Probabilistic, fuzzy, and quantum methods for retrieving biological information

Thomas Sierocinski

N° d'Ordre : 3790

THÈSE

Présentée

DEVANT L'UNIVERSITÉ DE RENNES I

pour obtenir

le grade de DOCTEUR DE L'UNIVERSITÉ DE RENNES I

Mention Mathématiques et Applications

par

Thomas SIEROCINSKI

Institut de Recherche Mathématique de Rennes

École Doctorale MATISSE

U.F.R. de Mathématiques

TITRE DE LA THÈSE :

Méthodes probabilistes, floues et quantiques pour l'extraction de l'information biologique

Soutenue le ?? septembre 2008 devant la Commission d'Examen

COMPOSITION DU JURY :

Pr.	J-F. Yao	Président
Pr.	P. Barbry	Rapporteur extérieur/ Examinateur
Pr.	F. Benatti	Rapporteur
Pr.	R. Fernández	Rapporteur
Pr.	D. Pétritis	Directeur
Pr.	N. Théret	Directrice

Préface

Comment pouvons-nous expliquer les différences observables entre un papillon et une chenille ? un neurone et une cellule musculaire ? ou encore entre une cellule cancéreuse et une cellule saine ? En principe, toutes les cellules d'un même organisme portent un même génotype, patron de toutes les molécules produites par une cellule. La réponse se trouve au niveau de l'expression de ce code, l'activité des gènes peut être modulée conduisant ainsi à des phénotypes différents. Nous pouvons cependant nous interroger quant aux causes de cette différenciation dans l'expression des gènes, surtout sachant que la plupart des êtres vivants sont issus d'un stade unicellulaire. Ou encore sur ce qui se cache derrière les capacités d'une cellule à répondre à des stimuli externes ou internes par des réactions appropriées. Celles-ci sont définies par les propriétés du système de régulation génétique de la cellule qui peut être perçu comme un réseau complexe de réactions chimiques auxquelles est corrélé l'expression des gènes.

Il n'y a pas si longtemps de cela, les chercheurs étaient limités à l'étude extérieure de ces systèmes, par des observations de haut niveau, au mieux en mesurant l'activité d'un nombre réduit de gènes simultanément. Cependant, les années 90 ont vu l'émergence de technologies permettant de mesurer l'activité de plusieurs milliers de gènes simultanément : les puces à ADN. Leur utilisation permet la mesure de la quantité d'ARNm provenant, en principe, de chacun des gènes de la cellule étudiée à un moment donné. En d'autres termes, nous disposons des moyens technologiques pour obtenir une représentation quasi-« photographique » (basée sur le niveau d'ARNm) du réseau de régulation génétique d'une cellule. Idéalement, ce type de données permet de répondre de manière pertinente à une question biologique, par exemple, inférer un rôle pour un gène donné, diagnostiquer ou déchiffrer des mécanismes de maladies complexes à partir d'échantillons cellulaires.

La taille et la complexité de ces systèmes nécessitent l'utilisation de méthodes de calcul afin d'en extraire une information biologiquement pertinente. Diverses méthodes issues des mathématiques, des statisques, de la physique ou de l'informatique se sont révélées très utiles dans l'analyse des données d'expression. Ces méthodes peuvent être réparties en deux catégories : les méthodes de classification non supervisée ne se basant sur aucune connaissance préalable, et les méthodes de classification supervisées nécessitant des données d'apprentissage. L'état des connaissances actuelles sur les fonctions des gènes ainsi que les possibilités d'expériences qu'offrent les puces à ADN impliquent parfois une méconnaissance totale du système étudié, invalidant ainsi une approche supervisée.

Ces approches dites de regroupement (ou clustering) repartissent généralement l'ensemble des gènes en groupes distincts en fonction de leurs similitudes d'expression. Ce type d'approche permet principalement, par rapprochement avec des gènes connus, d'inférer un rôle à un nouveau gène. Ici aussi, l'introduction de connaissance est inévitable. Que se passe-t-il lorsque tous les gènes d'un groupe sont inconnus ? De plus, les méthodes traditionnelles affectant généralement un gène à un seul et unique groupe, ne nous renseignent que peu sur l'implication d'un gène au sein du système (c'est à dire de la cellule ou d'un échantillon biologique). C'est dans ce cadre que nous proposons une nouvelle méthode permettant, sans aucune connaissance a priori, de caractériser la spécificité de l'ensemble des gènes pour les échantillons biologiques d'une expérience de puces à ADN et ainsi quantifier leur implication pour un ensemble de phénotypes et enfin, comme pour les approches classiques, de répondre de manière pertinente à des questions biologiques fines. Nous appelons cette méthode distillation sémantique.

Après une brève présentation générale de la thèse (chapitre 1), nous introduirons, dans le chapitre 2, les mécanismes de régulation géniques, les outils de mesures de l'activité des gènes ainsi que les principaux outils d'analyses. Le chapitre 3 traite de la représentation mathématique des données présentant la même structure que les mesures de puces à ADN ainsi que ses propriétés. Nous passerons ensuite en revue les fondements mathématiques des méthodes de classification non supervisées, tout d'abord les méthodes ensemblistes (chapitre 4) ensuite les méthodes spectrales (chapitre 5). Précedant la présentation de la distillation sémantique (chapitre 7), le chapitre 6 traite des fondements logiques de notre approche. Le chapitre 8 présente deux cas d'application de notre méthode.

Contents

Chapitre 1

Présentation générale de la thèse

1.1 Position du problème bioinformatique

Les progrès des technologies de mesure et le séquençage des génomes, ont permis l'émergence, dans les années 1990, de techniques de mesure globale de l'expression génique. Jusqu'alors les méthodes employées ne permettaient l'étude de l'expression que de quelques gènes, les puces à ADN ont alors donné aux biologistes les outils nécessaires pour effectuer ces mesures sur l'ensemble des gènes d'un échantillon biologique à un moment donné. Les résultats de mesure de l'activité des gènes dans divers échantillons biologiques sont présentés sous la forme d'une matrice réelle de données d'expression :

$$\mathsf{X} = (x_{ab})_{a \in \mathbb{A}, b \in \mathbb{B}}$$

de taille $|\mathbb{A}| \times |\mathbb{B}|$ où x_{ab} représente l'expression du gène b dans l'échantillon biologique a. Les puces à ADN pouvant couvrir l'ensemble du génome d'un individu, ces matrices sont généralement de grande taille. Idéalement l'étude de ces matrices permet de répondre de manière efficace à une question biologique. Par exemple : Existe-il des regroupements de gènes permettant d'inférer une fonction particulière à un gène dont on ignore le rôle ? Qu'est-ce qui distingue les divers échantillons de l'expérience ? Peut-on trouver des signatures dans l'expression des gènes des divers échantillons ? etc.

L'interprétation de ce type de résultats expérimentaux nécéssite des méthodes d'analyse performantes et robustes. Dans ce but, de nombreux outils mathématiques ont été développés ou adaptés. Ces méthodes d'extraction d'information reposent généralement sur des algorithmes de classification et permettent la répartition d'un ensemble d'objets en plusieurs sous-ensembles tout en minimisant la variabilité au sein de ce sous-ensemble.

En fonction de la question posée, il est nécessaire de définir une sémantique pour le jeu de données. La sémantique apparaît ici comme le sens implicite des données, elle permet de définir les objets ainsi que leur(s)rôle(s) dans le système. Cette sémantique correspond généralement à une mesure de distance traduisant la proximité (similitude) ou l'éloignement (dissimilitude) entre deux objets du jeu de données. Parmi les distances les plus utilisées nous pouvons citer, par exemple, la distance euclidienne, la distance de Manhattan, la correlation de Pearson, la distance angulaire, la corrélation de Spearman, la corrélation de Kendall. Chaque distance associe une sémantique différente au jeu des données et donne potentiellement des résultats différents, il est donc important de bien choisir sa distance en fonction de la question biologique à laquelle nous voulons répondre.

Nous pouvons aisement discerner deux groupes au sein des méthodes de classification :

- *les algorithmes de classification non supervisée* : sont des processus automatiques qui séparent les données observées en groupes distincts sans aucune connaissance préalable des classes existantes. Ces méthodes sont encore aujourd'hui couramment utilisées dans le cadre de l'interprétation des résultats expérimentaux de type puces à ADN. De nombreux logiciels sont aujourd'hui disponibles, regroupant la plupart du temps un certain nombre de méthodes comme :
  - le regroupement hierarchique en amas (hierarchical clustering)
  - l'algorithme des k-moyennes ou (k-means)
  - les réseaux de Kohonen (self organising map)
  - la sous-matrice préservant l'ordre (order preserving submatrix)
  - le regroupement lisse (smooth clustering)
  - les modèles en plaid
  - l'analyse en composantes principales
  - la décomposition en valeurs singulières
  - les méthodes spectrales des graphes
  - le regroupement flou
- *les algorithmes de classification supervisée* : contrairement aux méthodes de classification non supervisée, ces algorithmes nécéssitent l'introduction de connaissances *a priori* afin de répartir les données en classes. La phase d'apprentissage permet au système, à partir d'un jeu de données d'apprentissage connu et annoté, d'inférer des règles de classification qui servent ensuite à la classification d'autres jeu de données. Parmi les méthodes couramment utilisées nous pouvons citer :
  - les k plus proches voisins
  - la classification par l'analyse des centroïdes
  - l'analyse discriminante linéaire
  - les machines à vecteurs de support
  - les réseaux de neurones
  - l'algorithme EM (expectation-maximisation)

  Les résultats de ces méthodes dépendent directement de la phase d'apprentissage. Les dimensions et l'aspect bruité des matrices de données d'expression peuvent limiter l'application des ces algorithmes. Le risque principal est le sur-apprentissage (mauvaise généralisation des caractéristisques des données, le système classifie parfaitement les échantillons d'apprentissage mais perd son pouvoir de prédiction sur d'autres jeu de données).

Compte-tenu des connaissances actuelles sur la fonctionnalité des gènes, il est parfois avantageux de pouvoir conduire des analyses sans connaissance *a priori* sur le jeu de données, qu'elles soient du type données d'apprentissage ou simplement des connaissances liées à une annotation (caractéristique, fonction, rôle des gènes,...). Dans cette étude, nous allons tout d'abord détailler les approches mathématiques liées aux méthodes de classification non supervisée, ensuite nous allons présenter notre approche appellée « distillation sémantique », s'inscrivant dans cette même catégorie et permettant de caractériser l'appartenance de l'ensemble des gènes d'une puce aux divers échantillons biologiques de l'expérience.

1.2 Résumé de la méthode mathématique dévelloppée

Soit $\mathbb{B}$ un ensemble fini de gènes et $\mathbb{A}$ un ensemble fini de contextes cellulaires (ou échantillons biologiques). Le jeu de donnée est représenté par une matrice réelle $\mathsf{X} = (x_{ab})_{a\in\mathbb{A},b\in\mathbb{B}}$ de dimension $|\mathbb{A}| \times |\mathbb{B}|$. Celle-ci peut être vue comme un ensemble de $\mathbb{B}$ vecteurs colonnes

$$\mathbf{x}_b = \begin{pmatrix} x_{b1} \\ \vdots \\ x_{b|\mathbb{A}|} \end{pmatrix} \in \mathbb{R}^{\mathbb{A}}, \text{ pour } b \in \mathbb{B}$$

Les éléments de la matrice x_{ab} sont des réels codant l'intensité lumineuse mesurée sur la puce à ADN qui représente le niveau d'expression du gène b dans l'échantillon biologique a.

L'algorithme de la distillation sémantique peut être divisée en trois principales étapes. Chaque étape est réalisée sur une représentation différente du jeu de données et de son contenu informationnel. Ces représentations ainsi que les traitements associés agissent comme des filtres successifs permettant de conserver et révéler la part la plus pertinente et significative de l'information contenue dans le jeu de données.

1.2.1 Représentation des données dans l'espace des concepts

Pour $\mathbb{A}$ et $\mathbb{B}$ définis précedement, nous définissons l'*espace des concepts* $\mathcal{H}_{\mathbb{A}}$ comme un espace de Hilbert libre sur $\mathbb{A}$, les élements de $\mathbb{A}$ forment la base orthonormée de $\mathcal{H}_{\mathbb{A}}$. Le jeu de données complet peut donc être représenté comme un ensemble de $|\mathbb{B}|$ vecteurs $|\,\Xi_b\,\rangle$ décomposés sur la base orthonormée de $\mathcal{H}_{\mathbb{A}}$.

$$|\,\Xi_b\,\rangle = \sum_{a\in\mathbb{A}} x_{ab}|\,a\,\rangle \in \mathcal{H}_{\mathbb{A}}$$

avec $b \in \mathbb{B}$ et où $|\,a\,\rangle$ représente l'element de la base orthonormée de l'espace vectoriel libre correspondant à l'attribut a. L'espace vectoriel est équipé d'un produit scalaire et des rayons (vecteurs normalisés contruits en divisant les vecteurs par leurs normes) sont introduits dans l'espace $\mathcal{H}_{\mathbb{A}}$. Nous utiliserons la notation $|\,\xi_b\,\rangle$ pour représenter le rayon associé au vecteur $|\,\Xi_b\,\rangle$:

$$|\,\xi_b\,\rangle = \frac{|\,\Xi_b\,\rangle}{\||\,\Xi_b\,\rangle\|}.$$

La structure de $\mathcal{H}_{\mathbb{A}}$ permet une géométrisation naturelle de l'espace des concepts en l'équipant d'une pseudo-distance d

$$d : \mathbb{B} \times \mathbb{B} \to \mathbb{R}_+$$

définie par

$$d(b, b') = \||\,\xi_b\,\rangle - |\,\xi_{b'}\,\rangle\|.$$

L'espace $\mathcal{H}_{\mathbb{A}}$ devient donc l'espace de représentation de toute l'information logique et probabiliste contenue dans le jeu de données initial. Ceci est détaillé aux chapitres 2 et 6.

1.2.2 Construction et réduction dimensionnelle d'un graphe pondéré

À partir de la représentation précédente nous introduisons au paragraphe 5.2 l'ensemble $\mathbb{V}_{\mathbb{A}} = \mathbb{A} \cup \mathbb{B}$. Donc pour tout $v \in \mathbb{V}_{\mathbb{A}}$,

$$|\,\Xi_v\,\rangle = \begin{cases} |\,a\,\rangle & \text{si} \quad v = a \in \mathbb{A}, \\ \sum_{a \in \mathbb{A}} x_{ab} |\,a\,\rangle & \text{si} \quad v = b \in \mathbb{B}. \end{cases}$$

Les nouveaux vecteurs $|\,\Xi_a\,\rangle = |\,a\,\rangle$ sont ici inclus comme des *témoins de spécificité* aux divers attributs de $\mathbb{A}$. Comme ces nouveaux vecteurs sont aussi des élements de $\mathcal{H}_{\mathbb{A}}$, la pseudo-distance d s'étend naturellement à $\mathbb{V}_{\mathbb{A}}$. Supposons maintenant une fonction de similitude s basée sur la pseudodistance d.

Comme nous l'avons vu dans la première partie de ce résumé, il existe de nombreuses fonctions de similitude et chacune associe une sémantique particulière au jeu de données. Dans le cadre de nos expréριences nous avons utilisé la distance angulaire entre les vecteurs de $\mathbb{V}_{\mathbb{A}}$. Cette distance permet d'éviter d'accorder une importance trop grande (au regard de la sémantique) aux gènes très fortement exprimés.

Nous pouvons maintenant construire un graphe pondéré sur l'ensemble des sommets $\mathbb{V}_{\mathbb{A}}$. Les poids sont issus de la fonction de similitude s et sont assignés aux arêtes du graphe complet défini sur $\mathbb{V}_{\mathbb{A}}$, constituant ainsi une matrice $W_{vv'}$ représentant le graphe.

$$W_{vv'} = s(v, v').$$

Nous cherchons maintenant à révéler les interactions les plus pertinentes entre les gènes. Cette étape est réalisée en appliquant les techniques standard de réduction dimensionelle du graphe par sa représentation optimale dans des espaces euclidiens de petite dimension formés par les vecteurs propres du laplacien pondéré.

1.2.3 Regroupement flou et distillation

Les gènes (éléments de $\mathbb{B}$) ne sont en général pas spécifiques d'un seul échantillon biologique (éléments de $\mathbb{A}$). Ils interviennent dans plusieurs contextes cellulaires avec des niveaux d'expression différents et en collaboration avec d'autres gènes, justifiant ainsi l'approche d'un regroupement flou.

Jusqu'a cette étape, notre méthode est une suite d'algorithmes connus et utilisés de manière séparée dans divers contextes. L'originalité de notre méthode repose sur l'étape suivante.

La distillation sémantique est une procédure récursive d'extraction d'information (détaillée dans la partie 7.2) où chaque observation effectuée modifie le contenu informationnel du système, rappelant la procédure de mesure en mécanique quantique (il ne s'agit pas ici d'un traitement quantique de l'information mais d'un traitement classique à l'aide d'un formalisme *inspiré* de la mécanique quantique).

L'état du système $\mathcal{H}_{\mathbb{A}}$ peut être représenté par une matrice densité

$$\sigma = \frac{\mathsf{XX}^\dagger}{\mathsf{tr}(\mathsf{XX}^\dagger)}$$

permettant, à chaque itération de notre méthode, de calculer une probabilité d'appartenance des élements de $\mathbb{V}_{\mathbb{A}}$ à deux sous-groupes (sous-espaces de $\mathcal{H}_{\mathbb{A}}$). Les éléments de $\mathbb{A}$ sont associés à l'un ou l'autre de ces sous-espaces (en fonction de la probabilité mesurée), les éléments de $\mathbb{B}$ sont projetés dans les deux sous-espaces. De cette manière, nous réduisons l'indétermination du système. L'information extraite est ici réinjectée dans le système afin de procéder à une nouvelle mesure. La méthode reprend à la première étape jusqu'à ce que tous les objets de $\mathbb{A}$ soient répartis dans des sous espaces différents. La répétition de cette opération permet d'obtenir des mesures de probabilité extrémales qui s'interprétent somme sémantiquement non-ambiguës.

1.3 Application et résumé des résultats obtenus

Nous avons mis à l'épreuve notre méthode sur des données de puces à ADN publiées dans [77] et disponible sur la base Gene Expression Omnibus [75] sous la référence GSE803. Ce jeu de données est constitué d'un ensemble de 62637 sondes mesurées dans 12 tissus humains : moelle osseuse, cerveau, coeur, rein, foie, poumon, pancreas, prostate, muscle squeletique, moelle epinière, rate et thymus. L'ensemble des sondes est réparti sur cinq puces HG-U95Av2, HG-U95B, HG-U95C, HG-U95D, HG-U95E). La puce HG-U95Av2 contient 12613 gènes complets tandis que les autres representent approximativement 50000 EST (Expressed Sequence Tag). Dans le cadre de cette étude nous avons choisi de n'utiliser que les mesures issues de la puce HG-U95Av2.

Nous ne présentons ici que le cas de la représentation unidimensionnelle du graphe pour un échantillon biologique (ici le muscle squeletique) parmi les douze de l'étude.

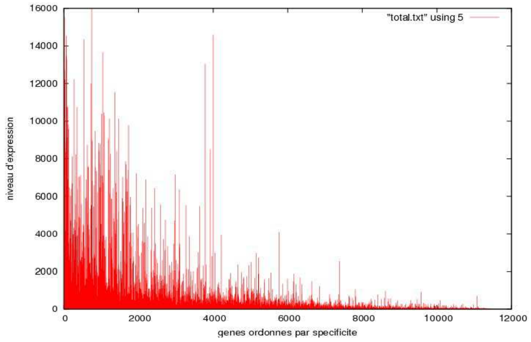


Fig. 1.1 – Classement des gènes du jeu de donné GSE803. L'axe horizontal représente l'ensemble des gènes classés par appartenance décroissante à l'échantillon biologique (muscle squeletique). L'axe vertical représente la mesure expérimentale de l'activité de ces gènes dans cet échantillon biologique.

Tab. 1.1 – Annotation des gènes les plus proches du témoins de spécificité « muscle squeletique ».

Ranking	Name	Annotation
1	ATPase	Ca++ transporting, cardiac muscle, fast twitch 1, calcium signaling pathway
2	Troponin I type 2	skeletal, fast
3	Myosin, light chain 1	alkali ; skeletal, fast
4	Ryanodine receptor 1	skeletal ; calcium signaling pathway
5	Fructose-1,6-bisphosphatase 2	glycolysis gluconeogenesis
6	Actinin, alpha 3	focal adhesion
7	Troponin C type 2	fast ; calcium signaling pathway
8	Carbonic anhydrase III	muscle specific ; nitrogen metabolism
9	Nebulin	
10	Troponin I type 1	skeletal, slow
11	Myosin, heavy chain 3	skeletal muscle
12	Myogenic factor 6, herculin	

La figure 1.1 représente le type de résultat que nous pouvons obtenir : pour chaque groupe ne contenant qu'un élement de $\mathbb{A}$ (échantillon biologique), l'axe horizontal représente l'ensemble des gènes classés par leur appartenance décroissante au contexte cellulaire (échantillon biologique). L'axe vertical représente la mesure expérimentale de l'activité de ces gènes dans cet échantillon biologique. Pour chaque groupe, nous pouvons observer un gradiant d'expression suivant notre gradiant de spécificité (classement des gènes). Pourtant, nous pouvons aussi remarquer que les gènes les plus exprimés ne sont pas obligatoirement les plus spécifiques. En fait, un gène très exprimé dans divers échantillons biologiques ne sera pas par définition, classé parmi les gènes les plus pertinents pour un contexte particulier. Respectivement, un gène faiblement exprimé dans un seul et unique échantillon sera considéré comme spécifique. Le tableau 1.1 donne un exemple de l'annotation fournie par la base de donnée Unigene [75] pour les gènes classés comme les plus spécifiques à l'échantillon muscle squeletique. Nous pouvons remarquer que la majorité de ces gènes est annotée comme spécifique du muscle squeletique. Toutefois, nous pouvons également remarquer que le premier de la liste (ATPase) est annoté comme spécifique du muscle cardiaque (autre échantillon biologique de l'analyse) ce qui pourrait apparaître comme une erreur de classification. Après avoir contrôlé l'expression de ce gène dans le jeu de données et avoir analysé la litterature le concernant, il est apparu que ce gène est exprimé cinq fois plus dans le muscle squeletique que dans le coeur (au regard des mesures expérimentales), de plus il est considéré comme responsable d'une forme de myopathie affectant les muscles squeletiques (maladie de Brody) [52]. Donc, au regard de ces observations notre méthode a effectivement bien classé ce gène est a permis de lever une ambiguité concernant sa spécificité. Nous présenterons dans le chapitre 8 des résultats plus détaillés ainsi qu'une étude statisque de l'efficacité de notre méthode.

Nous avons également mis à l'épreuve la méthode de distillation sémantique sur un jeu de données cliniques traduisant des mesures d'expression de gènes chez des patients atteints de fibroses hépatiques plus ou moins sévères. La fibrose hépatique est définie par l'accumulation excessive d'une matrice extracellulaire dans le foie. Conséquence des atteintes hépatiques chroniques, sa progression conduit à terme à la cirrhose et au cancer. Dans le cadre de cette étude nous disposions d'un ensemble $\mathbb{B}$ de 700 gènes exprimés chez 14 patients repartis en quatre catégories sur critères anatomopathologiques :

- trois patients souffrant de fibrose légère (F1)
- trois patients souffrant de fibrose modérée (F2)
- trois patients souffrant de fibrose sévère (F3)
- cinq patients souffrant de cirrhose (F4).

Cette analyse a été menée dans le but de trouver des signatures dans l'expression des gènes permettant de caractériser de manière précise les différents stades de fibrose.

La principale différence entre cette analyse et la précédente est que nous travaillons ici sur des tissus de type identique (biopsies de foie), contrairement à l'analyse précedente où la différence entre les échantillons était biologiquement plus marquée. Nous pouvons aisémment imaginer que, dans ce type de données, discriminer des gènes spécifiques est bien plus difficile que dans l'analyse précédente.

Tab. 1.2 – Termes KEGG associés aux gènes spécifiques des différents stades de fibroses, un X représente l'association du terme avec l'échantillon

KEGG terms	F1	F2	F3	F4
VEGF signalling pathway	X		X	
cell cycle	X	X		
Tcell receptor signalling pathway	X			
Adherens junction		X		
jak-STAT signalling pathway		X	X	X
focal adhesion		X	X	X
wnt signalling pathways		X		
pyrimidine metabolism		X		X
leukocyte transendothelial migration		X		
gap junction			X	
MAPK signalling pathway		X	X	
Regulation of actin cytoskeleton			X	
apoptosis			X	X
Bcell receptor signalling pathway			X	
gap junction			X	
Antigen processing and presentation			X	
One carbon pool by folate			X	
Ubiquitin mediated proteolisis				X
purine metabolism				X
huntington desease				X
neurodegenerative desease				X
dentatorubropallidoluysian atrophy				X
Insulin signalling pathway				X

Nous avons obtenu de bons résultats au regard des analyses statisques permettant de les évaluer (ces méthodes sont détaillées dans le chapitre 8). Comme pour l'analyse précédente, nous avons annoté les gènes spécifiques à chaque échantillon de l'analyse. Nous avons choisi deux sources d'annotations :

- KEGG (Kyoto Encyclopedia of Genes and Genomes) est une base de donnée associant à chaque gène référencé les voies métaboliques dans lesquelles celui-ci est impliqué.
- GO (Gene Ontology) est une ontologie dédiée à l'annotation des génomes.

Les résultats obtenus (tableau 8.6 pour l'annotation KEGG, et tableau 1.3 pour l'annotation GO) montrent les différences obtenues au niveau de l'annotation des gènes spécifiques issus de notre analyse. Nous pouvons observer que des signatures apparaissent dans les voies métaboliques ansi que dans les processus biologiques associés aux gènes spécifiques de chaque stade fibrotique. Les résultats seront détaillés dans le chapitre 8.

Biological process	F1	F2	F3	F4
positive regulation of I-kappaB kinase/NF-kappaB cascade	X			X
positive regulation of cell proliferation	X	X	X	X
inflammatory response	X			X
ossification	X			
cell division	X	X		
blood coagulation	X			X
positive regulation of transcription from RNA polymerase	X			
response to drug	X			
dephosphorylation	X			X
embryonic development (sensu Mammalia)	X		X	
DNA unwinding during replication	X			
mRNA export from nucleus	X			
DNA replication	X			X
cell surface receptor linked signal transduction	X		X	
protein amino acid phosphorylation	X	X	X	X
immune response	X			
cell cycle	X	X	X	X
skeletal development	X			
cell proliferation	X			X
cytokinesis		X	X	X
negative regulation of apoptosis		X	X	
regulation of cyclin-dependent protein kinase activity		X		X
regulation of progression through cell cycle		X		
transmembrane receptor protein tyrosine kinase signaling pathway		X		
negative regulation of cell proliferation		X	X	X
intracellular signaling cascade		X		X
development		X		X
cell adhesion		X	X	X
cell cycle arrest		X		
negative regulation of cell cycle		X		

suite du tableau page suivante

Biological process	F1	F2	F3	F4
chemotaxis		X		
traversing start control point of mitotic cell cycle		X		
apoptosis		X	X	X
protein ubiquitination			X	
wnt receptor signaling pathway			X	
signal transduction			X	
regulation of apoptosis			X	X
cell-cell signaling			X	X
insulin receptor signaling pathway			X	X
insulin-like growth factor receptor signaling pathway			X	
inner ear morphogenesis			X	
lung development			X	
bone mineralization			X	
angiogenesis			X	X
fibroblast growth factor receptor signaling pathway			X	
nuclear mRNA splicing, via spliceosome			X	
cell growth			X	X
nervous system development			X	X
response to hypoxia				X
induction of apoptosis				X
JNK cascade				X
response to wounding				X
transcription				X
DNA repair				X
response to stress				X
nucleobase, nucleoside, nucleotide and nucleic acid metabolic process				X
protein complex assembly				X
anterior/posterior pattern formation				X
cell motility				X
inactivation of MAPK activity				X
regulation of cell growth				X
negative regulation of cell cycle				X
nucleotide-excision repair				X
anti-apoptosis				X
regulation of transcription, DNA-dependent				X
liver development				X
placenta development				X
release of cytochrome c from mitochondria				X
positive regulation of transcription from RNA polymerase II promoter				X
apoptotic mitochondrial changes				X
germ cell development				X
protein import into nucleus				X
cell morphogenesis				X
keratinocyte differentiation				X
heart development				X
positive regulation of transcription				X
proteolysis				X
apoptotic program				X
organ morphogenesis				X
actin cytoskeleton organization and biogenesis				X
signal complex formation				X
positive regulation of T cell proliferation				X
epidermis development				X
actin filament organization				X
cell differentiation				X
cell migration				X
sensory perception				X
T cell activation				X
DNA replication initiation				X
UTP biosynthesis				X
CTP biosynthesis				X
GTP biosynthesis				X
nucleotide metabolism				X
brain development				X
regulation of Rho protein signal transduction				X
one-carbon compound metabolism				X
transcription from RNA polymerase II promoter				X
small GTPase mediated signal transduction				X
cellular defense response				X

Table 1.3 : Termes GO associés aux gènes spécifiques des différents stades de fibroses

Chapter 2

DNA arrays

2.1 Gene expression and regulation

The cell is a complex system, composed of a set of organic molecules, interacting with each other and constituting the building blocs of life. This system has amazing skills of communication, environment adaptation, state regulation from internal or external stimuli.

Functional and structural elements of cells are proteins and ribonucleic acids (RNA). Many of these molecules are involved in cell structure and various cellular reactions; i.e. intracellular signalisation or metabolism. These molecules are mostly produced by the cell itself. This production is based on genetic information stored in DNA (deoxyribonucleic acid), a double-helix shaped molecule composed of two complementary antiparallel strands each composed of a sequence of nucleotides. This sequence determines the informational content of DNA. The set of nucleotides is composed of four elements (Adenine, Guanine, Cytosine, Thymine) presenting a complementarity: adenine pairs with thymine by two hydrogen bonds, cytosine with guanine by three hydrogen bonds. This complementarity ensures cohesion of the two strands of the DNA double helix. A gene is a portion of the DNA wich contains the instructions for a specific molecule, its gene product.

In principle, all cells of a multicellular organism carry the same genetic code, identical to that of the egg cell. However, higher species have highly specialised cells grouped into tissues or organs that have a particular role within the organism. We can therefore ask ourselves the question: How, from an identical genetic code, can we obtain cells apparently so different? This can be explained by a different expression of genes within these cells. Some are induced, others repressed according to the cell type, producing a set of specific molecules in the cell.

The protein synthesis consists in a functional protein production from the information given by the corresponding gene. This process can be divided into two phases: transcription and translation.

During transcription, the genetic code of the gene is duplicated to a messenger RNA (mRNA) molecule, a single-stranded nucleic acid carrying the same nucleotides as DNA with the exception of thymine being replaced by uracil (U). Transcription starts when the protein RNA polymerase binds to the promoter region, the start of the gene, and locally unzips the DNA helix so that the strands become free for reading. The RNA polymerase propagates along the strand while constructing an mRNA molecule by adding nucleotides complementary to those of the DNA molecule. Eventually, the RNA polymerase reaches a terminator region and stops transcribing, the mRNA is then released and the DNA resumes its double helix configuration. Following this, the primary mRNA is processed into mature mRNA by other molecules, for example by removing the parts corresponding to introns, non-coding regions of the DNA, in a process called splicing.

Following transcription, translation takes place, where the four-letter alphabet of the DNA and mRNA is translated into the proteins alphabet. Like the nucleic acids, proteins are polymers, consisting of sequences of amino acids instead of nucleotides. The number of amino acids is 20 so the protein alphabet is one of 20 letters. In order to represent 20 amino acids with four nucleotides we need three nucleotides per amino acid. Such a three-nucleotide word is denoted a codon. The actual translation between the two alphabets is accomplished by transfer RNA molecules (tRNA) which attach themselves to the mRNA. The tRNA has one end with a specific anticodon, that is, a complementary codon, and another end to which the corresponding amino acid is attached. The last step in the translation is performed by the ribosomes which join the sequence of amino acids found on the tRNA along the mRNA together to form the protein.

The same protein can attain different spatial shapes (conformation) in the three-dimensional space, constituting a number of folds of the protein, each having their own properties. It is convenient to define the expression level of a gene as the amount of mRNA in the cell transcribed at a given instant. We also define the expression profile of a cell as the set of expression levels of all its genes.

The production of RNA and proteins from a given gene does not take place independently of the expression of other genes. Conversely, gene products influence the production of other gene products using positive or negative feedback. This regulation is essential for the cell to be able to respond to internal and external circumstances and takes place on all levels in the chain of reactions that produce a protein from a gene sequence.

To enable transcription of a gene, the binding of certain proteins to the DNA is necessary, they are called transcription factors. Different genes are either activated or repressed by different combinations of one or several transcription factors. Transcriptional control is not the only means of regulation. After transcription, mRNA molecules may interact with other gene products, resulting in altered structure or lifetime of the mRNA. After translation, subsequent protein-protein reactions may be required to finalise the functional protein.

The description above only sketches a few of the ways that genes interact to regulate each others expression. The main conclusion is that the cell can be viewed as a large dynamical system with different molecules interacting with each other. The explicit study of such genetic regulatory networks is often referred to as systems biology.

2.2 Methods for genes expression measurment

The 90's saw the emergence, enabled by advances in measurement technology and the sequencing of genomes, of technologies for global measurement of gene expression. Previous techniques were limited to the simultaneous study of a few genes, while the microarray techniques gave biologists the tools to sample the expression of, in principle, the whole genome in one single measurement.

Microarrays measure the abundance of mRNA from the set of genes at a given moment, giving us a "photographic" representation of gene expression in a biological sample. From a cell sample of interest, mRNA is extracted and put in contact with an array on which probes (complementary sequences of the genes) have been attached. The different mRNA in the solution then bind to their corresponding complements on the chip, and the amount of mRNA for each gene can be optically measured by fluorometry.

There are two main microarray platforms currently in use; spotted microarrays [60, 61] and high-density synthetic oligonucleotide microarrays [43], wich are basically two variations of the same general solution described above.

A spotted microarray has probes consisting of cDNA or long oligonucleotides strands attached on a glass slide in a grid shaped pattern. The platform is, in its most common form, a two-channel technique, meaning that in each measurement, the expression profiles of two cell samples are measured simultaneously. After extracting RNA from the two samples it is reverse-transcribed to cDNA, and fluorescently labelled with Cy3 (green) for one sample and Cy5 (red) for the other. The cDNA molecules of the samples are denoted targets. After labelling, the two samples are mixed and put in contact with the probes. During hybridisation, the targets bind to their corresponding probes. Finally, each spot is illuminated by a laser at two different wavelengths; one for Cy3 fluorescence and one for Cy5 fluorescence. Thus two images are obtained; one with green spots and one with red spots, measuring the abundances of the respective sample targets. The expression levels of the query sample, are then reported as relative values compared to the reference expressions:

$$x_i = \frac{x_i^{\mathsf{query}}}{x_i^{\mathsf{reference}}}$$

This ratio is transformed by taking the logarithm (discussed in section 2.4.1).

High-density synthetic oligonucleotide microarrays have a different construction. Produced only by specialised companies (Affymetrix, Agilent Technologies), they are increasingly used. The probes are made of small gene sequences, with a typical length of 25 nucleotides, and probes for one gene is spread over

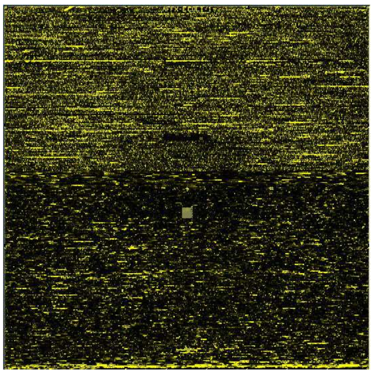


Figure 2.1: Image from a high-density synthetic oligonucleotide microarray. Expression level is represented by a real value. Measures on several biological samples allows expression level comparison through microarrays.

the chip in order to decrease the influence of spatial errors. Moreover, associated with each probe is a mismatch probe, where one nucleotide has been replaced by its complement, thus providing a means of estimating the amount of false positive binding. Total mRNA is extracted form a biological sample, marked and hybridised. As opposed to spotted microarrays, high-density oligonucleotide microarrays are single-channel, measuring one sample on each array. A direct estimation of gene expression is obtained by calculating the average signal through all probes representing the gene.

Comparing the two platforms, spotted microarrays are more flexible because they can be designed in the laboratory, the experimentalist deciding which probes to attach to the chip. On the other hand oligonucleotide chips have less risk of cross hybridisation and a wider dynamical range.

2.3 Genes expression data analysis

The data processing, from scanned array images to the final biological interpretation involves a long series of computational manipulations and analyses of the data. The initial steps are the estimation of expression levels from the raw image data through spot identification, segmentation, intensity estimation and the computation of expression indices. Follows a number of preprocessing steps, where various transformations of the data is applied in order to filter out non-biological variation and to "clean up" data to facilitate subsequent analysis. At this stage, data are ready to be analysed for a biological interpretation. A range of high-level analysis methods exist, aiming to extracting biologically relevant information from the data. Clustering, classification, and other types of methods are all frequently applied in gene expression data analysis. Finally, the extracted structure needs validation, and here too, computational methods are helpful, for example while associating the results to prior knowledge which is often stored in large databases.

2.4 Preprocessing

2.4.1 Data transformation

As described in Section 2.2, expression values on a spotted microarray are computed as the logarithm of the ratio between the two channel expressions. The reason for taking the logarithm is to symmetrize between up and down-regulation. In the original scale, down-regulation, when the query target is less abundant than the reference target, is reduced in the interval $[0, 1]$, while up-regulation is spread over the interval $[1, \infty[$. Taking the logarithm places the origin of expression values at 0, values in $]-\infty, 0]$ correspond to down-regulation while values in $[0, \infty[$ to up-regulation.

2.4.2 Data normalisation

Any given set of microarray measurements contains variation from different sources. The expression levels may vary across samples due to differences in the quantity of mRNA, different sample processing, scanner calibration, etc. It is necessary to minimise the impact of these changes in order to retain only the biological variation, relevant to the study. This is the goal of normalisation.

The sources of variation are numerous and all are not very well understood, therefore it is difficult to model them explicitly. Instead, typically some general assumptions about invariance of certain quantities over samples are made. For example, in total-intensity normalisation it is assumed that the true average gene expression is constant across samples, in which case each array is scaled by its total estimated expression. An alternative approach is to assume that a large majority of genes are non-differentially expressed across samples. Consider two microarray samples, for example the query and reference samples on a single spotted microarray or two samples hybridised on separate oligonucleotide arrays. In a scatter plot of gene expression in sample one against sample two, the points should lie along the main diagonal under the assumption that most genes are non-differentially expressed. If, in the observed data, they do not, the actual relation between the samples can be estimated by fitting a line through the point cloud. The data can then be transformed so that this line lies along the main diagonal. In its most simple form this methodology rotates the point cloud, but more commonly the nonlinear LOWESS (LOcally WEighed Scatter plot Smooth) regression is used [78].

Under some circumstances none of the assumptions underlying the normalisation methods described above are valid. This is, for example, the case if the microarray contains relatively few probes, the majority of which are known to be involved in the biological process under study. In this case, normalisation is often based on the assumption that expression properties, like those described above, of a subset of the genes is invariant. This subset can be genes that are biologically known to have a constant expression, so called housekeeping genes. If no prior knowledge about invariant genes is available, a suitable subset can be selected using for example the invariant set algorithm [42].

2.4.3 Missing values

Spotted microarray data sets often come perforated with missing values. In a study with many samples it is quite likely that a rather large fraction of the genes contain at least one missing value across samples. High-level analysis methods usually do not allow missing values so in order not to throw away too much potentially valuable information, the missing values need to be filled in.

Different strategies to achieve this missing value imputation exists. A possible approach is to use the average expression value of the gene across samples. Another solution is adopted in K nearest neighbor imputation [68], where, if gene contains a missing value in a particular sample, the K genes with most similar gene expressions in the rest of the samples (where the corresponding sample has a value) are found and the missing value is replaced by a weighted average of the values in the other genes.

2.4.4 Filtering

Filtering is often applied to a microarray data set before high-level analysis. By discarding genes that have noisy expression levels and/or do not vary significantly over samples it is believed that the performance of subsequent high-level analysis increases.

Different rules are applied in order to filter genes. For example, the Affymetrix$^{\mathrm{TM}}$ oligonucleotide microarray platform provides detection p-values estimating the confidence of the signal presence of each gene and filtering can thus be based on these p-values by requiring that a gene should be significantly present in at least a certain number of samples. For spotted microarray data, one can use similar criteria based on the ratio between foreground and background intensities or the fraction of missing values.

Furthermore, variation filters can be applied, excluding genes who, for example, have a ratio between standard deviation and mean value below some threshold value.

2.5 High-level analysis

Once data had been properly preprocessed, the next step is to extract some biological meaning from it. Ideally, the study of expression matrix allows biologists to answer to biological questions; are there groups of genes allowing inference of a particular role for an unknown gene? What distinguishes the various samples of the experience? Can we find signatures in the expression of genes for various samples? This type of interpretation requires efficient and robust analytical methods. In that way, many mathematical tools have been developed or adapted.

These methods for information retrieval are usually based on classification algorithms wich distribute the set of objects into several subsets by minimizing variability within this subset.

We can easily distinguish two groups within the classification methods:

- *Unsupervised classification algorithms*: are automatic processes that separate observed data into distinct groups without any prior knowledge about the existing classes. These methods are commonly used in the interpretation of DNA-chips experimental results. Many softwares are now available, covering most of the time a number of methods such as:
  - hierarchical clustering
  - k-means algorithm
  - self organising map
  - order preserving submatrix
  - smooth clustering
  - plaid models
  - principal components analysis
  - singular values decomposition
  - spectral methods for graphs
  - fuzzy clustering
- *Supervised classification algorithm*: unlike unsupervised classification methods, these algorithms requires introduction of a priori knowledge to separate data into classes. The learning phase allows the system, from a known and annotated dataset, to infer rules which are then used for classification of other datasets. Among commonly used methods, we can cite:
  - k nearest neighbors
  - centroids analysis
  - linear discriminant analysis
  - support vector machines
  - neural networks
  - expectation-maximisation algorithm

  Results of these methods depend directly on the learning phase. The dimensions and noisy appearance of expression matrix may limit the application of these algorithms. The main risk is over-learning (bad generalisation of data properties, the system classifies perfectly learning samples but loses its power of prediction on other data set).

Given the current knowledge on function of genes, it is sometimes convienient to be able to drive tests without *a priori* knowledge on the data set, whether learning data or simply anotations (feature, function and role of genes...). In this study, we will first give details on mathematical approaches related to unsupervised classification methods, then we will present our approach called "semantic distillation", within the same category wich allow to characterize membership of all genes of a DNA-chip to the various biological samples from experience.

Chapter 3

Abstract data representation

3.1 Mathematical form of the dataset

Let $\mathbb{A}$ be a finite set of *attributes* (or contexts, or keywords) and $\mathbb{B}$ be a finite set of *documents* (or objects, or books). The dataset is a $|\mathbb{A}| \times |\mathbb{B}|$ matrix $\mathsf{X} = (x_{ab})_{a\in\mathbb{A},b\in\mathbb{B}}$ of real or complex elements, where $|\cdot|$ represents cardinality. The matrix element x_{ab} quantifies the level of expression of document b with respect to the attribute a. Several equivalent ways of representing the dataset will be given in the subsequent sections.

Example 3.1.1 (Gene expression in various tissular contexts) In one of the experiments we analysed $\mathbb{B}$ is a set of 12613 human genes and $\mathbb{A}$ a set of 12 tissular contexts. The matrix elements x_{ab} are real numbers encoding luminescence intensities (or their logarithms) of DNA array ultimately representing the level of expression of gene b in context a.

Example 3.1.2 (Liver-specific genes expression for patients at various stages of hepatic cancer) Here $\mathbb{B}$ is a set of 700 human genes known to be differentially expressed in hepatocytes and $\mathbb{A}$ a set of 14 patients. Elements x_{ab} are as above.

Example 3.1.3 (Library documents indexing) Let $\mathbb{B}$ be a set of books in a library and $\mathbb{A}$ a set of bibliographic keywords. The matrix elements x_{ab} can be $\{0, 1\}$-valued: if the term a is present in the book b then $x_{ab} = 1$ else $x_{ab} = 0$. A variant of this example is when x_{ab} are integer valued: if the term a appears k times in document b then $x_{ab} = k$. An additional variant of this example is the authoritative weighing of the index, i.e. we suppose that an authority weight $w(a) \in \mathbb{R}$ is provided for every term a, that multiplies all entries $w(a)x_{ab}, b \in \mathbb{B}$, so that the matrix elements become now real valued. The authority weight is meant to give larger weight to the most important and informative terms.

Example 3.1.4 (Students evaluation) Let $\mathbb{B}$ be a set of students and $\mathbb{A}$ a set of papers they gave. The matrix elements x_{ab} are real valued; x_{ab} is the mark the student b got in paper a.

The previous examples demonstrate the versatility of the method by keeping the formalism at an abstract level to apply indistinctively into various very different situations without any change. Note also that the assignment as set of documents or attributes is a matter of point of view; for instance, example 3.1.4 as it stands is convenient in evaluating students. Interchanging the role of sets $\mathbb{A}$ and $\mathbb{B}$ renders it adapted to the evaluation of teaching. As a rule of thumb, in biological applications, $|\mathbb{A}| \ll |\mathbb{B}|$.

Definition 3.1.5 Let $\mathbb{X}$ be an arbitrary set.

1. A function $k : \mathbb{X} \times \mathbb{X} \to \mathbb{C}$ verifying the symmetry property
$$\forall x, x' \in \mathbb{X} : k(x, x') = \overline{k(x', x)},$$
where $\overline{\text{overlining}}$ means complex conjugation, is called **linkage** and the pair $(\mathbb{X}, k)$ a **linked space**.
2. A linkage such that for all *finite* subsets $\mathbb{J} \subseteq \mathbb{X}$ and all collections of complex numbers $(\eta_x)_{x\in\mathbb{J}}$ the positivity property
$$\sum_{x,x'\in\mathbb{J}} \overline{\eta}_x k(x, x')\eta_{x'} \geq 0$$
holds is called a **positive kernel** on $\mathbb{X}$. The set of positive kernels on $\mathbb{X}$ will be denoted by $\mathcal{K}(\mathbb{X})$.
3. A non-negative real valued linkage verifying the diagonal vanishing
$$\forall x \in \mathbb{X} : k(x, x) = 0$$
is called a **(dis)similarity linkage**.
4. A dissimilarity linkage verifying the two additional properties of
   - separation: $k(x, x') = 0 \Rightarrow x = x'$,
   - and triangular inequality: $\forall x, x', x'' \in \mathbb{X} : k(x, x'') \leq k(x, x') + k(x', x'')$,

   is called a **distance**.

Remark: Obviously [k is a distance] $\Rightarrow$ [k is a (dis)similarity linkage] $\Rightarrow$ [k is a linkage], but no implication can be reversed. Therefore, we carefully distinguish between linkage, (dis)similarity linkage and distance in this text.

A linkage is supposed to express relationships among objects. For instance, if k is a dissimilarity linkage and x_1, x_2 and x_3 three distinct objects, on observing that $k(x_1, x_3) \geq k(x_1, x_2)$ we infer that objects x_1 and x_2 are “more similar” than x_1 and x_3. On the contrary, if k is a similarity linkage, $k(x_1, x_3) \geq k(x_1, x_2)$ means that x_1 and x_2 are "less similar" than x_1 and x_3.

The problems tackled in this paper can be roughly formulated as follows:

1. Given the dataset X, is there a natural way to define a linkage k on the set of objects giving rise to an associated positive kernel?
2. If yes, is it possible to represent objects by vectors on an appropriate ν-dimensional Hilbert space (i.e. represent the dataset X by a cloud of points in this vector space) so that k becomes a Euclidean distance among these points? Is there a minimal ν?
3. If k is a dissimilarity linkage, how to distribute objects into clusters according to their mutual dissimilarity?
4. Is there a way to express different meanings conveyed by documents in terms of probability measures on the Hilbert space? If yes, how disambiguation arises?

These questions have constantly being asked in the literature of various disciplines like artificial intelligence and information retrieval [69, 20, 5], cognitive sciences [27], mathematical linguistics [76], statistics [54, 38, 35, 3, 31], biochemistry [71], protein folding [72], bioinformatics [50, 58, 7], image segmentation [47, 64], large networks (like Internet) [17, 40, 41, 6] to mention only a few of them. They have received the most elaborated formulation in the theory of the quantum mechanical measurement [70, 59, 57] where the existence of an abstract mathematical substratum composed from a Hilbert space and a representation of objects in terms of vectors of the Hilbert space is made. This formalism can be usefully adapted to all other aforementioned disciplines with minor changes.

Therefore, before continuing, we present here a very general result playing a crucial role in establishing the existence of a representation space that serves as the mathematical playground for describing all the statistical methods relying on spectral properties (principal components analysis, support vector machines, graph Laplacians, random walk methods, and of course our method of semantic distillation). This theorem goes back to Mercer [48], Aronszajn [4], Gel'fand [28], and others.

Theorem 3.1.6 (Gel'fand) *Let $(\mathbb{X}, k)$ an arbitrary (possibly uncountable) linked space. Suppose further that for every finite or countable subset $\mathbb{J} \subset \mathbb{X}$ and every $\eta \in \mathbb{C}^{\mathbb{J}}$,we have*

$$\sum_{x,x' \in \mathbb{J}} \overline{\eta}_x k(x, x') \eta_{x'} \geq 0.$$

Then there exists a (not necessarily separable) Hilbert space $\mathcal{H}_{\mathbb{X}}$ and a representation $\Psi : \mathbb{X} \to \mathcal{H}_{\mathbb{X}}$ such that

1. *The family of vectors $\{\Psi(x), x \in \mathbb{J}\}$ is total in $\mathcal{H}_{\mathbb{X}}$, and*

2. $k(x,y) = \langle\, \Psi(x) \,|\, \Psi(y) \,\rangle$ *where* $\langle \cdot | \cdot \rangle$ *denotes the scalar product of* $\mathcal{H}_{\mathbb{X}}$.

The pair $(\mathcal{H}_{\mathbb{X}}, \Psi)$*, unique up to unitary isomorphisms, is called the* ***Gel'fand pair*** *associated with the linkage* k.

Remark: This theorem has a very intuitive meaning providing us with a nice geometrical picture: every time there exists a linkage on a set of objects that is associated with a positive kernel k, there exist an essentially unique complex vector space and a unique mapping of objects into vectors such that the linkage among two objects can be expressed as a scalar product among their representing vectors. Since all sets of objects in this article are finite, we shall prove the theorem 3.1.6 only in the particular case where $\mathbb{X}$ is a finite set. We give here a probabilistic proof that can be extended almost unchanged to cover the infinite case [53].

Proof of theorem 3.1.6 (in the case of finite sets): In the finite case, without loss of generality, we can assume that $\mathbb{X} = \mathbb{J}$ in the sequel. Since k defines a positive Hermitean form $\kappa : \mathbb{C}^{\mathbb{X}} \times \mathbb{C}^{\mathbb{X}} \to \mathbb{C}$ by $\kappa(\eta, \eta') = \sum_{x,x' \in \mathbb{X}} \overline{\eta}_x k(x,x') \eta_{x'}$, there exists a $|\mathbb{X}| \times |\mathbb{X}|$ matrix C such that the matrix $K = (k(x,x'))_{x,x' \in \mathbb{X}}$ representing the form in the canonical basis can be written as $K = CC^\dagger$, where the symbol $C^\dagger$ means the Hermitean adjoint (complex conjugated and transposed) of C. Let $(\alpha_x)_{x \in \mathbb{X}}$ and $(\beta_x)_{x \in \mathbb{X}}$ be $2|\mathbb{X}|$ independent random variables identically distributed according to a centred reduced Gaussian. Define for all $x \in \mathbb{X}$, $\gamma_x = (\alpha_x + i\beta_x)$ and $\boldsymbol{\xi} \in \mathbb{C}^{\mathbb{X}}$ by $\xi_x = \sum_{x' \in \mathbb{X}} C_{xx'} \gamma_{x'}$. By construction, $\mathbb{E}(\xi_x) = 0$, while $\mathbb{E}(\boldsymbol{\xi} \otimes \boldsymbol{\xi}^\dagger) = CC^\dagger = S$. Therefore, there exists a probability space $(\Omega, \mathcal{F}, \mathbb{P})$ and a Hilbert space $L^2(\Omega, \mathcal{F}, \mathbb{P}; \mathbb{C})$ such that for all $x \in \mathbb{X}$, we have $\mathbb{E}(\xi_x) = 0$ and $\mathbb{E}(\overline{\xi}_x \xi_{x'}) = k(x,x')$. We then define $\mathcal{H}_{\mathbb{X}} = \mathsf{span}\{\xi_x, x \in \mathbb{X}\}$ the (trivially closed) subspace of $L^2(\Omega, \mathcal{F}, \mathbb{P}; \mathbb{C})$ and $\Psi(x) = \xi_x$. □

Let k be a positive kernel, i.e. $k \in \mathcal{K}(\mathbb{X})$, on a topological space $\mathbb{X}$ equipped with its Borel σ-algebra and a σ-finite measure μ. Denote by $\mathcal{H} = L^2(\mathbb{X}, \mathcal{X}, \mu; \mathbb{C})$. Under a continuity condition assumed on k, Mercer's theorem [48] (see also [22], vol. 2, page 1088), states that there exists a positive compact operator $K : \mathcal{H} \to \mathcal{H}$, defined through its action on $\xi \in \mathcal{H}$:

$$K\xi(x) = \int_{\mathbb{X}} k(x,y)\xi(y)\mu(dy).$$

Compactness and positivity of K imply that there exists a family of non-negative eigenvalues $(\lambda_n)_{n \in \mathbb{N}}$ that can be assumed ordered $\lambda_1 \geq \lambda_2 \geq \ldots \geq 0$ with $\lim_n \lambda_n = 0$ and a total set of corresponding eigenvectors $(\psi_n)_{n \in \mathbb{N}}$, such that the operator K admits the spectral decomposition $K = \sum_{n \in \mathbb{N}} \lambda_n |\, \psi_n \,\rangle\langle\, \psi_n \,|$, i.e. $k(x,y) = \sum_{n \in \mathbb{N}} \lambda_n \overline{\psi}_n(y) \psi_n(x)$. Suppose now that the set $\mathbb{X}$ is finite or countable. The Hilbert space whose existence is guaranteed by the previous theorem 3.1.6 can be realised as follows:

- Let $\Psi : \mathbb{X} \to \mathbb{C}^{\mathbb{X}}$ be defined by $\Psi(x) = k(\cdot, x)$. Obviously, $\Psi(\mathbb{X})$ is a vector space.
- Endow it with a scalar product by satifying the reproducing kernel property $\langle\, k(\cdot, x) \,|\, k(\cdot, x') \,\rangle = k(x, x')$.

- Complete the space $\Psi(\mathbb{X})$ in the corresponding norm to get a Hilbert space $\mathbb{H}_{\mathbb{X}}$.

In the case of a dissimilarity linkage over a finite set $\mathbb{B}$ the theorem 3.1.6 admits a more geometrical form formulated by Shoenberg [62] and going back the Fréchet who proposed [26] a reformulation of the von Neumann complex (Hermitean) scalar product on a Hilbert space by an equivalent symmetric one [32].

The problem of Schoenberg can be formulated as follows. Suppose that $(\mathbb{B}, k)$ is a space linked by dissimilarity instead of being merely linked. What are the necessary and sufficient conditions that there is a collection of points $\mathcal{Y} = (\Psi(b))_{b\in\mathbb{B}}$ lying in a Euclidean space $\mathbb{R}^\nu$ but not in $\mathbb{R}^{\nu-1}$ so that $k(b,b') = \|\Psi(b)-\Psi(b')\|$? In other words, what are the necessary and sufficient conditions so that there exists a cloud of points $\mathcal{Y}$, living in $\mathbb{R}^\nu$, so that the linkage is expressed as the lengths of the edges of $\mathcal{Y}$, while this is impossible if the cloud is projected in any proper subspace of $\mathbb{R}^\nu$? If this is the case, the map $\mathbb{B} \ni b \mapsto \Psi_b \in \mathbb{R}^\nu$ is a representation of the set of points incorporating all the available relationships described by the linkage into the distances among the points of the cloud $\mathcal{Y} = \Psi(\mathbb{B})$. Moreover, ν is the minimal dimension of this representation to faithfully represent the linkage, any trial to represent it into a lower dimensional space will cause an information loss.

Theorem 3.1.7 (Schoenberg [62]) *Let $(\mathbb{B}, k)$ be a document space linked by dissimilarity and $\kappa : \mathbb{R}^{\mathbb{B}} \times \mathbb{R}^{\mathbb{B}} \to \mathbb{R}$ the bilinear form defined for all $\eta, \eta' \in \mathbb{R}^{\mathbb{B}}$ by $\kappa(\eta,\eta') = \sum_{b,b'\in\mathbb{B}} \eta_b k(b,b')\eta'_{b'}$. Let $\Psi : \mathbb{B} \to \mathbb{R}^\nu$ be a representation of the documents to $\mathbb{R}^\nu$. Denote by β the symmetric matrix with matrix elements $\beta(b,b') = \|\Psi(b)-\Psi(b')\|^2$ on $\mathbb{R}^{|\mathbb{B}|}$ and by H the hyperplane $H = \{\eta = (\eta_b)_{b\in\mathbb{R}^{\mathbb{B}}} \in \mathbb{R}^{\mathbb{B}} : \sum_{b\in\mathbb{B}} \eta_b = 0\}$. The dissimilarity linkage k verifies $k(b,b') = \beta(b,b')$ if and only if the restriction of $\kappa\restriction_H$ is negative semi-definite. The minimal ν for such a representation to be possible is the rank of $\kappa\restriction_H$.*

Proof: Suppose first that there exists a cloud of points $(\Psi_b)_{b\in\mathbb{B}}$ in $\mathbb{R}^n$ for some n sufficiently large realising the dissimilarity linkage by $k(b,b') = \|\Psi_b - \Psi_{b'}\|^2$. Then, obviously, $k(b,b') = \|\Psi_b\|^2 + \|\Psi_{b'}\|^2 - 2\langle \Psi_b \mid \Psi_{b'} \rangle$. Therefore,

$$\kappa\restriction_H(\eta,\eta') = -2\langle V(\eta) \mid V(\eta') \rangle,$$

where $V(\eta) = \sum_{b\in\mathbb{B}} \eta_b \Psi_b$, so that the quadratic form $\kappa\restriction_H(\eta,\eta) = -2\|V(\eta)\|^2 \leq 0$. Now $\nu = \mathsf{rank}(\kappa\restriction_H) = \dim V(\mathbb{R}^{\mathbb{B}})$ is the minimal dimension of the space carrying the cloud of points and realising the dissimilarity linkage.

Suppose conversely that k is a dissimilarity linkage on $\mathbb{B}$ and that the corresponding bilinear form, restricted on the hyperplane, $\kappa\restriction_H$ is negative semi-definite with $\mathsf{rank}(\kappa\restriction_H) = \nu$. Particularise a point of $\mathbb{B}$ by calling it 0 and write $\mathbb{B} = \{0\} \cup \mathbb{B}_0$, where $\mathbb{B}_0 = \mathbb{B} \setminus \{0\}$. Denote by $(\epsilon_b)_{b\in\mathbb{B}}$ the canonical basis of $\mathbb{R}^{\mathbb{B}}$; obviously, $(v_b = \epsilon_b - \epsilon_0, b \in \mathbb{B}_0)$ is a basis of H. Denote by k' the matrix representing $\kappa\restriction_H$ in the basis $(v_b)_{b\in\mathbb{B}_0}$ (recalling that the dissimilarity linkage k

represents the form κ in $(\epsilon_b)_{b\in\mathbb{B}}$). Then,

$$\begin{aligned}\forall b, b' \in \mathbb{B}_0 : k'(b,b') &= \frac{1}{2}\kappa(v_b, v_{b'}) \\ &= \frac{1}{2}(k(b,b') - k(0,b') - k(b,0) + k(0,0)) \\ &= -\frac{1}{2}(k(0,b) + k(0,b') - k(b,b')).\end{aligned}$$

Now, since k' is supposed negative semi-definite of rank ν, there exists a matrix $R \in \mathcal{M}_{\nu\times|\mathbb{B}_0|}(\mathbb{R})$ such that $k' = -R^t R$ where R^t denotes the transposed matrix of R. The matrix can be viewed as a collection of $|\mathbb{B}_0|$ column vectors $[\Psi_b]_{b\in\mathbb{B}_0}$ with $\Psi_b \in \mathbb{R}^\nu$. Augment this set by adjoining the vector $\Psi_0 = 0 \in \mathbb{R}^\nu$. Then, for all $b, b' \in \mathbb{B}_0$, we have:

$$\begin{aligned}\|\Psi_b - \Psi_{b'}\|^2 &= \|\Psi_b\|^2 + \|\Psi_{b'}\|^2 - 2\langle\, \Psi_b \,|\, \Psi_{b'} \,\rangle \\ &= -k'(b,b) - k'(b',b') + 2k'(b,b') \\ &= \frac{1}{2}\,(2k(0,b) + 2k(0,b') - 2(k(0,b) + k(0,b') - k(b,b'))) \\ &= k(b,b').\end{aligned}$$

Additionally, $\|\Psi_b\|^2 = \|\Psi_b - \Psi_0\|^2 = -k'(b,b') = \frac{1}{2}(k(0,b) + k(0,b) - k(b,b)) = k(0,b)$. Hence for all $b, b' \in \mathbb{B}$, we have established that $k(b,b') = \|\Psi_b - \Psi_{b'}\|^2$. □

3.2 The space of document representations

The mere representation of the dataset in terms of a matrix with real or complex numerical entries conveys already the idea of an underlying vector space; the rows (or columns) of the dataset representing points within this vector space. This idea, already present — even not already explicitly explained — in the traditional methods, and thoroughly formalised in theorem 3.1.6, will be exploited here. Note however that in this subsection, we have not yet a linkage on the set of objects. Our purpose is to introduce such a linkage based on the dataset X.

For $\mathbb{A}$ and $\mathbb{B}$ as in the previous subsection, we define $\mathcal{H}_\mathbb{A}$, to be the real or complex free vector space over $\mathbb{A}$, i.e. elements of $\mathbb{A}$ serve as indices of an orthonormal basis of $\mathcal{H}_\mathbb{A}$. Therefore, the complete dataset X can be represented as the collection of $|\mathbb{B}|$ vectors $|\, \Xi_b \,\rangle = \sum_{a\in\mathbb{A}} x_{ab} |\, a \,\rangle \in \mathcal{H}_\mathbb{A}$, with $b \in \mathbb{B}$ and where $|\, a \,\rangle$ represents the element of the orthonormal basis of the free vector space corresponding to the attribute a. We use here Dirac's notation to represent vectors, linear forms and projectors on this space (see any book on quantum mechanics or [56] for a freely accessible document and [69] for the use of this notation in information retrieval).

Definition 3.2.1 Let $\mathbb{A}$ be a finite set of attributes, $\mathbb{B}$ a finite set of documents and $\mathsf{X} = (x_{ab})_{a\in\mathbb{A}, b\in\mathbb{B}}$ the dataset of expressions. Let $\mathcal{H}_\mathbb{A}$ be the free Hilbert space over $\mathbb{A}$ and $\Xi : \mathbb{B} \to \mathcal{H}_\mathbb{A}$ defined by

$$\Xi(b) \equiv |\, \Xi_b \,\rangle = \sum_{a\in\mathbb{A}} x_{ab} |\, a \,\rangle,\ \forall b \in \mathbb{B}.$$

Then the pair $(\mathcal{H}_{\mathbb{A}}, \Xi)$ is a **representation** of the set of documents with attribute values expressed by the dataset X.

The vector $|\,\Xi_b\,\rangle$ contains all available experimental information on document b in various contexts indexed by the attributes a; it can be thought as a convenient bookkeeping device of the data $(x_{ab})_{a\in\mathbb{A}}$, in the same way a generating function contains all the information on a sequence as formal power series.

In this language, documents are identified as particular vectors of the vector space that can be combined through addition and scalar multiplication. However, the space of concepts contains infinitely more vectors than the documents. In particular, it contains the attributes as particular vectors (orthonormal basis); it contains also all linear combinations of subsets of attributes, spanning specific subspaces. Concepts are precisely identified with subspaces of $\mathcal{H}_{\mathbb{A}}$.

3.3 The concept space and the metrisation of the set of documents

The vector space $\mathcal{H}_{\mathbb{A}}$ is equipped with a scalar product defined for every two vectors $|\,\psi\,\rangle = \sum_{a\in\mathbb{A}} \psi_a|\,a\,\rangle$ and $|\,\psi'\,\rangle = \sum_{a\in\mathbb{A}} \psi'_a|\,a\,\rangle$ by $\langle\,\psi\,|\,\psi'\,\rangle = \sum_{a\in\mathbb{A}} \overline{\psi}_a\psi'_a$, where $\overline{\psi}_a$ denotes the complex conjugate[1] of ψ_a (it coincides with ψ_a if it is real). Equipped with this scalar product, the vector space $\mathcal{H}_{\mathbb{A}}$ becomes a complex $|\mathbb{A}|$-dimensional Hilbert space. The scalar product induces a Hilbert norm on the space, denoted by $\|\cdot\|$.

Note that since the space $\mathcal{H}_{\mathbb{A}}$ is finite dimensional (as will be all Hilbert spaces that will be considered in this article), any vector subspace is a closed Hilbert subspace. Hence it will be identified with the corresponding orthogonal projection. We are now in position to define the important notion of the space of concepts.

Definition 3.3.1 Let $\mathcal{H}$ be a finite-dimensional Hilbert space. The family of all orthogonal projections $\mathcal{P}(\mathcal{H})$ (equivalently the family of all Hilbert subspaces of $\mathcal{H}$) is called **space of concepts**.

Remark: The space $\mathcal{H}$ in the previous definition must be thought as the representation space for some set of documents within a given dataset.

In the sequel we introduce also **rays** on the Hilbert space i.e. normalised vectors. Since the dataset X does not in principle verify any particular numerical constraints, rays are constructed by dividing vectors by their norms. We use the symbol $|\,\xi_b\,\rangle = |\,\Xi_b\,\rangle / \||\,\Xi_b\,\rangle\|$ to denote the ray associated with vector $|\,\Xi_b\,\rangle$.

[1] We consistently use a scalar product that is antilinear in the first argument and linear in the second.

The Hilbert space structure on $\mathcal{H}_{\mathbb{A}}$ allows a natural geometrisation of the space of documents by equipping it with a linkage derived from the Hilbert norm. Let $(\mathcal{H}_{\mathbb{A}}, \Xi)$ be a Hilbert space representation of the dataset X. Denote $|\,\xi_b\,\rangle = \frac{|\,\Xi_b\,\rangle}{\||\,\Xi_b\,\rangle\|}$ and define $k : \mathbb{B} \times \mathbb{B} \to \mathbb{R}$ by

$$k(b, b') = \||\,\xi_b\,\rangle - |\,\xi_{b'}\,\rangle\|^2.$$

Then k is a dissimilarity linkage on $\mathbb{B}$.

What is important here is not the precise form of the dissimilarity structure of $(\mathbb{B}, k)$; several other dissimilarity linkages can be introduced, not necessarily compatible with the scalar product. Let us mention for example the whole family of dissimilarity linkages K_p derived from the l^p distances ($p \in [1, +\infty]$) on $\mathcal{H}_{\mathbb{A}}$ by

$$K_p(b, b') = [\sum_{a \in \mathbb{A}} |x_{ba} - x_{b'a}|^p]^{2/p} = \|\Xi_b - \Xi_{b'}\|_p^2, \text{ for } p < +\infty$$

and

$$K_\infty(b, b') = (\sup_{a \in \mathbb{A}} |x_{ba} - x_{b'a}|)^2 = \|\Xi_b - \Xi_{b'}\|_\infty^2$$

or

$$k_p(b, b') = \|\xi_b - \xi_{b'}\|_p^2.$$

Note that the special cases K_2 and k_2 from the above family are the only dissimilarity linkages compatible with the Hilbert structure.

Once the space of documents has been linked by dissimilarity, $(\mathbb{B}, k)$, we can define the notion of **set linkage**, denoted again by k, between non-empty subsets of B_1 and B_2 of $\mathbb{B}$ as being the degree of linkage between documents agglomerated in B_1 and documents agglomerated in B_2. There is no canonical way of defining a set linkage. To give a flavour of linkages often used in the literature, we give a few examples.

Example 3.3.2 Let $(\mathbb{B}, k)$ be a space linked by dissimilarity and B and B' non-empty subsets of $\mathbb{B}$.

$$\begin{aligned} k(B, B') &= \min\{k(b, b'), b \in \mathbb{B}, b' \in \mathbb{B}'\}, \\ k(B, B') &= \max\{k(b, b'), b \in \mathbb{B}, b' \in \mathbb{B}'\}, \\ k(B, B') &= \frac{1}{|B||B'|} \sum_{b \in \mathbb{B}, b' \in \mathbb{B}'} k(b, b'), \end{aligned}$$

are possible set linkages. Note that only the first one is a dissimilarity! When elements of $\mathbb{B}$ are represented in $(\mathcal{H}_{\mathbb{A}}, \Xi)$ and the vector space is equipped with an arbitrary distance d, then

$$k(B, B') = \left[d\left(\frac{1}{|B|} \sum_{b \in B} \Xi_b, \frac{1}{|B'|} \sum_{b' \in B'} \Xi_{b'} \right) \right]^2$$

is a further example of set dissimilarity linkage.

Chapter 4

Set-theoretical methods of raw data clustering

Clustering is a generic term meaning dividing the documents into few groups of similar documents. The previous sentence is far from being a definition since the notion of clustering is manifestly ill-defined: how dissimilarity is defined, quantified and detected? how many are "few" groups?

Several methods[1] have been developed so far. Some of them (hierarchical clustering, l-means, principal components analysis, singular value decomposition, etc.) are extensively tested on small data sets; it remains still an open problem whether are still adapted to huge datasets generated by DNA arrays. Other classification or clustering methods have been proposeed (order preserving sub-matrix, smooth clustering, plaid models, fuzzy methods, etc.).

Very often some of the above methods are included as built-in parts of popular software suits used by biologists to analyse data. Therefore, for completeness and in order to avoid the black-box syndrome, a short presentation of the most commonly used algorithms is made in the sequel.

4.1 Hierarchical clustering

Starting from a set of documents $\mathbb{B}$ and a set linkage dissimilarity k defined for every pair of non-empty subsets of $\mathbb{B}$, **hierarchical clustering** is a method to produce an indexing structure on the family of subsets of $\mathbb{B}$, called a **dendrogamme**. The dendrogamme is a finite rooted tree whose root indexes the set $\mathbb{B}$ while leaves index all singleton subsets $\{b\}, b \in \mathbb{B}$. Edges of the tree connect vertices indexing a parent cluster with all its children. An hierarchical clustering algorithm can be **agglomerative**, if it starts from singletons and at each steps subsumes most similar clusters into a unique cluster, or **divisive**, if

[1] In the litterature, the method is usually referred as k-means. Since we have reserved the letter k for denoting linkage, we call it l-means

it starts from the set $\mathbb{B}$ and at each step splits clusters into their most dissimilar subclusters. Algorithm 1 below describes the agglomerative version; an example of the divisive version is given in section 7.1.

```
Data: Set of documents $\mathbb{B}$, set linkage $k$
Result: A Dendrogramme
initialise
Clusters $\leftarrow \{\{b\}, b \in \mathbb{B}\}$;
$L \leftarrow 0$;
Edges$_L \leftarrow \{\}$;
Vertices$_L \leftarrow$ Clusters;
while $|Clusters| > 1$ do
    $L \leftarrow L + 1$;
    $(C_1, C_2) = \arg\min\{k(C, C'), C, C' \in$ Clusters$\}$;
    $C = C_1 \cup C_2$;
    Clusters $\leftarrow$ Clusters $\setminus \{C_1, C_2\}$;
    Clusters $\leftarrow$ Clusters $\cup \{C\}$;
    Vertices$_L \leftarrow$ Clusters;
    Edges$_L \leftarrow$ Edges $\cup \{(C_1, C), (C_2, C)\}$;
end
$L_{\max} \leftarrow L$;
Dendrogramme $\leftarrow$ (Vertices$_L$, Edges$_L)_{L=0,\ldots,L_{\max}}$.
```

Algorithm 1: AgglomerativeHierarchicalClustering

Note that a possible ambiguity can arise in the step

$$(C_1, C_2) = \arg\min\{k(C, C'), C, C' \in \mathsf{Clusters}\}$$

of the algorithm in case several pairs saturate the minimum. The ambiguity is resolved either by randomly choosing the pair (C_1, C_2) among all minimising ones, or by any standard deterministic or non-deterministic disambiguation procedure. The produced dendrogrammes are topologically different but semantically equivalent.

4.2 *l*-means clustering

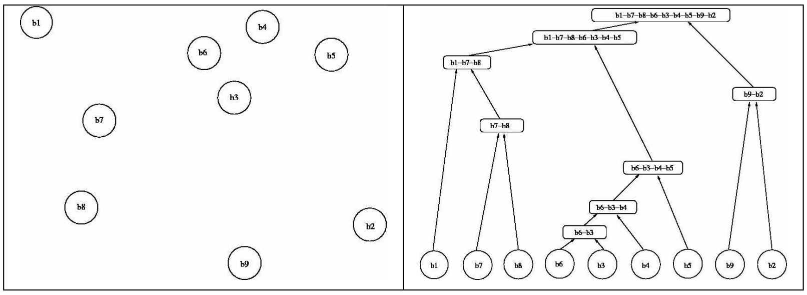
Prior to presenting the method of l-means clustering, we need a definition.

Definition 4.2.1 Let $(\mathbb{X}, d)$ be a metric space and l an integer larger than 1. Let $\mathcal{X} = \{x_1, \ldots, x_l\}$ be a family of l arbitrary distinct points of $\mathbb{X}$. The **Voronoi tesselation** associated with $\mathcal{X}$ is a splitting of $\mathbb{X}$ into l subsets $V_i, i = 1, \ldots, l$, the **Voronoi cells**, defined by

$$V_i = \{y \in \mathbb{X} : d(x_i, y) \leq d(x_j, y) \text{ for all } j \neq i\}, i = 1, \ldots, l.$$

We denote by $\mathcal{V}_{\mathcal{X}}$ the Voronoi tesselation associated with $\mathcal{X}$.

Table 4.1: Example of agglomerative hierarchical clustering. Raw data are represented on the left side and linked by Euclidian distance. Right side of the figure represents the dendogramme resulting from agglomerative hierarchical clustering.



Note that the Voronoi tesselation is not a partition of $\mathbb{X}$ since boundaries of the cells belong to the intersection of two adjacent ones.

To perform a l-means clustering, we work with the Hilbert space representation of the documents, i.e. with the family $\mathcal{X} = \{\Xi(b) \in \mathcal{H}_{\mathbb{A}}, b \in \mathbb{B}\}$. The Hilbert space is either naturally metrised with its Hilbert norm or with some *ad hoc* metric d. Assume that for some fixed integer $l > 1$, we are given l vectors $\Psi_i \in \mathcal{H}_{\mathbb{A}}, i = 1, \ldots, l$ and denote by $\mathcal{Y}$ this family. Let $\mathcal{V}_{\mathcal{Y}}$ be the correpsonding Voronoi tesseleation of $\mathcal{H}_{\mathbb{A}}$. For all cells V_i of the tesselation $\mathcal{V}_{\mathcal{Y}}$, we compute their **barycentres** relative to $\mathcal{X}$ [2], $c_i(\mathcal{X})$, defined by

$$c_i \equiv c_i(\mathcal{X}) = \frac{1}{N_i} \sum_{b \in \mathbb{B}} \Xi(b) \mathbb{1}_{V_i}(\Xi(b)), \ i = 1, \ldots, l$$

where N_i is the cardinality of the set of documents belonging to the cell V_i. The **objective cost function**, denoted by $\mathsf{ObjCostFun}$, of the tesselation $\mathcal{V}_{\mathcal{Y}}$ with respect to the family $\mathcal{X}$ is then defined by

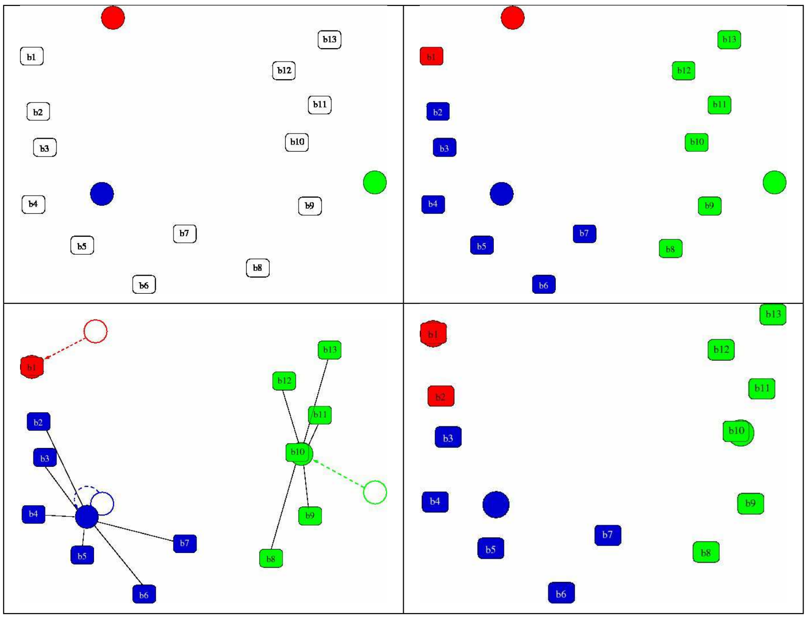
$$\mathsf{ObjCostFun}(\mathcal{V}_{\mathcal{Y}}, \mathcal{X}) = \sum_{i=1}^{l} d(\Psi_i, c_i(\mathcal{X})).$$

The l-means algorithm is then a standard iterative search scheme aiming in determing a family $\mathcal{Y}$ minimising the objective function. The minimisation procedure can be performed by any valid and/or efficient algorithm. The sought clustering is then performed by lumping documents belonging to the Voronoi cells determined by the optimal $\mathcal{Y}$ that has been found.

Remark: Note that, simple as it sounds, updating the family $\mathcal{Y}$ to the set of the previously obtained barycentres and starting afresh not always converge to an optimum clustering. More elaborated minimisation procedures are required to minimise the multi-valley objective function by efficiently searching the space $\mathcal{H}_{\mathbb{A}}^{l}$ of putative barycentres (like simulated annealing, genetic algorithms, etc.).

[2] Computer scientists use to call them centroids.

Table 4.2: Illustration of the l-means algorithm. The first figure (up left) shows the initial randomized centroids and a number of points. In the second, (up right) points are associated with the nearest centroid. Then (third figure, down left) the centroids are moved to the center of their respective clusters. Steps 2 and 3 are repeated until a suitable level of convergence has been reached (down right).



```
Data: Set $\mathcal{X} = \{\Xi(b), b \in \mathbb{B}\}$ representing the documents,
      $l$ the sought number of clusters,
      ObjCostFun the objective cost function,
      $\mathcal{Y} = \{\Psi_i, i = 1, \ldots, l\}$ an initial family of putative
      barycentres of the clustering,
      Minimisation an ad hoc minimisation algorithm.
Result: $\mathcal{Y} = \{\Psi_i, i = 1, \ldots, l\}$ the family of optimal barycentres,
        $l$ subsets $C_i, i = 1, \ldots, l$ tesselating $\mathbb{B}$ and having $\mathcal{Y}$ as
        family of barycentres.
while $\mathcal{Y}$ not optimal do
    assign Voronoi tesselation $\mathcal{V}_{\mathcal{Y}}$ corresponding to family $\mathcal{Y}$ of putatitve
    barycentres;
    compute barycentres of the tesselation;
    compute ObjCostFun$(\mathcal{V}_{\mathcal{Y}}, \mathcal{X})$;
    use Minimisation to propose new $\mathcal{Y}$.
end
assign optimal Voronoi tesselation $\mathcal{V}$ to optimal family $\mathcal{Y}$ of barycentres;
for $i = 1, \ldots, l$ do
    $C_i = \mathcal{X} \cap V_i$.
end
```

Algorithm 2: l-MeansClustering

Table 4.3: Illustration of the importance of choosing initial centroids. This table and table 4.4 shows different clustering results according to the initial centroids position.

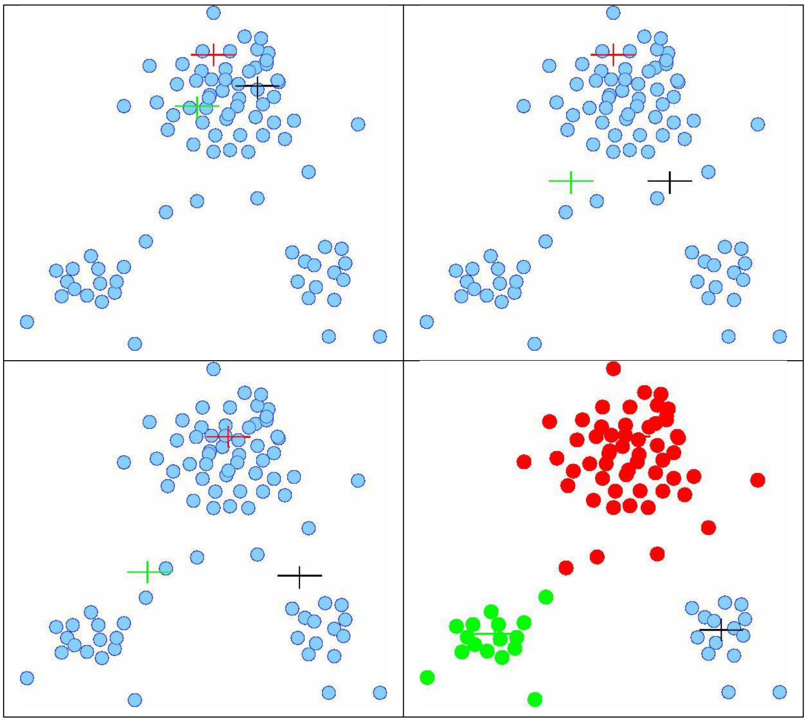
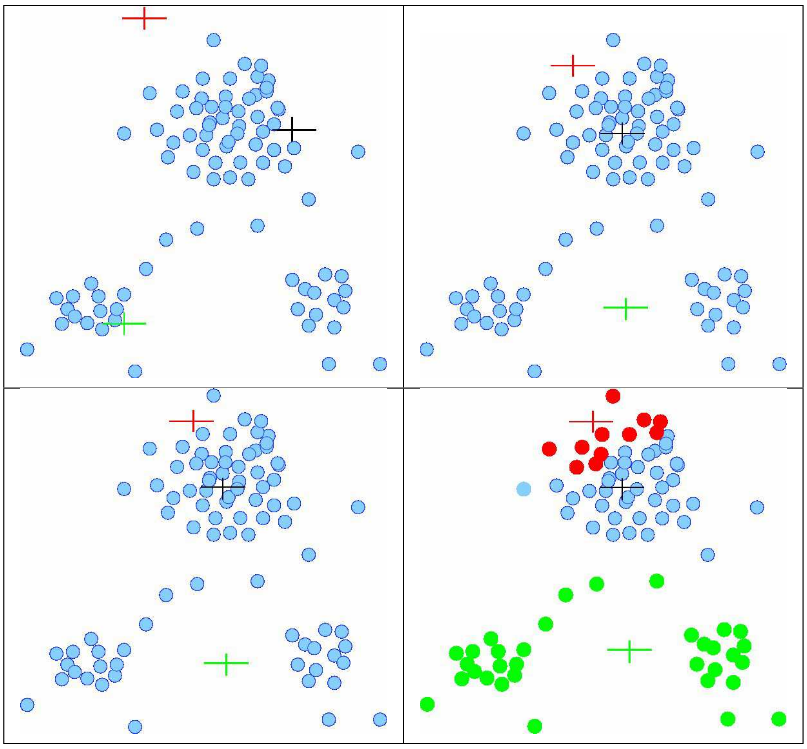


Table 4.4: Illustration of the importance of choosing initial centroids. This table and table 4.3 shows different clustering results according to the initial centroids position. In this case, the result is not the optimum clustering.



4.3 Order preserving submatrix

Suppose that the dataset matrix X is real-valued. For every document $b \in \mathbb{B}$, the expression levels induce a natural ordering of the attributes $\mathbb{A}$. Let $A \subseteq \mathbb{A}$ and $B \subseteq \mathbb{B}$ be two non-empty sets. Two documents b and b' of B induce the same ordering in A if for any distinct $a, a' \in A$, the differences $x_{ab} - x_{a'b}$ and $x_{ab'} - x_{a'b'}$ have the same sign (or are both 0).

An **order preserving submatrix** of X corresponds to subsets A of attributes and B of documents such that the expression levels within attributes A of all documents in B have the same ordering. We denote $\mathsf{X}\restriction_{A,B}$ the so-defined order preserving submatrix. The problem is to find maximal (with respect to cardinality) subsets A and B so that $X\restriction_{A,B}$ is order-preserving. The documents in B share obviously a more sophisticated dissimilarity property than mere linkage dissimilarity. However, since this problem is known, [67], to be NP-complete, no efficient algorithm to solve it is available.

It is algorithmically easier to fix $A \subseteq \mathbb{A}$ and seek the largest subset $B \subseteq \mathbb{B}$ such that $X\restriction_{A,B}$ is order-preserving. This task can been solved in linear time.

4.4 Smooth clustering

There is a whole family of possible smooth clusterings for datasets. Let $A \subseteq \mathbb{A}$ and $B \subseteq \mathbb{B}$ be two fixed non-empty subsets of the attribute and the document sets. The first step consists in a pre-processing of the dataset so that for all $b \in B$ and $a \in A$ change the value x_{ab} into $x'_{ab} = x_{ab} - \frac{1}{|B|}\sum_{b' \in B} x_{ab'}$. The smooth clustering depends then on the possible definitions of the **smoothness score** relative to the subsets A and B, denoted by $\mathsf{SmoothScore}(A, B)$, defined on $\mathcal{P}(\mathbb{A}) \times \mathcal{P}(\mathbb{B})$, where $\mathcal{P}(\cdot)$ denotes the family of subsets. (To avoid notational burden, we can define the score to take the value 0 whenever one of its arguments is void.) The scores proposed in the literature [67] are

$$\begin{aligned} \mathsf{SmoothScore}(A,B) &= \max_{a \in A}\left(\max_{b \in B} |x'_{ab}|\right), \\ \mathsf{SmoothScore}(A,B)' &= \max_{a \in A}\left(\sum_{b \in B} |x'_{ab}|^2\right). \end{aligned}$$

Note that reordering rows and columns of the dataset X' so that elements in A and B become contiguous, defines a $|A| \times |B|$ submatrix. We propose therefore two additional scores, induced by the operator norms on the space of matrices (see, for instance, [34] pp 294–295), and defined by

$$\begin{aligned} \mathsf{SmoothScore}(A,B)_1 &= \max_{a \in A}\left(\sum_{b \in B} |x'_{ab}|\right), \\ \mathsf{SmoothScore}(A,B)_\infty &= \max_{b \in B}\left(\sum_{a \in A} |x'_{ab}|\right). \end{aligned}$$

The smooth clustering problem can be formulated as follows:

```
Data: X: the $|\mathbb{A}| \times |\mathbb{B}|$ matrix representation of the dataset,
      SmoothScore any of the smooth scores defined above,
      $\theta > 0$ a smoothness threshold, and
      $A \subseteq \mathbb{A}$ a given set of attributes.
Result: The maximal $B \subseteq \mathbb{B}$ such that $\mathsf{SmoothScore}(A, B) \leq \theta$.
assign $\mathsf{X}' \leftarrow$ the column re-centred matrix X;
determine $\mathcal{F} = \{(A, B) : A \subseteq \mathbb{A}, B \subseteq \mathbb{B}, \mathsf{SmoothScore}(A, B) \leq \theta\}$;
determine maximal $B$ such that $(A, B) \in \mathcal{F}$.
```

Algorithm 3: SmoothClustering

The smooth clustering problem is known to be NP-hard. It can be viewed as an assignement, to every subset A of attributes, of the maximal subset $B = B(A)$ of documents considered as the most pertinent to the characteristics determined by A. Pertinence is measured by the function SmoothScore.

4.5 Plaid model

Imagine that real expression levels x_{ab} in the dataset are encoded into colours. The dataset matrix X will then be viewed as a collection of coloured cells disposed on the plane. Gene expressions are traditionally using a binary colour code (green, red) meaning (expressed, suppressed). The encoding proposed here is a refinement of this encoding to the case of larger colour sets; levels of expressions, in the fashion of heights on a geographical map, will be encoded in terms of the enlarged colour code. The idea behind the model is that we can rearrange rows and columns of the dataset as a collection of contiguous sub-blocks of the same colour. Documents in these sub-blocks are clustered as similar.

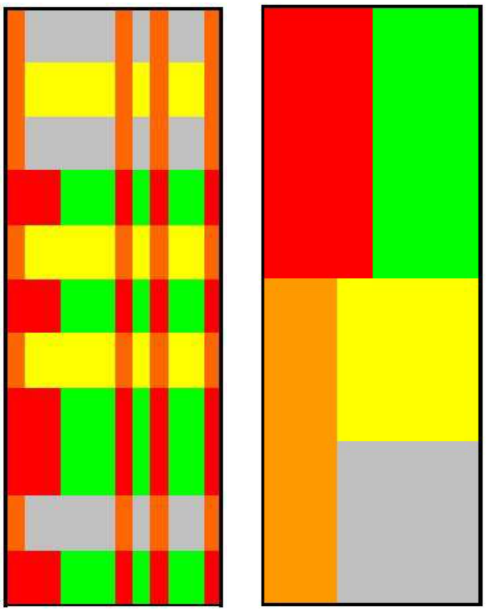
An ideal re-ordering consists in partitioning the set of attributes $\mathbb{A}$ into L disjoint subsets $A_l, l = 1, \ldots, L$ and the set of documents $\mathbb{B}$ into M disjoint subsets $B_m, m = 1, \ldots, M$ and define a colour matrix $C = (c_{lm})_{l=1,\ldots,L;m=1,\ldots,M}$. The assumption for this ideal case is then that the expression matrix X can be rewritten as

$$x_{ab} = c^{(0)} + \sum_{l=1}^{L} \sum_{m=1}^{M} c_{lm} \mathbb{1}_{A_l}(a) \mathbb{1}_{B_m}(b),$$

where $c^{(0)}$ is a background colour. It is evident that if the rows and columns are permuted so that those corresponding to every A_l and B_m are contiguous, then the dataset acquires a colouring in contiguous blocks of similar documents according to the attributes determined by the subsets A_l.

The plaid model is a modification of the previous ideal situation where instead of a partitioning of the set $\mathbb{A} \times \mathbb{B}$ induced by the Cartesian product of partitioning of the sets $\mathbb{A}$ and $\mathbb{B}$, a covering of the set $\mathbb{A} \times \mathbb{B}$ by M not necessarily disjoint patches. More precisely, we assume that there are subsets

Table 4.5: Illustration of sharp plaid model. The left figure shows the raw data. Clustered data are represented on the right.



$F_m, m = 1, \dots, M$ of $\mathbb{A} \times \mathbb{B}$ such that $\cup_{m=1}^{M} F_m = \mathbb{A} \times \mathbb{B}$ while the symmetric difference $F_m \triangle F_{m'}$ is "small" when $m \neq m'$. We assume further that there is a family of colours $c_m, m = 1, \dots, M$ and a family $\mathsf{Y}^{(m)} = (y^{(m)})_{ab}, m = 1, \dots, M$ of $|\mathbb{A}| \times |\mathbb{B}|$ matrices such that

$$y_{ab}^{(m)} = \begin{cases} c_m & \text{if } (a,b) \in F_m \\ 0 & \text{otherwise.} \end{cases}$$

If the original dataset matrix X can be re-expressed as

$$x_{ab} = c_0 + \sum_{m=1}^{M} y_{ab}^{(m)},$$

where c_0 is a background colour, we say that the dataset admits a plaid clustering. Instead of seeking a precise plaid clustering, the minimisation of a $\mathsf{PlaidCost}$ function is sought. For a given dataset X and family of patch matrices $\mathsf{Y}^{(m)}, m = 1, \dots, M$, a convenient cost function is

$$\mathsf{PlaidCost}(\mathsf{X}, (\mathsf{Y}^{(m)})_{m=1,\dots,M}) = \sum_{a \in \mathbb{A}} \sum_{b \in \mathbb{B}} \left(x_{ab} - c_0 - \sum_{m=1}^{M} y_{ab}^{(m)} \right)^2 .$$

The plaid clustering problem is to determine, for given M, the family of uniform matrices $\mathsf{Y}^{(m)}, m = 1, \dots, M$ minimising the $\mathsf{PlaidCost}$ function; it is known to be NP-hard hence no efficient algorithm to solve it is known.

Chapter 5

Spectral methods of clustering

5.1 Raw probabilistic and statistical methods

Beneath this class of methods lies the idea that the dataset X is a noisy version of an ideal dataset. Thus all column vectors $X^{(b)}$ of X are supposed to be random vectors.

5.1.1 Karhunen-Loève decomposition and principal components analysis

Principal components analysis was first introduced in [54] and further developed and popularised in [35] in the context of mathematical statistics; more accessible references are [38, 31]. The method is best understood as a discrete version of the so called Karhunen-Loève decomposition [36, 44] of a stochastic process. Let us recall briefly this decomposition.

Let $(\Omega, \mathcal{F}, \mathbb{P})$ be an abstract probability space and

$$\mathfrak{h} = \{\xi : \Omega \to \mathbb{C} \text{ s.t. } \mathbb{E}|\xi|^2 < \infty\} = L^2(\Omega, \mathcal{F}, \mathbb{P}; \mathbb{C})$$

the space of square integrable complex random variables on it. The space $\mathfrak{h}$ becomes a complex Hilbert space when equipped with the scalar product $\langle \xi \,|\, \xi' \rangle = \mathbb{E}(\overline{\xi}\xi') = \int \overline{\xi}(\omega)\xi'(\omega)\mathbb{P}(d\omega)$. Let $(\mathbb{X}, \mathcal{X}, \mu)$ be a measure space with $\mathbb{X}$ a compact topological space equipped with its Borel σ-algebra $\mathcal{X}$ and a finite regular measure μ. A centred square integrable stochastic process indexed by $\mathbb{X}$ is a measurable mapping $X : \mathbb{X} \to \mathfrak{h}$ such that for all $x \in \mathbb{X}$, we have $\mathbb{E}X_x = 0$. Let $k : \mathbb{X} \times \mathbb{X} \to \mathbb{C}$ be defined by $k(x, y) = \mathsf{Cov}_X(x, y) = \mathbb{E}(X_x \overline{X}_y) = \langle X_y \,|\, X_x \rangle$. We shall assume in the sequel that the covariance is continuous. Then obviously, k is a linkage on $\mathbb{X}$. For any finite $\mathbb{J} \subset \mathbb{X}$ and any $\eta \in \mathbb{C}^{\mathbb{J}}$,

$$\begin{aligned} \sum_{x,x' \in \mathbb{J}} \overline{\eta}_x k(x, x')\eta_{x'} &= \mathbb{E}(\overline{V}(\eta)V(\eta)) \\ &= \mathsf{Var}(V(\eta)) \geq 0, \end{aligned}$$

where $V(\eta) = \sum_{x\in\mathbb{J}} \eta_X X_x$. Hence, k is a positive kernel, i.e. $k \in \mathcal{K}(\mathbb{X})$. Denote by $\mathcal{H} = L^2(\mathbb{X}, \mathcal{X}, \mu; \mathbb{C})$. Under the continuity condition assumed on k, it induces, thanks to Mercer's theorem (see [22], vol. 2, page 1088), a positive compact operator $K : \mathcal{H} \to \mathcal{H}$, defined through its action on $\psi \in \mathcal{H}$:

$$K\psi(x) = \int_{\mathbb{X}} k(x,y)\psi(y)\mu(dy).$$

Compactness and positivity of K imply that there exists a family of non-negative eigenvalues $(\lambda_n)_{n\in\mathbb{N}}$ that can be assumed ordered $\lambda_1 \geq \lambda_2 \geq \ldots \geq 0$ with $\lim_n \lambda_n = 0$ and a total set of corresponding eigenvectors $(\psi_n)_{n\in\mathbb{N}}$, such that the operator K admits the spectral decomposition $K = \sum_{n\in\mathbb{N}} \lambda_n | \psi_n \rangle\langle \psi_n |$, i.e. $k(x,y) = \sum_{n\in\mathbb{N}} \lambda_n \overline{\psi}_n(y)\psi_n(x)$. Now, since $X : \mathbb{X} \to \mathfrak{h}$, it follows that for all $\omega \in \Omega$, $X(\omega) \in \mathcal{H}$. As the set of eigenvectors (ψ_n) is total, we can decompose $X(\omega) = \sum_{n\in\mathbb{N}} \zeta_n(\omega)\psi_n$, i.e. $X_x(\omega) = \sum_{n\in\mathbb{N}} \zeta_n(\omega)\psi_n(x)$.

Theorem 5.1.1 (Karhunen-Loève decomposition) *Let $(\mathbb{X}, \mathcal{X}, \mu)$ be a measure space composed by a compact topological space $\mathbb{X}$ equipped with its Borel σ-algebra $\mathcal{X}$ and a finite regular measure μ. Let $X = (X_x)_{x\in\mathbb{X}}$ be a centred (complex) stochastic process on a probability space $(\Omega, \mathcal{F}, \mathbb{P})$, indexed by $\mathbb{X}$ and having finite and continuous covariance function $\mathsf{Cov}_X : \mathbb{X} \times \mathbb{X} \to \mathbb{C}$. With the covariance as positive kernel on $\mathbb{X}$, consider the nuclear compact positive operator K on $\mathcal{H} = L^2(\mathbb{X}, \mathcal{X}, \mu; \mathbb{C})$. Let $(\lambda_n)_{n\in\mathbb{N}}$ be the sequence of eigenvalues of K monotonically decreasing to 0 and $(\psi_n)_{n\in\mathbb{N}}$ the sequence of associated normalised eigenvectors. Then*

1. *The stochastic process X admits a unique decomposition $X = \sum_{n\in\mathbb{N}} \zeta_n\psi_n$.*
2. *The random variables (ζ_n) are square integrable and centred verifying $\mathbb{E}(\overline{\zeta}_n\zeta_{n'}) = \lambda_n\delta_{nn'}$.*

Sketch of the proof: Since for every $\omega \in \Omega$ the realisation $X(\omega)$ belongs in $\mathcal{H}$ and (ψ_n) is a total system of vectors, we have the decomposition $X(\omega) = \sum_{n\in\mathbb{N}} \zeta_n(\omega)\psi_n$ (in the $\mathcal{H}$ norm). We have further $\langle \psi_n \,|\, X(\omega) \rangle = \zeta_n(\omega)$ establishing thus the unicity of the decomposition. Using this decomposition, compute for $x, y \in \mathbb{X}$,

$$\begin{aligned} k(x,y) &= \mathbb{E}(X_x\overline{X}_y) \\ &= \sum_{n,n'\in\mathbb{N}} \mathbb{E}(\zeta_n\overline{\zeta}_{n'})\psi_n(x)\overline{\psi}_{n'}(y). \end{aligned}$$

Now, the eigenvector condition for ψ_m implies:

$$\begin{aligned} \lambda_m\psi_m(x) &= K\psi_m(x) \\ &= \sum_{n,n'\in\mathbb{N}} \mathbb{E}(\zeta_n\overline{\zeta}_{n'})\psi_n(x) \int_{\mathbb{X}} \overline{\psi}_{n'}(y)\psi_m(y)\mu(dy) \\ &= \sum_{n\in\mathbb{N}} \mathbb{E}(\zeta_n\overline{\zeta}_m)\psi_n(x). \end{aligned}$$

Taking finally the scalar product with ψ_m, we get:

$$\lambda_m = \mathbb{E}(\zeta_m\overline{\zeta}_m) = \mathsf{Var}(\zeta_m).$$

Now, since $\zeta_n = \langle\, \psi_n \,|\, X \,\rangle$,

$$\begin{aligned}\mathbb{E}(\overline{\zeta_n}\zeta_{n'}) &= \int_{\mathbb{X}}\int_{\mathbb{X}} \mathbb{E}(X_x\overline{X}_y)\overline{\psi}_n(x)\psi_{n'}(y)\mu(dx)\mu(dy)\\ &= \langle\, \psi_n \,|\, K\psi_{n'} \,\rangle\\ &= \lambda_{n'}\delta_{nn'}.\end{aligned}$$

□

Remark: The total variance of the process reads:

$$\int_{\mathbb{X}} \mathbb{E}(X_x\overline{X}_x)\mu(dx) = \sum_{n\in\mathbb{N}} \lambda_n = \mathsf{tr}(K).$$

Since the eigenvalues are positive and in decreasing order, the above formula has a very intuitive meaning. If instead of using an arbitrary basis of $\mathcal{H}$ to decompose X, we use the basis of the eienvectors of K, the projection of X to the space spanned by the first eigenvector has a dominant contribution to the total variance equal to λ_1. If we project X to the space spanned by the two first eigenvectors the variance of the projection is $\lambda_1 + \lambda_2$, and so on. Thus adding new dimensions to the projection space improves the approximation of the total variance but these improvements are smaller and smaller since λ_n decreases monotonically to 0.

The principal components analysis can be viewed merely as the discrete Karhunen-Loève decomposition of a stochastic process defined on a finite set $\mathbb{X} = \mathbb{A}$ equipped with the counting measure μ. The stochastic process is now a mapping $X : \mathbb{A} \to \mathfrak{h}$. One of the basic assumptions of the method of principal component analysis, often consciously or unconsciously hidden in most expositions, is that the dataset $\mathsf{X} = (X^{(b)})_{b\in\mathbb{B}}$ can be viewed as a collection of $|\mathbb{B}|$ independent and identically distributed copies $X^{(b)} : \mathbb{A} \to \mathfrak{h}$ of the process X. This allows estimating the covariance Cov_X by the empirical covariance estimator[1].

Since the Karhunen-Loève decomposition is formulated for centred processes while the dataset X is not assumed to verify any obvious centring, we start by transforming it into a modified dataset Y that is empirically centred. In other words, we introduce the empirical mean column vector $m = (m_a)_{a\in\mathbb{A}}$ and define the transformed data $y_{ab} = x_{ab} - m_a$ for all a and b. This gives rise to the transformed dataset matrix $\mathsf{Y} = \mathsf{X} - m \otimes 1_{\mathbb{B}}^t$, where $1_{\mathbb{B}}$ is the vector of $\mathbb{C}^{\mathbb{B}}$ whose components are all equal to 1. The empirical covariance matrix estimator of X is

$$\hat{K}(a,a') = \widehat{\mathsf{Cov}}_X(a,a') = \frac{1}{|\mathbb{B}|-1}\sum_{b\in\mathbb{B}} y_{ab}\overline{y}_{a'b} = \frac{1}{|\mathbb{B}|-1}(\mathsf{Y}\mathsf{Y}^\dagger)_{aa'}.$$

[1] The requirement of independent and identically distributed copies can be somehow relaxed, nevertheless, it is essential that the empirical covariance estimator converges to the covariance of the process and this latter constraint imposes essentially strong stationarity and mixing properties.

The matrix $\hat{K}$ is obviously positive semi-definite and self-adjoint hence admits a complete orthogonal set of eigenvectors $(\psi_n)_{n=1,\ldots,|\mathbb{A}|}$ and corrsponding eigenvalues $(\lambda_n)_{n=1,\ldots,|\mathbb{A}|}$, assumed ordered in decreasing order $\lambda_1 \geq \ldots \geq \lambda_{|\mathbb{A}|} \geq 0$. Subsequently, $\hat{K} = \sum_{n=1}^{|\mathbb{A}|} \lambda_n | \psi_n \rangle\langle \psi_n |$. We have further that the total empirical variance of the process X verifies

$$\frac{1}{|\mathbb{B}|-1} \sum_{a\in\mathbb{A}} \sum_{b\in\mathbb{B}} |y_{ab}|^2 = \sum_{n=1}^{|\mathbb{A}|} \lambda_n = \mathsf{tr}(\hat{K}).$$

Now the process X can be decomposed into the basis of eigenvectors $X = \sum_{n=1}^{|\mathbb{A}|} c_n | \psi_n \rangle$. For any threshold $\theta \in [0,1]$, define

$$l \equiv l_\theta = \inf\{N = 1, \ldots, |\mathbb{A}| : \sum_{n=1}^{N} \frac{\lambda_n}{\mathsf{tr}(\hat{K})} \geq \theta\}$$

and $P_l = \sum_{n=1}^{l} | \psi_n \rangle\langle \psi_n |$ the orthogonal projector to the space spanned by the eigenvectors associated with the l dominant eigenvalues (obviously, $P_{|\mathbb{A}|} = \mathbb{1}_{\mathfrak{h}}$). Then $P_l X$ is an l-dimensional approximation of the process explaining a proportion θ of the total variance. Similarly, instead of considering the $|\mathbb{A}|$-dimensional cloud of $|\mathbb{B}|$ points of the dataset, the l-dimensional cloud $(P_l X^{(b)})_{b\in\mathbb{B}}$ explains a θ part of the total empirical variance. We can also restrict the spectral decomposition of $\hat{K}$ to subspace spanned by the first l most important eigenvectors by defining:

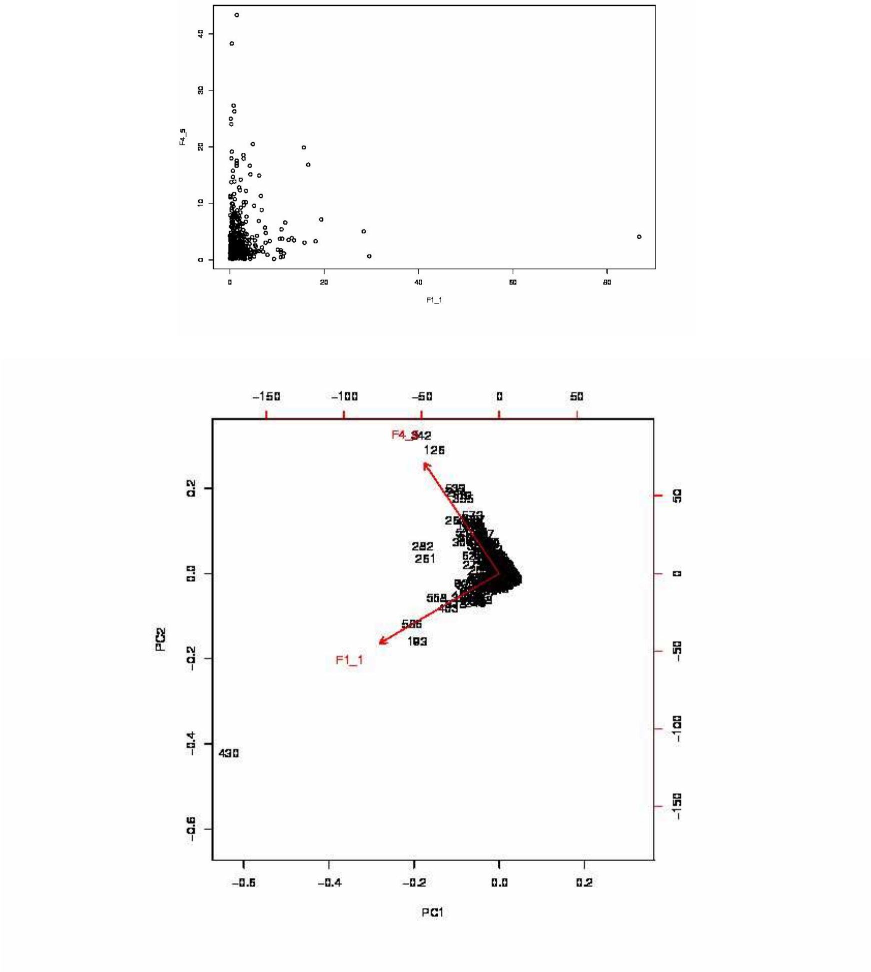
$$\hat{K}_l \equiv P_l \hat{K} P_l = \frac{1}{|\mathbb{B}|-1} P_l \mathsf{Y}\mathsf{Y}^\dagger P_l = \sum_{n=1}^{l} \lambda_n | \psi_n \rangle\langle \psi_n |.$$

In conclusion, principal components analysis is a representation of the dataset in a space of dimension lower than the cardinality of attributes, optimally explaining a given percentage, θ, of the total variance. It is therefore expected that this lower dimensional representation will be more easily exploited to regroup documents in similar clusters.

5.1.2 Singular value decomposition and latent semantic indexing

In the sequel, $\mathfrak{M}_{m,n}(\mathbb{C})$ denotes the set of matrices with complex coefficients having m rows and n columns; the algebra of square $n \times n$ complex matrices will be denoted $\mathfrak{M}_n(\mathbb{C})$. The set of Hermitean matrices is denoted by $\mathsf{Her}\mathfrak{M}_n(\mathbb{C})$ and the open cone of positive semi-definite Hermitean matrices by $\mathsf{HerPos}\mathfrak{M}_n(\mathbb{C})$. By $\mathsf{GL}(n,\mathbb{C})$ we denote the group of invertible $n \times n$ complex matrices and by $\mathsf{U}(n)$ its subgroup of unitary matrices. It is well known that the polar decomposition of a complex number $z = r\exp(i\theta)$ has a matrix analogue. Namely, for every $Z \in \mathsf{GL}(n,\mathbb{C})$ there exist uniquely determined matrices $R \in \mathsf{HerPos}\mathfrak{M}_n(\mathbb{C})$ and $U \in \mathsf{U}(n)$ such that $Z = RU$. This decomposition is called **polar** and establishes a homeomorphism between $\mathsf{GL}(n,\mathbb{C})$ and $\mathsf{HerPos}\mathfrak{M}_n(\mathbb{C}) \times \mathsf{U}(n)$. This result can be further extended to the case of not necessarily invertible matrices that can even be rectangular instead of square.

Table 5.1: Illustration of principal component analysis. The top figure shows raw data data in a bi-dimensional projection according to standard basis vectors of ℝ. The bottom figure shows the same data according to the basis of the 2 first eigenvectors.



Theorem 5.1.2 (Singular value decomposition) *Let $Z \in \mathfrak{M}_{m,n}(\mathbb{C})$, with $r = \mathsf{rank}(Z) \leq \min(m,n)$. Then there exist matrices $U \in \mathsf{U}(m)$, $V \in \mathsf{U}(n)$, and a block quasidiagonal real matrix $D = \begin{pmatrix} S & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \in \mathfrak{M}_{m,n}(\mathbb{R})$, where $S = \mathsf{diag}(s_1, \ldots, s_r)$ and for all $i = 1, \ldots, r$, we have $s_i \in]0, \infty[$, such that $Z = UDV^\dagger$. The numbers $s_1, \ldots, s_r$ are uniquely determined (up to permutations) and are called the* ***singular values*** *of Z.*

Remark: Note that the above factorisation is far from being unique. This can be easily seen by counting the dimensions of the vector spaces involved in the decomposition [63].

Proof of theorem 5.1.2: Both $ZZ^\dagger$ and $Z^\dagger Z$ are positive semi-definite and $\mathsf{rank}(ZZ^\dagger) = \mathsf{rank}(Z^\dagger Z) = r$. Further their spectra share the same positive eigenvalues (with the same multiplicity), differing only by the multiplicity of the eigenvalue 0, i.e. $\mathsf{spec}(ZZ^\dagger) = (s_1^2, \ldots, s_r^2, 0, \ldots, 0)$ has cardinality m while $\mathsf{spec}(Z^\dagger Z) = (s_1^2, \ldots, s_r^2, 0, \ldots, 0)$ has cardinality[2] n. Since $\mathsf{rank}(ZZ^\dagger) = \mathsf{rank}(Z^\dagger Z) = r = \mathsf{rank}(Z)$ and $\mathsf{im}(ZZ^\dagger) \subseteq \mathsf{im}(Z)$, it follows that $\mathsf{im}(Z) = \mathsf{im}(ZZ^\dagger)$. Now, $ZZ^\dagger \in \mathsf{Her}\mathfrak{M}_m(\mathbb{C})$; therefore $\ker(ZZ^\dagger) = \mathsf{im}(Z)^\perp = \ker(Z^\dagger)$. Consequently, $\mathbb{C}^m = \mathsf{im}(ZZ^\dagger) \oplus \ker(Z^\dagger)$, implying that there exists an orthonormal basis of $\mathbb{C}^m$ consisting of the eigenvectors $U_R = [u_1, \ldots, u_r]$ of $ZZ^\dagger$, associated with the positive eigenvalues $s_1^2, \ldots, s_r^2$, completed by a basis $U_K = [u_{r+1}, \ldots, u_m]$ of $\ker(Z^\dagger)$. Forming the unitary matrix $U = [u_1, \ldots, u_m] = [U_R, U_K]$, we have for the respective blocks: $ZZ^\dagger U_R = U_R S^2$ and $Z^\dagger U_K = 0$. On defining $V_R = Z^\dagger U_R S^{-1} \in \mathfrak{M}_{n,r}(\mathbb{C})$ we have $V_R^\dagger V_R = I_r$, i.e. the column vectors $V_R = [v_1, \ldots, v_r]$ form an orthonormal family of $\mathbb{C}^n$ that can be completed to an orthonormal basis $[v_1, \ldots, v_n]$ by adjoining the vectors $V_K = [v_{r+1}, \ldots, v_n]$ belonging to $\ker Z$. Writing blockwise $V = [V_R, V_K]$, and observing that $ZV_K = 0$ and $Z^\dagger U_K^\dagger = 0$, we obtain

$$U^\dagger Z V = \begin{pmatrix} U_R^\dagger Z V_R & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix},$$

with $U_R^\dagger Z V_R = S$. □

In the sequel, we assume that the singular values of Z are in decreasing order $s_1 \geq \ldots \geq s_r > 0$. For definitness, we assume also that $n > m$. For any integer $l = 1, \ldots, r$ let $\chi_l(i) = 1$ if $i \leq l$ and 0 otherwise. Define then $S_l = \mathsf{diag}(s_1\chi_l(1), \ldots, s_r\chi_l(r))$. Obviously, $S_r = S = \mathsf{diag}(s_1, \ldots, s_r)$. The construction made in the proof of the theorem 5.1.2 can be generalised, by replacing the block quasidiagonal matrix D by

$$D_l = \begin{pmatrix} S_l & \mathbf{0}_{l,m-l} & \mathbf{0}_{l,n-m} \\ \mathbf{0}_{m-l,l} & \mathbf{0}_{m-l,m-l} & \mathbf{0}_{m-l,n-m} \end{pmatrix},$$

where $\mathbf{0}_{p,q}$ means a matrix in $\mathfrak{M}_{p,q}(\mathbb{C})$ whose elements are identically 0. With this definition, we define then $Z_l = UD_lV^\dagger = \sum_{k=1}^{l} s_k | u_k \rangle\langle v_k | \in \mathfrak{M}_{m,n}(\mathbb{C})$ where $(u_k)_{k=1,\ldots,r}$ are the r first column vectors in $\mathbb{C}^m$ of U and $(v_k)_{k=1,\ldots,r}$ are the r first column vectors in $\mathbb{C}^n$ of V.

[2] To incorporate multiplicity, we consider the spectrum as an ordrered list of eigenvalues, possibly with repetitions.

Lemma 5.1.3 *If Z_l is as above and the singular values are decreasingly ordered, then $\|Z_l\|_2^2 = s_1^2$.*

Proof: We have $\|Z_l\|_2^2 = \sup_{x\in\mathbb{C}^n;\|x\|_2=1} \|Z_l x\|_2$. Hence, due to the orthonormality of the vectors (u_k),

$$\begin{aligned}
\|Z_l x\|^2 &= \sum_{k=1}^{l}\sum_{k'=1}^{l} s_k s_{k'} \langle\, x \,|\, v_{k'} \,\rangle\langle\, u_{k'} \,|\, u_k \,\rangle\langle\, v_k \,|\, x \,\rangle \\
&= \sum_{k=1}^{l} s_k^2 \langle\, x \,|\, v_k \,\rangle\langle\, v_k \,|\, x \,\rangle \\
&\leq s_1^2 \sum_{k=1}^{n} \langle\, x \,|\, v_k \,\rangle\langle\, v_k \,|\, x \,\rangle \\
&= s_1^2.
\end{aligned}$$

Furhter, the inequality becomes an equality if and only if $x = v_1$, proving thus the lemma. □

Proposition 5.1.4 *If $l \leq r = \mathsf{rank}(Z)$ and the singular values of Z are decreasingly ordered, then*

$$\inf_{W\in\mathfrak{M}_{m,n}(\mathbb{C});\mathsf{rank}(W)=l} \|Z - W\|_2 = \|Z - Z_l\|_2 = s_{l+1}.$$

Proof: Since $U^\dagger Z_l V = S_l$, it follows that $\mathsf{rank}(Z_l) = l$. Now, $U^\dagger(Z - Z_l)V = \mathsf{diag}(0, \ldots, 0, s_{l+1}, \ldots, s_r)$, and since the biggest singular value of $Z - Z_l$ is s_{l+1}, it follows that $\|Z - Z_l\|_2 = s_{l+1}$. Suppose now that $W \in \mathfrak{M}_{m,n}(\mathbb{C})$ and $\mathsf{rank}(W) = l \leq \min(n, m)$. Since $\mathbb{C}^n = \ker W \oplus \mathsf{im}(W)$, there exist orthonormal vectors $w_1, \ldots, w_{n-l} \in \mathbb{C}^n$ such that $\ker W = \mathsf{span}(w_1, \ldots, w_{n-l})$. Denoting $v_1, \ldots, v_{l+1}$ the first column vectors of V, a simple dimensional argument implies that the space $F = \mathsf{span}(w_1, \ldots, w_{n-l}) \cap \mathsf{span}(v_1, \ldots, v_{l+1}) \neq \{0\}$. We can therefore find a vector $y \in F$ with $\|y\|_2 = 1$. Since $Wy = 0$ and $Zy = \sum_{k=1}^{l+1} s_k |\, u_k \,\rangle\langle\, v_k \,|\, y \,\rangle$, we have

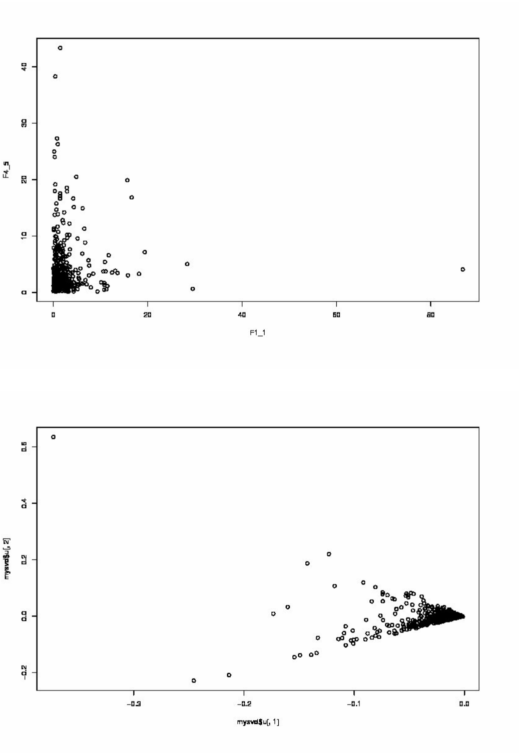
$$\|Z - W\|_2^2 \geq \|(Z - Z_l)y\|_2^2 = \|(Z - Z_l)y\|_2^2 = \sum_{k=1}^{l+1} s_k^2 \langle\, y \,|\, v_k \,\rangle\langle\, v_k \,|\, y \,\rangle \geq s_{l+1}^2.$$

□

Let $\mathsf{X} = (x_{ab})_{a\in\mathbb{A},b\in\mathbb{B}}$ be a matrix representing the occurrence of keyword a in document b or any other variant introduced in the example 3.1.3.

The set $\mathbb{A}$ is considered as the thesaurus containing all possible keyowrds in defining the database. A keyword $a \in \mathbb{A}$ is expressed as a vector $|\, a \,\rangle \in \mathbb{C}^{\mathbb{A}}$ having a coordinate 1 at position a and 0 in all other places. A query is identified to the collection of keywords $Q \subseteq \mathbb{A}$ it contains or, equivalently, to a vector $q = \sum_{a\in Q} |\, a \,\rangle$. A document is also identified with the collection of the keywords it contains, thus to a set $W \subseteq \mathbb{A}$, or, equivalently, with a vector $w = \sum_{a\in W} |\, a \,\rangle$.

Table 5.2: Illustration of singular value decomposition. Top figure shows the original dataset plot. Bottom figure shows its singular value decomposition.



Now there are many different ways to express a given concept (synonymy) and a term may have mulitple meanings (polysemy). The dataset matrix X is supposed to express a more subtle relationship between terms and documents than the Hamming distance comparison between vectors q and w. If $r = \mathsf{rank}\mathsf{X}$, **latent semantic indexing**, for any integer $l \leq r$, is an algorithm seeking to take advantage of the singular value decomposition $\mathsf{X}_l = UD_lV^\dagger = \sum_{k=1}^{l} |\, u_k \,\rangle\langle\, v_l \,|$ to map the original relationships among documents into l linearly independent factors [11, 23, 41, 40, 33]. Instead of performing a mere matching between q and w, we first transform vectors to a query vector $\mathcal{Q} = D_l^{-1}U^\dagger q$ and to a document vector $\mathcal{W} = D_l^{-1}U^\dagger w$ before comparing them, for instance, by using a dissimilarity determined by the scalar product $\langle\, \mathcal{Q} \,|\, \mathcal{W} \,\rangle$.

Remark: Note that singular value decomposition can be thought as a "square root" of the principal components analysis. As a matter of fact, suppose that the dataset is centred. Then,

$$\mathsf{X}\mathsf{X}^\dagger = \sum_{k=1}^{r}\sum_{k'=1}^{r} s_k s_{k'} |\, u_k \,\rangle\langle\, v_k \,|\, v_{k'} \,\rangle\langle\, u_{k'} \,| = \sum_{k=1}^{r} s_k^2 |\, u_k \,\rangle\langle\, u_k \,|,$$

expression to be compared with the spectral decomposition of the empirical covariance.

5.1.3 Support vector machines

The **support vector machine algorithm** is a supervised learning algorithm for classifying data. It can be useful when a training set of documents and their precise function (annotation) are already given. In the biological applications we have in mind in this work, such a precise information is lacking, or, more precisely, we don't wish to rely on any previous existing information for clustering our objects. Therefore, such a method cannot be applied directly in our situation. Nevertheless, we briefly describe it here because it illustrates some aspects of the Hilbert space representation and can prove useful, in the cases precise annotation is available, to incorporate learning functions into our method.

Let $(\mathbb{B}, k)$ be a space linked by dissimilarity where $k \in \mathcal{K}(\mathbb{B})$. Then there exists a *real* Hilbert space $\mathcal{H}$ yielding a natural representation $\psi : \mathbb{B} \to \mathcal{H}$ such that the set of objects $\mathbb{B}$ is mapped into a cloud of points $\mathcal{Y} = \{\psi_b \in \mathcal{H}, b \in \mathbb{B}\}$. Assume further that there exists an a priori known precise classifier $c : \mathbb{B} \to \{-1, 1\}$ such that every object b is assigned a value determining whether it possesses (1) or not (-1) a given property. The support vector machine algorithm relies on the (assumed) linear separability of the dataset, i.e. we suppose that there exist a hyperplane H of $\mathcal{H}$ such that the set $c^{-1}(\{-1\})$ is represented by vectors lying on the "left" of the hyperplane H. The algorithm determines the optimal hyperplane separating the objects; when a new object is added in the dataset, the position of its representing vector with respect to the hyperplane H uniquely determines whether it is classified as possessing the property.

In this subsection, denote by $\mathbb{X} = \mathbb{R}^\nu$ the Hilbert space $\mathbb{H}$ of representation of documents. The dataset is then represented by the cloud of points $\mathcal{Y} = \{x_b, b \in \mathbb{B}\}$, where $x_b = \psi(b) \in \mathbb{R}^\nu$. For every non zero vector $w \in \mathbb{X}$ and any real number r, denote by $H = H_{w,r}$ the affine hyperplane defined by $H = \{x \in \mathbb{X} : \langle\, w \,|\, x \,\rangle + r = 0\}$. It is evident that for every real $\lambda \neq 0$, the hyperplanes $H_{\lambda w, \lambda r}$ and $H_{w,r}$ coïncide. Hence, we define the **canonical hyperplane** with respect to the cloud of points $\mathcal{Y}$, the hyperplane determined by w and r such that $\min_{x \in \mathcal{Y}} |\langle\, w \,|\, x \,\rangle + r| = 1$. Note that if w and r defines a canonical hyperplane H for the cloud $\mathcal{Y}$, the distance of H to the closest datum in $\mathcal{Y}$ is $\min_{x \in \mathcal{Y}} \left| \langle\, \frac{w}{\|w\|} \,|\, x \,\rangle + \frac{r}{\|w\|} \right| = \frac{1}{\|w\|}$. The norm $\|w\|$ is called the **margin** of the separation by the canonical hyperplane $H_{w,r}$.

The optimisation problem we have to solve is to determine the optimal canonical separating hyperplane, i.e. the canonical hyperplane maximising the margin of separation and separating the data correctly. This task corresponds to minimising $\|w\|^2$ subject to the constraint $c(b)[\langle\, w \,|\, x_b \,\rangle + r] \geq 1$, for all $b \in \mathbb{B}$. Introducing the array of Lagrange multipliers $\alpha = (\alpha_b)_{b \in \mathbb{B}}$ with $\alpha_b \geq 0$ we must minimise the Lagrangean

$$L(w,r,\alpha;\mathcal{Y}) = \frac{1}{2}\|w\|^2 - \sum_{b\in\mathbb{B}} \alpha_b \left(c(b)[\langle\, w \,|\, x_b \,\rangle + r] - 1\right)$$

with respect to the variables w and r and maximise it with respect to α. At the extremum, we have $\frac{\partial}{\partial r} L(w,r,\alpha;\mathcal{Y}) = 0$ and $\frac{\partial}{\partial w_i} L(w,r,\alpha;\mathcal{Y}) = 0$ for all $i = 1,\ldots,\nu$, yielding $\sum_{b\in\mathbb{B}} \alpha_b c(b) = 0$ and $w = \sum_{b\in\mathbb{B}} \alpha_b c(b) x_b$. Substituting back into L, we must maximise $\sum_{b\in\mathbb{B}} \alpha_b - \frac{1}{2}\sum_{b,b'\in\mathbb{B}} \alpha_b \alpha_{b'} c(b)c(b')\langle\, x_b \,|\, x_{b'} \,\rangle$, subject to the constraints $\alpha_b \geq 0$ for all $b \in \mathbb{B}$ and $\sum_{b\in\mathbb{B}} \alpha_b c(b) = 0$. The main advantage of the latter form is that the objective function to maximise is determined in terms of the scalar products $\langle\, x_b \,|\, x_{b'} \,\rangle$ that by construction are equal to the kernel $k(b,b')$.

The method can be extended to handle non-separable problems by allowing the so-called soft margin generalisation.

5.2 Graph methods

Starting from a finite space $(\mathbb{X}, k)$ linked by (dis)similarity, we introduce now a different form of expressing relationships among elements of $\mathbb{X}$. Recall that a (dis)similarity linkage is symmetric and vanishes on the diagonal. Define thus an undirected graph $\mathbb{G} = (G^0, G^1)$, where $G^0 = \mathbb{X}$ is the set of vertices and $G^1 = \{\{x,y\} \in (G^0)^2 : k(x,y) > 0\}$ the set of undirected edges.

Every edge $e = \{x,y\} \in G^1$ will be assigned a real **weight** $w(e) \equiv w(x,y) \geq 0$. Assume that k is a similarity and $R :]0,\infty[\to]0,\infty[$ a strictly decreasing function. We define then various types of weights:

$$\begin{array}{rcl}
\text{adjacency} & : & w(x,y) = \left\{\begin{array}{ll} 1 & \text{if } k(x,y) > 0 \\ 0 & \text{otherwise;} \end{array}\right. \\
\text{conductance} & : & w(x,y) = k(x,y); \\
\text{resistance} & : & w(x,y) = \left\{\begin{array}{ll} R((k(x,y)) & \text{if } k(x,y) > 0 \\ 0 & \text{otherwise.} \end{array}\right.
\end{array}$$

If k is a dissimilarity, the definitions of conductance and resistance are interchanged. We denote $W = (w(x,y))_{x,y\in G^0}$ the corresponding weight matrix that is obviously real symmetric. For every $x \in G^0$, we denote by $d_x = \sum_{y\in G^0} w(x,y)$ the **degree** of the vertex x and $D = \mathsf{diag}(d_x, x \in G^0)$ the diagonal matrix containing the degrees on its diagonal. In the sequel we consider only connected graphs so that $d_x > 0$ for all x. Note that when W is an adjacency weight, d_x represents the number of vertices that are neighbours of x (in the sense that they are connected to x by an edge). The matrix W of adjacency weights represents then equivalently the undirected graph $\mathbb{G}$ while conductance or resistence weights W incorporate additionally the information encoded in the (dis)similarity linkage.

In the rest of this section, the matrix W, unless explicitly stated, will be always assumed to denote a conductance weight. If A and B are arbitrary subsets of G^0, we call **volume** of A the quantity $\mathsf{vol}A = \sum_{x\in A} d_x$ and **conductance** between A and B the quantity

$$\mho(A,B) \equiv \sum_{x\in A, y\in B} w(x,y) = \mathbf{1}_A^t W \mathbf{1}_B = \mho(B,A).$$

The **edge boundary** of A, denoted by ∂A, is defined as $\partial A = \{e = \{x,y\} \in G^1 : x \in A, y \in A^c \text{ or } y \in A, x \in A^c\}$.

A subset of edges $C \subset G^1$ is called a **cut** if the graph $\mathbb{G}' = (G^0, G^1 \setminus C)$ is disconnected. Cuts arise naturally in the study of connectivity of graphs, independentently of any consideration on the volume of the disconnected components. The cut problems considered in this section belong to the more complicated class of discrete isoperimetric problems; several continuous isoperimetric problems go back to the antiquity. The type of problems considered here are of the form: find a set $A \subset G^0$ such that $\mathsf{vol}A$ and $\mathsf{vol}A^c$ are comparable while $|\partial A|$, or $\mho(\partial A, \partial A)$, or $\mho(A, A^c)$ are as small as possible, for instance by minimising the expression

$$\mathsf{ObjCostFun}(A) = \frac{\sum_{x\in A; y\in A^c} w(x,y)}{\min(\mathsf{vol}(A), \mathsf{vol}(A^c))}.$$

It is proved in [15] that the infimum of $\mathsf{ObjCostFun}(A)$ is given by the Cheeger isoperimetric constant $h_{\mathbb{G}}$ of the graph. The purpose of any clustering algorithm is to determine a set A with $\mathsf{ObjCostFun}(A)$ as close as possible to $h_{\mathbb{G}}$.

Associated with the matrices W and D are the following graph operators [16, 29, 18]:

$$\begin{array}{rcl} \text{Laplacian} & : & L = -\Delta = D - W, \\ \text{normalised Laplacian} & : & \Lambda = I - D^{-1/2} W D^{-1/2}, \\ \text{Markovian} & : & M = I - D^{-1}W. \end{array}$$

These operators act to the right on vectors of the (complex) Hilbert space $\mathbb{C}^{G^0}$ and to the left on linear forms on this space.

Isoperimetric problems in the discrete setting are known to be hard problems (usually NP-hard). Spectral properties of these operators can be successfully used to algorithmically obtain approximately optimal cuts and provide us with quantifiers of their optimality.

5.2.1 Spectral analysis of the weighed Laplacian

Let $\mathbb{G} = (G^0, G^1)$ be a finite weighed graph with $|G^0| = N$. Let $\lambda_1, \ldots, \lambda_N$ be the eigenvalues of Λ (counted with multiplicities) and $\phi_1, \ldots, \phi_N$ the corresponding eigenvectors. Denote analogously by $\mu_1, \ldots, \mu_N$ be the eigenvalues of M and $\psi_1, \ldots, \psi_N$ the corresponding eigenvectors [49, 9].

Proposition 5.2.1 *1. The spectra of Λ and L are contained in the positive real axis.*

2. *If the eigenvalues of* Λ *are increasingly ordered* $\lambda_1 \leq \ldots \leq \lambda_N$*, then* $\lambda_1 = 0$.
3. $\sum_{k=1}^{N} \lambda_k = N$.
4. *The operator* M *has also real spectrum verifying*

$$1 \geq \mu_1 = 1 - \lambda_1 \geq \ldots \geq \mu_N = 1 - \lambda_N \geq -1$$

while the corresponding eigenvectors verify $\psi_k = D^{-1/2}\phi_k$ *for* $k = 1, \ldots, N$.

5. *The eigenvalues* $\lambda_1, \cdots, \lambda_N$ *of* Λ *and the vectors* $\psi_k = D^{-1/2}\phi_k$ *for* $k = 1, \ldots, N$ *verify the generalised eigenvalue problem:*

$$L\psi_k = \lambda D\psi_k, k = 1, \ldots, N.$$

Proof:

1. Both operators L and Λ are symmetric, therefore their spectra are contained in the real axis. For each edge e of the graph, define an arbitrary orientation (you can toss a coin to choose one). Once the orientations are defined, each edge acquires an arrow pointing from its source vertex to its terminal vertex. Hence there are two functions $s : G^1 \to G^0$ and $t : G^1 \to G^0$ (source and terminal maps) determining the source and terminal vertices of every oriented edge [14]. Define $\mathrm{d} : \ell^2(G^0) \to \ell^2(G^1)$ by

$$\mathrm{d}\, f(e) = f(s(e)) - f(t(e)), e \in G^1$$

and its adjoint $\mathrm{d}^* : \ell^2(G^1) \to \ell^2(G^0)$ by

$$\mathrm{d}^*\phi(x) = \sum_{e=s^{-1}(x)} w(e)\phi(e) - \sum_{e=t^{-1}(x)} w(e)\phi(e).$$

In the finite dimensional case, the operator d can be viewed merely as a $|G^1| \times |G^0|$ matrix with matrix elements

$$\mathrm{d}_{e,x} = \begin{cases} 1 & \text{if } s(e) = x \\ -1 & \text{if } t(e) = x \\ 0 & \text{otherwise,} \end{cases}$$

while d^* stands for the matrix representing the adjoint operator. Obviously $L = \frac{\mathrm{d}^*\,\mathrm{d}}{2}$. The cohomological operator represented by the matrix d is known as **incindence matrix** with respect to the given orientation; since it appears as the “square root” of the Laplacian, it is strongly connected to the Dirac operator on the graph.

Now $\langle f \,|\, 2Lf \rangle = \langle \mathrm{d}f \mid \mathrm{d}\, f \rangle$ proving thus the positivity of the operator L and consequently[3] of Λ, since $\Lambda = D^{-1/2}LD^{-1/2}$.

[3] A direct decomposition of $\Lambda = \frac{\mathsf{l}^*\mathsf{l}}{2}$ where

$$\mathsf{l}f(e) = \frac{1}{\sqrt{d_{s(e)}}}f(s(e)) - \frac{1}{\sqrt{d_{t(e)}}}f(t(e)),$$

2. To show that 0 belongs to the spectrum, it is enough to apply Λ on $D^{1/2}\mathbf{1}$ and recall that $W\mathbf{1} = D\mathbf{1}$.

3. We have $\sum_{k=1}^{N} \lambda_k = \mathsf{tr}\,\Lambda = N - \mathsf{tr}(D^{-1/2}WD^{-1/2}) = N$ because $\sum_{k=1}^{N} \langle\, e_k \,|\, D^{-1/2}WD^{-1/2}e_k \,\rangle = 0$.

4. The eigenvalue equation for M
$$M\psi_k = \mu_k\psi_k$$
implies $W\psi_k = \mu_k D\psi_k$. On defining $\phi_k = D^{1/2}\psi_k$ we have $\Lambda\phi_k = \phi_k - D^{-1/2}WD^{-1/2}\phi_k = \phi_k - D^{-1/2}W\psi_k = \phi_k - D^{-1/2}\mu_k D\psi_k = (1-\mu_k)\phi_k$. To conclude, remark that the spectrum of M is trivially contained into the unit disk.

5. Since $L = D^{1/2}\Lambda D^{1/2}$ and $\phi_k = D^{1/2}\psi_k$, we have $L\psi_k = D^{1/2}\Lambda D^{1/2}D^{-1/2}\phi_k = \lambda_k D\psi_k$.

□

Lemma 5.2.2 *Let $A \in \mathcal{M}_n(\mathbb{R})$ be a symmetric matrix with real components. Let $\lambda_1 \leq \ldots \leq \lambda_n$ be its eigenvalues increasingly ordered and $\psi_1, \ldots, \psi_n$ the corresponding normalised eigenvectors. For $k = 1, \ldots, n$, denote $\mathbb{V}_k = \mathsf{span}(\psi_1, \ldots, \psi_k)$ and $\mathbb{V}_0 = \{0\}$. Then,*
$$\inf_{x\in\mathbb{R}^n : x\neq 0, x\perp\mathbb{V}_{k-1}} \frac{\langle\, x \,|\, Ax \,\rangle}{\langle\, x \,|\, x \,\rangle} = \langle\, \psi_k \,|\, A\psi_k \,\rangle = \lambda_k.$$

Proof: Since A is symmetric, its spectrum is real and its eigenvectors can always be chosen orthonormal; they form therefore an orthonormal basis. Using this basis to decompose an arbitrary vector $x = \sum_{k=1}^{n} x_k |\, \psi_k \,\rangle$ and using the spectral decomposition $A = \sum_{k=1}^{n} \lambda_k |\, \psi_k \,\rangle\langle\, \psi_k \,|$, we get for $x \perp \mathbb{V}_{k-1}$, $\langle\, x \,|\, x \,\rangle = \sum_{j=k}^{n} x_j^2$ and $\langle\, x \,|\, Ax \,\rangle = \sum_{j=k}^{n} \lambda_j x_j^2$. Thus $\frac{\langle\, x \,|\, Ax \,\rangle}{\langle\, x \,|\, x \,\rangle} = \frac{\sum_{j=k}^{n} \lambda_j x_j^2}{\sum_{j=k}^{n} x_j^2} \geq \lambda_k = \langle\, \psi_k \,|\, A\psi_k \,\rangle$. Moreover, the inequality becomes an equality for $x = \psi_k$. □

The previous results allow an elegant approach to the problem of optimal cut of a graph. Denote, for $y \in \mathbb{R}^{G^0}$, by
$$R(y) = \frac{\langle\, y \,|\, (D-W)y \,\rangle}{\langle\, y \,|\, Dy \,\rangle}$$
the **Raleigh quotient** of the graph. For a subset $A \subset G^0$, define the cut weight associated with A as the conductance between A and A^c, i.e.
$$\mathsf{Cut}(A) = \mho(A, A^c) = \sum_{x\in A; y\in A^c} w(x,y) = \langle\, \mathbf{1}_A \,|\, W\mathbf{1}_{A^c} \,\rangle$$

and
$$\mathsf{I}^*\phi(x) = \frac{1}{\sqrt{d_x}}\left(\sum_{e=s^{-1}(x)} w(e)\phi(e) - \sum_{e=t^{-1}(x)} w(e)\phi(e)\right),$$
can be used as well.

the normalised cut weight

$$\mathsf{NormalisedCut}(A) = \mathsf{Cut}(A)\left(\frac{1}{\mathsf{vol}(A)} + \frac{1}{\mathsf{vol}(A^c)}\right).$$

Spectral analysis provides the solution to the unconstrained minimisation of the Raleigh quotient and hence an approximate solution to the normalised cut problem.

For $A \subseteq G^0$, define $r = \frac{\langle \mathbf{1}_A \,|\, D\mathbf{1}_{A^c} \rangle}{\langle \mathbf{1} \,|\, D\mathbf{1} \rangle}$. Let $\xi = \mathbf{1}_A - \mathbf{1}_{A^c}$; since $\mathbf{1}_A = \frac{\mathbf{1}+\xi}{2}$, we have:

$$\mathsf{NormalisedCut}(A) = \frac{\langle \mathbf{1} + \xi \,|\, W(\mathbf{1} - \xi) \rangle}{4r(1-r)\langle \mathbf{1} \,|\, D\mathbf{1} \rangle} = \frac{\langle \xi \,|\, (D - W)\xi \rangle}{4r(1-r)\langle \xi \,|\, D\xi \rangle}.$$

Now, giving any vector $\xi \in \{-1,1\}^{G^0}$ is equivalent in giving a set A because we can define $A = \{\xi > 0\}$. Hence, we can write $\mathsf{NormalisedCut}(\xi)$ instead of $\mathsf{NormalisedCut}(A)$ and seeking for the optimal A is equivelent is minising $\mathsf{NormalisedCut}(\xi)$ over discrete vectors $\xi \in \{-1,1\}^{G^0}$.

Proposition 5.2.3 *Let* $H_r = \{\eta \in \mathbb{R}^{G^0} : \forall x \in G^0 \Rightarrow \eta(x) = \pm\frac{1}{2}\sqrt{\frac{d_x}{4r(1-r)}}\}$. *Then*

$$\inf_{A \subset G^0} \mathit{NormalisedCut}(A) = \inf_{\eta \in H_r} \frac{\langle \eta \,|\, \Lambda\eta \rangle}{\langle \eta \,|\, \eta \rangle}.$$

Proof: We have that $\inf_{A\subset G^0} \mathsf{NormalisedCut}(A) = \inf_{\xi} \frac{\langle \xi \,|\, (D-W)\xi \rangle}{4r(1-r)\langle \xi \,|\, D\xi \rangle}$, where the minimisation runs over $\xi \in \{-1,1\}^{G^0}$. On defining $\eta(x) = \xi(x)\sqrt{\frac{d_x}{4r(1-r)}}$, we get that $\frac{\langle \xi \,|\, (D-W)\xi \rangle}{4r(1-r)\langle \xi \,|\, D\xi \rangle} = \frac{\langle \eta \,|\, \Lambda\eta \rangle}{\langle \eta \,|\, \eta \rangle}$. □

Of course, the discrete problem is very hard. The unconstrained minimisation over η is nevertheless extremely easy; an approximate solution is provided by lemma 5.2.2.

5.2.2 Random walk distances

Random walk approach to clustering [25, 24, 47, 64] is based on three very intuitive ideas:

- Assume that the symmetric weight matrix W is a resistance weight. Then the effective resistance among two arbitrary vertices x and y of the graph is computed classically using the Kirchoff's laws. Roughly speaking, the more low resistance paths exist among the two points, the lower is the effective resistance among them.
- The Markovian graph operator M induces a simple random walk on the graph. Asymptotic properties of the walk are obtained through the electrical circuit analogue (see [21] for an elementary introduction). The reason

is that harmonic functions on the graph transform the Markov chain to a martingale.

- Due to the close relationship between random walks and electric circuits, low effective resistance among two points means that the random walk jumps easily from one point to the other; large effective resistance means that the random walk need long times to jump from one to the other.

Let $(X_n)_{n\in\mathbb{N}}$ be a (G^0, M)-Markov chain on G^0 driven by the Markov operator M (we don't specify yet the initial probability). Let π be the probability $\pi(x) = \frac{d_x}{\mathsf{vol}(G^0)}$. Obviously, due to the symmetry of W, we have

$$\pi(x)M(x,y) = \frac{d_x}{\mathsf{vol}(G^0)} d_x^{-1} w(x,y) = \pi(y)M(y,x).$$

Thus the chain is reversible.

Definition 5.2.4 Denote by $\tau_y = \inf\{n \geq 1 : X_n = x\}$ the passage time to y and let $C = (c(x,y))_{x,y\in G^0}$ be a matrix of positive elements assigning cost $c(x,y)$ to the transition from x to y, for arbitrary vertices x and y of G^0. We define:

1. the **average passage time** from x to y: $\mathsf{pt}(x,y) = \mathbb{E}_x(\tau_y)$;
2. the **average commute time** between x and y: $\mathsf{ct}(x,y) = \mathsf{pt}(x,y) + \mathsf{pt}(y,x)$;
3. the **average passage cost** from x to y: $\mathsf{pc}(x,y) = \mathbb{E}_x(\sum_{k=1}^{\tau_y} c(X_{k-1}, X_k))$;
4. the **average commute cost** between x and y: $\mathsf{cc}(x,y) = \mathsf{pc}(x,y) + \mathsf{pc}(y,x)$.

All these quantities are explicitly expressed in terms of the pseudo-inverse, L^+ of the Laplacian operator.

Definition 5.2.5 Let $Z \in \mathfrak{M}_{m,n}(\mathbb{C})$ be a matrix of rank $r \leq \min(m,n)$ and let $Z = UDV^\dagger$ its singular value decomposition with D the pseudo-diagonal matrix $D = \begin{pmatrix} S & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \in \mathfrak{M}_{m,n}(\mathbb{C})$ and $S = \mathsf{diag}(s_1, \ldots, s_r)$. The **pseudo-inverse** of Z is the matrix $Z^+ = UD^+V^\dagger$ with $D^+ = \begin{pmatrix} S^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \in \mathfrak{M}_{m,n}(\mathbb{C})$.

Remark: Note that since the rank of Z is assumed r, all diagonal elements of the matrix S are non zero. Additionally, if $r = m = n$ then obviously $Z^+ = Z^{-1}$.

Lemma 5.2.6 *Let $\mathbb{G} = (G^0, G^1)$ be a connected graph with $|G^0| = N$. Let $\ell_1 \leq \ldots \leq \ell_N$ be the eigenvalues of L, increasingly ordered, and $\psi_1, \ldots, \psi_N$ the corresponding normalised eigenvectors. Then*

1. $\mathsf{rank}(L) = N - 1$
2. $\ell_1 = 0$ *and* $\psi_1(x) = 1/\sqrt{N}$ *for all* x,
3. $\ker L = \mathit{span}\{\psi_1\}$,
4. $L^+ = (L - |\,\psi_1\,\rangle\langle\,\psi_1\,|)^{-1} + |\,\psi_1\,\rangle\langle\,\psi_1\,|$.
5. $\mathsf{spec}(L^+) = \{0, \ell_2^{-1}, \cdots, \ell_N^{-1}\}$.

Proof: Recall the decomposition $L = \frac{\mathrm{d}^*\,\mathrm{d}}{2}$, where d is the incidence matrix of the graph associated with an arbitrary orientation, introduced in the the course of the proof of the proposition 5.2.1.

1. Obviously $\mathsf{rank}(L) = \mathsf{rank}(\mathrm{d}^*\,\mathrm{d})$. Now if $\psi \in \ker(\mathrm{d}^*\,\mathrm{d})$, then $\langle\,\psi\,|\,\mathrm{d}^*\,\mathrm{d}\,\psi\,\rangle = \langle\,\mathrm{d}\,\psi\,|\,\mathrm{d}\,\psi\,\rangle = 0$. Hence $f = \mathrm{d}\,\psi$ is the null vector of $\ell^2(G^1)$, which implies that $\mathsf{rank}(L) = \mathsf{rank}(\mathrm{d})$. Now suppose that $\{x, y\} \in G^1$ and (x, y) corresponds to the chosen orientation. Then $f(\{x, y\}) = 0 = \psi(x) - \psi(y)$; therefore, the vector $\psi \in \mathbb{R}^{G^0}$ is constant on any connected component of $\mathbb{G}$. The dimension of this subspace is c, where c is the number of connected components, showing that $\mathsf{rank}(\mathrm{d}) = N - c$.
2. It has been shown in proposition 5.2.1 that the spectrum of L is real and non-negative. It is immediate to verify that $L\psi_1 = 0$, showing that ψ_1 is a normalised eigenvector associated with the eigenvalue $\ell_1 = 0$.
3. Since $\mathsf{rank}(L) = N - 1$, it follows that that $\dim\ker(L) = 1$ and since $\psi_1 \in \ker(L)$ the result follows.
4. The operator L being symmetric, use its spectral decomposition $L = \sum_{k=1}^N \ell_k|\,\psi_k\,\rangle\langle\,\psi_k\,|$ and the vanishing of $\ell_1 = 0$, to write $L - |\,\psi_1\,\rangle\langle\,\psi_1\,| = -|\,\psi_1\,\rangle\langle\,\psi_1\,| + \sum_{k=2}^N \ell_k|\,\psi_k\,\rangle\langle\,\psi_k\,|$. Hence
$$\begin{aligned} L^+ &= (L - |\,\psi_1\,\rangle\langle\,\psi_1\,|)^{-1} + |\,\psi_1\,\rangle\langle\,\psi_1\,| \\ &= \sum_{k=2}^N \frac{1}{\ell_k}|\,\psi_k\,\rangle\langle\,\psi_k\,|. \end{aligned}$$
5. Immediately follows from the previous expression.

□

Theorem 5.2.7 *Let* (X_n) *be a Markov chain on* G^0 *the vertex set of a a connected graph. Then the function* $G^0 \times G^0 \ni (x, y) \mapsto \sqrt{\mathsf{ct}(x, y)}$, *where* ct *is the average commute time for the Markov chain, is a distance.*

Proof: Introducing taboo probabilities, a straightforward computation leads to the expression $\mathsf{ct}(x, y) = \mathsf{vol}(G^0)\langle\,e_x - e_y\,|\,L^+(e_x - e_y)\,\rangle$, where L^+ is the pseudo-inverse of the weighed Laplacian and $e_x(z) = \delta_{xz}$. Now, for all x and y, with $x \neq y$, the vector $e_x - e_y$ lies on the orthogonal complement of the eigenspace associated with the eigenvalue 0; this space has dimension $N-1$ and the restriction of L^+ on this subspace is positive definite. Hence $\mathsf{ct}(x,y) = 0$ implies $e_x - e_y = 0$, i.e. $x = y$. The other properties of the distance are also immediately verified. □

Remark: The distance d on $\mathbb{X} = G^0$ defined in terms of the commute time renders $(\mathbb{X}, d)$ a linked space. The intuitive meaning of this distance is that the random walk on the graph spends most of its time on clusters and from time to time there are rare transitions from one cluster to another. Thus clusters are identified by subsets whose occupation measure with respect to the random walk are high while cuts are only scarcely occupied. This observation gives a direct stochastic simulation algorithm to determine clusters.

5.2.3 Optimal low-dimensional representations of a graph

Suppose that we have N experimental observations, indexed by a finite set $\mathbb{X}$ with $|\mathbb{X}| = N$; suppose further that all the available information on the data is encoded into a weighed graph $\mathbb{G} = (G^0, G^1)$ with $G^0 = \mathbb{X}$ and weight matrix W. We have already seen in section 3 that the data can be represented in an Euclidean space $\mathbb{R}^\nu$ and theorem 3.1.7 determines the minimal ν for which such a representation can be made without loss of information. However, for all practical purposes, i.e. for N of the order of 10000, the dimension ν determined by the Schoenberg's theorem is very large. Therefore the informationally faithful representation of the dataset in $\mathbb{R}^\nu$ is very cumbersome and difficult to exploit by a human.

The problem we address in this section can be stated as follows. Is there a low-dimensional (typically one-, two-, or three-dimensional) Euclidean representation of the weighed graph so that vertices of the graph with small effective conductance (or small graph distance, or small commute time, etc.) be represented by points with small Euclidean distance? The idea behind this question is that the points of the dataset are not just random variables in $\mathbb{R}^\nu$ but variables related by some mostly hidden and unknown relationships, constraining them to live on some smooth manifold embedded in this space. If the relationships are sufficiently smooth and numerous, it is expected that the manifold containing the representing points is low-dimensional so that the represenation into a one-, two-, or three-dimensional space does not distort the local geometry too much.

Let $\rho : \mathbb{X} \to \mathbb{R}^\nu$ be some (not necessarily faithful) representation of the dataset. A reasonable criterion is to minimise the following objective cost function, penalising those representations mapping neighbouring points far apart,

$$\mathsf{ObjCostFun}(\rho) = \frac{1}{2} \sum_{x,y\in\mathbb{X}} \|\rho(x) - \rho(y)\|^2 w(x,y),$$

under some constraints. Now

$$\begin{aligned}\sum_{x,y\in\mathbb{X}} \|\rho(x) - \rho(y)|^2 w(x,y) &= \sum_{x,y\in\mathbb{X}} \left(\|\rho(x)\|^2 + \|\rho(y)\|^2 - 2\langle\, \rho(x) \,|\, \rho(y)\,\rangle\right) w(x,y) \\ &= \sum_{x\in\mathbb{X}} \|\rho(x)\|^2 d_x + \sum_{y\in\mathbb{X}} \|\rho(y)\|^2 d_y - 2 \sum_{x,y\in\mathbb{X}} \langle\, \rho(x) \,|\, \rho(y)\,\rangle w(x,y).\end{aligned}$$

Remark: Note that in the previous formula, $\|\cdot\|$ and $\langle\,\cdot\,|\,\cdot\,\rangle$ are meant to denote ν-dimensional Euclidean norm and scalar product respectively.

Lemma 5.2.8 *Let $\rho : G^0 \to \mathbb{R}^\nu$ be a not necessarily faithful representation of the dataset and denote by $R \in \mathfrak{M}_{\nu,|G^0|}(\mathbb{R})$ the matrix containing the vector $\rho(x) \in \mathbb{R}^\nu$ as its x^{th} comumn. Then*

$$\mathsf{ObjCostFun}(\rho) = 2\,\mathsf{tr}(RLR^t).$$

Proof: Fix an arbitrary orientation on the graph and denote by $\mathrm{d} : \ell^2(G^0) \to \ell^2(G^1)$ the incidence operator and by d^* its adjoint. (By abuse of notation, we shall use the same symbols to denote the matrices representing these operators). Within the proof of proposition 5.2.1, it is established that $L = \mathrm{d}^*\,\mathrm{d}/2$. Now $\mathrm{d}\,R^t \in \mathfrak{M}_{|G^1|,\nu}(\mathbb{R})$ whose (e,k) element, for $e = (x,y)$, reads $\mathrm{d}\,R^t(e,k) = \rho^{(k)}(x) - \rho^{(k)}(y)$. Letting $R\mathrm{d}^* \in \mathfrak{M}_{\nu,|G^1|}$ denote the matrix representing the adjoint, we get

$$R\mathrm{d}^*\,\mathrm{d}\,R^t(k,k') = \frac{1}{2}\sum_{x,y\in G^0} w(x,y)(\rho^{(k)}(x) - \rho^{(k)}(y))(\rho^{(k')}(x) - \rho^{(k')}(y)).$$

The result follows immediately by taking the trace. □

It is obvious that if we minimise the objective cost function without imposing any constraint, we get the trivial solution $\rho \equiv 0$. To prevent this collapse, we fix the scaling of the representation by imposing orthonormality $RR^t = \mathbf{1}_\nu$. Without loss of generality, we can assume that $\sum_{x\in G^0}\rho(x) = 0$; the representation is then called balanced. Balancing means that the centre of gravity of the representation is placed at the origin of the coordinate system [29].

Theorem 5.2.9 *Let $\mathbb{G} = (G^0, G^1)$ be a graph with $|G^0| = N$ and denote by L its weighed Laplacian. Assume that the eigenvalues of L are increasingly ordered $0 = \lambda_1 \leq \ldots \leq \lambda_N$ and that $\lambda_2 > 0$. Denote the corresponding eigenvectors $\psi_1, \ldots, \psi_k$. Then the infimum over orhtonormal balanced representations of dimension ν of $\mathsf{ObjCostFun}(\rho)$ is $\sum_{k=2}^{\nu+1}\lambda_k$. The matrix R saturating the previous minimum is a $\nu \times |G^0|$ matrix whose row k is the transposed eigenvector ψ_{k+1}^t.*

Proof: Start from the spectral decomposition of the Laplacian

$$L = \sum_{k=1}^{N}\lambda_k|\,\psi_k\,\rangle\langle\,\psi_k\,|.$$

Denote $\mathbb{V}_j = \mathsf{span}(\psi_1, \ldots, \psi_j)$ for $j = 1, \ldots, N$ and $\mathbb{V}_0 = \{0\}$. Since the matrix L is symmetric and semi-positive definite, for every $\eta \in \mathbb{V}_j$, we have from lemma 5.2.2 that $\inf_{\eta\neq 0;\eta\perp\mathbb{V}_j}\frac{\langle\,\eta\,|\,L\eta\,\rangle}{\langle\,\eta\,|\,\eta\,\rangle} = \lambda_{j+1}$. The eigenvalue $\lambda_1 = 0$ corresponds to the constant vector $\psi_1(x) = \frac{1}{N}$, for all x. Since the representation is balanced, for all $k = 1, \ldots, \nu$, we shall have $\rho^{(k)} \perp \mathbb{V}_1$. Therefore, $\inf_{k=1,\ldots,\nu}\frac{\langle\,\rho^{(k)}\,|\,L\rho^{(k)}\,\rangle}{\langle\,\rho^{(k)}\,|\,\rho^{(k)}\,\rangle} \geq \lambda_2$

and this infimum will be achieved for some $k_1 \in \{1, \ldots, \nu\}$. Then iteratively apply this inequality

$$\inf_{k=1,\ldots,\nu;k\neq k_1;\ldots;k\neq k_r} \frac{\langle\, \rho^{(k)} \mid L\rho^{(k)} \,\rangle}{\langle\, \rho^{(k)} \mid \rho^{(k)} \,\rangle} \geq \lambda_{r+1}.$$

Hence for a balanced orthonormal represenation ρ,

$$\begin{aligned} \mathsf{ObjCostFun}(\rho) &= \sum_{k=1}^{\nu}\sum_{j=1}^{N} \lambda_j (\langle\, \rho^{(k)} \mid \psi_j \,\rangle)^2 \\ &\geq \sum_{j=2}^{\nu+1} \lambda_j. \end{aligned}$$

Obviously the previous inequality becomes an equality if we choose $\rho^{(k)} = \psi_{k+1}$.

□

Chapter 6

Logics and semantic spaces

Logic investigates the methods of reasoning and provides us with the rules and techniques of correct and reliable inference. It deals with propositions and their relations to each other. Its applications can be found in mathematics, computer sciences, and more generally in every rigorous scientific reasoning but even in every day life argumentation. Besides the classical Boolean logic there have been established various generalisations such as modal, intuitionistic, quantum, or fuzzy logic, as well as propositional structures underlying substructural logics which focus on relaxations of structural rules governing validity and provability.

Experimental sciences produce experimental data that are analysed by statistical methods to infer relationships among various quantities and establish a **phenomenology** of their disciplines. When underlying mechanisms are discovered that explain all phenomenology, we speak about a scientific theory. The theory predicts new facts, new experiments are designed and performed to test these predictions, The new experiments enrich the existing phenomenology and either they confirm the predictions of the theory, in which case the theory is reinforced, or they infirm the theoretical predictions, in which case the theory is rejected as unsatisfactory.

Since statistical inference is used to analyse experimental data, the inferred relationships are necessarily probabilistic. Now classical probability theory relies on the notion of σ-algebra (i.e. a σ-complete Boolean lattice) and therefore to classical Boolean logic. The *Leitmotiv* of this work is that, at the present stage of ignorance of the underlying mechanisms governing gene interactions and expressions within the living matter, it is worth exploring whether non traditional analysis methods can infer relationships unreachable by classical methods. The results that will be presented in next sections confirm that it is. Therefore, before presenting our methods and results, we give a brief survey of some unconventional logical structures.

6.1 Lattices and logics

The basic notions exposed in this subsections are due to Dedekind and have been extended and elaborated by Birkhoff; a modern and accessible reference is [13]. A very useful and pedagogical reference is also [19]. The various aspects of quantum logic are thoroughly treated in [59, 57, 70].

Definition 6.1.1 Let $\mathbb{X}$ be an arbitrary set and $\preceq$ a partial order relation on $\mathbb{X}$ (i.e. a reflexive, antisymmetric, and transitive relation). The pair $(\mathbb{X}, \preceq)$ is called a partially ordered set or **poset**.

Example 6.1.2 Typical posets are $(\mathbb{R}, \leq)$ and $(\mathcal{P}(\mathbb{Y}), \subseteq)$ where $\mathcal{P}(\mathbb{Y})$ denotes the set of subsets of an arbitrary set $\mathbb{Y}$. The poset $(\mathbb{R}, \leq)$ is totally ordered in the sense that for every $x, y \in \mathbb{R}$, either $x \leq y$ or $y \leq x$, while the poset $(\mathcal{P}(\mathbb{Y}), \subseteq)$ is only partially ordered since there exist $x, y \subseteq \mathbb{Y}$ that are incomparable by inclusion.

If a poset contains a particular element $\mathbf{0} \in \mathbb{X}$ such that for all $x \in \mathbb{X}$ we have $\mathbf{0} \preceq x$, then this element is unique and is called the **least element** of $\mathbb{X}$. Dually, if a poset contains a particular element $\mathbf{1} \in \mathbb{X}$ such that for all $x \in \mathbb{X}$ we have $x \preceq \mathbf{1}$, then this element is unique and is called the **greatest element** of $\mathbb{X}$. In a poset with least element, an element $x \in \mathbb{X}$ such that $\mathbf{0} \prec x$ and there does not exist $y \in \mathbb{X}$ such that $\mathbf{0} \prec y \prec x$ is called an **atom**. If $X \subseteq \mathbb{X}$ and $u \in \mathbb{X}$ an element such that $\forall x : x \in X \Rightarrow x \preceq u$, then u is called an **upper bound** of X. An upper bound s of X such that $s \preceq u$ for all upper bounds u of X is called the **least upper bound**, denoted $\sup X$. Lower bounds and the greatest lower bound $\inf X$ are defined dually. Note that both $\sup X$ and $\inf X$ if they exist are unique.

Definition 6.1.3 A **lattice** is a poset $\mathbb{X}$ any two of whose elements have a greatest lower bound, called **meet** and denoted by $x \wedge y = \inf\{x, y\}$, and a least upper bound, called **join** and denoted by $x \vee y = \sup\{x, y\}$. A lattice is **complete** if when each of its subsets $X \subseteq \mathbb{X}$ has a greatest lower bound and a least upper bound; it is **σ-complete** when this property holds for all countable subsets X. A lattice is **atomic** if every subset X is the join of atoms.

Example 6.1.4 Let $\mathbb{Y}$ be an arbitrary non-empty set. Consider the poset $(\mathcal{P}(\mathbb{Y}), \subseteq)$. Then for all $X, Y \subseteq \mathbb{Y}$ the join is defined by $X \wedge Y = X \cap Y$ and the meet by $X \vee Y = X \cup Y$, rendering $\mathcal{P}(\mathbb{Y})$ into a lattice. For $\mathcal{Y} = \{Y_1, Y_2, \ldots\}$ an arbitrary family of subsets of $\mathbb{Y}$, we have $\inf \mathcal{Y} = \cap_i Y_i$ and $\sup \mathcal{Y} = \cup_i Y_i$. Thus $\mathcal{P}(\mathbb{Y})$ is a complete lattice.

Example 6.1.5 Denote by $\mathbb{K}$ the field of real or complex numbers and by $\mathbb{X} = \mathbb{V}(\mathbb{K}^n)$ the set of vector subspaces of $\mathbb{K}^n$, equipped with the partial order $\preceq$ defined as follows: for $x, y \in \mathbb{X}$ we say that $x \preceq y$ if x is a vector subspace of y. Obviously then $x \wedge y = x \cap y$ and $x \vee y = \mathsf{span}(x, y)$, turning the poset $\mathbb{X}$ into a lattice[1].

For all elements x, y, z of a lattice $\mathbb{X}$, whenever the expressions below exist, operations of join and meet verify the following properties:

1. $x \wedge x = x = x \vee x$ (idempotence),
2. $x \wedge y = y \wedge x$ and $x \vee y = y \vee x$ (commutativity),
3. $x \wedge (y \wedge z) = (x \wedge y) \wedge z$ and $x \vee (y \vee z) = (x \vee y) \vee z$ (associativity),
4. $x \wedge (x \vee y) = x = x \vee (x \wedge y)$ (absorption),
5. $x \preceq y \Leftrightarrow x \wedge y = x \Leftrightarrow x \vee (x \wedge y) = y$ (consistency).

Remark: It is worth noting that distributivity, although verified in several classical examples (see examples 6.1.4 for instance), **is not** a consequence of the lattice definition. Only some weaker notions are stemming from the definition as shown in the following

Theorem 6.1.6 *Let $\mathbb{X}$ be a lattice. For all $x, y, z \in \mathbb{X}$, we have*

1. $x \preceq z \Rightarrow x \vee (y \wedge z) \preceq (x \vee y) \wedge z$ *(modularity),*
2. $x \wedge (y \vee z) \succeq (x \wedge y) \vee (x \wedge z)$ *(distributive inequality),*
3. $x \vee (y \wedge z) \preceq (x \vee y) \wedge (x \vee z)$ *(dual distributive inequality).*

Proof: Suppose $x \preceq z$. Now $x \preceq x \vee y$, so that $x \preceq (x \vee y) \wedge z$. Similarly, $y \wedge z \preceq y \preceq x \vee y$ and since $y \wedge z \preceq z$, we have that $y \wedge z \preceq (x \vee y) \wedge z$ and combining with the previous inequality, we get finally $x \vee (y \wedge z) \preceq (x \vee y) \wedge z$ establishing modularity.

To establish distributive inquality, note that $x \wedge y \preceq x$ and $x \wedge y \preceq y \preceq y \vee z$, so that $x \wedge y \preceq x \wedge (y \vee z)$. Similarly, $x \wedge z \preceq x$ and $x \wedge z \preceq z \preceq y \vee z$, hence $x \wedge z \preceq x \wedge (y \vee z)$. Therefore, $x \wedge (y \vee z)$ is an upper bound of both $x \wedge y$ and $x \wedge z$. Dual distributive inequality follows by duality. □

Definition 6.1.7 A lattice is called **distributive** if the distributive inequality is an equality.

Remark: In a distributive lattice, the dual distributive inequality can easily be shown to be also an equality.

Definition 6.1.8 Let $\mathbb{X}$ be a lattice with universal bounds $\mathbf{0}$ and $\mathbf{1}$ (i.e. for all $x \in \mathbb{X}$, we have $\mathbf{0} \preceq x \preceq \mathbf{1}$).

[1] This lattice is complete in this finite dimensional setting but not in the infinite dimensional one.

1. A mapping $' : \mathbb{X} \to \mathbb{X}$ is called a **fuzzy negation** if it verifies for all $x, y \in \mathbb{X}$,
   - $x \preceq (x')'$ (weak double negation),
   - $x \preceq y \Rightarrow y' \preceq x'$ (antitony),
   - $\mathbf{0}' = \mathbf{1}$ and $\mathbf{1}' = \mathbf{0}$ (Boolean boundary condition).
2. The pair $(\mathbb{X}, ')$ is called a **fuzzy logic** and elements of the lattice are called **propositions**.
3. If the fuzzy negation further verifies the law of **non-contradiction** $x \wedge x' = \mathbf{0}$, the fuzzy logic is termed simply a (non-contradictory) **logic**.
4. In a (non-contradictory) logic, the element x' is called the **pseudo-complement** of x; if it verifies further $x \vee x' = \mathbf{1}$ then it is called the **complement** of x.

Remark: In a fuzzy logic, the relation $x \preceq y$ will be interpreted as "x implies y", the proposition $x \wedge y$ as the conjunction "x and y", while $x \vee y$ as the disjunction "x or y". The proposition $\mathbf{1}$ is the truth and $\mathbf{0}$ the falsehood or absurdity.

Theorem 6.1.9 *1. Let $\mathbb{X}$ be a lattice equipped with a mapping $' : \mathbb{X} \to \mathbb{X}$ verifying the weak double negation property. This mapping is antitone if and only if the disjunctive de Morgan law*

$$(x \vee y)' = x' \wedge y'$$

holds for all $x, y \in \mathbb{X}$.

2. In a fuzzy logic, the conjunctive de Morgan inequality

$$(x \wedge y)' \succeq x' \vee y'$$

holds for all $x, y \in \mathbb{X}$.

3. If in a fuzzy logic we have further $(x')' = x$ for all propositions, then the conjunctive de Morgan law

$$(x \wedge y)' = x' \vee y'$$

holds for all $x, y \in \mathbb{X}$.

Proof:

1. Assume that the disjunctive de Morgan law holds. Now for $x \preceq y$ it follows that $y = x \vee y$. Hence, $y' = (x \vee y)' = x' \wedge y' \preceq x'$ proving antitony.

   Conversely, suppose anitony holds, and let $v = x \vee y$ and $w = x' \wedge y'$. Then $v \succeq x$ and $v \succeq y$ while $w \preceq x'$ and $w \preceq y'$. By the antitony, $v' \preceq x'$ and $v' \preceq y'$, while $w' \succeq x$ and $w' \succeq y$. Hence $v' \preceq x' \wedge y' = w$, and consequently $v \succeq w'$ by antitony, as well as $v = x \vee y \preceq w'$; this yields $v = w'$.

2. By disjunctive de Morgan law, $(x' \vee y')' = (x')' \wedge (y')' \succeq x \wedge y$. Now, $x' \vee y' \preceq ((x' \vee y')')' \preceq (x \wedge y)'$, establishing the sought inequality.
3. The disjunctive de Morgan law yields $(x' \vee y')' = x \wedge y$. Hence $(x \wedge y)' = ((x' \vee y')')' = x' \vee y'$.

□

For a non-contradictory negation, the fuzzy logic becomes a logic where a stronger result holds, as described by the following

Theorem 6.1.10 (Tertium non datur) *In a logic, with a (non-contradictory) negation verifying further $(x')' = x$ for all propositions, the tertium non datur (exluded middle) $x \vee x' = \mathbf{1}$ holds.*

Proof: If the negation verifies $(x')' = x$, both disjunctive and conjuctive de Morgan laws hold. Non contradiction reads $x \wedge x' = \mathbf{0}$ and complementation of the latter yields $x \vee x' = \mathbf{1}$. □

Definition 6.1.11 1. A **paraconsistent logic** is a fuzzy logic where the paraconsistent relation

$$[x \preceq y \text{ and } x' \wedge y = 0] \Rightarrow [x = y]$$

holds.

2. An **intuitionistic logic** is a distributive logic in which there exist propositions x with $x \prec (x')'$.
3. A **quantum logic** is a logic verifying the orthomodularity condition

$$x \preceq y \Rightarrow [x \vee (x' \wedge y) = y].$$

4. A **Boolean logic** is a complemented distributive logic.

The previous definitions introduce a natural hierarchy among various types of logics as explained in the following graph.

$$\begin{tikzcd}
\text{lattice} \arrow[d, "\text{fuzzy negation}"] & \\
\text{fuzzy logic} \arrow[r, "\text{paraconsistency}"] \arrow[d, "\text{non-conradiction}"] & \text{paraconsistent logic} \arrow[d, "\text{orthomodularity}"] \\
\text{logic} \arrow[r, "\text{orthomodularity}"] \arrow[d, "\text{distributivity}"] & \text{quantum logic} \arrow[d, "x=(x')'"] \\
\text{distributive logic} \arrow[r, "x=(x')'"] \arrow[d, "\exists x: x \prec (x')'"] & \text{Boolean logic} \\
\text{intuitionistic logic} &
\end{tikzcd}$$

6.2 Boolean, fuzzy, and quantum logics

6.2.1 Boolean logic

Boolean logic is the most common classically used logic. It lies beneath the main classical mathematical theories. In particular, it is well known that the Kolmogorov axiomatisation of classical probability theory relies on the notion of σ-algebra (i.e. a σ-complete Boolean logic). Now this formulation is completed by the so called Loomis-Sikorski theorem stating that an arbitrary σ-algebra can be realised as the σ-complete Boolean algebra of subsets of a universal set Ω. More precisely, if $(\mathbb{X}_1, \preceq_1)$ and $(\mathbb{X}_2, \preceq_2)$ are two σ-complete lattices, a morphism f is a map $f : \mathbb{X}_1 \to \mathbb{X}_2$ verifying for all $x, y \in \mathbb{X}_1$ the join and meet-morphism relationships: $f(x \vee_1 y) = f(x) \vee_2 f(y)$ and $f(x \wedge_1 y) = f(x) \wedge_2 f(y)$. It is a σ-morphism if the previous equalities hold for countable joins and meets, it is called an epimorphism if f is surjective, a monomorphism if f is injective and an isomorphism if f is bijective. The precise formulation of Loomis-Sikorski theorem [45, 66] (see also [13], p. 255) is that any Borel algebra is the σ-epimorphic image of a σ-algebra of subsets of a universal set Ω.

The strong intuitive consequences of this theorem are scarcely underlined in standard texts on classical probability theory. For instance, if $(\mathbb{X}, \mathcal{X}, \mu)$ is a probability space and X a $\mathbb{X}$-valued random variable distributed according to μ, the Loomis-Sikorski theorem *guarantees* the existence of an abstract probability space $(\Omega, \mathcal{F}, \mathbb{P})$ on which the random variable X can be defined! This intuitive description of random variables defined as observables on abstract probability spaces, lurking in the original text by Komogorov [39], is almost lost in nowadays expositions on probability theory.

Let X be a real bounded random variable (i.e. a real-valued (measurable) observable defined on some probability space $(\Omega, \mathcal{F}, \mathbb{P})$). Let $\mathbb{X} = X(\Omega) \subseteq [a, b]$ for some reals $a < b$. For every $B \in \mathcal{B}([a, b])$ define $P(B) = \mathbb{1}_{X^{-1}(B)}$. For every $\epsilon > 0$ there exists a finite or countable partition (I_j) of $[a, b]$ into disjoint sets such that choosing arbitrary $x_j \in I_j$ for all j, we can approximate X with a precision at least ϵ by a linear combination of indicators

$$|X(\omega) - \sum_j x_j P(I_j)(\omega)| < \epsilon, \forall \omega \in \Omega.$$

When the partition gets finer and finer, the function P defined above becomes an indicator-valued measure on $\mathcal{B}([a, b])$ and we can write $X = \int_{\mathbb{X}} x P(dx)$. This integral is called the **spectral decomposition** of the random variable X. Note that since $\mathbb{E}X = \int_\Omega X(\omega)\mathbb{P}(d\omega) = \int_{\mathbb{X}} x \mathbb{P}_X(dx)$, where $\mathbb{P}_X$ is the law of X, we have $\mathbb{P}_X = \mathbb{E}P$, i.e. the law of X is the exected value of its spectral measure. It is worth noting also that the spectral measure of X contains all the information encoded in X: random variables can be *identified* with their spectral measures. Since spectral measures are as a matter of fact indicator functions, classical probability theory heavily relies on manipulations on indicators. Every indicator can in turn be identified with a crisp set. It becomes therefore evident why classical probability is defined on Boolean σ-algebras of subsets of a given universal set.

The purpose of this work is to explore probability measures defined on more general lattices than Boolean algebras. Before dealing with proabability measures (in the subsection 6.3), we explore first the necessary relaxation of the notion of crisp set. Otherwise stated, classical probability on Boolean lattices involves indicators hence crisp sets. What is the underlying model corrsponding to fuzzy or quantum lattices? We shall see in subsections 6.2.2 and 6.2.3 that to represent fuzzy lattices we must abandon the notion of crisp set and replace the notion of sharp indicator by the notion of fuzzy membership (leading to a fuzzy set theory) while for quantum lattices we must replace indicators by projectors to Hilbert subspaces.

6.2.2 Fuzzy logic

Fuzzy set theory was initiated in 1965 by Zadeh [79] as a generalisation of standard set theory (that will be termed *crisp* set theory in the sequel whenever this distinction or precision is necessary). The intuitive idea behind fuzzy set theory is the following: when the outdoor temperature is $32°C$ it is definitely a warm day, when it is $-10°C$ a cold day; but what about a temperature of $15°C$ or $25°C$? It is not possible to find a temperature T such that the set of temperatures $C =]-\infty, T[$ to be called indisputably "cold" while the set $W = [T, \infty[$ to be called indisputably "warm". It is more reasonable to trade crisp sets defined by the indicators $\mathbb{1}_C$ and $\mathbb{1}_W$ by some blurred versions of these sets, defined by the membership function cold : $\mathbb{R} \to [0,1]$ and a membership function warm : $\mathbb{R} \to [0,1]$ such that each otdoor temperature t be cold with possibility cold(t) and warm with possibility warm(t). An example of such functions is given by:

$$\text{cold}(t) = \begin{cases} 1 & \text{if } t \leq 10 \\ 1 - \frac{t-10}{20} & \text{if } 10 < t < 30 \\ 0 & \text{if } t \geq 30 \end{cases} ; \text{warm}(t) = \begin{cases} 0 & \text{if } t \leq 10 \\ \frac{t-10}{20} & \text{if } 10 < t < 30 \\ 1 & \text{if } t \geq 30. \end{cases}$$

More generally, a fuzzy logic is the set $\mathbb{X} = \{x : \Omega \to [0,1]\}$ of **membership functions** (propositions) on a given universal set Ω. Contrary to propositions in a Boolean logic that are either true or false, here propositions can take intermediate truth values. Conjunction is usually defined by $x \wedge y = \min(x,y)$ and disjunction by $x \vee y = \max(x,y)$. Fuzzy complementation is naturally defined by $x' = 1 - x$; therefore a fuzzy logic will be contradictory if there exists a proposition (membership function) x such that for some $\omega \in \Omega$ the following inequalities $0 < x(\omega) < 1$ occur.

More general definitions are possible in terms of t-norms and conorms.

Definition 6.2.1 Binary operations $\blacktriangle : [0,1]^2 \to [0,1]$ and $\blacktriangledown : [0,1]^2 \to [0,1]$ are called **triangular norm** and **triangular conorm** (briefly t-norm or t-conorm) if for $u, v, w \in [0,1]$ the following hold

- commutativity: $u \blacktriangle v = v \blacktriangle u$ and $u \blacktriangledown v = v \blacktriangledown u$
- associativity: $u \blacktriangle (v \blacktriangle w) = (u \blacktriangle v) \blacktriangle w$ and $u \blacktriangledown (v \blacktriangledown w) = (u \blacktriangledown v) \blacktriangledown w$,

- monotony: $u \leq v \Rightarrow u\blacktriangle w \leq v\blacktriangle w$ and $u \leq v \Rightarrow u\blacktriangledown w \leq v\blacktriangledown w$, and
- boundary neutrality: $1\blacktriangle u = u$ and $0\blacktriangledown u = u$.

Every t-norm can be used to define the meet of two propositions $x \wedge y(\omega) = x(\omega)\blacktriangle y(\omega)$ and a t-conorm the join and, consequently, turn the fuzzy logic into a poset by determining the corresponding partial order relation $\preceq$. Obviously, the min function is a t-norm (also called Gödel t-norm) while max is a t-conorm. Specific choices of the t-norm and -conorm lead to different variants of fuzzy logic (Łukasiewicz, Gödel, product logics, etc.).

Definition 6.2.2 Every t-norm $\blacktriangle$ induces a **material implication** on the fuzzy logic, denoted by $\rightarrow_\blacktriangle$, as the proposition defined by

$$(x \rightarrow_\blacktriangle y)(\omega) = \sup\{s \in [0,1] : x(\omega)\blacktriangle s \leq y(\omega)\}$$

for arbitrary propositions x and y.

Example 6.2.3 [Gödel material implication] Let the t-norm be defined by Gödel prescription: $x\blacktriangle y = \min(x, y)$. Then

$$x \rightarrow_\blacktriangle y = \begin{cases} 1 & \text{if } x \preceq y \\ y & \text{if } x \succ y. \end{cases}$$

The proposition $x \rightarrow_\blacktriangle y$ is false in the extent that x is “truer” than y.

Fuzzy logic is instrumental in several applied fields where sharp memberships are not available. In particular it can be used to perform fuzzy clustering as a generalisation of the classical (Boolean) clustering algorithms introduced in section 4.

Recently, fuzzy logic (with Gödel t-norm, t-conorm prescriptions) was used by Aerts [1] to analyse an experiment in cognitive science performed in 1988 by Hampton [30]. Namely, 40 students have been asked to assign a number in the set $\{-3, -2, -1, 0, 1, 2, 3\}$ to quantify the typicality of an “item being element of a concept”. $+3$ means that the item is judged being strongly pertinent to the concept, -3 strongly impertinent. The data were then transformed to produce empirical fuzzy memberships to the concepts i.e. propositions of the fuzzy logic. The novelty of the experiment laid in the fact that students were also asked to rank memberships to disjunction of concepts. Therefore, we have two manners to compute joins of propositions, either by the empirical membership determined for the disjunction of concepts, or by applying t-norm to the empirical memberships of individual concepts. It turned out that there is a discrepancy among these two determinations, implying that fuzzy logical operations do not fully explain the experimental observations. Aerts continued the analysis and encoded the experimental results to a quantum logic instead; this modification significantly improved the coïncidence of the two determinations.

We view these results as a strong indication that in some situations analysis of experimental data based on methods of quantum logic is more rewarding than using traditional methods. We don't speculate here on a hypothetical quantum nature of cognitive activities as was done in [55]; we only claim that quantum methods applied in the analysis can give better results. For that reason, we introduce quantum logic in section 6.2.3 below and examine the way to introduce probability measures on it in section 6.3.

6.2.3 Quantum logic

Quantum logic has been introduced in definition 6.1.11 as a complemented, orthomodular, non-contradictory logic $\mathbb{X}$; it can be assumed σ-complete if $\mathbb{X}$ is infinite. In the same manner propositions in a Boolean logic arise as indicators on a family of subsets of a universal set and those of fuzzy logic as membership functions, the natural model representing a quantum logic arises as the set of projections to closed subspaces of a Hilbert space $\mathcal{H}$.

More precisely, we have the following

Proposition 6.2.4 *Let $\mathcal{H}$ be a separable complex Hilbert space of dimension $\dim \mathcal{H} \geq 2$. Denote by $\mathbb{X} = \mathcal{P}(\mathcal{H})$ the set of self-adjoint operators with spectrum contained in $\{0,1\}$ and for $x, y \in \mathbb{X}$ define*

- $x' = \mathbb{1} - x$,
- $[\mathsf{im}(x) \subseteq \mathsf{im}(y)] \Rightarrow [x \preceq y]$.

Then $\mathbb{X}$ is the set of projectors to closed subspaces of $\mathcal{H}$; it is also a (σ-complete) quantum logic.

Proof: Since all experiments analysed in section 8 are modelled by finite dimensional Hilbert spaces, we only prove this proposition in the case of finite dimensional Hilbert spaces. Since every $x \in \mathbb{X}$ is self-adjoint and has $\mathsf{spec}(x) \subseteq \{0,1\}$, it follows that $x^2 = x$, i.e. x is a projection. Now, projections are in bijection with subspaces of the Hilbert space. It is then trivial to verify that pseudo-complementation x' verifies $x = (x')'$; consequently the conjuctive de Morgan law holds. Moreover, if $x \preceq y$ then $xy = yx = x$; define then the meet by $x \wedge y = xy = yx$. Thanks to the conjunctive de Morgan law, we have $x \vee y = (x' \wedge y')' = \mathbb{1} - x' \wedge y' = \mathbb{1} - (\mathbb{1} - x)(\mathbb{1} - y) = x + y - xy$. It is then immediate to verify that the pseudo-complementation is in fact a complementation and that the set $\mathbb{X}$ verifies the axioms of a quantum logic. □

Obviously, the quantum logic defined above fails to be distributive in general. Consider in fact a two-dimensional Hilbert space $\mathcal{H}$ and ξ, ψ two othonormal vectors in $\mathcal{H}$. Denote by x the projection to $\mathsf{span}(\xi)$, by y the projection to $\mathsf{span}(\psi)$ and by z the projection to $\mathsf{span}(\xi + \psi)$. Denote further $\mathbf{0}$ the projection to the trivial subspace $\{0\}$ and $\mathbf{1}$ the projection (identity) to $\mathcal{H}$. Then $x \vee (y \wedge z) = x \vee \mathbf{0} = x$, while $(x \vee y) \wedge (x \vee z) = \mathbf{1} \wedge \mathbf{1} = \mathbf{1}$. Hence $x \vee (y \wedge z) \neq (x \vee y) \wedge (x \vee z)$. Similarly, $x \wedge (y \vee z) = x \wedge \mathbf{1} = x$, while $(x \wedge y) \vee (x \wedge z) = \mathbf{0} \vee \mathbf{0} = \mathbf{0}$. Hence $x \wedge (y \vee z) \neq (x \wedge y) \vee (x \wedge z)$.

Now, the main use of logic is to make inferences. The minimal requirement that a quantum material implication $\rightarrow$ must verify is obviously

$$\text{if } [x \preceq y], \text{ then } [x \rightarrow y] = \mathbf{1} \text{ (law of entailement).}$$

However, this requirement is not enough to fully determine the logical structure. In classical logic we can make conditional inference of the form: if x is true and $x \rightarrow y$ holds then y is true, while if $x \rightarrow y$ holds and y is false then x is false. Therefore, a quantum material implication must also verify

$$x \wedge (x \rightarrow y) \preceq y \text{ (modus ponens),}$$

and

$$y' \wedge (x \rightarrow y) \preceq x' \text{ (modus tollens).}$$

The law of entailement, modus ponens, and modus tollens are collectively called **minimal inference criteria**. In a quantum logic there exist three different conditional operations, expressible as lattice polynomials, that verify the minimal inference criteria:

$$\begin{aligned} x \rightarrow_1 y &= x' \vee (x \wedge y) \\ x \rightarrow_2 y &= (x' \wedge y') \vee y \\ x \rightarrow_3 y &= (x \wedge y) \vee (x' \wedge y) \vee (x' \wedge y'). \end{aligned}$$

All these conditionals reduce to the classical material implication

$$x \rightarrow y = x' \vee y \text{ (classical material implication)}$$

when restricted to a Boolean sublogic of the quantum logic. However, they all violate certain implicative criteria that are satisfied by the classical material implication. One such violated criterion is

$$\text{if } (x \wedge z) \preceq y, \text{ then } z \preceq (x \rightarrow y) \text{ (law of exportation).}$$

As a matter of fact, minimal inference criteria and the law of exportation are equivalent to the

$$x \wedge z \preceq y \text{ if and only if } z \preceq (x \rightarrow y) \text{ (classical implicative criterion).}$$

Proposition 6.2.5 *Let $\mathbb{X}$ be an orthocomplemented lattice. If there exists a conditional $\rightarrow$ that verifies the classical implicative criterion then $\mathbb{X}$ is distributive and the conditional is in fact the classical material implication.*

Proof: Let x, y, z be three arbitrary elements of $\mathbb{X}$. Obviously,

$$\begin{aligned} x \wedge z &\preceq (x \wedge z) \vee (y \wedge z) \\ y \wedge z &\preceq (x \wedge z) \vee (y \wedge z), \end{aligned}$$

and since the conditional $\rightarrow$ verifies the classical implicative criterion, we conclude that

$$\begin{aligned} x &\preceq [z \rightarrow [(x \wedge z) \vee (y \wedge z)]] \\ y &\preceq [z \rightarrow [(x \wedge z) \vee (y \wedge z)]] \,. \end{aligned}$$

Combining disjunctively the two above inequalities, we get

$$x \vee y \preceq [z \rightarrow [(x \wedge z) \vee (y \wedge z)]],$$

and applying the classical implicative criterion once more, we obtain finally

$$(x \vee y) \wedge z \preceq [(x \wedge z) \vee (y \wedge z)].$$

Since, by theorem 6.1.6 the reverse inequality holds in any lattice, we conclude that the inequality is in fact an equality, proving thus the distributivity of the lattice. The proof is completed by remarking that in a distributive lattice the classical material implication is the only one verifying the minimal inference criteria. □

Corollary 6.2.6 *On a the lattice $\mathbb{X} = \mathcal{P}(\mathcal{H})$ of Hilbert subspaces, for a Hilbert space with $\dim \mathcal{H} \geq 2$, none of the conditionals $\rightarrow_i$, with $i = 1, 2, 3$ defined above satisfies the classical implicative criterion; hence, each of them violates the law of exportation.*

Proof: Obvious since $\mathbb{X}$ is not distributive. □

Classical probability theory is defined on a measurable space $(\Omega, \mathcal{F})$ where $\mathcal{F}$ is a Boolean σ-complete logic. Quantum mechanics, originally mathematically formulated in a different language and point of view, can be viewed as a probability theory (i.e. a theory of random variables and probability measures) on a quantum σ-complete logic. Weakening distributivity to orthomodularity, beyond modifying the quantum material implication, induces a structure on the quantum probability space richer than the Boolean one, with some new features, absent from the classical case.

6.3 Classical and quantum probability in a unified framework

The propositions considered so far are of a particularly simple type: they correspond to indicators in the Boolean case and to projectors in the quantum case. Our purpose is to construct a genuine probability theory [70, 59, 57, 56], i.e. define random variables as integrals with respect to an indicator-valued measure (in the Boolean case) or projector-valued measure (in the quantum case) over their spectral values. We shall use the term “classical” to speak about Boolean logics.

Axiom 6.3.1 *In any physical system (classical or quantum), the set of all experimentally verifiable propositions is a logic (classical or quantum).*

6.3.1 Observables associated with a logic

Suppose that $\mathbb{X}$ is the logic of verifiable propositions of a physical system and let X be any real physical quantity relative to this system. Denoting by $x(B)$

the proposition "the numerical results of the observation of X lie in B", it is natural and harmless to consider that $B \in \mathcal{B}(\mathbb{R})$; obviously then, x is a mapping $x : \mathcal{B}(\mathbb{R}) \to \mathbb{X}$. We regard two physical quantities X and X' as identical whenever the corresponding maps $x, x' : \mathcal{B}(\mathbb{R}) \to \mathbb{X}$ are the same. If $f : \mathbb{R} \to \mathbb{R}$ is a Borel function, we mean by $X' = f \circ X$ a physical quantity taking value $f(r)$ whenever X takes value r. The corresponding map is given by $\mathcal{B}(\mathbb{R}) \ni B : x' \mapsto x'(B) = x(f^{-1}(B)) \in \mathbb{X}$. Hence we are led naturally to the following

Definition 6.3.2 Let $\mathbb{X}$ be a logic. A real *observable* associated with $\mathbb{X}$ is a mapping $x : \mathcal{B}(\mathbb{R}) \to \mathbb{X}$ verifying:

1. $x(\emptyset) = \mathbf{0}$ and $x(\mathbb{R}) = \mathbf{1}$,
2. if $B_1, B_2 \in \mathcal{B}(\mathbb{R})$ with $B_1 \cap B_2 = \emptyset$ then $x(B_1) \perp x(B_2)$,
3. if $(B_n)_{n\in\mathbb{N}}$ is a sequence of mutually disjoint Borel sets, then $x(\cup_{n\in\mathbb{N}} B_n) = \vee_{n\in\mathbb{N}} x(B_n)$.

We write $\mathcal{O}(\mathbb{X})$ for the set of all real observables associated with $\mathbb{X}$.

Let $\mathbb{X}$ be a logic and $x \in \mathcal{O}(\mathbb{X})$. Then, for any sequence of Borel sets $(B_n)_{n\in\mathbb{N}}$, we have

$$x(\cup_{n\in\mathbb{N}} B_n) = \vee_{n\in\mathbb{N}} x(B_n)$$

and

$$x(\cap_{n\in\mathbb{N}} B_n) = \wedge_{n\in\mathbb{N}} x(B_n).$$

Definition 6.3.3 Let $\mathbb{X}$ be a logic and $\mathcal{O}(\mathbb{X})$ the set of its associated observables. A real number λ is called a *strict value* of an observable $x \in \mathcal{O}(\mathbb{X})$, if $x(\{\lambda\}) \neq \mathbf{0}$. The observable $x \in \mathcal{O}(\mathbb{X})$ is called *discrete* if there exists a countable set $C = \{c_1, c_2, \ldots\}$ such that $x(C) = \mathbf{1}$; it is called *constant* if there exists $c \in \mathbb{R}$ such that $x(\{c\}) = \mathbf{1}$. It is called *bounded* if there exists a compact Borel set K such that $x(K) = \mathbf{1}$.

Definition 6.3.4 We call *spectrum* of $x \in \mathcal{O}(\mathbb{X})$ the closed set defined by

$$\mathsf{spec}(x) = \cap_{C \text{ closed } : x(C) = \mathbf{1}} C.$$

The numbers $\lambda \in \mathsf{spec}(x)$ are called *spectral values* of x.

Any strict value is a spectral value; the converse is not necessarily true. Obviously, $\lambda \in \mathsf{spec}(x)$ if and only if any open set U containing λ verifies $x(U) \neq \mathbf{0}$.

If $(a_n)_{n\in\mathbb{N}}$ is a partition of unity, i.e. a family of mutually orthogonal propositions in $\mathbb{X}$ such that $\vee_{n\in\mathbb{N}} a_n = \mathbf{1}$, there exists a unique discrete observable admitting as spectral values a given discrete subset $\{c_1, c_2, \ldots\}$ of the reals. In fact, it is enough to define for all $n \in \mathbb{N}$, $x(\{c_n\}) = a_n$ and for any $B \in \mathcal{B}(\mathbb{R})$, $x(B) = \vee_{n : c_n \in B} a_n$. Notice however that discrete observables do not exhaust all the physics of quantum mechanics; important physical phenomena involve continuous observables.

6.3.2 States on a logic

With every classical system is associated a measurable space $(\Omega, \mathcal{F})$ (its phase space); observables are random variables and states are probability measures that may degenerate to Dirac masses on particular points of the phase space. This description is incompatible with the experimental observation for quantum systems. For the latter, the Heisenberg's uncertainty principle stipulates that no matter how carefully the system is prepared, there always exist observables whose values are distributed according to some non-trivial probability distribution.

Definition 6.3.5 Let $\mathbb{X}$ be a logic and $\mathcal{O}(\mathbb{X})$ its set of associated observables. A *state function* is a mapping $\rho : \mathcal{O}(\mathbb{X}) \ni x \mapsto \rho_x \in \mathcal{M}_1^+(\mathbb{R}, \mathcal{B}(\mathbb{R}))$.

For every Borel function $f : \mathbb{R} \to \mathbb{R}$, for every observable x, and every Borel set B on the line, we have:

$$\rho_{f \circ x}(B) = \rho_x(f^{-1}(B)).$$

Denoting by o the zero observable and 0 the zero of $\mathbb{R}$, we have that $\rho_o = \delta_0$. In fact, suppose that $f : \mathbb{R} \to \mathbb{R}$ is the identically zero map. Then $f \circ o = o$ and

$$f^{-1}(B) = \begin{cases} \mathbb{R} & \text{if } 0 \in B \\ \emptyset & \text{otherwise.} \end{cases}$$

Hence, if $0 \in B$, then $\rho_o(B) = \rho_{f \circ o}(B) = \rho_o(f^{-1}(B)) = 1$, because ρ_o is a probability on $\mathbb{R}$; if $0 \notin B$ then similarly $\rho_o(B) = 0$. Therefore, in all circumstances, $\rho_o(B) = \delta_0(B)$.

If $x \in \mathcal{O}(\mathbb{X})$ is any observable and $B \in \mathcal{B}(\mathbb{R})$ is such that $x(B) = \mathbf{0} \in \mathbb{X}$, then $\rho_x(B) = 0$. In fact, for this B, we have $\mathbb{1}_B \circ x = o$ and $\rho_x(B) = \rho_o(\{1\}) = \delta_0(\{1\}) = 0$. This implies that if x is discrete, the measure ρ_x is supported by the set of the strict values of x.

Definition 6.3.6 An observable $q \in \mathcal{O}(\mathbb{X})$ is a *question* if $q(\{0, 1\}) = \mathbf{1}$. A question is the necessarily discrete. If $q(\{1\}) = a \in \mathbb{X}$, then q is the only question such that $q(\{1\}) = a$; we call it *question associated with the proposition a* and denote by q_a if necessary.

Definition 6.3.7 Let $\mathbb{X}$ be a logic. A function $p : \mathbb{X} \to [0, 1]$ satisfying

1. $p(\mathbf{0}) = 0$ and $p(\mathbf{1}) = 1$,
2. if $(a_n)_{n \in \mathbb{N}}$ is a sequence of mutually orthogonal propositions of $\mathbb{X}$, and $a = \vee_{n \in \mathbb{N}} a_n$, then $p(a) = \sum_{n \in \mathbb{N}} p(a_n)$

is called *state (or probability measure) on the logic* $\mathbb{X}$. The set of states on $\mathbb{X}$ is denoted by $\mathcal{S}(\mathbb{X})$.

The concept of probability measure on a logic coincides with a classical probability measure when the logic is a Boolean σ-algebra. For non distributive logics however, the associated probability measures are genuine generalisations of the classical probabilities. For standard quantum logics, the associated states are called quantum probabilities.

Theorem 6.3.8 *Let $p \in \mathcal{S}(\mathbb{X})$, where $\mathbb{X}$ is a logic.*

1. *On defining a map $\rho^p : \mathcal{O}(\mathbb{X}) \to \mathcal{M}_1^+(\mathbb{R}, \mathcal{B}(\mathbb{R}))$, by the formula: for every $x \in \mathcal{O}(\mathbb{X})$ and for every $B \in \mathcal{B}(\mathbb{R})$, $\rho^p_x(B) = p(x(B))$, then ρ^p is a state function.*
2. *Conversely, if ρ is an arbitrary state function, then there exists a unique probability measure $p \in \mathcal{S}(\mathbb{X})$ such that, for every $x \in \mathcal{O}(\mathbb{X})$ and for every $B \in \mathcal{B}(\mathbb{R})$, we have $\rho_x(B) = p(x(B))$.*

Proof:

1. The map $\rho^p_x : \mathcal{B}(\mathbb{R}) \to [0,1]$ is certainly a σ-additive, non-negative map. Moreover, $\rho^p_x(\mathbb{R}) = p(\mathbf{1}) = 1$, hence it is a probability. If $f : \mathbb{R} \to \mathbb{R}$ is a Borel function,

   $$\rho^p_{f\circ x}(B) = p(f \circ x(B)) = p(x(f^{-1}(B))) = \rho^p_x((f^{-1}(B)).$$

   Hence ρ^p is a state function.

2. Let ρ be a state function. If $a \in \mathbb{X}$ and $q_a \in \mathcal{O}(\mathbb{X})$ the question associated with proposition a, then ρ_{q_a} is a probability measure on $\mathcal{B}(\mathbb{R})$. Since q_a is a question, $\rho_{q_a}(\{0,1\}) = 1$. Define $p(a) = \rho_{q_a}(\{1\})$. Obviously, for all $a \in \mathbb{X}$, $p(a)$ is well defined and is taking values in $[0,1]$. It remains to show that p is a probability measure on $\mathbb{X}$, that is to say verify σ-additivity and normalisation. For $\mathbf{0} \in \mathbb{X}$, $q_{\mathbf{0}}(\{1\}) = \mathbf{0}$. Hence $\rho_{q_{\mathbf{0}}}(\{1\}) = 0 = p(\mathbf{0})$. Similarly, we show that $= p(\mathbf{1}) = 1$. This shows normalisation.

   Let $(a_n)_{n\in\mathbb{N}}$ be a sequence of mutually orthogonal elements of $\mathbb{X}$, and denote by $a = \vee_{n\in\mathbb{N}} a_n$. Let $x \in \mathcal{O}(\mathbb{X})$ be the discrete observable defined by $x(\{0\}) = a^\perp$ and $x(\{n\}) = a_n$, for $n = 1, 2, \ldots$. Then, $\mathbb{1}_{\{n\}} \circ x(\{1\}) = x(\{n\}) = a_n$. Hence $q_{a_n} = \mathbb{1}_{\{n\}} \circ x$ and $p(a_n) = \rho_x(\{n\})$. Since ρ_x is a probability measure, $\sum_n p(a_n) = \rho_x(\{1,2,3,\ldots\}) = \rho_x(\mathbb{N})$. Similarly, $\mathbb{1}_\mathbb{N} \circ x = q_a$ because $\mathbb{1}_\mathbb{N} \circ x(\{1\}) = x(\mathbb{N}) = \vee_{n\in\mathbb{N}} x(\{n\}) = \vee_{n\in\mathbb{N}} a_n = a$. Hence, finally, $p(a) = \sum_n p(a_n)$ establishing thus σ-additivity of p. Finally, for $x \in \mathcal{O}(\mathbb{X})$ and $B \in \mathcal{B}(\mathbb{R})$,

   $$\rho_x(B) = \rho_{\mathbb{1}_B \circ x}(\{1\}) = \rho_{q_{x(B)}}(\{1\}) = p(x(B)).$$

□

If $p \in \mathcal{S}(\mathbb{X})$ and $x \in \mathcal{O}(\mathbb{X})$, the map $\mathcal{B}(\mathbb{R}) \ni B \mapsto p(x(B)) \in [0,1]$ defines a probability measure on $\mathcal{B}(\mathbb{R})$. It is called the *probability distribution* induced on the space of its values by the observable x when the system is in state p and is denoted ρ_x^p. The *expected value* of x in state p is

$$\mathbb{E}_p(x) = \int_{\mathbb{R}} t\rho_x^p(dt)$$

and for a Borel function $f : \mathbb{R} \to \mathbb{R}$, we have

$$\mathbb{E}_p(f \circ x) = \int_{\mathbb{R}} f(t)\rho_x^p(dt)$$

(provided the above integrals exist.) If $\mathbb{E}_p(x^2) < \infty$, the *variance* of x in p is $\mathsf{Var}_p(x) = \mathbb{E}_p(x^2) - (\mathbb{E}_p(x))^2$.

Axiom 6.3.9 *Observables of a physical system described by the logic $\mathbb{X}$ are $\mathcal{O}(\mathbb{X})$.*

Axiom 6.3.10 *States of a physical system described by the logic $\mathbb{X}$ are $\mathcal{S}(\mathbb{X})$.*

Axiom 6.3.11 *Measuring whether the values of a physical observable $x \in \mathcal{O}(\mathbb{X})$ lie in $B \in \mathcal{B}(\mathbb{R})$ when the system is prepared in state $p \in \mathcal{S}(\mathbb{X})$ means determining $\rho_x^p(B)$.*

6.3.3 Pure states, superposition principle, convex decomposition

Proposition 6.3.12 *Let $\mathcal{S}(\mathbb{X})$ be the set of states on the logic $\mathbb{X}$. Let $(p_n)_{n\in\mathbb{N}}$ be a sequence in $\mathcal{S}(\mathbb{X})$ and $(c_n)_{n\in\mathbb{N}}$ a sequence in $\mathbb{R}_+$ such that $\sum_{n\in\mathbb{N}} c_n = 1$. Then $p = \sum_{n\in\mathbb{N}} c_n p_n$, defined by $p(a) = \sum_{n\in\mathbb{N}} c_n p_n(a)$ for all $a \in \mathbb{X}$, is a state.*

Proof: Obvious! □

Corollary 6.3.13 *For any logic $\mathbb{X}$, the set $\mathcal{S}(\mathbb{X})$ is convex.*

Remark 6.3.14 Notice that if $p = \sum_{n\in\mathbb{N}} c_n p_n$ as above, for every $x \in \mathcal{O}(\mathbb{X})$, we have that $\rho_x^p = \sum_{n\in\mathbb{N}} c_n \rho_x^{p_n}$. In fact, for all $B \in \mathcal{B}(\mathbb{R})$,

$$\rho_x^p(B) = p(x(B)) = \sum_{n\in\mathbb{N}} c_n p_n(x(B)) = \sum_{n\in\mathbb{N}} c_n \rho_x^{p_n}(B).$$

This decomposition has the following interpretation: the sequence $(c_n)_{n\in\mathbb{N}}$ defines a classical probability on $\mathbb{N}$ meaning that in the sum defining p, each p_n is chosen with probability c_n. Therefore, for each integrable observable $x \in \mathcal{O}(\mathbb{X})$, the expectation $\mathbb{E}_p(x) = \sum_{n\in\mathbb{N}} c_n \mathbb{E}_{p_n}(x)$ consists in two averages: a classical average on the choice of p_n and a (may be) quantum average $\mathbb{E}_{p_n}(x)$.

Definition 6.3.15 A state $p \in \mathcal{S}(\mathbb{X})$ is said to be *pure* if the equation $p = cp_1 + (1-c)p_2$, for $p_1, p_2 \in \mathcal{S}(\mathbb{X})$ and $c \in [0,1]$ implies $p = p_1 = p_2$. We write $\mathcal{S}_p(\mathbb{X})$ for the set of pure states of $\mathbb{X}$. Obviously $\mathcal{S}_p(\mathbb{X}) = \mathsf{Extr}\,\mathcal{S}(\mathbb{X})$.

Definition 6.3.16 Let $\mathcal{D} \subseteq \mathcal{S}(\mathbb{X})$ and $p_0 \in \mathcal{S}(\mathbb{X})$. We say that p_0 is a *superposition of states in* $\mathcal{D}$ if for $a \in \mathbb{X}$,

$$\forall p \in \mathcal{D}, p(a) = 0 \Rightarrow p_0(a) = 0.$$

It is iimediate to show that the state $p = \sum_{n \in \mathbb{N}} c_n p_n$ defined in the proposition 6.3.12 is a superposition of states in $\mathcal{D} = \{p_1, p_2, \ldots\}$. In the case $\mathbb{X}$ is a Boolean σ-algebra, the next theorem 6.3.17 shows that this is in fact the only kind of possible superposition. This implies, in particular, the *unicity* of the decomposition of a classical state into extremal (pure) states. If $\mathbb{X}$ is a standard quantum logic, unicity of the decomposition does not hold any longer! Technically, the convex structure of classical states is described by a so-called **Choquet simplex**; the corresponding convex structure for quantum states is not Choquet simplex.

Remark: This difference is manifested even for the simplest finite dimensional situation: classical probability measures over a set of two elements is isomorphic to the unit segment $[0,1]$; quantum probability measures over a two-dimensional complex Hilbert space is isomorphic to the unit ball in $\mathbb{R}^3$. In the classical situation the extremal states correspond to the boundary points 0 and 1 of the segment, by identifying them with Dirac measures δ_0 and δ_1 and any internal point of the segment can be uniquely decomposed as a convex combination of the extremal ones. In the quantum case, extremal states are again on the boundary of the set (the unit sphere in this situation); points in the interior of the ball can still be represented as convex combinations of extremal points but this decomposition is not unique since there are infinitely many inequivalent such decompositions.

Theorem 6.3.17 *Let $\mathbb{X}$ be a Boolean σ-algebra of subsets of a space $\mathbb{X}$. Suppose that*

1. *$\mathbb{X}$ is separable*[2]*,*
2. *for all $a \in \mathbb{X}$, $\{a\} \in \mathbb{X}$.*

For any $a \in \mathbb{X}$ and any $A \subseteq \mathbb{X}$, let δ_a be the state defined by

$$\delta_a(A) = \begin{cases} 1 & \text{if } a \in A \\ 0 & \text{otherwise.} \end{cases}$$

Then, $(\delta_a)_{a \in \mathbb{X}}$ is precisely the set of all pure states in $\mathbb{X}$. If $\mathcal{D} \subseteq \mathcal{S}_p(\mathbb{X})$ and $p_0 \in \mathcal{S}_p(\mathbb{X})$, then p_0 is a superposition of states in $\mathcal{D}$ if and only if $p_0 \in \mathcal{D}$.

[2] i.e. there is a countable collection of subsets $A_n \subseteq \mathbb{X}$, $n \in \mathbb{N}$, generating $\mathbb{X}$ by complementation, intersections, and unions.

Proof: Denote $\{A_1, A_2, \ldots\}$ a denumerable collection of subsets of $\mathbb{X}$ generating $\mathbb{X}$. Purity of δ_a is trivially verified. Suppose that p is a pure state. If for some $A_0 \in \mathbb{X}$ we have $0 < p_0(A) < 1$, then, on putting for $A \in \mathbb{X}$

$$p_1(A) = \frac{1}{p(A_0)} p(A \cup A_0) \tag{$*$}$$

and

$$p_2(A) = \frac{1}{1 - p(A_0)} p(A \cap A_0^c), \tag{$**$}$$

we get $p(A) = p(A_0)p_1(A) + (1 - p(A_0))p_2(A)$. Yet, applying (*) and (**) to A_0, we get $p_1(A_0) = 1$ and $p_2(A_0) = 0$, hence $p_1 \neq p_2$. This is in contradiction with the assumed purity of p. Therefore, we conclude that for all $A \in \mathbb{X}$, we have $p(A) \in \{0, 1\}$. Replacing A_n by A_n^c if necessary, we can assume without loss of generality that $p(A_n) = 1$ for all the sets of the collection generating $\mathbb{X}$. Let $B = \cap_n A_n$. Then $p(B) = 1$ and consequently B cannot be empty. Now B cannot contain more than one point either. In fact, the collection of all sets $C \in \mathbb{X}$ such that either $B \subseteq C$ or $B \cap C = \emptyset$ is a σ-algebra containing all the sets A_n, $n \in \mathbb{N}$. Hence, it coincides with $\mathbb{X}$. As singletons are members of $\mathbb{X}$, the set B must be a singleton, i.e. $B = \{a\}$ for some $a \in \mathbb{X}$. Put then $p = \delta_a$. Finally, let p_0 be a superposition of states in $\mathcal{D}$ (all its elements are pure states). If $p_0 = \delta_{a_0}$ but $p_0 \notin \mathcal{D}$, then $p(\{a_0\}) = 0$ for all $p \in \mathcal{D}$ but $p_0(\{a_0\}) \neq 0$, a contradiction. □

6.3.4 Simultaneous observability

Another distinctive property of non-commutative probability theory, known as the Heisenberg's uncertainty principle, is that there are observables (quantum random variables) that cannot be simultaneously observed with arbitrary precision.

Definition 6.3.18 Let $a, b \in \mathbb{X}$. Propositions a and b are said to be *simultaneously verifiable*, denoted by $a \leftrightarrow b$, if there exists elements $a_1, b_1, c \in \mathbb{X}$ such that

1. a_1, b_1, c are mutually orthogonal and,
2. $a = a_1 \vee c$ and $b = b_1 \vee c$ hold.

Observables $x, y \in \mathcal{O}(\mathbb{X})$ are *simultaneously observable* if for all $B \in \mathcal{B}(\mathbb{R})$, $x(B) \leftrightarrow y(B)$. For $A, B \subseteq \mathbb{X}$, we write $A \leftrightarrow B$ if for all $a \in A$ and all $b \in B$ we have $a \leftrightarrow b$.

Lemma 6.3.19 *Let $a, b \in \mathbb{X}$. The following are equivalent:*

1. *$a \leftrightarrow b$,*
2. *$a \wedge (a \wedge b)^\perp \perp b$,*

3. $b \wedge (a \wedge b)^{\perp} \perp a$,

4. there exist $x \in \mathcal{O}(\mathbb{X})$ *and* $A, B \in \mathcal{B}(\mathbb{R})$ *such that* $x(A) = a$ *and* $x(B) = b$,

5. there exists a Boolean sub-algebra of $\mathbb{X}$ *containing* a *and* b.

Proof:

$1 \Rightarrow 2$:

$$\begin{aligned} a \leftrightarrow b \quad & \Leftrightarrow \quad a = a_1 \vee c \text{ and } b = b_1 \vee c \\ & \Rightarrow \quad c \leq a \text{ and } c \leq b \\ & \Rightarrow \quad c \leq a \wedge b. \end{aligned}$$

From the definition 6.1.8 (logic), it follows that there exists $d \in \mathbb{X}$ such that $c \perp d$ and $c \vee d = a \wedge b$.

Now $d \leq c \vee d = a \wedge b \leq a$ and $d \leq c^{\perp}$ (since $d \perp d$.) Hence, $d \leq a \wedge c^{\perp} = a_1$ (see remark immediately following the definition 6.1.8.) Similarly, $d \leq b_1 \Rightarrow d \leq b_1 \wedge q_1 = \mathbf{0}$. Therefore $d = \mathbf{0}$ and consequently $c = a \wedge b$. It follows $a_1 = a \wedge (a \wedge b)^{\perp}$. Yet, $a_1 \perp c$ and $a_1 \perp b_1$ so that $a_1 \perp (b_1 \perp c) = b$. Summarising, $a \wedge (a \wedge b)^{\perp} \perp b$.

$1 \Rightarrow 3$: By symmetry.

$2 \Rightarrow 1$: Since $a \wedge (a \wedge b)^{\perp} \perp b$, on writing $a_1 = a \wedge (a \wedge b)^{\perp}$, $b_1 = b \wedge (a \wedge b)^{\perp}$, and $c = a \wedge b$, we find $a = a_1 \vee c$ and $b = b_1 \vee c$. Since $a_1 \perp b$, it follows that $a_1 \perp b_1$ and $a_1 \perp c$, while, by definition, $c \perp b_1$ which proves the implication.

Henceforth, the equivalence $1 \Leftrightarrow 2 \Leftrightarrow 3$ is established.

$1 \Rightarrow 4$: If $a = a_1 \vee c$, $b = b_1 \vee c$ and a_1, b_1, c mutually orthogonal, write $d = a_1 \vee b_1 \vee c$ and define x to be the discrete observable such that $x(\{0\}) = a_1$, $x(\{1\}) = b_1$, $x(\{2\}) = c$, and $x(\{3\}) = d$. Then $x(\{0, 2\}) = a$ and $x(\{1, 2\}) = b$.

$4 \Rightarrow 5$: $x(A \cap (A \cap B)^c) = a \wedge (a \wedge b)^{\perp}$ and $x(B \cap (A \cap B)^c) = b \wedge (a \wedge b)^{\perp}$. On writing $a_1 = a \wedge (a \wedge b)^{\perp}$, $a_2 = a \wedge b$, $a_3 = b \wedge (a \wedge b)^{\perp}$, and $a_4 = (a \vee b)^{\perp}$, we see that $(a_i)_{i=1,\ldots,4}$ are mutually orthogonal and $a_1 \vee a_2 \vee a_3 \vee a_4 = \mathbf{1}$. If

$$\mathcal{A} = \{a_{i_1} \vee \ldots \vee a_{i_k} : k \leq 4; 1 \leq i_1 \leq \ldots \leq i_k \leq 4\},$$

it is easily verified that $\mathcal{A}$ is Boolean sub-algebra of $\mathbb{X}$. Since $a, b \in \mathcal{A}$, this proves the implication.

$5 \Rightarrow 2$: Let $\mathcal{A}$ be a Boolean sub-algebra of $\mathbb{X}$ containing a and b. Now, $[a \wedge (a \wedge b)^{\perp}] \wedge b = \mathbf{0}$. As $a, b, a \wedge (a \wedge b)^{\perp}, b^{\perp} \in \mathcal{A}$, it follows that

$$\begin{aligned} a \wedge (a \wedge b)^{\perp} \quad & = \quad [(a \wedge (a \wedge b)^{\perp}) \wedge b] \\ & \quad\quad \vee [(a \wedge (a \wedge b)^{\perp}) \wedge b^{\perp}] \\ & = \quad [(a \wedge (a \wedge b)^{\perp}) \wedge b^{\perp}] \\ & \leq \quad b^{\perp}. \end{aligned}$$

Therefore $a \wedge (a \wedge b)^{\perp} \perp b$.

□

The significance of this lemma is that if two propositions are simultaneously verifiable, we can operate on them as if they were classical.

Theorem 6.3.20 *Let $\mathbb{X}$ be any logic and $(x_\lambda)_{\lambda\in D}$ a family of observables. Suppose that $x_\lambda \leftrightarrow x_{\lambda'}$ for all $\lambda, \lambda' \in D$. Then there exist a space Ω, a σ-algebra $\mathcal{F}$ of subsets of Ω, a family of measurable functions $g_\lambda : \Omega \to \mathbb{R}$, $\lambda \in D$, and a σ-homomorphism $\tau : \mathcal{X} \to \mathbb{X}$ such that $\tau(g_l^{-1}(B) = x_\lambda(B)$ for all $\lambda \in D$ and all $b \in \mathcal{B}(\mathbb{R})$. Suppose further that either $\mathbb{X}$ is separable or D is countable. Then, for all $\lambda \in D$, there exist a $x \in \mathcal{O}(\mathbb{X})$ and a measurable function $f_\lambda : \mathbb{R} \to \mathbb{R}$ such that $x_\lambda = f_\lambda \circ x$.*

The proof of this theorem is omitted. Notice that it allows to construct functions of several observables that are simultaneously observable. This latter result is also stated without proof.

Theorem 6.3.21 *Let $\mathbb{X}$ be any logic and $(x_1, \ldots, x_n)$ a family of observables that are simultaneously observable. Then there exists a σ-homomorphism τ : $\mathcal{B}(\mathbb{R}^n) \to \mathbb{X}$ such that for all $B \in \mathcal{B}(\mathbb{R})$ and all $i = 1, \ldots, n$,*

$$x_i(B) = \tau(\pi_i^{-1}(B)), \tag{$*$}$$

where $\pi_i : \mathbb{R}^n \to \mathbb{R}$ is the projection $\pi(t_1, \ldots, t_n) = t_i$, $i = 1, \ldots, n$. If g is a Borel function on $\mathbb{R}^n$, then $g \circ (x_1, \ldots, x_n)(B) = \tau(g^{-1}(B))$ is an observable. If $g_1, \ldots, g_k$ are real valued Borel functions on $\mathbb{R}^n$ and $y_i = g_i \circ (x_1, \ldots, x_n)$, then $y_1, \ldots, y_k$ are simultaneously observable and for any real valued Borel function h on $\mathbb{R}^k$, we have $h \circ (y_1, \ldots, y_k) = h(g_1, \ldots, g_k) \circ (x_1, \ldots, x_n)$ where, for $\mathbf{t} = (t_1, \ldots, t_n)$, $h(g_1, \ldots, g_k)(\mathbf{t}) = h(g_1(\mathbf{t}), \ldots, g_k(\mathbf{t}))$.

An immediate consequence of this theorem is that if p is a probability measure on $\mathbb{X}$, then $\rho^p_{x_1,\ldots,x_n}(B) = p(\tau(B))$, for $B \in \mathcal{B}(\mathbb{R}^n)$, is the *joint probability distribution* of $(x_1, \ldots, x_n)$ in state p.

Chapter 7

The method of semantic distillation

The method of semantic distillation is a recursive divisive fuzzy clustering followed by a projection on a Hilbert subspace and a thinning of the graph. Each step, that can be thought as a subroutine of the whole algorithm, is described in the following sections.

7.1 Fuzzy divisive 2-clustering

Let $\mathbb{B}$ be a set of documents, $\mathbb{A}$ a set of attributes, and $\rho : \mathbb{V} \cup \mathbb{A} \to \mathbb{R}^\nu$ a (not necessarily faithful) geometric representation of both documents and attributes. We denote collectively $\mathbb{V} = \mathbb{A} \cup \mathbb{B}$ and call its elements objects. Suppose further that a fuzzy membership function $m : \mathbb{V} \times \mathcal{P}(\mathbb{A}) \to [0, 1]$ assigning a degree of membership of every object $v \in \mathbb{V}$ in a subset $A \subseteq \mathbb{A}$ is given. The purpose of a fuzzy divisive 2-clustering algorithm is to construct a partition of $\mathbb{A}$ into 2 subsets $(C_k)_{k=1,2}$ and lump objects into each such subset according to their fuzzy membership [12].

The algorithm starts with the set $\mathbb{A}$ and constructs a sequence of disjoint subsets of $\mathbb{A}$ indexed by the words κ of finite length on a two-letter alphabet. This set is isomorphic to a subset of the rooted binary tree. If κ is the root, then define $\mathbb{M}_\kappa = \mathbb{A}$. Otherwise, $\mathbb{M}_\kappa$ will be a proper subset of $\mathbb{A}$, i.e. $\emptyset \subset \mathbb{M}_\kappa \subset \mathbb{A}$, indexed by κ. When $|\mathbb{M}_\kappa| = 1$ then the corresponding κ is a leaf of the binary tree. The algorithm stops when all indices correspond to leaves.

More precisely, let $\mathbb{K} = \{1, 2\}$, $\mathbb{K}^0 = \{\kappa : \kappa = ()\}$, and for integers $n \geq 1$ let $\mathbb{K}^n = \{\kappa : \kappa = \kappa_1 \cdots \kappa_n; \kappa_i \in \mathbb{K}\}$. Finally let $\mathbb{K}^* = \cup_{n \geq 0} \mathbb{K}^n$ denote the set of words on two letters of indefinite length, including the empty sequence, denoted by $()$, of zero length that coincides with the root of the tree. If $\kappa = \kappa_1 \cdots \kappa_n$ is a word of n letters and $k \in \mathbb{K}$, we denote the concatenation κk as the word of $n + 1$ letters $\kappa_1 \cdots \kappa_n k$.

We start from the empty set $\mathsf{Leaves} = \{\}$, the empty sequence $\kappa = ()$ and the current attributes set $\mathbb{M}_\kappa = \mathbb{M}_{()} = \mathbb{A}$ and current tree $\mathsf{Tree} = \{\kappa\}$. We denote $\mathbb{V}_\kappa = \mathbb{B} \cup \mathbb{M}_\kappa$. The initial fuzzy membership function m induces a sequence of fuzzy membership functions indexed by the elements of the tree $m_\kappa : \mathbb{V}_\kappa \times \mathbb{K} \to [0, 1]$, with $m_{()} \equiv m$. The fuzzy clustering algorithm is succinctly described as Algorithm 4 below.

```
Data: $\kappa$, $\mathbb{M}_\kappa$, $\rho$, ObjCostFun
Result: Two sets $\mathbb{M}_{\kappa k}$ for $k = 1, 2$
the fuzzy membership $m_\kappa(v, k)$ for $v \in \mathbb{V}_\kappa$ in the clusters $\mathbb{M}_{\kappa k}$
if $|\mathbb{M}_\kappa| > 1$ then
    assign $(v_1, v_2) \leftarrow \arg\max\{\|\rho(v) - \rho(v')\|, v, v' \in \mathbb{V}_\kappa\}$;
    assign $\rho(v_1)$ and $\rho(v_2)$ as centroids for the two putative finer clusters
    $\mathbb{M}_{\kappa 1}$ and $\mathbb{M}_{\kappa 2}$;
    minimise ObjCostFun under the constraint $\sum_{k=1}^{2} m(v, k) = 1$, for all
    $v \in \mathbb{V}_\kappa$ ;
    assign $\mathbb{M}_{\kappa 1} \leftarrow \{v \in \mathbb{M}_\kappa : m_\kappa(v, 1) > m_\kappa(v, 2)\}$;
    assign $\mathbb{M}_{\kappa 2} \leftarrow \mathbb{M}_\kappa \setminus \mathbb{M}_{\kappa 1}$;
end
```

Algorithm 4: FuzzyClustering

Note that once the two clusters $\mathbb{M}_{\kappa 1}$ and $\mathbb{M}_{\kappa 2}$ are determined in every intermediate step, the fuzzy membership m_κ allows determining, for every cluster, an ordered list of documents $(b_1^i, \ldots, b_{|\mathbb{B}|}^i)$, $i = 1, 2$ by $m_\kappa(b_k^i, i) \geq m_\kappa(b_l^i, i)$ whenever $k \leq l$.

To be fully determined, this algorithm need a precise specification of the cost function $\mathsf{ObjCostFun}$ and of the sequence of fuzzy memberships m_κ. The fuzzy memberships will be specified in a subsequent section **??**. We give here an example of $\mathsf{ObjCostFun}$. A 2-clustering of the set $\mathbb{V}$ is equivalent with an element of $\beta \in \{1, 2\}^{\mathbb{V}}$, i.e. an assignment to every $v \in \mathbb{V}$ of a label $\beta(v)$ in $\{1, 2\}$. Define then

$$\mathsf{ObjCostFun}(\beta) = \sum_{v \in \mathbb{V}} m(v, \beta(v)) \|\rho(v) - \rho(v_{\beta(v)})\|,$$

where v_1 and v_2 are the centroids of the putative clusters. Other variants can of course be used, like $\mathsf{ObjCostFun}(\beta) = \sum_{v \in \mathbb{V}} m(v, \beta(v)) \|\rho(v) - \rho(v_{\beta(v)})\|^\alpha$, with some $\alpha > 0$. Although the precise numerical results depend on the particular choice of the function, no qualitative difference is observed.

7.2 The state described by the collection of documents

As explained in the previous chapters all the available information about a DNA array experiment is encoded into the dataset matrix $\mathsf{X} \in \mathfrak{M}_{|\mathbb{A}|, |\mathbb{B}|}$. In general, in the experiments we analysed, we have $|\mathbb{A}| \ll |\mathbb{B}|$. Without loss of generality, we can assume further that the matrix X is of full rank $\mathsf{rank}\mathsf{X} = |\mathbb{A}|$. Now, singular value decomposition reads

$$\mathsf{X} = \sum_{k=1}^{|\mathbb{A}|} s_k | u_k \rangle \langle V_k |,$$

where $u_k \in \mathbb{C}^{\mathbb{A}}$ and $V_k \in \mathbb{C}^{\mathbb{B}}$ are unitary vectors. Therefore

$$\mathsf{X}\mathsf{X}^\dagger = \sum_{k=1}^{|\mathbb{A}|} s_k^2 | u_k \rangle \langle u_k |,$$

is a self-adjoint positive definite matrix in $\mathfrak{M}_{|\mathbb{A}|,|\mathbb{A}|}$, reminiscent of a density matrix in quantum mechanics. To become precisely a density matrix, it remains to normalise it to unit trace, by

$$\sigma = \frac{\mathsf{X}\mathsf{X}^\dagger}{\mathsf{tr}(\mathsf{X}\mathsf{X}^\dagger)} = \sum_{k=1}^{|\mathbb{A}|} p_k | u_k \rangle \langle u_k |,$$

where $p_k = \frac{s_k^2}{\sum_{l=1}^{|\mathbb{A}|} s_l^2}$.

This writing has a profound intuitive meaning: σ is a state on the Hilbert space $\mathcal{H}_{\mathbb{A}}$ *encoding the information on the attributes induced by the experimental observation.* Note that both $(| a \rangle, a \in \mathbb{A})$ and $(| u_k \rangle, k = 1, \ldots, |\mathbb{A}|)$ are orthonormal bases of this space but the latter is more appropriate to represent elementary meanings as linear combinations of attributes

$$| u_k \rangle = \sum_{a \in \mathbb{A}} \Upsilon_{ak} | a \rangle, k = 1, \ldots, |\mathbb{A}|.$$

In other words, the attributes do not in general represent elementary meanings in this density matrix formalism; on the contrary *linear superpositions of attributes convey the appropriate meanings induced by the context of the particular experimental situation.* Once a state is constructed, the standard formalism of quantum mechanics can be deployed to determine, by a kind of quantum measurement, the degree of pertinence of every document to an attribute as a fuzzy membership (in fact a quantum probability) of documents to clusters of attributes.

More precisely, suppose that some putative cluster C_i, $i = 1, 2$ is proposed by the previous algorithm. In quantum mechanical terms, this cluster is to be viewed as a projection $P_i = \sum_{a \in C_i} | a \rangle \langle a |$, for $i = 1, 2$. Asking whether the pertinent attributes lie in C_i is equivalent in performing the quantum measurement of the observable P_i in state σ. According to the rules governing quantum measurement the state is then transformed as

$$\sigma \mapsto \Phi_i(\sigma) = \frac{P_i \sigma P_i}{\mathsf{tr}(P_i \sigma)}.$$

Now,

$$P_i \sigma P_i = \sum_{a', a'' \in \mathbb{A}} | a' \rangle \left(\sum_{k=1}^{|\mathbb{A}|} p_k \langle a' | u_k \rangle \langle u_k | a'' \rangle \right) \langle a'' |,$$

yielding $\mathsf{tr}(P_i\sigma) = \sum_{k=1}^{|\mathbb{A}|} p_k \sum_{a\in C_i} |\Upsilon_{ak}|^2$. It is then natural to define the (fuzzy) membership of attribute a in cluster C_i by

$$\begin{aligned} m(a,i) &= c_a\,\mathsf{tr}(|\,a\,\rangle\langle\,a\,|P_i\sigma P_i) \\ &= \begin{cases} c_a \sum_{k=1}^{|\mathbb{A}|} p_k|\Upsilon_{ak}|^2 & \text{if } a\in C_I \\ 0 & \text{otherwise,} \end{cases} \end{aligned}$$

where c_a is a proportionality constant fixed by the constraint $\sum_{i=1}^{2} m(a,i) = c_a \sum_{k=1}^{|\mathbb{A}|} p_k = 1$. It easily seen therefore that the thus determined (fuzzy) membership is as a matter of fact a crisp membership. Defining the membership in the previous manner, is however not as trivial as the previous remark might suggest. Consider in fact the fuzzy membership of an arbitrary document $b\in\mathbb{B}$ in the cluster C_i. Considering ξ_b the normalised vector representing b in the Hilbert space $\mathcal{H}_{\mathbb{A}}$, we have:

$$\begin{aligned} m(b,i) &\propto \mathsf{tr}(|\,\xi_b\,\rangle\langle\,\xi_b\,|P_i\sigma P_i) \\ &= \sum_{a',a''\in C_i}\sum_{k=1}^{|\mathbb{A}|} p_k\langle\,a'\,|\,u_k\,\rangle\langle\,u_k\,|\,a''\,\rangle\,\mathsf{tr}\left(|\,\xi_b\,\rangle\langle\,\xi_b\,||\,a'\,\rangle\langle\,a''\,|\right) \\ &= \sum_{k=1}^{|\mathbb{A}|} p_k \sum_{a',a''\in C_i} \langle\,u_k\,|\,a''\,\rangle\langle\,a''\,|\,\xi_b\,\rangle\langle\,\xi_b\,|\,a'\,\rangle\langle\,a'\,|\,u_k\,\rangle. \end{aligned}$$

Since the summation variables a', a'' are dummy, the right hand side of the previous equality is obviously real. The proportionality constant is determined by the condition

$$1 = \sum_{i=1}^{2} m(b,i) \propto \sum_{k=1}^{|\mathbb{A}|} p_k|\langle\,u_k\,|\,\xi_b\,\rangle|^2.$$

Remark: The previous relationships corroborate the claim made earlier that the natural meaning directions are not given by the orhonormal basis $(|\,a\,\rangle, a\in\mathbb{A})$ but by the basis $(|\,u_k\,\rangle, k=1,\ldots,|\mathbb{A}|)$.

The above procedure is inspired by the theory of quantum measurement as introduced in [73] and further extended to generalised measurements (see for instance [2, 10, 51]). Repeated applications of this kind of measurement leads to a purification of the quantum state [46]. Of course, a natural fundamental question can be asked: *is the information contained in such an experiment really of quantum nature? The answer is definitely not.* We can as a matter of fact make an arbitrary number of copies (violating thus the quantum noncloning theorem) of the intermediate vector states before acting on them by projections. Therefore, the information is definitely classical. The only thing that is quantum in this formalism is the intuition of the state collapse and its purification by successive measurements leading to the disambiguation of the initial information, encoded into mixed quantum states, to pure states.

7.3 Semantic distillation

The algorithm of semantic distillation starts with the Hilbert space $\mathcal{H}_{\mathbb{A}}$ and the graph with vertex set $G^0 = \mathbb{A} \cup \mathbb{B}$ and constructs a sequence of Hilbert subspaces and subgraphs Note that in the previous construction $\mathbb{M}_{\kappa k} \subset \mathbb{M}_{\kappa}$ for every κ and every $k \in \mathbb{K}$. Therefore, the algorithm explores the branches of a tree from the root to the leaves. Denote by π_κ the orthogonal projection from $\mathcal{H}_{\mathbb{A}}$ to $\mathcal{H}_{\mathbb{M}_\kappa}$. The distillation step is described by the following Algorithm 5.

```
Data: FuzzyClustering
Result: Leaves and sequence of singleton sets $\mathbb{M}_\kappa$ for $\kappa \in$ Leaves
Initialisation{
$\kappa \leftarrow ()$;
$\mathbb{M}_\kappa \leftarrow \mathbb{A}$;
Leaf($\kappa$) $\leftarrow \mathbb{M}\kappa$;
Leaves $\leftarrow \{\}$;
Tree $\leftarrow \{\kappa\}$;
Bookkeeping $\leftarrow \{\kappa\}$;
}
while Bookkeeping $\neq \emptyset$ do
    for $\kappa \in$ Bookkeeping do
        if $|\mathbb{M}_\kappa| = 1$ then
            Leaves $\leftarrow$ Leaves $\cup \{\kappa\}$;
            Bookkeeping $\leftarrow$ Bookkeeping $\setminus \{\kappa\}$;
        else
            Use $\pi_\kappa$ to project from $\mathcal{H}_{\mathbb{A}}$ to $\mathcal{H}_{\mathbb{M}_\kappa}$;
            Thin the graph: $\mathbb{V}_\kappa \leftarrow \mathbb{B} \cup \mathbb{M}_\kappa$;
            Compute weighed Laplacian $L$ on $\mathbb{V}_\kappa$;
            Diagonalise $L$;
            Compute $\nu$-dimensional representation $\mathbf{r}$;
            Call FuzzyClustering;
            for $k \in \mathbb{K}$ do
                $\kappa' \leftarrow \kappa k$;
                Leaf($\kappa'$) $\leftarrow \mathbb{M}_{\kappa'}$ /* $\mathbb{M}_\kappa$ as determined by FuzzyClustering */;
                Tree $\leftarrow$ Tree $\cup \{\kappa'\}$;
                Bookkeeping $\leftarrow$ Bookkeeping $\cup \{\kappa'\}$;
            end
        end
    end
end
```

Algorithm 5: Distillation

To illustrate the method, let us limit ourselves to the eigenvector u_2 corresponding to the second smallest eigenvalue of L, providing us with the best one-dimensional representation of the graph. Now, u_2 is a vector of $\mathbb{R}^N$, with $N = |G^0|$. Suppose we order the N components of u_2 in ascending order. This ordering corresponds to a bijective relabelling $r : G^0 \to \{1, \ldots, N\}$ of the vertices so that for any $i, j \in \{1, \ldots, N\}$ with $i < j$, we have $u_2(r^{-1}(i)) <$

$u_2(r^{-1}(j))$. Therefore the set of vertices becomes now a totally ordered set isomorphic to the set $\{1, \ldots, N\}$.

Now recall that the set G^0 is the set $\mathbb{B}$ of documents, augmented by the set $\mathbb{A}$ of attributes, the latter acting as specificity witnesses. In the relabelling induced by u_2, the specificity witnesses are relabelled as elements $\{r(a_1), \ldots, r(a_{|\mathbb{A}|})\} = \{\alpha_1, \ldots, \alpha_{|\mathbb{A}|}\} \subseteq \{1, \ldots, N\}$, with $\alpha_1 \leq \ldots \leq \alpha_{|\mathbb{A}|}$. Let

$$i = \arg\max\{u_2(r^{-1}(\alpha_j)) - u_2(r^{-1}(\alpha_{j-1})) : j = 2, \ldots, |\mathbb{A}|\}$$

correspond to the most frank separation of the attributes and define $\mathbb{A}_1 = \{r^{-1}(\alpha_1), \ldots, r^{-1}(\alpha_{i-1})\}$ and $\mathbb{A}_2 = \mathbb{A} \setminus \mathbb{A}_2$. Cluster the documents by assigning

$$C_{\mathbb{A}_1} = \{v \in G^0 : \forall a \in \mathbb{A}_2, |u_2(v) - u_2(a)| > \min_{a' \in \mathbb{A}_1} |u_2(v) - u_2(a')|\},$$

and similarly for $C_{\mathbb{A}_2}$ by exchanging the roles of $\mathbb{A}_1$ and $\mathbb{A}_2$. The set $C_{\mathbb{A}_1}$ contains the documents the most specific[1] to attributes $\mathbb{A}_1$ and similarly for the set $C_{\mathbb{A}_2}$.

Semantic distillation is a recursive procedure producing partitions $\mathbb{A}_1$ and $\mathbb{A}_2$ of subsets of $\mathbb{A}$ repeated until both sets $\mathbb{A}_1$ and $\mathbb{A}_2$ become singletons. The first step of semantic distillation corresponds to an orthogonal projection $P = \sum_{a \in \mathbb{D}} |\, a \,\rangle\langle\, a \,|$ from the Hilbert space $\mathcal{H}_{\mathbb{A}}$ onto the Hilbert subspace $\mathcal{H}_{\mathbb{D}}$, where the set $\mathbb{D}$ is the smallest of the above defined sets $\mathbb{A}_1$ and $\mathbb{A}_2$. The graph construction is started afresh on the vertex set of the *thinned graph* $G^0 \setminus C_{\mathbb{D}^c}$ and weights computed in terms of the Hilbert norm on the Hilbert subspace $\mathcal{H}_{\mathbb{D}^c}$. This gives rise to a new weight matrix W and a new Laplacian L on $\mathbb{B} \cup \mathbb{D}^c$. We proceed then with the computation of eigenvalues and eigenvectors of this new Laplacian leading to separation of remaining attributes into two new subsets and new clustering. The method ends when all objects are clustered.

[1] For a specificity crisply determined by the graph representation. Note however that nothing prevents us from using more sophisticated specificiy, for instance the quantum specificity introduced in the previous section.

Chapter 8

Application

The previous formalism has been applied to the bio-informatical problem explained in the introduction. An integrated computer code, written in C, has been developed (the main routines are given in appendix A) and applied to analyse various experimental datasets.

8.1 Analysis of data concerning tissues

This biological experiment clusters genes according to their tissular specificity. The obtained results have been published in the articles

- Semantic distillation: a method for clustering objects by their contextual specicity,“Studies in computational intelligence”, Springer-Verlag (2008), 431–442, Volume 129/2008. [65] and
- Robustness and efficiency of semantic distillation to retrieve genes by their contextual specificity, submitted for publication.

These two articles are reproduced here.

Semantic distillation: a method for clustering objects by their contextual specificity

Thomas Sierocinski[1], Antony Le Béchec[2], Nathalie Théret[2], and Dimitri Petritis[1]

[1] Institut de recherche mathématique (UMR6625), Université de Rennes 1
[2] INSERM U620, Université de Rennes 1

Summary. Techniques for data-mining, latent semantic analysis, contextual search of databases, etc. have long ago been developed by computer scientists working on information retrieval (IR). Experimental scientists, from all disciplines, having to analyse large collections of raw experimental data (astronomical, physical, biological, etc.) have developed powerful methods for their statistical analysis and for clustering, categorising, and classifying objects. Finally, physicists have developed a theory of quantum measurement, unifying the logical, algebraic, and probabilistic aspects of queries into a single formalism.

The purpose of this paper is twofold: first to show that when formulated at an abstract level, problems from IR, from statistical data analysis, and from physical measurement theories are very similar and hence can profitably be cross-fertilised, and, secondly, to propose a novel method of fuzzy hierarchical clustering, termed *semantic distillation* — strongly inspired from the theory of quantum measurement —, we developed to analyse raw data coming from various types of experiments on DNA arrays. We illustrate the method by analysing DNA arrays experiments and clustering the genes of the array according to their specificity.

Keywords: Quantum information retrieval, semantic distillation, DNA microarray, quantum and fuzzy logic

1 Introduction

Sequencing the genome constituted a culminating point in the analytic approach of Biology. Now starts the era of the synthetic approach in Systems Biology where interactions among genes induce their differential expression that leads to the functional specificity of cells, the coherent organisation of cells into tissues, organs, and finally organisms.

However, we are yet far from a complete explanatory theory of living matter. It is therefore important to establish precise and quantitative phenomenology before being able to formulate a theory. The contribution of this paper is to provide the reader with a novel algorithmic method, termed *semantic distillation*, to analyse DNA arrays experiments (where genes are hybridised with various cell lines corresponding to various tissues or specific individuals) by determining the degree of specificity of every gene to the particular context. The method provides experimental biologists with lists of candidate genes (ordered by their degree of specificity) for every biological context, clinicians with improved tools for diagnosis, pharmacologists with patient-tailored therapies, etc.

In the sequel we present the method split into several algorithmic tasks thought as subroutines of the general algorithm. It is worth noting that the method, although can profitably exploit, does not rely on any previous information stored in the existing databases; its rationale is to help analysing raw experimental data even in the absence of any previous knowledge.

The main idea of the method is summarised as follows. Experimental information hold on the objects of the system undergoes a sequence of processing steps; each step is performed on a different representation of the information. Those different representation spaces and the corresponding information processing act as successive filters revealing at the end the most pertinent and significant part of the information, hence the name “semantic distillation”.

At the first stage, raw experimental data, containing all available information, are represented in an abstract Hilbert space, *the space of concepts* — reminiscent of the space of pure states in Quantum Mechanics —, endowing the set of objects with a metric space structure that is exploited to quantify the interactions among objects and encode them into a weighed graph on the vertex set of objects and with object interactions as edge weights.

Now objects (genes) are parts of an organised system (cell, tissue, organism). Therefore their mutual interactions are not just independent random variables; they are interconnected through precise, although certainly very complicated and mostly unknown relationships. We seek to reveal (hidden and unknown) interactions among genes. This is achieved by trading the weighed graph representation for a low-dimensional representation and using spectral properties of the weighed Laplacian on the graph to grasp the essential interactions.

The following step consists in a fuzzy divisive clustering of objects among two subsets by exploiting the previous low-dimensional representation. This procedure assigns a fuzzy membership to each object relative to characters of the two subsets. Fuzziness is as a matter of fact a distinctive property of experimental biological data reflecting our incomplete knowledge of fundamental biological processes.

Up to this step, our method is a sequence of known algorithms that have been previously used separately in the literature in various contexts. The novelty of our method relies on the following steps. The previous fuzzy clustering reduced the indeterminacy of the system. This information is fed back to the system to perform a projection to a proper Hilbert subspace. In that way, the information content of the dataset is modified by the information gained by the previous observations. After this feeding back, the three previous steps are repeated but now referring to a Hilbert spaces of lower dimension. Therefore our method is not a mere fuzzy clustering algorithm but a genuine non-classical interaction information retrieval procedure where previous observations alter the informational content of the system, reminiscent of the measurement procedure in Quantum Mechanics.

2 A Hilbert space formulation

2.1 Mathematical form of the dataset

Let $\mathbb{B}$ be a finite set of *documents* (or objects, or books) and $\mathbb{A}$ a finite set of *attributes* (or contexts, or keywords). The dataset is a $|\mathbb{B}| \times |\mathbb{A}|$ matrix $\mathsf{X} = (x_{ba})_{b\in\mathbb{B},a\in\mathbb{A}}$ of real or complex elements, where $|\cdot|$ represents cardinality. Equivalent ways of representing the dataset are

- a collection of $|\mathbb{B}|$ row vectors $\mathbf{x}_b = (x_{b1},\ldots,x_{b|\mathbb{A}|}), b \in \mathbb{B}$ of $\mathbb{R}^{|\mathbb{A}|}$ (or $\mathbb{C}^{|\mathbb{A}|}$),
- a collection of $|\mathbb{A}|$ column vectors $\mathbf{x}^a = (x_{1a},\ldots,x_{|\mathbb{B}|a}), a \in \mathbb{A}$ of $\mathbb{R}^{|\mathbb{B}|}$ (or $\mathbb{C}^{|\mathbb{B}|}$).

Example 1. In the experiments we analysed $\mathbb{B}$ is a set of 12000 human genes and $\mathbb{A}$ a set of 12 tissular contexts. The matrix elements x_{ba} are real numbers encoding luminescence intensities (or their logarithms) of DNA array ultimately representing the level of expression of gene b in context a.

Example 2. Let $\mathbb{B}$ be a set of books in a library and $\mathbb{A}$ a set of bibliographic keywords. The matrix elements x_{ba} can be $\{0,1\}$-valued: if the term a is present in the book b then $x_{ba} = 1$ else $x_{ba} = 0$. A variant of this example is when x_{ba} are integer valued: if the term a appears k times in document b then $x_{ba} = k$.

Example 3. Let $\mathbb{B}$ be a set of students and $\mathbb{A}$ a set of papers they gave. The matrix elements x_{ba} are real valued; x_{ba} is the mark the student b got in paper a.

The previous examples demonstrate the versatility of the method by keeping the formalism at an abstract level to apply indistinctively into various very different situations without any change. Note also that the assignment as set of documents or attributes is a matter of point of view; for instance, example 3 as it stands is convenient in evaluating students. Interchanging the role of sets $\mathbb{A}$ and $\mathbb{B}$ renders it adapted to the evaluation of teaching. As a rule of thumb, in biological applications, $|\mathbb{A}| \ll |\mathbb{B}|$.

2.2 The space of concepts

For $\mathbb{A}$ and $\mathbb{B}$ as in the previous subsection, we define the *space of concepts*, $\mathscr{H}_{\mathbb{A}}$, as the real or complex free vector space over $\mathbb{A}$, i.e. elements of $\mathbb{A}$ serve as indices of an orthonormal basis of $\mathscr{H}_{\mathbb{A}}$. Therefore, the complete dataset X can be represented as the collection of $|\mathbb{B}|$ vectors $|\,\Xi_b\,\rangle = \sum_{a\in\mathbb{A}} x_{ba}|\,a\,\rangle \in \mathscr{H}_{\mathbb{A}}$, with $b \in \mathbb{B}$ and where $|\,a\,\rangle$ represents the element of the orthonormal basis of the free vector space corresponding to the attribute a. We use here Dirac's notation to represent vectors, linear forms and projectors on this space (see any book on quantum mechanics or [26] for a freely accessible document and [29] for the use of this notation in information retrieval). The vector $|\,\Xi_b\,\rangle$ contains all available experimental information on document b in various cellular contexts indexed by the attributes a; it can be thought as a convenient bookkeeping device of the data $(x_{ba})_{a\in\mathbb{A}}$, in the same way a generating function contains all the information on a sequence as formal power series.

The vector space is equipped with a scalar product defined for every two vectors $|\psi\rangle = \sum_{a\in\mathbb{A}} \psi_a |a\rangle$ and $|\psi'\rangle = \sum_{a\in\mathbb{A}} \psi'_a |a\rangle$ by $\langle\psi|\psi'\rangle = \sum_{a\in\mathbb{A}} \overline{\psi}_a \psi'_a$, where $\overline{\psi}_a$ denotes the complex conjugate of ψ_a (it coincides with ψ_a if it is real). Equipped with this scalar product, the vector space $\mathscr{H}_{\mathbb{A}}$ becomes a real or complex $|\mathbb{A}|$-dimensional Hilbert space. The scalar product induces a Hilbert norm on the space, denoted by $\|\cdot\|$. In the sequel we introduce also *rays* on the Hilbert space i.e. normalised vectors. Since the dataset X does not in principle verify any particular numerical constraints, rays are constructed by dividing vectors by their norms. We use the symbol $|\xi_b\rangle = |\Xi_b\rangle/\||\Xi_b\rangle\|$ to denote the ray associated with vector $|\Xi_b\rangle$.

The Hilbert space structure on $\mathscr{H}_{\mathbb{A}}$ allows a natural geometrisation of the space of documents by equipping it with a pseudo-distance[3] $d:\mathbb{B}\times\mathbb{B}\to\mathbb{R}_+$ defined by $d(b,b') = \||\xi_b\rangle - |\xi_{b'}\rangle\|$. What is important here is not the precise form of the pseudo-metric structure of $(\mathbb{B},d)$; several other pseudo-distances can be introduced, not necessarily compatible with the scalar product. In this paper we stick however to the previous pseudo-distance, postponing into a later publication explanations about the significance of other pseudo-distances.

As is the case in Quantum Mechanics, the Hilbert space description incorporates into a unified algebraic framework all logical and probabilistic information hold by the dataset. An enquiry of the type "does the system possess feature F" is encoded into a projector P_F acting on the Hilbert space. The subspace associated with the projector P_F is interpreted as the set of documents retrieved by asking the question about the feature F. Now all experimental information hold by the dataset is encoded into the *state* of the system represented by a *density matrix* ρ (i.e. a self-adjoint, positive, trace class operator acting on $\mathscr{H}_{\mathbb{A}}$ having unit trace). Retrieved documents possess the feature F with probability $\mathtt{tr}(\rho P_F)$. Thus the algebraic description incorporates logical information on the documents retrieved as relevant to a given feature and assign them a probability determined by the state defined by the experiment. For example, the probability that a gene b is relevant to an attribute a is given by the above formula with $P = |a\rangle\langle a|$ and $\rho = |\xi_b\rangle\langle\xi_b|$, yielding $\mathtt{tr}(\rho P) = |\langle\xi_b|a\rangle|^2$.

3 A weighed graph with augmented vertex set

The careful reader has certainly already noted that in the above description vectors $|\xi_b\rangle$, encoding the information about document b, and basis vectors $|a\rangle$, associated with attribute a, all belong to the same Hilbert space $\mathscr{H}_{\mathbb{A}}$. Therefore, although initially the sets $\mathbb{A}$ and $\mathbb{B}$ are disjoint since they have distinct elements, when passing to the Hilbert space representation, vectors $|\xi_b\rangle$ and $|a\rangle$ have very similar roles in representing indistinguishably objects or attributes as vectors of $\mathscr{H}_{\mathbb{A}}$. In the sequel, we introduce the set $\mathbb{V}$ (or more precisely $\mathbb{V}_{\mathbb{A}}$ to remove any ambiguity) as the set $\mathbb{V}_{\mathbb{A}} = \mathbb{A}\cup\mathbb{B}$. Thus, for any $v\in\mathbb{V}_{\mathbb{A}}$,

[3] It is termed pseudo-distance since it verifies symmetry and triangle inequality but $d(b,b')$ can vanish even for different b and b'. As a matter of fact, d is a distance on the projective Hilbert space.

$$|\,\Xi_v\,\rangle = \begin{cases} |\,a\,\rangle & \text{if } v = a \in \mathbb{A}, \\ \sum_{a\in\mathbb{A}} x_{ba} |\,a\,\rangle & \text{if } v = b \in \mathbb{B}. \end{cases}$$

The new vectors $|\,\Xi_a\,\rangle = |\,a\,\rangle$ are included as *specificity witnesses* in the dataset. Note that since these new vectors are also elements of the same Hilbert space, the pseudo-distance d naturally extends to $\mathbb{V}_{\mathbb{A}}$.

Suppose now that a *similarity function* $\sigma : \mathbb{V}_{\mathbb{A}} \times \mathbb{V}_{\mathbb{A}} \to [0,1]$ is defined. For the sake of definiteness, the reader can think of σ as being given, for example[4], by $\sigma(v,v') = \sqrt{1 - \frac{1}{2}d(v,v')^2}$; results we quote in section 5 are obtained with a slight modification of this similarity function. However, again, the precise form of the similarity function is irrelevant in the abstract setting serving as foundation of the method. Several other similarity functions have been used like, for example, $\sigma(v,v') = \exp(-\|\Xi_v - \Xi_{v'}\|^2/\tau)$ with τ a positive constant or some others, in particular, functions taking value 0 even for some vertices corresponding to non orthogonal rays but the explanation of their significance is postponed to a subsequent publication.

A weighed graph is now constructed with vertex set $\mathbb{V}_{\mathbb{A}}$. Weights are assigned to the edges of the complete graph over $\mathbb{V}_{\mathbb{A}}$; the weights being expressible in terms of the similarity function σ. Again, the precise expression is irrelevant for the exposition of the method. For the sake of concreteness, the reader can suppose that the weights $W_{vv'}$ are given by $W_{vv'} = \sigma(v,v')$. The pair $(\mathbb{V}_{\mathbb{A}}, W)$ with W being the symmetric matrix $W = (W_{vv'})_{v,v \in \mathbb{V}_{\mathbb{A}}}$, denotes the weighed graph.

At this level of the description we follow now standard techniques of reduction of the data dimensionality by optimal representation of the graph in low dimensional Euclidean spaces spanned by eigenvectors of the Laplacian. Such methods have been used by several authors [4, 24]. Here we give only the basic definitions and main results of this method. The interested reader may consult standard textbooks like [8, 10, 15] for general exposition of the method.

Definition 1. *A map* $\mathbf{r} : \mathbb{V}_{\mathbb{A}} \to \mathbb{R}^{\nu}$ *is called a* ν*-dimensional representation of the graph. The representation is always supposed non-trivial (i.e.* $\mathbf{r} \not\equiv 0$*) and balanced (i.e.* $\sum_{v\in\mathbb{V}_{\mathbb{A}}} \mathbf{r}(v) = 0$*).*

From the weights matrix W we construct the *weighed Laplacian matrix* $\Lambda = D - W$ where the matrix elements $D_{vv'}$ are 0 if $v \neq v'$ and equal to $\sum_{v''\in\mathbb{V}_{\mathbb{A}}} W_{vv''}$ if $v = v'$. More precisely, we denote by $\Lambda(\mathbb{V}_{\mathbb{A}})$ this weighed Laplacian to indicate that it is defined on the vertex set $\mathbb{V}_{\mathbb{A}}$. This precision will be necessary in the next section specifying the semantic distillation algorithm where the vertex set will be recursively modified at each step. The *weighed energy of the representation* is given by

$$\mathscr{E}_W(\mathbf{r}) = \sum_{v,v'\in\mathbb{V}_{\mathbb{A}}} W_{vv'} \|\mathbf{r}(v) - \mathbf{r}(v')\|^2,$$

where in this formula $\|\cdot\|$ denotes the Euclidean norm of $\mathbb{R}^{\nu}$.

[4] This function is well adapted to datasets $\mathsf{X} = (x_{ba})$, with $x_{ba} \in \mathbb{R}^{+}$; for more general datasets, the factor $1/2$ must be changed to $1/4$.

Theorem 1. *Let $N = |\mathbb{V}_{\mathbb{A}}|$ and $\{\lambda_1, \ldots, \lambda_N\}$ be the spectrum of Λ, ordered as $\lambda_1 \leq \lambda_2 \ldots \lambda_N$. Suppose that $\lambda_2 > 0$. Then $\inf_{\mathbf{r}} \mathscr{E}_W(\mathbf{r}) = \sum_{i=2}^{\nu+1} \lambda_i$, where the infimum is over all ν-dimensional non-trivial balanced representations of the graph.*

Remark 1. If $\mathbf{u}^1, \ldots \mathbf{u}^N$ are the eigenvectors of Λ corresponding to the eigenvalues $\lambda_1, \ldots, \lambda_N$ ordered as above, then $\mathbf{u}^2$ is the best one-dimensional, $[\mathbf{u}^2, \mathbf{u}^3]$ the best two-dimensional, etc, $[\mathbf{u}^2, \ldots, \mathbf{u}^{\nu+1}]$ the best ν-dimensional representation of the graph $(\mathbb{V}_{\mathbb{A}}, W)$.

4 Fuzzy semantic clustering and distillation

The algorithm of semantic distillation is a recursive divisive fuzzy clustering followed by a projection on a Hilbert subspace and a thinning of the graph. It starts with the Hilbert space $\mathscr{H}_{\mathbb{A}}$ and the graph with vertex set $\mathbb{V}_{\mathbb{A}}$ and constructs a sequence of Hilbert subspaces and subgraphs indexed by the words κ of finite length on a two-letter alphabet. This set is isomorphic to a subset of the rooted binary tree. If κ is the root, then define $\mathbb{M}_\kappa = \mathbb{A}$. Otherwise, $\mathbb{M}_\kappa$ will be a proper subset of $\mathbb{A}$, i.e. $\emptyset \subset \mathbb{M}_\kappa \subset \mathbb{A}$,indexed by κ. When $|\mathbb{M}_\kappa| = 1$ then the corresponding κ is a leaf of the binary tree. The algorithm stops when all indices correspond to leaves.

More precisely, let $\mathbb{K} = \{1, 2\}$, $\mathbb{K}^0 = \{\kappa : \kappa = ()\}$, and for integers $n \geq 1$ let $\mathbb{K}^n = \{\kappa : \kappa = \kappa_1 \cdots \kappa_n; \kappa_i \in \mathbb{K}\}$. Finally let $\mathbb{K}^* = \cup_{n \geq 0} \mathbb{K}^n$ denote the set of words on two letters of indefinite length, including the empty sequence, denoted by $()$, of zero length that coincides with the root of the tree. If $\kappa = \kappa_1 \cdots \kappa_n$ is a word of n letters and $k \in \mathbb{K}$, we denote the concatenation κk as the word of $n+1$ letters $\kappa_1 \cdots \kappa_n k$.

We start from the empty set $\mathsf{Leaves} = \{\}$, the empty sequence $\kappa = ()$ and the current attributes set $\mathbb{M}_\kappa = \mathbb{M}_{()} = \mathbb{A}$ and current tree $\mathsf{Tree} = \{\kappa\}$. We denote $\mathbb{V}_\kappa = \mathbb{B} \cup \mathbb{M}_\kappa$. We need further a *fuzzy membership* function $m : \mathbb{V}_\kappa \times \mathbb{K} \to [0, 1]$. The fuzzy clustering algorithm is succinctly described as Algorithm 1 below.

```
Data: $\kappa$, $\mathbb{M}_\kappa$, $\mathbf{r}$, objective function $F$
Result: Two sets $\mathbb{M}_{\kappa 1}$ and $\mathbb{M}_{\kappa 2}$ and the fuzzy membership $m(v,k)$ for $v \in \mathbb{V}_\kappa$ in the
        clusters $\mathbb{M}_{\kappa 1}$ and $\mathbb{M}_{\kappa 2}$
if $|\mathbb{M}_\kappa| > 1$ then
    assign $(v_1, v_2) \leftarrow \arg\max\{\|\mathbf{r}(v) - \mathbf{r}(v')\|, v, v' \in \mathbb{V}_\kappa\}$;
    assign $\mathbf{r}(v_1)$ and $\mathbf{r}(v_2)$ as centroids for the two candidate finer clusters $\mathbb{M}_{\kappa 1}$ and
    $\mathbb{M}_{\kappa 2}$;
    use standard 2-means fuzzy clustering algorithm to minimise objective function $F$
    under the constraint $\sum_{k=1}^{2} m(v,k) = 1$, for all $v \in \mathbb{M}_\kappa$ ;
    assign $\mathbb{M}_{\kappa 1} \leftarrow \{v \in \mathbb{M}_\kappa : m(v,1) > m(v,2)\}$;
    assign $\mathbb{M}_{\kappa 2} \leftarrow \mathbb{M}_\kappa \setminus \mathbb{M}_{\kappa 1}$;
end
```

Algorithm 1: FuzzyClustering

Note that in the previous construction $\mathbb{M}_{\kappa k} \subset \mathbb{M}_\kappa$ for every κ and every $k \in \mathbb{K}$. Therefore, the algorithm explores the branches of a tree from the root to the leaves. Denote by π_κ the orthogonal projection from $\mathscr{H}_\mathbb{A}$ to $\mathscr{H}_{\mathbb{M}_\kappa}$. The distillation step is described by the following Algorithm 2.

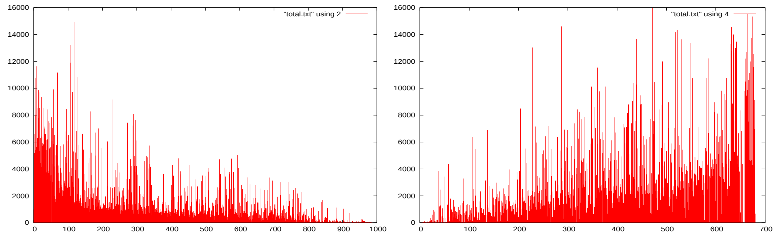
```
Data: FuzzyClustering
Result: Leaves and sequence of singleton sets $\mathbb{M}_\kappa$ for $\kappa \in$ Leaves
Initialisation{
$\kappa \leftarrow ()$;
$\mathbb{M}_\kappa \leftarrow \mathbb{A}$;
Leaf($\kappa$) $\leftarrow \mathbb{M}\kappa$;
Leaves $\leftarrow \{\}$;
Tree $\leftarrow \{\kappa\}$;
Bookkeeping $\leftarrow \{\kappa\}$;
}
while Bookkeeping $\neq \emptyset$ do
    for $\kappa \in$ Bookkeeping do
        if $|\mathbb{M}_\kappa| = 1$ then
            Leaves $\leftarrow$ Leaves $\cup \{\kappa\}$;
            Bookkeeping $\leftarrow$ Bookkeeping $\setminus \{\kappa\}$;
        else
            Use $\pi_\kappa$ to project from $\mathscr{H}_\mathbb{A}$ to $\mathscr{H}_{\mathbb{M}_\kappa}$;
            Thin the graph: $\mathbb{V}_\kappa \leftarrow \mathbb{B} \cup \mathbb{M}_\kappa$;
            Compute weighed Laplacian $\Lambda(\mathbb{V}_\kappa)$;
            Diagonalise $\Lambda(\mathbb{V}_\kappa)$;
            Compute $\nu$-dimensional representation $\mathbf{r}$;
            Call FuzzyClustering;
            for $k \in \mathbb{K}$ do
                $\kappa' \leftarrow \kappa k$;
                Leaf($\kappa'$) $\leftarrow \mathbb{M}_{\kappa'}$ /* $\mathbb{M}_\kappa$ as determined by FuzzyClustering */;
                Tree $\leftarrow$ Tree $\cup \{\kappa'\}$;
                Bookkeeping $\leftarrow$ Bookkeeping $\cup \{\kappa'\}$;
            end
        end
    end
end
```

Algorithm 2: Distillation

5 Illustration of the method, robustness and complexity issues

We tested the method on a dataset for an experiment on DNA array published in [35], with the set $\mathbb{A}$ of attributes corresponding to 12 cell lines (bone marrow, liver, heart, spleen, lung, kidney, skeletal muscle, spinal cord, thymus, brain, prostate, pancreas) and the set $\mathbb{B}$ of documents corresponding to 12000 human genes. To illustrate the method we present here only an example of the type of results we obtain by our method for the simplest case of one-dimensional representation of the graph. The complete lists of specificity degrees for the various genes (including their UniGene identifiers) for various dimensions are provided as supplemental material (at the home page of the first author).

Note that for one-dimensional representation, ordering by the magnitude of the eigenvector components is equivalent to a relabelling of genes. The figure 1 represents, within the previous mentioned relabelling, the levels of expressions for clustered genes. The same procedure has been applied for higher dimensional represen-tation of the graph (i.e. $\nu > 1$). These results are not presented here; they marginally improved some specifications and helped us removing apparent degeneracy in some cases. Finally, in the table 1, we give an example of the annotation provided by the database UniGene for the genes classified as specific of skeletal muscle cell line by our method.



liver

skeletal muscle

Fig. 1. For every singleton cluster, i.e. tissular context $\kappa \in \mathsf{Leaves}$ (we present solely the cases $\mathbb{M}_\kappa = \{$liver$\}$ and $\mathbb{M}_\kappa = \{$ skeletal muscle$\}$ in this example), the horizontal axis contains the set $\mathbb{B}$ of genes *relabelled* according to their decreasing (resp. increasing) fuzzy membership to $\mathbb{M}_\kappa$. Vertical axis represents the experimentally measured level of expression for those genes.

We observe that the majority of genes classified as most specific by our method are in fact annotated as specific in the database. To underline the power of our method, note that the UniGene annotation for the ATPase gene is "cardiac muscle". Our method determines it as most specific of "skeletal muscle". We checked the experimental data we worked on and realised that this gene is, as a matter of fact, 5 times more expressed in the skeletal muscle context than in the cardiac muscle. Therefore, our method correctly determines this gene as skeletal-muscle-specific.

In summarising, our method is an automatic and algorithmic method of analysis of raw experimental data; it can be used to any experiment of similar type *independently of any previous knowledge included in genomic databases* to provide biologists with a powerful tool of analysis. In particular, since most of the genes are not yet annotated in the existing databases, the method provides biologists with candidate genes for every particular context for further investigation. Moreover, the genetic character of documents and attributes is purely irrelevant; the same method

Table 1. Annotation of the genes closest (within the relabelling induced by $\mathbf{u}^2$) to the specificity witness “skeletal muscle”. Genes are separated by the – symbol.

ATPase, Ca++ transporting, cardiac muscle, fast twitch 1, calcium signaling pathway – Troponin I type 2; skeletal, fast – Myosin, light chain 1, alkali; skeletal, fast – Ryanodine receptor 1; skeletal; calcium signaling pathway – Fructose-1,6-bisphosphatase 2, glycolysis / gluconeogenesis – Actinin, alpha 3; focal adhesion – Adenosine monophosphate deaminase 1 (isoform M) purine metabolism – Troponin C type 2; fast; calcium signaling pathway – Carbonic anhydrase III, muscle specific; nitrogen metabolism – Nebulin – Troponin I type 1; skeletal, slow – Myosin, heavy chain 3, skeletal muscle – Myogenic factor 6, herculin – Myosin binding protein C, fast type – Calcium channel, voltage-dependent, beta 1 subunit – Metallothionein 1X – Bridging integrator 1 – Bridging integrator 1 – Calpain 3, (p94) – Tropomyosin 3 – Phosphorylase, glycogen; muscle (McArdle syndrome, glycogen storage disease type V); starch and sucrose metabolism – Myozenin 3 – Myosin binding protein C, slow type – Troponin T type 3; skeletal, fast – Superoxide dismutase 2; mitochondrial – Nicotinamide N-methyltransferase – Sarcolipin – Interleukin 32 – Sodium channel, voltage-gated, type IV, alpha subunit – Guanidinoacetate N-methyltransferase; urea cycle and metabolism of amino groups.

can be used to any other dataset of similar structure, let them concern linguistic, genetic, or image data.

Concerning the algorithmic complexity of the method, the dominant contribution comes from the diagonalisation of a $|\mathbb{B}| \times |\mathbb{B}|$ dense real symmetric matrix, requiring at worst $\mathscr{O}(|\mathbb{B}|^3)$ time steps and $\mathscr{O}(|\mathbb{B}|^2)$ space. The time complexity can be slightly reduced, if only low-dimensional (dimension v) representations are sought, to $\mathscr{O}(v \times |\mathbb{B}|^2)$ time steps. Moreover, we tested the method against additive or multiplicative random perturbations of the experimental data; it proved astonishingly robust.

6 Connections to previous work

The algorithm of semantic distillation maps the dataset into a graph and uses spectral methods and fuzzy clustering to analyse the graph properties. As such, this algorithm is inspired by various pre-existing algorithms and borrows several elements from them.

The oldest implicit use of a vector space structure to represent dataset and application of spectral methods to analyse them is certainly “principal components analysis” introduced in [25]. The method seeks finding directions of maximal variability in the space corresponding to linear combinations of the underlying vectors. The major drawbacks of principal components analysis are the assumptions that dataset matrix is composed of row vectors that are independent and identically distributed realisations of the same random vector (hence the covariance matrix whose principal components are sought can be approximated by the empirical covariance of the process) and that there exists a linear transformation maximising the variability.

Vector space representations and singular value decomposition, as reviewed in [5], have been used to retrieve information from digital libraries. Implementations of these ideas range from the famous PageRank algoritm used by Google (see [18] and [17] for expository reviews) to whole genome analysis based on latent semantic indexing [23, 16].

From the information contained in the dataset X, a weighed graph of interactions among documents is constructed. To palliate the weaknesses of principal component analysis, reproducing kernel methods can be used. The oldest account of these methods seems to be [21] and their formulation in the context of Hilbert spaces can be found in [1]. In [31], analysis of features of a microarray experiment is proposed based on kernel estimates on a graph. Note however that in that paper, the graph incorporates extrinsic information coming from participation of genes in specific pathways as documented in the KEGG database. On the contrary, in the method we are proposing here, the graph can be constructed in an intrinsic way, even in the absence of any additional information from existing databases. In [4, 9, 24], kernel methods and Laplace eigenspace decomposition are used to generalise principal components analysis to include non-linear interactions among genes. Particular types of kernels, defined in terms of commuting times for a random walk on the graph are used in [13, 20, 30]. All these methods, although not always explicitly stated in these articles, are as a matter of fact very closely related since the kernels, the weighed graph Laplacian and the simple random walk on the graph can be described in a unified formalism [7, 8, 10, 15, 22]. It is worth noting that analysis of Laplacian of the graph is used in many different contexts, ranging from biological applications (proteins conformation [32], gene arrays [23]) to web search [3] or image analysis [28].

Fuzzy clustering has been introduced in [6]; lately it was shown [33] equivalent to probabilistic clustering if the objective function is expressed in terms of the Rényi entropy.

The idea of describing the data in terms of abstract Hilbert spaces has been used (in the context of database search) in [2, 12, 14, 29, 34].

The semantic distillation algorithm is based on a quantum-inspired subspace projection, strongly reminiscent of the quantum procedure of measurement. Although fully implemented on classical computers, it shares with general quantum algorithms features of non-distributive quantum logic [26, 27]. The semantic approach of Quantum Mechanics can be found in [27, 11]. It is worth underlying that the full fledged fuzzy logic induced by quantum semantics is not equivalent to the standard fuzzy logic introduced in [36]; it represents a genuine extension of it [11].

7 Perspectives

Various data sets (not only biological) are presently semantically distilled and the method compared with more traditional approaches. Preliminary results obtained so far seem to confirm the power of the method.

Several directions are in progress:

- Although the method is quantum-inspired, the fuzzy logic induced is still standard fuzzy logic. We are currently working on the extension to generalised fuzzy logic induced by full-fledged quantum semantics.

- The graph analysis we performed provided us with degrees of specificities of every gene in a particular context. These data can be reincorporated to the graph as internal degrees of freedom of a multi-layered graph that can be further analysed.
- The connections of the algorithm of semantic distillation with the algorithm of purification of quantum states [19] introduced in the context of quantum computing are currently explored.

References

1. N. Aronszajn. Theory of reproducing kernels. *Trans. Amer. Math. Soc.*, 68:337–404, 1950.
2. Ricardo Baeza-Yates. Information retrieval in the web: beyond current search engines. *Internat. J. Approx. Reason.*, 34(2-3):97–104, 2003. Soft computing applications to intelligent information retrieval on the Internet (Mérida/Granada, 2002).
3. Pierre Baldi, Paolo Frasconi, and Padhraic Smyth. *Modeling the Internet and the Web: Probabilistic Methods and Algorithms*. Wiley InterScience, New York, 2003.
4. Mikhail Belkin and Partha Niyogi. Towards a theoretical foundation for Laplacian-based manifold methods. In *Learning theory*, volume 3559 of *Lecture Notes in Comput. Sci.*, pages 486–500. Springer, Berlin, 2005.
5. Michael W. Berry, Zlatko Drmač, and Elizabeth R. Jessup. Matrices, vector spaces, and information retrieval. *SIAM Rev.*, 41(2):335–362 (electronic), 1999.
6. James C. Bezdek. *Pattern recognition with fuzzy objective function algorithms*. Plenum Press, New York, 1981. With a foreword by L. A. Zadeh, Advanced Applications in Pattern Recognition.
7. Massimo Campanino and Dimitri Petritis. On the physical relevance of random walks: an example of random walks on a randomly oriented lattice. In *Random walks and geometry*, pages 393–411. Walter de Gruyter GmbH & Co. KG, Berlin, 2004.
8. Fan R. K. Chung. *Spectral graph theory*, volume 92 of *CBMS Regional Conference Series in Mathematics*. Published for the Conference Board of the Mathematical Sciences, Washington, DC, 1997.
9. Ronald R. Coifman and Stéphane Lafon. Diffusion maps. *Appl. Comput. Harmon. Anal.*, 21(1):5–30, 2006.
10. Dragoš M. Cvetković, Michael Doob, and Horst Sachs. *Spectra of graphs*. Johann Ambrosius Barth, Heidelberg, third edition, 1995. Theory and applications.
11. Maria Luisa Dalla Chiara, Roberto Giuntini, and Roberto Leporini. Compositional and holistic quantum computational semantics. *Natural Computing*, 6(5):113–132, 2007.
12. Sándor Dominich. *Mathematical foundations of information retrieval*, volume 12 of *Mathematical Modelling: Theory and Applications*. Kluwer Academic Publishers, Dordrecht, 2001.
13. Francois Fouss and Jean-Michel Renders. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 19(3):355–369, 2007. Member-Alain Pirotte and Member-Marco Saerens.
14. Peter Gärdenfors. Induction, conceptual spaces and AI. *Philos. Sci.*, 57(1):78–95, 1990.
15. Chris Godsil and Gordon Royle. *Algebraic graph theory*, volume 207 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 2001.
16. Ramin Homayouni, Kevin Heinrich, Lai Wei, and Michael W. Berry. Gene clustering by latent semantic indexing of medline abstracts. *Bioinformatics*, 21(1):104–115, 2005.

Robustness and efficiency of Semantic Distillation to retrieve genes by their contextual specificity

Thomas Sierocinski
Institut de recherche mathématique de Rennes
Université de Rennes 1
Campus de Beaulieu
35042 Rennes Cedex, France
thomas.sierocinski@univ-rennes1.fr

May 30, 2008

Abstract

Since knowledge on human genes functions is actually incomplete and exhaustive analysis cannot be done in a reasonable amount time, efficient interpretation of DNA microarrays experimental measurements without a priori knowledge can be a precious tool for experimental biologists. We introduced in [16] an algorithm called semantic distillation (inspired from the theory of quantum measurement) to cluster genes by their contextual specificity (biological context) from raw genechip experimental measurements. The method can be divided into three steps: raw experimental data representation in the space of concepts, construction and dimensional reduction of a weighed graph, and, finally fuzzy semantic clustering and distillation. Here we study DNA arrays experiments and efficiently clustering the genes of the array according to their tissular specificity.
All supplemental material is available at **mettre une adresse**

1 Introduction

Microarrays provide a powerful basis to monitor the expression of thousands of genes in several biological samples [11]. Interpretation of such massive experimental results requires the use of powerful and effective methods to extract relevant information on genes of interest. For this purpose, several data-mining tools have been developped using different clustering approaches like hierarchical clustering [6], principal component analysis [10, 20], *k*-means [9, 17], singular value decomposition [8, 1]. In general, clustering is a division of a dataset into groups of similar items. In the case of DNA microarrays analysis, the grouping of genes is made in terms of the expression profile similarity: genes with similar expression in different situations are identified and clustered accordingly. Clustering coexpressed genes into biologically meaningful groups can help in discovering regulatory motifs from microarray data. Clustering also helps in inferring the biological role of an unknow gene that is coexpressed with a known gene. An inherent problem with the commonly used clustering algorithmis that they force every data point into a cluster. In the case of microarray data, a considerable number of genes do not contribute to the biological process being studied (and hence lack expression with other genes). Inclusion of such noisy genes in any cluster causes contamination, making it less suitable for further analysis. The contribution of this paper is twofold: first, we recall to the reader a novel algorithmic method we have introduced in [16] called Semantic Distillation (SD) to analyse DNA micorarray experimental measures. In a second step we will proceed with the statistical analysis of results obtained with our method.

SD provide the user with an ordered list of genes for every biological context allowing, for example, to prune a liste of candidate genes for further investigations or to detect signatures in genes expression for a particular biological sample of the experiment. SD does not relies on any prior knowledge such as database annotations or learning data, it rationale is to help analysing raw experimental data even in the absence of any previous knowledge. As principal components analysis or singular value decomposition, our method is principaly based on two points: the choice of a distance (or semantics) associated with the dataset and spectral properties of the graph (or matrix) constructed from this distance. The main difference with the methods developed so far relies ot the last step of the method named “ fuzzy semantic clustering and distillation” (detailed in the eponymous section).

2 Approach

The method presently used is fully formalised and detailed in [16]. Let $\mathbb{B}$ be a finite set of *genes* and $\mathbb{A}$ a finite set of *cellular contexts* (or biological samples). The dataset is a $|\mathbb{B}| \times |\mathbb{A}|$ matrix $\mathsf{X} = (x_{ba})_{b\in\mathbb{B},a\in\mathbb{A}}$ of real or complex elements, where $|\cdot|$ represents cardinality. It can be represented by a collection of $|\mathbb{B}|$ row vectors $\mathbf{x}_b = (x_{b1},\ldots,x_{b|\mathbb{A}|}), b\in\mathbb{B}$ of $\mathbb{R}^{|\mathbb{A}|}$. The matrix elements x_{ba} are real numbers encoding luminescence intensities of DNA array ultimately representing the level of expression of gene b in biological sample a.

Experimental information hold on the objects of the system undergoes a sequence of processing steps; each step is performed on a different representation of the information. Those different representation spaces and the corresponding information processing act as successive filters revealing at the end the most pertinent and significant part of the information.

2.1 Data representation in the space of concepts

For $\mathbb{A}$ and $\mathbb{B}$ as in the previous subsection, we define the *space of concepts*, $\mathscr{H}_{\mathbb{A}}$, as the real free vector space over $\mathbb{A}$. The space $\mathscr{H}_{\mathbb{A}}$ is equipped with the usual scalar product so that turns it into a finite dimensional Hilbert space. Therefore, the complete dataset X can be represented as the collection of $|\mathbb{B}|$ vectors $|\,\Xi_b\,\rangle$ decomposed on the orthonormal basis of $\mathscr{H}_{\mathbb{A}}$.

$$|\,\Xi_b\,\rangle = \sum_{a\in\mathbb{A}} x_{ba}|\,a\,\rangle \in \mathscr{H}_{\mathbb{A}}$$

with $b\in\mathbb{B}$ and where $|\,a\,\rangle$ represents the element of the orthonormal basis of the free vector space corresponding to the attribute a. We also introduce rays (by dividing vectors by their norm) on the space. We use the symbol $|\,\xi_b\,\rangle$ to denote the ray associated with vector $|\,\Xi_b\,\rangle$.

$$|\,\xi_b\,\rangle = \frac{|\,\Xi_b\,\rangle}{\||\,\Xi_b\,\rangle\|}$$

The Hilbert space structure on $\mathscr{H}_{\mathbb{A}}$ allows a natural geometrisation of the space of documents by equipping it with a pseudo-distance d.

$$d : \mathbb{B} \times \mathbb{B} \to \mathbb{R}_+$$

defined by

$$d(b, b') = \||\,\xi_b\,\rangle - |\,\xi_{b'}\,\rangle\|$$

The Hilbert space description incorporates into a unified algebraic framework all logical and probabilistic information hold by the dataset.

2.2 Construction and dimensional reduction of a weighed graph

In the sequel, we introduce the set $\mathbb{V}_{\mathbb{A}} = \mathbb{A} \cup \mathbb{B}$. Thus, for any $v \in \mathbb{V}_{\mathbb{A}}$,

$$|\,\Xi_v\,\rangle = \begin{cases} |\,a\,\rangle & \text{if } v = a \in \mathbb{A}, \\ \sum_{a \in \mathbb{A}} x_{ba} |\,a\,\rangle & \text{if } v = b \in \mathbb{B}. \end{cases}$$

The new vectors $|\,\Xi_a\,\rangle = |\,a\,\rangle$ are included as *specificity witnesses* in the dataset. Note that since these new vectors are also elements of the same Hilbert space, the pseudo-distance d naturally extends to $\mathbb{V}_{\mathbb{A}}$. Suppose now that a *similarity function* σ based on pseudodistance d (described in the previous section).

Many similarity (or dissimilarity) functions are described in the literature [7] (Pearson sample correlation distance, cosine correlation distance, Spearman sample correlation distance, Kendall's correlation distance...). Each one of these distances expresses different relationships between objects, resulting semantics (the implied meaning of data, used to define the significance of entities and their role within the system) is therefore affected. Cosine distance was used in all experiments described in this paper. This technique, by privileging directions instead of positions in space, avoid giving too much semantic significance to highly expressed genes.

A weighed graph is now constructed with vertex set $\mathbb{V}_{\mathbb{A}}$. Weights are assigned to the edges of the complete graph over $\mathbb{V}_{\mathbb{A}}$. The weights are expressible in terms of the similarity function σ and constitute a weight matrix $W_{vv'}$ representing the graph.

$$W_{vv'} = \sigma(v, v').$$

We seek to reveal most pertinents interactions among genes. This is achieved by following standard techniques of reduction of the data dimensionality by optimal representation of the weighed graph in low dimensional Euclidian spaces spanned by eigenvectors of the Laplacian. Such methods have been used by several authors [3, 13].

2.3 Fuzzy semantic clustering and distillation

The following step consists in a fuzzy divisive clustering of objects among two subsets by exploiting the previous low-dimensional representation. This procedure assigns a fuzzy membership to each object relative to characters of the two subsets. Fuzziness is as a matter of fact a distinctive property of experimental biological data reflecting our incomplete knowledge of fundamental biological processes.

The algorithm of SD is a recursive divisive fuzzy clustering followed by a projection on a Hilbert subspace and a thinning of the graph. It starts with the Hilbert space $\mathscr{H}_{\mathbb{A}}$ and the graph with vertex set $\mathbb{V}_{\mathbb{A}}$ and constructs recursively a sequence of Hilbert subspaces and subgraphs. We need further a *fuzzy membership* function m that quantifies membership to one of the two subsets.

$$m : \mathbb{V}_{\kappa} \times \mathbb{K} \to [0,1].$$

with $\mathbb{K}$ the set of Hilbert subspaces $\mathbb{K} = \{K_1, K_2\}$ The fuzzy clustering algorithm is described in [4].

Up to this step, our method is a sequence of known algorithms that have been previously used separately in the literature in various contexts. The novelty of our method relies on the following steps. The previous fuzzy clustering reduced the indeterminacy of the system. This information is fed back to the system to perform a projection to a proper Hilbert subspace. In that way, the information content of the dataset is modified by the information gained by the previous observations. After this feeding back, the three previous steps are repeated but now referring to a Hilbert spaces of lower dimension. Therefore our method is not a mere fuzzy clustering algorithm but a genuine non-classical interaction information retrieval procedure where previous observations alter the informational content of the system (reminiscent of the measurement procedure in Quantum Mechanics [12]). Fuzzy membership is computed for every elements of $\mathbb{V}_{\mathbb{A}}$. Elements of $\mathbb{A}$ are assigned to one of the two Hilbert subspaces according to their membership. Elements of $\mathbb{B}$ are projected in both Hilbert spaces. The method starts afresh step one until all objects from $\mathbb{A}$ are clustered in different subsets.

3 Dataset

We tested the method on the dataset obtained by an experiment on DNA array published in [19] available on Gene Expression Omnibus database [18] under accession number GSE803. This dataset is composed of a set of 62837 probesets for a collection of 12 major human tissues including bone marrow, brain, heart, kidney, liver, lung, pancreas, prostate, skeletal muscle, spinal cord, spleen and thymus. The 62837 probesets consisted in five genechips arrays (HG-U95Av2, HG-U95B, HG-U95C, HG-U95D, HG-U95E). The HG-U95Av2 represents 12613 full length genes, while arrays B through E represent approximately 50000 EST clusters. For this study we choose only measures from the HG-U95Av2 array.

4 Results

We present here the simplest case of a one dimensional representation of the graph (higher dimensional representation are actually under consideration) for one tissular context (skeletal muscle) of the twelve tissues of the study. Complementary result files are provided as supplemental material.

Fig. 1 represents the type of results we obtain: for every singleton cluster (tissular context), the horizontal axis represents the set of genes relabelled according to their decreasing fuzzy membership to the cellular context. Vertical axis represents the experimentally measured level of expression for those genes in this particular context. For each cluster we can observe a specificity gradient and also a “global expression

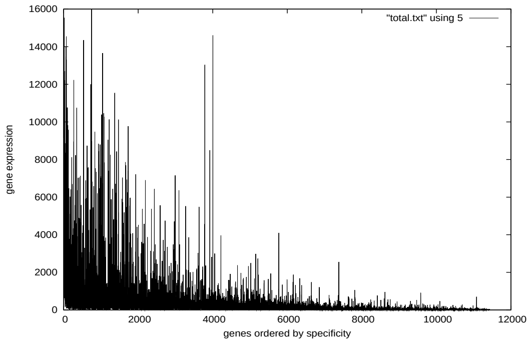


Figure 1: Ranking of GSE803 dataset genes. The horizontal axis contains the set of genes relabelled according to their decreasing fuzzy membership to the skeletal muscle cellular context. Vertical axis represents the experimentally measured level of expression for those genes.

gradient". Nevertheless, we note that genes with high expression level are not necessarily specific. In fact, if a gene is very expressed in several tissular contexts, it will not be, by definition, uni-tissular specific (respectively a weakly-expressed gene can be nevertheless context specific).

In Table 1, we give an example of annotation provided by the UniGene database [18, 15] for the first skeletal muscle most specific genes retrieved by our method. We observe that the majority of those genes are effectively annotated as skeletal muscle specific. However we can underline that the first one (ATPase) is annotated as cardiac muscle specific. We checked the experimental data we worked on and bibliography [14, 2], and realised that this gene is, as matter of fact, five times more expressed in the skeletal muscle context than in the cardiac muscle, and is also described as skeletal muscle specific [2]; mutations of this gene leads to Brody disease (affecting skeletal muscle) [14]. Therefore, our method correctly determines this genes as skeletal muscle specific.

5 Results analysis

The method previously described allows extraction of the information contained in experimental results from DNA chips. In the context of information retrieval, the two desired properties that have been accepted by research community for measurment of search effectiveness are recall (the proportion of relevant documents retrieved by the system) and precision (the proportion of retrieved documents that are relevant).

Measuring recall and precision requires a good knowledge of the dataset on which the analysis was made. For DNA microarrays, a part of the information required for these measures is missing (genes whose functions are unknown, missing annotations…), it is unclear whether the classification of a gene is actually relevant to a biological sample or not. To enable these measures, we must define what is a specific gene for a given biological sample.

Broad sense specificity: We consider here only values from the initial dataset. A gene is considered here to be specific to a biological sample if the value of its expression is higher in this particular sample than in the others. According to this definition, we can measure the number of specific genes for each sample of the GSE803 dataset

Table 1: Annotation of the genes closest to the specificity witness “skeletal muscle”.

Ranking	Name	Annotation
1	ATPase	Ca++ transporting, cardiac muscle, fast twitch 1, calcium signaling pathway
2	Troponin I type 2	skeletal, fast
3	Myosin, light chain 1	alkali; skeletal, fast
4	Ryanodine receptor 1	skeletal; calcium signaling pathway
5	Fructose-1,6-bisphosphatase 2	glycolysis gluconeogenesis
6	Actinin, alpha 3	focal adhesion
7	Troponin C type 2	fast; calcium signaling pathway
8	Carbonic anhydrase III	muscle specific; nitrogen metabolism
9	Nebulin	
10	Troponin I type 1	skeletal, slow
11	Myosin, heavy chain 3	skeletal muscle
12	Myogenic factor 6, herculin	

Table 2: GSE803 genes distribution in the case of broad sense specificity

cellular context	number of “broad sense specific” genes
bone marrow	749
brain	1796
heart	866
kidney	1050
liver	488
lung	919
pancreas	545
prostate	376
skeletal muscle	951
spinal cord	2503
spleen	212
thymus	2158
total	12613

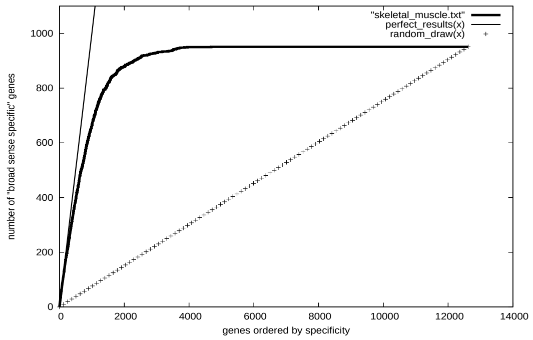


Figure 2: Ranking evaluation in the case of skeletal muscle context. Bold curve is obtained by applying (1) on our results. Solid line represents perfect results according to a line of slope 1 (mind the difference of scales of horizontal and vertical axes), crossed line represents results from a random draw of genes.

(reported in table 2). Remind that this definition express specificity for a particular DNA chip experiment, genes considered here as sample specific (i.e. muscle specific) may not be defined as sample specific in another experiment.

Annotation specificity: The role of this specificity is to enable an evaluation of our method by confrontation with an external knowledge source. To achieve this, we checked annotation in SOURCE [5]. We define "annotation specificity" as follows: a gene is specific to a particular context if its *normalised expression* is higher in this context than in the other contexts of the experiment.

Both definitions will allow us to describe the relevance of the classification of genes for a biological sample.

It is well accepted that good information retrieval systems should retrieve as many relevant documents as possible (have a high recall), and it should retrieve very few non relevant documents (have a high precision). Unfortunately these two goals have proven to be quite contradictory over the years. Techniques that tend to improve recall tend to hurt precision and vice-versa. Both recall and precision are set oriented measures and have no notion of rank retrieval. Researchers have used several variants of recall and precision to evaluate ranked retrieval.

In our case, we define the following measure:

$$O(i) = |\mathsf{Spec} \bigcap \mathsf{Ret}_i| \tag{1}$$

Where Ret_i is the subset of Ret containing the first *i* most specific genes retrieved by our method and Spec the set of specific genes (broad or annotation) to a particular biological sample.In fact, this measure is the common root of recall and precision. By dividing it by the number of genes we get the precision value, by dividing it by the total number of specific genes we get the recall value.

Fig. 2 illustrates (1) for broad sense specificity in the case of skeletal muscle cellular context. For this evaluation function, a perfect result would be represented by a slope line of 1 and results for a random draw are represented by the crossed line. We can observe that for the first genes retrieved by our method the curve has a slope very close to 1 and keeps values nearby until all specific genes are retrieved.

As an example, figure 2 shows recall and precision for the same sample. We can observe that recall follows closely the curve of $O(i)$. In the case of precision, we can

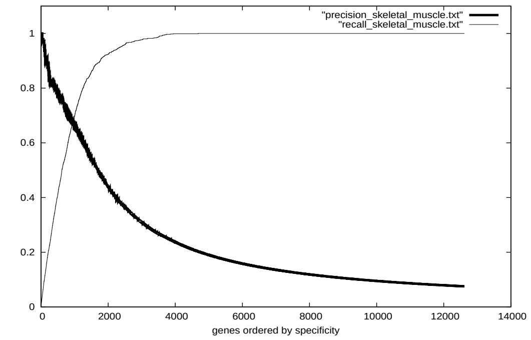


Figure 3: Precision and recall in the case of skeletal muscle sample. Bold curve denotes precision given for a cut-off rank (horizontal axis), thin line denotes recall.

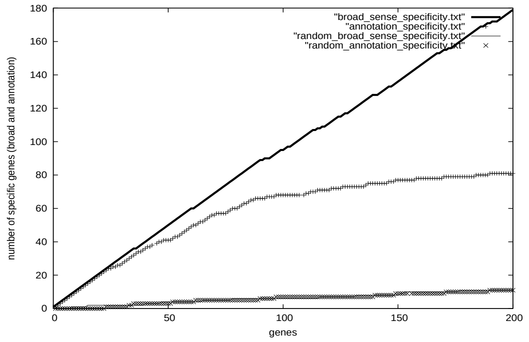


Figure 4: Evaluation of our results for the first 200 genes in the case of skeletal muscle context. Bold curve is obtained by applying (1), crossed $(+)$ line is obtained with respect to annotation specificity, thin and crossed $(\times)$ lines represents respectively annotation and broad sense specificity in the case of a random draw of genes in the dataset.

observe that its value is very close to 1 for the first retrived genes and then decreases while the number of retrieved genes increases, reflecting the inclusion of false positive in the set Ret_i.

The second stage of our analysis is to draw $O(i)$ for the annotation specificity (previously defined). Figure 4 shows the results for the first 200 genes retrieved by our method (both specificities were traced, broad sense with the bold line, annotation with the crossed line (+)). We observe that there are only few differences for the first 30 genes between the two curves. However, this observation is not true for genes ranked after 30; Although the slope of broad sense specificity curve remains close to 1, the slope of annotation specificity curve quickly decreases to 0.

In a first time, we can assume that the definition of broad sense specificity is too general explaining the large number of genes considered as specific in the dataset. However, the two definitions are, in fact, very similar; we determine a gene as specific exclusively with respect to the other biological samples of the microarray experiment, which should therorically give us the same number of specific genes in both cases. As an additional test, we conducted a random sampling from the list of genes and measured $O(i)$ for two specificities (also shown in Figure 4: thin line for broad sense, crossed line (x) for annotation). It turns out that these two specificities coïncide so that

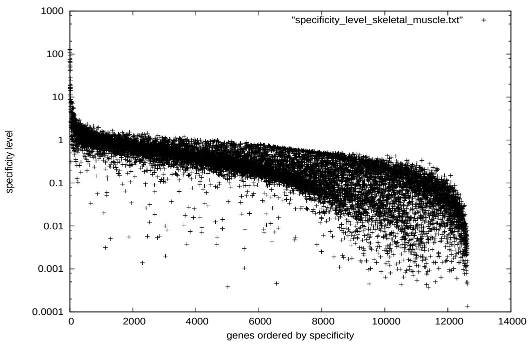


Figure 5: Specificity level in the case of skeletal muscle context. Horizontal axis represents genes ordered by decreasing membership to the cellular context. Vertical axis denotes (2) for those genes.

their curves coïncide. With a simple calculation we can estimate the number of specific genes in the sample:

$$S = \frac{|\mathsf{Spec}||\mathbb{B}|}{|R|}$$

Where R is the random sample of genes. In both cases, we have 11 specific genes for 200 picked randomly which allows us to estimate 694 specific genes in the dataset. This value is significantly closer to the number of broad sense specific genes than to the number of annotation specific genes. We believe that this difference in the results derives from the normalised gene expression computation method from SOURCE. Indeed, it is based on the relative frequencies of genes in Unigene clusters[1] and not on raw gene expression measures. However, regardless of the specificity defined, our method is able to characterize specific genes to biological samples and effectively retrieve them from the data set.

Now that we have shown our method efficiency to retrieve specific genes, we can ask about the relevance of genes ranking: Are genes effectevely ranked by decreasing specificity? Based on broad sense specificity definition we can "quantify" specificity by comparing gene expression in a context with the highest expression in the complementary set of cellular contexts. We define the specificity level of a gene b in context a by:

$$L_{ba} = \frac{x_{ba}}{\max\{x_{ba'}, a' \in \mathbb{A}, a' \neq a\}} \tag{2}$$

If $L_{ba} \geq 1$ then gene b is specific in context a and the higher this value, the more specific is gene b in context a. Inversely, if $L_{ba} < 1$ then b is not specific in a. Fig. 5 shows (in logarithmic scale) this measure in the case of skeletal muscle cellular context. We note that genes determined as most specific by our method are those with highest ratio (on the left of Fig. 5), respectively the less specific are those with the lower ratio (on the left of Fig. 5).

[1] The computation method is fully detailed at http://smd.stanford.edu/help/SOURCE/normalization.html

Table 3: Robustness measures for perturbed datasets

λ	M	δ_{max}	δ_{ave}
0	0	0	0.000000
0.1	20	3	1.500000
0.2	39	6	1.846154
0.3	37	10	3.027027
0.4	43	12	3.953488
0.5	42	11	3.666667
1	44	40	10.50000
multiplicative	43	31	6.976744

6 Complexity and robustness

Concerning the algorithmic complexity of the method, the dominant contribution comes from the diagonalisation of a $|\mathbb{B}| \times |\mathbb{B}|$ dense real symmetric matrix, requiring at worst $\mathscr{O}(|\mathbb{B}|^3)$ time steps and $\mathscr{O}(|\mathbb{B}|^2)$ space. The time complexity can be slightly reduced, if only low-dimensional representations are sought, to $\mathscr{O}(\nu \times |\mathbb{B}|^2)$ time steps.

We also tested robustness of SD by adding noise to the experimental data. For this purpose, two methods were chosen: additive and multiplicative random perturbations of the dataset. Let X be the original real $|\mathbb{B}| \times |\mathbb{A}|$ matrix with $x_{ba} \in \mathbb{R}$, R a real $|\mathbb{B}| \times |\mathbb{A}|$ matrix with r_{ba} a random variable $\in [0,1]$, $\lambda \in \mathbb{R}$ a perturbation coefficient and X' a real $|\mathbb{B}| \times |\mathbb{A}|$ matrix containing perturbed data. We define additive perturbation by:

$$x'_{ba} = x_{ba}(1 + \lambda(2r_{ba} - 1)) \tag{3}$$

Here λ is an adjustable parameter and r_{ba} are independent (for different (a,b)) random variables uniformly distributed over $[0,1]$. Therefore x'_{ba} are random variables uniformly distributed over $[(1-\lambda)x_{ba}, (1+\lambda)x_{ba}]$. Obviously $\mathbb{E}(x'_{ba}) = x_{ba}$ where $\mathbb{E}(\cdot)$ denotes expectation. By applying it, we can add noise to the dataset proportionally to a coefficient and to original expression values x_{ba}. Similarly we define multiplicative perturbation by:

$$x'_{ba} = x_{ba}(r_{ba} + 1/2) \tag{4}$$

satisfying again $\mathbb{E}(x'_{ba}) = x_{ba}$. There is no perturbation coefficient in (4).

To measure robustness we picked up randomly 50 genes from the GSE803 dataset and their associated measures for the 12 cellular contexts. We perturbed this matrix with both methods and various λ and ran SD. Ranking lists are provided as supplemental material.

Random perturbations on X induce bad ranking on certain genes. Denoting by $p(b)$ the rank of gene b when using X and $p'(b)$ its ranking when using X', we say that there is a misplacement of b under perturbation if $p(b) \neq p'(b)$. The total number M of misplacements is a figure of merit for the method. As expected, it increases when the strength of the perturbation λ increases, as shown in the table 3. The two last columns contain the maximal displacement

$$\delta_{max} = \max_{b \in \mathbb{B}} |p(b) - p(b')|$$

and the average displacement

$$\delta_{ave} = \frac{1}{M}\Sigma_{b\in\mathbb{B}}|p(b)-p(b')|.$$

We can observe that the more we perturb the dataset, the more we put noise in its information content which has a direct impact on ranking quality. Yet, results given by SD are still pertinent in the case of additive perturbation for low λ e.g. for $\lambda = 0.1$ there is only few direct permutation in ranking of genes.

7 Discussion

In this paper we have presented a new information retrieval method called Semantic Distillation (SD). This method could prove very usefull when interested in tissue specific genes signatures and/or relating gene deletion phenotypes to specific deseases/ tissues / cellular states without previous knowledge. We have previously shown that this method is effective and robust against noise. Moreover, it is a versatile method that can be applied to any data set with the same structure.

7.1 Limitations

Although distillation is a very efficient method to assign membership to a gene for several cellular contexts (biological samples), reducing the space of concepts at each iteration may affect efficiency. It slightly decreases when the number of cellular contexts is small i.e $|\mathbb{B}| = 2$ (in this case $O(i)$ keeps good values for the first retrieved genes but decreases faster than in the case of many tissular contexts). To avoid this problem, we are actually working on a weighed feed-back of information into the system.

We have presented here the simplest case of a one dimensional representation of the weighed graph. In fact, it should be the less robust low-dimensional representation. Perturbation of the dataset has consequences on the graph structure, the ν-dimensional representation is then also affected, by representing the graph in a one-dimensional space we reduce self-averaging (noise absorption) capacities of the system: for example, a bi-dimensional representation should rank genes correctly where a one-dimensional shows permutations in genes order.

A third limitation is that $|\mathbb{B}|$ (the number of genes) has to be greater than $|\mathbb{A}|$ (number of biological samples). This is induced by the construction of the space of concepts: few vectors represented in a very high-dimensional Hilbert space have a high probability of living in orthogonal subspaces which means that similarity functions should be zero in this case. In fact, we do not encounter this phenomenon when ranking genes but this remark take all its meaning when conducting a study on ranking tissues with respect to gene expression (in other word on the transposed matrix of X).

Finally, SD complexity does not allow computation on standard desktop computers for very large sets of genes (tenths of thousands genes). For example in the case of GSE803 data diagonalisation step needs about 600 Mo of memory aqnd approximately 30 hours of cpu time to be achieved.

7.2 Perspectives

Our method gives to all the genes a membership value for each biological sample of the analysis. In the microarray field, commonly used approaches are mainly based on ”crisp clustering” algorithms, making comparison between our results and those of currently used methods difficult. However, various data sets (not only biological) are presently semantically distilled and the method compared with more traditional approaches. Results obtained so far tend to confirm the power of the method.

Several directions are in progress: first, although the method is quantum-inspired, the fuzzy logic induced is still standard fuzzy logic. We are currently working on the extension to generalised fuzzy logic induced by full-fledged quantum semantics. Secondly, the graph analysis we performed provided us with degrees of specificities of every gene in a particular context. These data can be reincorporated to the graph as internal degrees of freedom of a multi-layered graph that can be further analysed. The connections of the algorithm of SD with the algorithm of purification of quantum states [12] introduced in the context of quantum computing are currently explored. Finally, requests on the datasets are actually limited to context specificity, quantum logical operators (NOT, AND, OR) are under test.

8 Conclusion

We presented a novel approach for clustering genes and a measure of its efficiency. The goal is to correctly characterise specificity of a set of genes to one (or several) biological contexts from raw experimental DNA microarrays data. The method had been tested on a dataset composed of 12613 genes expression measures in 12 different tissular contexts. Our results indicate that not only our method correctly classifies genes with respect to experimental data and databases annotations, but also, can remove ambiguity for some genes annotations. Additionally, SD proves robust.Moreover, SD is a very versatile method that can be applied to any dataset with the same structure.

References

[1] O. Alter, P. O. Brown, and D. Botstein. Singular value decomposition for genome-wide expression data processing and modeling. *Proc Natl Acad Sci U S A*, 97(18):10101–10106, Aug 2000.

[2] F. Baba-Aissa, L. Raeymaekers, F. Wuytack, L. Dode, and R. Casteels. Distribution and isoform diversity of the organellar ca2+ pumps in the brain. *Mol Chem Neuropathol*, 33(3):199–208, Apr 1998.

[3] Mikhail Belkin and Partha Niyogi. Towards a theoretical foundation for Laplacian-based manifold methods. In *Learning theory*, volume 3559 of *Lecture Notes in Comput. Sci.*, pages 486–500, Berlin, 2005. Springer.

[4] K. F. Chung, S. T. Wang, H. B. Shen, and R. Q. Zhu. Note on the relationship between probabilistic and fuzzy clustering. *Soft Comput.*, 8(7):523–526, 2004.

[5] Maximilian Diehn, Gavin Sherlock, Gail Binkley, Heng Jin, John C Matese, Tina Hernandez-Boussard, Christian A Rees, J. Michael Cherry, David Botstein, Patrick O Brown, and Ash A Alizadeh. Source: a unified genomic resource of functional annotations, ontologies, and gene expression data. *Nucleic Acids Res*, 31(1):219–223, Jan 2003.

[6] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proc Natl Acad Sci U S A*, 95(25):14863–14868, Dec 1998.

[7] R. Gentleman, V. Carey, W. Huber, R. Irizarry, and S. Dudoit. *Bioinformatics and Computational Biology Solutions Using R and Bioconductor*. Springer Series in Statistics for Biology and Health, 2005.

[8] G. H. Golub and C. F. Van Loan. *Matrix computation*. The Johns Hopkins University Press, 1996.

[9] Hartigan and Wong. Algorithm as136: A k-means clustering algorithm. *Applied statistics*, 28:100–108, 1979.

[10] I. T. Joljiffe. *Principal component analysis*. Springer, 1986.

[11] E. S. Lander. Array of hope. *Nat Genet*, 21(1 Suppl):3–4, Jan 1999.

[12] Maassen and Kümmerer. Purification of quantum trajectories. *Dynamics and stochastics*, vol 48 of IMS Lecture Notes Monogr. Ser.:252–261, 2006.

[13] J. Nilsson. *Nonlinear dimensionality reduction of gene expression data*. PhD thesis, Lund University, Faculty of Engineering, 2006.

[14] A. Odermatt, K. Barton, V. K. Khanna, J. Mathieu, D. Escolar, T. Kuntzer, G. Karpati, and D. H. MacLennan. The mutation of pro789 to leu reduces the activity of the fast-twitch skeletal muscle sarco(endo)plasmic reticulum ca2+ atpase (serca1) and is associated with brody disease. *Hum Genet*, 106(5):482–491, May 2000.

[15] G. D. Schuler. Pieces of the puzzle: expressed sequence tags and the catalog of human genes. *J Mol Med*, 75(10):694–698, Oct 1997.

[16] T. Sierocinski, A. Le Bechec, N. Théret and D. Pétritis. Semantic distillation: a method for clustering objects by their contextual specificity. In Springer-Verlag, *Studies in Computational Intelligence*, 2008.

[17] R. Somogyi. Making sense of gene-expression data. *Immunology Today*, Volume 1999, Supplement 1:17–24, 1999.

[18] David L Wheeler, Deanna M Church, Scott Federhen, Alex E Lash, Thomas L Madden, Joan U Pontius, Gregory D Schuler, Lynn M Schriml, Edwin Sequeira, Tatiana A Tatusova, and Lukas Wagner. Database resources of the national center for biotechnology. *Nucleic Acids Res*, 31(1):28–33, Jan 2003.

[19] Itai Yanai, Hila Benjamin, Michael Shmoish, Vered Chalifa-Caspi, Maxim Shklar, Ron Ophir, Arren Bar-Even, Shirley Horn-Saban, Marilyn Safran, Eytan Domany, Doron Lancet, and Orit Shmueli. Genome-wide midrange transcription profiles reveal expression level relationships in human tissue specification. *Bioinformatics*, 21(5):650–659, Mar 2005.

[20] K. Y. Yeung and W. L. Ruzzo. Principal component analysis for clustering gene expression data. *Bioinformatics*, 17(9):763–774, Sep 2001.

8.2 Analysis of clinical data from patient suffering from hepatic fibrosis

8.2.1 Dataset

Semantic distillation had also been tested on clinical data from DNA chips for patients suffering from hepatic fibrosis.

Hepatic fibrosis is defined by an excessive accumulation of extracellular matrix in the liver. As a result of chronic liver damage, its progress ultimately leads to cirrhosis and cancer. Phenotypic transformations of a fibrotic liver involves complex gene regulation networks.

For this study we have a set $\mathbb{B}$ of 700 genes expressed in 14 patients divided into four categories on anatomopathological criteria:

- Three patients with mild fibrosis (F1)
- Three patients with moderate fibrosis (F2)
- Three patients with severe fibrosis (F3)
- Five patients with cirrhosis (F4).

This analysis was conducted in order to find signatures in the expression of genes to accurately characterize the different stages of fibrosis.

The main difference between this study and the previous one is that we work here on histological samples coming from the same type of tissue (liver biopsy), unlike the previous analysis where the difference between samples was biologically more marked. Figure 8.2.1 represents standard deviation of gene expression for the previous (GSE803) dataset (left) and the fibrosis dataset (right). We can note that the fibrosis dataset has a low variability in the expression of these genes. It is therefore easy to imagine that discriminating specific genes in this type of data is far more difficult than in the previous analysis.

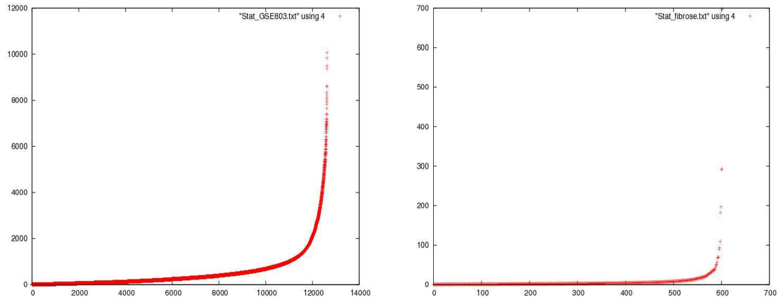
8.2.2 Results

In a first step, the analysis was conducted in the same manner as for the previous dataset, and followed by the same analysis of results (measure of $O(i)$). We recall here the following definitions:

- $\mathsf{precision}$ is the fraction of the genes retrieved that are effectively specific. Let Ret_i represent the subset of Ret containing the first i most specific genes retrieved by our method and Spec the set of specific genes (broad or annotation) to a particular biological sample. Then we define

$$\mathsf{precision} = \frac{|\mathsf{Spec} \bigcap \mathsf{Ret}_i|}{|\mathsf{Ret}_i|}.$$

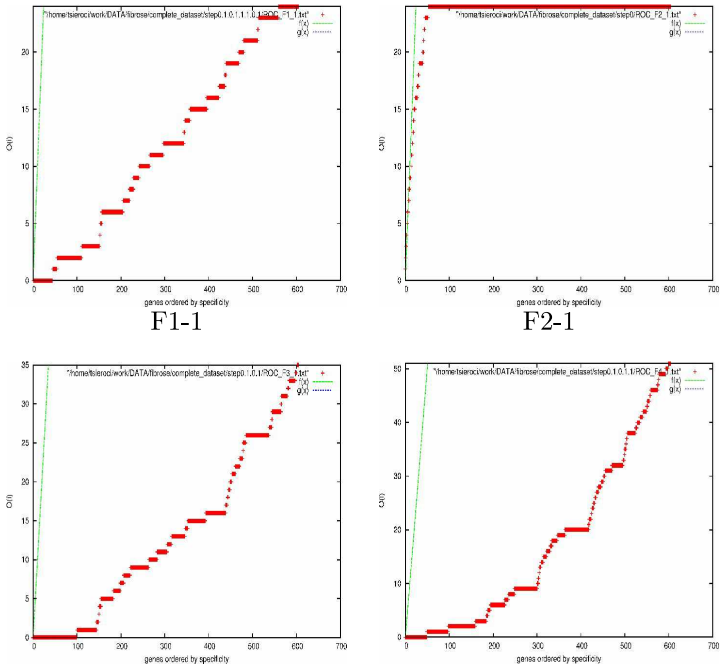
Table 8.1: Comparative analysis of standard deviation of gene expression for GSE803 dataset (left) and fibrosis dataset (right).



GSE803 dataset

fibrosis dataset

Table 8.2: Measures of $O(i)$ for results obtained with the first analysis of fibrosis dataset



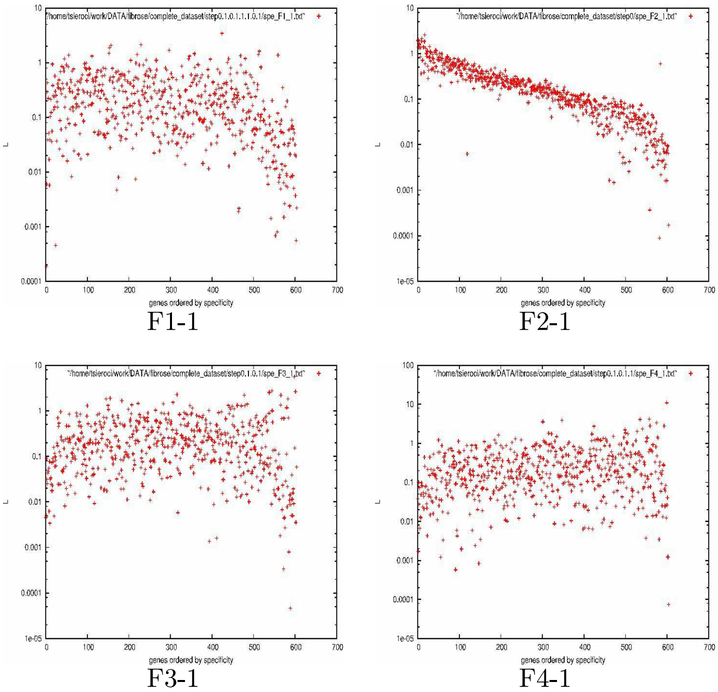
F1-1

F2-1

F3-1

F4-1

Table 8.3: Measures of L for results obtained with the first analysis of fibrosis dataset



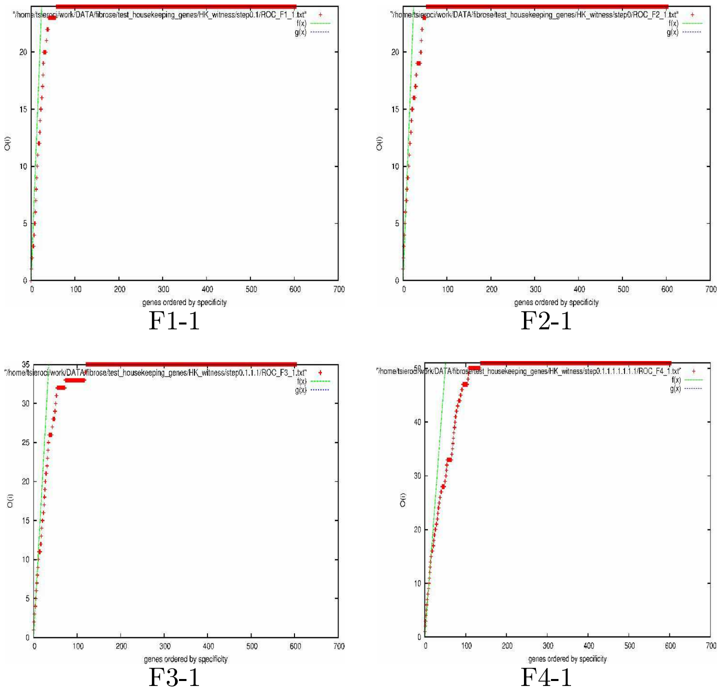
F1-1

F2-1

F3-1

F4-1

Table 8.4: Measures of $O(i)$ for results obtained with the second analysis (with m witness) of fibrosis dataset



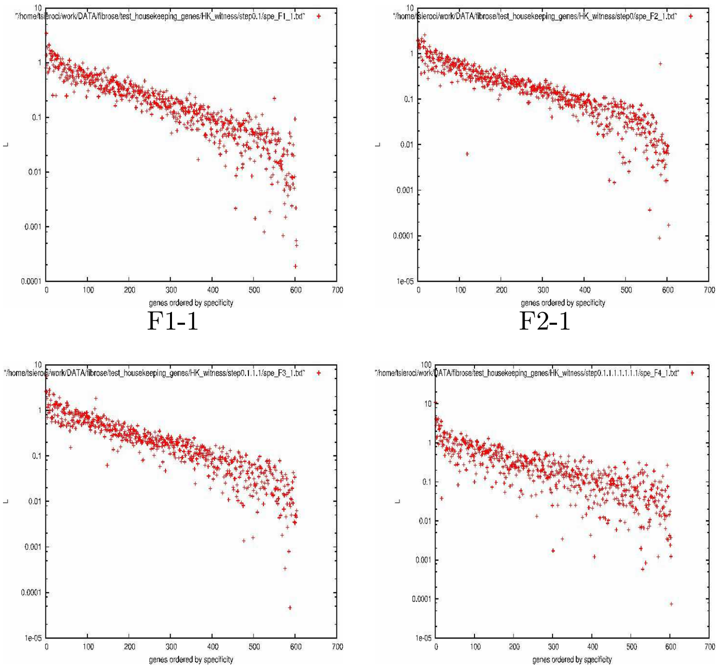
F1-1

F2-1

F3-1

F4-1

Table 8.5: Measures of L for results obtained with the second analysis (with m witness) of fibrosis dataset



F1-1

F2-1

F3-1

F4-1

In binary classification, precision is analogous to positive predictive value. precision takes all retrieved genes into account. It can also be evaluated at a given cut-off rank, considering only the topmost results returned by the system. This measure is called precision at n or $P@n$.

- recall is the fraction of specific genes that are successfully retrieved, more precisely

$$\mathsf{recall} = \frac{|\mathsf{Spec} \bigcap \mathsf{Ret}_i|}{|\mathsf{Spec}|}.$$

  In binary classification, recall is called sensitivity. So it can be viewed as the probability that a specific gene is retrieved by the query. It is trivial to achieve recall of 100% by returning all genes in response to any query. Therefore recall must be combined with precision to decide whether the retrieval is successful.

- Define finally

$$O(i) = |\mathsf{Spec} \bigcap \mathsf{Ret}_i|.$$

  In fact, this measure is the common root of recall and precision. By dividing it by the number of genes we get the precision value, by dividing it by the total number of specific genes we get the recall value.

It appeared that the classification of genes by our method is not optimal here. Figure 8.2.2 shows values of $O(i)$ for some patients (all of these measures are provided in the appendix). We can observe that the classification presented here is much closer to a random draw of genes. We can assume that the low variability of gene expression (as shown in the previous section) does not allow fine and relevant classification of specific genes for each sample biological analysis.

We can ask wether it is possible to make our analysis more discriminant and thus improve the outcome. Given our observation on genes expression in this experience, we can say that a large proportion of genes behave, in the space of concepts, like housekeeping genes, i.e. that their expression is virtually identical regardless of the biological sample analysis. In order to characterise housekeeping genes, we introduce an additional specificity witness vector:

$$m = \sum_{a \in \mathbb{A}} |\, a \,\rangle \in \mathcal{H}_{\mathbb{A}}.$$

Adding this new vector in the dataset does not change its rank. However, recall that the weighed graph encoding the gene interactions will contain now an additional vertex corresponding to m. Seeking a low-dimensional optimal representation of the graph can be though as a minimisation of the energy of springs joining the vertices. Now, the new vertex m strongly attracts all vertices whose vectors form a small angle with m, they are removed from the subspace corresponding to the elementary specificity witnesses $|\, a \,\rangle \in \mathcal{H}_{\mathbb{A}}$. These genes will consequently have a small fuzzy membership to the elementary clusters.

We present in figure 8.2.2 measures effectiveness of the method for the same samples as shown in figure 8.2.2. We can observe that the results are greatly improved by adding the m witness.

Table 8.6: KEGG terms associated to specific genes of each stage of fibrosis, an X represents a term / sample association

KEGG terms	F1	F2	F3	F4
VEGF signalling pathway	X		X	
cell cycle	X	X		
Tcell receptor signalling pathway	X			
Adherens junction		X		
jak-STAT signalling pathway		X	X	X
focal adhesion		X	X	X
wnt signalling pathways		X		
pyrimidine metabolism		X		X
leukocyte transendothelial migration		X		
gap junction			X	
MAPK signalling pathway		X	X	
Regulation of actin cytoskeleton			X	
apoptosis			X	X
Bcell receptor signalling pathway			X	
gap junction			X	
Antigen processing and presentation			X	
One carbon pool by folate			X	
Ubiquitin mediated proteolisis				X
purine metabolism				X
huntington desease				X
neurodegenerative desease				X
dentatorubropallidoluysian atrophy				X
Insulin signalling pathway				X

As for the previous analysis we annotated specific genes for each sample analysis. We chose two annotation sources:

- KEGG (Kyoto Encyclopedia of Genes and Genomes) is a database where each gene is associated to a metabolic pathway in wich it is involved.
- GO (Gene Ontology) is an ontology dedicated to genome annotation. This database associate each gene product with an ontology term (biological process, cellular component and molecular function).

Each gene was annotated with KEGG and GO (only biological process) via M@IA [8]. In this study, we have selected only term with p-value lower than 0.01 for Fisher test.

The results (table 8.6 for KEGG annotation and table 9.14 for GO annotation) shows the differences obtained for annotation of specific genes from our analysis. These tables shows us different signatures obtained for the metabolic pathways and for biological processes associated with specific genes of each desease stage (each stage represents the addition of annotations of its associated samples).

Biological process	F1	F2	F3	F4
positive regulation of I-kappaB kinase/NF-kappaB cascade	X			X
positive regulation of cell proliferation	X	X	X	X
inflammatory response	X			X
ossification	X			
cell division	X	X		
blood coagulation	X			X
positive regulation of transcription from RNA polymerase	X			
response to drug	X			
dephosphorylation	X			X
embryonic development (sensu Mammalia)	X		X	
DNA unwinding during replication	X			
mRNA export from nucleus	X			
DNA replication	X			X
cell surface receptor linked signal transduction	X		X	
protein amino acid phosphorylation	X	X	X	X
immune response	X			
cell cycle	X	X	X	X
skeletal development	X			

suite du tableau page suivante

Biological process	F1	F2	F3	F4
cell proliferation	X			X
cytokinesis		X	X	X
negative regulation of apoptosis		X	X	
regulation of cyclin-dependent protein kinase activity		X		X
regulation of progression through cell cycle		X		
transmembrane receptor protein tyrosine kinase signaling pathway		X		
negative regulation of cell proliferation		X	X	X
intracellular signaling cascade		X		X
development		X		X
cell adhesion		X	X	X
cell cycle arrest		X		
negative regulation of cell cycle		X		
chemotaxis		X		
traversing start control point of mitotic cell cycle		X		
apoptosis		X	X	X
protein ubiquitination			X	
wnt receptor signaling pathway			X	
signal transduction			X	
regulation of apoptosis			X	X
cell-cell signaling			X	X
insulin receptor signaling pathway			X	X
insulin-like growth factor receptor signaling pathway			X	
inner ear morphogenesis			X	
lung development			X	
bone mineralization			X	
angiogenesis			X	X
fibroblast growth factor receptor signaling pathway			X	
nuclear mRNA splicing, via spliceosome			X	
cell growth			X	X
nervous system development			X	X
response to hypoxia				X
induction of apoptosis				X
JNK cascade				X
response to wounding				X
transcription				X
DNA repair				X
response to stress				X
nucleobase, nucleoside, nucleotide and nucleic acid metabolic process				X
protein complex assembly				X
anterior/posterior pattern formation				X
cell motility				X
inactivation of MAPK activity				X
regulation of cell growth				X
negative regulation of cell cycle				X
nucleotide-excision repair				X
anti-apoptosis				X
regulation of transcription, DNA-dependent				X
liver development				X
placenta development				X
release of cytochrome c from mitochondria				X
positive regulation of transcription from RNA polymerase II promoter				X
apoptotic mitochondrial changes				X
germ cell development				X
protein import into nucleus				X
cell morphogenesis				X
keratinocyte differentiation				X
heart development				X
positive regulation of transcription				X
proteolysis				X
apoptotic program				X
organ morphogenesis				X
actin cytoskeleton organization and biogenesis				X
signal complex formation				X
positive regulation of T cell proliferation				X
epidermis development				X
actin filament organization				X
cell differentiation				X
cell migration				X
sensory perception				X
T cell activation				X
DNA replication initiation				X
UTP biosynthesis				X
CTP biosynthesis				X
GTP biosynthesis				X
nucleotide metabolism				X
brain development				X
regulation of Rho protein signal transduction				X
one-carbon compound metabolism				X
transcription from RNA polymerase II promoter				X
small GTPase mediated signal transduction				X
cellular defense response				X

Table 9.14: GO terms (biological processes) associated with genes wich are specific to each fibrosis stages

8.3 Comparison with other methods

Results obtained by semantic distillation were compared with traditional approaches commonly used in the context of DNA arrays analysis: agglomerative hierarchical clustering, fuzzy clustering, *k*-means method. These three methods are provided in M@IA (Microarray Integrated Application) [8]. We chose (for computing time reasons) to take fibrosis dataset as reference.

For this comparison, several difficulties have been encountered. The first one is to compare set-theorical methods with ours. Indeed, semantic distillation performs for each biological sample, a ranking of all genes according to their sample specificity, unlike set-theorical methods that affect each gene in a given group. Secondly, traditional methods usually cluster genes according to their expression profile similarities and not according to their specificity to a particular biological sample. Hence, in the context of traditional approaches, even if we set the number of groups equal to the number of samples, how can we assign a group to a particular sample?

Given these difficulties, the method adopted for the comparison was as follows: each of the methods mentioned previously were parametrised to separate the genes in a number of groups equal to the number of biological samples from the experience (i.e. 14). After clustering, we have counted for each group, the number of broad sense specific genes (defined in section 8.1) to each biological sample. For each of these measures, we then calculated recall and precision of each group for each sample and took their maximal values per sample. We then decided to turn semantic distillation to set-theorical by truncating rankings provided at a rank equal to the number of genes in the group (resulting from a standard approach) with the maximum value (for the recall or accuracy) for a sample.

8.3.1 Agglomerative hierarchical clustering

First, we compared semantic distillation (using angular distance to quantify similarity) with agglomerative hierarchical clustering using Ward's minimum variance method [74] in order to minimise intra-group variability. The results of the measurements were are given in tables 8.8 and 8.9.

Table 8.8 presents recall measures for all groups formed by agglomerative hierarchical clustering. A first observation relates to the size of groups, we can see that it is relatively homogenous. The second point concerns the values of this table and mainly their distribution. We can observe a significant proportion of zero values, they reflect a lack of specific genes in a group associated to a biological sample. The presence of zero values for these measures is normal. Indeed, unless a group contains all the genes of the experience, it is rare that specific genes for each biological sample are represented in the same group. The distribution of non-zero values we gives us information on the contents of each group, we can see that every group (except 12) contains specific genes for several biological samples. Likewise, each biological sample sees its specific genes distributed in several groups. Therefore, which group represents at best the var-

Table 8.8: Comparative analysis for recall between agglomerative hierarchical clustering method and semantic distillation (SD)

	f11	f12	f13	f21	f22	f23	f31	clusters size
cluster1	0	0	0	0	0	0	0,08571	29
cluster2	0	0	0	0,04167	0,10714	0	0	76
cluster3	0,08333	0	0	0,04167	0,10714	0,05556	0	24
cluster4	0	0	0,06122	0	0	0	0,65714	39
cluster5	0	0,03333	0,02041	0	0	0	0	93
cluster6	0,16667	0	0,55102	0,04167	0,03571	0	0	43
cluster7	0	0	0	0	0	0	0	54
cluster8	0,04167	0,76667	0,2449	0	0	0	0,02857	48
cluster9	0	0	0	0,125	0	0,58333	0	42
cluster10	0,45833	0,06667	0	0	0	0	0	23
cluster11	0	0	0	0,54167	0,5	0,11111	0	42
cluster12	0	0	0	0	0	0	0	28
cluster13	0	0	0	0	0	0	0	26
cluster14	0	0	0	0	0	0	0,05714	22
max	0,45833	0,76667	0,55102	0,54167	0,5	0,58333	0,65714	
SD	0,83333	0,73333	0,28571	0,875	0,67857	0,66667	0,74286	

	f32	f33	f41	f42	f43	f44	f45
cluster1	0,48485	0,06061	0	0	0,01493	0	0
cluster2	0	0	0,13725	0,07547	0,67164	0	0
cluster3	0	0	0,01961	0	0	0,07317	0,08824
cluster4	0,15152	0	0	0,01887	0	0	0,0098
cluster5	0,0303	0,09091	0	0,01887	0,01493	0,12195	0,58824
cluster6	0	0,06061	0	0,01887	0	0,02439	0
cluster7	0	0,06061	0,60784	0,13208	0	0,02439	0,06863
cluster8	0	0,0303	0,03922	0	0,01493	0,04878	0
cluster9	0,06061	0	0	0,03774	0,0597	0	0,0098
cluster10	0	0	0,03922	0	0	0	0,01961
cluster11	0	0,0303	0	0	0	0	0,0098
cluster12	0	0	0	0,4717	0	0	0
cluster13	0	0	0	0,01887	0,01493	0,41463	0
cluster14	0	0,45455	0	0	0	0	0
max	0,48485	0,45455	0,60784	0,4717	0,67164	0,41463	0,58824
SD	0,51515	0,24242	0,62745	0,28302	0,59701	0,21951	0,10784

Table 8.9: Comparative analysis for precision between agglomerative hierarchical clustering method and semantic distillation (SD)

	f11	f12	f13	f21	f22	f23
cluster1	0	0	0	0	0	0
cluster2	0	0	0	0,01316	0,03947	0
cluster3	0,08333	0	0	0,04167	0,125	0,08333
cluster4	0	0	0,07692	0	0	0
cluster5	0	0,01075	0,01075	0	0	0
cluster6	0,09302	0	0,62791	0,02326	0,02326	0
cluster7	0	0	0	0	0	0
cluster8	0,02083	0,47917	0,25	0	0	0
cluster9	0	0	0	0,07143	0	0,5
cluster10	0,47826	0,08696	0	0	0	0
cluster11	0	0	0	0,30952	0,33333	0,09524
cluster12	0	0	0	0	0	0
cluster13	0	0	0	0	0	0
cluster14	0	0	0	0	0	0
max	0,47826	0,47917	0,62791	0,30952	0,33333	0,5
SD	0,65217	0,45833	0,32558	0,5	0,45238	0,57143

	f32	f33	f41	f42	f43	f44	f45
cluster1	0,55172	0,06897	0	0	0,03448	0	0
cluster2	0	0	0,09211	0,05263	0,59211	0	0
cluster3	0	0	0,04167	0	0	0,125	0,375
cluster4	0,12821	0	0	0,02564	0	0	0,02564
cluster5	0,01075	0,03226	0	0,01075	0,01075	0,05376	0,64516
cluster6	0	0,04651	0	0,02326	0	0,02326	0
cluster7	0	0,03704	0,57407	0,12963	0	0,01852	0,12963
cluster8	0	0,02083	0,04167	0	0,02083	0,04167	0
cluster9	0,04762	0	0	0,04762	0,09524	0	0,02381
cluster10	0	0	0,08696	0	0	0	0,08696
cluster11	0	0,02381	0	0	0	0	0,02381
cluster12	0	0	0	0,89286	0	0	0
cluster13	0	0	0	0,03846	0,03846	0,65385	0
cluster14	0	0,68182	0	0	0	0	0
max	0,55172	0,68182	0,57407	0,89286	0,59211	0,65385	0,64516
SD	0,58621	0,36364	0,59259	0,5	0,52632	0,34615	0,35484

Table 8.10: Comparative analysis for recall between *k*-means algorithm and semantic distillation (SD)

	f11	f12	f13	f21	f22	f23	f31	cluster size
cluster1	0	0	0	0	0	0	0,085714	10
cluster2	0	0	0	0	0	0	0	8
cluster3	0,375	0	0,469388	0,541667	0,714286	0,611111	0,457143	298
cluster4	0,416667	0	0,367347	0,125	0,035714	0,138889	0,314286	241
cluster5	0	0	0	0	0	0	0	6
cluster6	0	0	0	0	0,035714	0,027778	0	3
cluster7	0	0	0	0	0	0	0	8
cluster8	0	0	0	0	0	0	0	4
cluster9	0	0	0,040816	0	0	0	0	4
cluster10	0	0	0	0	0	0	0	1
cluster11	0	0	0	0	0	0	0	1
cluster12	0	0	0	0,041667	0	0	0	2
cluster13	0	0	0	0	0	0	0	2
cluster14	0	0	0	0	0	0	0	1
max	0,416667	0	0,469388	0,541667	0,714286	0,611111	0,457143	
SD	1	1	0,77551	1	1	1	1	

	f32	f33	f41	f42	f43	f44	f45
cluster1	0,030303	0,121212	0	0	0	0	0
cluster2	0	0	0	0,018868	0,074627	0	0
cluster3	0,454545	0,30303	0,333333	0,113208	0,238806	0,463415	0,519608
cluster4	0,181818	0,393939	0,372549	0,603774	0,373134	0,219512	0,284314
cluster5	0	0	0	0,056604	0	0	0
cluster6	0	0	0	0	0	0	0
cluster7	0	0	0,137255	0	0	0	0
cluster8	0	0	0	0	0,044776	0	0
cluster9	0	0	0	0	0	0,02439	0
cluster10	0	0	0	0	0	0	0
cluster11	0	0	0	0	0	0	0
cluster12	0	0	0	0	0	0	0
cluster13	0	0	0	0	0,014925	0	0
cluster14	0	0	0	0	0	0	0
max	0,454545	0,393939	0,372549	0,603774	0,373134	0,463415	0,519608
SD	1	1	1	1	1	0,97561	0,843137

ious biological samples from the experience? As we have seen before, we chose the group presenting the maximum value for recall (or precision) for a sample as characteristic of it. A group may therefore represent several samples, this mark a huge disadvantage of the traditional methods compared with semantic distillation. The last two rows present, for each sample, the maximum value of the recall obtained with hierarchical clustering and our degraded method (truncature of our ranking for a rank equal to the size of the group associated with the maximum value of the recall). Despite the handicap of these measures for our method (loss of genes ranking by truncature, size of the group determined by the competitor method), we can observe that in half cases semantic distillation had a better recall than agglomerative hierarchical clustering.

Table 8.9 present precision measurements made on the groups formed by hierarchical clustering. The same observations can be made on these measures (distribution of non-zero values, better precision in half the cases). We can, however, add an additional remark: groups for which semantic distillation presents a highest recall are also generally more accurate. We, therefore, for these groups, jointly improved recall and precision.

8.3.2 *k*-means algorithm

Semantic distillation was then compared with the *k*-means method detailled in the chapter 4. In the same way than for the previous analysis tables 8.10 and 8.11 respectively introduced measures of recall and precision for groups formed by this method.

Table 8.11: Comparative analysis for precision between k-means algorithm and semantic distillation (SD)

	f11	f12	f13	f21	f22	f23	f31
cluster1	0	0	0	0	0	0	0,3
cluster2	0	0	0	0	0	0	0
cluster3	0,030201	0,036913	0,077181	0,043624	0,067114	0,073826	0,053691
cluster4	0,041494	0,062241	0,074689	0,012448	0,004149	0,020747	0,045643
cluster5	0	0	0	0	0	0	0
cluster6	0	0	0	0	0,333333	0,333333	0
cluster7	0	0	0	0	0	0	0
cluster8	0	0	0	0	0	0	0
cluster9	0	0	0,5	0	0	0	0
cluster10	0	0	0	0	0	0	0
cluster11	0	0	0	0	0	0	0
cluster12	0	0	0	0,5	0	0	0
cluster13	0	0	0	0	0	0	0
cluster14	0	0	0	0	0	0	0
max	0,041494	0,062241	0,5	0,5	0,333333	0,333333	0,3
SD	0,099585	0,124481	1	1	1	0,333333	0,4

	f32	f33	f41	f42	f43	f44	f45
cluster1	0,1	0,4	0	0	0	0	0
cluster2	0	0	0	0,125	0,625	0	0
cluster3	0,050336	0,033557	0,057047	0,020134	0,053691	0,063758	0,177852
cluster4	0,024896	0,053942	0,078838	0,13278	0,103734	0,037344	0,120332
cluster5	0	0	0	0,5	0	0	0
cluster6	0	0	0	0	0	0	0
cluster7	0	0	0,875	0	0	0	0
cluster8	0	0	0	0	0,75	0	0
cluster9	0	0	0	0	0	0,25	0
cluster10	0	0	0	0	0	0	0
cluster11	0	0	0	0	0	0	0
cluster12	0	0	0	0	0	0	0
cluster13	0	0	0	0	0,5	0	0
cluster14	0	0	0	0	0	0	0
max	0,1	0,4	0,875	0,5	0,75	0,25	0,177852
SD	0,8	0,6	1	0,666667	1	0,75	0,288591

We can observe that in this case, the size of groups formed by the k-means method is very heterogeneous. Indeed, on 14 groups 12 contain less than 10 elements and two groups (3 and 4) contain respectively 298 and 241 genes. This explains mainly the distribution of non-zero values in the table wich are grouped in groups 3 and 4 and is due to the low variability in the expression of genes in this experiment. Given the size of these two groups, their chances to contain specific genes are not negligible and results in relatively high recall values. However, recall obtained with semantic distillation is higher for all biological samples from experience. Regarding precision, large groups 3 and 4 here contributes to a sharp decline in the accuracy of predicitions for those groups. Those with high precision contain, in general, a small number of genes. As for recall values, we can see that the precision values for our method are higher for all samples than values for k-means method.

8.3.3 Fuzzy clustering

We present here the comparison with fuzzy clustering algorithm. The method used here is detailed in [37]. Tables 8.12 and 8.13 respectively measure of recall and precision. The size of groups formed here is more homogeneous than previously. The recall values are generally better for semantic distillation (except for samples f13, f23, f42, f43 and f45). Precision values are also generally better for the semantic distillation (except for samples f13, f23, f42, f43 and f45). As for comparison with the agglomerative hierarchical clustering, groups for which semantic distillation presents a higher recall are also generally more accurate. We, therefore, for these groups, jointly improved the recall and precision.

Table 8.12: Comparative analysis for recall between fuzzy clustering algorithm and semantic distillation (SD)

	f11	f12	f13	f21	f22	f23	f31	cluster size
cluster1	0	0,033333	0,142857	0	0	0	0,285714	25
cluster2	0	0	0	0	0	0	0	70
cluster3	0,041667	0	0,040816	0,583333	0,535714	0,138889	0	38
cluster4	0	0,033333	0	0,041667	0	0	0,028571	25
cluster5	0	0	0	0	0	0,027778	0	41
cluster6	0,083333	0,233333	0,102041	0	0	0	0,142857	42
cluster7	0,041667	0	0,020408	0,041667	0,071429	0	0	29
cluster8	0,041667	0	0	0,166667	0,214286	0,138889	0,028571	30
cluster9	0	0	0	0	0,035714	0	0	53
cluster10	0,25	0,2	0,122449	0	0	0	0	30
cluster11	0,541667	0,433333	0,530612	0,041667	0,035714	0	0	57
cluster12	0	0	0	0	0	0	0,485714	64
cluster13	0	0	0,020408	0	0	0	0	63
cluster14	0	0	0	0,125	0,071429	0,666667	0	36
max	0,541667	0,433333	0,530612	0,583333	0,535714	0,666667	0,485714	
SD	1	0,833333	0,306122	0,791667	0,607143	0,583333	0,914286	

	f32	f33	f41	f42	f43	f44	f45
cluster1	0,090909	0,090909	0	0	0	0	0
cluster2	0	0,030303	0	0	0	0,219512	0,578431
cluster3	0	0	0	0	0	0	0
cluster4	0,030303	0,030303	0	0,018868	0,134328	0,04878	0,068627
cluster5	0	0	0,235294	0,075472	0	0,170732	0,156863
cluster6	0,121212	0,030303	0	0	0	0,219512	0,078431
cluster7	0,030303	0,060606	0,392157	0	0	0	0
cluster8	0,060606	0	0	0	0	0,146341	0,04902
cluster9	0	0	0,215686	0,735849	0	0,02439	0
cluster10	0	0	0,058824	0	0	0,097561	0,04902
cluster11	0	0	0	0	0,014925	0,02439	0
cluster12	0,636364	0,666667	0,019608	0,037736	0	0,02439	0
cluster13	0,030303	0	0,019608	0,075472	0,80597	0,02439	0
cluster14	0	0,060606	0,019608	0,018868	0,029851	0	0
max	0,636364	0,666667	0,392157	0,735849	0,80597	0,219512	0,578431
SD	0,848485	0,818182	0,411765	0,283019	0,522388	0,365854	0,27451

Table 8.13: Comparative analysis for precision between fuzzy clustering algorithm and semantic distillation (SD)

	f11	f12	f13	f21	f22	f23	f31
cluster1	0	0,04	0,28	0	0	0	0,4
cluster2	0	0	0	0	0	0	0
cluster3	0,026316	0	0,052632	0,368421	0,394737	0,131579	0
cluster4	0	0,04	0	0,04	0	0	0,04
cluster5	0	0	0	0	0	0,02439	0
cluster6	0,047619	0,166667	0,119048	0	0	0	0,119048
cluster7	0,034483	0	0,034483	0,034483	0,068966	0	0
cluster8	0,033333	0	0	0,133333	0,2	0,166667	0,033333
cluster9	0	0	0	0	0,018868	0	0
cluster10	0,2	0,2	0,2	0	0	0	0
cluster11	0,22807	0,22807	0,45614	0,017544	0,017544	0	0
cluster12	0	0	0	0	0	0	0,265625
cluster13	0	0	0,015873	0	0	0	0
cluster14	0	0	0	0,083333	0,055556	0,666667	0
max	0,22807	0,22807	0,45614	0,368421	0,394737	0,666667	0,4
SD	0,578947	0,438596	0,263158	0,5	0,447368	0,583333	0,5

	f32	f33	f41	f42	f43	f44	f45
cluster1	0,12	0,12	0	0	0	0	0
cluster2	0	0,014286	0	0	0	0,128571	0,842857
cluster3	0	0	0	0	0	0	0
cluster4	0,04	0,04	0	0,04	0,36	0,08	0,28
cluster5	0	0	0,292683	0,097561	0	0,170732	0,390244
cluster6	0,095238	0,02381	0	0	0	0,214286	0,190476
cluster7	0,034483	0,068966	0,689655	0	0	0	0
cluster8	0,066667	0	0	0	0	0,2	0,166667
cluster9	0	0	0,207547	0,735849	0	0,018868	0
cluster10	0	0	0,1	0	0	0,133333	0,166667
cluster11	0	0	0	0	0,017544	0,017544	0
cluster12	0,328125	0,34375	0,015625	0,03125	0	0,015625	0
cluster13	0,015873	0	0,015873	0,063492	0,857143	0,015873	0
cluster14	0	0,055556	0,027778	0,027778	0,055556	0	0
max	0,328125	0,34375	0,689655	0,735849	0,857143	0,214286	0,842857
SD	0,4375	0,421875	0,724138	0,471698	0,555556	0,357143	0,4

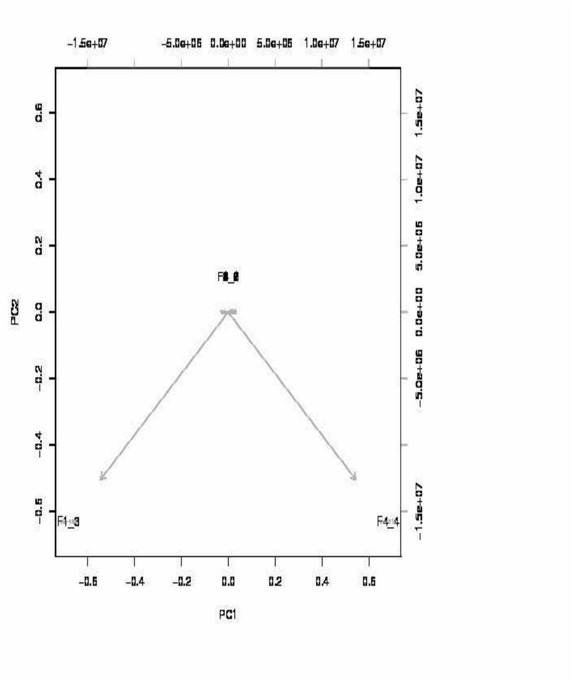


Figure 8.1: Principal components analysis of fibrosis dataset.

8.3.4 Principal component analysis

We also sought to compare our approach to the principal components analysis, but the low variability of the dataset does not allow this analysis to separate data into 14 distinct groups. We present in figure 8.1 a representation based on the two principal components of the dataset, we can observe the small number of genes that differs from the compact group, explaining the difficulty of separating the data into 14 groups.

8.3.5 Conclusion

It seems obvious, with the previous analysis, that semantic distillation does not characterise the same information as traditional methods of data clustering. As we have mentioned several times during this study, our method provides the user with an ordered list of specific genes for every biological sample while standard approaches presented here characterize groups of genes with similar expression profiles. This explains the difficulty to make a clear comparison of results.

However, those measurements allow us to get a glimpse of the quality of the results obtained with our method. The latter, although disadvantaged by comparison method (loss of the concept of rank and size of groups imposed) performs equally well (in the worst case) or largely outperforms the traditional methods.

Chapitre 9

Conclusion et perspectives

Cette étude porte sur l'extraction d'information appliquée au cadre de l'analyse de résultats expérimentaux de puces à ADN. Les puces à ADN permettent de mesurer l'activité des gènes (au travers du taux d'ARN messager) d'une cellule à un instant donné. Ces mesures fournissent des données sous forme de matrices :

$$\mathsf{X} = (x_{ab})_{a \in \mathbb{A}, b \in \mathbb{B}}$$

de taille $|\mathbb{A}| \times |\mathbb{B}|$ où x_{ab} représente l'expression du gène b dans l'échantillon biologique a. Une étude de cette représentation des données est effectuée dans le chapitre 3.

Ce type d'expérience est dit à « haut débit » en raison du volume de données qu'elles génèrent excluant ainsi une interprétation « humaine » des résultats. Dans ce but, de nombreuses approches (mathématiques, statistiques, informatiques) ont été développées, celles-ci pouvant être classées en deux grandes familles : les méthodes de classification supervisées et non supervisées. Les premières requièrent l'apport de connaissances *a priori* afin d'être effectives. Malheureusement, l'état actuel des connaissances ainsi que la variabilité des expériences de puces ne permet pas toujours leur application.

Il peut donc être avantageux de pouvoir analyser des données sans aucune connaissance *a priori* du système. C'est dans ce cadre que nous avons situé le travail de cette thèse. Les fondements des approches de classification non supervisées sont détaillées dans les chapitres 4 et 5 présentant respectivement les méthodes ensemblistes et les méthodes spectrales de regroupement de données. Ces approches de regroupement (ou clustering) repartissent généralement l'ensemble des gènes en groupes distincts en fonction de leurs similitudes d'expression. Ce type d'approche permet principalement, par rapprochement avec des gènes connus, d'inférer un rôle à un nouveau gène rendant inévitable l'introduction de connaissance. De plus, ces méthodes affectent généralement un gène à un seul et unique groupe ne nous renseignant que peu sur l'implication d'un gène au sein du système.

L'approche dévellopée ici s'avère être profondémment différente, elle repose sur trois points fondamentaux. Tout d'abord, en l'absence totale de connaissance *a priori* sur le système, il est nécessaire de pouvoir caractériser de manière précise chacun de ses « états » (par exemple, en mettant en évidence des signatures d'expression pour chacun des échantillons). De la même manière, le rôle de chaque objet (gènes) au sein du système doit être clairement défini (en quantifiant la participation de chaque gène au phénotype de l'échantillon). Le dernier point, spécifique à la thématique des puces à ADN, repose sur la fonction et la régulation des gènes : ceux-ci ne sont en général pas spécifiques d'un seul échantillon biologique. Ils interviennent dans plusieurs contextes cellulaires avec des niveaux d'expression différents et en collaboration avec d'autres gènes.

Ces trois points ont conduit à la conception d'une nouvelle approche appelée « distillation sémantique ». Cette méthode fournit à l'utilisateur une liste ordonnée des gènes par spécificité pour chaque échantillon biologique de l'expérience, décrivant chaque contexte cellulaire ainsi que la participation de chaque gène dans ces contextes. L'originalité de cette approche (détaillé dans le chapitre 7) réside dans son formalisme probabiliste issu de la mécanique quantique decrit dans le chapitre 6 et surtout dans la fonction d'appartenance floue (chapitre 7) permettant d'attribuer à chaque gène une probabilité d'appartenance à un échantillon biologique.

La distillation sémantique a, tout d'abord, été mise à l'épreuve sur un jeu de données de mesures dites « tissus-spécifiques » (partie 8.1) permettant ainsi de valider notre approche en caractérisant les gènes spécifiques à chaque tissu de l'expérience. Dans un second temps, nous avons distillé un jeu de données cliniques, composé de mesure d'activité de gènes chez des patients souffrant de fibroses hépatiques à divers stades (partie 8.2). Cette étude nous a permis de caractériser des signatures dans les voies métaboliques ainsi que les processus biologiques associés aux gènes spécifiques de chaque stade fibrotique. Les résultats de la distillation sémantique ont été comparés à ceux obtenus avec des approches plus traditionnelles et permis de montrer (dans la limite de la comparaison effectuée) que notre méthode égale (dans le pire des cas) ou surpasse les approches classiques (partie 8.3).

Bien que la distillation sémantique semble être une méthode efficace pour l'extraction d'information, elle présente pour l'instant un certain nombre de limitations (celle-ci sont détaillées dans la partie 8.1). Tout d'abord, la réduction de l'espace des concepts à chaque itération de la méthode peut avoir un impact sur la qualité des résultats lorsque le nombre d'échantillon biologique devient faible. Une solution pour contourner ce problème pourrait être un retour pondéré de l'information dans le système. Deuxièmement, nous avons présenté dans cette thèse le cas simple de la représentation unidimensionelle du graphe pondéré. En fait, celle-ci est potentiellement la moins robuste des représentations de faible dimension. Une exploration des représentations de dimension supérieure est entamée avec des résultats préliminaires prometteurs. Une troisième limitation est que le nombre de gènes doit être (très) supérieur au nombre d'échantillons biologiques (ce qui représente la majorité des cas d'expériences de puces) et non l'inverse. Ensuite, la complexité de notre algorithme ne permet pas l'analyse de très grands jeux de données (puces pangénomiques par exemple) sur des ordinateurs de bureau classiques. Par exemple, l'analyse du jeu de données GSE803 nécessite 600 Mo de mémoire vive et environ 36 heures de temps CPU. Pour finir, comme pour toutes les approches d'analyse de données, une réponse précise à une question biologique nécessite une connaissance et une comprehension des grands principes de la méthode, un investissement minimum est donc requis de la part de l'utilisateur (un package R est en cours de développement afin de faciliter l'accès à la méthode).

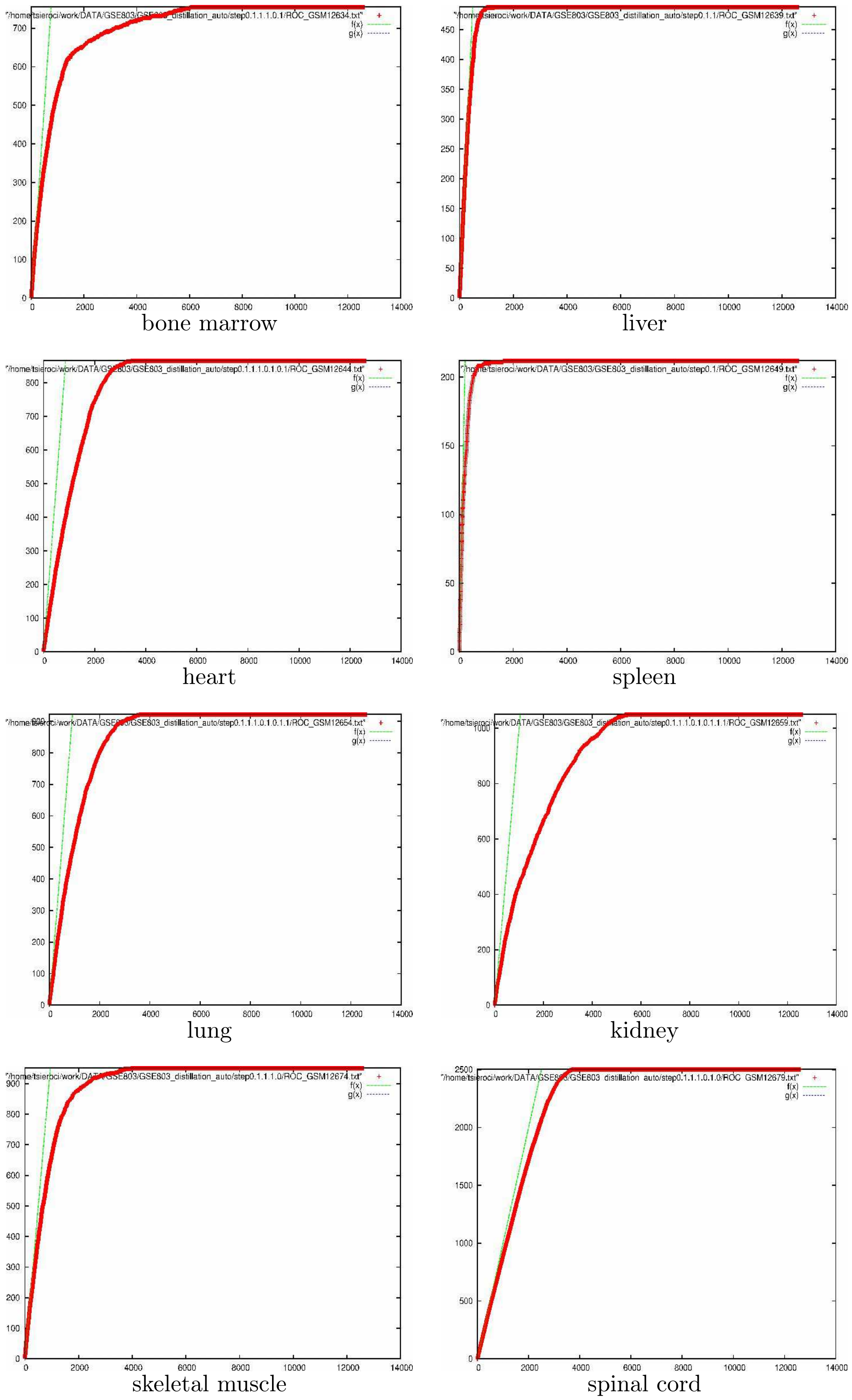
L'état actuel de nos travaux laisse entrevoir de nombreuses perspectives d'utilisation de la méthode : la plus évidente, au regard des expériences menées dans ce travail, est la détection de signatures dans l'expression des gènes permettant la caractérisation de pathologies (comme les fibroses par exemple) et la détermination de gènes candidats pour des recherches approfondies, de manière générale servir d'outil d'aide à la décision. Une autre utilisation, moins directe, intervient dans la cadre de la reconstruction de réseaux de régulation nécessitant la résolution de diverses problématiques notamment l'identification des gènes impliqués dans la réponse d'un organisme à un stimulus particulier. Enfin, de manière générale, tous les jeux de données présentant une structure similaire aux résultats de puces peuvent être analysés avec cette méthode (linguistique, web-semantique, analyse d'images,...).

Appendix A

Supplemental figures for GSE803 dataset analysis

Supplemental figures for measures of $O(i)$ are provided in table A.1 and table A.2. Supplemental figures for measures of L are provided in table A.3 and table A.4.

Table A.1: Supplemental figures for measures of $O(i)$ in the case of GSE803 dataset (part I)



bone marrow

liver

heart

spleen

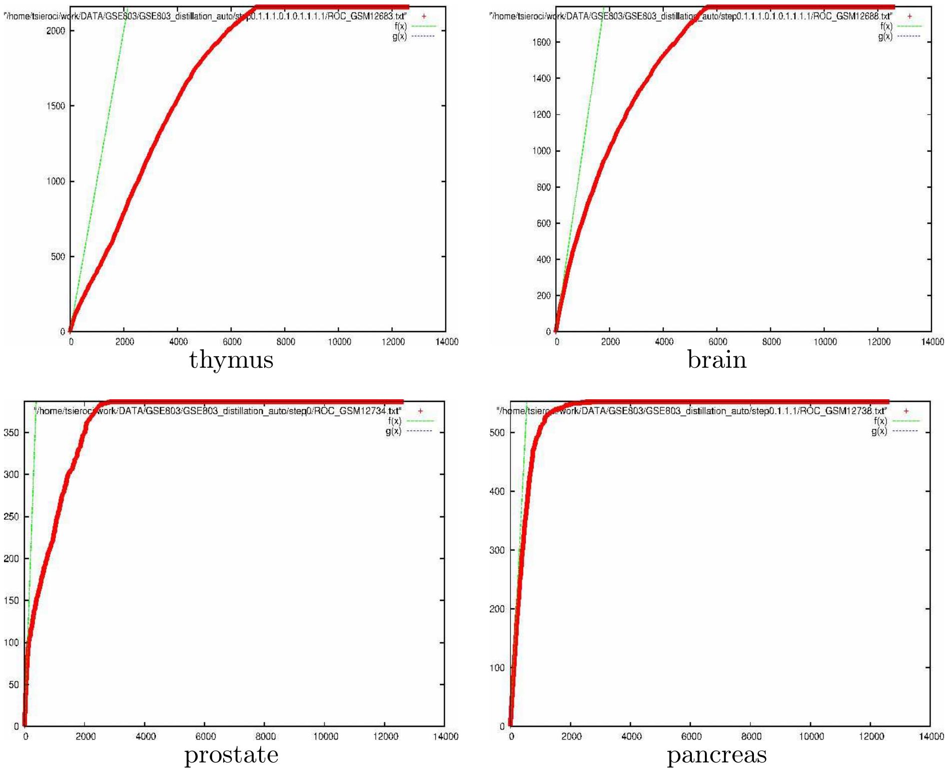
lung

kidney

skeletal muscle

spinal cord

Table A.2: Supplemental figures for measures of $O(i)$ in the case of GSE803 dataset (part II)



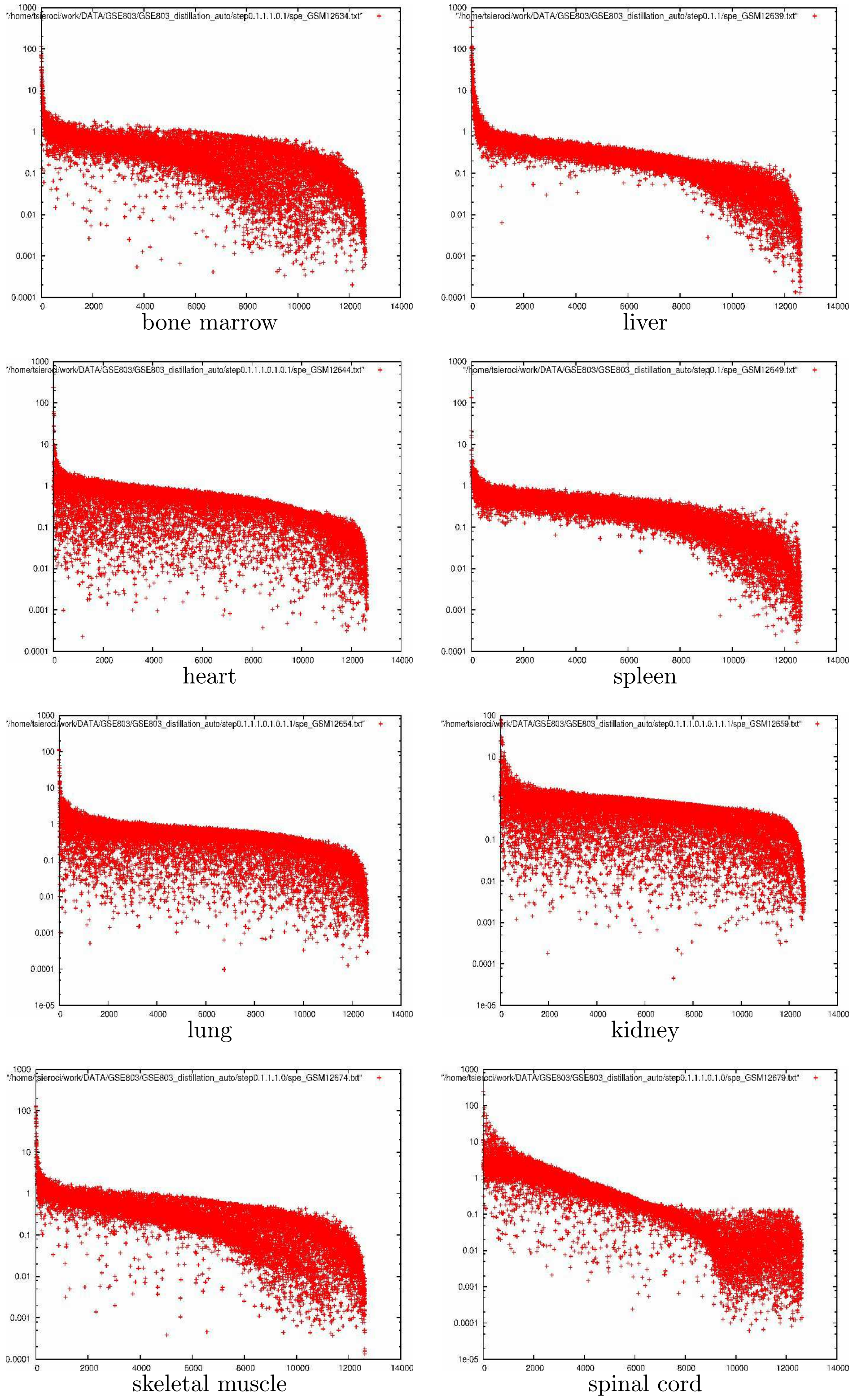
thymus

brain

prostate

pancreas

Table A.3: Supplemental figures for measures of L in the case of GSE803 dataset (part I)



bone marrow

liver

heart

spleen

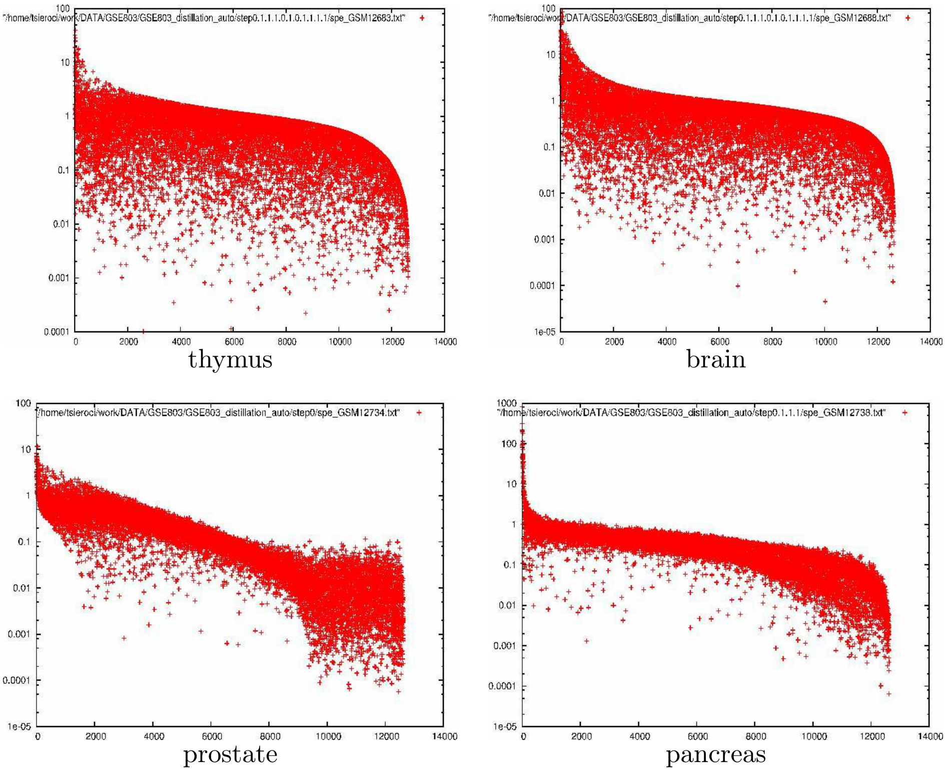
lung

kidney

skeletal muscle

spinal cord

Table A.4: Supplemental figures for measures of L in the case of GSE803 dataset (part II)



thymus

brain

prostate

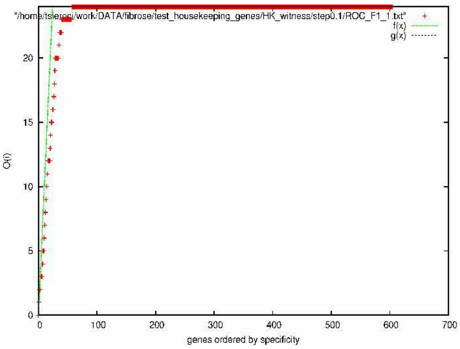
pancreas

Appendix B

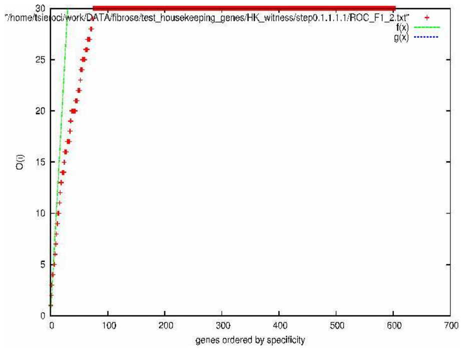
Supplemental figures for fibrosis dataset analysis

Supplemental figures for measures of $O(i)$ are provided in table B.1 and table B.2. Supplemental figures for measures of L are provided in table B.3 and table B.4.

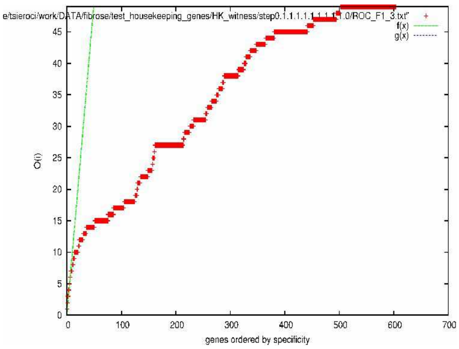
Table B.1: Supplemental figures for measures of $O(i)$ in the case of fibrosis dataset (part I)



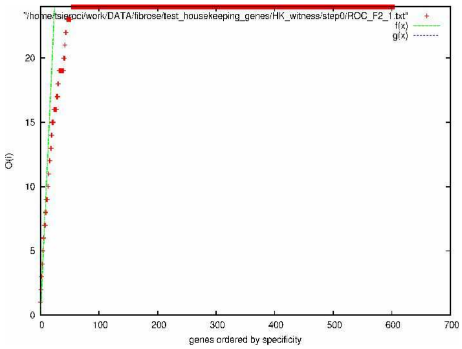
F11



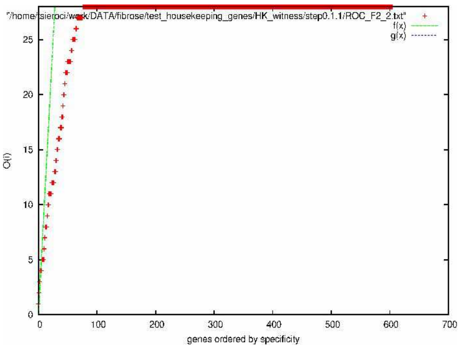
F12



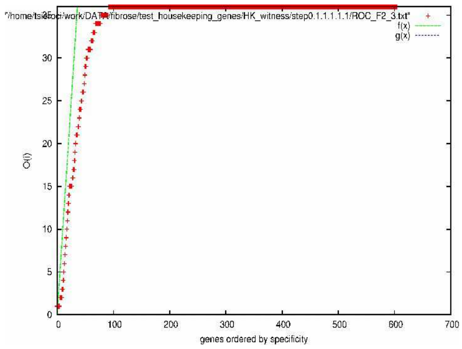
F13



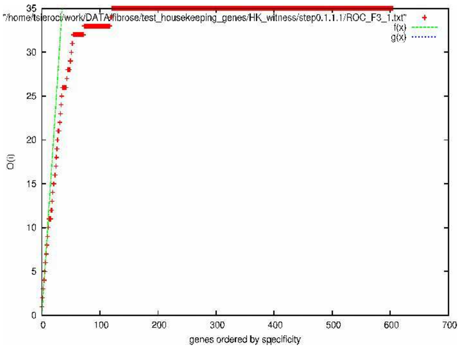
F21



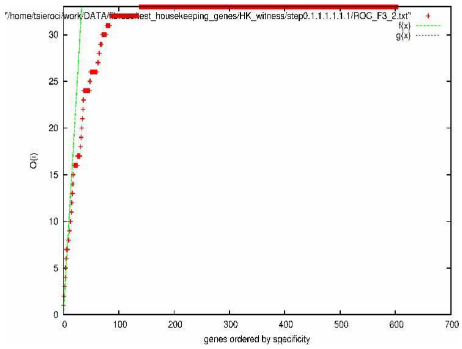
F22



F23

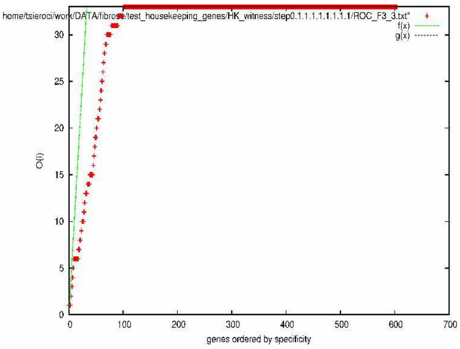


F31

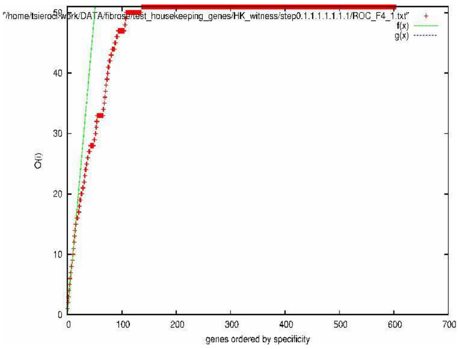


F32

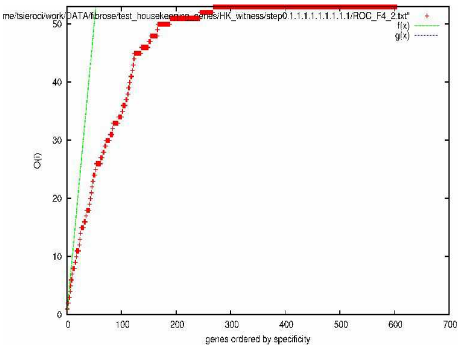
Table B.2: Supplemental figures for measures of $O(i)$ in the case of fibrosis dataset (part II)



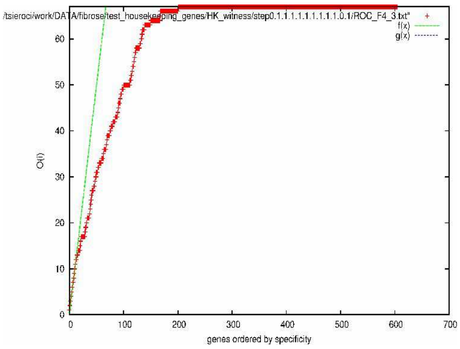
F33



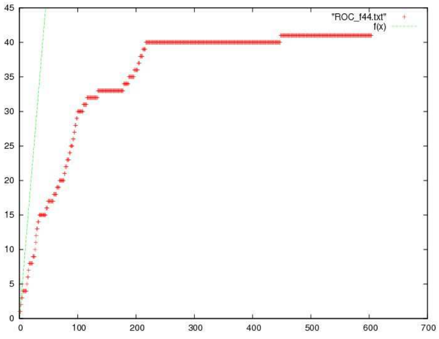
F41



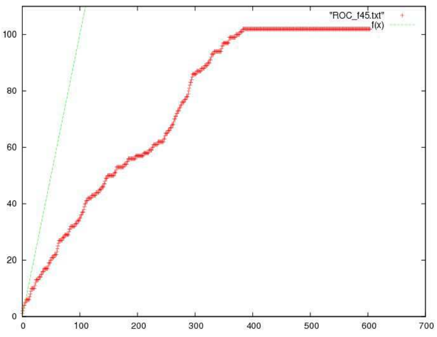
F42



F43

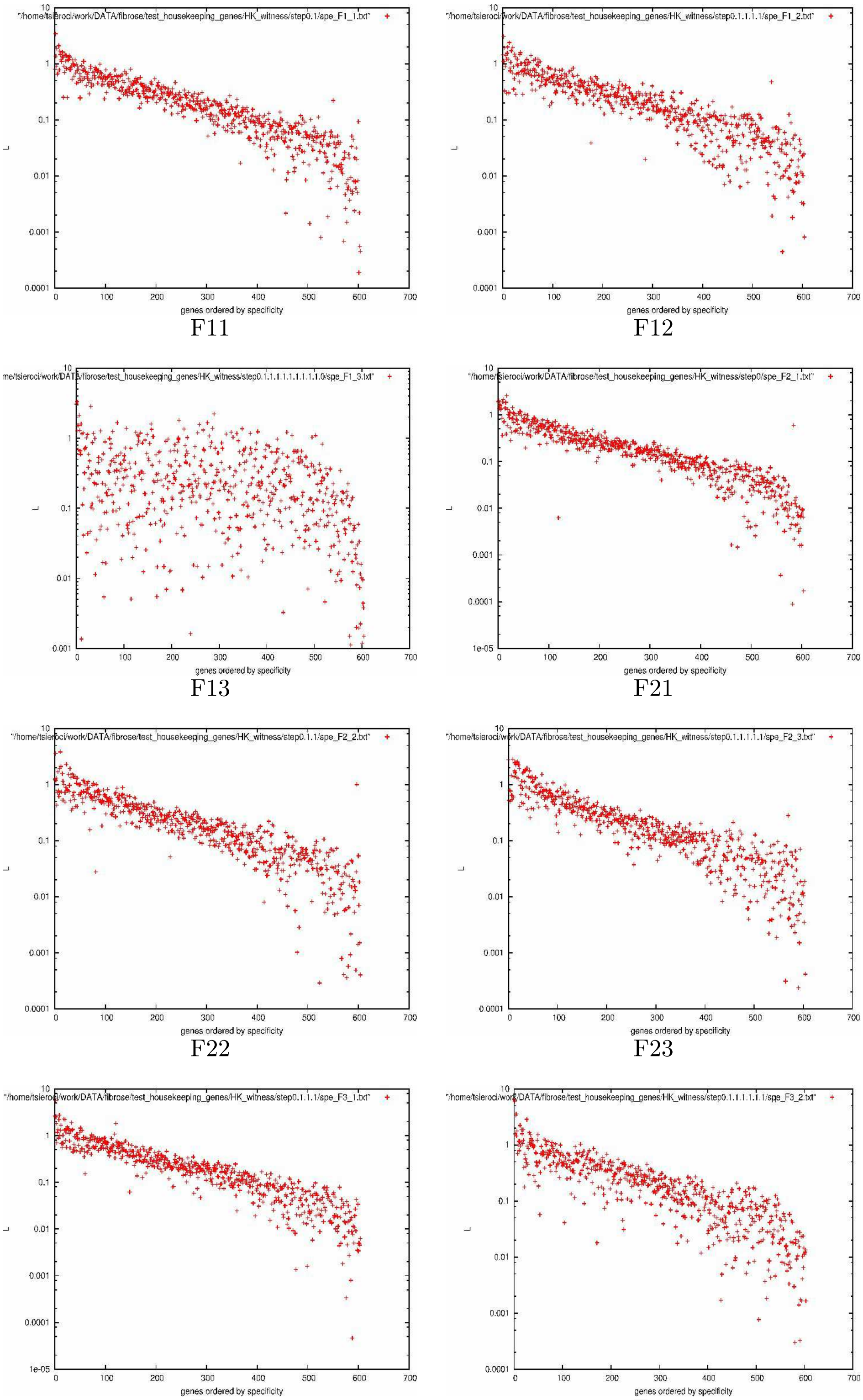


F44



F45

Table B.3: Supplemental figures for measures of L in the case of fibrosis dataset (part I)



F11

F12

F13

F21

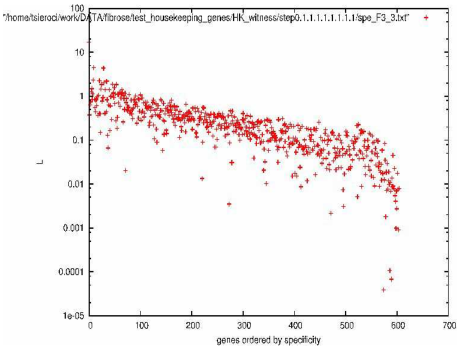
F22

F23

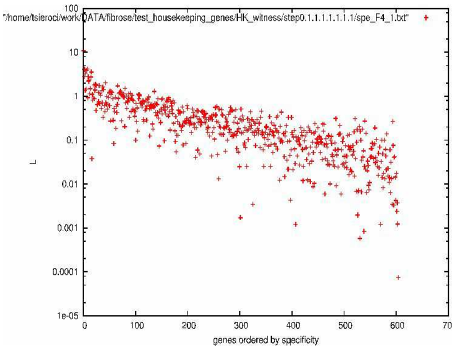
F31

F32

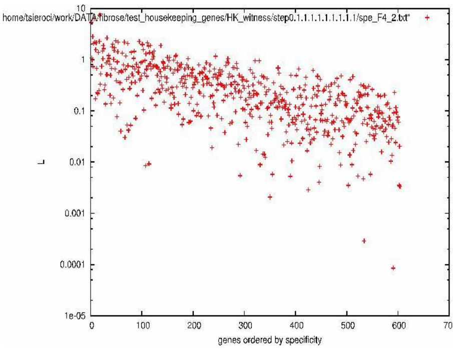
Table B.4: Supplemental figures for measures of L in the case of fibrosis dataset (part II)



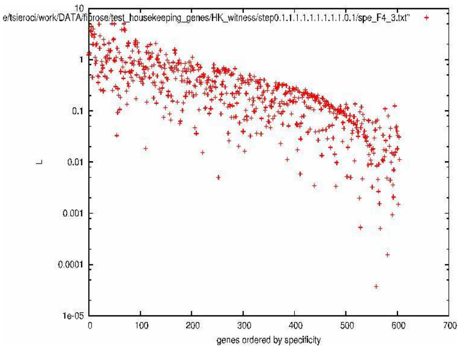
F33



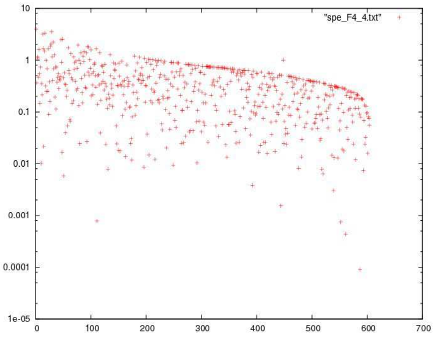
F41



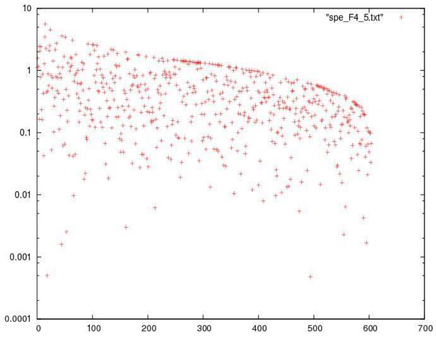
F42



F43



F44



F45

Appendix C

Main routines of semantic distillation source code

We also provide here source code of the main routines implemented in this study.

```
/* allocate.c */

#include <stdio.h>
#include <stdlib.h>
#include "allocate.h"



char **f2(int a, int b){
     int i;

     char **myPointer = (char **)malloc (a * sizeof(char*));
     for (i=0; i<a; i++)
         myPointer[i] = (char *)malloc (b * sizeof(char));
     return myPointer;
}

float **f2float(int a, int b){
     int i;

     float **myPointer = (float **)malloc (a * sizeof(float*));
     for (i=0; i<a; i++)
         myPointer[i] = (float *)malloc (b * sizeof(float));
     return myPointer;
}

int **f2int(int a, int b){
     int i;

     int **myPointer;
     if (!(myPointer = (int **)malloc (a * sizeof(int*)))){
        printf("erreur d'allocation\n");
     }
     else {
        for (i=0; i<a; i++){
                if (!(myPointer[i] = (int *)malloc (b * sizeof(int)))){
                        printf("erreur d'allocation\n");

                }
        }
     }

     return myPointer;
}


char ***f3(int *Dim){
     int i, j;
     int a = Dim[0];
     int b = Dim[1];
     int c = Dim[2];

     char ***myPointer = (char ***)malloc (a * sizeof(char**));
     for (i=0; i<a; i++)
         myPointer[i] = (char **)malloc (b * sizeof(char*));
     for (i=0; i<a; i++)
         for (j=0; j<b; j++)
             myPointer[i][j] = (char *)malloc (c * sizeof(char));
     return myPointer;
}

char ***f33(int a, int b, int c){
     int i, j;

     char ***myPointer = (char ***)malloc (a * sizeof(char**));
     for (i=0; i<a; i++)
         myPointer[i] = (char **)malloc (b * sizeof(char*));
     for (i=0; i<a; i++)
         for (j=0; j<b; j++)
             myPointer[i][j] = (char *)malloc (c * sizeof(char));
     return myPointer;
}
```

```c
char ***deletePointer3(char ***myPointer, int a, int b, int c){
     int i, j;

     for (i=0; i<a; i++)
     {
         for (j=0; j<b; j++)
             free (myPointer[i][j]);
         free(myPointer[i]);
     }
     free (myPointer);
     myPointer = NULL;
     return myPointer;
}

char **deletePointer2(char **myPointer, int a, int b){
     int i;

     for (i=0; i < a; i++)
     {
         free(myPointer[i]);
     }
     free (myPointer);
     myPointer = NULL;
     return myPointer;
}

float **deletePointer2float(float **myPointer, int a, int b){
     int i;

     for (i=0; i < a; i++)
     {
         free(myPointer[i]);
     }
     free (myPointer);
     myPointer = NULL;
     return myPointer;
}

int **deletePointer2int(int **myPointer, int a, int b){
     int i;

     for (i=0; i < a; i++)
     {
         free(myPointer[i]);
     }
     free (myPointer);
     myPointer = NULL;
     return myPointer;
}
```

```c
#ifndef _ALLOCATE_
#define _ALLOCATE_

char **f2(int a, int b);
float **f2float(int a, int b);
int **f2int(int a, int b);
char ***f3(int *Dim);
char ***f33(int a, int b, int c);
char ***deletePointer3(char ***myPointer, int a, int b, int c);
char **deletePointer2(char **myPointer, int a, int b);
float **deletePointer2float(float **myPointer, int a, int b);
int **deletePointer2int(int **myPointer, int a, int b);

#endif
```

```c
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include "filework.h"
#include "out.h"
#include "allocate.h"

#define LG 2000
char filename[LG];
char filename2[LG];


char **NBase;
char **NVecteurs;
int **Coord;
int *Size, *S;
float **EvCL;

int main (int argc, char *argv[]){

        long nbcar = 0;
        char *tabfile;
        char ***Mtab;
        int i, temp;
        strcpy(filename, argv[1]);
        strcpy(filename2, argv[2]);
        nbcar = DimDefine(filename);
        tabfile = file_to_tab(nbcar, filename);
        Size = TabDimDefine(tabfile);
        Mtab = parse(tabfile, Size);
        free(tabfile);
        NVecteurs = GetNamesCol(Mtab, Size);
        Mtab = deletePointer3(Mtab, Size[0], Size[1], Size[2]);
        nbcar = DimDefine(filename2);
        tabfile = file_to_tab(nbcar, filename2);
        S = TabDimDefine(tabfile);
        Mtab = parse(tabfile, S);
        free(tabfile);
        printAnnotation(NVecteurs, Mtab, Size, S);
        return 0;
}
```

```
# a executer apres avoir filtre sur la norme
ulimit -s unlimited
AddR data.txt > AddR_data.txt
GT2GenLap AddR_data.txt > matrice.dat
FSpecU
VP2DimInf vpvp.dat 1 > 2ev.txt
VP2DimInfInd vpvp.dat 1 > Ind2ev.txt
EvNamesInd AddR_data.txt Ind2ev.txt > Nind2ev.txt
PrintNindContext AddR_data.txt Nind2ev.txt > NindContext.txt
PrintIndContext AddR_data.txt Nind2ev.txt > IndContext.txt
ProfileSortNo_Ind AddR_data.txt Nind2ev.txt > total.txt
VP2DimInf vpvp.dat 2 > 2D.txt
VP2DimInfCtx vpvp.dat AddR_data.txt 2 > 2DContext.txt
VP2DimInfCtxInd vpvp.dat AddR_data.txt 2 > 2DIndContext.txt
VP2DimInf vpvp.dat 3 > 3D.txt
VP2DimInfCtx vpvp.dat AddR_data.txt 3 > 3DContext.txt
VP2DimInfCtxInd vpvp.dat AddR_data.txt 3 > 3DIndContext.txt
echo "set term postscript color" > script.plot
echo "set output \"2ev.ps\"" >> script.plot
echo "plot \"2ev.txt\", \"IndContext.txt\"" >> script.plot
echo "set output \"2D.ps\"" >> script.plot
echo "plot \"2D.txt\", \"2DContext.txt\"" >> script.plot
echo "set output \"3D.ps\"" >> script.plot
echo "splot \"3D.txt\", \"3DContext.txt\"" >> script.plot
gnuplot script.plot
FuzzyClustering AddR_data.txt NindContext.txt > Klust.txt
rm -f matrice.dat
rm -f vpvp.dat
```

```c
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include "allocate.h"
#include "filework.h"
#include "out.h"
#define LG 2000

int *Size, COL, sense;


void F2Coord(char *filename){
        long taille;
        char *Tfile;
        char ***T, **B, **V;
        float **C;

        taille = DimDefine(filename);
        Tfile = (char *)malloc (taille * sizeof(char));
        Tfile = file_to_tab(taille, filename);
        Size = TabDimDefine(Tfile);
        T = parse(Tfile, Size);
        free(Tfile);
        C = GetCoord(T, Size);
        B = GetNames(T, Size);
        V = GetNamesCol(T, Size);
        T = deletePointer3(T, Size[0], Size[1], Size[2]);
        if (sense == 1 ) printR(C, Size, COL);
        else if (sense == 2 ) printR2(C, Size, COL);
        else printf("you must specify a reading sense (1 or 2) see readme for further details\n");

}


int main (int argc, char *argv[]){

        char filename[LG];

        strcpy(filename, argv[1]);
        COL = atoi(argv[2]);
        sense = atoi(argv[3]);
        F2Coord(filename);
        return 0;
}
```

```
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include "filework.h"
#include "out.h"
#include "allocate.h"

#define LG 1000
char filename[LG];
char filename2[LG];


char **NBase;
char **NVecteurs;
int **Coord;
int *Size, S;
float **EvCL;

int main (int argc, char *argv[]){

        long nbcar = 0;
        char *tabfile;
        char ***Mtab;
        int i, temp, j;
        strcpy(filename, argv[1]);
        strcpy(filename2, argv[2]);
        nbcar = DimDefine(filename);
        tabfile = file_to_tab(nbcar, filename);
        Size = TabDimDefine(tabfile);
        Mtab = parse(tabfile, Size);
        free(tabfile);
        NBase = GetNames(Mtab, Size);
        NVecteurs = f2(Size[0], Size[2]);
        for (i = 0; i < Size[0]-1; i++){
                strcpy(NVecteurs[i], Mtab[i+1][0]);
        }
        Mtab = deletePointer3(Mtab, Size[0], Size[1], Size[2]);
        nbcar = DimDefine(filename2);
        tabfile = file_to_tab(nbcar, filename2);
        S = TabDimDefine2(tabfile);
        EvCL = parse2(tabfile, S);
        free(tabfile);
        printEvC(NVecteurs, EvCL, Size[0]);
        return 0;
}
```

```
/* filework.c */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "out.h"
#include "filework.h"
#include "allocate.h"


int GNAMELG = 100;
int Dimension[3];
char *filename;

long DimDefine(char *filename){
        FILE * entree;
        char c;
        long i=0;

        entree = fopen(filename, "r");
        while (fread (&c, sizeof(char), 1, entree), ! feof(entree)){
                i++;
        }
        fclose(entree);
        return i;
}

char * file_to_tab(long z, char *filename){
        FILE * entree2;
        char d;
        int ind = 0;

        char *table = (char *)malloc (z * sizeof(char));
        entree2 = fopen(filename, "r");

        while (fread (&d, sizeof(char), 1, entree2), ! feof(entree2)){
                table[ind] = d;
                ind++;
        }
        fclose(entree2);
        return table;
}

int *TabDimDefine(char * TFile){
        int *dim = (int *)malloc (3 * sizeof(int));
        int i = 0, count = 0;
        while (TFile[i] != '\0'){
                if (TFile[i]=='\n') count++;
                i++;
        }
        Dimension[0] = count;
        dim[0]=count;
        i=0;
        count=0;
        while (TFile[i] != '\n'){
                if (TFile[i] == '\t') count++;
                i++;
        }
        Dimension[1] = count;
        dim[1]=count+1;
        Dimension[2] = GNAMELG;
        dim[2]= GNAMELG;
        return dim;
}


char ***parse(char * TabStart, int * dim){
        char ***matr;
        int ind = 0;
        int line = 0;
        int col = 0;
        int tdim = 0;
```

```c
        matr = f3(dim);

        while(line < dim[0]){
                if (TabStart[ind] != '\t' && TabStart[ind] != '\n'){
                        if (TabStart[ind] == ',') matr[line][col][tdim] = '.';
                        else matr[line][col][tdim] = TabStart[ind];
                        tdim++;
                }
                else{
                        if(TabStart[ind] == '\t'){
                                col++;
                                tdim = 0;
                        }
                        if(TabStart[ind] == '\n'){
                                line++;
                                col = 0;
                                tdim = 0;
                        }
                }
                ind++;
        }
        return matr;
}

char **GetNames(char ***M, int * DimTab){
        int i;
        char **Names1;

        Names1 = f2(DimTab[1]-1, DimTab[2]);
        for (i=1; i < DimTab[1]; i++) strcpy(Names1[i-1], M[0][i]);
        return Names1;
}

char **GetNamesCol(char ***M, int * DimTab){
        int i;
        char **Names;

        Names = f2(DimTab[0]-1, DimTab[2]);
        for (i=1; i < DimTab[0]; i++) strcpy(Names[i-1], M[i][0]);
        return Names;
}

float **GetCoord(char ***M, int *DimTab){
        float **coord;
        int i, j;

        coord = f2float(DimTab[0]-1,DimTab[1]-1);
        for (i = 1; i < DimTab[0]; i++){
                for (j = 1; j < DimTab[1]; j++) {
                        coord[i-1][j-1] = atof(M[i][j]);
                }
        }
        return coord;
}

int TabDimDefine2(char * TFile){
        int i = 0, count = 0;

        while (TFile[i] != '\0'){
                if (TFile[i]=='\n') count++;
                i++;
        }
        return count;
}

float **parse2(char * TabStart, int dim){
        char ***matr;
        float **R;
        int ind = 0;
        int line = 0;
        int col = 0;
        int tdim = 0;
        int flag = 0;

        matr = f33(dim,2,20);
        while(line < dim && TabStart[ind]!= '\0'){
                if (TabStart[ind] != '\t' && TabStart[ind] != '\n'){
                        matr[line][col][tdim] = TabStart[ind];
                        tdim++;
                        flag = 1;
                }
                else{
                        if(TabStart[ind] == '\t' && flag != 0){
                                matr[line][col][tdim] = '\0';
                                col++;
                                tdim = 0;
                                flag =0;
                        }
                        if(TabStart[ind] == '\n'){
                                matr[line][col][tdim] = '\0';
                                line++;
                                col = 0;
                                tdim = 0;
                                flag = 0;
                        }
                }
                ind++;
        }
        R = f2float(dim,2);

        for (line = 0; line < dim; line++){
                R[line][0] = atof(matr[line][1]);
                R[line][1] = atof(matr[line][0]);
        }
        matr = deletePointer3(matr, dim, 2, 20);
        return R;

}
```

```
#ifndef _FILEWORK_
#define _FILEWORK_

long DimDefine(char *filename);
char *file_to_tab(long j, char *filename);
int *TabDimDefine(char *t);
char ***parse(char *t, int *d);
char **GetNames(char ***M, int *d);
char **GetNamesCol(char ***M, int *d);
float **GetCoord(char ***M, int *d);
int TabDimDefine2(char * TFile);
float **parse2(char * TabStart, int dim);

#endif
```

```
     OPEN(1,FILE="matrice.dat",
    *FORM="FORMATTED")
     READ (1,*) N
     WRITE(*,*) N
     WRITE(*,*) ('call passe')
     WRITE (*,*) N
     CALL PASSE (N)
     END

     SUBROUTINE PASSE(N)
     INTEGER INFO, LDA,  LW
     REAL A(N, N), VP(N), W(3*N-1)
     CHARACTER*1 JOBZ
     CHARACTER*1 UPLO
     WRITE(*,*) ('debut routine')
     OPEN(2, FILE="vpvp.dat",FORM="UNFORMATTED")
     DO I=1,N
     READ(1,*) (A(I,J), J=1,N)
     ENDDO
     INFO = 1
     LDA = N
     LW = 3*N-1
     JOBZ = 'V'
     UPLO = 'U'
     CALL SSYEV (JOBZ,UPLO,N,A,LDA,VP,W,LW,INFO)
     WRITE (*,*) INFO
     WRITE(2) VP
     WRITE(2) A
1000 FORMAT(1X, 10000(E20.12,1X))
     END
```

```
pwd
pwd >> ../distillation.log
head -1 data.txt | tail -1 >> ../distillation.log
if IsLeaf data.txt
then
directory=$(pwd)
distillation_lite.sh
un=.1
zero=.0
tmp=$directory$zero
mkdir $tmp
multictxrm AddR_data.txt Klust.txt 1 > $tmp/data.txt
cd $tmp
FullDistillation_lite.sh
cd $directory
tmp=$directory$un
mkdir $tmp
multictxrm AddR_data.txt Klust.txt 2 > $tmp/data.txt
cd $tmp
FullDistillation_lite.sh
fi
```

```
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include "filework.h"
#include "out.h"
#include "allocate.h"

#define LG 200
char filename[LG];
char filename2[LG];


char **Attributes, **Objects;
float **Repres3Dk;
int *Size, *S;
float alpha, **membershipk, *v1, *v2;

void GetArgmax(){
        int i, j, k;
        float arg, argmax;

        argmax = 0;
        for (i = Size[0]-Size[1]; i < Size[0]-1; i++){
                arg = 0;
                for (j = i; j < Size[0]-1; j++){
                        for (k = 0; k < S[1]; k++) arg = arg + (float)(pow(Repres3Dk[i][k]-Repres3Dk[j…
                        arg = (float)(sqrt(arg));
                        if (argmax < arg){
                                v1 = Repres3Dk[i];
                                v2 = Repres3Dk[j];
                        }
                }
        }

}

void Getmxk(){
        int i, j;
        float num1, num2;

        for (i = 0; i < Size[0]-1; i++){
                num1 = 0;
                num2 = 0;
                for ( j = 0; j < S[1]; j++){
                        num1 = num1 + (float)(pow(Repres3Dk[i][j]-v1[j], alpha));
                        num2 = num2 + (float)(pow(Repres3Dk[i][j]-v2[j], alpha));
                }
                num1 = (float)(pow(num1, 2/(alpha-1)));
                num2 = (float)(pow(num2, 2/(alpha-1)));
                membershipk[i][0] = num1/(num1+num2);
                membershipk[i][1] = num2/(num2+num1);
        }
}

float Getdis(float *Y, float *Y2, int Siz){
        int k;
        float dis;

        dis = 0;
        for (k=0; k < Siz; k++){
                dis = dis + (float)(pow(Y[k]-Y2[k], 2));
        }
        if(dis != 0) dis = 1/dis;
        else dis = 99999;
        return dis;

}

float GetF(){
        int i;
        float F1, F2, F, dis;
        F = 0;
        F1 = 0;
        F2 = 0;
        for (i = 0; i < Size[0]-1; i++){
                dis = Getdis(Repres3Dk[i], v1, S[1]);
                F1 = F1 + (float)(pow(membershipk[i][0], alpha))*dis;
        }
        for (i = 0; i < Size[0]-1; i++){
                dis = Getdis(Repres3Dk[i], v2, S[1]);
                F2 = F2 + (float)(pow(membershipk[i][1], alpha))*dis;
        }
        F = F1 + F2;
        return F;
}

void GetV(){
        int i, j;
        float tmpnum1, tmpnum2, tmpden1, tmpden2;

        tmpnum1 = 0;
        tmpnum2 = 0;
        tmpden1 = 0;
        tmpden2 = 0;
        for (i = 0; i < S[1]; i++){
                for (j = 0; j< Size[0]-1; j++){
                        tmpnum1 = tmpnum1 + (float)(pow(membershipk[j][0], alpha))*Repres3Dk[j][i];
                        tmpnum2 = tmpnum2 + (float)(pow(membershipk[j][1], alpha))*Repres3Dk[j][i];
                        tmpden1 = tmpden1 + (float)(pow(membershipk[j][0], alpha));
                        tmpden2 = tmpden2 + (float)(pow(membershipk[j][1], alpha));
                }
        v1[i] = tmpnum1/tmpden1;
        v2[i] = tmpnum2/tmpden2;
        }
}

void GetMembership(float SEUIL){
        float F1, F2, F;
        int i, j;

        membershipk = f2float(Size[0], 2);
        v1 = (float *)malloc (S[1] * sizeof(float));
        v2 = (float *)malloc (S[1] * sizeof(float));
        F = 999999;

        GetArgmax();
        Getmxk();
        F1 = GetF();
        while (F > SEUIL){
                GetV();
                Getmxk();
                F2 = F1;
                F1 = GetF();
                F = F2-F1;
        }
}


int main(int argc, char *argv[]){

        long nbcar;
        char *tabfile;
        char ***Mtab;
        float SEUIL;
        int i, j;

        strcpy(filename, argv[1]);
        strcpy(filename2, argv[2]);
        alpha = atof(argv[3]);
        SEUIL = atof(argv[4]);


        nbcar = DimDefine(filename);
        tabfile = file_to_tab(nbcar, filename);
        Size = TabDimDefine(tabfile);/
        Mtab = parse(tabfile, Size);
        free(tabfile);
        Attributes = GetNames(Mtab, Size);
        Objects = GetNamesCol(Mtab, Size);
        Mtab = deletePointer3(Mtab, Size[0], Size[1], Size[2]);
        nbcar = DimDefine(filename2);
        tabfile = file_to_tab(nbcar, filename2);
        S = TabDimDefine2(tabfile);
        Mtab = parse2(tabfile, S);
        free(tabfile);
        Repres3Dk = GetCoord2(Mtab, S);
        Mtab = deletePointer3(Mtab, S[0], S[1], S[2]);
        GetMembership(SEUIL);
        printMembership(Objects, membershipk, Size);

        return 0;
}
```

```
/* MAdjSem.c */

#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include "allocate.h"
#include "filework.h"
#include "out.h"
#include "vectcomp.h"
#define LG 100

int *Size;
float **K;

float VertexCount(float **G, int i){
        int j;
        float R;

        R = 0;
        for (j = 0; j < Size[0]-1; j++) R = R + G[i][j];
        return R;
}

void Laplacian(float **G){
        float Vertex;
        int i, j;

        for (i = 0; i < Size[0]-1; i++){
                Vertex = VertexCount(G,i);
                for(j = 0; j < Size[0]-1; j++){
                        if (i == j) G[i][j] = Vertex;
                        else G[i][j] = -(G[i][j]);
                }
        }
}

int **Sim2Adj(float **Cos, float SEUIL){
        int i, j;
        int **TEMP;

        TEMP = f2int(Size[0], Size[0]);
        for(i = 0; i < Size[0]-1; i++){
                for(j = 0; j <= i; j++){
                        if (Cos[i][j] > SEUIL)  TEMP[i][j] = 1;
                        else TEMP[i][j] = 0;
                        TEMP[j][i] = TEMP[i][j];
                }
                TEMP[i][i] = 0;
        }

        return TEMP;
}

int main (int argc, char *argv[]){

        char filename[LG];
        long taille;
        char *Tfile;
        char **Names;
        char ***T;
        float **C;

        strcpy(filename, argv[1]);
        taille = DimDefine(filename);
        Tfile = file_to_tab(taille, filename);
        Size = TabDimDefine(Tfile);
        T = f3(Size);
        T = parse(Tfile, Size);
        free(Tfile);
        C = GetCoord(T, Size);
        Names = GetNamesCol(T, Size);
        T = deletePointer3(T, Size[0], Size[1], Size[2]);
        K = CosCompute(C, Size);
        C = deletePointer2float(C, Size[0]-1, Size[1]-1);
        Laplacian(K);
        printMSci2(K, Size);
        return 0;
}
```

```c
/* MAdjSem.c */

#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include "allocate.h"
#include "filework.h"
#include "out.h"
#define LG 100

int *Size;

int main (int argc, char *argv[]){

        char filename[LG];
        long taille;
        char *Tfile;

        strcpy(filename, argv[1]);

        taille = DimDefine(filename);
        Tfile = file_to_tab(taille, filename);
        Size = TabDimDefine(Tfile);
        /*printf("taille =  %d\n", Size[1]);*/
        if (Size[1] < 3){
                printf("%s is leaf\n", filename);
                return 1;
        }
        else{
                printf("%s is not leaf\n", filename);
                return 0;
        }
}
```

```c
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include "allocate.h"
#include "filework.h"
#include "out.h"
#define LG 2000

int *Size, COL, sense;


void F2Coord(char *filename){
        long taille;
        char *Tfile;
        char ***T, **B, **V;
        float **C;
        int i,j;

        taille = DimDefine(filename);
        Tfile = (char *)malloc (taille * sizeof(char));
        Tfile = file_to_tab(taille, filename);
        Size = TabDimDefine(Tfile);
        T = parse(Tfile, Size);
        free(Tfile);
        for (i = Size[0]-1; i >= 0; i--){
                for (j = 0; j < Size[1]; j++){
                        printf("%s",T[i][j]);
                        if (j < Size[0]-2) printf("\t");
                }
                printf("\n");
        }

}


int main (int argc, char *argv[]){

        char filename[LG];

        strcpy(filename, argv[1]);
        F2Coord(filename);
        return 0;
}
```

```c
/* MAdjSem.c */

#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include "allocate.h"
#include "filework.h"
#include "out.h"
#include "vectcomp.h"
#define LG 100

int *Size, *S;

int main (int argc, char *argv[]){

        char filename[LG], filename2[LG];
        long taille;
        char *Tfile;
        char **Names, **Contexts;
        char ***T;
        float **C;
        float **L;
        int COL;

        strcpy(filename, argv[1]);
        strcpy(filename2, argv[2]);
        COL = atoi(argv[3]);
        taille = DimDefine(filename);
        Tfile = file_to_tab(taille, filename);
        Size = TabDimDefine(Tfile);
        T = parse(Tfile, Size);
        free(Tfile);
        C = GetCoord(T, Size);
        Names = GetNamesCol(T, Size);
        Contexts = GetNames(T, Size);
        T = deletePointer3(T, Size[0], Size[1], Size[2]);
        taille = DimDefine(filename2);
        Tfile = file_to_tab(taille, filename2);
        S = TabDimDefine(Tfile);
        T = f3(Size);
        T = parse(Tfile, S);
        free(Tfile);
        printctxrm(Contexts, Names, C, Size, T, S, COL);
        return 0;
}
```

```c
/* out.c */
#include <stdio.h>
#include <stdlib.h>
#include "out.h"
#include "allocate.h"
#include <string.h>


void printR(char **Base, char ** Vecteurs, int **Coord, int *Dim){
        int i, j;

        printf("\t");
        for (i = 0; i < Dim[1]-1; i++) {
                printch(Base[i]);
                printf("\t");
        }
        printf("\n");
        for (i = 0; i < Dim[0]-1; i++){
                printch(Vecteurs[i]);
                printf("\t");
                for (j = 0; j < Dim[1]-1; j++){
                        printf("%d\t", Coord[i][j]);
                }
                printf("\n");
        }
}


void printctxrm(char **Contexts, char **Names, float **Coord, int *Size, char ***T, int *S, int COL){
        int i, j, z;
        int flag[Size[1]];

        z = 0;
        for (i = Size[0]-Size[1]+1; i < Size[0]; i++){
                if (atof(T[i][COL]) >= 0.5){
                        printf("\t%s", T[i][0]);
                        flag[z] = 1;
                }
                else flag[z] = 0;
                z++;
        }
        printf("\n");
        for (i = 0; i < Size[0]-Size[1]; i++){
                printf("%s", Names[i]);
                for(j = 0; j < Size[1]-1; j++){
                        if (flag[j] != 0) printf("\t%f", Coord[i][j]);
                }
                printf("\n");
        }

}

void printMSci2(float **M, int *Size){
        int i, j;

        printf("%d\n", Size[0]-1);
        for(i = 0; i < Size[0]-1; i++){
                        for(j = 0; j < Size[0]-1; j++){
                                        printf("%f", M[i][j]);
                                if (j < Size[0]-2) printf(" ");
                        }
                        printf("\n");
        }
}

void printEvC(char **NBase, float **EvCL, int S){
        int i;

        for(i = 0; i < S-1; i++){
                printf("%s\t%d\t%.10f\n",NBase[((int)(EvCL[i][1]))],(int)(EvCL[i][1]), EvCL[i][0]);
        }
}
```

```
void printMembership(char **Objetcs, float **membershipk, int *Size){
        int i;

        printf("\tK1\tK2\n");
        for (i = 0; i< Size[0]-1; i++){
                printf("%s\t%f\t%f\n", Objetcs[i], membershipk[i][0], membershipk[i][1]);
        }
}

void printAnnotation(char **NVecteurs, char ***Mtab, int *Size, int * S){
        int i , j, k;
        for (i = 0; i < Size[0]-1; i++){
                for (j = 0; j < S[0]-1; j++){
                        if (strcmp(NVecteurs[i], Mtab[j][0]) == 0) {
                                for (k = 0; k < S[1]-1; k++) printf("%s\t",Mtab[j][k]);
                                printf("\n");
                        }
                }
        }
}

void printR1(float **Coord, int *Size, int COL){
        int i, j, count, tmp;

        count = 0;
        for (i = 0; i < Size[0]-1; i++){
                tmp = 0;
                for(j = 0; j < Size[1]; j++)if(Coord[i][COL] < Coord[i][j]){
                        tmp++;
                }
                if (tmp == 0) count++;
                printf("%d\t%d\n", i, count);
        }
}

void printR2(float **Coord, int *Size, int COL){
        int i, j, count, tmp;

        count = 0;
        for (i = Size[0]-2; i > 0; i--){
                tmp = 0;
                for(j = 0; j < Size[1]; j++)if(Coord[i][j] > Coord[i][COL]) tmp++;
                if (tmp == 0) count++;
                printf("%d\t%d\n", Size[0]-1-i, count);
        }
}
```

```c
#ifndef _OUT_
#define _OUT_

void printR(char **Base, char ** Vecteurs, int **Coord, int *Dim);
void printctxrm(char **Contexts, char **Names, float **Coord, int *Size, char ***T, int *S, int COL);
void printMSci2(float **M, int *Size);
void printEvC(char **NBase, float **EvCL, int S);
void printMembership(char **Objetcs, float **membershipk, int *Size);
void printAnnotation(char **NVecteurs, char ***Mtab, int *Size, int *S);
void printR1(float **Coord, int *Size, int COL);
void printR2(float **Coord, int *Size, int COL);

#endif
```

```c
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include "filework.h"
#include "out.h"
#include "allocate.h"
#include "tri.h"

#define LG 100
char filename[LG];
char filename2[LG];


char **NBase;
char **NVecteurs, **Names;
float **Coord;
int *S;
int *Size, *Index;
float **EvCL;

char ***GetNindCtx(char ***Mtab2){
        char ***NindCtx;
        int i, j, k;

        NindCtx = f33(Size[1], 3, Size[2]);
        Index = (int *)malloc (Size[1] * sizeof(int));
        for (i = 0; i < Size[1]-1; i++){
                for (j = 0; j < S[0]; j++){
                        if (strcmp(NBase[i], Mtab2[j][0])==0){
                                for(k = 0; k < 3; k++) NindCtx[i][k] = Mtab2[j][k];
                                Index[i] = j;
                        }
                }
        }
        return NindCtx;
}


float **NindSort(char ***NindCtx){
        int i, j;
        float **fctx, *tmp;

        tmp = (float *)malloc (Size[1] * sizeof(float));
        for (i = 0; i < Size[1]-1; i++){
                tmp[i] = atof(NindCtx[i][2]);
        }
        fctx = quicksort(tmp, Size[1]-1);

        return fctx;
}


int main (int argc, char *argv[]){

        long nbcar = 0;
        char *tabfile;
        char ***Mtab, ***Mtab2, ***NindCtx;
        int i, temp, j, k;
        float **NSort;
        strcpy(filename, argv[1]);
        strcpy(filename2, argv[2]);

        nbcar = DimDefine(filename);
        tabfile = file_to_tab(nbcar, filename);
        Size = TabDimDefine(tabfile);
        Mtab = parse(tabfile, Size);
        free(tabfile);
        NBase = GetNames(Mtab, Size);
        nbcar = DimDefine(filename2);
        tabfile = file_to_tab(nbcar, filename2);
        S = TabDimDefine(tabfile);
        Mtab2 = parse(tabfile, S);
        free(tabfile);
        NindCtx = GetNindCtx(Mtab2);
        NSort = NindSort(NindCtx);/
        PrintNindContext(NindCtx, NSort, Size, S, Index);
        return 0;
}
```

```c
/* EvList */

#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include "filework.h"
#include "out.h"
#include "allocate.h"
#include "tri.h"

#define LG 100
char filename[LG];
char filename2[LG];


char **NBase;
char **NVecteurs, **Names;
float **Coord;
int *S;
int *Size, *Index;
float **EvCL;

int main (int argc, char *argv[]){

        long nbcar = 0;
        char *tabfile;
        char ***Mtab, ***Mtab2, ***NindCtx;
        int i, temp, j, k;
        float **NSort;
        strcpy(filename, argv[1]);
        strcpy(filename2, argv[2]);

        nbcar = DimDefine(filename);
        tabfile = file_to_tab(nbcar, filename);
        Size = TabDimDefine(tabfile);
        Mtab = parse(tabfile, Size);
        free(tabfile);
        nbcar = DimDefine(filename2);
        tabfile = file_to_tab(nbcar, filename2);
        S = TabDimDefine(tabfile);
        Mtab2 = parse(tabfile, S);

        for (i = 0; i < S[0]; i++){
                for (j = Size[0]-Size[1]+1; j < Size[0]; j++){
                        if (strcmp(Mtab2[i][0], Mtab[j][0])==0) printf("%s\t%s\t%s\n", Mtab2[i][0], Mt…
                }
        }
        return 0;
}
```

```c
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include "filework.h"
#include "out.h"
#include "allocate.h"

#define LG 1000
char filename[LG];
char filename2[LG];


char **NBase, **NVecteurs, **NVecteurs2;
float **Coord;
int *Size, *S;

int main (int argc, char *argv[]){

        long nbcar, nbcar2;
        char *tabfile, *tabfile2;
        char ***Mtab, ***Mtab2;
        int i, j, temp;

        strcpy(filename, argv[1]);
        strcpy(filename2, argv[2]);
        nbcar = DimDefine(filename);
        tabfile = file_to_tab(nbcar, filename);
        Size = TabDimDefine(tabfile);
        Mtab = parse(tabfile, Size);
        free(tabfile);
        printf("#");
        NBase = GetNames(Mtab, Size);
        for (i = 0; i < Size[1]-1; i++){
                printf("%s",NBase[i]);
                if (i < Size[1]-2) printf("\t");
        }
        printf("\n");
        NVecteurs = GetNamesCol(Mtab, Size);
        Coord = GetCoord(Mtab, Size);
        Mtab = deletePointer3(Mtab, Size[0], Size[1], Size[2]);

        nbcar2 = DimDefine(filename2);
        tabfile2 = file_to_tab(nbcar2, filename2);
        S = TabDimDefine(tabfile2);
        Mtab2 = parse2(tabfile2, S, nbcar2);
        NVecteurs2 = GetNamesCol2(Mtab2, S);
        printSortProfile(NVecteurs2, NVecteurs, NBase, Coord, Size, S);
}
```

```
#! /bin/bash

i=1
un=1
deux=2
dix=10
cent=100
max=`cat distillation.log | wc -l`

#crée un nouveau fichier .log et efface le contenu de celui eventuellement présent
echo "" > result_analysis.log
mkdir ROC_total && mkdir ANNOTATIONS
echo -n "0% done"

while read line
        do

          #teste le numéro de ligne (paire ou impaire)
          if test $(($i%$deux)) -eq 1
          then
                #stocke la ligne contenant le pathway
                pathway=$line

                #supprime les éventuels " des fichiers necéssaires à l'analyse
                cp ${pathway}/data.txt ${pathway}/data2.txt && sed -e "s/\"//g" ${pathway}/data2.txt >
${pathway}/data.txt && rm ${pathway}/data2.txt


          else
            echo
"***************************************************************************************" >>
result_analysis.log

            #compte le nombre de mots dans la ligne lue
            nb_ctx=`echo $line | wc -w`
            echo "context number: "$nb_ctx >> result_analysis.log
            if test $nb_ctx -eq 1
              then

                 #path_length prend la longueur de la chaine de caractère pathway
                 path_lenght=${#pathway}
                 echo "path_length: "$path_lenght >> result_analysis.log

                 #path_length prend la longueur de la chaine de caractère pathway moins deux caractères
                 path_lenght=$(($path_lenght-$deux))

                 #pathway est tronquée des ses deux caractères finaux
                 pathway=${pathway:0:$path_lenght}
                 echo "previous pathway: "$pathway >> result_analysis.log

                 #supprime les éventuels " des fichiers necéssaires à l'analyse
                 cp ${pathway}/Nind2ev.txt ${pathway}/Nind2ev2.txt && sed -e "s/\"//g" ${pathway}/
Nind2ev2.txt > ${pathway}/Nind2ev.txt && rm ${pathway}/Nind2ev2.txt

                 #all_ctx prends la liste des contextes (première ligne de data.txt)
                 all_ctx=`head -1 step0/data.txt`
                 local_ctx=`head -1 ${pathway}/data.txt`
                 echo "all contexts: "$all_ctx >> result_analysis.log
                 echo "local contexts: "$local_ctx >> result_analysis.log

                 #tmp prends la chaîne de caractère la plus longue avant $line
                 tmp=${all_ctx%$line*}
                 local_tmp=${local_ctx%$line*}

                 #comptage du nombre de mots dans $tmp qui nous donne l'indice de la colonne du
contexte en question
                 id_column=`echo $tmp | wc -w`
                 id_local_column=`echo $local_tmp | wc -w`
                 echo "researched context: "$line >> result_analysis.log
                 echo "id of researched context: "$id_column >> result_analysis.log
                 echo "local id: "$id_local_column >> result_analysis.log
                #comptage du nombre d'objets (lignes dans l'analyse)
                limit=`cat ${pathway}/Nind2ev.txt | wc -l`
                half_limit=$(($limit/$deux))
                echo "limit for reading sense: "$half_limit >> result_analysis.log

                #recherche du contexte étudié parmi les objets pos_ctx prends la chaine la plus longue avant le caractère ":"
                tmp=`grep -n "$line" ${pathway}/Nind2ev.txt`
                pos_ctx=${tmp%\:*}
                echo "researched context ranking: "$pos_ctx >> result_analysis.log
                if test $pos_ctx -gt $half_limit
                then
                  sense=2
                else
                  sense=1
                fi
                echo "reading sense: "$sense >> result_analysis.log

                #série de calcul necessaires à l'analyse
                Annotate ${pathway}/Nind2ev.txt annotation.txt > ${pathway}/annotation_${line}.txt
                ProfileSortNo_Ind step0/data.txt ${pathway}/Nind2ev.txt > ${pathway}/total_${line}.txt
                ProfileSortNo_Ind ${pathway}/data.txt ${pathway}/Nind2ev.txt > ${pathway}/local_total_${line}.txt
                ROCcurveSpe ${pathway}/total_${line}.txt $id_column $sense > ${pathway}/ROC_${line}.txt
                ROCcurveSpe ${pathway}/local_total_${line}.txt $id_local_column $sense > ${pathway}/local_ROC_${line}.txt


                #nb_g_spe prend la dernière ligne du fichier ROC.txt qui sera remplacé par la chaine la plus longue presente apres une tabulation
                nb_g_spe=`tail -1 ${pathway}/ROC_${line}.txt`
                nb_g_spe=${nb_g_spe##* }
                echo "number of specific genes: "$nb_g_spe >> result_analysis.log
                DivisiveSpeROCCheck ${pathway}/total_${line}.txt $id_column $sense > ${pathway}/spe_${line}.txt
                DivisiveSpeROCCheck ${pathway}/local_total_${line}.txt $id_local_column $sense > ${pathway}/local_spe_${line}.txt
                if test $sense -eq $un
                then
                    head -${nb_g_spe} ${pathway}/Nind2ev.txt > ${pathway}/SPE_${line}.txt && mv ${pathway}/SPE_${line}.txt ANNOTATIONS/SPE_${line}.txt
                    #head -${nb_g_spe} ${pathway}/annotation.txt > ${pathway}/ANNOT_SPE_${line}.txt && mv ${pathway}/ANNOT_SPE_${line}.txt ANNOTATIONS/ANNOT_SPE_${line}.txt
                fi
                if test $sense -eq $deux
                then
                    tail -${nb_g_spe} ${pathway}/Nind2ev.txt > ${pathway}/2_SPE_${line}.txt && Mreturn ${pathway}/2_SPE_${line}.txt > ${pathway}/SPE_${line}.txt && rm ${pathway}/2_SPE_${line}.txt && mv ${pathway}/SPE_${line}.txt ANNOTATIONS/SPE_${line}.txt
                    #tail -${nb_g_spe} ${pathway}/annotation.txt > ${pathway}/2ANNOT_SPE_${line}.txt && Mreturn ${pathway}/2ANNOT_SPE_${line}.txt > ${pathway}/ANNOT_SPE_${line}.txt && rm ${pathway}/2ANNOT_SPE_${line}.txt && mv ${pathway}/ANNOT_SPE_${line}.txt ANNOTATIONS/ANNOT_SPE_${line}.txt
                fi



                #création du script gnuplot
                echo -e "set term postscript color\nset output \"${pathway}/ROC_${line}.ps\"\nf(x)=x\ng(x)=$nb_g_spe/$limit*x\nset yrange[0:$nb_g_spe]\nplot \"${pathway}/ROC_${line}.txt\",f(x),g(x)\nreset\nset logscale y\nset output \"${pathway}/spe_${line}.ps\"\nplot \"${pathway}/spe_${line}.txt\"" > ${pathway}/ROC_${line}.plot
                echo -e "set term postscript color\nset output \"${pathway}/local_ROC_${line}.ps\"\nplot \"${pathway}/local_ROC_${line}.txt\"\n" > ${pathway}/local_ROC_${line}.plot

                #execution du script gnuplot et fin des calculs
                gnuplot ${pathway}/ROC_${line}.plot
                gnuplot ${pathway}/local_ROC_${line}.plot
                cp ${pathway}/ROC_${line}.ps ROC_total/ROC_${line}.ps
                cp ${pathway}/spe_${line}.ps ROC_total/spe_${line}.ps
                cp ${pathway}/annotation_${line}.txt ANNOTATIONS/annotation_${line}.txt
                rm ${pathway}/ROC_${line}.plot
```

```
                 rm ${pathway}/local_ROC_${line}.plot
                 rm ${pathway}/ROC_${line}.txt
                 rm ${pathway}/local_ROC_${line}.txt
                 rm ${pathway}/totalzoom_${line}.txt

                 #conversion en pdf pour les windoziens ;-)
                 ps2pdf ROC_total/ROC_${line}.ps ROC_total/ROC_${line}.pdf
                 ps2pdf ROC_total/spe_${line}.ps ROC_total/spe_${line}.pdf

             fi
        fi

        #affichage de la progression du script
        bar=$(($cent*$i/$max))
        if test $bar -lt $dix
        then echo -ne "\b\b\b\b\b\b\b\b\b\b"$bar"% done"
        else if test $bar -eq $cent
        then echo -ne "\b\b\b\b\b\b\b\b\b\b\b\b"$bar"% done"
        else echo -ne "\b\b\b\b\b\b\b\b\b\b\b"$bar"% done"
        fi
        fi

        #incrémentation du compteur de la boucle
        i=$(($i+$un))

        #instruction d'arrêt de boucle
        if test $i -gt $max
        then break
        fi

        done < distillation.log

echo ""
cd ROC_total
psmerge -oROCtotal.ps ROC_*.ps
psmerge -ospetotal.ps spe*.ps
ps2pdf ROCtotal.ps ROCtotal.pdf
ps2pdf spetotal.ps spetotal.pdf
echo "finished! see log file for more information"
```

```c
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include "allocate.h"
#include "filework.h"
#include "out.h"
#define LG 2000

int *Size, COL, sense;


void F2Coord(char *filename){
        long taille;
        char *Tfile;
        char ***T, **B, **V;
        float **C;
        int i;


        taille = DimDefine(filename);
        Tfile = (char *)malloc (taille * sizeof(char));
        Tfile = file_to_tab(taille, filename);
        Size = TabDimDefine(Tfile);
        T = parse(Tfile, Size);
        free(Tfile);
        C = GetCoord(T, Size);
        B = GetNames(T, Size);
        V = GetNamesCol(T, Size);
        T = deletePointer3(T, Size[0], Size[1], Size[2]);
        if (sense == 1 ) printR(C, Size, COL);
        else if (sense == 2 ) printR2(C, Size, COL);
        else printf("you must specify a reading sense (1 or 2) see readme for further details\n");

}


int main (int argc, char *argv[]){

        char filename[LG];

        strcpy(filename, argv[1]);
        COL = atoi(argv[2]);
        sense = atoi(argv[3]);
        F2Coord(filename);
        return 0;
}
```

```c
/* tri.c */

#include <stdio.h>
#include "allocate.h"
#include "tri.h"
#include <stdlib.h>

float **Resultat;

int rapideEtape(float **t, int min, int max){
        float temp = t[1][max];
        float temp0 = t[0][max];

        while(max > min){
                while(max > min && t[1][min] <= temp) min++;
                if(max > min){
                        t[1][max] = t[1][min];
                        t[0][max] = t[0][min];
                        max--;
                        while(max > min && t[1][max] >= temp) max--;
                        if (max > min){
                                t[1][min] = t[1][max];
                                t[0][min] = t[0][max];
                                min++;
                        }
                }
        }
        t[1][max] = temp;
        t[0][max] = temp0;
        return max;
}

void rapide (float **t, int deb, int fin){
        int mil;

        if (deb < fin){
                mil = rapideEtape(t, deb, fin);
                if (mil - deb > fin - mil){
                        rapide(t, mil+1, fin);
                        rapide(t, deb, mil-1);
                }
                else{
                        rapide(t, deb, mil-1);
                        rapide(t, mil+1, fin);
                }
        }
}

float **quicksort(float *tab, int Dim){
        int i;
        float **Resultat;

        Resultat = f2float(2, Dim);

        for (i = 0; i < Dim; i++){
                Resultat[0][i] = i;
                Resultat[1][i] = tab[i];
        }
        rapide(Resultat, 0, Dim-1);
        return Resultat;
}


int *SEV(int *res, float **R, int Size, int NBDIM){
        int i, j;


        i = 1;
        while(R[1][i+1] == R[1][i] && i < Size){
                i++;
        }
        for (j = 0; j < NBDIM; j++){
                res[j] = R[0][i];
                i++;
        }
        return res;
}
```

```
#ifndef _TRI_
#define _TRI_

int rapideEtape(float **t, int min, int max);
void rapide (float **t, int deb, int fin);
float **quicksort(float *tab, int Dim);
int *SEV(int *sev, float **R, int Size, int NBDIM);

#endif
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "allocate.h"
#include "vectcomp.h"


float GetNorm(float *Coord, int *Dimension){
        float result = 0;
        float temp = 0;
        int i = 0;
        int siz = Dimension[1];


        for (i = 0; i < siz-1; i++){
                temp = temp + (Coord[i]*Coord[i]);
        }
        result = sqrt(temp);
        return result;
}

float GetSProduct(float *V1, float *V2, int *Dimension){
        int i = 0;
        float result = 0;
        int siz = Dimension[1];

        for (i = 0; i < siz-1; i++){
                result = result + (V1[i]*V2[i]);
        }
        return result;
}

float **CosCompute(float **Coord, int *Dimension){
        int line = Dimension[0]-1;
        int i,j;
        float Scalaire;
        float NormV[line];
        float **Cosinus;
        Cosinus = f2float(Dimension[0],Dimension[0]);

        for (i = 0; i < line; i++){
                NormV[i] = GetNorm(Coord[i], Dimension);
        }

        for (i = 0; i < line; i++){
                for (j = i; j < line; j++){
                        if (i == j) Cosinus[i][j] = 0;
                        else{
                                Scalaire = GetSProduct(Coord[i], Coord[j], Dimension);
                                if((NormV[i])*(NormV[j])!=0) Cosinus[i][j] = (float)fabs((Scalaire)/((…
                                else Cosinus[i][j] = 0;
                                Cosinus[j][i] = Cosinus[i][j];
                        }
                }
        }
        return Cosinus;
}
```

```
#ifndef _VECTCOMP_
#define _VECTCOMP_

float GetNorm(float *Coord, int *Dimension);
float GetSProduct(float *V1, float *V2, int *Dimension);
float **CosCompute(float **Coord, int *Dimension);

#endif
```

```
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include "allocate.h"
#include "filework.h"
#include "vectcomp.h"
#include "out.h"
#include "tri.h"
#define LG 100

int *Size, *S;
float *ValP, **VectP;
int NBDIM;

void GetValP(char *filename){
        int i;
        int *a;
        FILE *fp;

        fp = fopen(filename, "rb");
        a = malloc(2*sizeof(int));
        fread(a, sizeof(int), 2, fp);
        Size[0] = (a[0]/4)+(a[1]*32767);
        if (a[1]!=0) printf("verifier le VP, dimension de la matrice elevée\n");
        ValP = (float *)malloc((Size[0])*sizeof(float));
        fread(ValP, sizeof(float), Size[0], fp);
        fread(a, sizeof(int), 2, fp);
        fread(a, sizeof(int), 2, fp);
        VectP = f2float(Size[0], Size[0]);
        for (i = 0; i < Size[0]; i++) fread(VectP[i], sizeof(float), Size[0], fp);
        fread(a, sizeof(int), 2, fp);
        free(a);
        fclose(fp);
}


void **GetMtrx(char *filename2){
        long taille;
        char *Tfile;

        taille = DimDefine(filename2);
        Tfile = file_to_tab(taille, filename2);
        S = TabDimDefine(Tfile);
        free(Tfile);
}

void F2Coord(int NBDIM){
        int i, j;

        if (ValP[0] == ValP[1]) printf("ATTENTION VALEURS PROPRES DEGENEREREES!\n");
        else{
                for (j = Size[0]-S[1]; j < Size[0]; j++){
                        printf("%d\t", j);
                        for (i = 1; i <= NBDIM; i++){
                                printf("%.12f", VectP[i][j]);
                                if (i < NBDIM) printf("\t");
                        }
                        printf("\n");
                }
        }
}



int main (int argc, char *argv[]){

        char filename[LG];
        char filename2[LG];
        float SEUIL;

        Size = (int*) malloc(3 * sizeof(int));
        strcpy(filename, argv[1]);
        strcpy(filename2, argv[2]);
        NBDIM = atoi(argv[3]);
        GetMtrx(filename2);
        GetValP(filename);
        F2Coord(NBDIM);
        return 0;
}
```

```c
/* SpecSort.c */

#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include "allocate.h"
#include "filework.h"
#include "vectcomp.h"
#include "out.h"
#include "tri.h"
#define LG 100

int *Size, *S;
float *ValP, **VectP;
int NBDIM;

void GetValP(char *filename){
        int i;
        int *a;
        FILE *fp;

        fp = fopen(filename, "rb");
        a = malloc(2*sizeof(int));
        fread(a, sizeof(int), 2, fp);
        Size[0] = (a[0]/4)+(a[1]*32767);
        if (a[1]!=0) printf("verifier le VP, dimension de la matrice elevée\n");
        ValP = (float *)malloc((Size[0])*sizeof(float));
        fread(ValP, sizeof(float), Size[0], fp);
        fread(a, sizeof(int), 2, fp);
        fread(a, sizeof(int), 2, fp);
        VectP = f2float(Size[0], Size[0]);
        for (i = 0; i < Size[0]; i++) fread(VectP[i], sizeof(float), Size[0], fp);
        fread(a, sizeof(int), 2, fp);
        free(a);
        fclose(fp);
}


void **GetMtrx(char *filename2){
        long taille;
        char *Tfile;

        taille = DimDefine(filename2);
        Tfile = file_to_tab(taille, filename2);
        S = TabDimDefine(Tfile);
        free(Tfile);
}

void F2Coord(int NBDIM){
        int i, j;

        if (ValP[0] == ValP[1]) printf("ATTENTION VALEURS PROPRES DEGENEREREES!\n");
        else{
                for (j = Size[0]-S[1]; j < Size[0]; j++){
                        for (i = 1; i <= NBDIM; i++){
                                printf("%.12f", VectP[i][j]);
                                if (i < NBDIM) printf("\t");
                        }
                        printf("\n");
                }
        }
}



int main (int argc, char *argv[]){

        char filename[LG];
        char filename2[LG];
        float SEUIL;
        Size = (int*) malloc(3 * sizeof(int));
        strcpy(filename, argv[1]);
        strcpy(filename2, argv[2]);
        NBDIM = atoi(argv[3]);
        GetMtrx(filename2);
        GetValP(filename);
        F2Coord(NBDIM);
        return 0;
}
```

```c
/* SpecSort.c */

#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include "allocate.h"
#include "filework.h"
#include "vectcomp.h"
#include "out.h"
#include "tri.h"
#define LG 100

int *Size;
float *ValP, **VectP;


void GetValP(char *filename){
        int i;
        int *a;
        FILE *fp;

        fp = fopen(filename, "rb");
        a = malloc(2*sizeof(int));
        fread(a, sizeof(int), 2, fp);
        Size[0] = (a[0]/4)+(a[1]*32767);
        if (a[1]!=0) printf("verifier le VP, dimension de la matrice elevée\n");
        ValP = (float *)malloc((Size[0])*sizeof(float));
        fread(ValP, sizeof(float), Size[0], fp);
        fread(a, sizeof(int), 2, fp);
        fread(a, sizeof(int), 2, fp);
        VectP = f2float(Size[0], Size[0]);
        for (i = 0; i < Size[0]; i++) fread(VectP[i], sizeof(float), Size[0], fp);
        fread(a, sizeof(int), 2, fp);
        free(a);
        fclose(fp);
}

void F2Coord(int NBDIM){
        int i, j;

        if (ValP[0] == ValP[1]) printf("ATTENTION VALEURS PROPRES DEGENEREREES!\n");
        else{
                for (j = 0; j < Size[0]; j++){
                        for (i = 1; i <= NBDIM; i++){
                                printf("%.12f", VectP[i][j]);
                                if (i < NBDIM) printf("\t");
                        }
                        printf("\n");
                }
        }
}

void F2Coord2(int NBDIM){
        int i;
        float **VPtrie;

        if (ValP[0] == ValP[1]) printf("ATTENTION VALEURS PROPRES DEGENEREREES!\n");
        else{
                VPtrie = quicksort(VectP[1], Size[0]);
                for (i = 0; i < Size[0]; i++) printf("%d\t%.12f\n", (int)VPtrie[0][i], VPtrie[1][i]);
        }
}

int main (int argc, char *argv[]){

        char filename[LG];
        int NBDIM;

        Size = (int*) malloc(3 * sizeof(int));
        strcpy(filename, argv[1]);
        NBDIM = atoi(argv[2]);
        GetValP(filename);
        if (NBDIM == 1) F2Coord2(NBDIM);
        else F2Coord(NBDIM);

        return 0;
}
```

```
/* SpecSort.c */

#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include "allocate.h"
#include "filework.h"
#include "vectcomp.h"
#include "out.h"
#include "tri.h"
#define LG 100

int *Size;
float *ValP, **VectP;


void GetValP(char *filename){
        int i;
        int *a;
        FILE *fp;

        fp = fopen(filename, "rb");
        a = malloc(2*sizeof(int));
        fread(a, sizeof(int), 2, fp);
        Size[0] = (a[0]/4)+(a[1]*32767);
        if (a[1]!=0) printf("verifier le VP, dimension de la matrice elevée\n");
        ValP = (float *)malloc((Size[0])*sizeof(float));
        fread(ValP, sizeof(float), Size[0], fp);
        fread(a, sizeof(int), 2, fp);
        fread(a, sizeof(int), 2, fp);
        VectP = f2float(Size[0], Size[0]);
        for (i = 0; i < Size[0]; i++) fread(VectP[i], sizeof(float), Size[0], fp);
        fread(a, sizeof(int), 2, fp);
        free(a);
        fclose(fp);
}

void F2Coord(int NBDIM){
        int i, j;

        if (ValP[0] == ValP[1]) printf("ATTENTION VALEURS PROPRES DEGENEREREES!\n");
        else{
                for (j = 0; j < Size[0]; j++){
                        for (i = 1; i <= NBDIM; i++){
                                printf("%.12f", VectP[i][j]);
                                if (i < NBDIM) printf("\t");
                        }
                        printf("\n");
                }
        }
}

void F2Coord2(int NBDIM){
        int i;
        float **VPtrie;

        if (ValP[0] == ValP[1]) printf("ATTENTION VALEURS PROPRES DEGENEREREES!\n");
        else{
                VPtrie = quicksort(VectP[1], Size[0]);
                for (i = 0; i < Size[0]; i++) printf("%.12f\n", VPtrie[1][i]);
        }
}

int main (int argc, char *argv[]){

        char filename[LG];
        int NBDIM;

        Size = (int*) malloc(3 * sizeof(int));
        strcpy(filename, argv[1]);
        NBDIM = atoi(argv[2]);
        GetValP(filename);
        if (NBDIM == 1) F2Coord2(NBDIM);
        else F2Coord(NBDIM);

        return 0;
}
```

Bibliography

[1] Diederik Aerts. Quantum interference and superposition in cognition: Development of a theory for the disjunction of concepts, 2007. 66

[2] Robert Alicki and Mark Fannes. *Quantum dynamical systems*. Oxford University Press, Oxford, 2001. 82

[3] T. W. Anderson. *An introduction to multivariate statistical analysis*. Wiley Series in Probability and Statistics. Wiley-Interscience [John Wiley & Sons], Hoboken, NJ, third edition, 2003. 23

[4] N. Aronszajn. Theory of reproducing kernels. *Trans. Amer. Math. Soc.*, 68:337–404, 1950. 23

[5] Ricardo Baeza-Yates. Information retrieval in the web: beyond current search engines. *Internat. J. Approx. Reason.*, 34(2-3):97–104, 2003. Soft computing applications to intelligent information retrieval on the Internet (Mérida/Granada, 2002). 23

[6] Pierre Baldi, Paolo Frasconi, and Padhraic Smyth. *Modeling the Internet and the Web: Probabilistic Methods and Algorithms*. Wiley InterScience, New York, 2003. 23

[7] Pierre Baldi and G. Wesley Hatfield. *DNA Microarrays and Gene Expression: From Experiments to Data Analysis and Modeling*. Cambridge University Press, Cambridge, 2002. 23

[8] Antony Le Béchec, Pierre Zindy, Thomas Sierocinski, Dimitri Pétritis, Audrey Bihouée, Nolwenn Le Meur, Jean Léger, and Nathalie Théret. M@ia: a modular open-source application for microarray workflow and integrative datamining. *In Silico Biol*, 8(1):63–69, 2008. 116, 118

[9] Mikhail Belkin and Partha Niyogi. Towards a theoretical foundation for Laplacian-based manifold methods. In *Learning theory*, volume 3559 of *Lecture Notes in Comput. Sci.*, pages 486–500, Berlin, 2005. Springer. 49

[10] Fabio Benatti and Roberto Floreanini. Open quantum dynamics: complete positivity and entanglement. *Internat. J. Modern Phys. B*, 19(19):3063–3139, 2005. 82

[11] Michael W. Berry, Zlatko Drmač, and Elizabeth R. Jessup. Matrices, vector spaces, and information retrieval. *SIAM Rev.*, 41(2):335–362 (electronic), 1999. 46

[12] James C. Bezdek. *Pattern recognition with fuzzy objective function algorithms*. Plenum Press, New York, 1981. With a foreword by L. A. Zadeh, Advanced Applications in Pattern Recognition. 79

[13] Garrett Birkhoff. *Lattice theory*, volume 25 of *American Mathematical Society Colloquium Publications*. American Mathematical Society, Providence, R.I., third edition, 1979. 60, 64

[14] Massimo Campanino and Dimitri Petritis. On the physical relevance of random walks: an example of random walks on a randomly oriented lattice. In *Random walks and geometry*, pages 393–411. Walter de Gruyter GmbH & Co. KG, Berlin, 2004. 50

[15] Jeff Cheeger. A lower bound for the smallest eigenvalue of the Laplacian. In *Problems in analysis (Papers dedicated to Salomon Bochner, 1969)*, pages 195–199. Princeton Univ. Press, Princeton, N. J., 1970. 49

[16] Fan R. K. Chung. *Spectral graph theory*, volume 92 of *CBMS Regional Conference Series in Mathematics*. Published for the Conference Board of the Mathematical Sciences, Washington, DC, 1997. 49

[17] Rudi Cilibrasi and Paul M. B. Vitanyi. The google similarity distance. *IEEE Transactions on Knowledge and Data Engineering*, 19:370, 2007. 23

[18] Dragoš M. Cvetković, Michael Doob, and Horst Sachs. *Spectra of graphs*. Johann Ambrosius Barth, Heidelberg, third edition, 1995. Theory and applications. 49

[19] Andreas de Vries. Algebraic hierarchy of logics unifying fuzzy logic and quantum logic, 2007. 60

[20] Sándor Dominich. *Mathematical foundations of information retrieval*, volume 12 of *Mathematical Modelling: Theory and Applications*. Kluwer Academic Publishers, Dordrecht, 2001. 23

[21] Peter G. Doyle and J. Laurie Snell. *Random walks and electric networks*, volume 22 of *Carus Mathematical Monographs*. Mathematical Association of America, Washington, DC, 1984. 52

[22] Nelson Dunford and Jacob T. Schwartz. *Linear operators. Part I*. Wiley Classics Library. John Wiley & Sons Inc., New York, 1988. General theory, With the assistance of William G. Bade and Robert G. Bartle, Reprint of the 1958 original, A Wiley-Interscience Publication. 24, 40

[23] Joshua J. Forman, Paul A. Clemons, Stuart L. Schreiber, and Stephen J. Haggarty. Spectralnet – an application for spectral graph analysis and visualization. *BMC Bioinformatics*, 6:260, 2005. 46

[24] F. Fouss, A. Pirotte, J. Renders, and M. Saerens. A novel way of computing dissimilarities between nodes of a graph, In Proc., ECML workshop on Statistical Approaches for Web Mining (2004). 52

[25] Francois Fouss and Jean-Michel Renders. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 19(3):355–369, 2007. Member-Alain Pirotte and Member-Marco Saerens. 52

[26] Maurice Fréchet. Sur la définition axiomatique d'une classe d'espaces vectoriels distanciés applicables vectoriellement sur l'espace de Hilbert. *Ann. of Math. (2)*, 36(3):705–718, 1935. 25

[27] Peter Gärdenfors. Induction, conceptual spaces and AI. *Philos. Sci.*, 57(1):78–95, 1990. 23

[28] I. M. Gel′fand, M. I. Graev, and I. I. Pyatetskii-Shapiro. *Representation theory and automorphic functions*, volume 6 of *Generalized Functions*. Academic Press Inc., Boston, MA, 1990. Translated from the Russian by K. A. Hirsch, Reprint of the 1969 edition. 23

[29] Chris Godsil and Gordon Royle. *Algebraic graph theory*, volume 207 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 2001. 49, 56

[30] J. A. Hampton. Disjunction of natural concepts. *Memory and Cognition*, 16:579–591, 1988. 66

[31] Wolfgang Härdle and Léopold Simar. *Applied multivariate statistical analysis*. Springer, Berlin, second edition, 2007. 23, 39

[32] Jean-Claude Hausmann and Eugenio Rodriguez. The space of clouds in Euclidean space. *Experiment. Math.*, 13(1):31–47, 2004. 25

[33] Ramin Homayouni, Kevin Heinrich, Lai Wei, and Michael W. Berry. Gene clustering by latent semantic indexing of medline abstracts. *Bioinformatics*, 21(1):104–115, 2005. 46

[34] Roger A. Horn and Charles R. Johnson. *Topics in matrix analysis*. Cambridge University Press, Cambridge, 1994. Corrected reprint of the 1991 original. 35

[35] Harold Hotelling. New light on the correlation coefficient and its transforms. *J. Roy. Statist. Soc. Ser. B.*, 15:193–225; discussion, 225–232, 1953. 23, 39

[36] Kari Karhunen. Zur Spektraltheorie stochastischer Prozesse. *Ann. Acad. Sci. Fennicae. Ser. A. I. Math.-Phys.*, 1946(34):7, 1946. 39

[37] Leonard Kaufman and Peter J. Rousseeuw. *Finding groups in data*. Wiley Series in Probability and Mathematical Statistics: Applied Probability and Statistics. John Wiley & Sons Inc., New York, 1990. An introduction to cluster analysis, A Wiley-Interscience Publication. 121

[38] Maurice G. Kendall and Alan Stuart. *The advanced theory of statistics. Vol. 2*. Hafner Publishing Co., New York, third edition, 1973. Inference and relationship. 23, 39

[39] A. N. Kolmogorov. *Foundations of the theory of probability.* Chelsea Publishing Co., New York, 1956. Translation edited by Nathan Morrison, with an added bibliography by A. T. Bharucha-Reid. 64

[40] Amy N. Langville and Carl D. Meyer. A survey of eigenvector methods for Web information retrieval. *SIAM Rev.*, 47(1):135–161 (electronic), 2005. 23, 46

[41] Amy N. Langville and Carl D. Meyer. A reordering for the PageRank problem. *SIAM J. Sci. Comput.*, 27(6):2112–2120 (electronic), 2006. 23, 46

[42] C. Li and W. H. Wong. Model-based analysis of oligonucleotide arrays: expression index computation and outlier detection. *Proc Natl Acad Sci U S A*, 98(1):31–36, Jan 2001. 17

[43] D. J. Lockhart, H. Dong, M. C. Byrne, M. T. Follettie, M. V. Gallo, M. S. Chee, M. Mittmann, C. Wang, M. Kobayashi, H. Horton, and E. L. Brown. Expression monitoring by hybridization to high-density oligonucleotide arrays. *Nat Biotechnol*, 14(13):1675–1680, Dec 1996. 15

[44] Michel Loève. Fonctions aléatoires à décomposition orthogonale exponentielle. *Revue Sci.*, 84:159–162, 1946. 39

[45] L. H. Loomis. The lattice theoretic background of the dimension theory of operator algebras. *Mem. Amer. Math. Soc.*, 1955(18):36, 1955. 64

[46] Maassen and Kümmerer. Purification of quantum trajectories. *Dynamics and stochastics*, vol 48 of IMS Lecture Notes Monogr. Ser.:252–261, 2006. 82

[47] M. Meilă and J. Shi. A random walks view of spectral segmentation, In AI and Statistics, (2001). 23, 52

[48] J. Mercer. Functions of positive and negative type and their connection with the theory of integral equations. *Phil. Trans. Roy. Soc. London Ser. A*, 209:415–446, 1909. 23, 24

[49] Bojan Mohar. Some applications of Laplace eigenvalues of graphs. In *Graph symmetry (Montreal, PQ, 1996)*, volume 497 of *NATO Adv. Sci. Inst. Ser. C Math. Phys. Sci.*, pages 225–275. Kluwer Acad. Publ., Dordrecht, 1997. 49

[50] See-Kiong Ng, Zexuan Zhu, and Yew-Soon Ong. Whole-genome functional classification of genes by latent semantic analysis on microarray data. 29, 2004. 23

[51] Michael A. Nielsen and Isaac L. Chuang. *Quantum computation and quantum information.* Cambridge University Press, Cambridge, 2000. 82

[52] A. Odermatt, K. Barton, V. K. Khanna, J. Mathieu, D. Escolar, T. Kuntzer, G. Karpati, and D. H. MacLennan. The mutation of pro789 to leu reduces the activity of the fast-twitch skeletal muscle sarco(endo)plasmic reticulum ca2+ atpase (serca1) and is associated with brody disease. *Hum Genet*, 106(5):482–491, May 2000. 9

[53] K. R. Parthasarathy. *An introduction to quantum stochastic calculus*, volume 85 of *Monographs in Mathematics*. Birkhäuser Verlag, Basel, 1992. 24

[54] K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2:559–572, 1901. 23, 39

[55] Roger Penrose. *Shadows of the mind*. Oxford University Press, Oxford, 1994. A search for the missing science of consciousness. 67

[56] Dimitri Petritis. Mathematical foundations of quantum mechanics and its applications, Preliminary version (2006) available at http://perso.univ-rennes1.fr/dimitri.petritis/ps/qiccc.pdf. 26, 69

[57] Pavel Pták and Sylvia Pulmannová. *Orthomodular structures as quantum logics*, volume 44 of *Fundamental Theories of Physics*. Kluwer Academic Publishers Group, Dordrecht, 1991. Translated from the 1989 Slovak original by the authors. 23, 60, 69

[58] Franck Rapaport, Andrei Zinovyev, Marie Dutreix, Emmanuel Barillot, and Jean-Philippe Vert. Classification of microarray data using gene networks. *BMC Bioinformatics*, 8:35+, February 2007. 23

[59] Miklós Rédei. *Quantum logic in algebraic approach*, volume 91 of *Fundamental Theories of Physics*. Kluwer Academic Publishers Group, Dordrecht, 1998. 23, 60, 69

[60] M. Schena, D. Shalon, R. W. Davis, and P. O. Brown. Quantitative monitoring of gene expression patterns with a complementary dna microarray. *Science*, 270(5235):467–470, Oct 1995. 15

[61] M. Schena, D. Shalon, R. Heller, A. Chai, P. O. Brown, and R. W. Davis. Parallel human genome analysis: microarray-based expression monitoring of 1000 genes. *Proc Natl Acad Sci U S A*, 93(20):10614–10619, Oct 1996. 15

[62] I. J. Schoenberg. Remarks to Maurice Fréchet's article "Sur la définition axiomatique d'une classe d'espace distanciés vectoriellement applicable sur l'espace de Hilbert" [MR1503246]. *Ann. of Math. (2)*, 36(3):724–732, 1935. 25

[63] Denis Serre. *Matrices*, volume 216 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 2002. Theory and applications, Translated from the 2001 French original. 44

[64] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905, 2000. 23, 52

[65] Thomas Sierocinski, Anthony Le Béchec, Nathalie Théret, and Dimitri Petritis. Semantic distillation: a method for clustering objects by their contextual specificity. *CoRR*, abs/0710.1203, 2007. 85

[66] Roman Sikorski. *Boolean algebras*. Third edition. Ergebnisse der Mathematik und ihrer Grenzgebiete, Band 25. Springer-Verlag New York Inc., New York, 1969. 64

[67] Jinsong Tan, Kok Seng Chua, Louxin Zhang, and Song Zhu. Algorithmic and complexity issues of three clustering methods in microarray data analysis. *Algorithmica*, 48(2):203–219, 2007. 35

[68] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. B. Altman. Missing value estimation methods for dna microarrays. *Bioinformatics*, 17(6):520–525, Jun 2001. 18

[69] C. J. van Rijsbergen. *The geometry of information retrieval*. Cambridge University Press, Cambridge, 2004. 23, 26

[70] V. S. Varadarajan. *Geometry of quantum theory. Vol. I*. D. Van Nostrand Co., Inc., Princeton, N.J.-Toronto, Ont.-London, 1968. The University Series in Higher Mathematics. 23, 60, 69

[71] Sebastien Vast, Pierre Dupont, and Yves Deville. Automatic extraction of relevant nodes in biochemical networks, citeseer.ist.psu.edu/742958.html. 23

[72] Saraswathi Vishveshwara, K. V. Brinda, and N. Kannan. Protein structure: insights from graph theory. *Journal of Theoretical and Computational Chemistry*, 1:187–212, 2002. 23

[73] John von Neumann. *Mathematical foundations of quantum mechanics*. Princeton Landmarks in Mathematics. Princeton University Press, Princeton, NJ, 1996. Translated from the German and with a preface by Robert T. Beyer, Twelfth printing, Princeton Paperbacks. 82

[74] Joe H. Ward, Jr. Hierarchical grouping to optimize an objective function. *J. Amer. Statist. Assoc.*, 58:236–244, 1963. 118

[75] David L Wheeler, Deanna M Church, Scott Federhen, Alex E Lash, Thomas L Madden, Joan U Pontius, Gregory D Schuler, Lynn M Schriml, Edwin Sequeira, Tatiana A Tatusova, and Lukas Wagner. Database resources of the national center for biotechnology. *Nucleic Acids Res*, 31(1):28–33, Jan 2003. 7, 9

[76] Dominic Widdows. *Geometry and meaning*, volume 172 of *CSLI Lecture Notes*. CSLI Publications, Stanford, CA, 2004. With a foreword by Pentti Kanerva. 23

[77] Itai Yanai, Hila Benjamin, Michael Shmoish, Vered Chalifa-Caspi, Maxim Shklar, Ron Ophir, Arren Bar-Even, Shirley Horn-Saban, Marilyn Safran, Eytan Domany, Doron Lancet, and Orit Shmueli. Genome-wide midrange transcription profiles reveal expression level relationships in human tissue specification. *Bioinformatics*, 21(5):650–659, Mar 2005. 7

[78] Yee Hwa Yang, Sandrine Dudoit, Percy Luu, David M Lin, Vivian Peng, John Ngai, and Terence P Speed. Normalization for cdna microarray data: a robust composite method addressing single and multiple slide systematic variation. *Nucleic Acids Res*, 30(4):e15, Feb 2002. 17

[79] L. A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965. 65

Résumé

Les progrès des technologies de mesure et le séquençage des génomes, ont permis l'émergence, dans les années 1990, de techniques de mesure globale de l'expression génique, les puces à ADN. Ce type d'expérience, dit à « haut débit », en raison du volume de données qu'elles génèrent nécessitent un traitement automatique pour l'interprétation des résultats. Dans ce but, de nombreuses approches ont été développées, essentiellement réparties en deux familles : les méthodes de classification supervisées et non supervisées.

Nous présentons ici la distillation sémantique, une approche de classification non supervisée originale fondée sur un formalisme inspiré de la mesure physique en mécanique quantique permettant l'analyse des résultats d'analyse de puces à ADN. Cette méthode fournit à l'utilisateur une liste de gènes ordonnée par spécificité pour chaque échantillon biologique de l'expérience, décrivant ainsi chaque contexte cellulaire ainsi que l'influence de chaque gène dans ces contextes. Celle-ci a été mise à l'épreuve sur deux jeux de données : un jeu « tissus-spécifique » pour lequel notre méthode a correctement caractérisé les gènes spécifiques de chaque tissu, et un jeu de données cliniques de patients atteints de fibroses hépatiques à divers stades pour lequel la distillation sémantique a permis de trouver des signatures dans les voies métaboliques et les processus biologiques associés aux gènes spécifiques de chaque stade de la maladie.

Abstract

Advances in measurement technology and sequencing of genomes, have led to the emergence of DNA microarray technology in the 90's, allowing overall measurement of gene expression. This type of experience is said "high throughput" because of the volume of data they generate requiring automatic processing for results interpretation. In this context, many approaches have been developed and can be divided into two families: supervised and unsupervised classification methods.

We present here "semantic distillation" a novel unsupervised classification approach, based on a formalism inspired by the physical measurement in quantum mechanics, for the analysis of results from DNA chips. This method provides the user with an ordered list of specific genes for each biological sample of the experience, and describing each cellular context and the influence of each gene in these contexts. Semantic distillation was tested on two data sets: a "tissue-specific" set for which our method has correctly characterised specific genes for each tissue, and clinical data sets of patients with liver fibrosis at various stages for which semantic distillation helped to find signatures in metabolic pathways and biological processes associated with specific genes of each stage of the disease.