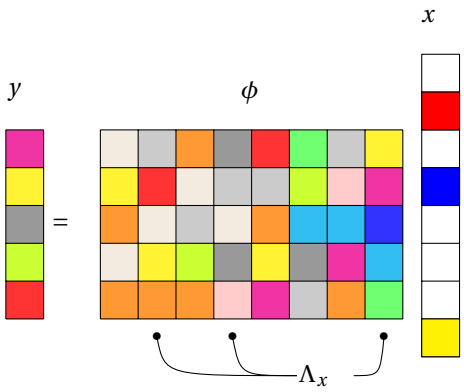

Figure 6.1 – Illustration of $y = \phi x$. The support Λ_x of x is sparse, with a cardinality $k \ll d$. x is unknown, y is the observed vector whose coordinates are linear combinations of those of x.

due to stochastic dependence issues, our theoretical results and our simulations yields interesting insights and we propose several directions for future research. The tools developed to address our problem could serve as a toolbox for future advancements.

Throughout the chapter, the following assumptions are made:

The unknown signal is modeled as a d-dimensional real random vector x. x is sparse with k i.i.d. nonzero entries, Gaussian, with mean 0 and variance σ_x^2: $x_\Lambda \sim \mathcal{N}\left(0, \sigma_x^2\right)$, where $\Lambda = \text{supp}(x)$ is the support of x, representing the set of indices corresponding to the nonzero entries. Due to the symmetric construction of the random observation matrix (the columns are exchangeable), we further assume, without loss of generality, that $\Lambda = [\![1, k]\!]$. Sometimes, we index Λ by the dimension of x and write Λ_d, particularly when considering the limit as $d \to +\infty$.

The observed vector y is formed of m linear combinations of x:

$$y = \phi x = \sum_{j \in \Lambda} \phi_j x_j, \tag{6.3}$$

where ϕ is a $m \times d$ random matrix with i.i.d. Gaussian entries $\mathcal{N}\left(0, \sigma^2/m\right)$. Here, ϕ_j denotes the j-th column of matrix ϕ and ϕ_{ij} denotes the i-th coordinate of vector ϕ_j. If S is a subset of $[\![1, d]\!]$, then ϕ_S is the matrix formed by the columns of ϕ indexed by S. The matrix ϕ and the signal x are stochastically independent.

The observation matrix ϕ is commonly referred to as the sensing or measurement matrix in compressive sensing, and as the dictionary in sparse approximation literature, where a column of ϕ is called an atom. The normalization factor $1/m$ is necessary to prevent variance from diverging and to ensure a finite limit.

Let's summarize the two models:

Definition 5 (Finite-length statistical model).

- *Size parameters:* $k, m, d \in \mathbb{N}^*$, $k \ll m \ll d$.
- *Observed vector:* $y = \phi x$.
- *Sensing matrix:* $\phi = \left(\phi_{ij}\right) \in \mathbb{R}^{m \times d}$, ϕ_{ij} *i.i.d with* $\phi_{ij} \sim \mathcal{N}(0, \sigma^2/m)$.
- *Unknown sparse vector:* $x \in \mathbb{R}^d$, $\forall x_i \in \Lambda$, x_i *i.i.d with* $x_i \sim \mathcal{N}(0, \sigma_x^2)$.

- *Independence assumption:* $\phi \perp\!\!\!\perp x$.

Definition 6 (Asymptotic statistical model).

- *The assumptions from Def. 5 apply.*
- *Large system regime (LSR):* $d \to +\infty$, $k/d = \gamma \in]0,1[$ *and* $m/k = \mu > 1$.

The chapter is organized as follows: in section 2 we present technical lemmas concerning projections and products of Gaussian distributions. These lemmas allow us to derive the probability distributions of the statistics involved in sparse recovery methods, both for finite lengths and in the large system regime. In section 3, we derive an asymptotic equivalence for a Gaussian integral representing the probability of an event involving the comparison of the maximum of two Gaussian vectors. Section 4 connects the results from the previous two sections to evaluate the probability of success at a given iteration of the OMP algorithm. Eventually, in section 5, we discuss the challenges that prevented us from analyzing all iterations of the OMP algorithm collectively and, as a provisional conclusion, suggest some avenues for future research.

6.2 Distribution of the statistics involved in some sparse recovery methods

To derive the asymptotic distribution of the scalar products, we first obtain the marginal of the scalar products at finite length, then derive their limit as $d \to +\infty$, in the LSR. The results rely on three lemmas, which we present and prove first. In addition to their application in the OMP algorithm, this section can be considered as a probabilistic toolbox useful for future investigations. Additional reminders concerning the relevant probability distributions are provided in Appendix B.1.

6.2.1 Three lemmas about products and projections of Gaussians

The first lemma is a well-known result. One possible proof is given in Muirhead [Mui82], as a consequence of theorem 3.2.5, when $\Sigma = I$. Here, we provide an original demonstration using the characteric function which both identifies the distribution and proves the independence. The two next lemmas are, to the best of our knowledge, unpublished results.

Lemma 4 (Dot product of an i.i.d. Gaussian vector and a random vector uniformly distributed on the sphere). *Let* $X \sim \mathcal{N}(0, \sigma^2 I_d)$ *and* Y *be a random vector of size* d*, independent from* X *and almost surely different from zero. Let* $U = Y/||Y||$ *and* $Z = \langle X, U \rangle$*. Then* $Z \sim \mathcal{N}(0, \sigma^2)$ *and is independent of* Y.

Proof. Let $\phi(t)$ the characteristic function of Z.

$$\phi(t) = \mathbb{E}\left[e^{it\langle X,U\rangle}\right] = \mathbb{E}\left[\mathbb{E}\left[e^{it\langle X,Y\rangle/||Y||} \,|\, Y\right]\right] \tag{6.4}$$

$$\phi(t) = \mathbb{E}\left[\mathbb{E}\left[e^{it\sum_{j=1}^{d} X_j Y_j/||Y||} \,|\, Y\right]\right] \tag{6.5}$$

Given Y, $\langle X, Y\rangle / ||Y||$ is a linear combination of independent, centered Gaussian random variables X_j. Thus, it is a centered Gaussian random variable with variance

$$\mathbb{V}\left(\sum_{j=1}^{d} X_j Y_j / ||Y||\right) = \frac{\sigma^2}{||Y||^2} \sum_{j=1}^{d} Y_j^2 = \sigma^2 \tag{6.6}$$

This variance no longer depends on Y, so that

$$\phi(t) = \mathbb{E}\left[e^{-\frac{\sigma^2}{2}t^2}\right] = e^{-\frac{\sigma^2}{2}t^2} \tag{6.7}$$

To prove independence, we now evaluate the characteristic function of (Z, Y)

$$\phi_{(Z,Y)}(s,t) = \mathbb{E}\left[e^{i(sZ+tY)}\right] = \mathbb{E}\left[\mathbb{E}\left[e^{i(sZ+tY)}|Y\right]\right] \tag{6.8}$$

$$= \mathbb{E}\left[\mathbb{E}\left[e^{isZ}|Y\right]e^{itY}\right] \tag{6.9}$$

$$= e^{-\frac{\sigma^2}{2}t^2}\mathbb{E}\left[e^{itY}\right] = \phi_Z(s) \times \phi_Y(t) \tag{6.10}$$

which proves the independence of Z and Y. □

This result does not hold if the variance of X is not a scalar variance matrix.

Lemma 5 (**Random projection of a Gaussian vector**). *Let $X \sim \mathcal{N}(0, I_d)$ and let P_F be a random orthogonal projector from $\mathbb{R}^d$ onto a (random) subspace F of dimension $r < d$. We suppose X and P_F independent. Let $Z = P_F X$.*

Then there exists an orthonormal basis U s.t. $Y = U^T Z$ is a Gaussian vector whose nonzero coordinates are i.i.d. $\mathcal{N}(0,1)$ and independent from P_F. Moreover, $||P_F X|| \sim \chi_r$ and is independent of P_F.

Proof. Given F, as P_F is an orthogonal projector of rank r, there exists an orthogonal matrix $U = (u_1, ..., u_d)$, such that

$$P_F = UDU^T = (u_1, ..., u_r) \times (u_1, ..., u_r)^T \tag{6.11}$$

where $(u_1, ..., u_r)$ form an orthonormal basis of F, and where

$$D = \begin{pmatrix} I_r & 0 \\ 0 & 0 \end{pmatrix}. \tag{6.12}$$

Now, the vector $Z = U^T X = (Z_1, ..., Z_r, 0, ..., 0)^T$, where the subvector $(Z_1, ..., Z_r) \sim \mathcal{N}(0, I_r)$ and is independent of U. This can be shown by generalizing the proof of Lem. 4 based on the characteristic function (6.4-6.5) to a subspace of dimension $r > 1$. Eventually, as U preserves the Euclidian norm,

$$||P_F X||^2 = ||U^T P_F X||^2 = \sum_{i=1}^{r} Z_i^2 \tag{6.13}$$

follows a χ distribution with r degrees of freedom and is independent of F and U, due to the independence of $(Z_1, ..., Z_r)$ and U.

□

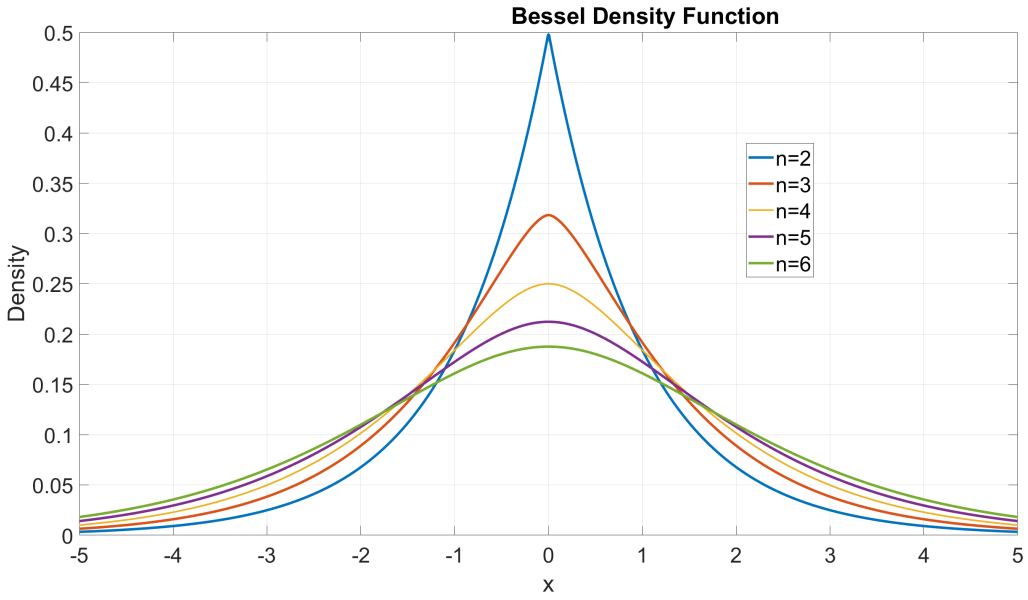

Figure 6.2 – Density function of a standard Bessel distribution for $\sigma = 1$ and $n = 2,3,4,5,6$.

If F is not random, the previous lemma reduces to the Cochran theorem.

The following lemma utilizes the Bessel distribution. Although this distribution is presented in detail in Appendix B.1, we summarize here the essential formulas and properties relevant to the proof of the lemma.

Consider a Gaussian random variable $X \sim \mathcal{N}(0,1)$ and a Gamma random variable $V \sim \Gamma(\alpha,\beta)$ independent of X. The random variable

$$Y = \mu + \theta V + \sigma\sqrt{V}X \tag{6.14}$$

follows a Bessel distribution, denoted as $\mathfrak{B}(\alpha,\beta,\mu,\theta,\sigma)$. Y is a Gaussian variable whose random variance and mean follow a Gamma distribution. In the following lemma, we focus on the standard form $\mathfrak{B}(n,\sigma)$ of the Bessel distribution, characterized by $\theta = 0, \alpha = n/2, \beta = 1/2, \mu = 0$. In this case, the density of $Y = \sqrt{V}X$ simplifies to

$$f_Y(x) = \left(\sigma\sqrt{\pi} \times \Gamma(\frac{n}{2}) \times 2^{\frac{n-1}{2}}\right)^{-1} \left(\frac{|x|}{\sigma}\right)^{\frac{n-1}{2}} K_{\frac{n-1}{2}}\left(\frac{|x|}{\sigma}\right). \tag{6.15}$$

where K_ν is the modified Bessel function of the second kind. The characteristic function of the standard $\mathfrak{B}(n,\sigma)$ Bessel distribution is given by

$$\phi(u) = \left(1+\sigma^2 u^2\right)^{-n/2} \tag{6.16}$$

and the Mellin transform of its absolute value is:

$$\widehat{f}(s) = \frac{1}{\sqrt{\pi}} \times \frac{\sigma^s 2^{s-1}}{\Gamma(n/2)} \Gamma\left(\frac{s}{2}\right) \Gamma\left(\frac{s+n-1}{2}\right). \tag{6.17}$$

A key property of the Bessel distribution is that the product of two independent centered Gaussian random variables $X \sim \mathcal{N}(0, \sigma_X^2)$ and $Y \sim \mathcal{N}(0, \sigma_Y^2)$ follows a $\mathfrak{B}(1, \sigma_X \sigma_Y)$ distribution (Thm. 38). It it therefore easy to deduce that the dot product of two independent centered Gaussian vectors $X \sim \mathcal{N}(0, \sigma_X^2 I_n)$ and $Y \sim \mathcal{N}(0, \sigma_Y^2 I_n)$ follows a $\mathfrak{B}(n/2, \sigma_X \sigma_Y)$ Bessel distribution (Thm. 39). That said,

Lemma 6 (**Product of two χ distributions**). *Let U be random Gaussian variable $\mathcal{N}(0,1)$. Let Y follow a χ distribution of dimension m. Suppose that U and Y are independent.*
Then UY follows a standard Bessel distribution $\mathfrak{B}(m,1)$.

Proof. To derive the distribution of the product of two random variables, we use the fact that the Mellin transform of the product of two random variables is the product of their Mellin transforms B.2. First, note that $|U|$ follows a χ distribution with 1 degree of freedom. Let f and g be the densities of $|U|$ and Y, respectively and let us define the constants

$$\lambda = \frac{2^{1-m/2}}{\Gamma(m/2)} \text{ and } \lambda' = \frac{1}{\sqrt{\pi}} \frac{1}{\Gamma(m/2)} \frac{1}{2^{\frac{m-1}{2}-1}}. \tag{6.18}$$

f and g are given by

$$f(x) = \frac{2}{\sqrt{2\pi}} e^{-x^2/2} \mathbb{1}_{[0,+\infty[}(x) \text{ and } g(x) = \frac{2^{1-m/2}}{\Gamma(m/2)} x^{m-1} e^{-x^2/2} \mathbb{1}_{[0,+\infty[}(x). \tag{6.19}$$

The Mellin transform $\hat{g}(s)$ of $g(x)$ is given by

$$\begin{aligned}
\hat{g}(s) &= \int_0^{+\infty} x^{s-1} g(x) dx = \lambda \int_0^{+\infty} x^{m+s-2} e^{-x^2/2} dx \\
&= \lambda \int_0^{+\infty} (\sqrt{2u})^{m+s-2} e^{-u} \frac{du}{\sqrt{2u}} && (6.20) \\
&= \lambda 2^{(m+s-2)/2} \int_0^{+\infty} u^{\frac{m+s-1}{2}-1} e^{-u} du && (6.21) \\
&= \frac{1}{\Gamma(m/2)} 2^{(s-1)/2} \Gamma\left(\frac{m+s-1}{2}\right). && (6.22)
\end{aligned}$$

The Mellin transform is evaluated on positive values only. However, since U is symmetric, U is completely determined by the Mellin transform $\hat{f}(s)$ of $|U|$, which can be computed using $\hat{g}(s)$ for $m = 1$:

$$\hat{f}(s) = \frac{1}{\sqrt{\pi}} 2^{(s-1)/2} \Gamma\left(\frac{s}{2}\right). \tag{6.23}$$

The product of the two Mellin transforms is

$$\hat{g}(s)\hat{f}(s) = \frac{1}{\sqrt{\pi}} \times \frac{2^{s-1}}{\Gamma(m/2)} \Gamma\left(\frac{s}{2}\right) \Gamma\left(\frac{s+m-1}{2}\right) \tag{6.24}$$

$$= \int_0^{+\infty} x^{s-1} \left(\lambda' x^{\frac{m-1}{2}} K_{(m-1)/2}(x)\right) dx, \tag{6.25}$$

where the last equality follows from the change of variables s into $s+(m-1)/2$ and ν into $(m-1)/2$ in the Mellin transform of the Bessel function K_ν (c.f. B.2 and [Obe74]):

$$\int_0^{+\infty} x^{s-1} K_\nu(x) dx = 2^{s-2} \times \Gamma\left(\frac{s-\nu}{2}\right) \times \Gamma\left(\frac{s+\nu}{2}\right). \tag{6.26}$$

From (6.25) and by identification, we conclude that the distribution of $|U|Y$ is the absolute value of a Bessel distribution $\mathfrak{B}(m,1)$. As UY is symmetric, we can deduce that $UY \sim \mathfrak{B}(m,1)$. □

6.2.2 Distributions at finite length

Theorem 26 (**Distribution of the observed vector** y)**.** *The observed vector $y = \phi x$ follows a standard multivariate Bessel distribution $\mathfrak{B}_m(k, \sigma\sigma_x/\sqrt{m})$ of dimension m, scale parameter $\sigma\sigma_x/\sqrt{m}$ and size parameter k.*

Proof. The characteristic function of the random vector y is a function of $s = (s_1, \ldots, s_m)^T$ and satisfies

$$\begin{aligned} G(s) &= \mathbb{E}[\exp\left(i\langle s, y\rangle\right)] \\ &= \prod_{l\in\Lambda} \mathbb{E}\left[\prod_{j=1}^{m} \exp\left(i s_j \phi_{jl} x_l\right)\right] \qquad (6.27) \\ &= \prod_{l\in\Lambda} \mathbb{E}\left[\mathbb{E}\left[\exp\left(i x_l \sum_{j=1}^{m} \left(s_j \phi_{jl}\right)\right) \middle| x_l\right]\right] = \prod_{l\in\Lambda} \mathbb{E}\left[\exp\left(-\sigma^2 x_l^2 ||s||^2)/(2m)\right)\right] \\ &= \mathbb{E}\left[\exp(-tZ)\right]^k \qquad (6.28) \\ &= \left[1 + \left(\sigma\sigma_x ||s||/\sqrt{m}\right)^2\right]^{-k/2} \qquad (6.29) \end{aligned}$$

(6.27) follows from the independence of the $\phi_l x_l$. Introducing $t = \sigma^2\sigma_x^2||s||^2/(2m)$ and $Z = (x_l/\sigma_x)^2$, since the x_l are i.i.d., we obtain (6.28). Equation (6.29) comes from the fact that $Z \sim \chi_1^2$, whose moment-generating function is $\mathbb{E}[e^{tZ}] = (1-2t)^{-1/2}$. According to [KKP01, p. 257] and [MS90, p. 156], $G(s)$ is the characteristic function of a multivariate standard Bessel distribution. Therefore, y follows a multivariate standard Bessel distribution with dimension m, scale parameter $\sigma\sigma_x/\sqrt{m}$ and size parameter k. □

Theorem 27 (**Distribution of scalar products at iteration** 1)**.** *The scalar product $W_j^d(1) = \langle \phi_j, y\rangle$ is the product of two random variables Z and V, where $\sqrt{m}Z$ follows a χ distribution with m degrees of freedom and V follows a Bessel distribution,*

- $\mathfrak{B}_1(k, \sigma^2\sigma_x/\sqrt{m})$*, independent of Z, if $j \notin \Lambda$,*

- $\mathfrak{B}_1(m+k-1, \sigma^2\sigma_x/\sqrt{m}))$, *dependent on* Z, *if* $j \in \Lambda$.
- $W^d(1)$ *is partially exchangeable and its coordinates are uncorrelated.*

Proof. We first introduce the normalized atom vector

$$u_j = \phi_j / ||\phi_j|| \tag{6.30}$$

and decompose $W_j^d(1)$ into the product:

$$W_j^d(1) = \underbrace{\frac{||\phi_j||}{\sigma}}_{Z} \underbrace{\sigma \sum_{l\in\Lambda} x_l \langle \phi_l, u_j \rangle}_{V=\sigma\langle y, u_j\rangle} \tag{6.31}$$

$\phi_j \sim \mathcal{N}(0, \sigma^2/mI_m)$, thus $\sqrt{m}Z \sim \chi_m$. For V, we need to consider two cases:

- If $j \notin \Lambda$, ϕ_l and u_j are independent. Therefore, $\langle \phi_l, u_j \rangle$ follows a $\mathcal{N}(0, \sigma^2/m)$ distribution (see Lem. 4), and V follows a standard univariate Bessel random variable $\mathfrak{B}(k, \sigma^2\sigma_x/\sqrt{m})$, as a dot product of two independent Gaussian vectors (see Thm. 39). Each $\langle \phi_l, u_j \rangle$ is independent of ϕ_j (see Lem. 4), thus Z and V are independent.

- If $j \in \Lambda$, then y and u_j are dependent, and

$$V = \sigma \sum_{l\in\Lambda_{-j}} x_l \langle u_j, \phi_l \rangle + \sigma x_j ||\phi_j|| \tag{6.32}$$

$\langle u_j, \phi_l \rangle$ follows a $\mathcal{N}(0, \sigma^2/m)$ distribution (see Lem. 4) independent of x_l, so the first sum in (6.31) follows a Bessel distribution $\mathfrak{B}(k-1, \sigma_x\sigma^2/\sqrt{m})$ (cf. Thm. 39). The second term in (6.31) follows a Bessel distribution $\mathfrak{B}(m, \sigma_x\sigma^2/\sqrt{m})$ (cf. Lem. 6). In the polar decomposition $\phi_j = ||\phi_j|| \times u_j$, u_j and $||\phi_j||$ are independent [FKN90, Thm. 2.3. p. 30], so $x_j||\phi_j||$ and $\sum_l x_l \langle u_j, \phi_l \rangle$, are also independent for $l \in \Lambda_{-j}$. Eventually, the sum of two independent Bessel distributions with same scale parameter is a Bessel distribution whose size is the sum of the size of each component [KKP01, Prop. 4.1.1, p. 181].

- For the sake of simplicity, we omit here the index d and 1 and write W_j instead of $W_j^d(1)$. From (6.31) and (6.32), the coordinates of each subvector $(W_j)_{j\in\Lambda}$ and $(W_j)_{j\in\overline{\Lambda}}$ are exchangeable: by symmetry, they are identically distributed and invariant by any permutation of their coordinates. Thus, the vector $(W_j)_{j=1..d}$ is partially exchangeable with two classes corresponding to Λ and $\overline{\Lambda}$.

We now prove that $\mathrm{cov}(W_i, W_j) = 0$ whenever $i \neq j$. Since W_i is centered,

$$\mathrm{cov}(W_i, W_j) = \mathbb{E}\left[W_i W_j\right] = \mathbb{E}\left[\langle \phi_i, y \rangle \langle \phi_j, y \rangle\right] = \mathbb{E}\left[\sum_{l,l'=1}^{k} x_l x_{l'} \langle \phi_i, \phi_l \rangle \langle \phi_j, \phi_{l'} \rangle\right] \tag{6.33}$$

$$= \sum_{l,l'=1}^{k} \mathbb{E}[x_l x_{l'}] \mathbb{E}\left[\langle \phi_i, \phi_l \rangle \langle \phi_j, \phi_{l'} \rangle\right], \tag{6.34}$$

because x and ϕ are independent. If $l \neq l'$, by independence of the centered x entries, $\mathbb{E}[x_l x_{l'}] = \mathbb{E}[x_l]\mathbb{E}[x_{l'}] = 0$. If $l = l'$, we prove that $\mathbb{E}\left[\langle \phi_i, \phi_l \rangle \langle \phi_j, \phi_l \rangle\right]$=0, which implies $\mathrm{cov}(W_i, W_j) = 0$:

$$\mathbb{E}\left[\langle\phi_i,\phi_l\rangle\langle\phi_j,\phi_l\rangle\right] = \mathbb{E}\left[\phi_i^T\phi_l\phi_l^T\phi_j\right] \tag{6.35}$$
$$= \mathbb{E}\left[\phi_i^T\right]\mathbb{E}\left[\phi_l\phi_l^T\phi_j\right] \text{ if } i \neq l \tag{6.36}$$
$$= 0, \tag{6.37}$$

because the entries of ϕ_i are centered and independent of ϕ_l for $l \neq j$. If $i = l$, then $j \neq l$ (otherwise $i = j$) and the expression is also zero, which concludes the proof. □

The next theorem determines the distribution of scalar products when, at a given iteration t, the $t-1$ already selected atoms are fixed, and belong to the good support. For reasons of symmetry, we can assume that the selected atoms are indexed by $1,...,t-1$. Note that we do not consider any conditional distribution, but a deterministic initial situation where $\phi_1,...,\phi_{t-1}$ have been selected and we just consider what happens at the next iteration. The proof uses Lem. 5, where Q_t is random, whereas here Q_t is deterministic. We retain the reference to this Lemma, as the conclusion remains valid whether Q_t is random or not. We remind that Q_t represents the orthogonal projector onto the complement of the space spanned by the vectors $\phi_1,...,\phi_t$.

Theorem 28 (**Distribution of scalar products at iteration** $t > 1$)**.** *$W_j^d(t) = \langle r_t, \phi_j\rangle$ is either zero (if $j \leq t-1$), or the product of two random variables Z and V. $\sqrt{m}Z$ follows a χ distribution with $m-t+1$ degrees of freedom and V follows a Bessel distribution whose parameters depend on the fact that ϕ_j belongs to the true support or not:*

- *$\mathfrak{B}_1(k-t+1, \sigma^2\sigma_x/\sqrt{m})$ whenever $j \notin \Lambda$,*
- *$\mathfrak{B}_1(m+k-2t+1, \sigma^2\sigma_x/\sqrt{m})$ whenever $j \in \Lambda$.*
- *$W^d(t)$ is partially exchangeable and its coordinates are uncorrelated.*

Proof. Note first that if $j \in \Lambda_{t-1}$, since $\Lambda_{t-1} = [\![1, t-1]\!]$, then ϕ_j is orthogonal to Q_{t-1}, and $W_j(t) = \langle y, Q_{t-1}\phi_j\rangle = 0$ for all $j = 1,..,t-1$.

We introduce the normalized vector

$$u_j = Q_{t-1}\phi_j / ||Q_{t-1}\phi_j|| \tag{6.38}$$

and decompose $W_j^d(t)$ into

$$W_j^d(t) = \langle Q_{t-1}y, \phi_j\rangle = \langle y, Q_{t-1}\phi_j\rangle = \frac{||Q_{t-1}\phi_j||}{\sigma}\sigma\langle y, u_j\rangle = Z.V \tag{6.39}$$

where the last equality in (6.39) follows from the fact that Q_{t-1} is an orthogonal projector. This decomposition is similar to (6.31) in Thm. 27, except that u_j is no more computed from an atom but a projected atom. This requires new proof to derive the distribution.

For Z, note that $\phi_j \sim \mathcal{N}(0, \sigma^2/mI_m)$, that Q_{t-1} is the random orthogonal projector onto the orthogonal of the subspace spanned by the vectors $\{\phi_i\}_{i\in\Lambda_{t-1}}$, and that Q_{t-1} is independent of ϕ_j. Then, from Lem. 5, $\sqrt{m}Z \sim \chi(m-t+1)$. For V, we need to consider two cases:

- If $j \in [\![k+1, d]\!]$, i.e. $j \notin \Lambda$,

$$V = \sigma \sum_{l=1}^{t-1} x_l \langle \phi_l, u_j \rangle + \sigma \sum_{l=t}^{k} x_l \langle \phi_l, u_j \rangle \tag{6.40}$$

$$= \sigma \sum_{l=t}^{k} x_l \langle \phi_l, u_j \rangle \tag{6.41}$$

since u_j is projected with Q_{t-1} and is orthogonal to all ϕ_l, with $1 \leq l \leq t-1$. By Lem. 5, $\langle \phi_l, u_j \rangle$ is independent of u_j, so that $\langle \phi_l, u_j \rangle$ for $l = t, .., k$ are mutually independent. The rest of the proof follows the same line as the proof of Thm. 27 with in the first case $k-t+1$ terms instead of k terms. Therefore, $V \sim \mathfrak{B}(k-t+1, \sigma^2 \sigma_x / \sqrt{m})$.

• If $j \in [\![t, k]\!]$, i.e. $j \in \Lambda$

$$V = \sigma \sum_{l \in \Lambda_{t \setminus j}} x_l \langle \phi_l, u_j \rangle + \sigma x_j \langle \phi_j, u_j \rangle = V_1 + V_2. \tag{6.42}$$

For V_2 since Q_{t-1} is an orthogonal projector,

$$\langle \phi_j, Q_{t-1} \phi_j \rangle = \langle Q_{t-1} \phi_j, Q_{t-1} \phi_j \rangle = ||Q_{t-1} \phi_j||^2 \tag{6.43}$$

thus

$$V_2 = \sigma x_j \langle \phi_j, u_j \rangle = \sigma x_j ||Q_{t-1} \phi_j|| \tag{6.44}$$

Now, $\langle \phi_l, u_j \rangle$ for $l \neq j$ is independent from $||Q_{t-1}\phi_j||$, so V_1 and V_2 are independent. The rest of the proof follows the same line as the proof of Thm. 27 in the second case, with respectively $k-t$ terms instead of k for V_1 and $m-t+1$ terms instead of m for V_2. Therefore,

$$V \sim \mathfrak{B}(m+k-2t+1, \sigma^2 \sigma_x / \sqrt{m}), \tag{6.45}$$

• The proof of partial exchangeability of $W^d(t)$ is identical to that of Thm. 27. Let us prove the non-correlation. Since $W_j^d(t)$ is centered,

$$\operatorname{cov}(W_i^d(t), W_j^d(t)) = \mathbb{E}\left[W_i^d(t) W_j^d(t) \right] = \mathbb{E}\left[\langle \phi_i, r_t \rangle \langle \phi_j, r_t \rangle \right] \tag{6.46}$$

$$= \mathbb{E}\left[\langle \phi_i, Q_{t-1} y \rangle \langle \phi_j, Q_{t-1} y \rangle \right] \tag{6.47}$$

$$= \sum_{l,l'=1}^{k} \mathbb{E}[x_l x_{l'}] \mathbb{E}\left[\langle Q_{t-1}\phi_i, \phi_l \rangle \langle Q_{t-1}\phi_j, \phi_{l'} \rangle \right] \tag{6.48}$$

If $l \neq l'$, by independence of the centered x entries, $\mathbb{E}[x_l x_{l'}] = \mathbb{E}[x_l]\mathbb{E}[x_{l'}] = 0$, and if $l = l'$,

$$\mathbb{E}\left[\langle Q_{t-1}\phi_i, \phi_l \rangle \langle Q_{t-1}\phi_j, \phi_l \rangle \right] = \mathbb{E}\left[\phi_i^T Q_{t-1} \phi_l \phi_l^T Q_{t-1} \phi_j \right] \tag{6.49}$$

$$= \mathbb{E}\left[\phi_i^T \right] \mathbb{E}\left[Q_{t-1} \phi_l \phi_l^T Q_{t-1} \phi_j \right] \text{ if } i \neq l \tag{6.50}$$

$$= 0, \tag{6.51}$$

because the entries of ϕ_i are centrered and independent of ϕ_l for $l \neq j$. If $i = l$, then $j \neq l$ (otherwise $i = j$) and the expression is also zero, hence the conclusion.

□

6.2.3 Asymptotic distributions in the large system regime

Theorem 29 (**Independent Gaussian scalar products at iteration 1**). *When d tends to infinity, in the large system regime, the sequence of vectors $(W_j^d(1))_j \in \mathbb{R}^d$ tends to an infinite length sequence $(\widetilde{W}_j(1))_j$ of scalar products. The entries of this infinite sequence are independent and follow one of the two distributions:*
- $\widetilde{W}_j(1) \sim \mathcal{N}\left(0, \sigma^4\sigma_x^2 \times \frac{1}{\mu}\right)$ *if* $j \notin \Lambda$,
- $\widetilde{W}_j(1) \sim \mathcal{N}\left(0, \sigma^4\sigma_x^2 \times \left(1 + \frac{1}{\mu}\right)\right)$ *if* $j \in \Lambda$.

Proof. We decompose $W_j^d(1) = Z.V$ as in (6.31). Since $\sqrt{m}Z$ follows a χ distribution, from the strong law of large number (LLN) [Fel71, p.237] Z converges to 1 almost surely (and in probability), as d tends to infinity.

Now, since the Bessel distribution is infinitely divisible [KKP01, Prop. 4.1.1 p.181], it is always possible to write a Bessel variable as a sum of independent Bessel variables, and the central limit theorem (CLT) shows that, if $Y_d \sim \mathfrak{B}(n_d, \sigma_d)$ and $\lim_{d\to+\infty} n_d \times \sigma_d^2 = \beta$, then Y_d converges in distribution toward $\mathcal{N}(0, \beta)$. Therefore, if $j \notin \Lambda$, from Thm. 27

$$n_d\sigma_d^2 = k\frac{\sigma^4\sigma_x^2}{m} \underset{d\to+\infty}{\longrightarrow} \frac{\sigma^4\sigma_x^2}{\mu} \tag{6.52}$$

and if $j \in \Lambda$, from Thm. 27,

$$n_d\sigma_d^2 = (m+k-1)\frac{\sigma^4\sigma_x^2}{m} \underset{d\to+\infty}{\longrightarrow} \sigma^4\sigma_x^2(1+\frac{1}{\mu}). \tag{6.53}$$

Now, since $\widetilde{W}(1)$ is a Gaussian sequence, non-correlation implies independence of the entries. This concludes the proof. □

Theorem 30 (**Gaussian scalar products at iteration $t > 1$**). *Assume $\phi_1, ..., \phi_{t-1}$ have been selected during the $t-1$ previous iterations. In the large system regime, each scalar product converges in distribution*

$$W_j^d(t) \underset{d\to\infty}{\xrightarrow{d}} \widetilde{W}_j(t)\ ;\ \forall t, \forall j \geq t. \tag{6.54}$$

- *The entries of $W^d(t)$ are independent.*
- *If $j \in \Lambda_t$, $\widetilde{W}_j(t) = 0$,*
- *If $j \in \Lambda/\Lambda_t$, $\widetilde{W}_j(t) \sim \mathcal{N}(0, v)$ with $v = \sigma^4\sigma_x^2\left(1 + \frac{1-2\tau}{\mu}\right)\left(1 - \frac{\tau}{\mu}\right)$,*
- *If $j \in \overline{\Lambda}$, $\widetilde{W}_j(t) \sim \mathcal{N}\left(0, v'\right)$, with $v' = \sigma^4\sigma_x^2\frac{1-\tau}{\mu}\left(1 - \frac{\tau}{\mu}\right)$,*

where $\tau = t/k \in [0, 1[$ is the fraction of iterations already done.

Proof. We decompose $W_j^d(t) = Z.V$ as in (6.39). Since $\sqrt{m}Z$ follows a χ distribution with $m - t + 1$ degrees of freedom, from the strong law of large number (LLN) [Fel71, p.237] Z converges to

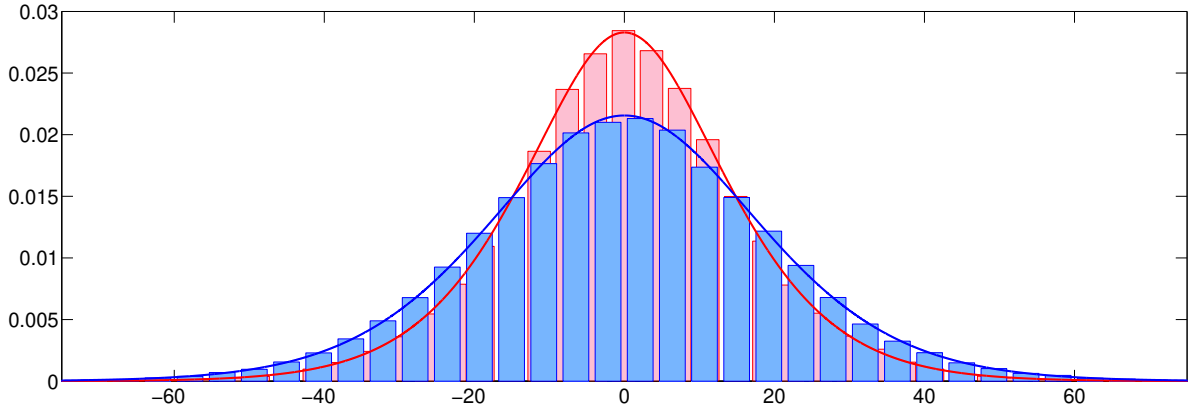

Figure 6.3 – 10^5 samples of a scalar product from the right support (in blue) and its complementary (in red) for $d = 10$, $m = 5$, $\sigma = 3$, $\sigma_x = 2$. The red and blue lines represent a Gaussian probability density function with the two possible limit variances, at the first iteration.

$$\lim_{d\to+\infty} \sqrt{\frac{m}{m-t+1}} = \left(1-\frac{\tau}{\mu}\right)^{-1/2} \tag{6.55}$$

almost surely (and in probability), as d tends to infinity.

The Bessel distribution is infinitely divisible [KKP01, Prop. 4.1.1 p.181], so by the central limit theorem (CLT), if $Y_d \sim \mathfrak{B}(n_d, \sigma_d)$ and $\lim_{d\to+\infty} n_d \times \sigma_d^2 = \beta$, Y_d converges in distribution toward $\mathcal{N}(0,\beta)$. Therefore, if $j \notin \Lambda$, from Thm. 28,

$$n_d\sigma_d^2 = \sigma^4\sigma_x^2\frac{k-t+1}{m} \underset{d\to+\infty}{\longrightarrow} \sigma^4\sigma_x^2\frac{1-\tau}{\mu} \tag{6.56}$$

and if $j \in \Lambda \setminus \Lambda_t$, from Thm. 28,

$$n_d\sigma_d^2 = \sigma^4\sigma_x^2\frac{m+k-2t+1}{m} \underset{d\to+\infty}{\longrightarrow} \sigma^4\sigma_x^2\left(1+\frac{1-2\tau}{\mu}\right). \tag{6.57}$$

The limit of ZV in distribution is the limit of V times the multiplicative factor $(1-\tau/\mu)^{-1/2}$, which can be included in the variance of the Gaussian limit and gives the announced result.

If $j \in \Lambda_t$, since $W_j^d(t) = 0$, so is the limit $\widetilde{W}_j(t)$.

Eventually, since $\widetilde{W}(t)$ is a Gaussian sequence, non-correlation implies independence of the entries. □

6.3 A Gaussian integral approximation

The OMP algorithm relies on comparisons between the maxima of (absolute values of) independent Gaussian scalar products. To determine the success probability, it is therefore necessary to evaluate the probability that the maximum of one semi-Gaussian vector is greater than the maximum of another independent semi-

Gaussian vector. This is the goal of this subsection.

6.3.1 A probability involving the maximum of two Gaussian vectors

Let $X = (X_1, ..., X_k)$ and $Y = (Y_1, ..., Y_{n-k})$ be the absolute values of independent centered Gaussian vectors of variance respectively σ_1^2 and σ_2^2, with

$$\sigma_1^2 = 1 + 1/\mu \text{ and } \sigma_2^2 = 1/\mu. \tag{6.58}$$

X and Y are independent. $\mu > 1$, $\gamma \in]0,1[$ are real parameters and $k = \gamma n$ is supposed to be an integer (this implies that $\gamma \in \mathbb{Q}$ but this condition will be removed later). Let

$$\alpha = \sigma_1/\sigma_2 = \sqrt{1+\mu} > \sqrt{2}. \tag{6.59}$$

Let $X_\bullet = (X_{(1)}, ..., X_{(k)})$ and $Y_\bullet = (Y_{(1)}, ..., Y_{(n-k)})$ be the order statistics of the two vectors ($X_{(1)}$ is the min of the X_i and $X_{(k)}$ is the max) and let us consider the event

$$\left[X_{(k)} > Y_{(n-k)}\right] = \left[\max_{i=1}^{k} X_i > \max_{j=1}^{n-k} Y_j\right]. \tag{6.60}$$

The event occurs when the maximum of $X_1, ..., X_k$ and $Y_1, ..., Y_{n-k}$ is attained by one of the X_i's. X_i and Y_j follow half normal distributions of density $g(x/\sigma)/\sigma$ and cumulative function $G(x/\sigma)$ with $\sigma = \sigma_1$ or $\sigma = \sigma_2$ and

$$g(x) = \frac{2}{\sqrt{2\pi}} e^{-x^2/2} \mathbb{1}_{[0,+\infty[}(x). \tag{6.61}$$

The cumulative density function of $X_{(k)}$ is, by independence of the X_i, $G(x/\sigma_1)^k$. Thus, the density function is given by

$$\frac{k}{\sigma_1} g\left(\frac{x}{\sigma_1}\right) G\left(\frac{x}{\sigma_1}\right)^{k-1} \tag{6.62}$$

and likewise, the cumulative density function of $Y_{(n-k)}$ is given by

$$\mathbb{P}\left[Y_{(n-k)} < x\right] = G\left(\frac{x}{\sigma_2}\right)^{n-k}. \tag{6.63}$$

With the previous notations, after changing x to x/σ_1, one has:

$$\mathbb{P}\left[X_{(k)} > Y_{(n-k)}\right] = \mathbb{P}\left[X_{(k)} > Y_{(n-k)}\right] = \int_0^{+\infty} \mathbb{P}\left[Y_{(n-k)} < x\right] \mathbb{P}_{X_{(k)}}(dx) \tag{6.64}$$

$$= k \int_0^{+\infty} G(\alpha x)^{n-k} G(x)^{k-1} g(x)\, dx = J_n(\alpha, \gamma). \tag{6.65}$$

From now on, we omit α and γ and refer to J_n instead of $J_n(\alpha, \gamma)$. Our main result is the following theorem.

Theorem 31.

$$J_n - 1 \overset{n\to+\infty}{\sim} -\lambda \frac{(\ln n)^{(A-1)/2}}{n^{\mu}}, \tag{6.66}$$

where

$$\lambda = \pi^{\frac{A-1}{2}} \frac{\alpha(1-\gamma)}{\gamma^A} \Gamma(\alpha^2 - A + 1) \prod_{j=1}^{A-1} (\alpha^2 - j), \tag{6.67}$$

Γ *is the Euler gamma function,* $A = \lfloor \alpha^2 \rfloor$ *if* α^2 *is not an integer, and* $A = \alpha^2 - 1 = \mu$ *if* α^2 *is an integer.*

The semi-Gaussian random variables X_j and Y_j can represent the absolute values of the scalar products W_j involved in the OMP algorithm. These scalar products indeed follow Gaussian and independent distributions in the asymptotic LSR model. Consequently, the probability represented by J_n can be viewed as the success probability in the first iteration of OMP, under the assumptions of the asymptotic regime. The theorem proves that this success probability tends to 1 as the dimension of the vectors tends to infinity and specifies the rate of convergence towards 1.

Proof. We begin by writing J_n in a more convenient form:

$$J_n = k \int_0^{+\infty} e^{nL(x)} \frac{g(x)}{G(x)} dx, \tag{6.68}$$

with

$$L(x) = (1-\gamma) \ln G(\alpha x) + \gamma \ln G(x). \tag{6.69}$$

L is a negative, increasing and concave function on $]0, +\infty[$ (c.f. Fig. 6.4). g, G and L are functions of class C^∞ on $]0, +\infty[$. L has a singularity in 0^+. We will need the function h_1, defined by

$$h_1(x) = \frac{g(x)}{G(x)L'(x)}. \tag{6.70}$$

One can easily prove that h_1 is a class C^∞ function on $]0, +\infty[$, not defined at 0, but such that :

$$h_1(x) = 1 + \frac{1}{3}(\alpha^2 - 1)(1-\gamma)x^2 + o(x^2). \tag{6.71}$$

$$h_1(x) = \frac{1}{\gamma} - \frac{\alpha(1-\gamma)}{\gamma^2} e^{-(\alpha^2-1)x^2/2} + \frac{1}{\gamma^2}\sqrt{\frac{2}{\pi}}\alpha(1-\gamma)\frac{e^{-\alpha^2 x^2/2}}{x}\left[1 - \frac{1}{x^2} + o(\frac{1}{x^2})\right], \tag{6.72}$$

so that h_1 can be extended in the neighborhood of 0, into a C^1 function, by letting $h_1(0) = 1$ and $h_1'(0) = 0$. For the sake of simplicity, we will continue to refer to this extended function as h_1. It is defined on $[0, +\infty[$ and is both upper and lower bounded (c.f. Fig. 6.4).

Let's return to J_n. Integrating by parts is licit because this generalized integral is well-defined (since it represents a probability) and the right-hand side of the expression has a finite limit. Additionally, $\exp(nL(x))$ is

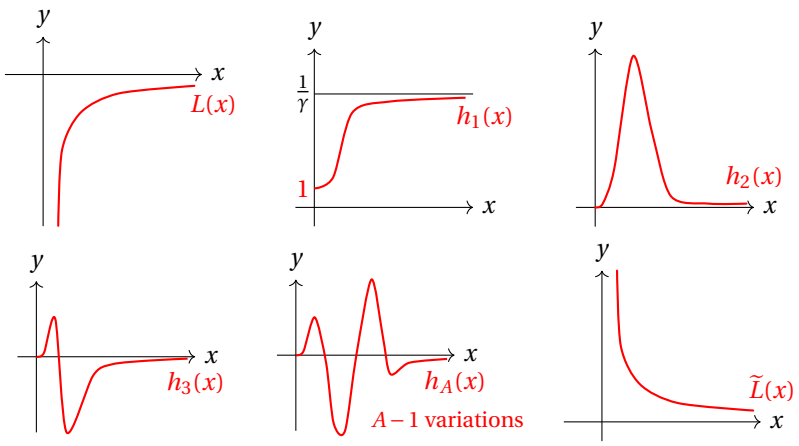

Figure 6.4 – Sketch of the representative curves of functions L, h_i, $\widetilde{L}$.

bounded between 0 and 1, $h_1'(x)$ is a C^∞ function on $]0,+\infty[$ and tends to 0 as x approaches 0 and $+\infty$. Therefore,

$$J_n = \frac{k}{n}\left[e^{nL(x)} \times \frac{g(x)}{G(x)L'(x)}\right]_0^{+\infty} - \frac{k}{n}\int_0^{+\infty} e^{nL(x)} \times h_1'(x)dx \tag{6.73}$$

$$= \frac{k}{n}\left[e^{nL(x)} \times h_1(x)\right]_0^{+\infty} - \frac{k}{n}\int_0^{+\infty} e^{nL(x)} \times h_1'(x)dx. \tag{6.74}$$

We have

$$\lim_{x\to 0} e^{nL(x)} \times h_1(x) = 0 \tag{6.75}$$

and

$$\lim_{x\to+\infty} e^{nL(x)} \times h_1(x) = \frac{1}{\gamma}, \tag{6.76}$$

so that

$$J_n = \frac{k}{n\gamma} - \frac{k}{n}\int_0^{+\infty} e^{nL(x)} \times h_1'(x)dx. \tag{6.77}$$

$h_1'(x)$ is bounded and continuous on $[0,\infty[$, tends to 0 as x approaches 0 and ∞. It is thus integrable and by the Dominated Convergence Theorem of Lebesgue, we have:

$$\lim_{n\to+\infty}\int_0^{+\infty} e^{nL(x)}h_1'(x)dx = \int_0^{+\infty}\lim_{n\to+\infty} e^{nL(x)}h_1'(x)dx = 0. \tag{6.78}$$

So far, we have proven that for all $\mu > 1$ and $\gamma \in]0,1[$,

$$\lim_{n\to+\infty} J_n = 1. \tag{6.79}$$

Now, define recursively - if any - for $i \geq 2$,

$$h_i(x) = \frac{h'_{i-1}(x)}{L'(x)}. \tag{6.80}$$

$L'(x)$ is a C^∞ function on $]0,\infty[$ and is never zero, so that $h_i(x)$ is also C^∞. As long as $h'_i(x)$ is integrable at infinity, integration by parts is permissible. Under this assumption :

$$\int_0^{+\infty} e^{nL(x)} \times h'_1(x)dx = \frac{1}{n}\left[e^{nL(x)} \times \frac{h'_1(x)}{L'(x)}\right]_0^{+\infty} - \frac{1}{n}\int_0^{+\infty} e^{nL(x)} \times h'_2(x)dx. \tag{6.81}$$

$h_2(x)$ tends to 0 as x approaches 0 and when x approaches infinity. Therefore, the expression in brackets is zero and

$$J_n - 1 = O\left(\frac{1}{n}\right). \tag{6.82}$$

An asymptotic expansion of h_i at infinity gives (by recurrence)

$$h_i(x) \sim \lambda_i x^{i-1} \exp\left(-(\alpha^2 - i)x^2/2\right), \tag{6.83}$$

with

$$\lambda_i = (-1)^i \left(\frac{\pi}{2}\right)^{\frac{i-1}{2}} \frac{\alpha(1-\gamma)}{\gamma^{i+1}} \prod_{j=1}^{i-1}(\alpha^2 - j). \tag{6.84}$$

After A integrations by parts (where A equals the integer part of α^2 if α^2 is not an integer, or $A = \alpha^2 - 1$) otherwise, $h_i(x)$ tends to infinity and no other integration by parts are possible. Thus:

$$J_n - 1 = O\left(\frac{1}{n^{A-1}}\right). \tag{6.85}$$

Let $\lambda = \lambda_A$. We now use a generalization of the Watson lemma [Die80; Won89] to obtain a more precise rate of convergence. After $A-1$ integrations by parts,

$$J_n = \frac{k}{n\gamma} + \frac{k}{n} \times \frac{(-1)^{A-1}}{n^{A-2}} \int_0^{\infty} e^{nL(x)} h'_{A-1}(x)dx. \tag{6.86}$$

Let I_n be the integral in the right-hand side of the equation. Let

$$t = -L(x). \tag{6.87}$$

This change of variable is valid because L is one-to-one on $]0,\infty[$ and differentiable.

$$I_n = \int_0^{+\infty} e^{-nt} h_A(L^{-1}(-t))dt, \tag{6.88}$$

because $x = L^{-1}(-t)$ and $dt = -L'(x)dx$. Let $\widetilde{L}$ be the function defined by $\widetilde{L}(t) = L^{-1}(-t)$. We have

$$I_n = \int_0^{+\infty} h_A \circ \widetilde{L}(t) \times e^{-nt} dt, \tag{6.89}$$

which is exactly the Laplace transform of $h_A \circ \widetilde{L}$. Let Ψ denote this function. $\widetilde{L}$ is a C^∞ function, decreasing from $]0,\infty[$ to $]0,\infty[$. It tends to $+\infty$ as t approaches 0 and to 0 at $+\infty$. h_A is also C^∞, tends to zero at 0 and $+\infty$. $h_A \circ \widetilde{L}$ is a C^∞ function on $]0,\infty[$ and tends to 0 as x approaches 0^+.

Although Ψ is not defined in 0, the existence of a finite limit at that point allows us to extend Ψ into a C^1 function. We can then provide an asymptotic expansion:

$$-t = L(x) = -\gamma\sqrt{\frac{2}{\pi}}\frac{e^{-x^2/2}}{x}\left[1 + o(1/x)\right]. \tag{6.90}$$

So,

$$\ln t = \ln\left(\gamma\sqrt{\frac{2}{\pi}}\right) - \frac{x^2}{2} - \ln x + o(1/x). \tag{6.91}$$

As x tends to infinity, the dominant term of the right-hand side is $-x^2/2$. Therefore,

$$x^2 \sim -2\ln t \tag{6.92}$$

As x tends to infinity and t tends to 0^+, we write:

$$x^2 = -2\ln t + \epsilon(t), \tag{6.93}$$

where $\epsilon(t) = o(\ln t)$. By substituting x^2 with its expansion, we get

$$\epsilon(t) = \ln\left(\frac{\pi}{2\gamma^2}\right) + \ln(-2\ln t) + o(1) \tag{6.94}$$

so that

$$x^2 = -2\ln t + \ln(-2\ln t) + \ln\left(\frac{\pi}{2\gamma^2}\right) + o(1). \tag{6.95}$$

Now $\Psi(t) = h_A(x(t))$, and an equivalent of h_A as x tends to infinity is

$$h_A(x) \sim (-1)^A \left(\frac{\pi}{2}\right)^{(A-1)/2} \frac{\alpha(1-\gamma)}{\gamma^{A+1}} \prod_{j=1}^{A-1} (\alpha^2 - j) \times x^{A-1} e^{-(\alpha^2 - A)x^2/2}. \tag{6.96}$$

Let us substitute x^2 with its expansion in this asymptotic equivalent of h_A:

$$\Psi(t) \sim \left(\frac{\pi}{2}\right)^{(A-1)/2} \frac{\alpha(1-\gamma)}{\gamma^{A+1}} \prod_{j=1}^{A-1} (\alpha^2 - j) \times 2^{(A-1)/2} (-\ln t)^{(A-1)/2} t^{\alpha^2 - A} \tag{6.97}$$

As t tends to 0. The conditions of the Laplace-Watson lemma are not satisfied, because Ψ has a logarithmic singularity in 0, but there exists a generalization of this lemma proven by Wong [Won89, p.69-70]. Generalizing the identity

$$\int_0^{\infty e^{i\theta}} t^{u-1} e^{-zt}\, dt = \Gamma(u) z^{-u} \tag{6.98}$$

with $\mathbb{R}e(u) > 0$ and $|\arg(ze^{i\theta})| < \pi/2$, Wong proves that, as $|z| \to +\infty$,

$$\int_0^{\infty e^{i\theta}} t^{u-1} (-\ln t)^{\sigma} e^{-zt}\, dt \sim (\ln z)^{\sigma} \Gamma(u) z^{-u} \tag{6.99}$$

with $\mathbb{R}e(\sigma) > -1$. Applying this formula with $\theta = 0$ (so the path of integration is along the real axis), $z = n$, $\sigma = (A-1)/2$ and $u = \alpha^2 - A + 1$, we obtain, as n tends to infinity,

$$J_n - 1 \sim -\pi^{(A-1)/2} \frac{\alpha(1-\gamma)}{\gamma^A} \prod_{j=1}^{A-1} (\alpha^2 - j) \times \Gamma(\alpha^2 - A + 1) \frac{(\ln n)^{(A-1)/2}}{n^{\alpha^2 - 1}}. \tag{6.100}$$

This concludes the proof. □

6.3.2 Convergence results

Some applications require k and m to be integers. If $\gamma \notin \mathbb{Q}$, one can define $k = \lfloor \gamma n \rfloor$, in which case

$$\left| \gamma - \frac{k}{n} \right| \leq \frac{1}{n}, \tag{6.101}$$

and more generally, one might wonder if the asymptotic equivalent still holds if $k/n \sim \gamma$ instead of $k = \gamma n$ and $m/k \sim \mu$ instead of $m = k\mu$. The following result clarifies the validity of the asymptotic approximation depending on the rates at which k/n tends to γ and m/k tends to μ.

Theorem 32. *Let* $\alpha_n = \sqrt{1 + m/k}$,

$$L_n(x) = \left(1 - \frac{k}{n}\right) \ln G(\alpha_n x) + \frac{k}{n} \ln G(x) \tag{6.102}$$

and

$$\widetilde{J}_n(\alpha) = k \int_0^{+\infty} e^{nL_n(x)} \frac{g(x)}{G(x)} dx. \tag{6.103}$$

Then

$$\left| J_n(\alpha) - \widetilde{J}_n(\alpha) \right| \leq \left| 1 - e^{|k - n\gamma| c(\alpha)} e^{(n-k)\epsilon\left(\frac{m}{k} - \mu\right)} \right| \times J_n(\alpha), \tag{6.104}$$

where $c(\alpha)$ *is a (positive or negative) constant depending only on* α *and* $\epsilon(u)$ *is a function tending to* 0 *as* u *tends to* 0.

Proof. Let

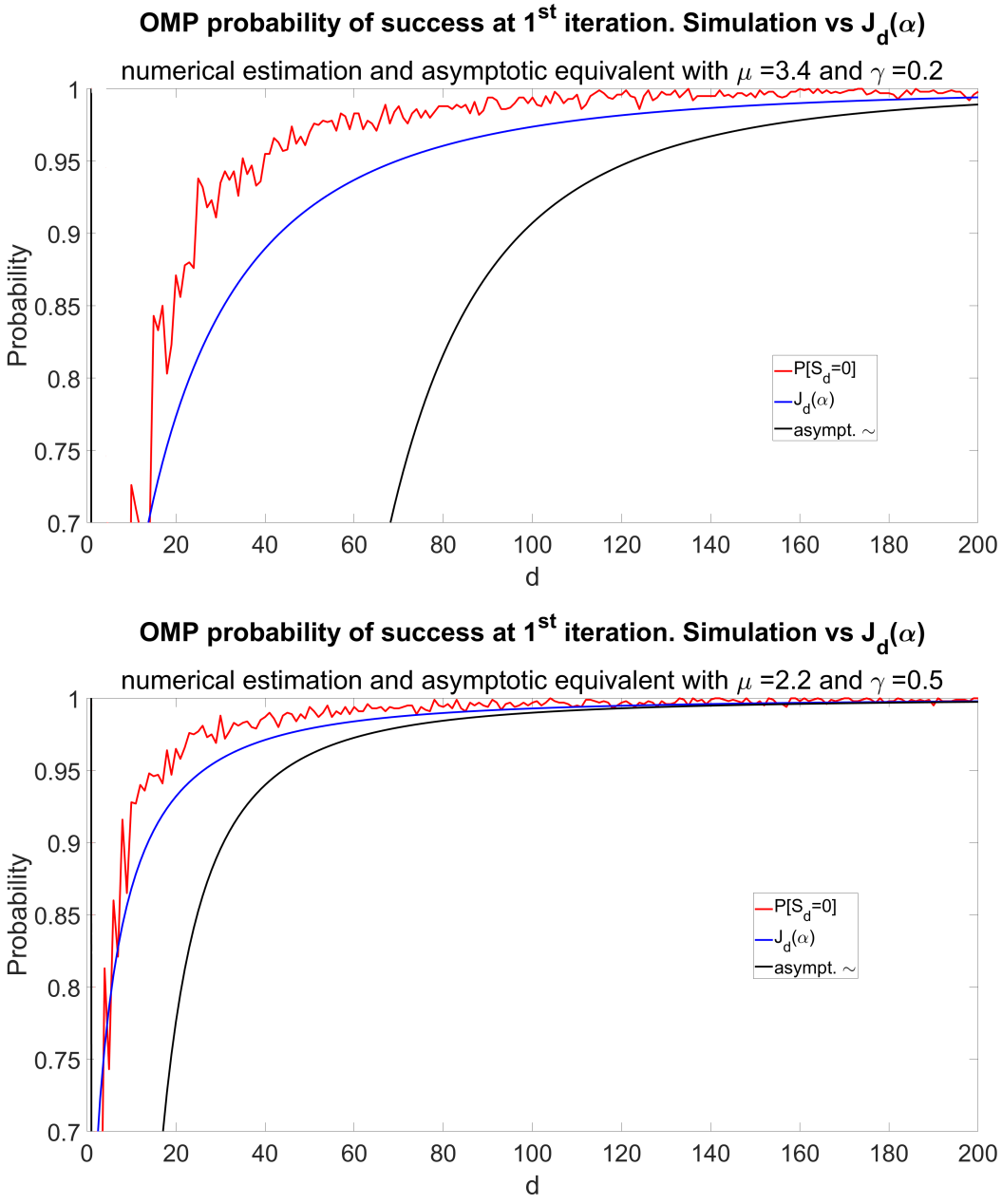

Figure 6.5 – Comparison of the probability of selecting an atom from the right support, at the first iteration of OMP, by simulation of 1000 samples (red line), numerical evaluation of integral $J_n(\alpha)$ (blue line) and asymptotic equivalent (black line). Left: case $\mu = 3.4$ and $\gamma = 0.2$. Right: case $\mu = 2.2$ and $\gamma = 0.5$.

$$\left|J_n(\alpha) - \widetilde{J}_n(\alpha)\right| = \left|\int_0^{+\infty} \frac{g(x)}{G(x)} \left(e^{nL(x)} - e^{nL_n(x)}\right) dx\right| \tag{6.105}$$

$$\leq \sup_{x>0} \left|1 - e^{n[L_n(x)-L(x)]}\right| \times J_n(\alpha) \tag{6.106}$$

$$\Delta_n(x) = L(x) - L_n(x) \tag{6.107}$$

$$= \left(\frac{k}{n} - \gamma\right) \ln G(\alpha x) - \left(\frac{k}{n} - \gamma\right) \ln G(x) - \left(1 - \frac{k}{n}\right) (\ln G(\alpha_n x) - \ln G(\alpha x)) \tag{6.108}$$

$$= \left(\frac{k}{n} - \gamma\right) \ln \frac{G(\alpha x)}{G(x)} + \left(1 - \frac{k}{n}\right) \ln \frac{G(\alpha x)}{G(\alpha_n x)}. \tag{6.109}$$

Let

$$\Lambda_{\alpha,\beta}(x) = \ln\left(\frac{G(\alpha x)}{G(\beta x)}\right). \tag{6.110}$$

If $0 < \alpha < \beta$, Λ is a negative continuous strictly increasing function that maps $]0,\infty[$ onto $]\ln(\alpha/\beta), 0[$. If $\alpha > \beta > 0$, Λ is a positive continuous strictly decreasing function that maps $]0,\infty[$ onto $]\ln(\alpha/\beta), 0[$. Therefore, the maximum of the absolute value of the function is given by $\left|\ln \frac{\alpha}{\beta}\right|$. The first term in $\Delta_n(x)$ is such that

$$\ln\left(\frac{G(\alpha x)}{G(x)}\right) \in]0, \ln \alpha[. \tag{6.111}$$

Thus, the maximum of the exponential of the first term is of the form $\pm|k - n\gamma| \ln \alpha$ with the sign of this expression depending only on the sign of $k/n - \gamma$. In the second term, $n - k$ is always positive but the sign of $\ln(G(\alpha x)/G(\alpha_n x))$ depends on the sign of $m/k - \mu$. Nonetheless, the maximum of the absolute value of the corresponding Λ function is $|\ln(\alpha/\alpha_n)|$, and this expression tends to 0 at the rate at which m/k converges to μ. Hence,

$$\sup_{x>0} \left|1 - e^{n[L_n(x)-L(x)]}\right| = \left|1 - \exp\left(|k - n\gamma| c(\alpha)\right) \times \exp\left((n-k)\epsilon(m/k - \mu)\right)\right| \tag{6.112}$$

with $c(\alpha) = \pm \ln \alpha$ and $\epsilon(m/k - u) = \pm \ln(\alpha/\alpha_n)$. □

As a corollary of this theorem, choosing $k = \lfloor \gamma n \rfloor$ is not sufficient to ensure convergence. To maintain the same limit (of 1) in $\tilde{J}_n(\alpha)$, it is sufficient to require that $k = k_n$ (as a function of n) satisfies

$$\frac{k_n}{n} - \alpha = o\left(\frac{1}{n}\right) \text{ and } \frac{m_n}{k_n} - \mu = o\left(\frac{1}{n}\right). \tag{6.113}$$

Similarly, for the asymptotic equivalence to hold for $\tilde{J}_n(\alpha)$, it is sufficient to require that

$$\frac{k_n}{n} - \alpha = o\left(\frac{1}{n^\mu}\right) \text{ and } \frac{m_n}{k_n} - \mu = o\left(\frac{1}{n^\mu}\right). \tag{6.114}$$

6.4 Gaussian approximation for the probability of success in OMP at a given iteration

We now connect the results of the previous subsections (distributions of the scalar products and the Gaussian integral approximation) to the evaluation of the probability of success in OMP, at a given iteration. This is done in three steps:

- We prove that, in the Large System Regime, the probability of success at the first iteration is the same whether calculated from the exact dependent Bessel-Chi distributions involved in OMP, or from the independent Gaussian integral approximation.
- We prove that the same holds at any iteration except the last one, provided no error occurs up to that iteration.
- We conjecture an approximation of the probability of success for the entire OMP process.

Apart from this main objective, we highlight certain results that may be useful in other contexts, as the three previous technical lemmas, the distributions of scalar products at finite size and its asymptotic limit, and the equivalence of the Gaussian integral.

6.4.1 Approximation at iteration 1

Theorem 33. *Let S_d the probability of success at the first iteration of OMP, where d is the dimension of x. Under the assumptions of the LSR (Def. 6),*

$$\lim_{d\to+\infty} \mathbb{P}(S_d) = 1 \tag{6.115}$$

The rest of this subsection is dedicated to demonstrating this result.

Under the assumptions of the finite length statistical model (Def. 5), the scalar product at iteration 1 is the vector $W^d(1) = W^d \in \mathbb{R}^d$ (for the sake of simplicity, we omit the 1 indicating the iteration) given by

$$W^d = \left(W_1^d, ..., W_d^d\right) = \phi^T \phi X \in \mathbb{R}^d, \tag{6.116}$$

with $W_i^d = \langle \phi_i, y \rangle = \phi_i^T y$. The first k coordinates corresponds to the real support $\Lambda_d = [\![1..k]\!]$ and the last $d-k$ coordinates to the complementary of the real support $\overline{\Lambda}_d = [\![k+1..d]\!]$.

According to Thm. 27 and from the symmetry of the coordinates, $W_1^d, ..., W_k^d$ are uncorrelated, stochastically dependent but exchangeable and so is $W_{k+1}^d, ..., W_d^d$. Thus, the coordinates of W^d are partially exchangeable: they are identically distributed, depending on the fact that the index belongs to the right support Λ_d or not.

When d tends to infinity (remember that when d changes, the vectors W^d are independent), under the assumptions of the LSR (Def. 6), the support of x becomes infinite. Let Λ and $\overline{\Lambda}$, respectively this support and its complementary in $\mathbb{N}$. For all d,

$$\Lambda_d \subset \Lambda \text{ and } \overline{\Lambda}_d \subset \overline{\Lambda}. \tag{6.117}$$

By letting $W_i^d = 0$ for all $i > d$, one can consider $W^d \in \mathbb{R}^\infty$, set of all real sequences. It is important to note that the notation $\Lambda_d = [\![1..k]\!]$ can be misleading: when d increases, Λ_{d+1} and $\overline{\Lambda}_{d+1}$ are complementary sets of

$[\![1..k+1]\!]$ and do not overlap, because a reordering has to be operated with the indices of $\Lambda_d = [\![1..k]\!]$. The space $\mathbb{R}^\infty$ is Polish, so every Borel probability measure defined on $\mathbb{R}^\infty$ is tight [Bil99].

When d tends to infinity, we have proven that $W^d \rightsquigarrow \widetilde{W}$ (convergence in distribution in $\mathbb{R}^\infty$), where $\widetilde{W} = \left(\widetilde{W}_j\right)$ is an infinite Gaussian sequence of centered elements, independent, with two possible variances σ_1^2 and σ_2^2, whether j belongs to Λ or not:

$$\begin{cases} \widetilde{W}_j \sim \mathcal{N}\left(0, \sigma_1^2 = \sigma^4\sigma_x^2\left(1+\frac{1}{\mu}\right)\right) & \text{if } j \in \Lambda \\ \widetilde{W}_j \sim \mathcal{N}\left(0, \sigma_2^2 = \frac{\sigma^4\sigma_x^2}{\mu}\right) & \text{if } j \in \overline{\Lambda} \end{cases} \tag{6.118}$$

This is the result of Thm. 29.

Some sparsity pattern recovery algorithms like OMP rely on scalar products comparisons. From Thm. 25, we know that OMP succeeds at iteration 1 if and only if the following event occurs:

$$\left[W^d \in S_d\right] = \left[\max_{j\in\Lambda_d}\left|W_j^d\right| \geq \max_{i\in\overline{\Lambda}_d}\left|W_i^d\right|\right] \subset \mathbb{R}^d \tag{6.119}$$

with

$$S_d = \left\{w \in \mathbb{R}^d : \max_{j\in\Lambda_d}\left|w_j\right| \geq \max_{i\in\overline{\Lambda}_d}|w_i|\right\}. \tag{6.120}$$

S_d is clearly a Borel measurable subset of $\mathbb{R}^d$. According to OMP, at each iteration, a comparison is made on the scalar products to choose an element from the support of x: the index of the maximum of $|W_j^d|$ is chosen for the estimated support.

Let $\mathbb{P}_d$ the probability distribution of W^d. The W_i^d can be expressed as a product of dependent Chi and Bessel random variables (see Thm. 29), and the probability of $[W^d \in S_d]$ can't be estimated directly. An asymptotic equivalent will be of great interest and in particular, we would like to know if

$$\lim_{d\to+\infty} \mathbb{P}_d(S_d) = 1 \tag{6.121}$$

for some values of α and γ parameters.

$\mathbb{P}_d$ is singular in $\mathbb{R}^d$ because $\phi^T\phi$ is a.s. of rank $m < d$ and x is sparse; the support of W^d lies in a subspace F of $\mathbb{R}^d$ of dimension $k < d$; thus, W^d has no density w.r.t. the Lebesgue measure in $\mathbb{R}^d$.

Let $\widetilde{\mathbb{P}}$ the distribution of $\widetilde{W}$ in $\mathbb{R}^\infty$ and $\widetilde{\mathbb{P}}_d = \widetilde{\mathbb{P}} \circ \Pi_d^{-1}$ where Π_d is the projection on the d first coordinates. $\circ$ indicate the pushforward probability (image) of $\widetilde{\mathbb{P}}$ by Π_d [Bil99]. $\widetilde{W}^d$ is the random vector of $\mathbb{R}^d$ formed of the d first elements of $\widetilde{W}$. Since $\widetilde{W}$ is a Gaussian independent sequence, $\widetilde{\mathbb{P}}_d$ is a product of d Gaussian density functions, absolutely continuous w.r.t. the Lebesgue measure in $\mathbb{R}^d$ and it is clear that $\left(\widetilde{W}^d\right)_d \rightsquigarrow \widetilde{W}$ in $\mathbb{R}^\infty$.

From Thm. 31 we know that

$$\lim_{d\to+\infty} \widetilde{\mathbb{P}}_d(S_d) = 1. \tag{6.122}$$

The Gaussian approximation consists in proving that

$$\lim_{d\to+\infty}\left|\mathbb{P}_d(S_d)-\widetilde{\mathbb{P}}_d(S_d)\right|=0. \tag{6.123}$$

In words, we hope that the asymptotic probability of success is the same whether it is calculated by the initial dependent Chi-Bessel distribution of W^d or by the independent Gaussian asymptotic distribution. If so, it proves that $\mathbb{P}_d(S_d)$ tends to 1 when d tends to infinity. The idea is to work in $\mathbb{R}^\infty$ and to consider a kind of "limit" of the events S_d. It would be an expression of the form

$$\left[\max_{j\in\Lambda}\left|\widetilde{W}_j\right|\geq\max_{i\in\overline{\Lambda}}\left|\widetilde{W}_i\right|\right]. \tag{6.124}$$

But this expression does not makes sense, since the maximum of Gaussian random variables tend almost surely to infinity. Instead, let us consider

$$S=\liminf_d S_d=\bigcup_{d\geq1}\bigcap_{k\geq d}S_k=\left\{w\in\mathbb{R}^\infty:\exists\, d \text{ s.t. } w\in S_k,\ \forall k\geq d\right\}. \tag{6.125}$$

S is the set of all infinite sequence w that belong to S_d from a given index d. It means that there exists a index d above which the maximum of the (absolute value of the) sequence stays in the right support. The limit inf always exists and defines a borel set of $\mathbb{R}^\infty$, so $\mathbb{P}(S)$ is defined whatever the probability $\mathbb{P}$ on $\mathbb{R}^\infty$.

Theorem 34 (Gaussian approximation)**.**

$$\lim_{d\to+\infty}\left|\mathbb{P}_d(S_d)-\widetilde{\mathbb{P}}_d(S_d)\right|=\lim_{d\to+\infty}\left|\mathbb{P}\left[W^d\in S\right]-\mathbb{P}\left[\widetilde{W}^d\in S\right]\right|=0. \tag{6.126}$$

Proof. We work on the probability space $(\mathbb{R}^\infty,\mathfrak{B},\mathbb{P})$ where $\mathfrak{B}$ is the Borel σ-algebra of $\mathbb{R}^\infty$ and $\mathbb{P}$ is a probability measure on $\mathbb{R}^\infty$. For example, we can choose $\mathbb{P}=\widetilde{\mathbb{P}}$. Recall that

$$\widetilde{\mathbb{P}}=\prod_{i\in\Lambda}\nu_1\prod_{i\in\overline{\Lambda}}\nu_2 \tag{6.127}$$

where $\nu_1\sim\mathcal{N}(0,\sigma_1^2)$ and $\nu_2\sim\mathcal{N}(0,\sigma_2^2)$. We first note that:

$$\left[W^d\in S_d\right]\subset\left[W^d\in S\right] \tag{6.128}$$

Indeed, as $\Lambda_d\subset\Lambda$, $\overline{\Lambda}_d\subset\overline{\Lambda}$ and $W_i^d=0$ if $i>d$, if the maximum of W_j^d is attained in Λ, it is necessarily in Λ_d :

$$\max_{i\in\Lambda_d}W_i^d>\max_{j\in\overline{\Lambda}_d}W_j^d\Rightarrow\max_{i\in\Lambda}W_i^d>\max_{j\in\overline{\Lambda}}W_j^d \tag{6.129}$$

The reciprocal is also true, so that

$$\left[W^d\in S_d\right]=\left[W^d\in S\right] \tag{6.130}$$

Similarly,

$$\left[\widetilde{W}^d \in S_d\right] = \left[\widetilde{W}^d \in S\right] \tag{6.131}$$

Now, $\mathbb{P}(\partial S) = 0$ because S is a countable union and intersection of strict hyperplanes, each of probability 0 under $\widetilde{\mathbb{P}}$. As $\left(W^d\right)_d \rightsquigarrow \widetilde{W}$, by the properties of convergence in distribution [Bil99] and from Thm. 31,

$$\widetilde{\mathbb{P}}_d(S_d) = \mathbb{P}\left[\widetilde{W}^d \in S_d\right] = \mathbb{P}\left[\widetilde{W}^d \in S\right] \underset{n\to+\infty}{\longrightarrow} \mathbb{P}\left[\widetilde{W} \in S\right] = 1, \tag{6.132}$$

which we deduce

$$\mathbb{P}_d(S_d) = \mathbb{P}\left[W^d \in S_d\right] = \mathbb{P}\left[W^d \in S\right] \underset{d\to+\infty}{\longrightarrow} \mathbb{P}[\widetilde{W} \in S] = 1. \tag{6.133}$$

This completes the proof. □

For the sake of clarity, let us specify the events $[W^d \in S_d]$ as a function of the empirical measure. Let us consider a deterministic vector $w \in \mathbb{R}^d$, and the two subvectors $(w_i)_{i\in\Lambda_d}$ and $(w_i)_{i\in\overline{\Lambda}_d}$. Let us consider their (deterministic) empirical distributions:

$$\mu_d = \frac{1}{k}\sum_{i\in\Lambda_d} \delta_{w_i} \text{ and } \overline{\mu}_d = \frac{1}{d-k}\sum_{i\in\overline{\Lambda}_d} \delta_{w_i}. \tag{6.134}$$

This defines two empirical cumulative distribution functions F_d^w and $\overline{F}_d^w$ (for the sake of readibility, we omit the index w whenever possible, but it is important to keep in mind the fact that F_d is a function of w):

$$F_d(t) = \frac{1}{k}\sum_{i\in\Lambda_d} \mathbb{1}_{]-\infty,t]}(w_i) \text{ and } \overline{F}_d(t) = \frac{1}{d-k}\sum_{i\in\overline{\Lambda}_d} \mathbb{1}_{]-\infty,t]}(w_i). \tag{6.135}$$

Let Q_d and $\overline{Q}_d$ the two corresponding quantile functions (generalized inverse of F_d and $\overline{F}_d$). We can express the event S_d using these functions, by noting that

$$Q_d(1) = \max_{i\in\Lambda_d} |w_i|. \tag{6.136}$$

so that

$$S_d = \left\{w \in \mathbb{R}^\infty : F_d^w \circ \overline{Q}_d^w(1) < 1\right\} = \left[F_d \circ \overline{Q}_d(1) < 1\right]. \tag{6.137}$$

For all real sequences w, even unbounded ones, $F_d^w \circ \overline{Q}_d^w(1)$ is bounded between 0 and 1. The limit inf always exists and the event S_d occurs if and only if $F_d^w \circ \overline{Q}_d^w(1)$ is strictly below 1.

So far, we have found the limit, but in order to use the equivalent for $\mathbb{P}(S_d)$ we have to specify the speed of convergence between the two sequences of events and prove it is at least $O(1/n^\mu)$.

Let $u_d(w) = \mathbb{1}_{S_d}(w)$, a function from $\mathbb{R}^\infty$ onto $\{0,1\}$. The sequence $\left(u_d(\widetilde{W}^d)\right)_d$ is a i.i.d. Bernoulli sequence.

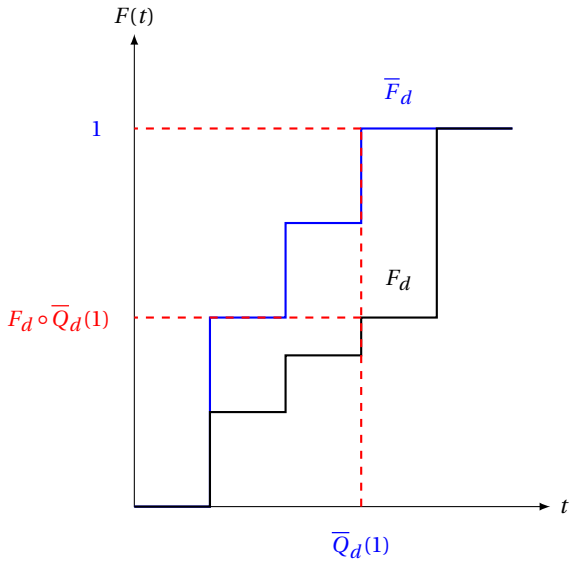

Figure 6.6 – Visualization of the value $F_d \circ \overline{Q}_d(1)$ from the curve of F_d and $\overline{F_d}$. $w \in S_d$ if and only if $F_d \circ \overline{Q}_d(1) < 1$. This means that $\overline{\mu}_d$ attains its maximum before μ_d on the x-axis.

$$\lim_d \mathbb{P}\left[W^d \in S_d\right] = \lim_d \mathbb{E}\left[\mathbb{1}_{S_d}(W^d)\right] = \liminf_d \mathbb{E}\left[\mathbb{1}_{S_d}(W^d)\right] \tag{6.138}$$

$$= \mathbb{E}\left[\liminf_d \mathbb{1}_{S_d}(W^d)\right] \tag{6.139}$$

$$= \mathbb{E}\left[\mathbb{1}_{\liminf_d [W^d \in S_d]}\right] \tag{6.140}$$

$$= 1 \tag{6.141}$$

The first line (6.138) express the fact that the lim inf is equal to the limit, since this limit exists and is equal to 1. (6.139) is a consequence of Beppo-Levi Lemma and (6.140) a property of the limit inf of a sequence of events. Since the liminf of a sequence of events is a tail event, from the Kolmogorov $0-1$ law [Fel71], the limit of the sequence $\mathbb{1}_{S_d}(W^d)$ is an almost sure constant random variable, equal to 1. This proves that the limit in distribution is also an almost sure and L^1 limit.

By independence of the sequence of events $\left(\left[W^d \in S_d\right]\right)_d$, we can apply the Borel $0-1$ law [Fel71]. Since

$$\mathbb{P}\left(\limsup_d \left[W^d \notin S_d\right]\right) = 0 \tag{6.142}$$

then

$$\sum_{d=1}^{+\infty} \mathbb{P}\left[W^d \notin S_d\right] < \infty \tag{6.143}$$

So

$$\mathbb{P}\left[W^d \notin S_d\right] = o\left(\frac{1}{d}\right). \tag{6.144}$$

This proves that the convergence of $\mathbb{P}[W^d \in S_d]$ toward 1 has a rate of at least $o(1/d)$. However, this result is not sufficient to establish a convergence rate of $O(1/d^\mu)$.

6.4.2 Approximation at iteration $t > 1$

The previous calculation, valid for the first iteration, can be adapted for the following: let $S_d(t)$ denote the event "an atom from $k-t$ atoms of the correct support is selected among the $d-k$ atoms". Given that t atoms have been selected from the correct support, $S_d(t)$ measures the success at the $t+1$-th iteration (assuming we do not account for the dependence between iterations). The evaluation remains the same for all iterations $t \geq 0$ such that $\tau < 1$ and we just have to replace the power $k-1$ by $k-t-1$ in (6.65). In fact, in the LSR, we can categorize the iterations into three classes: initial iterations, where $t = o(k)$ (or $\tau = o(1)$), intermediate iterations where $0 < \tau < 1$ and final iterations where $\tau \sim 1$. The approximation is valid for the initial and intermediate iterations.

6.4.3 Putting all the iterations together

Let OMP be carried out under the assumptions of the finite-size model in Def. 5. OMP succeeds if, and only if the event $S_d(1) \cap S_d(2) \cap \ldots \cap S_d(k)$ occurs. The events $S_d(t)$ are not independent because the selection at each iteration involves all the atoms. For instance, if atom ϕ_1 is selected at iteration 1, then, given that event, all atoms become dependent in the subsequent iterations, and we can no longer assert that the distributions of the scalar products follow Bessel-Chi distributions. Therefore, our results for iterations $t > 1$ are no longer valid.

In any case, it is legitimate to ask to what extent the probability of success

$$\mathbb{P}\left(\tilde{S}_d(1) \cap \tilde{S}_d(2) \cap \ldots \cap \tilde{S}_d(k)\right) \tag{6.145}$$

calculated based on the approximations made at a given iteration is close, when d is high, to the true probability taking into account the dependencies. This is our conjecture. Simulations must be conducted to validate this conjecture.

We propose some avenues for future research in the final chapter to continue investigations into this problem.

6.5 Conclusion

In this chapter, we laid the foundation for a statistical analysis of the OMP algorithm within an information-theoretic framework, where both the signal to be recovered and the observation matrix are random and their sizes tend to infinity. This provides an average-case analysis, distinct from traditional uniform or non-uniform performance frameworks.

We examined two models - finite-size and asymptotic - and derived for both the joint densities of all statistics involved in the recovery process. We proved that the probability of recovery at a given iteration tends to 1 as the dimension d of the signal tends to infinity, under condition of linear sparsity and overmeasure rate.

Additionally, we propose useful theoretical results concerning Gaussian-related distributions.

While we did not succeed in analyzing the entire OMP algorithm due to stochastic dependencies between iterations, we conjecture that the asymptotic probability of success can be approximated by neglecting these dependencies. Further investigations in this direction are necessary.

CONCLUSIONS AND PERSPECTIVES

Conclusions

In this thesis, we addressed three aspects of the dimensionality reduction problem. Each aspect focuses on reducing a size or dimension parameter while aiming to preserve a relevant quantity of interest: sampling a database represented by a datamatrix while preserving the underlying information of the data, reducing the number of edges from a graph while preserving its connectivity and studying the performance of a sparse signal reconstruction algorithm.

In the first part, we studied an instance of column subset selection problem through volume maximization. By constraining the matrix to have normalized columns, we have shown that the continuous relaxation of this problem is equivalent to spectrum equalization and maximization of the smallest eigenvalue. We proposed several algorithms to solve the CSSP: a new implementation of the greedy algorithm RECTMAXVOL, making it slightly less complex than the original greedy approach, a new greedy algorithm with quadratic complexity achieving performance close to that of the cubic algorithm and three variants of an algorithm that solve the problem in a single step, providing performance comparable to that of greedy algorithms. Eventually, we proposed two direct applications of these algorithms: matrix conditioning and compressive sensing.

In the second part of the thesis, we proposed two greedy algorithms to sparsify a graph while preserving its connectivity. These algorithms emerged from a geometric interpretation of the volume of the graph Laplacian matrix and modify the sprectum of the initial graph to optimize its robustness. Both algorithms are deterministic, adapted to unweighted graphs and of reasonable quadratic complexity in time. We provided a detailed implementation of several variants and empirically demonstrated that they perform better than state-of-the-art sparsification algorithms for a cubic complexity in time.

We proposed an application to GNNs that simplifies the architecture and accelerates the processes executed on the network, without significantly degrading performance. Our method seems to be an effective way to limit the over-squashing phenomenon, but only for homophilic graphs where the neighbors of a node bring meaningful information.

In the final part of the thesis, we addressed a problem in compressive sensing: reconstructing the support of a sparse signal using a small number of linear measurements. Our objective was to conduct a statistical analysis of some existing sparse recovery methods, including the Orthogonal Matching Pursuit (OMP) algorithm. We aimed to analyze the performance of OMP within an information-theoretic framework, where both the signal to be recovered and the sensing matrix were random, and their respective dimension tended to infinity. Traditional performance analysis framework focus on uniform or non-uniform recovery. Our approach differed by providing an average-cage analysis, allowing both x and ϕ to be random. The first model considered had fixed d, m, k linked by linear constraints: $k = \gamma d$ and $m = \mu k$, where γ was the sparsity rate and μ the overmeasure rate. The second model, called the Large System Regime (LSR) was asymptotic, with d tending to infinity while maintaining linear constraints between parameters.

We derived the joint densities of all statistics involved in the recovery process for both the finite-size model and the asymptotic one. In the LSR, the distribution of the scalars product followed a independent Gaussian sequence. This enabled us to prove the convergence of the asymptotic probability of recovering a correct atom from the initial signal to 1, under the sole condition of linear sparsity and overmeasure rate. This limit was also valid for any given iteration $t > 1$, provided that the algorithm does not select a wrong atom before that iteration. Evaluating the probability of success of the algorithm over all iterations requires accounting for the stochastic dependencies between iterations, which we have not yet addressed. However, we conjectured that an approximation of the probability of success in the entire OMP process could be obtained by considering the probabilities at each iteration, while neglecting these dependencies.

During the analysis we obtained intermediate results that could be useful for research related to compressive sensing: technical lemmas concerning projections and products of Gaussian distributions, scalar products following Bessel-Chi distributions and their Gaussian limits in the asymptotic regime, and an asymptotic equivalence for a Gaussian integral representing the probability of an event involving the comparison of the maximum of two Gaussian vectors.

Perspectives

In this paragraph, we propose perspectives for each of our three problems, which sometimes overlap.
Regarding the first part, several perspectives deserve to be explored:

- In light of the analogies drawn between the first two parts of the thesis, it would be worthwhile to explore the potential for initializing the algorithms using structures similar to spanning trees, aiming for an initialization with near-linear complexity.
- Recommender systems use weights to quantify user preferences. In this context, it would be interesting to explore algorithms that enable weighted selection of columns, similar to methods used in weighted graph sparsification.

Several possible lines of research emerge from the work in the second part, regarding graph sparsification and preservation of the connectivity, as developed in this chapter:

- We can initialize a square Laplacian submatrix with a deterministic algorithm, in a nearly linear time complexity in the parameters dimension; deterministic approaches are always preferable to random algorithms, due to their reproducibility and consistent performance. Achieving linear time complexity would be particularly valuable for large graphs, as the learning process must be repeated for each graph. Building on the principle of the WaterMaxVol algorithms from Chap. 4, it might be worthwhile to define a deterministic one-shot algorithm by selecting all edges at once, in decreasing order of their effective resistance, based on their correlation with the second smallest eigenvalue. Theoretically, this algorithm should exhibit nearly linear complexity and be deterministic.
- The twice Ramanujan sparsifier of Batson and al. shares some similarities with our greedy algorithms: both are deterministic and select edges at each iteration relative to their effective resistance. However Batson and al.'s algorithm has a five degree polynomial complexity, while ours is only quadratic. It would be interesting to verify whether our edge selection method is compatible with the conditions of Batson's algorithm, potentially providing an instance where the algorithm would exhibit quadratic complexity.

- The sparsification operation can be viewed as a method for partitioning the nodes of a graph, and can therefore be used as a clustering algorithm. It would be interesting to study the performance of clustering algorithms based on our two connectivity maximization algorithms and compare their effectiveness with other spectral clustering algorithms.

In the final part of the thesis, we were not able to fully achieve the objective we initially set for ourselves. Consequently, we propose various ideas and insights for completing the analysis of OMP, as perspectives regarding this part.

- The first step is to more precisely evaluate the convergence rate of the success probability towards the value calculated using the Gaussian approximation. We have been able to estimate this convergence rate as at least $o(1/n)$. If the rate of convergence is at least $o(1/n^{\mu})$, then:
  - — The equivalence derived from the Gaussian integral would also served as an asymptotic equivalent for the success of OMP at a given iteration.
  - — A union bound argument could then demonstrate that the entire OMP process succeeds with probability tending to 1, except for the very last iterations.

  To prove this result, two approaches should be considered:
  - — By leveraging the partial exchangeability of the vector W and the finite version of de Finetti's theorem [Ald83], we can express the success probability as an average sum formed from the empirical measures of W, as considered by Lindenstrauss [Lin01] or Austern and Orbanz [AO22]. Ergodic theory results may then be applied to estimate the convergence rate.
  - — Another approach is to explore the property of chaos in Boltzmann's statistical mechanics and mean-field theory [Gra08; Rou14; HM14]. The vector W represents the state of a symmetric system of dependent but exchangeable particles, whose number tends to infinity. The system exhibits Kac-type chaos where, as d tends to infinity, the particles become stochastically independent. In such a scenario (which applies to our settings) bounds on the convergence rate are available.
- If x becomes deterministic, can we derive the distribution of the underlying scalar products and deduce uniform bounds on the success probability?
- Starting from Tropp's ERC (Exact Recovery Condition) of success for OMP, Tropp and Gilbert [Tro04; TG07] demonstrated the convergence of OMP in an asymptotic framework, without needing to analyze each iteration individually. Could our method be applied in this case?
- OLS and OMP differ only by a normalization of the vectors implied [Sou+13]. What distributions do we obtain for OLS under our assumptions ?
- Can we establish the success probability for very simple algorithms, like Hard Thresholding, in a one-shot analysis?

APPENDIX A

TOOLS FROM LINEAR ALGEBRA

A.1 Tools from the rank-one perturbation theory

Rank-one perturbation theory studies the effect, on the spectrum, of adding a rank-one linear operator to an initial matrix [GV96; IN09].

Here, Σ is a symmetric square definite positive matrix of size $d \times d$ and $b \in \mathbb{R}^d$ is a column vector. We want to study the modification on the spectrum $\lambda_1 > \lambda_2 > ... > \lambda_d > 0$ of Σ when we add the rank-one operator $bb^T \in \mathbb{R}^{d\times d}$.

Theorem 35 (Effect of a rank-one operator on the sprectum of a symmetric matrix)**.**

$$\lambda_i(\Sigma + bb^T) = \lambda_i + \mu_i,\ i = 1, ..., d, \tag{A.1}$$

with $\lambda_i = \lambda_i(\Sigma)$, $0 \le \mu_i \le 1$ *and* $\sum_{i=1}^d \mu_i = 1$.

Proof. See [Wil88]. □

The d eigenvalues of bb^T are $||b||^2 = 1$, with multiplicity 1 and 0, with multiplicity $d-1$.

The first step is to deflate the problem, that is, to simplify the problem by considering three particular cases as described in [BNS78]. Let Q be an orthogonal matrix such that $\Sigma = QDQ^T$, with $QQ^T = Q^TQ = I_d$ and $D = \text{diag}(\lambda_1, ..., \lambda_d)$. Without loss of generality, up to a change of basis, we can work in the eigenbasis formed by the columns of Q. For the sake of simplicity, we keep the same notation b for the selected column instead of Q^Tb. If $b_i = 0$ for some i, then $\lambda_i(\Sigma + bb^T) = \lambda_i$ and the corresponding eigenvector remains unchanged. If $b_i = \pm 1$ for some i, then $\lambda_i(\Sigma + bb^T) = \lambda_i + ||b||^2 = \lambda_i + 1$. This case occurs if and only if b is an eigenvector corresponding to λ_i. The third case arises when Σ has multiple eigenvalues; let λ an eigenvalue of multiplicity $r \ge 2$ and $Q_1 \in \mathbb{R}^{d\times r}$ whose columns span the corresponding eigenspace. By partitioning Q (up to a permutation), we can decompose D such that

$$D + bb^T = \begin{pmatrix} D_1 & 0 \\ 0 & D_2 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & b_2 b_2^T \end{pmatrix}. \tag{A.2}$$

Eventually, this simplification can be performed for every multiple eigenvalue, as detailed in [BNS78].

From now on, let us assume that no deflation is possible: all eigenvalues are simple and $0 < b_i < 1$ for all $i = 1, ..., d$. In that case, the eigenvalues of $\Sigma + bb^T$ are the solutions of the so-called "secular equation" [Gol73]:

$$1 + \sum_{i=1}^{d} \frac{b_i^2}{\lambda_i - \lambda} = 0, \tag{A.3}$$

and the new normalized eigenvectors v_i (corresponding to $\lambda_i(\Sigma + bb^T)$) are given by

$$v_i = \frac{Q^T D_i^{-1} b}{||D_i^{-1} b||_2} \tag{A.4}$$

where $D_i = D - \lambda_i(\Sigma + bb^T) \times I_d$.

The coordinates of v_i are proportional to $b_k/(\lambda_k - \lambda_i(\Sigma + bb^T))$; if the eigenvalues of Σ are poorly separated, the denominator $\lambda_k - \lambda_i(\Sigma + bb^T)$ can be very small, in which case the direction of the eigenvectors can change abruptly.

Our algorithms rely on maximizing the smallest eigenvalues of the initial matrix by selecting the column(s) b that contribute most to its increase. This contribution is exactly μ_i for the eigenvalue λ_i. Therefore, we are interested in evaluating μ_i and understanding its relationship with b, with the spectrum and the eigenvectors of Σ.

Benasséni [BM19] provides the following equality:

$$\mu_i = \frac{(b^T u_i)(b^T v_i)}{u_i^T v_i} \tag{A.5}$$

when $u_i^T v_i \neq 0$, where u_i and v_i are the normalized eigenvectors associated with $\lambda_i(\Sigma)$ and $\lambda_i(\Sigma + bb^T)$, respectively.

If b is expressed in the eigenvectors basis of Σ, then $b^T u_i = b_i$, meaning this term represents the i-th coordinate of b. It is also the coefficient in the numerator of the secular equation (A.3) in the term $b_i/(\lambda_i - \lambda)$. However, since v_i is not known before selecting b, (A.5) does not help to evalute μ_i.

To bound μ_i, we can use Thm. 35 and Cauchy's interlacing property [GV96; HJ85]:

$$\lambda_1(\Sigma + bb^T) \geq \lambda_1(\Sigma) \geq ... \geq \lambda_d(\Sigma + bb^T) \geq \lambda_d(\Sigma) > 0. \tag{A.6}$$

This gives rise to the following family of inequalities:

$$\begin{cases} \lambda_i(\Sigma) \leq \lambda_i(\Sigma + bb^T) \leq \lambda_{i-1}(\Sigma), & i = 2, .., d \\ \\ \lambda_i(\Sigma)) \leq \lambda_i(\Sigma + bb^T) \leq \lambda_i(\Sigma) + ||b||^2, & i = 1, .., d \end{cases} \tag{A.7}$$

There also exists several families of inequalities establishing relationships between μ_i, $\mathrm{Sp}(\Sigma)$ and u_i, with many depending on the gap between consecutive eigenvalues. Let

$$\delta_i = \lambda_{i-1} - \lambda_i, \ i = 2, .., d. \tag{A.8}$$

In the two-sided inequality families bounding $\lambda_i(\Sigma + bb^T)$ [IN09; Ben90], the formulas differ for each eigenvalue. For the sake of brevity, and because we are mainly concerned with the smallest eigenvalue, we present only the inequalities for λ_d.

Theorem 36 (Inequalities of Ipsen and Nadler)**.** *There exists $L_+(d)$ and $U_+(d)$ such that*

$$\lambda_d \leq L_+(d) \leq \lambda_d(\Sigma + bb^T) \leq U_+(d) \leq \lambda_{d-1} \tag{A.9}$$

$$\text{with} \begin{cases} L_+(d) & = \lambda_d + \frac{1}{2}\left(\beta - \sqrt{\beta^2 - 4\gamma}\right) \\ \beta & = \delta_d + ||Q^T b||^2 = \delta_d + 1 \\ \gamma & = \delta_d (u_d^T b)^2 \end{cases}$$

$$\text{and} \begin{cases} U_+(d) & = \lambda_d + \frac{1}{2}\left(\beta' - \sqrt{\beta'^2 - 4\gamma}\right) \\ \beta' & = \delta_d + (u_{d-1}^T b)^2 + (u_d^T b)^2 \\ \gamma & = \delta_d (u_d^T b)^2 \end{cases}$$

Proof. See [IN09] □

Theorem 37.

$$(u_d^T b)^2 \frac{\delta_d}{\delta_d + 1} \leq \lambda_d(\Sigma + bb^T) - \lambda_d \leq |u_d^T b| \sqrt{\delta_d} \tag{A.10}$$

Proof. The double inequality is a corollary of the previous theorem. For the proof of the right-hand side inequality, see [IN09]. Let's prove the left-hand side inequality:

$$\frac{\beta - \sqrt{\beta^2 - 4\gamma}}{2} = \frac{2\gamma}{\beta + \sqrt{\beta^2 - 4\gamma}} \geq \frac{\gamma}{\beta} = \frac{\delta_d (u_d^T b)^2}{\delta_d + 1} \tag{A.11}$$

□

We can derive another useful inequality: from the min-max Courant–Fischer theorem,

$$\lambda_d(\Sigma + bb^T) = v_d^T(\Sigma + bb^T) v_d \leq u_d^T(\Sigma + bb^T) u_d \tag{A.12}$$

$$= u_d^T \Sigma u_d + (u_d^T b)^2 = \lambda_d + (u_d^T b)^2. \tag{A.13}$$

Combining this inequality with the LHS of (A.10) and (A.7) gives:

$$(u_d^T b)^2 \frac{\delta_d}{\delta_d + 1} \leq \lambda_d(\Sigma + bb^T) - \lambda_d \leq \min((u_d^T b)^2, \delta_d) \tag{A.14}$$

In light of these formulas, two main parameters control the increase of the smallest eigenvalue (and this is true for all eigenvalues) consecutive to the concatenation of a column:

- the angle $(u_i^T b)^2$ between the corresponding eigenvector and the column (or the coordinate b_d if working in the eigenbasis).
- the gap δ_d between the smallest eigenvalue and the next one.

When the eigenvalues are close to one another (as occurs in low dimensions, for example), the increase is impaired by δ_d and can be small, even if b is strongly correlated with the eigenvector u_d.

A.2 Effective resistance and bi-harmonic distances on a graph

The effective resistance allows the definition of two Euclidean distances on the nodes of a graph and provides an efficient method for embedding the graph nodes into an Euclidean space.

The geodesic distance is probably the most popular graph distance, representing the length of the shortest path between two vertices. While intuitive and useful, it is a local concept that does not account for the entire geometry of the graph. This distance is also sensitive to small topological perturbations [LRF10] and is difficult to evaluate.

The Euclidean commute-time distance (ECTD) is simply defined for any pair of vertices v_i, v_j by their effective resistance:

$$d(v_i, v_j)^2 = R_{ij}. \tag{A.15}$$

This is indeed a distance: let $u_1, ..., u_n$ an orthonormal eigenbasis for L (with, as usual, $u_1 = (1, ..., 1)^T$ being the eigenvector associated with $\lambda_1 = 0$) and let $U = (u_1, ..., u_n) \in \mathbb{R}^{n\times n}$. Then, for all $i = 1, ..., n$,

$$v_i = U^T \delta_i. \tag{A.16}$$

Given $L = U\Lambda U^T$,we define for all $i = 1, ..., n$,

$$y_i = \Lambda^{+/2} v_i = \Lambda^{+/2} U^T \delta_i, \tag{A.17}$$

where $\Lambda^{+/2} = \left(\Lambda^+\right)^{1/2}$. Then the vectors y_i and y_j in $\mathbb{R}^n$ satisfy

$$\begin{aligned}
\left|\left|y_i - y_j\right|\right|_2^2 &= (y_i - y_j)^T (y_i - y_j) && \text{(A.18)}\\
&= (v_i - v_j)^T \Lambda^+ (v_i - v_j) && \text{(A.19)}\\
&= (v_i - v_j)^T U^T L^+ U (v_i - v_j) && \text{(A.20)}\\
&= \delta_{ij}^T L^+ \delta_{ij} && \text{(A.21)}\\
&= R_{ij}, && \text{(A.22)}
\end{aligned}$$

which prove that the $(y_i)_i$ form an embedding of the vertex space in $\mathbb{R}^n$ and that the resistance distance is the Euclidean distance between y_i and y_j.

This calculation also partially proves that L^+ is a kernel (positive semidefinite) matrix and the ECTD is the Mahalanobis distance associated with this matrix. To complete the proof, we need to show that the y_i's are centred:

$$\sum_{i=1}^{n} y_i = \Lambda^{1/2} U^T \sum_{i=1}^{n} \delta_i = \Lambda^{1/2} U^T u_1 = 0, \tag{A.23}$$

since u_1 is the eigenvector associated with the zero eigenvalue. If we define $Y = (y_1, ..., y_n) \in \mathbb{R}^{n\times n}$, then $Y = U\Lambda^{1/2}$ and $Y^T Y = \Lambda$. Since this is a diagonal matrix, the eigenvectors of λ are the basis vectors y_i, forming the principal components coordinate system. y_{ij} is the projection of the node v_i on the j-th principal component and its value is λ_j. This projection is similar to a PCA in the Euclidean space where the nodes are embedded.

This decomposition defines the projection of nodes with maximum variance among all projections.

Another useful distance associated with resistance distance is the biharmonic distance, first proposed by Lipman [WLY22; LRF10]. While the effective resistance measures how well two vertices are connected within the graph, incorporating both local topology and global geometry, the biharmonic distance offers a balanced measure between local and global influences. It is more sensitive to the global geometry of the graph and provides a quantitative measure of the change in effective resistance when an edge is added, removed or its conductance modified [LRF10; Bla+24].

Definition 7. *The biharmonic distance between two vertices v_i, v_j of G is given by*

$$d_B(v_i,v_j)^2 = (\delta_i - \delta_j)^T L^{2+}(\delta_i - \delta_j) = \sum_{l=2}^{n} \frac{\left(u_l(i) - u_l(j)\right)^2}{\lambda_l^2} = \left|\left|L^+ \delta_{ij}\right|\right|^2, \tag{A.24}$$

where $L^{2+} = \left(L^+\right)^2$.

The biharmonic distance is related to graph connectivity [WLY22]:

$$d_B(v_i,v_j)^2 = \frac{1}{n}\frac{\tau(G/e)^2}{\tau(G)^2}. \tag{A.25}$$

On a global scale, the biharmonic index is the sum of all biharmonic distance between all pairs of vertices:

$$\beta(G) = n\sum_{i=1}^{n}\frac{1}{\lambda_k^2}. \tag{A.26}$$

For an edge $e = (v_i, v_j)$, the biharmonic distance is related to the variation of conductance [GBS08; Bla+24]:

$$\frac{\partial R(G)}{\partial w_e} = -n d_B(v_i,v_j)^2. \tag{A.27}$$

The decrease in effective resistance between G and $\overline{G} = G \cup \{e\}$ is given by [GBS08; Bla+24]:

$$R(G) - R(\overline{G}) = n\frac{d_B(v_i,v_j)^2}{1+R_{ij}}. \tag{A.28}$$

Thus, the gain in terms of effective resistance is proportional to the biharmonic distance.

APPENDIX B

MATHEMATICAL TOOLS FOR COMPRESSIVE SENSING

B.1 Bessel, Pearson and other symmetrical distributions

In this section, we summarize classical results about the probability distributions used in the third part of this thesis. The Gaussian assumption on x and B naturally lead to Bessel distributions (as products of independent Gaussians), chi or chi-square distributions (which arise from the norm of Gaussian vectors), Beta or Pearson type II distributions (linked to Gaussian vectors projections) and uniform distribution on the sphere (connected to the the polar decomposition of a Gaussian vector). Many of these distributions exhibit spherical or elliptical symmetry, so we recall some useful properties of this family. The main reference on the subject is the book of Fang, Kotz and Ng [FKN90].

B.1.1 Bessel distribution $\mathfrak{B}$

The univariate case

The Bessel distribution is known by several names in the literature: Variance Gamma, generalized Cauchy, Bessel among others. The following definitions clarify the reasons for these different names.

A random variable V follows a $\Gamma(\alpha,\beta)$ distribution if its density function is given by

$$v(x) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x} \mathbb{1}_{\mathbb{R}_+}(x). \tag{B.1}$$

The characteristic function of a Gamma-distributed random variable is given by

$$\phi(u) = \left(1 - \frac{iu}{\beta}\right)^{-\alpha}, \tag{B.2}$$

where $\beta > 0$ is a intensity parameter and $\alpha > 0$ a shape parameter, often representing a number of random variables or a number of coordinates.

Now, consider a random Gaussian $X \sim \mathcal{N}(0,1)$ and a random variable $V \sim \Gamma(\alpha,\beta)$ independent of X. The random variable

$$Y = \mu + \theta V + \sigma\sqrt{V}X \tag{B.3}$$

follows a Variance Gamma distribution denoted $\mathfrak{B}(\alpha,\beta,\mu,\theta,\sigma)$. Y is a Gaussian variable whose random variance and mean follow a Gamma distribution (hence the name). More precisely, given the event $[V=v]$, Y is a Gaussian variable with mean $\mu+\theta v$ and variance $\sigma^2 v$.

μ is a location parameter, θ is an asymmetry parameter and σ^2 is a scale parameter for the variance.

The density function can be derived by integrating the joint density of X and V or by identifying it from the characteristic function. Using one of these methods, we obtain

$$f_Y(x)=\sqrt{\frac{2}{\pi}}\frac{\beta^\alpha}{\sigma\Gamma(\alpha)}\exp\left(\frac{x-\mu}{\sigma^2}\theta\right)\times\left(\frac{|x-\mu|}{\sqrt{\theta^2+2\beta\sigma^2}}\right)^{\alpha-1/2}\times K_{\alpha-1/2}\left(\frac{|x-\mu|}{\sigma^2}\sqrt{\theta^2+2\beta\sigma^2}\right). \tag{B.4}$$

K_ν is the McDonald, also known as the modified Bessel function of the second kind and is given by the following integral representation (for example):

$$K_\nu(x)=\frac{1}{2}\int_0^{+\infty}z^{\nu-1}\exp\left(-\frac{x}{2}(z+\frac{1}{z})\right)dz. \tag{B.5}$$

The characteristic function of Y is

$$\phi(u)=e^{i\mu u}\left(1-\frac{i\theta u}{\beta}+\frac{\sigma^2u^2}{2\beta}\right)^{-\alpha}. \tag{B.6}$$

The moments of Y are evaluated by conditionning on V or by differentiating the characteristic function. Using the fact that $\mathbb{E}[V]=\alpha/\beta$ and $\mathrm{var}V=\alpha/\beta^2$, we deduce the following expressions for the mean and variance of Y:

$$\mathbb{E}[Y]=\mu+\theta\frac{\alpha}{\beta}, \tag{B.7}$$

$$\mathrm{var}Y=\frac{\alpha}{\beta}\sigma^2\left(1+\frac{\theta^2}{\beta\sigma^2}\right). \tag{B.8}$$

The family of $\mathfrak{B}$ distributions is stable under convolution. Specifically, if $Y_1\sim\mathfrak{B}(\alpha_1,\beta,\theta,\mu_1,\sigma^2)$ and $Y_2\sim\mathfrak{B}(\alpha_2,\beta,\theta,\mu_2,\sigma^2)$ are two independent $\mathfrak{B}$ random variables, their sum Y_1+Y_2 follows a $\mathfrak{B}(\alpha_1+\alpha_2,\beta,\theta,\mu_1+\mu_2,\sigma^2)$ distribution. Hence, the α coefficient is acts like a size parameter. Additionally, the Bessel family is stable under addition or multiplicative by a constant.

We focus on the standard case of $\mathfrak{B}$ random variables denoted $\mathfrak{B}(n,\sigma)$, where $\theta=0,\alpha=n/2,\beta=1/2,\mu=0$. In this case, the density of $Y=\sqrt{V}X$ simplifies to

$$f_Y(x)=\left(\sigma\sqrt{\pi}\times\Gamma(\frac{n}{2})\times 2^{\frac{n-1}{2}}\right)^{-1}\left(\frac{|x|}{\sigma}\right)^{\frac{n-1}{2}}K_{\frac{n-1}{2}}\left(\frac{|x|}{\sigma}\right) \tag{B.9}$$

and its characteristic function is given by

$$\phi(u)=\left(1+\sigma^2u^2\right)^{-n/2}. \tag{B.10}$$

The standard case corresponds to a zero mean variable with no asymmetry.

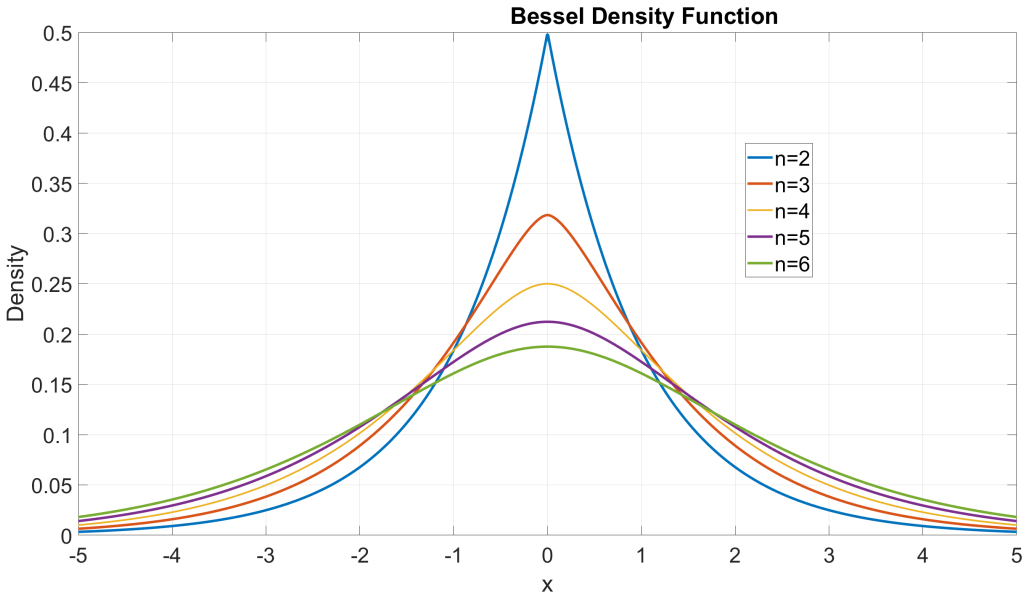

Figure B.1 – Density function of a standard Bessel distribution for $\sigma = 1$ and $n = 2,3,4,5,6$.

The multivariate case

There are several generalizations of the $\mathfrak{B}$ distribution in a multivariate framework. We adopt the definition from Madan and Seneta [MS90], where $Y \in \mathbb{R}^m$ is a random Gaussian vector whose variance is of the form ΣV, with V being a random Gamma variable and $\Sigma \in \mathbb{R}^{m\times m}$ a covariance matrix. Given V, the Y vector follows a multivariate Gaussian distribution and the variance of each coordinate is proportional to the same (random) factor V. In the standard case (where α is later replaced by $n/2$), the characteristic function of Y is given by

$$\phi_Y(u) = \left(1 + u'\Sigma u\right)^{-\alpha}, \tag{B.11}$$

where $u \in \mathbb{R}^m$. The density fonction is therefore

$$f_Y(x) = \left[2^{\alpha-1}(2\pi)^{m/2}\Gamma(\alpha)\sqrt{\det\Sigma}\right]^{-1}\left(\sqrt{x'\Sigma^{-1}x}\right)^{\alpha-\frac{m}{2}} K_{\alpha-\frac{m}{2}}\left(\sqrt{x'\Sigma^{-1}x}\right), \tag{B.12}$$

with $x \in \mathbb{R}^m$. If $\Sigma = \sigma^2 \times Id$,

$$\phi_Y(u) = \left[1 + \sigma^2||u||^2\right]^{-\alpha} \tag{B.13}$$

and

$$f_Y(x) = \left[2^{\alpha-1}(2\pi)^{m/2}\Gamma(\alpha)\sigma^m\right]^{-1}\left(\frac{||x||}{\sigma}\right)^{\alpha-\frac{m}{2}} K_{\alpha-\frac{m}{2}}\left(\frac{||x||}{\sigma}\right). \tag{B.14}$$

The distribution is then spherically symmetric. We will denote it as $\mathfrak{B}_m(n,\sigma)$ with $\alpha = n/2$.

Product of independent Gaussian random variables

The product of two independent real random Gaussian variables $X \sim \mathcal{N}(0,\sigma_X^2)$ and $Y \sim \mathcal{N}(0,\sigma_Y^2)$ is a distribution known as the normal product distribution. Its characteristic function is given by

$$\phi(t) = \frac{1}{\sqrt{1+\sigma_X^2\sigma_Y^2 t^2}}. \tag{B.15}$$

A direct calculation [KKP01] yields the density function:

$$f(u) = \frac{1}{\pi\sigma_X\sigma_Y} K_0\left(\frac{|u|}{\sigma_X\sigma_Y}\right) \tag{B.16}$$

where K_0 is the second kind modified Bessel function of the second kind, defined by

$$K_0(x) = \int_0^{+\infty} \cos(x\sinh t)\,dt = \int_0^{+\infty} \frac{\cos(xt)}{\sqrt{t^2+1}}\,dt. \tag{B.17}$$

It is a particular case of $\mathfrak{B}$ distribution, corresponding to $n = 1$. The Cauchy distribution is a particular case of the $\mathfrak{B}$ distribution, corresponding to $n = 2$. The Bessel distribution is centred (when the Gaussians are centered), supported on $\mathbb{R}$, and is an even function.

Theorem 38 (**Product of two centered Gaussians).** *Let $X \sim \mathcal{N}(0,\sigma_X^2)$ and $Y \sim \mathcal{N}(0,\sigma_Y^2)$ be two independent centered Gaussian random variables.*
Then XY follows a $\mathfrak{B}(1,\sigma_X\sigma_Y)$ standard Bessel distribution.

Distribution of the dot product of two independent Gaussian vectors

The following result is a corollary of the previous discussion. Consider two independent Gaussian vectors $X \sim \mathcal{N}(0,\sigma_X^2 I_n)$ and $Y \sim \mathcal{N}(0,\sigma_Y^2 I_n)$.

We are interested in the distribution of their scalar product $Z =< X,Y >= X^T Y$. Previous results indicate that Z follows a $\mathfrak{B}$ distribution; Since each term $X_i Y_i$ is the product of two independent real Gaussians and thus follows a Bessel distribution, the independence of the $X_i Y_i$'s and the stability of this distribution under convolution ensure that $Z = X_1 Y_1 + ... + X_n Y_n$ follows a standard $\mathfrak{B}(n,\sigma_X\sigma_Y)$ distribution. The characteristic function is given by

$$\phi_Z(u) = \left[1+(\sigma_X\sigma_Y u)^2\right]^{-n/2}. \tag{B.18}$$

Theorem 39 (**Dot product of two centered Gaussians vectors).** *Let $X \sim \mathcal{N}(0,\sigma_X^2 I_n)$ and $Y \sim \mathcal{N}(0,\sigma_Y^2 I_n)$ be two independent Gaussian random vectors.*
Then $\langle X,Y\rangle = X^T Y$ follows a $\mathfrak{B}(n/2,\sigma_X\sigma_Y)$ standard Bessel distribution.

B.1.2 Uniform distribution $\mathfrak{U}$ on the n dimensional sphere

The n-dimensional unit sphere is the set denoted $\mathbb{S}^n$ of vectors from $\mathbb{R}^n$ whose Euclidean norm is 1. We sometimes omit n if there is no ambiguity.

$$\mathbb{S}^n = \{x \in \mathbb{R}^n : \ ||x|| = 1\}, \tag{B.19}$$

where $||x||$ denotes the Euclidean norm. $\mathbb{S}^n$ is of dimension $n-1$ and its Lebesgue measure in $\mathbb{R}^n$ is zero. However, the density with respect to the Lebesgue measure on $\mathbb{S}^n$ is

$$f(x) = \frac{\Gamma(n/2)}{2\pi^{n/2}} \mathbb{1}_{\mathbb{S}^n}(x). \tag{B.20}$$

The characteristic function for $t \in \mathbb{R}^n$ is given by

$$\phi(t) = 2^{n/2-1} \times \Gamma(n/2) \times ||t||^{1-n/2} \times J_{n/2-1}(||t||), \tag{B.21}$$

where J_ν is the Bessel function of the first kind given by

$$J_\nu(x) = \sum_{i=0}^{+\infty} \frac{(-1)^i}{i\,! \times \Gamma(i+\nu+1)} \left(\frac{x}{2}\right)^{2i+\nu}. \tag{B.22}$$

We denote such a distribution by $\mathfrak{U}_{\mathbb{S}^n}$ or simply $\mathfrak{U}$. If $U \sim \mathfrak{U}$, then $\mathbb{E}[U] = 0$.

The distribution U is spherical symmetric distribution. Indeed, a classical method to generate a uniform distribution on the sphere is to consider a standard Gaussian vector X and define $U = X/||X||$.

Properties of spherical distributions

We denote $O(n)$ as the orthogonal group of order n. $O(n)$ is the set of orthogonal matrices of size $n \times n$. A matrix Γ is orthogonal if $\Gamma^{-1} = \Gamma^T$, which is equivalent to the condition that all column vectors are orthogonal and normalized to 1. Such a matrix represents an orthonormal basis of $\mathbb{R}^n$ and preserves the norm of vectors; thus, $O(n)$ is indeed the group of vectorial isometries of $\mathbb{R}^n$.

We write $X \sim Y$ when random variables X and Y have the same probability distribution. $X_n \rightsquigarrow X$ is the traditional notation to indicate that the sequence $(X_n)_n$ converges in distribution toward X, when n tends to infinity [Vaa00] .

Definition 8 (Theorem 2.5. from [FKN90])**.** *$X = (X_1, ..., X_n)^T \in \mathbb{R}^n$ is spherical symmetric if and only if one of the following equivalent conditions is satisfied:*

- $\forall \Gamma \in O(n), X \sim \Gamma X,$ (B.23)
- *the density function of X is $g\left(||x||^2\right)$* $\forall x \in \mathbb{R}^n,$ (B.24)
- $\phi_X(t) = \mathbb{E}\left[e^{i<t,X>}\right] = h\left(||t||\right)$ $\forall t \in \mathbb{R}^n,$ (B.25)
- $X = R \times U$ *with* $R = ||X||$ *and* $U = X \div ||X|| \sim \mathfrak{U},$ (B.26)
- $\forall a \in \mathbb{R}^n, < a, X > \sim ||a|| \times X_1.$ (B.27)

A spherical symmetric distribution is invariant under all orthogonal transformation. In particular, it is centered, exchangeable and rotation invariant. The two following points ensure that X is spherical symmetric if

and only if its characteristic function depends solely on the norm of X. The function h characterizes this dependence and is called the generative function.

The fourth point demonstrates that spherical symmetric density functions are those where, in the polar decomposition, the non radial component is uniformly distributed on the sphere. Moreover, one has $R \perp\!\!\!\perp U$. The last point proves that the scalar product of a spherical vector by any other non zero vector of $\mathbb{R}^n$ is a real random variable whose distribution is proportional to any coordinate of X.

Theorem 40 (Chapter 2 of [FKN90])**.** *Let $X = (X_1, ..., X_n)^T \in \mathbb{R}^n$ a spherical vector. Then,*

- *the coordinates of X are independent if and only if $X \sim \mathcal{N}(0, \sigma^2 \times I_n)$.*
- *$cov(X_i, X_j) = 0 \ \forall i \neq j$.*
- *The marginals and conditional densities of X are spherical.*

The moments of a spherical distribution do not depend on the radial distribution of R:

$$\mathbb{E}[X] = 0, \tag{B.28}$$

$$\text{cov}(X) = \text{cov}(R \times U) = \frac{1}{n}\mathbb{E}[R^2] \times I_n. \tag{B.29}$$

The radial component R is sufficient to characterize the distribution: if $X = RU$ and $Y = R'U'$ are the polar distributions of two spherical vectors, then $X \sim Y$ implies $R \sim R'$. Eventually, if the density of $X = RU$ is of the form $g(||x||^2)$, then the density of R is given by

$$f(r) = \frac{2\pi^{n/2}}{\Gamma(n/2)} r^{n-1} g(r^2) \times \mathbb{1}_{\mathbb{R}^+}(r). \tag{B.30}$$

The spherical symmetry generalizes to elliptical symmetry: X is elliptical if $X = A^T Y$, where Y is a spherical random variable and $A \in \mathbb{R}^{k \times n}$ is a matrix such that $A^T A = \Sigma$ of rank k. In the formulas, $||x||^2$ is replaced by $x^T \Sigma x$.

B.1.3 Pearson law of type II

Distribution of the angle between two uniform vectors on the sphere

We saw that $Y \in \mathbb{R}^m$ is spherical symmetric if and only if $Y = R \times U$ where $R \geq 0$ a.s. and U is uniformly distributed on the sphere, with R independent of U. Thus, $R = ||Y||$ and $U = Y/||Y||$. Dividing two spherical vectors by their respective norm yields two vectors U and V that are uniformly distributed on the unit sphrere.

When U and V are independent, the distribution of the scalar product $\rho =< U, V >$ is supported on the interval $]-1, 1[$ and follows a Pearson type II distribution, whose density function is

$$g(z) = \frac{1}{\sqrt{\pi}} \frac{\Gamma(m/2)}{\Gamma((m-1)/2)} (1 - z^2)^{(m-3)/2} \mathbb{1}_{]-1,1[}(z). \tag{B.31}$$

To obtain this result, note that the surface of the sphere of radius r in $\mathbb{R}^m$ is given by

$$S = \frac{2\pi^{m/2}}{\Gamma(m/2)} r^{m-1}. \tag{B.32}$$

The subset of $\mathbb{S}^m$ defined by $R_z = \{v \in \mathbb{S}^m : \ v_m \leq z\}$ has measure

$$\lambda(R_z) = \frac{2\pi^{m/2}}{\Gamma(m/2)} \int_{-1}^{z} (1-x^2)^{(m-2)/2} dx. \tag{B.33}$$

Furthermore, since U and V are spherically symmetric, we can always assume - by performing a rotation if necessary - that $< U, V >= V_m$, by choosing as the last basis vector a unit vector in the same direction as V. If U and V are independent, this rotation Γ depends on V, but not on U, so $\Gamma U \sim U$. The distribution function of $\rho =< U, V >$ is then given by

$$F(z) = \frac{\lambda(R_z)}{S}. \tag{B.34}$$

By differentiating the integral, we find the expression for the Pearson type II distribution density mentioned earlier.

If ρ follows a Pearson distribution of dimension m, then $\mathbb{E}[\rho] = 0$ and $\text{var}(\rho) = \frac{1}{m+1}$

ρ represents the distribution of the linear correlation coefficient between two uniform random variables. It is also the law of the cosine of the angle between two uniform vectors or two Gaussian vectors. As the dimension increases to infinity, vectors chosen uniformly in space tend to become orthogonal.

• Joint law of angles between Gaussian or uniform vectors

When we consider an n-dimensional normal sample $X = (X_1, ..., X_n)$ and normalize it by dividing each X_i by its norm, we form a vector $U = (U_1, ..., U_n)$ whose elements are uniformly distributed on the unit sphere. The matrix of empirical correlation coefficients of X or U is then given by

$$R = U^T \times U = \begin{pmatrix} 1 & \rho_{12} & ... & \rho_{1n} \\ \rho_{21} & 1 & ... & \rho_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ \rho_{n1} & \rho_{n2} & ... & 1 \end{pmatrix}. \tag{B.35}$$

This matrix is symmetric and the $n(n-1)/2$ elements ρ_{ij} that compose it follow type II Pearson distributions. These random variables are pairwise independent, but not mutually independent. Their joint density is given by

$$h(R) = \frac{\Gamma(m/2)^n}{\Gamma_n(m/2)} (\det R)^{\frac{m-n-1}{2}}, \tag{B.36}$$

$$\Gamma_n(x) = \pi^{n(n-1)/4} \Gamma(x)\Gamma(x-1/2)...\Gamma(x-(n-1)/2). \tag{B.37}$$

We find the joint law by performing a change of variables in the density of a Wishart distribution. The

distribution of correlations is similar to the Wishart distribution, but instead of covariances between variables, it yields correlations.

B.1.4 Distribution of χ^2 and χ

• Density and characteristics

A random variable X follows a chi-square distribution with m degrees of freedom if its probability density function is

$$f(x) = \frac{2^{-m/2}}{\Gamma(m/2)} x^{m/2-1} e^{-x/2} \mathbb{1}_{[0,+\infty[}(x). \tag{B.38}$$

The mean of the chi-square distribution is $\mathbb{E}[X] = m$ and its variance is $\text{var}(X) = 2m$. This distribution corresponds to the the square of the Euclidean norm of a standard Gaussian vector with m dimension.

The χ distribution is the distribution of the square root of a variable following a chi-square distribution. It represents the norm of a standard Gaussian vector whose m coordinates are independent. Its density function is given by

$$g(x) = \frac{2^{1-m/2}}{\Gamma(m/2)} x^{m-1} e^{-x^2/2} \mathbb{1}_{[0,+\infty[}(x), \tag{B.39}$$

where Γ is the Euler Gamma function defined by

$$\Gamma(x) = \int_0^{+\infty} t^{x-1} e^{-t} dt. \tag{B.40}$$

When $m = 1$, the chi distribution corresponds to the absolute value of a standard normal distribution $\mathcal{N}(0,1)$, which is also known as the half-normal distribution. When $m = 2$, it corresponds to a Rayleigh distribution and when $m = 3$, to the Maxwell-Boltzmann distribution.

B.1.5 Beta distribution β

The Beta distribution's density function (defined on $[0,1]$) is given by

$$f(x) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \times x^{\alpha-1}(1-x)^{\beta-1} \mathbb{1}_{[0,1]}(x). \tag{B.41}$$

If $X \sim \beta$, then

$$\mathbb{E}[X] = \frac{\alpha}{\alpha+\beta}, \tag{B.42}$$

and

$$\text{var}X = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}. \tag{B.43}$$

Theorem 41. *If $x \sim \mathcal{N}(0, I_m)$, $u \sim \mathfrak{U}_m$ and $x \perp\!\!\!\perp u$, then $< \frac{x}{||x||}, u >^2$ follows a Beta distribution and is independent of x.*

The square of a Pearson distribution follows a Beta distribution $\beta(1/2, (m-1)/2)$.

B.2 Mellin transform

The idea behind the Mellin transform is to define a tool analog to the Fourier transform stable for product of probabilistic distributions. For a complete review on the subject, we refer the reader to [Obe74].

Definition 9. *The Mellin transform of a random variable X of density function f is, if any, the function denoted $\widehat{f}$ and defined for $s \in \mathbb{C}$ by:*

$$\widehat{f}(s) = \mathbb{E}[X^{s-1}] = \int_0^{+\infty} f(x) x^{s-1} dx. \tag{B.44}$$

If $f(x)x^{s-1}$ is integrable in $]0, +\infty[$, $\widehat{f}$ is well-defined and analytic in a vertical strip in the complex plane.

Examples:

- If $f(x) = e^{-x}\mathbb{1}_{[0,+\infty[}(x)$, $\widehat{f}(s) = \Gamma(s)$.
- If $f(x) = \mathbb{1}_{[0,1]}(x)$, $\widehat{f}(s) = 1/s$ and is defined on $\mathbb{R}e(s) > 0$.
- Let

$$f(s) = \frac{x^\alpha}{\Gamma(\alpha+1)} e^{-x} \mathbb{1}_{[0,+\infty[}(x). \tag{B.45}$$

If $\mathbb{R}e(s) > -\alpha$,

$$\widehat{f}(s) = \frac{\Gamma(\alpha+s)}{\Gamma(\alpha+1)}. \tag{B.46}$$

- The Mellin transform of the modified Bessel function of second kind $K_\nu(x)$ is given by the following formula:

$$\widehat{K}_\nu(s) = 2^{s-2}\Gamma\left(\frac{s-\nu}{2}\right)\Gamma\left(\frac{s+\nu}{2}\right). \tag{B.47}$$

The inverse Mellin transform allows us to recover f from $\widehat{f}$.

Theorem 42.

$$f(x) = \frac{1}{2\pi i}\int_{c-i\infty}^{c+i\infty} \widehat{f}(s)\frac{ds}{x^s}, \tag{B.48}$$

for all $x \in \mathbb{R}$, f continuous and for a path of integration parallel to the imaginary axis and included in the domain of analycity of $\widehat{f}$.

The Mellin transform is linear and satisfies the following properties:

Theorem 43. *Let f with a Mellin transform $\widehat{f}(s)$.*

- *If $g(x) = x^\alpha f(x)$ then $\widehat{g}(s) = \widehat{f}(s+\alpha)$*
- *If $g(x) = f(x^\alpha)$ then $\widehat{g}(s) = \frac{1}{|\alpha|}\widehat{f}(\frac{s}{\alpha})$*
- *If $f \star g$, defined by*

$$f \star g(x) = \int_0^{+\infty} f\left(\frac{x}{u}\right) g(u) \frac{du}{u}, \tag{B.49}$$

has a Mellin transform, then

$$\widehat{f \star g}(s) = \widehat{f}(s) \times \widehat{g}(s). \tag{B.50}$$

This last property is a tool to determine the distribution of a product of two random variables X and Y. If f and g are the corresponding density functions, the product XY has a density function $f \star g$. It should be noted that this convolution product is not the classical one, but a product in the sense of Mellin.

Mellin transform of Bessel function of the first kinf J_ν

Bessel functions appear as solutions of the Bessel differential equation. The simpliest defintion is given by the power serie:

$$J_\nu(x) = \sum_{k=0}^{+\infty} \frac{(-1)^k}{k\,! \times \Gamma(\nu+k+1)} \left(\frac{x}{2}\right)^{\nu+2k}. \tag{B.51}$$

This formula is valid $\forall \nu \in \mathbb{R}$. When $\nu = n \in \mathbb{Z}$,

$$J_{-n}(x) = (-1)^n J_n(x), \tag{B.52}$$

and for $n \in \mathbb{Z}$,

$$J_n(x) = \frac{1}{\pi}\int_0^\pi \cos(x\sin\theta - n\theta)\,d\theta = \frac{1}{2\pi}\int_{-\pi}^{\pi} e^{-i(n\theta - x\sin\theta)}\,d\theta. \tag{B.53}$$

If $n = 0$,

$$J_0(x) = \frac{1}{2\pi}\int_0^{2\pi} \cos(x\sin\theta)\,d\theta. \tag{B.54}$$

If

$$f(x) = \frac{1}{\pi}\frac{1}{\sqrt{a^2 - x^2}}, \tag{B.55}$$

then the Fourier transform is given by:

$$\phi(t) = \frac{1}{\pi}\int_{-\infty}^{+\infty} e^{itx}\frac{dx}{\sqrt{a^2-x^2}} = \frac{1}{\pi}\int_{-\pi/2}^{+\pi/2} e^{iat\cos\theta}\,d\theta = J_0(at) \tag{B.56}$$

and the Mellin transform of J_0 is given for $0 < \mathbb{R}e(s) < 1$ by

$$\widehat{J}_0(s) = \frac{2^{s-1}}{\pi} \sin\left(\frac{\pi s}{2}\right) \Gamma\left(\frac{s}{2}\right)^2. \tag{B.57}$$

Mellin transform of the modified Bessel function of the second kind K_ν

Let

$$I_\nu(x) = \sum_{k=0}^{+\infty} \frac{1}{k\,! \times \Gamma(\nu + k + 1)} \left(\frac{x}{2}\right)^{\nu+2k} \tag{B.58}$$

and

$$K_\nu(x) = \frac{\pi}{2} \frac{I_{-\nu}(x) - I_\nu(x)}{\sin \pi\nu}. \tag{B.59}$$

K_ν has a integral representation:

$$\begin{aligned} K_\nu(x) &= \frac{1}{2} \int_0^{+\infty} z^{\nu-1} \exp\left(-\frac{x}{2}(z + \frac{1}{z})\right) dz \\ &= 2 \int_1^{+\infty} \cosh(\nu t) e^{-x \cosh t}\, dt \\ &= \frac{x^\nu}{2^{\nu+1}} \int_0^{+\infty} t^{-\nu-1} e^{-t - x^2/(4t)}\, dt \\ &= \frac{\sqrt{\pi}}{2^\nu \times \Gamma(\nu + 1/2)} x^\nu \int_1^{+\infty} (t^2 - 1)^{\nu - 1/2} e^{-xt}\, dt. \end{aligned} \tag{B.60}$$

In particular,

$$K_0(x) = \int_0^{+\infty} \frac{\cos(tx)}{\sqrt{t^2 + 1}}\, dt. \tag{B.61}$$

One can prove [Gla+12] that:

$$\int_1^{+\infty} (t^2 - 1)^{\nu - 1/2} e^{xt}\, dt = \frac{1}{\sqrt{\pi}} \left(\frac{2}{x}\right)^\nu \Gamma\left(\nu + \frac{1}{2}\right) K_\nu(x). \tag{B.62}$$

If the Fourier transform of f is given by

$$\phi(t) = \frac{1}{\sqrt{1 + t^2}}, \tag{B.63}$$

then

$$f(x) = \frac{1}{\pi} K_0(|x|). \tag{B.64}$$

And the Mellin transform of $K_\nu(x)$ is

$$\widehat{K}_\nu(s) = \int_0^{+\infty} x^{s-1} K_\nu(x)\, dx = 2^{s-2}\Gamma\left(\frac{s-\nu}{2}\right)\Gamma\left(\frac{s+\nu}{2}\right). \tag{B.65}$$

BIBLIOGRAPHY

[AB13] Haim Avron and Christos Boutsidis, « Faster Subset Selection for Matrices and Applications », *in*: *SIAM Journal on Matrix Analysis and Applications* 34.*4* (2013), pp. 1464–1499, URL: https://doi.org/10.1137/120867287.

[Abu+19] Firas Abuzaid et al., « To Index or Not to Index: Optimizing Exact Maximum Inner Product Search », *in*: Apr. 2019, pp. 1250–1261, DOI: 10.1109/ICDE.2019.00114.

[AE12] Waseem Abbas and Magnus Egerstedt, « Robust Graph Topologies for Networked Systems », *in*: *IFAC Proceedings Volumes* 45.*26* (2012), 3rd IFAC Workshop on Distributed Estimation and Control in Networked Systems, pp. 85–90, ISSN: 1474-6670, DOI: https://doi.org/10.3182/20120914-2-US-4030.00052.

[Ald83] David J. Aldous, « Exchangeability and related topics », *in*: *École d'été de probabilités de Saint-Flour, XIII—1983*, vol. 1117, Lecture Notes in Math. Berlin: Springer, 1983, pp. 1–198.

[AO22] Morgane Austern and Peter Orbanz, « Limit theorems for distributions invariant under groups of transformations », *in*: *The Annals of Statistics* 50.*4* (2022), pp. 1960–1991, DOI: 10.1214/21-AOS2165.

[Arn+22] Adrian Arnaiz-Rodriguez et al., *DiffWire: Inductive Graph Rewiring via the Lovász Bound*, 2022, arXiv: 2206.07369 [cs.LG].

[AY20] Uri Alon and Eran Yahav, « On the Bottleneck of Graph Neural Networks and its Practical Implications », *in*: *CoRR* abs/2006.05205 (2020), arXiv: 2006.05205, URL: https://arxiv.org/abs/2006.05205.

[Ban+22] Pradeep Kr. Banerjee et al., « Oversquashing in GNNs through the lens of information contraction and graph expansion », *in*: *2022 58th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, Monticello, IL, USA: IEEE Press, 2022, pp. 1–8, DOI: 10.1109/Allerton49937.2022.9929363.

[Bar+20] Pablo Barceló et al., « The Logical Expressiveness of Graph Neural Networks », *in*: *International Conference on Learning Representations*, 2020, URL: https://openreview.net/forum?id=r1lZ7AEKvB.

[Bas92] Hyman Bass, « THE IHARA-SELBERG ZETA FUNCTION OF A TREE LATTICE », *in*: *International Journal of Mathematics* 03 (1992), pp. 717–797.

[Bat+13] Joshua Batson et al., « Spectral sparsification of graphs: theory and algorithms », *in*: *Commun. ACM* 56.*8* (2013), pp. 87–94, ISSN: 0001-0782, DOI: 10.1145/2492007.2492029.

[BBC20] Ayoub Belhadji, Rémi Bardenet, and Pierre Chainais, « A determinantal point process for column subset selection », *in*: *Journal of Machine Learning Research* 21.*197* (2020), pp. 1–62, URL: http://jmlr.org/papers/v21/19-080.html.

[Ben90] Jacques Benasseni, « A note on bounds to the variation of eigenvalues in symmetric matrix perturbation of rank one », *in*: *Linear and Multilinear Algebra* 27.*2* (1990), pp. 111–116, DOI: 10.1080/03081089008818000.

[Ben92] Adi Ben-Israel, « A volume associated with m x n matrices », *in*: *Linear Algebra and its Applications* 167 (Apr. 1992), pp. 87–111, DOI: 10.1016/0024-3795(92)90340-G.

[Ber99] D.P. Bertsekas, *Nonlinear Programming*, Athena Scientific, 1999.

[BF05] Ulrik Brandes and Daniel Fleischer, « Centrality Measures Based on Current Flow », *in*: *STACS 2005*, ed. by Volker Diekert and Bruno Durand, Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 533–544, ISBN: 978-3-540-31856-9.

[BF13] Enrico Bozzo and Massimo Franceschet, « Resistance distance, closeness, and betweenness », *in*: *Social Networks* 35.*3* (2013), pp. 460–469, ISSN: 0378-8733, DOI: https://doi.org/10.1016/j.socnet.2013.05.003.

[BH09] John S. Baras and Pedram Hovareshti, « Efficient and robust communication topologies for distributed decision making in networked systems », *in*: *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, 2009, pp. 3751–3756, DOI: 10.1109/CDC.2009.5400448.

[Big93] Norman Biggs, *Algebraic Graph Theory*, Second, Cambridge: Cambridge University Press, 1993, ISBN: 0521458978.

[Bil99] Patrick Billingsley, *Convergence of probability measures*, Second, Wiley Series in Probability and Statistics: Probability and Statistics, A Wiley-Interscience Publication, New York: John Wiley & Sons Inc., 1999, pp. x+277, ISBN: 0-471-19745-9.

[BK96] András A. Benczúr and David R. Karger, « Approximating s-t minimum cuts in Õ(n2) time », *in*: *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '96, Philadelphia, Pennsylvania, USA: Association for Computing Machinery, 1996, pp. 47–55, ISBN: 0897917855, DOI: 10.1145/237814.237827.

[Bla+23] Mitchell Black et al., « Understanding oversquashing in GNNs through the lens of effective resistance », *in*: *Proceedings of the 40th International Conference on Machine Learning*, ICML'23, , Honolulu, Hawaii, USA, JMLR.org, 2023.

[Bla+24] Mitchell Black et al., « Biharmonic Distance of Graphs and its Higher-Order Variants: Theoretical Properties with Applications to Centrality and Clustering. », *in*: *CoRR* abs/2406.07574 (2024), URL: http://dblp.uni-trier.de/db/journals/corr/corr2406.html#abs-2406-07574.

[BM19] Jacques Bénasséni and Alain Mom, « Inequalities for the eigenvectors associated to extremal eigenvalues in rank one perturbations of symmetric matrices », *in*: *Linear Algebra and its Applications* 570 (2019), pp. 123–137, DOI: https://doi.org/10.1016/j.laa.2019.01.021.

[BNS78] J.R. Bunch, C.P. Nielsen, and D.C. Sorensen, « Rank-One Modification of the Symmetric Eigenproblem. », *in*: *Numerische Mathematik* 31 (1978), pp. 31–48, URL: http://eudml.org/doc/132565.

[BSS12] Joshua Batson, Daniel A. Spielman, and Nikhil Srivastava, « Twice-Ramanujan Sparsifiers », *in*: *SIAM Journal on Computing* 41.*6* (2012), pp. 1704–1721, DOI: 10.1137/090772873.

[BT97] Dimitris Bertsimas and John N. Tsitsiklis, *Introduction to linear optimization*, Athena scientific series in optimization and neural computation 6, Athena Scientific, 1997.

[CA16] Hau Chan and Leman Akoglu, « Optimizing network robustness by edge rewiring: a general framework », *in*: *Data Mining and Knowledge Discovery* 30 (2016), pp. 1395–1425, URL: https://api.semanticscholar.org/CorpusID:14653006.

[CD11] Romain Couillet and Mérouane Debbah, *Random Matrix Methods for Wireless Communications*, Cambridge University Press, 2011.

[CDD09] Albert Cohen, Wolfgang Dahmen, and Ronald Devore, « Compressed sensing and best k-term approximation », *in*: *J. Amer. Math. Soc* (2009), pp. 211–231.

[Cel+18] Elisa Celis et al., « Fair and Diverse DPP-Based Data Summarization », *in*: *Proceedings of the 35th International Conference on Machine Learning*, ed. by Jennifer Dy and Andreas Krause, vol. 80, Proceedings of Machine Learning Research, PMLR, 2018, pp. 716–725, URL: https://proceedings.mlr.press/v80/celis18a.html.

[CH11] Marcos Manzano Castro and David Harle, « Metrics to Evaluate Network Robustness in Telecommunication Networks », *in*: 2011, URL: https://api.semanticscholar.org/CorpusID:16293304.

[Cha+22] Pak Hay Chan et al., « Network Design for s-t Effective Resistance », *in*: *ACM Trans. Algorithms* 18.*3* (2022), ISSN: 1549-6325, DOI: 10.1145/3522588.

[Cha+89] A. K. Chandra et al., « The electrical resistance of a graph captures its commute and cover times », *in*: *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*, STOC '89, Seattle, Washington, USA: Association for Computing Machinery, 1989, pp. 574–586, ISBN: 0897913078, DOI: 10.1145/73007.73062.

[Cha00] Bernard Chazelle, « A minimum spanning tree algorithm with inverse-ackermann type complexity. Journal of the ACM 47(6), 1028-1047 », *in*: *J. ACM* 47 (Nov. 2000), pp. 1028–1047, DOI: 10.1145/355541.355562.

[Che+23] Yuhan Chen et al., « Demystifying Graph Sparsification Algorithms in Graph Properties Preservation », *in*: *Proc. VLDB Endow.* 17.*3* (2023), pp. 427–440, ISSN: 2150-8097, DOI: 10.14778/3632093.3632106.

[CK20] Alice Cortinovis and Daniel Kressner, « Low-Rank Approximation in the Frobenius Norm by Column and Row Subset Selection », *in*: *SIAM Journal on Matrix Analysis and Applications* 41.*4* (2020), pp. 1651–1673, DOI: 10.1137/19M1281848.

[CKM20] Alice Cortinovis, Daniel Kressner, and Stefano Massei, « On maximum volume submatrices and cross approximation for symmetric semidefinite and diagonally dominant matrices », *in*: *Linear Algebra And Its Applications* 593 (2020), pp. 251–268, DOI: 10.1016/j.laa.2020.02.010, URL: http://infoscience.epfl.ch/record/276910.

[ÇM07] Ali Çivril and Malik Magdon-Ismail, « Finding maximum Volume sub-matrices of a matrix », *in*: *Transactions on Computational Science* (May 2007).

[ÇM09] Ali Çivril and Malik Magdon-Ismail, « On selecting a maximum volume sub-matrix of a matrix and related problems », *in*: *Theoretical Computer Science* 410.*47* (2009), pp. 4801–4811, ISSN: 0304-3975, URL: https://www.sciencedirect.com/science/article/pii/S0304397509004101.

[CT05] E.J. Candes and T. Tao, « Decoding by linear programming », *in*: *IEEE Transactions on Information Theory* 51.*12* (2005), pp. 4203–4215, DOI: 10.1109/TIT.2005.858979.

[CT06a] Peter G. Casazza and Janet Crandell Tremain, « The Kadison–Singer Problem in mathematics and engineering », *in*: *Proceedings of the National Academy of Sciences* 103.*7* (2006), pp. 2032–2039, DOI: 10.1073/pnas.0507888103.

[CT06b] Thomas M. Cover and Joy A. Thomas, *Elements of Information Theory 2nd Edition (Wiley Series in Telecommunications and Signal Processing)*, Wiley-Interscience, 2006.

[CW20] Chen Cai and Yusu Wang, « A Note on Over-Smoothing for Graph Neural Networks », *in*: *ArXiv* abs/2006.13318 (2020), URL: https://api.semanticscholar.org/CorpusID:220042028.

[Dem97] James W. Demmel, *Applied Numerical Linear Algebra*, SIAM, 1997.

[Des+06] Amit Deshpande et al., « Matrix Approximation and Projective Clustering via Volume Sampling », *in*: *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithm*, SODA '06, Miami, Florida: Society for Industrial and Applied Mathematics, 2006, pp. 1117–1126, ISBN: 0898716055.

[Di +23] Francesco Di Giovanni et al., « On over-squashing in message passing neural networks: the impact of width, depth, and topology », *in*: *Proceedings of the 40th International Conference on Machine Learning*, Honolulu,Hawaii,USA: JMLR.org, 2023.

[Die80] Jean Dieudonné, *Calcul infinitésimal*, Hermann, 1980.

[DKM06a] Petros Drineas, Ravi Kannan, and Michael W. Mahoney, « Fast Monte Carlo Algorithms for Matrices I: Approximating Matrix Multiplication », *in*: *SIAM Journal on Computing* 36.*1* (2006), pp. 132–157, DOI: 10.1137/S0097539704442684.

[DKM06b] Petros Drineas, Ravi Kannan, and Michael W. Mahoney, « Fast Monte Carlo Algorithms for Matrices II: Computing a Low-Rank Approximation to a Matrix », *in*: *SIAM Journal on Computing* 36.*1* (2006), pp. 158–183, DOI: 10.1137/S0097539704442696.

[DKM06c] Petros Drineas, Ravi Kannan, and Michael W. Mahoney, « Fast Monte Carlo Algorithms for Matrices III: Computing a Compressed Approximate Matrix Decomposition », *in*: *SIAM Journal on Computing* 36.*1* (2006), pp. 184–206, DOI: 10.1137/S0097539704442702.

[DLV22] Andreea Deac, Marc Lackenby, and Petar Veličković, « Expander Graph Propagation », *in*: *Proceedings of the First Learning on Graphs Conference*, ed. by Bastian Rieck and Razvan Pascanu, vol. 198, Proceedings of Machine Learning Research, PMLR, 2022, 38:1–38:18, URL: https://proceedings.mlr.press/v198/deac22a.html.

[DR07] F.R. De Hoog and Mattheij R.M.M, « Subset selection for matrices », *in*: *Linear Algebra and its Applications 422* (2007), pp. 349–359, URL: https://core.ac.uk/download/pdf/82104534.pdf.

[DS84] Peter G. Doyle and S. J. Snell, *Random Walks and Electrical Networks (Carus Mathematical Monographs, No 22)*, Mathematical Assn of America, 1984.

[DW10] Mark A. Davenport and Michael B. Wakin, « Analysis of orthogonal matching pursuit using the restricted isometry property », *in*: *IEEE Trans. Inf. Theor.* 56.*9* (2010), pp. 4395–4401, ISSN: 0018-9448, DOI: 10.1109/TIT.2010.2054653.

[Dyk71] Otto Dykstra, « The Augmentation of Experimental Data to Maximize [XtX] », *in*: *Technometrics* 13.*3* (1971), pp. 682–688, DOI: 10.1080/00401706.1971.10488830.

[EK12] Yonina C. Eldar and Gitta Kutyniok, *Compress Sensing, Theory and Applications*, Cambridge, 2012.

[EK13] Wendy Ellens and Robert E. Kooij, « Graph measures and network robustness », *in*: *ArXiv* (2013), URL: https://api.semanticscholar.org/CorpusID:14309555.

[Fel71] William Feller, *An introduction to probability theory and its applications. Vol. II.* Second edition, New York: John Wiley & Sons Inc., 1971.

[FKN90] Kai-Tai Fang, Samuel Kotz, and Kai-Wang Ng, *Symmetric Multivariate and Related Distributions*, Springer, 1990.

[Fou+07] Francois Fouss et al., « Random-Walk Computation of Similarities between Nodes of a Graph with Application to Collaborative Recommendation », *in*: *IEEE Transactions on Knowledge and Data Engineering* 19.*3* (2007), pp. 355–369, DOI: 10.1109/TKDE.2007.46.

[FR13] Simon Foucart and Holger Rauhut, *A Mathematical Introduction to Compressive Sensing*, Basel Birkhäuser, 2013.

[Fre+23] Scott Freitas et al., « Graph Vulnerability and Robustness: A Survey », *in*: *IEEE Transactions on Knowledge and Data Engineering* 35.*6* (2023), pp. 5915–5934, DOI: 10.1109/TKDE.2022.3163672.

[GBS08] Arpita Ghosh, Stephen Boyd, and Amin Saberi, « Minimizing Effective Resistance of a Graph », *in*: *SIAM Review* 50.*1* (2008), pp. 37–66, DOI: 10.1137/050645452.

[GE96] Ming Gu and Stanley C. Eisenstat, « Efficient Algorithms for Computing a Strong Rank-Revealing QR Factorization », *in*: *SIAM Journal on Scientific Computing* 17.*4* (1996), pp. 848–869, eprint: https://doi.org/10.1137/0917055.

[Git11] Alex Gittens, « The spectral norm error of the naive Nystrom extension », *in*: *ArXiv* (2011), URL: https://api.semanticscholar.org/CorpusID:14401831.

[Gla+12] Larry Glasser et al., « The integrals in Gradshteyn and Ryzhik. Part 22: Bessel-K functions. », *in*: *Sci., Ser. A, Math. Sci. (N.S.)* 22 (2012), pp. 129–151.

[Gol73] Gene H. Golub, « Some Modified Matrix Eigenvalue Problems », *in*: *SIAM Review* 15.*2* (1973), pp. 318–334, URL: https://doi.org/10.1137/1015032.

[Gor+] S. A. Goreinov et al., « How to Find a Good Submatrix », *in*: *Matrix Methods: Theory, Algorithms and Applications*, pp. 247–256, DOI: 10.1142/9789812836021_0015.

[Gra08] Carl Graham, « Chaoticity for multi-class systems and exchangeability within classes », *in*: *Journal of Applied Probability* 45.*4* (Dec. 2008), Third revision, v4. The paper is similar to the second revision v3, with several improvements. To appear in Journal of Applied Probability 45.4 (December 2008), pp. 1196–1203, DOI: 10.1239/jap/1231340243.

[GT01] S. Goreinov and E. Tyrtyshnikov, « The maximal-volume concept in approximation by low-rank matrices », *in*: *Structured Matrices in Mathematics, Computer Science, and Engineering I*, 2001, pp. 47–51, DOI: 10.1090/conm/280/4620, URL: https://app.dimensions.ai/details/publication/pub.1089204836.

[GTZ97] S.A. Goreinov, E.E. Tyrtyshnikov, and N.L. Zamarashkin, « A theory of pseudoskeleton approximations », *in*: *Linear Algebra and its Applications* 261.*1* (1997), pp. 1–21, ISSN: 0024-3795, DOI: https://doi.org/10.1016/S0024-3795(96)00301-1.

[GV96] Gene H. Golub and Charles F. Van Loan, *Matrix Computations*, Third, The Johns Hopkins University Press, 1996.

[Has90] Ki-ichiro Hashimoto, « ON ZETA AND L-FUNCTIONS OF FINITE GRAPHS », *in*: *International Journal of Mathematics* 01 (1990), pp. 381–396.

[HJ85] Roger A. Horn and Charles R. Johnson, *Matrix Analysis*, Cambridge University Press, 1985, DOI: 10.1017/CBO9780511810817.

[HM14] Maxime Hauray and Stéphane Mischler, « On Kac's Chaos and related problems », *in*: *Journal of Functional Analysis* 266.*10* (2014), pp. 6055–6157, ISSN: 0022-1236, DOI: 10.1016/j.jfa.2014.02.030.

[HP92] Y Hong and C. T. Pan, « Rank-Revealing QR Factorizations and the Singular Value Decomposition », *in*: *Mathematics of Computation* 58.*197* (1992), pp. 213–232, ISSN: 00255718, 10886842, URL: http://www.jstor.org/stable/2153029 (visited on 07/06/2024).

[HYL17] Will Hamilton, Zhitao Ying, and Jure Leskovec, « Inductive Representation Learning on Large Graphs », *in*: *Advances in Neural Information Processing Systems* (2017), pp. 1024–1034.

[Iha66] Yasutaka Ihara, « On discrete subgroups of the two by two projective linear group over $\mathfrak{p}$ -adic fields », *in*: *Journal of the Mathematical Society of Japan* 18.*3* (1966), pp. 219–235, DOI: 10.2969/jmsj/01830219.

[IN09] I. C. F. Ipsen and B. Nadler, « Refined Perturbation Bounds for Eigenvalues of Hermitian and Non-Hermitian Matrices », *in*: *SIAM Journal on Matrix Analysis and Applications* 31.*1* (2009), pp. 40–53, DOI: 10.1137/070682745.

[JL84] William Johnson and Joram Lindenstrauss, « Extensions of Lipschitz maps into a Hilbert space », *in*: *Contemporary Mathematics* 26 (Jan. 1984), pp. 189–206, DOI: 10.1090/conm/026/737400.

[JS18] Charles R. Johnson and Carlos M. Saiago, *Eigenvalues, Multiplicities and Graphs*, Cambridge Tracts in Mathematics, Cambridge University Press, 2018.

[KBM22] Kedar Karhadkar, Pradeep Kr. Banerjee, and Guido Montúfar, « FoSR: First-order spectral rewiring for addressing oversquashing in GNNs », *in*: *ArXiv* abs/2210.11790 (2022), URL: https://api.semanticscholar.org/CorpusID:253080708.

[Kir58] G. Kirchhoff, « On the Solution of the Equations Obtained from the Investigation of the Linear Distribution of Galvanic Currents », *in*: *IRE Transactions on Circuit Theory* 5.*1* (Mar. 1958), pp. 4–7, ISSN: 0096-2007, DOI: 10.1109/TCT.1958.1086426.

[KKP01] Samuel Kotz, Tomasz J. Kozubowski, and Krzysztof Podgorski, *The Laplace Distribution and Generalizations*, Springer, 2001.

[Kol09] E.D. Kolaczyk, « Statistical Analysis of Network Data: Methods and Models », *in*: *Springer Series In Statistics* (2009).

[KS96] David R. Karger and Clifford Stein, « A new approach to the minimum cut problem », *in*: *J. ACM* 43.*4* (1996), pp. 601–640, ISSN: 0004-5411, DOI: 10.1145/234533.234534.

[KSR17] Omid Keivani, Kaushik Sinha, and Parikshit Ram, « Improved maximum inner product search with better theoretical guarantees », *in*: *2017 International Joint Conference on Neural Networks (IJCNN)*, 2017, pp. 2927–2934, DOI: 10.1109/IJCNN.2017.7966218.

[KT12] Alex Kulesza and Ben Taskar, *Determinantal Point Processes for Machine Learning*, Hanover, MA, USA: Now Publishers Inc., 2012, ISBN: 1601986289.

[Kut12] Gitta Kutyniok, « Compressed Sensing: Theory and Applications », *in*: *CoRR* abs/1203.3815 (2012), URL: http://arxiv.org/abs/1203.3815.

[KW17] Thomas N. Kipf and Max Welling, « Semi-Supervised Classification with Graph Convolutional Networks », *in*: *Proceedings of the 5th International Conference on Learning Representations* (ICLR), ICLR '17, Palais des Congrès Neptune, Toulon, France, 2017, URL: https://openreview.net/forum?id=SJU4ayYgl.

[Lau20] Claire Launay, « Discrete determinantal point processes and their application to image processing », Theses, Université Paris Cité, June 2020, URL: https://theses.hal.science/tel-03189384.

[LB07] G. Travieso L. da F. Costa F. A. Rodrigues and P. R. Villas Boas, « Characterization of complex networks: A survey of measurements », *in*: *Advances in Physics* 56.*1* (2007), pp. 167–242, DOI: 10.1080/00018730601170527.

[LGD20] Claire Launay, Bruno Galerne, and Agnès Desolneux, « Exact sampling of determinantal point processes without eigendecomposition », *in*: *Journal of Applied Probability* 57.*4* (2020), pp. 1198–1221, DOI: 10.1017/jpr.2020.56.

[Li+18] Gang Li et al., « Maximizing Algebraic Connectivity via Minimum Degree and Maximum Distance », *in*: *IEEE Access* 6 (2018), pp. 41249–41255, DOI: 10.1109/ACCESS.2018.2857411.

[Lin01] Elon Lindenstrauss, « Pointwise theorems for amenable groups », *in*: *Inventiones mathematicae* 146 (2001), pp. 259–295, URL: https://doi.org/10.1007/s002220100162.

[Liu+21] Paul Liu et al., « Diversity on the Go! Streaming Determinantal Point Processes under a Maximum Induced Cardinality Objective », *in*: *Proceedings of the Web Conference 2021*, Ljubljana, Slovenia: Association for Computing Machinery, 2021, pp. 1363–1372, ISBN: 9781450383127, DOI: 10.1145/3442381.3450089.

[LJS17] Chengtao Li, Stefanie Jegelka, and Suvrit Sra, « Polynomial Time Algorithms for Dual Volume Sampling », *in*: *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, Curran Associates Inc., 2017, pp. 5045–5054, ISBN: 9781510860964.

[LRF10] Yaron Lipman, Raif Rustamov, and Thomas Funkhouser, « Biharmonic Distance », *in*: *ACM Transactions on Graphics* 29.*3* (June 2010).

[LS18] Yin Tat Lee and He Sun, « Constructing Linear-Sized Spectral Sparsification in Almost-Linear Time », *in*: *SIAM Journal on Computing* 47.*6* (2018), pp. 2315–2336, DOI: 10.1137/16M1061850.

[Lux07] Ulrike von Luxburg, « A tutorial on spectral clustering », *in*: *CoRR* abs/0711.0189 (2007).

[LWC23] Yang Lou, Lin Wang, and Guanrong Chen, « Structural Robustness of Complex Networks: A Survey of A Posteriori Measures », *in*: *CoRR* abs/2302.03745 (2023), DOI: 10.48550/ARXIV.2302.03745.

[Mar+18] Jose L Marzo et al., « On Selecting the Relevant Metrics of Network Robustness », *in*: *2018 10th International Workshop on Resilient Networks Design and Modeling (RNDM)*, 2018, pp. 1–7, DOI: 10.1109/RNDM.2018.8489809.

[Mas22] Stefano Massei, « Some Algorithms for Maximum Volume and Cross Approximation of Symmetric Semidefinite Matrices », *in*: *BIT* 62.*1* (2022), pp. 195–220, ISSN: 0006-3835, DOI: 10.1007/s10543-021-00872-1.

[MO17] Aleksandr Mikhalev and I.V. Oseledets, « Rectangular maximum-volume submatrices and their applications », English (US), *in*: *Linear Algebra and Its Applications* 538 (2017), pp. 187–211, ISSN: 0024-3795, DOI: 10.1016/j.laa.2017.10.014.

[MOA11] Albert W. Marshall, Ingram Olkin, and Barry C. Arnold, *Inequalities: Theory of Majorization and its Applications*, Second, vol. 143, Springer, 2011.

[Moh89] Bojan Mohar, « Isoperimetric numbers of graphs », *in*: *Journal of Combinatorial Theory, Series B* 47.*3* (1989), pp. 274–291, ISSN: 0095-8956, DOI: https://doi.org/10.1016/0095-8956(89)90029-4.

[MS90] Dilip B Madan and Eugene Seneta, « The Variance Gamma (V.G.) Model for Share Market Returns », *in*: *The Journal of Business* 63.*4* (1990), pp. 511–24.

[MSN10] Attilio Milanese, Jie Sun, and Takashi Nishikawa, « Approximating spectral impact of structural perturbations in large networks », *in*: *Phys. Rev. E* 81 (4 2010), p. 046112, DOI: 10.1103/PhysRevE.81.046112.

[MSS13] Adam Marcus, Daniel Spielman, and Nikhil Srivastava, « Interlacing Families II: Mixed Characteristic Polynomials and the Kadison-Singer Problem », *in*: *Annals of Mathematics* 133 (June 2013), DOI: 10.4007/annals.2015.182.1.8.

[MSS19] Roy Mitz, Nir Sharon, and Yoel Shkolnisky, « Symmetric Rank-One Updates from Partial Spectrum with an Application to Out-of-Sample Extension », *in*: *SIAM Journal on Matrix Analysis and Applications* 40.*3* (2019), pp. 973–997, DOI: 10.1137/18M1172120.

[MT20] Thomas Maugey and Laura Toni, « Large Database Compression Based on Perceived Information », *in*: *IEEE Signal Processing Letters* 27 (2020), pp. 1735–1739, DOI: 10.1109/LSP.2020.3025478.

[Mui82] Robb John Muirhead, *Aspects of multivariate statistical theory*, Wiley series in probability and mathematical statistics. Probability and mathematical statistics, New York: John Wiley & Sons, 1982, ISBN: 0-471-09442-0.

[Nd11] Mariá C.V. Nascimento and André C.P.L.F. de Carvalho, « Spectral methods for graph clustering – A survey », *in*: *European Journal of Operational Research* 211.*2* (2011), pp. 221–231, ISSN: 0377-2217, DOI: https://doi.org/10.1016/j.ejor.2010.08.012.

[Nor98] Sam Northshield, « A Note on the Zeta Function of a Graph », *in*: *Journal of Combinatorial Theory, Series B* 74.*2* (1998), pp. 408–410, ISSN: 0095-8956, DOI: https://doi.org/10.1006/jctb.1998.1861.

[Obe74] Fritz Oberhettinger, *Tables of Mellin transforms*, Springer-Verlag, 1974.

[Pan00] C.-T. Pan, « On the existence and computation of rank-revealing LU factorizations », English, *in*: *Linear Algebra and Its Applications* 316.*1-3* (2000), pp. 199–222.

[PAS14] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena, « DeepWalk: online learning of social representations », *in*: *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, New York, New York, USA: Association for Computing Machinery, 2014, pp. 701–710, ISBN: 9781450329569, DOI: 10.1145/2623330.2623732.

[Pea82] Judea Pearl, « Reverend Bayes on Inference Engines: A Distributed Hierarchical Approach. », *in*: *AAAI*, ed. by David L. Waltz, AAAI Press, 1982, pp. 133–136, ISBN: 0-262-51051-0, URL: http://dblp.uni-trier.de/db/conf/aaai/aaai82.html#Pearl82.

[PRM23] Claude Petit, Aline Roumy, and Thomas Maugey, « A Water-filling Algorithm Maximizing the Volume of Submatrices Above the Rank », *in*: *EUSIPCO 2023 - 31st European Signal Processing Conference*, Helsinki, Finland, Sept. 2023, pp. 1–5, URL: https://inria.hal.science/hal-04132343.

[Rou14] Nicolas Rougerie, « Théorèmes de de Finetti, limites de champ moyen et condensation de Bose-Einstein », Sept. 2014, URL: https://hal.archives-ouvertes.fr/hal-01060125.

[RV07] Mark Rudelson and Roman Vershynin, « Sampling from large matrices: An approach through geometric functional analysis », *in*: *J. ACM* 54.*4* (2007), 21–es, ISSN: 0004-5411, DOI: 10.1145/1255443.1255449.

[Saa11] Yousef Saad, *Numerical Methods for Large Eigenvalue Problems*, Society for Industrial and Applied Mathematics, 2011, DOI: 10.1137/1.9781611970739.

[SBG23] Karim Shahbaz, Madhu Belur, and Ajay Ganesh, « Algebraic Connectivity: Local and Global Maximizer Graphs », *in*: *IEEE Transactions on Network Science and Engineering* 10 (May 2023), pp. 1636–1647, DOI: 10.1109/TNSE.2022.3232397.

[Sca+09] Franco Scarselli et al., « The Graph Neural Network Model », *in*: *IEEE Transactions on Neural Networks* 20.*1* (2009), pp. 61–80, DOI: 10.1109/TNN.2008.2005605.

[Sch18] Aaron Schild, « An almost-linear time algorithm for uniform random spanning tree generation », *in*: *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, Los Angeles, CA, USA: Association for Computing Machinery, 2018, pp. 214–227, ISBN: 9781450355599, DOI: 10.1145/3188745.3188852.

[Sou+13] C. Soussen et al., « Joint k-Step Analysis of Orthogonal Matching Pursuit and Orthogonal Least Squares », *in*: *IEEE Trans. on Information Theory* 59.*5* (May 2013).

[Spi] Daniel A. Spielman, *spectral and Algebraic Graph Theory*, Version : 4-12-2019, URL: http://cs-www.cs.yale.edu/homes/spielman/sagt/sagt.pdf.

[Spi10] Daniel Spielman, « Algorithms, Graph Theory, and Linear Equations in Laplacian Matrices », *in*: June 2010, pp. 2698–2722, DOI: 10.1142/9789814324359_0164.

[SS11] Daniel A. Spielman and Nikhil Srivastava, « Graph Sparsification by Effective Resistances », *in*: *SIAM Journal on Computing* 40.*6* (2011), pp. 1913–1926, DOI: 10.1137/080734029.

[ST04] Daniel A. Spielman and Shang-Hua Teng, « Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems », *in*: STOC '04, Chicago, IL, USA: Association for Computing Machinery, 2004, pp. 81–90, ISBN: 1581138520, DOI: 10.1145/1007352.1007372.

[ST06] Daniel A. Spielman and Shang-Hua Teng, « Nearly-Linear Time Algorithms for Preconditioning and Solving Symmetric, Diagonally Dominant Linear Systems », *in*: *ArXiv* abs/cs/0607105 (2006), URL: https://api.semanticscholar.org/CorpusID:1750944.

[ST11] Daniel A. Spielman and Shang-Hua Teng, « Spectral Sparsification of Graphs », *in*: *SIAM Journal on Computing* 40.*4* (2011), pp. 981–1025, DOI: 10.1137/08074489X.

[TBA18] Nicolas Tremblay, Simon Barthelme, and Pierre-Olivier Amblard, *Optimized Algorithms to Sample Determinantal Point Processes*, 2018.

[Ter05] Audry Terras, « What are zeta functions of graphs and what are they good for ? », *in*: 2005, URL: https://api.semanticscholar.org/CorpusID:18668576.

[Ter10] Audrey Terras, *Zeta Functions of Graphs: A Stroll through the Garden*, Cambridge Studies in Advanced Mathematics, Cambridge University Press, 2010.

[TG07] J. A. Tropp and A. C. Gilbert, « Signal recovery from random measurements via Orthogonal Matching Pursuit », *in*: *IEEE Trans. on Information Theory* 53.*12* (2007), pp. 4655–4666.

[Top+22] J Topping et al., « Understanding over-squashing and bottlenecks on graphs via curvature », *in*: International Conference on Learning Representations, OpenReview, 2022.

[Tro04] J. A. Tropp, « Greed is good: Algorithmic results for sparse approximation », *in*: *IEEE Trans. on Information Theory* 50 (2004), pp. 2231–2242.

[Tro09] Joel A. Tropp, « Column subset selection, matrix factorization, and eigenvalue optimization », *in*: *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2009, New York, NY, USA, January 4-6, 2009*, ed. by Claire Mathieu, SIAM, 2009, pp. 978–986.

[Tyr00] E. Tyrtyshnikov, « Incomplete Cross Approximation in the Mosaic-Skeleton Method », *in*: *Computing (Vienna/New York)* 64 (June 2000), pp. 367–380.

[Vaa00] A. W. van der Vaart, *Asymptotic Statistics*, Cambridge University Press, 2000.

[Vel+17] Petar Veličković et al., « Graph Attention Networks », *in*: *6th International Conference on Learning Representations* (2017).

[Vis13] Nisheeth K. Vishnoi, « Lx = b », *in*: *Foundations and Trends® in Theoretical Computer Science* 8.*1–2* (2013), pp. 1–141, ISSN: 1551-305X, DOI: 10.1561/0400000054.

[Wai19] Martin J. Wainwright, *High-Dimensional Statistics: A Non-Asymptotic Viewpoint*, Cambridge Series in Statistical and Probabilistic Mathematics, Cambridge University Press, 2019.

[Wan+14] Xiangrong Wang et al., « Improving robustness of complex networks via the effective graph resistance », *in*: *The European Physical Journal B* 87 (Sept. 2014), pp. 1–12, DOI: 10.1140/epjb/e2014-50276-0.

[Wea04] Nicholas Weaver, « The Kadison–Singer problem in discrepancy theory », *in*: *Discrete Mathematics* 278 (Mar. 2004), pp. 227–239, DOI: 10.1016/S0012-365X(03)00253-X.

[Wil88] J. H. Wilkinson, *The Algebraic Eigenvalue Problem*, USA: Oxford University Press, Inc., 1988, ISBN: 0198534183.

[WLY22] Yulong Wei, Rong-hua Li, and Weihua Yang, *Biharmonic distance of graphs*, 2022, URL: https://arxiv.org/abs/2110.02656.

[Won89] Roderick Wong, *Asymptotic Approximations of Integrals*, Academic Press, 1989, pp. iii–, ISBN: 978-0-12-762535-5.

[Wu+22] Lingfei Wu et al., eds., *Graph Neural Networks: Foundations, Frontiers, and Applications*, en, Singapore: Springer Nature Singapore, 2022, DOI: 10.1007/978-981-16-6054-2, (visited on 07/27/2022).

COLLEGE DOCTORAL BRETAGNE
MATHS, TELECOMS
INFORMATIQUE, SIGNAL
SYSTEMES, ELECTRONIQUE

Université de Rennes

Titre : Échantillonnage de données : compression de base de données, réduction de graphe et acquisition comprimée ; trois aspects de la réduction de dimension.

Mot clés : réduction de dimension, sparsification de graphe, échantillonnage, acquisition comprimée.

Résumé : Dans cette thèse, nous étudions trois aspects du problème de réduction de la dimension. Le premier concerne la compression de base de données. Nous proposons plusieurs algorithmes d'échantillonnage préservant l'information contenue dans les données, ainsi que deux applications au conditionnement de matrices et à l'acquisition comprimée. Ces algorithmes sont déterministes et leur faible complexité en font une alternative intéressante aux meilleurs algorithmes connus. Le second aspect abordé concerne la sparsification de graphe. Nous proposons de réduire le nombre d'arêtes d'un graphe tout en préservant sa connectivité. Nous élaborons deux algorithmes itératifs, déterministes et de faible complexité, permettant d'approcher la solution de ce problème NP-difficile. Nous présentons également une application possible à la simplification du graphe sous-jacent à un réseau neuronal sur graphe. La troisième partie de la thèse traite d'acquisition comprimée et propose une analyse statistique d'un algorithme de reconstruction de signaux parcimonieux. Dans le cadre d'un modèle asymptotique où la matrice de mesure et le signal sont aléatoires et pour lequel les paramètres de taille tendent vers l'infini à la même vitesse, nous montrons que la probabilité de succès à une itération donnée tend vers 1.

Title: Data Sampling: Database Compression, Graph Reduction, and Compressed Sensing; Three Aspects of Dimensionality Reduction.

Keywords: dimensionnality reduction, graph sparsification, sampling, compressive sensing.

Abstract: In this thesis, we study three aspects of the dimensionality reduction problem. The first concerns database compression. We propose several sampling algorithms that preserve the information contained in the data, along with two applications in matrix conditioning and compressive sensing. These algorithms are deterministic, and their low complexity makes them a interesting alternative to the state-of-the-art algorithms. The second aspect addressed is graph reduction. We aim to reduce the number of edges, while attempting to preserve the graph's connectivity. We develop two iterative, deterministic, and low-complexity algorithms that approximate the solution to this NP-hard problem. We also present a possible application in simplifying the underlying graph of a Graph Neural Network. The third part of the thesis deals with compressive sensing and provides a statistical analysis of a reconstruction algorithm for sparse signals. In the context of an asymptotic model where both the measurement matrix and the sparse signal are random, and the size parameters tend to infinity at the same rate, we show that the probability of success at a given iteration tends to 1.