Use of data analysis techniques to solve specific bioinformatics problems

Serge Moulin

THESE DE DOCTORAT DE L'ETABLISSEMENT UNIVERSITE BOURGOGNE FRANCHE-COMTE

PREPAREE AU LABORATOIRE DISC (DEPARTEMENT D'INFORMATIQUE DES SYSTEMES COMPLEXES) DE L'INSTITUT FEMTO-ST (FRANCHE-COMTE ELECTRONIQUE MECANIQUE THERMIQUE ET OPTIQUE – SCIENCES ET TECHNOLOGIES)

Ecole doctorale n° 37

SPIM (Sciences Pour l'Ingénieur et Microtechniques)

Doctorat d'informatique

Par

M. Serge MOULIN

Use of data analysis techniques to solve specific bioinformatics problems

Thèse présentée et soutenue à Belfort, le 12 décembre 2018

Composition du Jury :

M Julien JACQUES	Professeur à l'Université de Lyon - Lumiere	Président
M. Arnaud LE ROUZIC	Chargé de Recherche CNRS à l'EGCE	Rapporteur
M. Christophe GUYEUX	Professeur à l'Université de Franche-Comté	Directeur de thèse
M. Stéphane CHRÉTIEN	Senior Research Scientist à National Physical Laboratory	Codirecteur de thèse

SPIM

Thèse de Doctorat

école doctorale **sciences pour l'ingénieur et microtechniques**

UFC UNIVERSITÉ DE FRANCHE-COMTÉ

N° 2 1 3 1 1 1 7 5

Apport de techniques d'analyse de donnée pour résoudre des problèmes spécifiques en bio-informatique

By

Serge MOULIN

A Dissertation Submitted to the

University of Franche-Comté

in Partial Fulfillment of the Requirements for the Degree of

DOCTOR OF PHILOSOPHY

in Computer Science

Dissertation Committee:

PR. JULIEN JACQUES	Université de Lyon - Lumiere	Reviewer
ARNAUD LE ROUZIC	EGCE	Reviewer
PR. CHRISTOPHE GUYEUX	Université de Franche-Comté	Supervisor
STÉPHANE CHRÉTIEN	National Physical Laboratory	Co-supervisor

SOMMAIRE

TABLE DES FIGURES

LISTE DES TABLES

ABRÉVIATIONS ET GLOSSAIRE

ABRÉVIATIONS

ADN ..... Acide désoxyribonucléique (en anglais : DNA)
ACP ..... Analyse en composantes principales (en anglais PCA).
AIC ..... Akaike information criterion
AUC ..... Area Under the Curve
ARN ..... Acide ribonucléique (en anglais RNA)
BIC ..... Bayesian information criterion
DNA ..... Deoxyribonucleic acid (en français : ADN)
ET ..... Élément transposable (en anglais TE).
GMM ..... Gaussian Mixture Model
LASSO ... Least Absolute Shrinkage and Selection Operator
LTR ..... Long terminal repeats
NCBI ..... National Center for Biotechnology Information
OTU ..... Operational Taxonomic Unit
PCA ..... Principal component analysis (en français : ACP)
RNA ..... Ribonucleic acid (en français : ARN)
ROC ..... Receiver Operating Characteristic
TE ..... Transposable element (en français ET)

GLOSSAIRE

Acides aminés protéinogènes : Composants de base de la protéine.
Clusteriser : Faire des groupes.
Codon : Suite de trois nucléotides codant un acide aminé protéinogène.
Colinéraires : Deux vecteurs $\vec{u}$ et $\vec{v}$ sont colinéaires si $\vec{u} = k\vec{v}$ où k est un nombre ; autrement dit si $\vec{v}$ est un multiple de $\vec{u}$. En statistique, si deux variables sont colinéaires (ex : le taux d'hormone A sécrétée par chaque patient est toujours le triple du taux d'hormone B sécrétée par ce même patient), alors les informations qu'elles apportent sont redondantes.
Bruit : Processus aléatoire. Dans le cadre d'un modèle de régression, le bruit désigne ce qu'on ne parvient pas à expliquer.
Diagonale (Matrice diagonale) : Une matrice diagonale est une matrice dont tous les coefficients en dehors de la diagonale sont nuls. Autrement dit, si M est une matrice diagonale et $j \neq i$ alors $M_{i,j} = 0$.

Éléments transposables : Séquence d'ADN capable de se déplacer dans le génome.
Eucaryote : Une cellule eucaryote est une cellule qui possède un noyau. Un organisme eucaryote est un organisme dont les cellules possèdent des noyaux par opposition aux procaryotes. Ex : l'homme est un organisme eucaryote.
Épissage : Procédure au cours de laquelle les introns (partie "inutile" de l'ARN) sont retirés et les exons sont conservés.
Éxons : Partie "codante" de l'ARN, conservée à l'épissage.
Intron : Partie "non-codante" de l'ARN, retirée à l'épissage.
Libres (vecteurs libres) : Un ensemble de vecteurs est libre si aucun ne peut s'écrire comme une combinaison linéaire des autres. C'est une extension aux dimensions supérieures de la non colinéarité.
Métagénomique (données métagénomiques) : Données génétiques issues d'environnements complexes (ex : intestin, océan, sols, air, etc.) prélevées dans la nature (par opposition à des échantillons cultivés en laboratoire).
Nucléotide : Élément de base de l'ADN. Peut être de type adénine (A), cytosine (C), guanine (G) ou thymine (T).
Phénotype : Caractères observables d'un individu (par opposition au génotype). Ex : la couleur d'une fleur est un caractère phénotypique.
Procaryote : Une cellule procaryote est une cellule qui ne possède pas de noyau. Un organisme procaryote est un organisme dont les cellules ne possèdent pas de noyau, par opposition aux eucaryotes. Les bactéries sont des organismes procaryotes.
Programmation dynamique : Mode de programmation consistant à décomposer le problème en sous-problèmes, puis à résoudre les sous-problèmes, des plus petits aux plus grands en stockant les résultats intermédiaires.
Régression logistique : Modèle statistique dont l'objectif est de prédire la valeur d'une variable qualitative, éventuellement qualitative ordonnée (par opposition à la régression linéaire). Ex : on cherche à prédire si un patient va attraper une maladie ou non en fonction de différentes variables.
Régression linéaire : Modèle statistique dont l'objectif est de prédire la valeur d'une variable quantitative (par opposition à la régression logistique). Ex : on cherche à prédire le prix adéquat d'un bien immobilier en fonction de différentes variables.
Rétrotransposons : Éléments mobiles du génome capables de se dupliquer en utilisant une transcription suivie d'un transcription inverse.
Surparamétrage : Un modèle statistique est surparamétré lorsqu'il a trop de paramètres. De fait certains sont alors inutiles car redondants.
Taxon : Ensemble d'individus partageant des caractères communs. Ce terme très générique peut donc désigner n'importe quel niveau de la classification du vivant. C'est-à-dire qu'il peut aussi bien définir une espèce (ex : espèce humaine) qu'une famille (ex : cervidés) ou une classe (ex : les mammifères) par exemple. **Transcription** : En biologie, la transcription est le mécanisme au cours duquel une molécule d'ARN est créée en copiant une partie de l'un des deux brins d'une molécule d'ADN.
Univariée : Une régression est dite univariée s'il n'y a qu'une seule variable explicative (cf. partie 2.2.1 de l'état de l'art).
Vraisemblance : La vraisemblance d'un modèle statistique est égale à la probabilité d'obtenir les données observées d'après ce modèle. Par exemple, si on tire à pile ou face et que l'on obtient pile, la vraisemblance du modèle "la pièce n'est pas truquée" est 0.5, la vraisemblance du modèle "la pièce est truquée et tombe toujours sur pile" est 1, la vraisemblance du modèle "la pièce est truquée et tombe toujours sur face" est 0. Le modèle le plus vraisemblable n'est toutefois pas toujours le meilleur, notamment du fait

des risques de sur-interprétation.

REMERCIEMENTS

À mes directeurs de thèse, Stéphane et Christophe qui m'ont permis de réaliser cette thèse et m'ont fait aborder tous les aspects de la recherche académique. Ce fut un plaisir de travailler avec vous.

À mes coauteurs, Emmanuelle, Nicolas, Cyril, Michel, Marine, Thierry et Franz. A Valentin et Césarion qui ont accepté mon encadrement. A tout mes collaborateurs en général avec qui nous avons pu mettre en commun nos compétences durant ces trois années.

À Sylvia et Louise qui ont supporté un doctorant à la maison pendant trois ans. A Émile qui est né pendant ce doctorat.

À mes parents pour leurs corrections orthographiques du manuscrit, et aussi accessoirement pour m'avoir fait naître et élevé sans quoi je n'aurais pas fait ce travail.

À mes rapporteurs MM. Julien JAQUES et Arnaud Le ROUZIC pour avoir pris le temps d'évaluer ce manuscrit ainsi que ma soutenance.

Aux collègues du laboratoire DISC de Besançon, aux doctorants du laboratoire pour les croissants du mercredi, à mes sœurs, à ma famille en général, à mes amis évidemment.

INTRODUCTION

INTRODUCTION GÉNÉRALE

Le nombre de séquences génétiques complètement décryptées augmente de manière exponentielle sous l'impulsion d'outils de séquençage de plus en plus performants. En particulier, l'apparition d'outils de séquençage haut débit (en anglais high-throughput sequencing ou HTS) tels que Ion Torrent `rusk2010torrents`, 454 `el2007evolution` ou Illumina MiSeq `Illumina` a drastiquement fait chuter les coûts de ces séquençages. Ainsi, le premier séquençage du génome humain `international2004finishing`, s'est achevé en 2003 après 13 ans de travaux d'un consortium international réunissant 16 laboratoires pour un coût total d'environ 2,7 milliards de dollars. Une telle opération coûte aujourd'hui un peu plus de 1000 dollars (cf. figure 1).

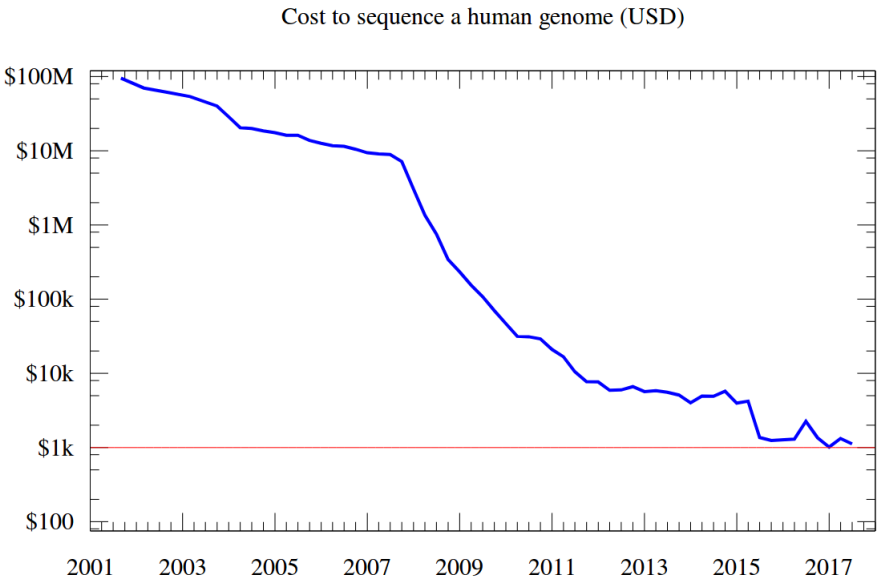

FIGURE 1 – Évolution du coût du séquençage du génome humain. Graphique de Ben Moore et Grendel Khan pour https://fr.wikipedia.org/ (séquençage de l'ADN) d'après des données de https://www.genome.gov/sequencingcostsdata/

Une telle augmentation des capacités de séquençage a permis la constitution de larges bases de données. Ainsi par exemple, en écologie, les chercheurs ont pu constituer des bases de données métagénomiques recensant l'ensemble des populations d'une zone géographique donnée `zappelini2015diversity`, `foulon2016impact`, `danielsen2012fungal`. De telles bases de données se sont constituées également dans le domaine médical ou simplement en recherche biologique (séquençage de différentes espèces). De plus, ces séquences génétiques deviennent de plus en plus

facilement et librement accessibles grâce à la création de bases de données en ligne. On peut évoquer en premier lieu le site du Centre américain pour les informations biotechnologiques (en anglais National Center for Biotechnology Information ou NCBI NCBI, mais aussi des sites plus spécialisés comme Flybase flybase qui traite exclusivement d'insectes, ou encore des sites affiliés à une université comme celle de Californie à Santa Cruz par exemple UCSC. Cette plus grande disponibilité des données ouvre de nouveaux sujets d'étude qui nécessitent de la part des statisticiens et bio-informaticiens de développer des outils adaptés.

Par ailleurs, les progrès constants de la statistique nécessitent d'être régulièrement adaptés au contexte de la bio-informatique. Parmi ces avancées, notons celles qui ont été réalisées dans le domaine de la réduction de dimension comme les Laplacian eigenmaps qui permettent à la fois de visualiser des données en grandes dimensions mais aussi servent d'étape préliminaire au clustering de ces données. Notons également les avancées dans le domaine des régressions, où des méthodes comme le LASSO (Least Absolute Shrinkage and Selection Operator Tibshirani:JRSSB96) permettent une sélection plus efficace des variables explicatives parmi un grand nombre de variables candidates.

L'objectif de cette thèse, est l'application de techniques avancées de statistiques à des problématiques de bio-informatique. Au gré de nos collaborations, nous avons été amenés à travailler plus précisément sur les questions de clustering des séquences génétiques, de propagation des éléments transposables, d'analyse de données métagénomiques et de régression polytomique ordonnée.

Ainsi ce travail de thèse s'attelle tout d'abord à une question extrêmement générale : comment clusteriser des séquences génétiques de la façon la plus efficace possible ? C'est-à-dire comment partager une base de données de séquences génétiques en différents groupes ? Cette question extrêmement générale peut être appliquée de différentes façons. Par exemple, le clustering peut être utilisé pour déterminer des espèces. Ce type d'espèces, définies par leur patrimoine génétique plutôt que par leur phénotype, est appelé "Operational Taxonomic Unit" (OTU). Les OTUs sont généralement définis par clustering de l'ARN 16S hao2011clustering. Le clustering de séquences génétiques peut également être utilisé pour définir des taxons parmi un ensemble d'espèces représentées par leur ADN. Enfin le clustering peut également permettre d'étudier la répartition de sous-populations à l'intérieur d'une même espèce torroni1992native. Des outils de clustering pour séquences génétiques existaient déjà avant ces travaux de thèse. Mais récemment, le clustering a vu des progrès très significatifs dus aux méthodes spectrales et aux plongements non linéaires. Un des objectifs de cette thèse est d'apporter une nouvelle pierre à l'édifice en montrant comment ces techniques peuvent être mises en œuvre efficacement pour la bio-informatique. Dans ce manuscrit nous présentons un outil de clustering basé sur une combinaison de Laplacian eigenmaps belkin2001laplacian et de Modèle de Mélange Gaussien (GMM) day1969estimating. Les tests que nous avons effectués sur notre outil utilisant des données réelles et simulées montrent des résultats encourageants. En particulier, les essais sur données simulées montrent que les clusterisations effectuée par notre outil retrouvent les clusters attendus nettement plus efficacement que les outils de clustering les plus populaires. Ce travail sur le clustering de séquences génétiques a ainsi été le plus "généraliste" des travaux effectués dans le cadre de ce doctorat. Les travaux suivants portent sur des aspects plus spécifiques de la bio-informatique, qui requièrent leurs outils propres.

Une partie importante de ces travaux de doctorat a concerné l'étude des éléments transposables. Ces éléments mobiles du génome, découverts durant les années 50 par Barbara McClintock mcclintock1950or2, sont une clef de compréhension importante de la constitution du génome et donc de l'évolution. Ils représentent ainsi 45% lander2001initial du génome de l'homme, 15% de celui de la mouche Drosophile (*Drosophila melanogaster*) et plus de 70% chez le maïs (*Zea mais*) sanmiguel1998evidence. Nous nous sommes plus particulièrement intéressés ici au cas des rétrotransposons (ou éléments transposables de classe I) qui se propagent dans le génome par un système de copier-coller (par opposition aux transposons à ADN où éléments transposables de classe II qui se propagent principalement par couper-coller). Nous avons proposé un modèle mathématique de propagation de ces rétrotransposons. Ce modèle suppose principalement que les copies filles apparaissent plus probablement à proximité de leur copie mère, que le rétrotransposon peut être dégradé à tout moment par des mutations de ses nucléotides, et enfin, que les dégradations subies par un rétrotransposon affectent la capacité de ce rétrotransposon à se dupliquer. Nous proposons ensuite un programme informatique permettant d'estimer les paramètres de ce modèle.

Une autre situation qui a attiré notre attention durant ce doctorat est l'analyse des données métagénomiques. Plus précisément, dans le cadre d'une collaboration avec le laboratoire d'écologie (laboratoire chrono-environnement), il nous a été demandé de déterminer parmi un grand ensemble d'OTUs de champignons et de bactéries quelles populations étaient les plus diminuées par une pollution au mercure, et quelles populations étaient au contraire renforcées par cette pollution. Dit autrement, on s'intéresse à connaître les meilleurs prédicteurs de la pollution parmi les différentes OTUs. Pour déterminer cela, nous avons proposé un modèle de courbe ROC. Ce modèle très utilisé en médecine est beaucoup plus marginalement appliqué dans le cadre d'études métagénomiques en écologie, alors que nous pensons qu'il y a toute sa place. Notre contribution ici a été de produire un outil pour effectuer une analyse ROC sur chacun des OTU, de collecter les résultats et d'exhiber les OTUs les plus discriminantes. L'objectif était que cet outil soit le plus simple possible d'utilisation pour des utilisateurs non habitués à la programmation informatique. L'application de cette méthode à la base de données fournie par le laboratoire chrono-environnement a ainsi permis d'exhiber des OTUs particulièrement prédictives qui n'étaient pas détectées par les précédentes analyses.

Finalement, nous avons concentré notre attention sur un problème de statistique dont les applications médicales (notamment) sont particulièrement saillantes. En langage de statisticien, ce problème est celui de la régression polytomique ordonnée quand $p > n$. Dit de manière plus profane, la question est de créer un modèle pour prédire une variable qualitative ordonnée (typiquement une tumeur qui aurait plusieurs niveaux de gravité) en fonction d'un grand nombre de variables quantitatives (typiquement le niveau d'expression d'un grand nombre de gènes), y compris si le nombre de variables est plus grand que le nombre de sujets (typiquement : y compris si le nombre de gènes étudiés est supérieur au nombre de patients). Résoudre ce problème de régression logistique ordonnée nécessite, comme pour tout problème de régression en général, de réaliser une sélection des variables véritablement utiles. Ce genre de situation, dans laquelle le nombre de variables est grand, est particulièrement délicat du point de vue statistique, car il rend impraticables les procédures classiques de sélections de variables de type forward ou backward (cf. partie 2.2.2.3 de l'état de l'art). Pour résoudre ce problème de sélection

de variables, nous avons implémenté une pénalisation par la norme somme des coefficients (ou pénalisation de norme ℓ_1) similaire à ce que propose le modèle du LASSO dans le cadre d'une régression linéaire. Une partie importante de ce travail a consisté à choisir le degré de pénalisation à utiliser. Nous avons pour cela implémenté différente méthodes, des classiques (AIC `akaike1998information`, BIC `schwarz1978estimating`) et des plus récentes (Quantile Universal threshold `giacobino2015quantile`, Online Frank-Wolfe `chretien2018hedging`).

PLAN DU MANUSCRIT

A la suite de cette introduction, se trouve un état de l'art. Cet état de l'art est partagé en deux parties. La partie "bioinformatique" de cet état de l'art présente le vocabulaire de base nécessaire à la compréhension de cette thèse, et décrit quelques méthodes d'alignement de séquences. La partie "statistique" décrit des méthodes de clustering ainsi que des méthodes de réductions de dimensions souvent indispensables au clustering. Cette partie statistique présente également différentes méthodes de régressions (linéaire, logistique, polytomique ordonnée) et explique pourquoi et comment les variables pertinentes sont sélectionnées dans le cadre de ces régressions. A la suite de cet état de l'art, la partie "contributions" est partagée en 4 sous-parties : les travaux inhérents au clustering de séquences, ceux qui concernent la propagation des éléments transposables au sein du génome, ceux qui traitent de l'application des courbes ROC aux données métagénomiques en écologie et enfin ceux dont le sujet est la régression polytomique ordonnée. Chacune de ces parties reprend l'article publié ou proposé au sujet de ces travaux, accompagné si nécessaire d'informations complémentaires. Enfin une conclusion permet de revenir sur les avancées de ce doctorat et de développer les possibilités d'amélioration.

PUBLICATIONS

[**1**] *Simulation-based estimation of branching models for LTR retrotransposons.*
Serge Moulin, Nicolas Seux, Stéphane Chrétien, Christophe Guyeux et Emmanuelle Lerat.
Bioinformatics, Volume 33, Issue 3, 1 February 2017, Pages 320–326
https ://doi.org/10.1093/bioinformatics/btw622

[**2**] *A clustering package for nucleotide sequences using Laplacian Eigenmaps and Gaussian Mixture Model.*
Marine Bruneau, Thierry Mottet, Serge Moulin, Maël Kerbiriou, Franz Chouly, Stéphane Chretien et Christophe Guyeux.
Computers in Biology and Medicine, Volume 93, 1 February 2018, Pages 66-74
https ://doi.org/10.1016/j.compbiomed.2017.12.003

[**3**] *l1-Penalised Ordinal Polytomous Regression Estimators with Application to Gene Expression Studies.*
Stéphane Chrétien, Christophe Guyeux et Serge Moulin.
18th International Workshop on Algorithms in Bioinformatics (WABI 2018)
http ://drops.dagstuhl.de/opus/volltexte/2018/9319/

En soumission : *Dominance and characterization of Pseudomonas at a chlor-alkali tailings dumps.* Cyril Zappelini, Serge Moulin, Nicolas Capelli, François Maillard, Christophe Guyeux, Didier Hocquet et Michel Chalot

I

ÉTAT DE L'ART

1

ÉLÉMENTS DE BIO-INFORMATIQUE

1.1/ PRÉAMBULE : ADN ET SÉQUENCES GÉNÉTIQUES (RAPPEL DE VOCABULAIRE)

L'acide désoxyribonucléique (ADN) découvert en 1953 par Francis Crick et James Watson `watson1953molecular` est une molécule biologique présente dans toutes les cellules et qui contient l'information génétique. Il se présente sous la forme bien connue d'une double hélice (cf. image 1.1).

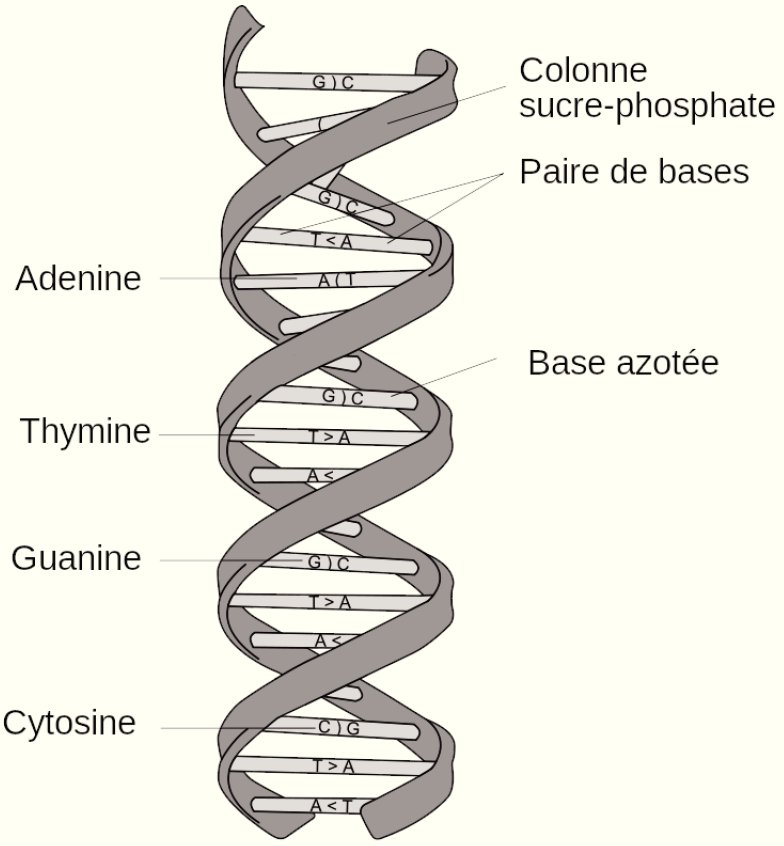

FIGURE 1.1 – Structure en double hélice de l'ADN. Image de Messer Woland pour https://fr.wikipedia.org (Acide désoxyribonucléique)

Les éléments de base de l'ADN se nomment les "nucléotides" ou simplement "bases". Ils

peuvent être de 4 types : adénine (A), cytosine (C), guanine (G) ou thymine (T). Ainsi, un brin d'ADN peut être représenté comme un mot composé uniquement des 4 lettres A, T, C et G. Dans la structure en double hélice de l'ADN, l'adénine est toujours opposée à la thymine et la cytosine est toujours opposée à la guanine. Ainsi la connaissance d'un seul des deux brins est nécessaire pour connaître la composition d'une molécule d'ADN. Le code génétique contenu dans l'ADN permet notamment la création de protéines, éléments essentiels, au fonctionnement de la cellule. Ce processus commence par la "transcription" c'est-à-dire la création d'une copie d'une partie d'un brin d'ADN en ARN messager (à l'exception de la thymine (T) qui est alors remplacée par l'uracile (U)). Puis, l'ARN messager subit une phase "d'épissage" dans laquelle les parties qui vont effectivement être lues, appelé "exons", sont conservées, tandis que les parties non lues appelées "introns", sont éliminées. L'ARN messager ayant subi cette opération est appelé "ARN messager mature". Finalement, l'ARN messager mature est traduit en protéine par des ribosomes. Les éléments de base de la protéine sont les acides aminés protéinogènes. Il existe 22 sortes différentes de ces acides aminés protéinogènes. Une protéine peut ainsi être vue comme un long mot dont l'alphabet est composé de 22 lettres. Le choix de l'acide aminé à incorporer à la protéine est déterminé par la lecture d'une succession de 3 nucléotides aussi appelée "codon". Il existe donc 64 codons différents (4^3), certains pouvant représenter un même acide aminé. La lecture de l'ARN messager par les ribosomes s'arrête lorsque ceux-ci rencontrent un des 3 "codons stop". La transcription de l'ADN en ARN ne produit pas uniquement de l'ARN messager, mais également entre autre l'ARN de transfert qui apporte les acides aminés au ribosome, et l'ARN ribosomique qui est le constituant principal des ribosomes. Dans ce cas également, l'ARN subit une phase d'épissage qui conserve les exons et rejette les introns. Chaque partie de la molécule d'ADN vouée à un rôle précis (transcription en ARN messager, ou ARN de transfert ou ARN ribosomique) est appelé un gène.

Un des rôles essentiels de la bio-informatique est la compréhension de ces différentes séquences (ADN, ARN, protéines) et de leur lien entre elles. Ces séquences sont alors considérées comme des mots dans leur alphabet respectif (de 4 lettres pour l'ADN et l'ARN, de 22 lettres pour les protéines) afin d'être traitées par les programmes informatiques adéquats.

1.2/ ALIGNEMENT DE SÉQUENCES ET SIMILARITÉ

L'alignement de séquences est une technique fondamentale de la bio-informatique. Cette technique, comme son nom l'indique, consiste à "placer" les séquences côte à côte de telle façon qu'un maximum de nucléotides coïncident. Une méthode d'alignement de séquences peut être "globale" si elle cherche à aligner au mieux l'ensemble des séquences, ou "locale" si son objectif est de chercher des morceaux de ces séquences ayant une grande similarité. Dans les deux chapitres suivants, nous présentons une méthode d'alignement globale (algorithme de Needleman-Wunsch) et une méthode d'alignement locale (algorithme de Smith-Waterman) parmi les plus utilisées en bio-informatique.

1.2.1/ L'ALGORITHME DE NEEDLEMAN-WUNSCH

L'algorithme de Needleman-Wunsch, `needleman1970general`, est un algorithme de programmation dynamique permettant d'aligner deux séquences nucléotidiques ou deux protéines entre elles. On peut pour cela rallonger chaque séquence en insérant des cases vides ("gap"). Pour fonctionner, l'algorithme a besoin de deux séquences, mais également des scores associés au fait que deux nucléotides se correspondent ("match") ou ne se correspondent pas ("mismatch"), et la pénalité associée au fait d'insérer une case vide ("gap"). La matrice de la figure 1.2 montre formellement comment aligner la séquence GCATGCG avec la séquence GATTACA si on considère un gain de 1 par match ainsi qu'une pénalité de 1 par mismatch et par gap.

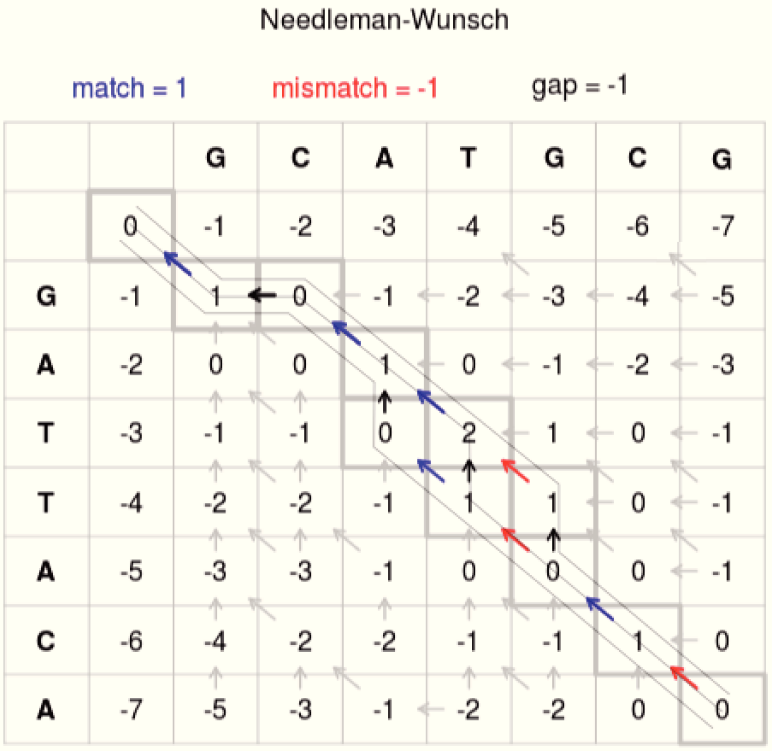

FIGURE 1.2 – Matrice à remplir pour obtenir le meilleur alignement de Needleman-Wunsch. Source : https://en.wikipedia.org (Needleman–Wunsch algorithm).

La procédure pour remplir cette matrice est la suivante :

- Placer 0 dans la case en haut à gauche.
- Pour remplir une case de la matrice, vous devez avoir accès à la valeur de la case supérieure, de la case à gauche et de la case en diagonale supérieure gauche. Pour les cases de la ligne supérieure il n'y a besoin que de la case à la gauche. Pour les cases de la colonne de gauche, il n'y a besoin que de la case supérieure.
- Lorsqu'on dispose des valeurs nécessaires pour remplir une case, on calcule le maximum des trois valeurs suivantes :
  - La valeur de la case à gauche (si elle existe) pénalisée par la pénalité attribuée au gap. En effet, un décalage à droite dans la matrice correspond à la création d'un gap dans la séquence représentée horizontalement.

— La valeur de la case supérieure (si elle existe) pénalisée par la pénalité attribuée au gap. En effet, un décalage en bas dans la matrice correspond à la création d'un gap dans séquence représentée verticalement.
— La valeur de la case en diagonale supérieure gauche à laquelle on ajoute le score de match si les nucléotides correspondent ou le score de mismatch si les nucléotides ne correspondent pas.

Quand la matrice est complètement remplie, la valeur de la case inférieure droite indique le score obtenu par les deux séquences. Plus les séquences sont similaires, plus ce score est important. Il faut alors "remonter" depuis la case en bas à droite en suivant le(s) chemin(s) possible(s) pour trouver le(s) meilleur(s) alignement(s) possible(s). Dans notre cas, les meilleurs alignements possibles sont

G	C	A	T	G	-	C	G
G	-	A	T	T	A	C	A

, ainsi que

G	C	A	-	T	G	C	G
G	-	A	T	T	A	C	A

, ou finalement

G	C	A	T	-	G	C	G
G	-	A	T	T	A	C	A

Le score et l'alignement obtenus dépendent des valeurs attribuées aux "matchs", "mismatchs" et "gap". Dans l'exemple proposé ci-dessus, chaque match obtient le même bonus et chaque mismatch obtient le même malus. Ce n'est pas nécessairement le cas en général. Par exemple, la matrice BLOSUM (figure 1.3), souvent utilisée dans le cadre de l'alignement de protéines, accorde des bonus différents aux matchs selon l'acide aminé concerné et accorde également des malus différents aux mismatchs selon le couple d'acides aminés concerné. La matrice indiquant les valeurs accordées aux matchs et mismatchs est appelée matrice de similarité des caractères (attention le terme "matrice de similarité" peut prendre des sens différents au cours de ce manuscrit). Dans le cas de l'exemple proposé ci-dessus, la matrice de similarité des caractères est celle qui est montrée dans la table 1.1. En général, pour les alignements de chaînes de nucléotides, on utilise plutôt la matrice EDNAFULL, dans laquelle les matchs entre nucléotides obtiennent un bonus de 5 et les missmatchs obtiennent un malus de 4. Il est également possible de distinguer, dans le score, l'ouverture d'un gap (*i.e.* ajouter une case vide après un nucléotide) et l'extension d'un gap (*i.e.* ajouter une case vide après une autre case vide). Pénaliser moins l'extension d'un gap que l'ouverture est assez naturel dans le sens où les séquences de nucléotides peuvent éventuellement subir des délétions de blocs.

TABLE 1.1 – Matrice de similarité des caractères avec match = 1 et mismatch = -1

	A	T	C	G
A	1	-1	-1	-1
T	-1	1	-1	-1
C	-1	-1	1	-1
G	-1	-1	-1	1

1.2.2/ L'ALGORITHME DE SMITH-WATERMAN

L'algorithme de Smith-Waterman `smith1981comparison` est un algorithme d'alignement local de séquences génétiques. Son fonctionnement est très proche de celui de Needleman-Wunsch. Ces deux algorithmes présentent toutefois deux différences :

— La valeur d'une case de la matrice à compléter ne peut pas être négative. Le calcul de la valeur d'une case se fait de la même façon que dans le cas de Needleman-

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
Ala	4																			
Arg	-1	5																		
Asn	-2	0	6																	
Asp	-2	-2	1	6																
Cys	0	-3	-3	-3	9															
Gln	-1	1	0	0	-3	5														
Glu	-1	0	0	2	-4	2	5													
Gly	0	-2	0	-1	-3	-2	-2	6												
His	-2	0	1	-1	-3	0	0	-2	8											
Ile	-1	-3	-3	-3	-1	-3	-3	-4	-3	4										
Leu	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4									
Lys	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5								
Met	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5							
Phe	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6						
Pro	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7					
Ser	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4				
Thr	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5			
Trp	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11		
Tyr	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	
Val	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4

FIGURE 1.3 – Matrice BLOSUM. Source : Hannes Röst pour https://es.wikipedia.org/wiki/BLOSUM

Wunsch, sauf qu'au lieu de prendre le maximum entre les trois valeurs citées au chapitre précédent, on prend le maximum entre ces trois valeurs et zéro.

— Le score d'alignement n'est pas la valeur de la case en bas à droite, mais c'est la valeur maximale entre toutes les valeurs de la ligne inférieure et de la colonne de droite.

La conséquence de ces modifications est que ce qui est obtenu ici n'est pas l'alignement de la totalité de la première séquence avec la totalité de la deuxième mais plutôt le meilleur alignement d'une partie de la première séquence avec une partie de la deuxième. C'est pourquoi l'algorithme de Smith-Waterman est un algorithme d'alignement local.

La figure 1.4 présente une application de l'algorithme de Smith-Waterman pour aligner les séquences ATCGAA et CATAC. Dans cet exemple, le gain par match est +5, un missmatch apporte -3 et un gap -4. Ici le meilleur alignement est

A	T	-	C
A	T	A	C

, pour un score de 11. Comme dans le cas de l'algorithme de Needleman-Wunsch, on retrouve ce meilleur alignement en remontant le chemin depuis la case contenant le meilleur score. Ici le chemin à remonter est symbolisé par les flèches rouges. Cet alignement ne prend donc en compte qu'une partie des nucléotides de la première chaîne (du premier au troisième nucléotide) et qu'une partie des nucléotides de la seconde chaîne (du deuxième au dernier nucléotide).

	-	A	T	C	G	A	A
-	0	0	0	0	0	0	0
C	0	0	0	5	1	0	0
A	0	5	1	1	2	5	5
T	0	1	10	6	2	1	2
A	0	5	6	7	3	7	6
C	0	1	2	11	7	3	4

FIGURE 1.4 – Exemple d'application de l'algorithme de Smith-Waterman. Image de Jock Banan pour https://fr.wikipedia.org (Algorithme de Smith-Waterman)

1.3/ ÉLÉMENTS TRANSPOSABLES

Découverts dans les années 50 par Barbara McClintock `mcclintock1950or2`, les éléments transposables désignent l'ensemble des éléments mobiles du génome, c'est-à-dire des chaînes d'ADN mobiles. Ces éléments transposables peuvent constituer une part importante du génome. Notamment ils constituent environ 45% du génome humain `lander2001initial` et plus de 70% du génome du maïs `sanmiguel1998evidence`. De ce fait, ils sont considérés comme un moteur important de l'évolution et de la biodiversité. Ces éléments transposables peuvent fonctionner sur un principe de couper-coller ou de copier-coller. Ils sont partagés en deux grandes catégories, les éléments transposables de classe I ou rétrotransposons, et les éléments transposables de classe II ou transposons.

Les rétrotransposons, ou éléments transposables de classe I, sont des éléments transposables qui fonctionnent sur un principe de copier-coller grâce à une transcription de l'ADN en ARN et une rétrotranscription de cet ARN en ADN. C'est de cette rétrotranscription que vient leur nom de "rétro"transposons. Les transposons ou éléments transposables de classe II peuvent fonctionner par couper-coller (ex :Tn10, Tn5 Mos1...) ou par copier-coller (ex : IS911). Mais, dans les deux cas, leur propagation n'implique pas de transcription.

La propagation des retrotransposons au sein du génome est le sujet de notre contribution 2. Comme nous utilisons un modèle de branchement pour modéliser cette propagation, cette contribution présente brièvement des utilisations précédentes de modèles de branchement dans le cadre de l'étude des ETs. Il s'agit toutefois généralement, dans ces utilisations précédentes, d'étudier via modèle de branchement l'évolution de populations dont les membres possèdent des éléments transposables.

2

ÉLÉMENTS DE STATISTIQUE

2.1/ PARTITIONNEMENT DE DONNÉES

En statistique, le partitionnement de données désigne le fait de partager des données en différents groupes. Les membres d'un même groupe sont alors supposés avoir des similarités entre eux ou être proches du point de vue de la métrique choisie.

Au sein du partitionnement de données, on distingue deux grandes catégories qui sont la classification supervisée (en anglais "classification") et la classification non supervisée (en anglais "clustering"). La classification supervisée désigne le cas où l'utilisateur humain connaît le sens des groupes qu'il veut obtenir. Il fournit alors à son algorithme des exemples d'éléments de ces différents groupes, et l'algorithme doit par la suite être capable de classer les nouveaux sujets dans les groupes adéquats. Par exemple, les travaux de reconnaissance d'images visant à permettre aux machines de reconnaître si une image contient ou non une personne rentrent dans ce cadre de la classification supervisée. Dans le cadre de la classification non supervisée, au contraire, l'utilisateur humain fournit à son algorithme directement toutes les données sans lui fournir d'a priori sur le sens des groupes qu'il doit constituer, mais seulement une métrique. C'est alors l'algorithme qui définit les groupes et, selon les cas, qui en définit le nombre. Charge à l'utilisateur humain d'interpréter le sens de cette classification s'il y en a une.

Dans ce manuscrit, on s'intéressera principalement à la classification non-supervisée, car c'est ce qui va nous permettre de générer des clusters de séquences génétiques sans avoir à fournir d'a priori. Les deux chapitres suivants présentent deux des méthodes les plus utilisées. La partie 2.1.3 présentera des méthodes de réduction de dimension, ce qui est une étape préalable souvent nécessaire à la classification non supervisée.

2.1.1/ GMM

Le modèle de mélange gaussien (en anglais Gaussian Mixture Model ou GMM `day1969estimating`) est un modèle de clustering non supervisé. Ce modèle assume que les données suivent une distribution $\sum_{j=1}^{k} \tau_j \ \mathcal{N}(\mu_j, \Sigma_j)$, où k est le nombre de clusters, τ_j est la probabilité pour un sujet d'être dans le j^{eme} cluster et $\mathcal{N}(\mu_j, \Sigma_j)$ est la loi normale de moyenne μ_j et de matrice de variance-covariance Σ_j. En d'autres termes, cette distribution est une moyenne pondérée de plusieurs distributions gaussiennes. La

figure 2.1 montre une distribution de mélange gaussien en 1 dimension avec 2 clusters tandis que la figure 2.2 montre un nuage de points répartis en 3 clusters suivant un mélange gaussien dans un espace à dimension 2.

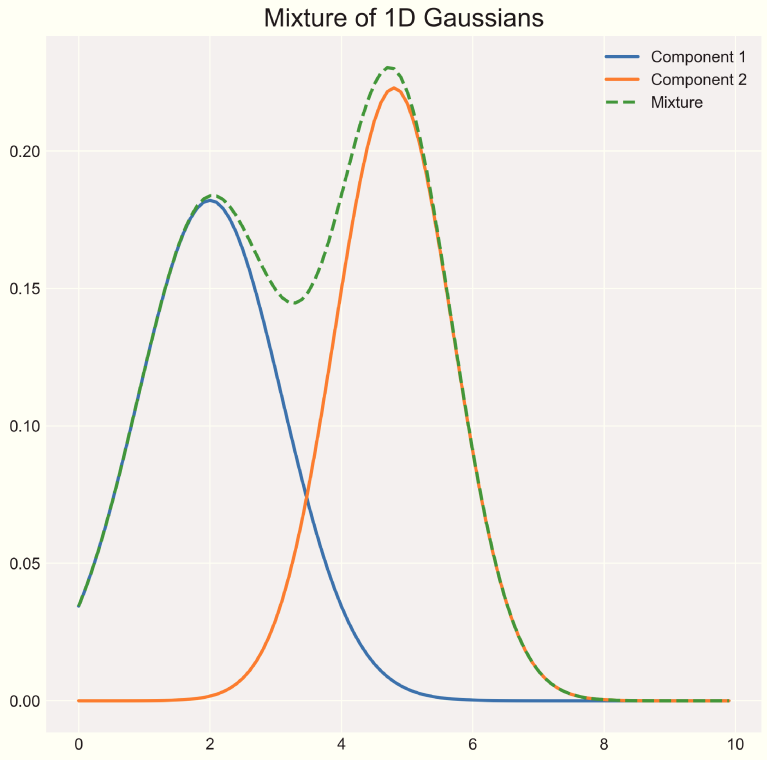

FIGURE 2.1 – Distribution d'un mélange gaussien à 1 dimension, source : https://angusturner.github.io

Un exemple simple et parlant de mélange gaussien est celui du rapport de la taille de l'index sur celle de l'annulaire, aussi appelé rapport 2D:4D pour "2nd to 4th digit ratio". Dans tous les groupes ethniques, ce rapport est plus faible pour les hommes que pour les femmes `nelson2006news`. Pour autant ce rapport varie d'un groupe ethnique à l'autre, et ainsi il est inconnu pour les groupes disparus. Par conséquent, lorsqu'il a été question de savoir si peindre des traces de mains sur les grottes préhistoriques était un rituel réservé à un sexe, ou accessible aux deux sexes, ceci s'est résumé à une question statistique. Si le rituel est accessible aux deux sexes alors les rapports 2D:4D des mains visibles sur la grotte suivent un mélange gaussien à deux clusters tel que celui de la figue 2.1. Si le rituel n'est accessible qu'à un seul sexe, alors la distribution des rapports 2D:4D est une simple gaussienne. Cet exemple est assez intuitif car c'est une utilisation de mélange gaussien dans un espace à une dimension (les sujets ne sont représentés que par une variable, leur rapport 2D:4D). En général toutefois, les modèles de mélanges gaussiens s'appliquent à des espaces multidimensionnels (les sujets sont représentés par plusieurs variables).

Les paramètres du modèle de mélange gaussien sont généralement estimés par

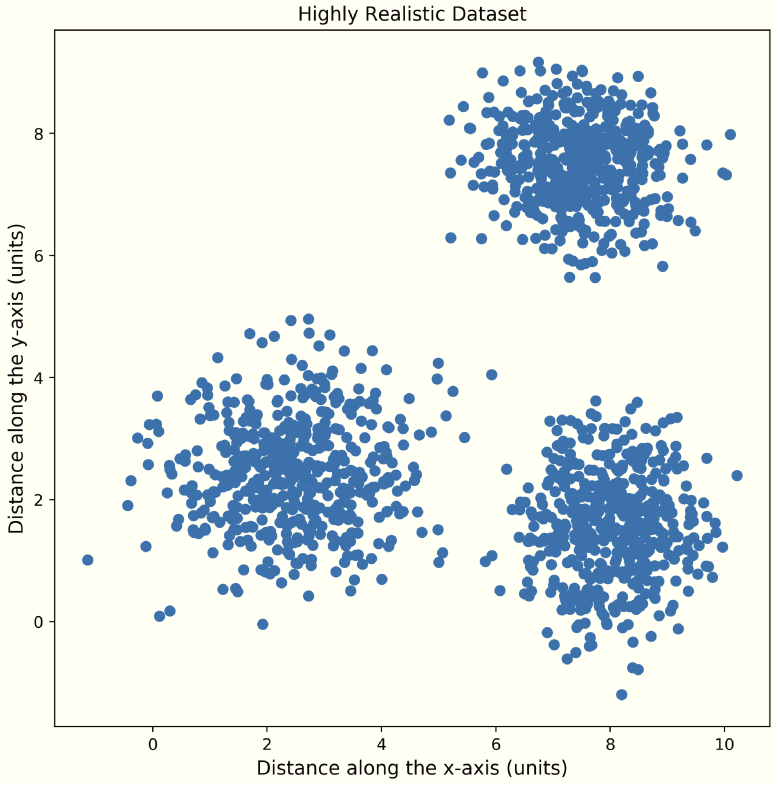

FIGURE 2.2 – Nuage de points suivant un mélange gaussien source : https://angusturner.github.io

FIGURE 2.3 – Traces des mains dans une grotte. Source Mariano Cecowski pour https://fr.wikipedia.org (Cueva de las Manos)

l'algorithme espérance-maximisation `dempster1977maximum` (en anglais expectation-maximization algorithm, ou EM algorithm). Le fonctionnement de l'algorithme EM dans la cadre d'un mélange gaussien est tel que décrit ci-dessous. Soit $\mathbf{x} = (x_1, ..., x_n)$ nos données. Notons Z la matrice de taille $n \times k$ telle que $Z_{i,j}$ est la probabilité pour le su-

jet i d'être dans le cluster j. Ainsi nécessairement $\sum_{j=1}^{k} Z_{i,j} = 1$ et $\frac{\sum_{i=1}^{n} Z_{i,j}}{n} = \tau_j$. Pour effectuer l'algoritme EM, on commence par se choisir un vecteur initial de centres de variances et de poids des clusters $\theta^{(0)} = (\theta_1^{(0)}, ..., \theta_k^{(0)})$ où $\theta_j^{(0)} = (\mu_j^{(0)}, \Sigma_j^{(0)}, \tau_j^{(0)})$. Puis on iter de la façon suivante, à l'itération l, on calcule, sachant $\theta^{(l)}$ et $\mathbf{x}$, l'espérance pour le sujet i d'être dans le cluster j. C'est-à-dire qu'on calcule $t_{i,j} = E(Z_{i,j}|\mathbf{x}, \theta^{(l)})$. On effectue ce calcul grâce à la formule de Bayes $t_{i,j} = \frac{\tau_j^{(l)} f(x_i, \theta_j^{(l)})}{\sum_{m=1}^{k} \tau_m^{(l)} f(x_i, \tau_m^{(l)})}$. Ce calcul des valeurs $t_{i,j}$ est donc "l'étape d'esperance" de l'algorithme d'espérance-maximisation. "L'étape de maximisation", quand à elle, consiste à prendre comme valeur de $\theta^{(l+1)}$ la valeur de θ qui maximise la vraisemblance du modèle sachant $\mathbf{x}$ et les valeurs de $t_{i,j}$. C'est-à-dire $\theta^{(l+1)} = \text{argmax}_\theta \sum_{i=1}^{n} \sum_{j=1}^{k} t_{i,j} \log(\tau_j f(x_i, \theta_j))$. Cette optimisation s'effectue en prenant $\tau_j^{(l+1)} = \frac{1}{n} \sum_{i=1}^{n} t_{i,j}$, $\mu_j^{(l+1)} = \frac{\sum_{i=1}^{n} t_{i,j} x_i}{\sum_{i=1}^{n} t_{i,j}}$ et finalement $\sigma_j^{(l+1)} = \frac{\sum_{i=1}^{n} t_{i,j}(x_i - \mu_j)(x_i - \mu_j)^T}{\sum_{i=i}^{n} t_{i,j}}$

Pour appliquer concrètement un modèle de mélange gaussien à une base de données, on peut utiliser des bibliothèques spécialisées. Dans nos contributions, nous avons utilisé la fonction `GMM` de la bibliothèque `sklearn.mixture` `buitinck2013api` du langage `Python`. Notons aussi l'existence du projet Mixmod `lebret2015rmixmod` `Mixmod` qui propose des bibliothèques en Python (Pymixmod), R (Rmixmod), C++ (mixmodLib), ainsi que sa propre interface graphique (mixmodGUI).

2.1.2/ K-MEANS

L'algorithme des k-moyennes (ou en anglais k-means) est un algorithme de partitionnement de données proposé par Hugo Steinhaus en 1957 `steinhaus1956division`. Son fonctionnement est le suivant :

1. Choisir k points m_1, ..., m_k dans l'espace du nuage de points (par exemple la position de k points du nuage tirés au hasard). m_1, ..., m_k sont "les moyennes de nos k clusters". Évidement à cette étape ces moyennes sont généralement mal positionnées et il va falloir les améliorer petit à petit.
2. Créer les k clusters en assignant chaque point au cluster dont la moyenne est la plus proche de lui. Dit autrement, le j^{eme} cluster est constitué de tous les points qui sont plus proches de m_j que de m_l, $\forall l \neq j$.
3. Recalculer les k moyennes en prenant effectivement les moyennes des k clusters nouvellement créés. En d'autres termes, $m_j = \frac{1}{\#C_j} \sum_{x_i \in C_j} x_i$ où C_j désigne le j^{eme} cluster et $\#C_j$ désigne le nombre de sujets de C_j.
4. Recommencer les étapes 2 et 3 jusqu'à obtenir une convergence (*i.e.* jusqu'à ce que les clusters ne changent plus d'une itération sur l'autre).

Contrairement à la GMM, l'algorithme des k-moyennes est un algorithme non paramétrique. C'est-à-dire qu'il ne suppose pas que les clusters suivent une loi particulière. Pour autant il suppose que les clusters s'inscrivent dans des boules, ce qui est en fait une supposition assez proche de celle de la GMM. La différence principale vient du fait que,

dans le cas de l'algorithme des k-moyennes, c'est l'utilisateur qui doit fixer le nombre de clusters, là où, dans le cas de la GMM, il est possible de se fier à des critères statistiques tels que l'AIC où le BIC (cf. partie 2.2.2.2 pour la définition d'AIC et BIC).

2.1.3/ RÉDUCTION DE DIMENSION

En mathématique, une "réduction de dimension" est une opération qui consiste à remplacer des données d'un espace de grande dimension par des données d'un espace de dimension plus petite. C'est un sujet d'étude important des statistiques. En effet, dès lors que l'on dispose d'une base de données de n sujets pour p variables, les sujets peuvent être vus comme n points d'un espace à p dimensions. Une application évidente des réductions de dimension est que cela permet, pour les êtres humains que nous sommes, de visualiser les données. En effet, sur papier nous sommes en mesure de visualiser des données en deux dimensions. Sur un ordinateur, on peut éventuellement visualiser des données en trois dimensions en faisant pivoter l'image, mais guère plus. Réduire la dimension pour placer les données dans un espace de dimension deux ou trois permet donc de les rendre visualisables. Cependant, la réduction de dimension est également utilisée en tant qu'étape préliminaire au clustering. En effet, en grande dimension, les données deviennent généralement éparses, ce qui rend leur clustering compliqué. Ce phénomène, découvert par Richard Bellman en 1957 [bellman2013dynamic] est nommé "fléau de la dimension". Les deux chapitres suivants détaillent le fonctionnement de deux méthodes de réduction de dimension, l'analyse en composantes principales (ACP) et les Laplacian eigenmaps. L'ACP est la plus connue et la plus utilisée des méthodes de réduction de dimension, nous y avons eu recours plusieurs fois dans les contributions. Les Laplacian eigenmaps quant à elles, sont une méthode de réduction de dimension qui s'applique lorsque, pour chaque couple de sujets, on peut définir une similarité entre ces deux individus. Cette méthode est à la base de notre contribution 1.

2.1.3.1/ ACP

L'analyse en composante principale est une très ancienne et éprouvée méthode de réduction de dimension dont les prémisses remontent à Karl Pearson en 1901 [pearson1901liii]. On considère $X = \begin{bmatrix} X_{1,1} & \dots & X_{1,p} \\ \vdots & & \vdots \\ X_{n,1} & \dots & X_{n,p} \end{bmatrix}$ une base de données à n sujets et p variables. $X_{1,1}...X_{n,1}$ représente le premier sujet et $X_{1,1}...X_{1,p}$ représente la première variable. Pour effectuer une ACP sur cette base, on applique les étapes suivantes :

1. On calcule la matrice de variance-covariance $C = \frac{1}{p}X^T X$. C est alors une matrice $p \times p$ symétrique.
2. On diagonalise C. Comme C est symétrique, cela est toujours possible. On obtient alors $C = P^{-1}DP$ où D est une matrice diagonale et P est la matrice de passage. Comme C est symétrique, P est une matrice orthogonale, c'est-à-dire que tous ses vecteurs colonnes sont orthogonaux. De plus, la matrice étant orthogonale, on a $P^{-1} = P^T$.
3. On appelle $u_1, u_2, ..., u_p$ les vecteurs propres de C (*i.e.* les vecteurs colonnes de P).

De toutes les droites vectorielles, celle générée par le vecteur u_1 est alors celle qui maximise la variance de la projection de X sur une droite. De toutes les droites vectorielles orthogonales à celle-ci, celle générée par u_2 est celle qui maximise la variance de la projection de X. Le plan engendré par u_1 et u_2 est le plan qui maximise la variance de la projection de X sur un plan. De même pour l'espace engendré par u_1, u_2 et u_3 etc.

En général, en statistique, on n'applique pas l'ACP sur les données brutes, mais plutôt sur les données centrées réduites $\overline{X} = \begin{bmatrix} \frac{X_{1,1}-\overline{X}_1}{\sigma X_1} & \cdots & \frac{X_{1,p}-\overline{X}_p}{\sigma X_p} \\ \vdots & & \vdots \\ \frac{X_{1,1}-\overline{X}_1}{\sigma X_1} & \cdots & \frac{X_{n,p}-\overline{X}_p}{\sigma X_p} \end{bmatrix}$ où $\overline{X}_i$ est la moyenne de la variable i et σX_i est l'écart type de la variable i. Cette étape est nécessaire dès lors que l'on considère que chaque variable doit avoir le même "poids" dans la construction de l'espace dans lequel les données sont projetées. On effectue donc cette étape de centrer et réduire les données avant la première étape de l'ACP.

Une fois l'ACP effectuée, on peut éventuellement visualiser les données dans le plan engendré par u_1 et u_2, ce qui donnera un nuage de points en deux dimensions du type du graphique 2.2.

2.1.3.2/ LAPLACIAN EIGENMAPS

La méthode des Laplacian eigenmaps est une méthode de réduction de dimension proposée par Mikhail Belkin et Partha Niyogi en 2001 belkin2001laplacian. Cette méthode fonctionne dans le cas où il est possible, pour tout couple de sujets, de proposer une valeur de similarité entre ces deux sujets. La première étape des Laplacian eigenmaps consiste donc à construire une matrice de similarité W telle que pour tout i et j inférieurs à n, $W_{i,j}$ est la similarité entre le i^{eme} et le j^{eme} sujet. Cette similarité est une valeur d'autant plus grande que les sujets sont "proches" au sens de la métrique choisie. La deuxième étape est de construire la matrice Laplacienne $L = D - W$ où D est la "matrice des degrés" c'est-à-dire la matrice diagonale qui vérifie $\forall i \in [\![1, n]\!], D_{i,i} = \sum_{j=1}^{n} W_{i,j}$. Notons qu'à ce stade, on peut également préférer utiliser la Laplacian normalisée $L = D^{-1/2}(D - W)D^{-1/2}$ comme proposé par Fan Chen en 2007 chen2007resistance. La troisième étape est de diagonaliser cette matrice L. Appelons alors $\phi_1, \phi_2, ..., \phi_n$ les n vecteurs propres de L. Pour tout i dans $1, ..., n$, l'image du sujet i dans $\mathbb{R}^k$ est alors le vecteur $(\phi_2(i), ..., \phi_{k+1}(i))$ où $\phi_2(i)$ désigne la i^{eme} composante du vecteur ϕ_2. k désigne ici la dimension de l'espace dans lequel on veut plonger les données. On peut par exemple prendre $k = 2$ pour visualiser les données sur un plan et obtenir ainsi un schéma du type de la figure 2.2. Cependant, comme indiqué plus haut, on peut également utiliser la réduction de dimension comme étape préliminaire à du clustering pour éviter le fléau de la dimension. Dans ce cas, k peut être plus élevé. Nous avons utilisé les Laplacian eigenmaps en tant qu'étape préliminaire à du clustering de séquences génétiques dans notre contribution 1. Cette méthode s'y prête bien dans ce cas, puisqu'il est possible de définir une similarité entre séquences grâce aux scores associés aux méthodes d'alignements comme ceux définis en partie 1.2. Pour une revue plus détaillée et précise des différentes méthodes de clustering, on pourra se référer à jacques2014functional.

2.2/ RÉGRESSION ET SÉLECTION DE VARIABLES

2.2.1/ MODÈLES DE RÉGRESSION

D'une manière générale on appelle "modèle de régression" tout modèle visant à prédire la valeur d'une variable (appelée "variable à expliquer") en fonction d'une ou plusieurs autres (appelées "variables explicatives"). Par exemple, si vous souhaitez prédire le poids d'un chat en fonction de sa race, de la marque de ses croquettes et de l'âge de son ou sa propriétaire, vous allez avoir besoin d'un modèle de régression. Les parties 2.2.1.1, 2.2.1.2 et 2.2.1.3 décrivent respectivement la régression linéaire, la régression logistique et la régression logistique ordonnée. Ces trois parties ne traitent donc pas de tous les modèles de régression possibles, pour cause il y en a une infinité, mais présentent deux des plus usuels (la régression linéaire et la régression logistique) ainsi qu'un modèle sur lequel nous avons travaillé (la régression logistique ordonnée). Dans les parties 2.2.1.1, 2.2.1.2 et 2.2.1.3, n désignera toujours le nombre de sujets et p désignera toujours le nombre de variables explicatives.

2.2.1.1/ RÉGRESSION LINÉAIRE

La régression linéaire est le plus ancien et le plus usuel des modèles de régression. On retrouve des calculs de coefficients de régressions linéaires dans les travaux de Ruđer Josip Bošković en 1755-1757 `kusters2008dodge`. Elle s'applique dans le cas où la variable à expliquer est quantitative. Elle repose sur la supposition que la variable à expliquer est égale à une combinaison linéaire des variables explicatives, plus des variations non expliquées qu'on appelle "le bruit" (ou "les erreurs", ou encore "les résidus"). La formulation mathématique de ce modèle est donc : $Y_i = \beta_0 + \beta_1 X_{i,1} + \beta_2 X_{i,2} + ... + \beta_p X_{i,p} + \epsilon_i$ où Y_i est la valeur de la variable à expliquer pour le i^{eme} sujet, $X_{i,1}...X_{i,p}$ sont les valeurs des variables explicatives pour le i^{eme} sujet et ϵ_i est le bruit associé au i^{eme} sujet. $\beta_0, \beta_1, ..., \beta_p$ sont donc les paramètres du modèle (à déterminer). On peut aussi utiliser l'écriture matricielle suivante : $Y = \beta X + \epsilon$ où Y est le vecteur de taille n qui représente les valeurs de la variable à expliquer pour tous les sujets, $\beta = (\beta_0, \beta_1, ..., \beta_p)$ est le vecteur des paramètres à estimer (c'est donc un vecteur de taille p+1), X est la matrice de taille $n \times p+1$ dont les lignes représentent les sujets et les colonnes représentent les variables explicatives, la première colonne étant uniquement composée de 1 afin d'inclure la composante constante (*i.e.* β_0) dans le modèle.

En général, on assume que le bruit ϵ suit une loi normale centrée $N(0, \sigma^2 I)$ dans laquelle la variance σ^2 est à déterminer. Dans ce cas, maximiser la vraisemblance du modèle revient à minimiser la somme des carrés des composantes de ϵ (aussi appelée somme des carrés des erreurs). On utilise alors la méthode des moindres carrés pour estimer les paramètres du modèle.

A titre d'exemple, la figure 2.4 représente le prix de 546 maisons vendues à Winsor (Canada) en fonction de leur superficie `anglin1996semiparametric`. Le modèle obtenu est ici Prix $= 34136 + 6.5988 \times$ Superficie. c'est-à-dire que, dans ce cas, $\beta_0 = 34136$, $\beta_1 = 6.5988$, X_i est la superficie de la i^{eme} maison et Y_i est le prix de cette i^{eme} maison.

On note sur la figure 2.4 que la variance résiduelle est importante. Les points du nuage de points sont loin d'être alignés sur la droite de régression. Ceci vient simplement du fait que la superficie n'explique pas complètement le prix d'une maison. Pour améliorer le modèle,

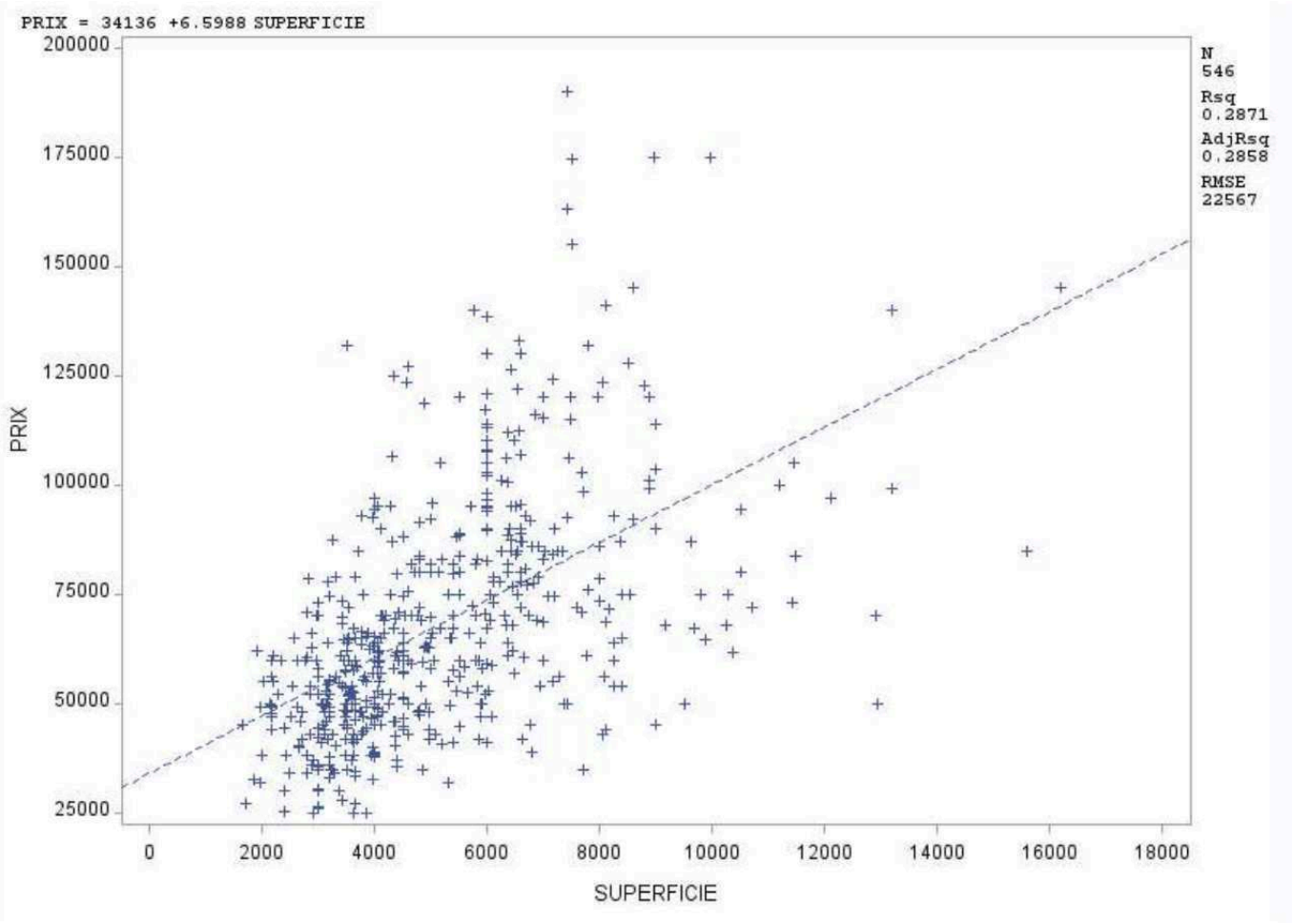

FIGURE 2.4 – Prix des maisons vendues à Winsor (Canada) en fonction de la superficie

il faut inclure d'autres variables pertinentes telles que le nombre de chambres, l'accès ou non au gaz, le nombre de salles de bain, etc. Cependant, seules les régressions linéaires univariées (i.e. avec une seule variable explicative) peuvent être présentées sur une figure en deux dimensions comme la figure 2.4.

2.2.1.2/ RÉGRESSION LOGISTIQUE

La régression logistique est un modèle de régression qui s'applique lorsque la variable à expliquer est binaire. Typiquement, il peut s'agir de savoir si un individu est malade ou sain, vivant ou décédé, etc. On note généralement 0 et 1 les deux états possibles. L'hypothèse principale de la régression logistique est que l'état de la variable à expliquer Y dépend d'une variable continue Y^* (non observée), aussi appelée "trait latent". On peut alors appliquer une régression linéaire sur ce trait latent. $Y_i^* = \beta_0 + \beta_1 X_{i,1} + \beta_2 X_{i,2} + ... + \beta_p X_{i,p} + \epsilon_i$. Ici ϵ est supposé suivre une loi logistique standard. Cette loi est une approximation de loi normale qui a pour avantage d'avoir une fonction de répartition définie explicitement. L'hypothèse de la régression logistique est que $Y_i = 0$ si et seulement si $Y_i^* < 0$, et donc $Y_i = 1$ si et seulement si $Y_i^* \geq 0$. Il en découle que $P(Y_i = 1) = \Phi(\beta_0 + \beta_1 X_{i,1} + \beta_2 X_{i,2} + ... + \beta_p X_{i,p})$ où $\Phi : x \to \frac{1}{1+e^{-x}}$ est la fonction de répartition

de loi logistique standard. En d'autres termes, la probabilité que Y_i soit égal à 1 est d'autant plus grande que $\beta_0 + \beta_1 X_{i,1} + \beta_2 X_{i,2} + ... + \beta_p X_{i,p}$, aussi appelé "le prédicteur linéaire", est grand. Pour finir sur la régression logistique, notons juste qu'il n'y pas besoin ici de calculer de variance pour le bruit ϵ, contrairement à la régression linéaire. En effet, si on considérait ϵ comme une loi logistique centrée de variance à déterminer, alors on aurait une situation de surparamétrage.

Si la régression logistique est un modèle relativement éprouvé, aujourd'hui encore des travaux sont effectués sur ce modèle. Citons par exemple `sur2018modern` qui propose entre autre une méthode pour déterminer le biais et la variance de l'estimateur de maximum de vraisemblance de ce modèle ainsi que la distribution du rapport de vraisemblance. Ce papier montre ainsi entre autre que l'idée "si j'ai k fois plus de sujets que de variables, l'estimation des paramètres va bien se passer" n'est pas forcement juste. Il détaille aussi entre autre les conditions pour que l'estimation des paramètres par maximum de vraisemblance soit possible quand p et n grandissent avec $k = \frac{n}{p} n$ fixé. Ces conditions dépendent de k et de la "puissance de signal" $\gamma = \sqrt{\lim_{n,p\to\infty} \frac{\|\beta\|_2^2}{n}}$.

2.2.1.3/ RÉGRESSION LOGISTIQUE ORDONNÉE

La régression logistique ordonnée est un modèle qui s'applique dans le cas où la variable à expliquer est qualitative ordonnée. Typiquement une tumeur qui aurait plusieurs degrés de gravité. On appellera "Q" le nombre de modalités possibles de la variable à expliquer et m_1, ..., m_Q les modalités elles-mêmes. Comme dans la régression logistique, on suppose que la variable à expliquer Y dépend d'une variable continue Y^* telle que $Y_i^* = \beta_1 X_{i,1} + \beta_2 X_{i,2} + ... + \beta_p X_{i,p} + \epsilon_i$, où ϵ suit une loi logistique. La différence avec une régression logistique est que ce modèle assume également l'existence de seuils $\gamma_0 = -\infty < \gamma_1 < ... < \gamma_Q = +\infty$ tels que $\forall q \in 1...Q, Y_i = m_q$ si et seulement si $Y_i^* \in]\gamma_{q-1}, \gamma_q[$. Notons au passage qu'il n'y a pas besoin ici de composante constante (*i.e.* β_0) dans la régression linéaire des variables explicatives sur le trait latent, car elle serait redondante avec les seuils et causerait donc un surparamétrage. Notons aussi que, comme $\beta_0 = -\infty$ et $\beta_Q = +\infty$, seuls $Q-1$ paramètres de seuil sont à estimer. Au final, ce modèle compte $p+Q-1$ paramètres en tout. C'est sur ce modèle de régression logistique ordonnée que nous nous penchons particulièrement dans le cadre de la contribution 4.

2.2.2/ SURINTERPRÉTATION ET SÉLECTION DE VARIABLES

2.2.2.1/ SURINTERPRÉTATION

En statistique, la surinterprétation désigne le fait de choisir un modèle trop "compliqué" par rapport aux données dont on dispose. Le terme "modèle compliqué" signifie ici un modèle nécessitant l'estimation de nombreux paramètres. Ce modèle compliqué permettra de s'ajuster parfaitement aux données dont on dispose mais se généralisera très mal à de nouvelles données et fournira de piètres prédictions. La figure 2.5 issue de `OverFitting` montre un exemple didactique de surinterprétation. Dans cet exemple, on dispose d'une évaluation du bien-être d'un couple à chacune des 10 premières années suivants leur mariage. On dispose donc de 10 données et d'une seule variable prédictive, le temps (noté t). Une idée pour coller parfaitement aux données pourrait alors être de

définir le bien-être comme une combinaison linéaire de t, t^2, ..., t^9. Ainsi, on obtiendrait le modèle de prédiction du bien être représenté par la courbe ondulante bleue (celle qui passe par tous les points). Selon ce modèle, le couple devrait connaître une période d'euphorie extatique juste après la dixième année. Le problème de ce modèle est que, si on modifie très légèrement une seule donnée, on peut obtenir le résultat complètement inverse et être amené à prédire pour ce couple une rapide et profonde dépression dès le passage de la dixième année effectué.

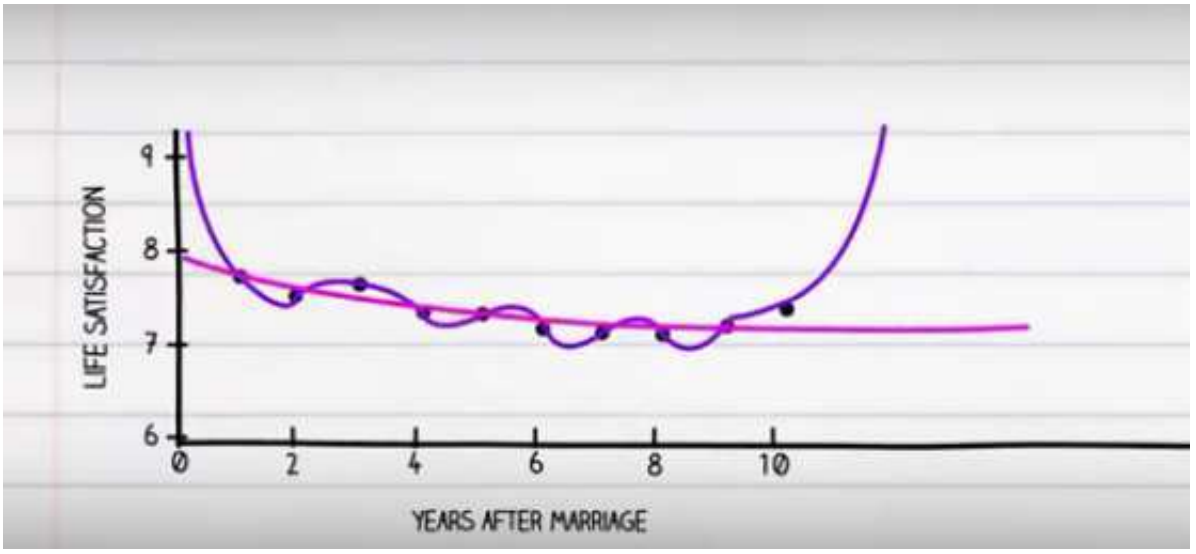

FIGURE 2.5 – Surinterprétation

De même l'ajout ou le retrait d'une seule donnée peuvent complètement modifier les prédictions. En d'autres termes, le modèle n'est pas robuste, c'est pourquoi il se généralise mal à de nouvelles données. On préférera alors généralement un modèle qui ajuste un peu moins bien les données mais plus robuste. Dans l'exemple de la figure 2.5, on préférera la courbe "du bas" qui représente une régression linéaire du bien-être en fonction de t et t^2. Évidemment, l'objectif n'est pas non plus de sacrifier complètement l'ajustement aux données au profit de la robustesse. Par exemple, dans la figure 2.5, un modèle qui prédirait toujours un bien-être de 0 sans s'occuper des données serait parfaitement robuste car non affecté par l'ajout, le retrait ou une légère modification d'une donnée. Pour autant, ce modèle ne s'ajusterait pas du tout aux données, on serait ici dans un cas de sous-interprétation, et de ce fait, ce modèle se généraliserait tout aussi mal que le modèle surinterprété. Un des principaux objectifs de la statistique est de trouver un équilibre entre l'ajustement et la robustesse des modèles, en déterminant le nombre adéquat de paramètres acceptables dans le modèle. L'idée générale est que, plus on dispose de données, plus on peut s'autoriser des modèles compliqués. Dans la pratique, on utilise des critères statistiques comme ceux définis aux chapitres 2.2.2.2.

L'exemple présenté dans la figure 2.5 a l'avantage d'être visualisable en deux dimensions, car on ne dispose ici que d'une seule variable explicative (le temps) et on provoque de la surinterpretation en intégrant différentes puissances de cette variable dans le modèle (t, t^2, ... , t^9). Dans la pratique, et particulièrement dans le cadre de la bio-informatique, le risque de surinterprétation est plutôt lié au fait que l'on dispose au départ d'un très grand nombre de variables. Par exemple, dans la contribution 4, on étudie le cas où l'on cherche

à prédire des niveaux de gravité d'une tumeur en fonction de niveaux d'expression d'un grand nombre de gènes. Dans cette situation, dès lors qu'on dispose de plus de gènes que de patients on peut facilement se retrouver en mesure de proposer un modèle qui ajuste parfaitement les données, quand bien même les gènes étudiés n'auraient en réalité aucun effet sur la tumeur. (Rmq : Il y a quelques conditions mathématiques à cela, il suffit de disposer d'au moins $n-1$ gènes tels que les vecteurs les représentant soient "libres" (cf. glossaire) et que les vecteurs représentant les patients soient également libres).

2.2.2.2/ AIC ET BIC

Une méthode simple et éprouvée pour éviter le phénomène de surinterprétation est de chercher à optimiser des "critères d'information". Ce type de critères est basé sur un compromis entre la qualité de l'ajustement (que l'on cherche à maximiser) et le nombre de paramètres du modèle (que l'on cherche à minimiser). Parmi cela, le critère d'information d'Akaike (en anglais Akaike information criterion ou AIC), développé par Hirotugu Akaike en 1973 `akaike1998information`, est un des plus populaires. Sa formule est $AIC = 2k - 2\ln(L)$, où k est le nombre de paramètres du modèle et L et la vraisemblance ($\ln(L)$ est donc la log-vraisemblance du modèle). L'AIC décroît donc lorsque la vraisemblance du modèle croît et lorsque le nombre de paramètres décroît. L'objectif est alors de trouver, entre différents modèles candidats, celui dont l'AIC est le plus faible. En d'autres termes, le critère AIC "autorise l'utilisateur" à inclure un nouveau paramètre dès lors que ce paramètre permet d'ajouter au moins 1 à la log-vraisemblance du modèle (c'est-à-dire de multiplier la vraisemblance du modèle par e).

Un autre "critère d'information", apparu un peu plus tard mais tout aussi populaire, est le critère d'information bayésien (en anglais Bayesian Information Criterion ou BIC) `schwarz1978estimating`. Ce critère est très proche de l'AIC mais tient compte du nombre de sujets dans la pénalisation. Plus précisément, $BIC = \ln(n) \times k - 2\ln(L)$ où n est le nombre de sujets. Dans ce cas, on est "autorisé" à inclure un nouveau paramètre dans le modèle dès lors que ce nouveau paramètre permet d'ajouter $(\frac{\ln(n)}{2})$ à la log-vraisemblance, c'est-à-dire permet de multiplier la vraisemblance par $\sqrt{n}$. La pénalité du BIC est plus forte que celle de l'AIC dès lors que $n > e^2$, c'est-à-dire dès lors que $n > 8$, ce qui est quasiment toujours le cas en pratique. De ce fait, le BIC est plus sélectif que l'AIC dans le sens où il choisit des modèles avec moins de variables explicatives.

2.2.2.3/ PROCÉDURES STEPWISE

Dans le cadre d'une régression, si on dispose de p variables explicatives potentielles, on peut alors envisager 2^p modèles différents. Rapidement, il devient inenvisageable de tester l'AIC où le BIC pour chacun de ces modèles. Les approches standard pour traiter cela sont des procédures par étapes (stepwise) comme la sélection ascendante (Forward selection) ou la sélection descendante (Backward elimination).

Le principe de la sélection ascendante est de partir du modèle nul (*i.e.* modèle qui n'inclut aucune variable explicative), puis de voir si le meilleur modèle à une variable explicative améliore le critère choisi. Si c'est le cas, appelons v_1 la meilleure variable possible pour un modèle univarié. On teste alors tous les modèles à deux variables incluant v_1 et une autre variable. On incrémente ainsi successivement des variables tant que cela permet d'améliorer le critère choisi.

Le principe de la sélection descendante, au contraire, est de partir du modèle complet (modèle qui inclut toutes les variables explicatives potentielles), puis de voir si l'un des modèles à $p-1$ variables est meilleur. Puis ainsi de suite, on retire une variable tant que cela permet d'améliorer le critère choisi.

La procédure ascendante, comme la procédure descendante, peut conduire à estimer $\frac{p\times(p+1)}{2}$ modèles au total. On utilise donc l'une ou l'autre selon qu'on suppose que le meilleur modèle inclura un nombre de variables plutôt bas (proche de 0) ou plutôt haut (proche de p). Notons toutefois que les modèles avec beaucoup de variables sont plus longs à calculer que les modèles avec peu de variables. Il est donc raisonnable de préférer la sélection ascendante dans le doute. Notons également que la sélection descendante est impossible en général lorsque $p > n$, car tous les modèles à $p-1$ éléments permettent un ajustement parfait aux données : on ne peut pas les distinguer.

2.2.2.4/ RÉGULARISATION PAR NORME ℓ_1

Dans la partie 2.2.2.2, l'AIC a été défini comme $AIC = 2k - 2\ln(L)$ où k est le nombre de paramètres du modèle. Une manière parfaitement équivalente d'écrire cela dans le cadre d'une régression linéaire est $AIC = 2\|\beta\|_0 - 2\ln(L)$ où $\|\beta\|_0$ est la norme ℓ_0 de β, c'est-à-dire le nombre de composantes non nulles. Cette écriture permet de mieux mettre en évidence que l'AIC est purement une fonction de β puisque à X et Y donnés, L est lui-même une fonction de β. Dès lors, on peut se demander s'il ne serait pas possible d'optimiser cette fonction $AIC(\beta) = 2\|\beta\|_0 - 2\ln(L(\beta))$ sur l'espace $\mathbf{R}^{p+1}$ des valeurs possibles de β. En fait, ceci n'est pas possible directement, car cette fonction $AIC(\beta)$ n'est pas convexe sur $\mathbf{R}^{p+1}$ (ni continue).

En revanche, $\forall\lambda \in \mathbf{R}$ la fonction $f(\beta) = \lambda\|\beta\|_1 - \ln(L(\beta))$ est quant à elle convexe sur $\mathbf{R}^{p+1}$. $\|\beta\|_1$ définit la norme ℓ_1 de β, c'est-à-dire $\sum_{k=1}^{k=p}|\beta_k|$. Étant convexe, cette fonction est simple à optimiser via l'algorithme du gradian `cauchy1847methode` et ses variantes telles l'algorithme de Nesterov `Nesterov:MathProg05` ou l'algorithme de Frank-Wolfe `frank1956algorithm`. Cette idée de pénaliser la log-vraisemblance par la norme ℓ_1 plutôt que par la norme ℓ_0 est la base de la méthode du LASSO (Least Absolute Shrinkage and Selection Operator), créé par Robert Tibshirani en 1996 `Tibshirani:JRSSB96`. Dans cette formule, le paramètre λ fixe l'importance de la pénalité. Plus ce paramètre est élevé, plus la pénalité est élevée, et donc moins il y aura de variables explicatives sélectionnées. Il n'y a pas de consensus sur une façon simple et rapide de choisir ce paramètre λ. Actuellement, en pratique, le LASSO est effectué pour différentes valeurs de λ et on peut comparer les résultats obtenus pour un critère donné (comme AIC ou BIC), ou par de la validation croisée. À la base, cette méthode du LASSO s'applique pour des régressions linéaires ; toutefois, cette idée de pénaliser la log-vraisemblance par la norme ℓ_1 peut s'appliquer à différents autres modèles. L'enjeu de la contribution 4 est l'application de cette pénalisation par la norme ℓ_1 à la régression polytomique ordonnée, ainsi que la recherche de la meilleure méthode de choix de λ, en utilisant notamment des méthodes assez récentes pour ce choix de λ comme le "Quantile Universal Thresholding" `giacobino2015quantile`, ou le "Online Frank-Wolfe algorithm" `chretien2018hedging`.

2.3/ COURBES ROC

La courbe ROC (receiver operating characteristic) est un outil permettant de mesurer la performance d'un classificateur binaire. Dans ce modèle, la variable à expliquer est une variable binaire et la variable explicative est une variable continue. On définit comme "positifs" les sujets de l'une des deux catégories de la variable à expliquer et comme "négatifs" les sujets de l'autre catégorie. Le postulat du modèle est que, au-dessus d'un certain seuil de la variable explicative, les sujets sont supposés positifs, et en dessous de ce seuil, les sujets sont supposés négatifs. Ainsi on appelle "vrais positifs" les sujets supposés positifs qui sont effectivement positifs, "faux positifs" les sujets supposés positifs qui sont en fait négatifs, "vrais négatifs" les sujets supposés négatifs qui le sont effectivement et "faux négatifs" les sujets supposés négatifs qui sont en fait positifs. On nommera "taux de vrais positifs" ou "sensibilité" la proportion de vrais positifs parmi les sujets positifs, "taux de vrais négatifs" ou "spécificité" la proportion de vrais négatifs parmi les sujets négatifs et "taux de faux positifs" la proportion de faux positifs parmi les sujets négatifs. Une courbe ROC est un graphique sur lequel sont affichés les taux de vrais positifs et de faux positifs pour tous les seuils possibles. Le graphique 2.6 est une courbe ROC qui représente la capacité de la bactérie *Rhodococcus* à déterminer si l'on se trouve on non dans une zone polluée dans le cadre de l'étude `zappelini2015diversity`.

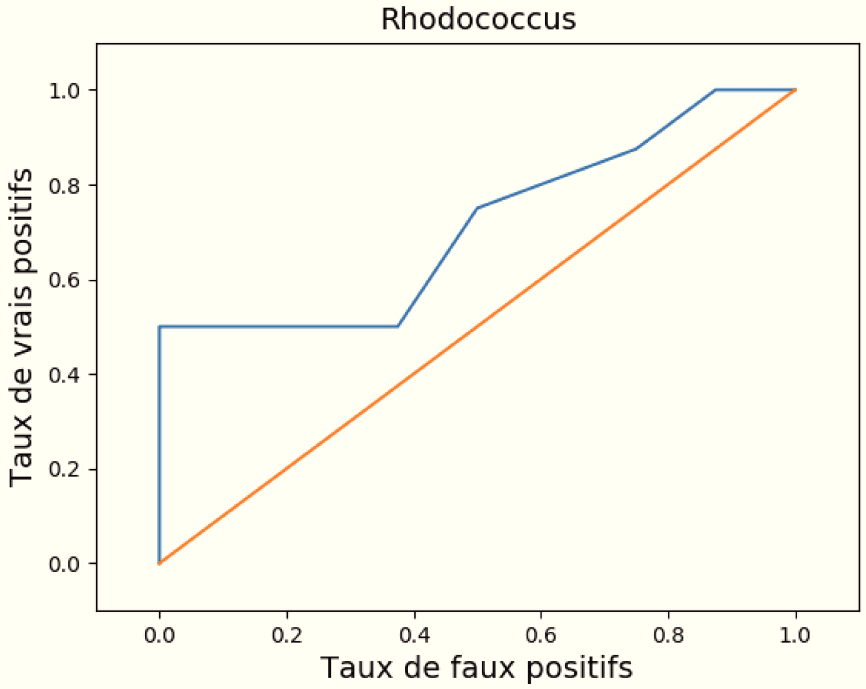

FIGURE 2.6 – Courbe ROC. Capacité de *Rhodococcus* à indiquer si l'on se trouve ou non sur la zone polluée.

Une courbe ROC démarre toujours au point (0,0) et se termine toujours au point (1,1). Elle est toujours croissante (au sens large, c'est-à-dire soit croissante soit stable). L'idéal du point de vue de la qualité du prédicteur étant que la courbe passe par le point (0,1). Dans ce cas, cela signifie qu'il existe un seuil tel que le taux de faux positifs est nul et le taux de vrais positifs est égal à 1. c'est-à-dire qu'il existe un seuil qui classe parfaite-

ment tous les sujets. Pour avoir une idée de la capacité globale de la variable prédictive à classer correctement la variable binaire, on calcule généralement l'aire sous la courbe (en anglais area under the curve ou AUC). Cette aire est d'autant plus grande que la variable prédictive permet de classer correctement la variable binaire. Cette aire vaut 1 si la variable prédictive permet de classer parfaitement la variable binaire. Dans la contribution 3, nous développons un outil pour appliquer l'analyse par courbes ROC à des données métagénomiques en écologie. À titre d'exemple nous les utilisons pour retrouver les bactéries et champignons les plus affectés par la pollution dans le cadre de l'étude `zappelini2015diversity`.

II

CONTRIBUTIONS

1

CLUSTERING

Cette partie présente les travaux que nous avons effectués sur le clustering de séquences. Dans ces travaux nous utilisons une combinaison de Laplacian eigenmaps `belkin2001laplacian` et modèles de mélanges gaussiens `day1969estimating` pour effectuer ce clustering. Pour tester ces travaux, nous avons utilisé une base de données composée de séquences génétiques de *Plathelminthes* (figure 1.1) et de *Nematodes* (figure 1.2) ; c'est-à-dire respectivement des vers plats et des vers ronds.

FIGURE 1.1 – Plathelminthes. Source : Richard Ling pour https://fr.wikipedia.org (*Pseudoceros dimidiatus*).

Les parties de "Abstract" à la partie 1.4.3 correspondent à l'article publié dans "Computers in Biology and Medicine" en février 2018 sous le titre "A clustering package for nucleotide sequences using Laplacian Eigenmaps and Gaussian Mixture Model" qui détaille les processus et les résultats de nos travaux de clustering.

FIGURE 1.2 – Nematodes. Source : United States Department of Agriculture.

ABSTRACT

In this article, a new Python package for nucleotide sequences clustering is proposed. This package, freely available on-line, implements a Laplacian eigenmap embedding and a Gaussian Mixture Model for DNA clustering. It takes nucleotide sequences as input, and produces the optimal number of clusters along with a relevant visualization. Despite the fact that we did not optimise the computational speed, our method still performs reasonably well in practice. Our focus was mainly on data analytics and accuracy and as a result, our approach outperforms the state of the art, even in the case of divergent sequences. Furthermore, an a priori knowledge on the number of clusters is not required here. For the sake of illustration, this method is applied on a set of 100 DNA sequences taken from the mitochondrially encoded NADH dehydrogenase 3 (ND3) gene, extracted from a collection of *Platyhelminthes* and *Nematoda* species. The resulting clusters are tightly consistent with the phylogenetic tree computed using a maximum likelihood approach on gene alignment. They are coherent too with the NCBI taxonomy. Further test results based on synthesized data are then provided, showing that the proposed approach is better able to recover the clusters than the most widely used software, namely Cd-hit-est and BLASTClust.

1.1/ INTRODUCTION

As the amount of available genetic sequences increases drastically every year, accurate methods to deeply study them are strongly demanded `vgrm+15:ij`. Among these methods, clustering is a very powerful tool that helps to understand relations between sequences. It can be used, for instance, to classify 16S RNA sequences into OTUs `hao2011clustering`, which are standard proxies for microbial species (clustering is here a way to identify an a priori unknown number of “species”). Clustering is used too to define taxa within groups of species represented by their DNA sequences. Other utilizations of sequence clustering in genomics encompass the study of sub-populations within the same species `torroni1992native`, the discovery of possible hidden variables that

can explain differences between such sub-populations, and so on suzek2007uniref.

However, most of the times, only generic methods for clustering a matrix of similarity scores is applied, and methods that are more specific to DNA sequences are still waited. Furthermore, the few existing methods that focus on sequence clustering mainly target on speed and scalability, and they need a strong similarity between sequences to produce accurate clusters. Furthermore, an a priori knowledge on the number of clusters is required, for instance by providing a similarity threshold cutoff. The objective of this article is to provide a new method that relaxes such constrains, allowing the sequences to be really divergent, and returning an a posteriori of the optimal number of clusters in an efficient manner.

One important issue in any clustering procedure is to find an appropriate embedding of the data under study, that will make the respective salient features in each of the groups more clearly delineateable. However, as often stated in the Machine Learning (ML) literature, high dimensional data are often much too scattered in order for an off-the-shelf method to be able to work properly. This phenomenon, often referred as the "curse of dimensionality" bellman2013dynamic, bellman2015adaptive, explains why several embeddings have been proposed recently. Some of these embeddings are very over-parametrized and can thus only be implemented in the supervised setting. This is the case for methods based on neural networks (*e.g.*, auto-encoders). Other methods need less parameters and are more suitable to unsupervised learning, which is the case of our proposal. Among many nonlinear embedding methods, the Laplacian Eigenmap belkin2001laplacian approach has been extensively studied from both the theoretical and the application viewpoint spielman2009spectral. More involved methods relying on Semi-Definite programming have also appeared recently with higher separation power in practice than spectral methods, see *e.g.*, chretien2016semi.

In this article, the aim is to establish the practical efficiency, for DNA sequence clustering, of the combination of a plain Laplacian Eigenmap approach coupled with a Gaussian Mixture based clustering. This particular choice is motivated by its remarkable computational efficiency, even in the case where the objects to classify are really divergent. The overall procedure can be divided in three steps.

1. Compute a similarity matrix between each pair of DNA sequences, *i.e.*, provide a matrix W of size $n \times n$, where n is the number of sequences, which is such that $W_{i,j}$ increases with the "similarity" between sequences number i and j.
2. Diagonalise the Laplacian matrix of W. By such an operation, DNA sequences are mapped to elements of a given vector space, whose dimension is much smaller than the sequence lengths. This reduction of the problem dimension is a key element that usually has a great impact on both visualization and clustering. The combination of the two stages above is often referred as the "Laplacian eigenmap" approach.
3. Cluster the transformed data using a Gaussian Mixture Model (GMM day1969estimating).

By using a ready to use Python package designed at this occasion, and freely available on-line, we have demonstrated the accuracy and efficiency of this approach for DNA sequence clustering. Indeed, the methodology has firstly been tested on a sample of 100 ND3 genes (DNA sequences) from *Platyhelminthes* and *Nematoda* species that have been downloaded from the NCBI website NCBI. The classification obtained via this approach has been compared with the phylogenetic tree of these species obtained by a

likelihood maximization method using PhyML [guindon2005phyml]. Obtained clusters are consistent with both clades appearing in the phylogenetic tree and the NCBI taxonomy. In particular, this clustering perfectly separates the *Nematoda* and *Platyhelminthes* phyla.

To evaluate the method further, extensive simulation experiments have secondly been run on synthesized data : n DNA sequences have been randomly generated, and random mutations and block deletions have been applied on them, leading finally to $N > n$ sequences. The objective was then to group these N sequences within n clusters (n is not a priori known), in such a way that all the elements in each cluster are originated from the same initial DNA sequence. On this set of synthesized data, our proposal has outperformed the two other state-of-the-art software for DNA clustering, namely Cd-hit-est and BLASTClust.

The remainder of this article is structured as follows. The three stages of the proposed method are detailed in the next section. Numerical results are then presented : the application example involving a real dataset is provided in Section 1.3.1, while the simulation based procedure is detailed in Section 1.3.2. A general discussion about the proposed approach is provided in Section 4.4. This research work ends by a conclusion section, in which the contribution is summarized and intended future work is provided.

1.2/ THE CLUSTERING METHOD

1.2.1/ LAPLACIAN EIGENMAP

The so-called Laplacian Eigenmap [belkin2001laplacian] is an original method for embedding data living in a structured set into a k-dimension vector space. The main information needed to compute the eigenmap is a matrix containing the value of the measured similarity between pairs of data. The main motivation for such embeddings is dimension reduction and visualization. Moreover, spectral methods often exhibit the nice property of separating clusters.

1.2.1.1/ THE MATRIX OF SIMILARITY

The first step in the construction of a good embedding is the creation of a similarity matrix W. This matrix is intended to measure the similarity between each pair of sequences by providing a number ranging between 0 and 1. The main assumption on W is that the greater the similarity is, the closer are the sequences to each other.

In order to create this similarity matrix, a multiple global alignment of the DNA sequences is first run using the MUSCLE (Multiple Sequence Comparison by Log-Expectation [edgar2004muscle]) software. Then, an ad hoc "Needleman Wunsch distance" [needleman1970general] is computed for each pair of aligned sequence, and with the "EDNAFULL" scoring matrix. This distance takes into account that DNA sequences usually face (1) mutations and (2) insertion/deletion. Note that, by using MUSCLE as first stage of this matrix computation, we operate only one (multiple) sequence alignment, instead of $\frac{n(n-1)}{2}$ (pairwise) alignments in the classical Needleman Wunsch algorithm (that usually contains two stages : finding the best pairwise alignment, and then compute the edit distance). Introducing Muscle leads to a real acceleration in the construction of the

similarity matrix.

Let us call M the distance matrix obtained by this way. M is then divided by the largest distance value, so that all its coefficients are between 0 and 1. W can finally be obtained as follows :

$$\forall\ i, j \in [\![1, n]\!],\ W_{i,j} = 1 - M_{i,j},$$

in such a way that $W_{i,j}$ represents the similarity score between sequences i and j.

1.2.1.2/ OPERATIONS ON W

Once the similarity matrix has been constructed, the next step is to create the normalized Laplacian matrix, as follows `chen2007resistance` :

$$L = D^{-1/2}(D - W)D^{-1/2}$$

, where W is the similarity matrix defined previously and D is the degree matrix of W. That is to say, D is the diagonal matrix defined by :

$$\forall i \in [\![1, n]\!],\ D_{i,i} = \sum_{j=1}^{n} W_{i,j}\ .$$

L being symmetric and real, it is diagonalisable in a basis of pairwise orthogonal eigenvectors $\{\phi_1, ..., \phi_n\}$ associated with eigenvalues $0 = \lambda_1 \leqslant \lambda_2 \leqslant ... \leqslant \lambda_n$. The Laplacian Eigenmap consists in considering the following embedding function :

$$c_{k_1}(i) = \begin{pmatrix} \phi_2(i) \\ \phi_3(i) \\ \vdots \\ \phi_{k_1+1}(i) \end{pmatrix} \in \mathbb{R}^{k_1},$$

where $c_{k_1}(i)$ is the coordinate vector of the point corresponding to the i^{th} sequence. In other words, the coordinate vector of the point corresponding to the i^{th} sequence is composed of the i^{th} coordinate of each of the k_1 first eigenvectors, ordered according to the size of their eigenvalues.

The choice of the k_1 cutoff is a crucial step and one usually proceeds as follows. The ordered eigenvalues are plotted, and we stop at the index where the increase in the eigenvalue is negligible : the number of eigenvalues that are not discarded is k_1. For instance, in our program, we have chosen to set k_1 as the first time the difference between the k^{th} and $(k+1)^{th}$ value is lower than 0.01.

Note that W can be seen as a weighted adjacency matrix of a graph, where nodes are the DNA sequences while edges are labeled by the degree of affinity between their adjacent nodes. In the literature, the Laplacian matrix is often described as constructed from the weighted adjacency matrix of such a graph rather than constructed from a similarity matrix. These definitions are equivalent.

1.2.2/ GAUSSIAN MIXTURE BASED CLUSTERING

The final step is performed by applying Gaussian Mixture based clustering (GMM, day1969estimating) to the point cloud. Gaussian Mixture Models belong to the class of unsupervised learning schemes friedman2001elements, and allow to distribute the data points into different clusters without *a priori* assumption about the clusters interpretation. One of the very useful features of model based clustering is that the model allows to use information criteria in order to estimate the number of clusters using Akaike Information Criterion (AIC akaike1974new), Bayesian Information Criterion (BIC schwarz1978estimating), or Integrated Completed Likelihood (ICL biernacki2000assessing) well-known criteria. The mathematical assumption of a GMM is that the point cloud follows the distribution :

$$\sum_{i=1}^{k_2} \tau_i \, \mathcal{N}(\mu_i, \Sigma_i),$$

where k_2 is the number of clusters, τ_i is the probability for a point to be in cluster i, and $\mathcal{N}(\mu_i, \Sigma_i)$ is the normal distribution of mean μ_i and covariance matrix Σ_i. GMM parameters are computed with the Expectation-Maximization (EM) algorithm mclachlan2004finite. Notice that the EM algorithm may converge to singular distributions exponentially fast biernacki2003degeneracy. However, degenerate situations can be easily discarded and consistent estimators can be easily obtained in practice. Gaussian Mixture models are still a topic of current extensive research, both from the statistical perspective he2011laplacian, wang2014high and the computational one yi2015regularized.

The Bayesian Information Criterion has been chosen to determine the most relevant number of clusters k_2 to be considered. The BIC, which is a criterion for model selection, is defined as follows :

$$BIC = -2\ln(L) + \ln(n)p,$$

where L is the likelihood of the estimated model, n is the number of observations in the sample, and p is the number of model parameters. This criterion allows us to select a model whose validity is based on a balance between the value of the model's likelihood (fidelity term) and the number of parameters to estimate (complexity term). The likelihood of the model increases with k_2 as well as the number of parameters. The selected model will be by default the one that minimizes this criterion. In the proposed package, the user can also set the number of clusters manually.

1.2.3/ THE CLUSTERING SOFTWARE

The Python program corresponding to the algorithm described in this section is freely available online[1]. The main function of this package provides a clustering from nucleotide sequences. Its prototype meets the following canvas :

```
clustering = Gclust(liste, nbClusters='BIC', drawgraphs=True,
nbEVMethod = 'delta', nbEVCutOff = 'default',
```

1. https ://github.com/SergeMOULIN/clustering-tool-for-nucleotide-sequences-using-Laplacian-Eigenmaps-and-Gaussian-Mixture-Models

`AddToNamesOfOutputs = ''):`
where :

- `list` is an in-memory image of a fasta file containing the sequences associated with their names. The fasta file must be configured as the "ND3.fasta" file in our github repository.
- `nbClusters` is the number of clusters desired by the user. By default, the program applies the BIC criterion to determine it. The user may also choose to use the AIC criterion by writing "nbClusters = AIC".
- `drawgraphs` is an optional Boolean value to produce some graphics. If `drawgraphs = TRUE`, a two dimensional clustering of data is plotted (*cf.* Figures 1.6, 1.7, and 1.8), as well as the graphical representation of similarities (as in Figure 1.3).

— `nbEVMethod` is the method chosen to determine the number of considered eigenvectors k_1. The user can choose between 3 methods, usually reported in the literature.

  — If `nbEVMethod = 'delta'` then k_1 is the lowest value such that $\lambda_{k_1+1} - \lambda_{k_1} < \delta$ where δ is a constant to be fixed by the user. `'delta'` is the default method, and with $\delta = 0.01$.

  — If `nbEVMethod = 'energy'` then k_1 is the lowest value such that $\sum_{i=1}^{k_1} (\lambda_{max} - \lambda_i) \geq C \times \sum_{i=1}^{n} (\lambda_{max} - \lambda_i)$ where C is a constant to be chosen by the user (default C = 0.9)

  — If `nbEVMethod = 'log'` then k_1 is the rounded value of $log(n)$.

— `nbEVCutOff` is the constant δ if `nbEVMethod == 'delta'` or the constant C if `nbEVMethod == 'energy'`. By default, `nbEVCutOff = 0.01` if `nbEVMethod == 'delta'` and `nbEVCutOff` = 0.9 if `nbEVMethod == 'energy'`.

— `AddToNamesOfOutputs` is a character string that is inserted as prefix in the output filenames. By default, the file that contains the clustering is named "Clustering.txt" while the one that contains the graphical representation of the similarity matrix is named "similarity_matrix.png". If the user specifies that `AddToNamesOfOutputs = 'Job1'`, then these filenames become "Job1Clustering.txt" and "Job1similarity_matrix.png" respectively.

The `GClust` function can be, for instance, launched as follows within a Python script :

```
list = open('MyFileName.fasta').read()
groups = gclust(list, drawgGraphs=True)
```

The output is a list of k_2 lists, each list containing the references of the sequences grouped in a particular cluster.

1.2.4/ MODULE AND PACKAGE DEPENDENCIES

As explained in Section 1.2.1.1, the DNA sequence alignment software used during the similarity matrix stage is MUSCLE. More precisely, the "muscle_v38" function of `cogent` package has been used `knight2007pycogent`. The Gaussian Mixture Model, for its part, is performed using the `GMM` function of `sklearn.mixture` package `buitinck2013api`.

1.3/ NUMERICAL EVALUATIONS

1.3.1/ EVALUATION ON REAL GENOMIC DATA

The proposed method has been applied to a sample of 100 DNA sequences from the mitochondrial gene ND3 taken from various species of both *Platyhelminthes* and *Nematoda*. Figure 1.3 graphically displays the similarity matrix that was obtained from the data. As expected, the largest similarities are obtained on the diagonal, that is, when the sequences are compared to themselves. However, blocks seem to appear as well. This justifying the prior intuition that the embedding method separates the clusters if there happens to be more than one in the dataset.

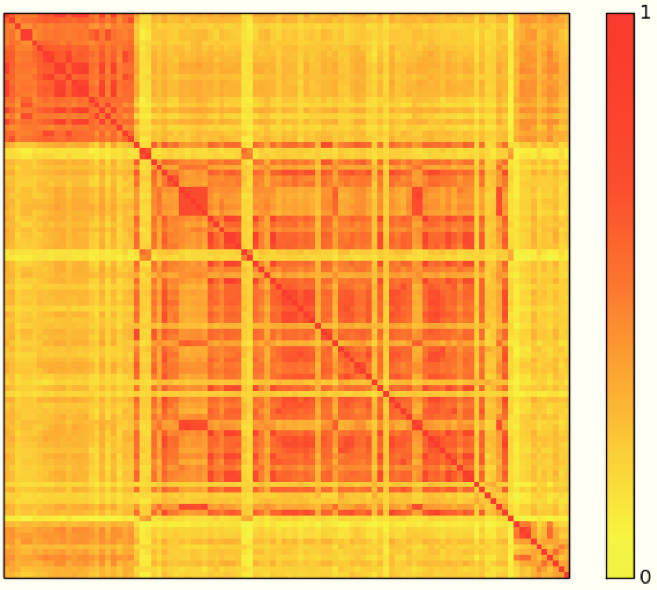

FIGURE 1.3 – Similarity matrix

Figure 1.4 displays the 14 first ascending eigenvalues, labeled λ_i, and satisfying $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_{14}$. According to the proposed methodology, we have chosen k_1 as the minimal index k such that $\lambda_{k+1} - \lambda_k$ is lower than or equal to 0.01. In this case study, we found $k_1 = 4$.

Figure 1.5 shows the Bayesian Information Criterion of the Gaussian Mixture Models for various number of clusters. One can see that this BIC reaches its minimum for $k_2 = 4$. Thus, the program automatically partitions the sequence dataset into four clusters.

Figures 1.6, 1.7, and 1.8 represent the point cloud divided in 4 clusters. This point cloud is projected into the plane formed by the first and second eigenvectors in Figure 1.6, to the one formed by the first and third eigenvectors in Figure 1.7, and to the plane formed by the second and the third eigenvectors in Figure 1.8. In these graphs, clusters 0, 1, 2, and 3 are represented in blue, red, yellow and cyan respectively.

A FASTA file was then written, in which each nucleotide sequence has been labeled according to its taxonomy and its cluster number. The taxonomies have been found

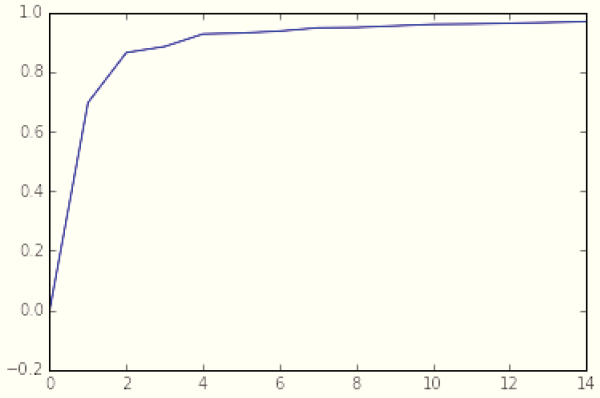

FIGURE 1.4 – Curve representing the first 14 eigenvalues

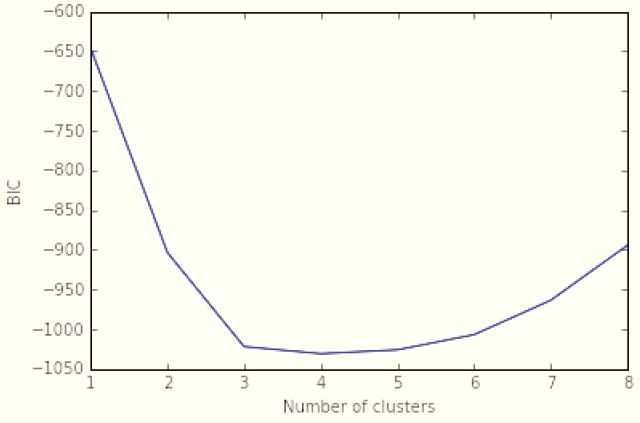

FIGURE 1.5 – Bayesian Information Criterion of the Gaussian Mixture Models

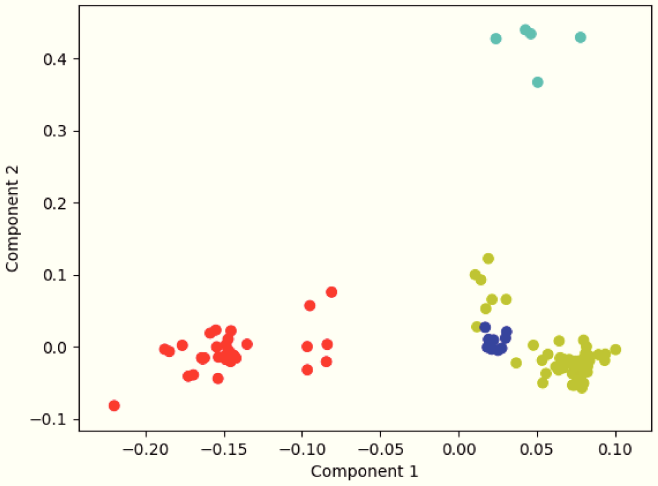

FIGURE 1.6 – GMM clustering in the plane formed by the eigenvectors 1 and 2

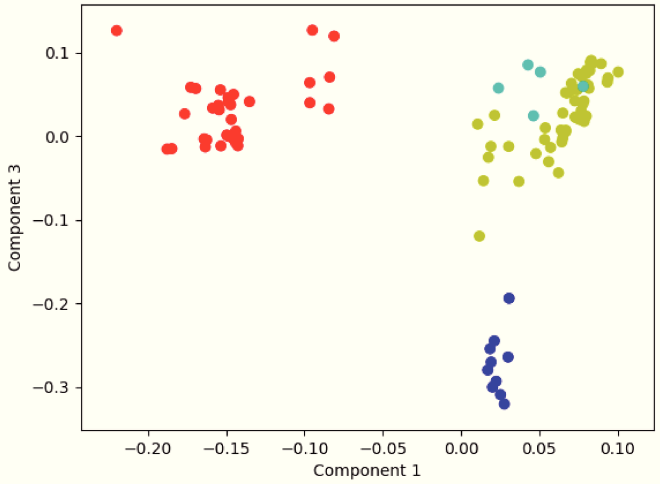

FIGURE 1.7 – GMM clustering in the plane formed by the eigenvectors 1 and 3

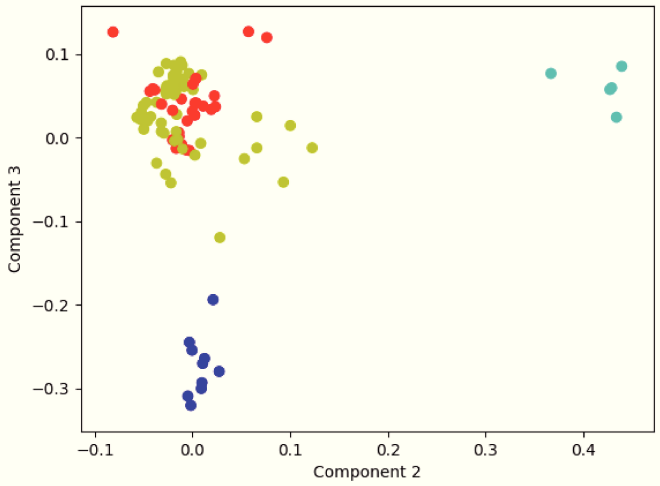

FIGURE 1.8 – GMM clustering in the plane formed by the eigenvectors 2 and 3

thanks to the `efetch` function (sub-package `Entrez`[2], Python package `Bio`). We then have used the online software *PhyML* (maximum likelihood method for phylogenetic tree reconstruction) with default options `guindon2005phyml`, to build a tree based on the same mitochondrial ND3 gene that we have used during clustering. The tree, whose leaves contain taxa names and cluster id, has been displayed using *FigTree* software `morariu08figtree`. This tree is depicted in Figures 1.9 and 1.10.

The `efetch` function provides 8 taxonomic levels for each species in our sample. In Figures 1.9 and 1.10, for readability reasons, we only show the taxonomy between levels 4 and 6. Note first that this clustering perfectly separates *Platyhelminthes* and *Nematoda* phyla. Indeed *Trematoda* and *Cestoda* are two classes of *Platyhelminthes*. Cluster 1 corresponds exactly to the *Platyhelminthes*, while Clusters 0, 2, and 3 represent the

2. A package that provides code to access NCBI over the world wide web.

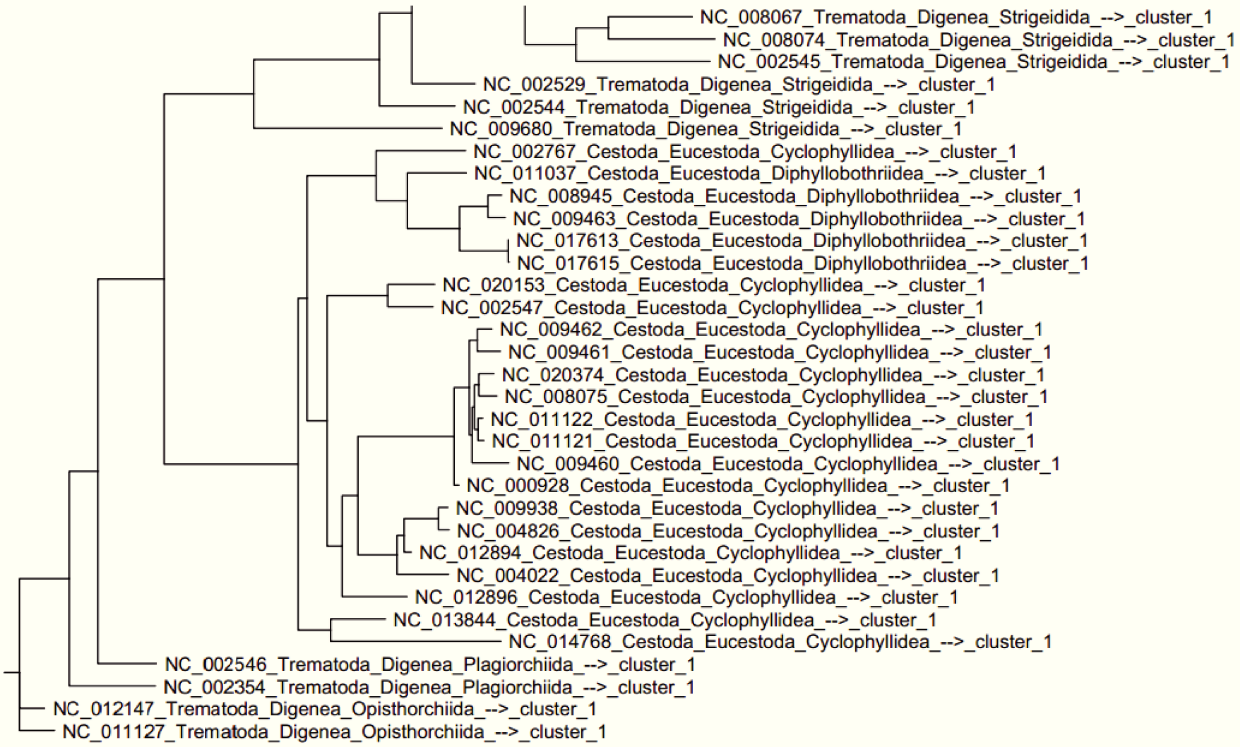

FIGURE 1.9 – First part of the phylogenetic tree (*Platyhelminthes*)

Nematoda. Cluster 0 is only composed of *Spirurida*, it contains 10 of the 12 members of this taxon. More precisely, when considering the seventh taxonomic level, we found that Cluster 0 contains nine *Filarioidea* and one *Thelazioidea*, while the *Spirurida* that are not in this cluster are a *Dracunculoidea* and a *Physalopteroidea*. Cluster 3, for its part, corresponds exactly with the *Trichocephalida* taxon.

Finally, in addition to recover taxonomy, the clustering agrees very well with the tree obtained via PhyML, as shown in Figures 1.9 and 1.10. Note that the taxonomy, based on morphology and nuclear genome, fully agrees with the PhyML tree and our clustering, which are both based on the mitochondrial genome. This fact suggests that this mitochondrial gene evolves like the nuclear genome.

1.3.2/ TESTS ON SIMULATED DATA

In addition to the analysis of real data processed in the previous section, we also performed tests on simulated data. This new series of tests allow us to assess the accuracy of our tool and to compare this accuracy with other existing tools. The data are simulated according to the following steps :

— We generate a “global root”. That is to say, a random word of 500 letters {A,T,C,G} representing a sequence of 500 nucleotides.
— From this global root, we generate 3 “cluster roots” which are each an evolution of the global root for which each nucleotide has mutated with probability 0.05.
— From each cluster root, we generate a cluster of 10 sequences. Each sequence is an evolution of the cluster root for which each nucleotide has mutated with proba-

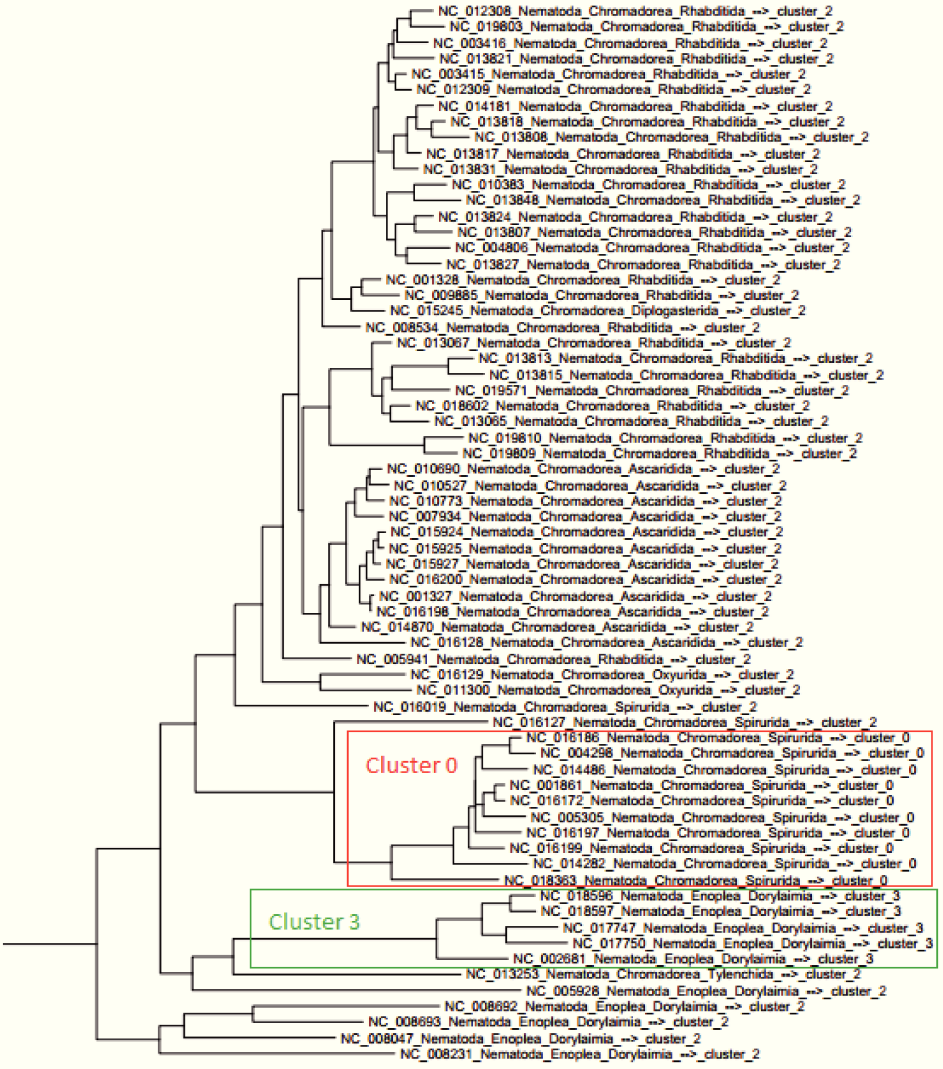

FIGURE 1.10 – Second part of the phylogenetic tree (*Nematoda*)

bility 0.05.

— Each of the obtained sequences undergoes a block deletion of 2.5% to 7.5% of its nucleotides with a probability of 0.5. That is to say, for each sequence :
  — A random draw is performed in order to decide on whether a block deletion is carried out or not. The deletion is carried out with probability 0.5.
  — A uniform random draw is performed to determine how many nucleotides are deleted between 2.5% and 7.5% of the sequence size.
  — A uniform random draw is performed in order to decide on where the block deletion starts.

The procedure defined above makes it possible to generate a simulated base of 30 sequences. We repeatedly applied this procedure 25 times with different random seeds (seed = 0 ... 24) in order to simulate 25 different databases. Figure 1.11 shows the similarity matrix obtained via these simulations (with a random seed of 0).

The clusters are sorted in each of the simulated databases (as can be seen on the similarity matrix). Thus, the ideal outcome we would like to obtain in applying our tool to these databases is, for example :

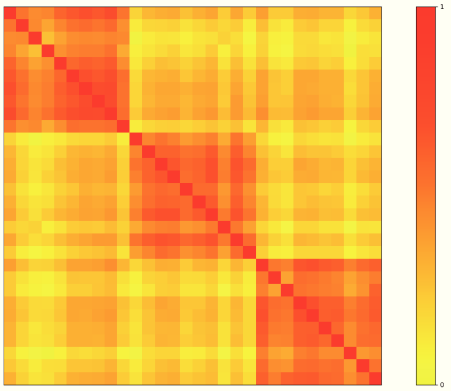

FIGURE 1.11 – Similarity matrix of the simulated clusters (seed = 0)

[['S11', 'S12', 'S13', 'S14', 'S15', 'S16', 'S17', 'S18', 'S19', 'S20'],

['S21', 'S22', 'S23', 'S24', 'S25', 'S26', 'S27', 'S28', 'S29', 'S30'],

['S1', 'S2', 'S3', 'S4', 'S5', 'S6', 'S7', 'S8', 'S9', 'S10']]

In order to evaluate the performance of our tool to find the right clustering, we have created a distance that corresponds to the number of sequences that need to move to get the perfect clustering. For instance, the following clustering :

[['S1', 'S2', 'S3', 'S4', 'S5', 'S6', 'S7', 'S8', 'S9', 'S10'],

['S21', 'S23', 'S24', 'S25', 'S26', 'S27', 'S28', 'S29', 'S30'],

['S11', 'S12', 'S13', 'S14', 'S15', 'S16', 'S17', 'S18', 'S19', 'S20'],

['S22']]

has a distance from the perfect clustering equal to 1.

Table 1.1 summarizes the distances obtained with our tool for each of the simulated database. This table also summarizes the distance obtained with the "CD-Hit-est" and "BlastClust" tools.

1.3.2.1/ TESTS ON SIMULATED DATA WITH OTHER TOOLS

"CD-Hit-est" and "BLASTClust" are two of the most used clustering tools for nucleotide sequences. According to `li2006cd` : "Cd-hit-est is a greedy incremental clustering algorithm. Briefly, the sequences are first sorted in decreasing length. The longest sequence becomes the representative of the first cluster. Then, each remaining sequence is compared with the representatives of the existing clusters. If the similarity with any representative is above a given threshold, it is grouped into that cluster. Otherwise, a new cluster is defined with that sequence as the representative." And, according to `BLASTClust` : "The program [BLASTClust] begins with pairwise matches and places a sequence in a cluster if the sequence matches at least one sequence already in the cluster. In the case (...) of nucleotide sequences, the Megablast algorithm is used."

random seed	Distance our tool	Distance CD-hit-est	Distance BlastClust
0	0	7	3
1	0	11	19
2	2	7	1
3	10	13	20
4	0	8	3
5	0	7	5
6	0	8	2
7	0	7	9
8	0	7	12
9	1	10	19
10	0	6	5
11	0	7	10
12	1	6	5
13	1	10	2
14	0	7	9
15	0	7	5
16	0	6	4
17	0	11	17
18	0	12	16
19	0	6	17
20	0	8	17
21	10	8	17
22	0	7	1
23	0	7	2
24	0	8	6
total	25	201	226

TABLE 1.1 – Distance from the perfect clustering

In both cases, the user has to provide a similarity threshold. In the case of CD-hit-est, this threshold indicates the minimum similarity that a sequence must have with the reference of a cluster to integrate this cluster. In the case of BlastClust, this threshold indicates the minimum similarity that a sequence must have with at least one other sequence in the cluster. This necessity to specify the threshold for similarity makes a important difference with our tool. Indeed, our tool determines the number of clusters automatically using BIC. Of course, CD-hit-est and BLASTClust provide a default value for this similarity (*e.g.*, 0.9 for CD-hit-est) but it is simple to find situations in which this value is not adapted. Indeed, this value does not fit the data as BIC does.

We have not found a way to choose similarity thresholds based only on the simulated databases. Thus, to apply CD-hit-est and BLASTClust to the simulated data we have deliberately biased our inputs in favor of these programs : we have sought similarity thresholds that minimize the distance to the ideal clustering (as if this ideal clustering was known in advance) in the case of the first simulated database. Then we have applied these thresholds to the 25 simulated bases.

In the case of CD-hit-est, we have tested different similarities between 0.82 and 0.90. Indeed, we have calculated that, given the parameters of the simulations, the ideal threshold should be in this area. The results are shown in Table 1.2. On can see that the distance to the ideal clustering is minimized for a similarity threshold of 0.84. When studying these CD-hit-est clustering in detail, one can see that, even with this ideal similarity of 0.84, some sequences are isolated and do not fit into their ideal cluster. If the chosen similarity threshold increases, this number of isolated sequences increases. On the other hand, if the chosen similarity threshold decreases (below 0.84), some clusters merge together.

The case of BLASTClust is a little more complicated. In this case, we have varied four parameters detailed below :

- -S <threshold> similarity threshold
  - if <3 then the threshold is set as a BLAST score density (0.0 to 3.0 ; default = 1.75)
  - if ≥ 3 then the threshold is set as a percent of identical residues (3 to 100)
- -L <threshold> minimum length coverage (0.0 to 1.0 ; default = 0.9)
- -b <T—F> require coverage as specified by -L and -S on both (T) or only one (F) sequence of a pair (default = TRUE)

There are thus two binary parameters (-b on the one hand and the fact that $S < 3$ or $S \geq 3$ on the other hand) and two continuous parameters (S et L). The two binary parameters constitute 4 configurations (b = 'T 'and $S < 3$ ... b =' F' and $S \geq 3$). For each of these 4 cases, we have tested BlastClust on a 10,000 points grid (100 possibilities for S $\times$ 100 possibilities for L). We have observed the minimum distance to the ideal clustering obtained on these 40,000 points (it is 3). Then we have counted which of the four configurations contain the largest number of points that minimize the distance to the ideal clustering (it is b = 'F' and $S < 3$). After this, we have selected the point of this configuration closest to the average of the points that minimize the distance to the ideal clustering among the points that minimize the distance to the ideal clustering. The setting obtained is : S = 1.91 (BLAST score density), b = 'F', L = 0.49 and p = 'F' (p = 'F' means that we work on nucleotide sequences, not on proteins).

After we had obtained these parameters for CD-hit-est and BLASTClust, we finally applied these two tools to our 25 simulated datasets. The results are shown in Table 1.1 with those of our tool.

Similarity Intra-Cluster	0.82	0.83	0.84	0.85	0.86	0.87	0.88	0.89	0.90
Distance to Ideal Clustering	10	8	7	9	9	9	10	10	13

TABLE 1.2 – Search for the best similarity threshold for CD-hit-est

1.4/ DISCUSSION

1.4.1/ COMPARISON WITH OTHER TOOLS

Compared to CD-hit-est and BLASTClust, our tool has several advantages.

1. It uses a statistical criterion (in this case, the reputed BIC) to determine the number of clusters to consider. Thus, it can propose a number of clusters adapted to the database without the user having to provide any *a priori*.

2. As shown in Section 1.3.2, it allows a better reconstruction of the ideal clustering, even when the similarity threshold is chosen advantageously for CD-hit-est and BLASTClust.
3. It allows the user to plot useful graphical representations of the clustered data.

Finally, we note that CD-hit-est works only for intra-cluster similarities larger than 75%.

One drawback of our tool as compared to CD-hit-est and BLASTClust is the computation speed. According to `CD-hitFasterThanBLASTClust`, CD-hit-est is the fastest of these two programs. We have not been able to rigorously confirm this remark in our study, simply because we used BLASTClust on our computers while we used CD-hit-est online.

To sum up, the main features of our tool are different from that of CD-hit-is and BLASTClust. For users wanting to cluster their dataset into meaningful taxa, which makes it possible to apprehend the evolution, our tool might be of greater interest. On the other hand, if the objective is to reduce the size of the dataset by removing duplicates (or retain only one sequence per group of close sequences), it is more appropriate to use one of the other tools with a similarity of 100% (or close to 100%). In particular, CD-hit-est is written so as to provide a representative sequence for each cluster.

1.4.2/ POSSIBLE ALTERNATIVES WITH THE SAME CANEVA

Various options are possible to perform the analysis we have presented previously, some of them being listed below.

1.4.2.1/ SIMILARITY MATRIX

As stated previously, the multiple global alignment step is performed first before computing similarities, using MUSCLE software `edgar2004muscle`. Among the most extensively used methods, we can choose MAFFT `katoh2013mafft` too, as well as ClustalW or ClustalX `larkin2007clustal`. In addition, instead of defining the similarity matrix W as $W_{i,j} = 1 - M_{i,j}$, it could be possible to consider $W_{i,j} = \frac{1}{M_{i,j}}$ or $W_{i,j} = e^{-M_{i,j}}$.

1.4.2.2/ NUMBER OF CONSIDERED EIGENVECTORS

The number of eigenvectors to keep is another point to investigate. As explained in Section 1.2.1.2, it is usually advised to check graphically when the increase of eigenvalues is reducing. In this article, we have chosen to consider as default proposal : k_1 such that $\delta = \lambda_{k_1+1} - \lambda_{k_1} < 0.01$. This criterion has led to $k = 4$ in the case study, which seems acceptable according to the considered taxonomy.

Some authors in the literature proposed to compute k_1 as the logarithm of n `matiasnotes`. This method has been implemented as an option, as specified in Section 1.2.3. In addition, the "energy" method defined in Section 1.2.3 has been implemented too, in order to propose something similar to what is done in usual Principal Component Analysis (PCA). Other approaches can be considered to solve this problem, which is still an open one.

1.4.2.3/ NUMBER OF CLUSTERS

We have chosen to consider the BIC schwarz1978estimating to determine the optimal number of clusters, which is a common choice for this type of problem. An alternative may be to use the Akaike Information Criterion (AIC, akaike1974new). The principle of calculating the AIC is the same as the BIC, since the goal is to maximize log-likelihood penalized by the number of parameters (or more precisely, to minimize the number of parameters to which the log-likelihood is subtracted). AIC formula is the following :

$$AIC = -2ln(L) + 2 \times p,$$

where L is the likelihood of the estimated model and p the number of model parameters.

When $ln(n) \geq 2$, that is to say $n \geq 8$ (which is always the case in practice), BIC penalization is larger than that of AIC. Thus the number of clusters obtained by BIC is lower or equal to the one obtained by AIC. BIC is said to be "more conservative" than AIC. The choice between these two criteria can be dependent on how stringent the clustering is desired. The user of the `GClust` function may choose to use the AIC rather than the BIC as specified in Section 1.2.3. The user of the `GClust` function is also able to choose the number of clusters of his or her choice.

1.4.3/ CONCLUSION

In this work, we have proposed a new method of nucleotide sequence clustering. This clustering is produced by a methodology combining Laplacian Eigenmap with Gaussian Mixture models, while the number of clusters is automatically determined by using the Bayesian Information Criterion. The proposed methodology was applied to 100 sequences of mitochondrially encoded NADH dehydrogenase 3. The resulting clusters appeared to be coherent with the phylogeny (gene tree obtained with PhyML) as well as with the NCBI taxonomy. In addition, further tests have also been carried out on fully simulated data. These tests showed that our methodology allows to recover the expected clusters with greater accuracy.

One possible extension for future work could be to investigate more deeply the impact of parameters in the obtained clusters. The effects of using a different similarity matrix, or choosing a different dimension of the image space in Laplacian Eigenmap, or the number of desired clusters, could be investigated. Moreover, our tests on real data allowed us to watch our tool in action while performing a clustering of species into taxons. It might be interesting to also test the ability of our tool to classify 16S RNA sequences into OTUs on real data. Another avenue could consist in adapting the code to the very similar problem of protein clustering.

On a more computer-oriented aspect, our tools could be easier to access by being packaged with pypi. An online tool is also possible. Finally, the graphical interface could also be enhanced, for instance so as to make easier the identification of sequences associated to each point cloud.

2

ELÉMENTS TRANSPOSABLES

Cette partie présente nos travaux sur la propagation des rétrotransposons à l'intérieur du génome. Cette propagation est traitée sous la forme d'un processus de branchement impliquant différentes hypothèses, telles que le fait que la copie fille apparaisse avec une plus grande probabilité à proximité de la copie mère, que les mutations subies par les rétrotransposons affectent leurs capacités à se dupliquer etc.

Les données utilisées pour appliquer ce modèle sont des séquences génétiques de *Drosophila melanogaster* ou mouche du vinaigre (figure 2.1), une espèce particulièrement appréciée des biologistes pour son cycle de génération court (environs deux semaines), la facilité d'élevage et la facilité de la collecte de leur ADN.

FIGURE 2.1 – Drosophila melanogaster. Source : https://www.syngenta.fr

Les parties depuis "Abstract" jusqu'à la partie 2.4.5 correspondent à l'article publié en février 2017 dans le journal "Bioinformatics" sous le titre "Simulation-based estimation of branching models for LTR retrotransposons" qui présente nos travaux sur la propagation de ces rétrotransposons.

ABSTRACT

Motivation : LTR retrotransposons are mobile elements that are able, like retroviruses, to copy and move inside eukaryotic genomes. In the present work, we propose a branching model for studying the propagation of LTR retrotransposons in these genomes. This model allows us to take into account both the positions and the degradation level of LTR retrotransposons copies. In our model, the duplication rate is also allowed to vary with the degradation level.
Results : Various functions have been implemented in order to simulate their spread and visualization tools are proposed. Based on these simulation tools, we have developed a first method to evaluate the parameters of this propagation model. We applied this method to the study of the spread of the transposable elements ROO, GYPSY, and DM412 on a chromosome of *Drosophila melanogaster*.
Availability : Our proposal has been implemented using Python software. Source code is freely available on the web at https ://github.com/SergeMOULIN/retrotransposons-spread.
Supplementary information : Supplementary data are available at *Bioinformatics* online.

2.1/ INTRODUCTION

A transposable element (TE) is a DNA sequence able to move from one location to another inside a genome. These sequences, discovered during the 50's by Barbara McClintock mcclintock1950or2 exist in almost all living organisms and are the source of a huge number of mutations. They are considered as a major cause of genetic disease in human belancio2008mammalian or in *Drosophila* where they are responsible for more than 80% of the spontaneous mutations green1988mobile. DNA sequences derived from these TEs can represent a large part of a genome. For example, they represent about 45% of the human genome lander2001initial and over 70% of the corn genome sanmiguel1998evidence. Fortunately, most of these sequences correspond to fragments or "dead" elements that have lost their ability to move in the genome due to several lethal mutations or are controlled, especially via epigenetic mechanisms.

TEs have two possible ways to move in a genome, according to their type finnegan1989eukaryotic wicker2007unified. The first class of mobile elements are cut from their original place to move to another one, and are called "DNA transposons" or "Class II transposable elements". The other class of mobile elements, called "retrotransposons" or "Class I transposable elements", use an RNA intermediate to duplicate themselves, the new copy being inserted into another location of the genome. Two orders are identified among the retrotransposons according to the presence or absence of Long Terminal Repeat (LTR) sequences at their extremities. The LTR retrotransposons are similar in structure to retroviruses such as HIV. In both classes, TEs can be classified as either "autonomous", if they encode the enzymes that will allow them to move, or "non autonomous" if they use the enzymes produced by other elements. In an assembled genome, the various sequences corresponding to TE insertions can be found using different bioinformatic approaches (see lerat2011comparative for a review), which allow us to determine the exact number and positions of each TE insertion. In this article, we focused on the important problem of inferring the history of the spreading of LTR re-

trotransposons. For this purpose, we modeled the evolution using a branching process where each element (*i.e.*, a copy of a given TE) can randomly evolve via duplication or mutation.

Instances of branching processes have already been proposed in the literature, as putative models for the propagation of TEs. However, most of these studies focus on the evolution of the host population, and not on the propagation of the TEs in the host. The "subject" of these branching models (*i.e.*, the entity able to change or duplicate) is generally the host, while in our case it is the TE itself. For instance, Michael E. Moody [moody1988branching] has used a branching model, where the studied variable was the number of individuals owning i copies of a given TE. Sawyer *et al.* [sawyer1987distribution] produced almost the same model, in order to study the distribution and abundance of insertion sequences.

Kaplan *et al.* proposed a model where TEs can be either of wild type (*i.e.*, non mutated) or of mutant one, which is a little closer to our proposal. When a host gives birth to its child, wild copies can mutate or be deleted, whereas mutant ones can only be removed. New copies can be additionally created. This number of new created copies is supposed to decrease with the proportion of mutants. More recently, interesting models have been proposed that take into account the location of TEs. For instance, Drakos and Wahl [drakos2015extinction] suggested a model of mobile promoter evolution, where the probabilities for promoters to duplicate inside or outside their region is potentially not the same.

In the present work, the objective is to propose a new approach for the propagation of LTR retrotransposons that combines a location-dependent model with the fact that LTR retrotransposons can face degradation (*i.e.*, mutations, recombination, etc.), which may decrease their duplication rate, that is, their potentiality to copy and insert elsewhere in the genome. Then, we have developed a first method to evaluate the model parameters : average distance traveled by the TEs before insertion, location of the original copy, average time between two degradations (mutations, recombination, etc.), average impact of a degradation, and the impact of degradations undergone by copies on their duplication speed. This method requires to define a distance between the results of the simulations and the observed chromosome, which is based on the Hungarian method [kuhn1955hungarian, munkres1957algorithms]. This method has been applied to the spreading of the LTR retrotransposons ROO, DM412, and GYPSY on the chromosome 3L of *Drosophila melanogaster*. The parameters associated to each TE are computed and a branching tree is proposed in each case. Our results show that, according to our model and method, the roots of ROO, DM412, and GYSPSY on the chromosome 3L could correspond to the annotated copies FBti0059644, FBti0061034, and FBti0062705 respectively.

2.2/ SYSTEM AND METHODS

2.2.1/ THE BRANCHING MODEL

2.2.1.1/ THE BRANCHING TREE

An example of branching tree is shown in Figure 2.2.

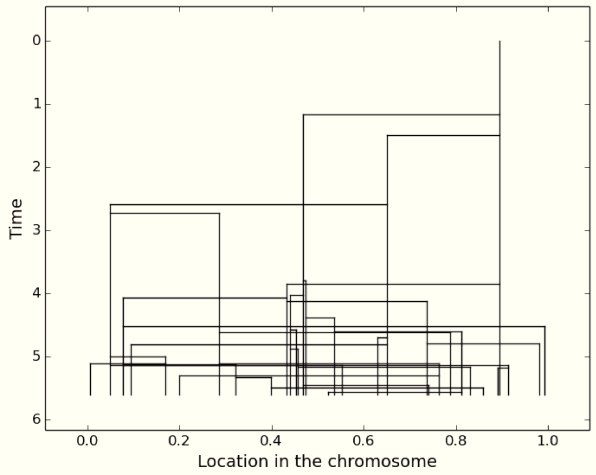

FIGURE 2.2 – ROO spread

This branching tree represents the spread of the LTR retrotransposon “ROO” between times 0 and 5.601. At time 0 there is only one copy, called the “root” in this article. At time 1.488, the root duplicates itself to give birth to its first “child”.

Finally, the process ends at time T_{obs} = 5.601 with 32 copies. The branching tree represents only the duplications, but copies are also subject to degradations as explained in Section 2.2.1.2. The state of the tree (*i.e.*, number of TE copies, copy positions, degradations undergone by the copies) at time T_{obs} = 5.601 is named “final state” of this tree. The working principle of our estimation method is to simulate trees, in order to determine in which conditions the final states of simulated trees match well with the observed chromosome. To compare the final state of a simulated tree with the observed chromosome, a distance was build, as detailed in Section 2.2.2.2.

2.2.1.2/ THE GENERAL MODEL

The branching model is constructed as follows.

1. The spread starts with only one copy, called the root, at time zero in a location X_0 to be determined.
2. Each copy can either be duplicated or undergo degradations (*i.e.*, mutations, recombinations, etc.) at any time.
3. The number of copies increases due to duplications. When a new copy is created, it receives an index equal to the number of existing copies at the time of its birth, including itself. In the remainder of this article, let T_i be the birth date of the i^{th} copy and let $\tau_{i,k}$ be the time when the i^{th} copy faces its k^{th} degradation.
4. The time interval $\tau_{i,k+1} - \tau_{i,k}$ between two degradations is supposed to be independent and identically distributed (i.i.d.) with an exponential distribution $\mathcal{E}(\frac{1}{\mu})$, where μ (*i.e.*, average time between two degradations) must be determined.
5. Each copy is also associated to its Needleman-Wunsch `needleman1970general` similarity to the original state of the root. This similarity is a value between 0 and 1. Let us denote $S_i(t)$ the similarity between the original state of the root and the

state of the i^{th} copy at time t. This similarity decreases as a function of time, due to degradation effects. In addition, we defined the *state of deterioration* by $D_i(t) = 1 - S_i(t)$.

6. At each degradation, the similarity to the root is divided by $1 + E(\frac{1}{\beta})$, where β has to be determined. In other terms, $S_i(\tau_{i,k+1}) = \frac{S_i(\tau_{i,k})}{1+E(\frac{1}{\beta})}$.

7. At time t, for the i^{th} copy, conditionally on $D_i(t)$, we assume that the time before the next duplication follows a distribution $\mathcal{E}(\frac{1}{1+p\times D_i(t)})$, where $p > 0$ is a parameter to be determined. In other words, the time before the next duplication is longer when the copy is far from the original state of the root (in terms of Needleman-Wunsch distance). Note that, in this article, "duplication rate" is just the inverse of the "average time before the next duplication". In other words, duplication rate = $1 + p \times D_i(t)$.

8. Moreover, each copy is also associated to its position in the chromosome. This position is denoted by X_i for the i^{th} copy. This position is constant with time. We assume that each child j of a copy i satisfies $X_j = X_i + \chi_{i,j}$, where $\chi_{i,j}$ follows a distribution $\mathcal{U}\{-1,1\} \times \mathcal{E}(\frac{1}{L})$, in which $\mathcal{U}$ represents the uniform law (*i.e.*, the probability to choose -1 or 1 is the same) and L is a parameter to be determined.

9. We also take into account the host structure (*i.e.*, position of host genes) to insert or not the child in the chromosome. Concretely, we calculate

$$\text{density} = \frac{\frac{\text{Number of TEs in genes}}{\text{Surface occupied by genes}}}{\frac{\text{Number of TEs out of genes}}{\text{Unoccupied surface}}}$$

on the real chromosome. During simulations, when the child moves into a gene, it can be inserted with a probability equal to "density", otherwise the child position is recomputed (if "density" ¿ 1, the child can always insert itself). Furthermore, the child position is also relaunched when it goes outside the chromosome (*cf.* Part 2.3.1.3).

Our goal is thus to estimate the parameters of this model, *i.e.*, X_0, μ, β, p, L and T_{obs}. Note that the duplication speed of the non-degraded root is set to 1 and it does not need to be determined. Indeed, this duplication speed is redundant with μ and p. In addition, note that the parameter L is not really informative about the mean distance traveled by the child before its insertion, due to putative relaunching processes. This is why we also provide the mean traveled distance in simulations, that is, the mean jump, which is denoted by J. See Part 2.3.1 for an example of its computation.

2.2.2/ THE ESTIMATION METHOD

As explained in Section 2.2.1.1, the working principle of our estimation method is to simulate trees in order to determine in which conditions the final states of simulated trees match well with the observed chromosome.

Trees are simulated according to the model defined in Section 2.2.1.2. The stopping criterion of these simulated trees depends on the number of copies in the observed chromosome. Actually, the simulation was constrained to stop at the birth date of the $n + 1^{th}$ copy, where n is the number of copies in the observed chromosome, see Section 2.3.1.2 for further details.

In order to improve computation speed, parameters estimation has been split in three parts, i) we estimate μ, β, and L (*cf.* Section 2.2.2.1) ; ii) we estimate X_0 and L (*cf.* Section 2.2.2.2), which requires to define a distance between simulated trees and the observed chromosome ; iii) we estimate J and T_{obs} (*cf.* Section 2.2.2.3).

2.2.2.1/ ESTIMATION OF μ, β, AND p

The objective here is to estimate the parameters μ, β, and p. We can note that these parameters only affect the distribution of deterioration states. They have no direct influence on copy positions. Thus, the goal at this step is to minimize the differences between the distribution of states of deterioration in the simulated trees, and the states of deterioration distribution in the observed chromosome. More precisely, we want to minimize $D1(Tr, C) = \sum_{i=1}^{n}(D_{i,Tr} - D_{i,C})^2$ where $D_{1,Tr}...D_{n,Tr}$ is the sorted distribution of the states of deterioration for the simulated tree, $D_{1,C}...D_{n,C}$ is the sorted distribution of the states of deterioration for the observed chromosome, and n is the number of TEs in the observed chromosome.

For this purpose, a 3-dimensional grid has been constructed, where each point of this grid represents a triplet (μ, β, p), while X_0 and L are set to predefined values (they do not matter at this stage). A score $S_1 = \sum_{i=1}^{N1} D1(Tr_i, C)$ has been associated to each of these triplets. In this formula, Tr_i is the i^{th} tree simulated with the parameter set, C represents the observed chromosome, and $N1$ is a parameter chosen by the user of the optimization method (for instance, $N1$ = 10,000 in the case study of Section 2.4). The best of these points is selected, and a smaller grid is constructed around it. This iterative process is continued until the precision chosen by the user of the optimization method has been obtained.

2.2.2.2/ DISTANCE BETWEEN TREES, ESTIMATION OF X_0 AND L

The first step, in the estimation of X_0 and L, is to define a distance between the final state of a simulated tree and the observed chromosome. For this purpose, we first need to design a distance between a copy of the simulated tree and a copy of the observed chromosome. Let us name R_i the i^{th} copy of the simulated tree, X_i its position, and $D_i(T_{obs})$ its state of deterioration at the end of the process. Similarly, $\widetilde{R}_j$ is the j^{th} copy of the observed chromosome, $\widetilde{X}_j$ its position, and $\widetilde{D}_j(T_{obs})$ its state of deterioration. The distance between two copies has been designed as follows :

$$D2(R_i, \widetilde{R}_j) = \frac{(X_i - \widetilde{X}_j)^2}{w_1} + \frac{(D_i(T_{obs}) - \widetilde{D}_j(T_{obs}))^2}{w_2}$$

where $w_1 = \sum_{i=1...n, j=1...n} (\widetilde{X}_i - \widetilde{X}_j)^2$ and

$w_2 = \sum_{i=1...n, j=1...n} (\widetilde{D}_i(T_{obs}) - \widetilde{D}_j(T_{obs}))^2$. This weighting by w_1 and w_2 allows us to give the same weight to positions and states of deterioration.

From these distances between two copies, we can now create a matrix of distances W, verifying $W_{ij} = D2(R_i, \widetilde{R}_j)$. Then, the distance between final states of two trees has been defined as the best possible adjustment between copies, using the so-called Kuhn-Munkres algorithm, also named the Hungarian method `kuhn1955hungarian, munkres1957algorithms`. The Hungarian method is an algorithm that allows us to minimize the sum of n elements of a $n \times n$ matrix, under the condition that there is only one element by row and only one element by column. In our case, the Hungarian method allows us to assign exactly one copy of the simulated tree to each copy of the real chromosome while minimizing the sum of distances between paired copies. Let us name $D3$ this distance created by this way.

Once this distance between trees has been defined, we use it to estimate X_0 and L with the same type of process as in the previous step. In other words, a 2-dimensional grid has been constructed, where each point of this grid represents a couple of parameters (X_0, L) while μ, β, and p are fixed to the values found in the previous stage. The score $S_2 = \sum_{i=1}^{N1} D3(Tr_i, C)$ is then computed for each of these points, and a smaller grid is recursively built around the best point.

2.2.2.3/ ESTIMATION OF J AND T_{obs}

Conversely to these μ, β, p, X_0, and L, which are inputted in our simulation algorithm, J (mean jump) and T_{obs} are outputs. It is thus easier to estimate them. In this step, we run $N2$ simulations where μ, β, p, X_0, and L are set to the values found in the two previous steps. The estimations of J and T_{obs} are then the mean results of the output J and T_{obs} of these $N2$ simulations (for instance $N2$ = 20,000 in the case study of Section 2.4).

2.3/ ALGORITHM

Our proposal has been implemented using Python[1]. A short application programming interface is detailed thereafter.

2.3.1/ TREEBUILD

This main function is used to build branching trees following the model defined in Section 2.2.1.2. Its halt condition is the targeted number of copies. Its prototype meets the following canvas :

$$(S, T, T_{obs}) = TreeBuild(X, \mu, \beta, p, L, n, genes, density),$$

where n is the desired number of copies, while X_0, μ, β, p, and L are the model parameters as defined in Section 2.2.1.2. Moreover $genes$ is the positions of each gene in the observed chromosome, and $density$ is the value defined in Part 2.2.1.2. Concerning the outputs, S is a $n \times 1$ vector representing states of deterioration while T_{obs} is the propagation time. Finally T is a $n \times 3$ matrix containing, for each copy : its position, its birth date, and the row of its mother, like in Table 2.1.

1. Available at **https ://github.com/SergeMOULIN/retrotransposons-spread**

TABLE 2.1 – Example of the output T

[[	0.5	0.	0.]
[	0.19031606	1.83699228	0.]
[	0.18321005	11.25706728	0.]
[	0.66442132	17.61532334	2.]
[	0.48479738	25.45993783	1.]
[	0.13876928	28.11662473	1.]]

In this example, the mother of the copy located in 0.1832 is the root. The mother of the copy located in 0.6644 is the copy located in 0.1832. Other details regarding this main function are provided thereafter. The mean jump J is not a direct output of TreeBuild, but it can be easily computed with T. In this example, $J = \frac{|0.19-0.5|+|0.18-0.5|+|0.66-0.18|+|0.48-0.19|+|0.14-0.19|}{5}$.

2.3.1.1/ MULTIPLE CLOCKS MANAGEMENT

The working principle of TreeBuild can be summarized as follows : it determines the next event (deterioration or duplication) and executes it until the stopping criterion is satisfied. To determine the next event means to know its nature (deterioration or duplication), its time, and in which of the available copies it happens. Let j be the number of available copies at time t_1. The easiest way to determine the next event is to simulate $2 \times j$ exponential laws, one for each possible deterioration or duplication. The minimum of these $2 \times j$ simulations can thus provide the time, the nature, and the copy related to the next event.

Actually, TreeBuild does not really simulate $2 \times j$ exponential laws, as two properties of this law have been used to shorten computations. Indeed, $\forall (p_1, ..., p_{2j}) \in \mathbb{R}^{2j}, \forall (Y_1, ..., Y_{2j}) \sim \left(\mathcal{E}(p_1), ..., \mathcal{E}(p_{2j})\right)$, we have :

1. $min\left(Y_1, ..., Y_{2j}\right) \sim \mathcal{E}\left(\sum_{i=1}^{2j} p_i\right)$,
2. $\forall i \in 1...2j, P\left(Y_i = min\left(Y_1, ..., Y_{2j}\right)\right) = \frac{p_i}{\sum_{k=1}^{2j} p_k}$.

Hence, due to the first property, the time of the next event can be simulated by a single exponential law. The second property, for its part, allows us to determine the nature and the copy affected by the next event using a single uniform law.

2.3.1.2/ STOPPING CRITERION

As stated before, the stopping criterion of TreeBuild is related to n (the number of copies of the observed chromosome). But when a chromosome is observed, there is no way to detect that a new duplication has just occurred. Thus, the program cannot stop exactly at the birth of the n^{th} copy. Actually, TreeBuild must run until the T_{n+1} iteration (the birth date of the $n + 1^{th}$ copy), and then the propagation time T_{obs} can be determined by :

$T_{obs} = \frac{T_n + T_{n+1}}{2}$

Furthermore, each value taken by S and T between T_n and T_{n+1} is kept in memory. Thereby, the values of T and S returned by TreeBuild are values of T and S at time T_{obs}.

2.3.1.3/ THE MANAGEMENT OF COPY LOCATIONS

Copy positions in the chromosome are in the interval [0,1]. The distance traveled by a TE before insertion is assumed to follow an exponential law, but this latter can send the new copy outside the interval [0,1]. The solution chosen in this case is to launch again the computation of the new copy position.

In addition, the copy position is also relaunched with a probability "density" if its position falls into a host gene, as explained in Part 2.2.1.2. In other words :

$$\text{while } (X_{child} \notin [0,1] \text{ or } (X_{child} \in \textit{gene and } U1 < \text{density})) :$$
$$X_{child} = X_{mother} + U2 \times Y$$

where $U \sim U\{-1,1\}$, $U2 \sim U\{-1,1\}$, and $Y \sim \mathcal{E}(\frac{1}{L})$.

2.3.1.4/ CRITICAL SITUATIONS

When TreeBuild is launched for each point of the grid of parameters, some critical situations can happen, which may induce a significant slowdown of the program. In particular, when μ is small and β is large, the probability for an event to be a duplication rather than a deterioration becomes very low. Thus, TreeBuild executes an inordinate number of deteriorations before reaching the desired number of copies. To solve this issue, we have decided that when the similarity to the root becomes lower than 0.03, then this copy cannot be degraded anymore.

2.3.2/ ESTIMATION METHOD

In the available package, the estimation of the branching model parameters is realized by the *Optim* function. Its prototype is as follows :

$$(Best, Score) = Optim(Grid, Case, n1, N1, N2, genes).$$

Here, $Grid$ is a 5×4 matrix of settings defined exactly as in Section 2.2. $Case$, for its part, is a $2 \times n$ matrix containing locations and state of deterioration for each copy of the observed chromosome. $N1$ and $N2$ are settings defined in Sections 2.2.2.1 and 2.2.2.3, while $n1$ indicates how the grid is shrunk at each step after obtaining the best point (*cf.* the following section). Finally, $genes$ are the positions of each gene in the observed chromosome. The output $Best$ is the parameter set $(X_0, \mu, \beta, p, L, MJ, T_{obs})$ returned by the *Optim* function, while $Score$ is the sum of $N2$ differences between simulations and the observed chromosome (this $Score$ is useful if we relaunch Optim several times).

2.3.2.1/ INTERVAL REDUCTION

As explained in Sections 2.2.2.1 and 2.2.2.2, the estimation method works with a grid where each point represents a parameter set. When the best point of the grid is found, a

new grid is constructed around this point. Note that the new grid is not necessarily included in the previous one, in order to provide a larger degree of freedom of the parameters (in particular, when the latter are close to zero). For instance, in the case of parameter L, the minimum of the new interval is $min\left(\frac{L_{min}}{2}, L_{best} - \frac{L_{delta}}{2\times n1}\right)$, where L_{min} and L_{max} are the minimum and maximum of the previous interval, $L_{delta} = L_{max} - L_{min}$, L_{best} is the L coordinate of the best parameter set, and $n1$ is the reduction parameter selected by the user. Thus, the minimum value of the test interval is divided by two at each time the best point of the grid is close enough to zero. The maximum value of the new interval is simply $L_{best} - \frac{L_{delta}}{2\times n1}$. These formulas, written for L, are also valid for β, μ, and p.

2.3.2.2/ LOCATION IN THE CHROMOSOME

Unlike the other parameters for which we scan a continuous interval, in the case of X_0, we only consider the positions of the TE copies on the observed chromosome. Thus we scan the interval of integers $1, .., n$. However, we process in the same way as with the other parameters (excepted that we use rounded values), and we never get out of the original interval $1, .., n$ in this case.

For instance, if we choose to test this interval of integers in four points (as we do in our case studies) : at the first step, X_0 is tested in the first, $round(1 + \frac{n-1}{3})$-th, $round(1 + \frac{2\times(n-1)}{3})$-th, and n-th position of TE copies in the observed chromosome. In the next step, the new test interval thus becomes $max(1, round(X_{best} - \frac{X_{delta}}{2\times n1}) ... min(n, round(X_{best} - \frac{X_{delta}}{2\times n1}))$ where X_{delta} is $n - 1$ here.

2.3.3/ MODULE AND PACKAGE DEPENDENCIES

The Hungarian method has been applied using the “munkres” module, implemented in 2008 by Brian M. Clapper (Brian M. Clapper, munkres 1.0.7 for Python, https ://-pypi.python.org/pypi/munkres/).

2.4/ RESULTS AND DISCUSSION

2.4.1/ THE DATA

This proposal has been applied to the spread of the LTR retrotransposons ROO, DM412, and GYPSY on the euchromatin part of the chromosome 3L of the *Drosophila melanogaster* genome. This sequence corresponds to the left arm of the chromosome 3, which is the largest autosomal chromosome of *D. melanogaster*.

This is also the most prolific chromosome for each of the LTR retrotransposons we considered, this is why it has been chosen for this case study. ROO corresponds to the LTR retrotransposon in *Drosophila melanogaster* with the largest number of copies `kaminker2002transposable, lerat2003sequence, de2009evolutionary`. DM412 is supposed to have been recently acquired by *D. melanogaster* through horizontal transfer from a close relative species `bartolome2009widespread, lerat2011comparative, modolo2014new`. Fi-

nally, GYPSY is an older and likely well regulated LTR retrotransposon lerat2011comparative.

Chromosome 3L contains 32 copies of ROO (with a mean nucleotide identity of 68.82%), 16 copies of DM412 (mean nucleotide identity of 60.24%), and six copies of GYPSY (mean nucleotide identity of 13.6%).

Three databases have been used during the experiments. The first one contains positions and nucleotide sequences for each TE copy annotated in *D. melanogaster* (flybase website[2] version number 5.51 of the *D. melanogaster* genome adams2000genome, smith2007release). The second database has been downloaded from the RepBase website[3] and contains the consensus sequences for each TE corresponding to reference elements. The Needleman-Wunsch distance between each TE copy (from the first database) and its reference (from the second database) has been calculated, in order to obtain the deterioration states.

Finally, the third database comes from flybase too. This is the position of all the annotated genes in the euchromatin part of chromosome 3L, in version number 5.51 of the *D. melanogaster* genome.

In this case study, the estimation method described in Section 2.2.2 has actually been applied not only once but 40 times in each situation, in order to check the consistency of the obtained parameter sets. The best parameter set of each case study, considering the output "score" (*cf.* Part 2.3.2), is presented in Section 2.4.3. The whole obtained parameter sets are presented in supplementary data with their descriptive statistics. Some indications about consistency of these results are provided in Section 2.4.4.

2.4.2/ SETTINGS

Let us first recall that X_0, which represents the root position in the chromosome, is inside the interval [0 , 1]. In other words, copy positions have been divided by the chromosome size. For the euchromatic part of chromosome 3L, this size has been set at 24,543,557 base pairs (bp) in the version 5.51. smith2007release.

In Table 2.2, each row represents the beginning and the end of the test interval, the number by which the test interval has been divided, and the final desired accuracy regarding the parameter. In particular, in the third column, the value associated to each is 3. This latter means that these parameters have been tested at the beginning, in the first third, in the second one, and at the end of the test interval.

Finally, at each iteration, the grid used for μ, β, and p estimations contains $4^3 = 64$ points while, at each iteration, the grid used to estimate X_0 and L contains $4^2 = 16$ points.

The other parameters are :

- $n1$ = 1.5 : at each step, after the best point has been found, the grid's dimensions have been divided by 1.5.
- $N1$ = 10,000 : each point has been tested 10,000 times during estimation of X_0, μ, β, p, and L.
- $N2$ = 20,000 : J and T_{obs} estimations are the average values of 20,000 simulations. The output "score" is computed based of these simulations.

2. **http ://flybase.org/**
3. **http ://www.girinst.org/repbase/**

TABLE 2.2 – Setting table

parameter	starting point	end point	interval division	desired accuracy
X_0	1	n	3	10^{-3}
μ	0.1	10	3	10^{-2}
β	0.01	1	3	10^{-3}
p	0.1	100	3	10^{-1}
L	0.01	1	3	10^{-3}

TABLE 2.3 – Results and consistency

Best parameter sets			
parameter	**ROO**	**DM412**	**GYPSY**
X_0	29	14	5
μ	2.396	1.135	0.050
β	0.351	0.235	0.093
p	0.051	0.001	0.007
L	1.331	0.336	0.016
J	0.300	0.216	0.013
T_{obs}	4.070	3.353	2.379
Consistency indicators			
parameter	**ROO**	**DM412**	**GYPSY**
X_0	0.104	0.096	0.392
μ	0.020	0.017	0.002
β	0.040	0.064	0.044
p	0.001	0.000	0.000
L	0.262	0.036	0.033
J	0.010	0.013	0.024
T_{obs}	0.042	0.014	0.031

Furthermore, we can notice that in the case of ROO, 18 of the 32 copies are located inside genes while genes hold 72.7% of the studied part (euchromatin). Thus, density = $\frac{\frac{18}{72,7}}{\frac{14}{27,2}} = 0.482$. In the cases of DM412 and GYPSY, the densities are respectively equal to 0.625 ($\frac{10}{16}$) and 0.187 ($\frac{2}{6}$).

2.4.3/ RESULTS

The obtained parameters are summarized in Table 2.3.

If we consider for instance the spread of ROO, the obtained parameters can be interpreted as follows :

— $X_0 = 29$. The root is on the 29^{th} position in chromosome 3L. This is the copy FBti0059644, which is located between the $21,954,331^{th}$ and the $21,954,698^{th}$ nucleotide.

— $\mu = 2.396$. The average time between two degradations is 2.396, where 1 is the average time before duplication of the root. Degradations are thus less frequent than duplications. Please note that this estimation of μ is without time unit : it is related to the duplication speed of the root. It allows us to estimate duplication

speed when the deterioration speed is known, and *vice versa*.

— $\beta = 0.351$. Each degradation causes a division by $1 + E(\frac{1}{0.351})$ of the similarity. Thus the similarity is divided by 1.351 on average at each degradation.

— $p = 0.051$. p allows us to determine how many degradations led to a decrease in the duplication speed. For example, in this case, if the identity between a copy and the reference is 0.75 (*i.e.*, state of deterioration = 0.25), then the duplication speed of this copy is reduced by 1.275% (indeed $0.25 \times 0.051 = 0.01275$). This looks like a low effect.

— L = 1.331. The distance traveled by the TE before insertion follows a distribution $E(\frac{1}{1.331})$, with the relaunching process : (1) when the position of the child is out of the chromosome (*cf.* Section 2.3.1.3) and (2) possibly when the position of the child is inside a gene. As explained in Part 2.2.1.2, the parameter J is better to represent the average distance before insertion. However, we can notice that L = 1.331 is a pretty large number (larger than 1). This value tends to suggest that the child inserts itself more or less anywhere in the case of ROO (compared to DM412 or GYPSY).

— J = 0.300. The mean distance traveled by the TE before insertion is 0.300.

— T_{obs} = 4.070. The time of ROO propagation is 4.070 larger than the non-deteriorated root. In addition, 4.070/2.396 = 1.699, thus the root has faced 1.699 degradations on average. From a global perspective, we can note that in each case, the root corresponds to the border elements. They are in position 29^{th} over 32, 14^{th} over 16, and 5^{th} position over 6 respectively for ROO, DM412, and GYPSY. The latter correspond to positions 0.894, 0.965, and 0.969 when the chromosome is considered as a [0,1] interval. This can be due to a larger density of TEs in this area. In addition, we can notice that p is really close to 0 in each case. This should imply no (linear) effect of the degradation on the duplication rate. This is an unexpected result. Indeed, the fact that the degradation undergone by the copies reduces their ability to duplicate sounds natural. This is why the parameter p was added in the model. Finally, we can notice that in the case of ROO, the results suggest a few big degradations, while in the case of GYPSY, they suggest a lot of little ones.

The fact that some of the obtained parameters are outside the test interval chosen at the beginning of the program (for instance, L = 1.331 in the case of ROO or μ = 0.050 in the case of GYSPY) is a desired effect, to let a larger freedom to the parameters. In particular, the aim was to let parameters to be as close as possible to zero if required (*cf.* Section 2.3).

In each of these three cases, one billion trees have been simulated with the obtained parameter set. The best of these trees is shown in Figure 2.2 for ROO and in supplementary data for DM412 and GYPSY.

2.4.3.1/ FOCUSING ON THE ROOTS

According to these models and methods, the root of ROO could correspond to the FBti0059644 copy. This is an incomplete copy (368 bp, compared to the reference which has 9,112 bp) that corresponds to a solo-LTR, a remnant from a LTR-LTR recombination. This copy is thus no longer active, but it is quite recent since its divergence to the reference is rather low (95.71% of identity on the aligned part of the sequences).

The root of DM412 could correspond to the FBti0061034 copy. This is a very degraded copy that has 88 bp in length, corresponding to an internal portion of the reference element, whose length is 7440 bp. The copy is old since it is very divergent compared to the reference (80.90% of identity).

Finally, the root of GYPSY could correspond to the FBti0062705 copy. This copy is an incomplete and very degraded element of 1,282 bp length (7,471 bp for the reference) with a very high divergence to the reference (70.59% identity). This copy corresponds to a piece of the inner part of the gypsy element, and this is a very old and degraded copy that is not currently active.

2.4.4/ CONSISTENCY OF RESULTS

As explained previously, the optimization method has been actually applied 40 times for each TE. The descriptive statistics for these three cases are summarized in the supplementary data. In addition, for each parameter that has been estimated by scanning an interval (*i.e.*, X_0, μ, β, p, and L), quotients $\frac{\text{Standard deviation of the results}}{\text{Test interval}}$ have been computed in each case, in order to assess the consistency of the results. These quotients are reproduced in Table 2.3. Standard deviations of the output J and T_{obs} are also set in the same table.

Several versions of the code have been implemented in this project in order to increase the consistency of these results. In a previous version, X_0, μ, β, p, and L were estimated all together (*i.e.*, steps 2.2.2.1 and 2.2.2.2 were merged). This approach implied to work in a $4^5 = 1024$ points grid, thus, it provided a lower number of trials by point, which reduced consistency. In our very first version, the duplication rate was a parameter to be estimated while $T_{obs} = 1$ was the stopping criterion of TreeBuild. The difference between n and the number of copies at the end of the simulation (which was not necessarily n in this setup) was penalized. However, in this setup, the duplication rate was the only parameter to be consistent.

Finally, in the setup presented here, the consistency looks acceptable in most of the cases (*i.e.*, excepted for L in the case of ROO or for μ in the case of GYPSY). Nevertheless, this requires a large number of simulations implying that the program runs several days.

2.4.5/ CONCLUSION AND FUTURE PERSPECTIVES

In this article, a model has been proposed for the propagation of LTR retrotransposons in a genome. Various functions have been implemented to simulate this spread as well as graphic representations. Finally, a first method for estimating the parameters of this propagation model has been proposed and applied to the spread of TEs corresponding to the ROO, GYPSY, and DM412 elements in a chromosome of *Drosophila melanogaster*. However, this work can be improved in various directions, some of them being listed below.

The first point is that the model of propagation should be applied to the full genome instead of a single chromosome of *Drosophila melanogaster*. Indeed, a copy inserted in a given chromosome can produce a child that will not necessarily inserts itself in the same chromosome. An idea to extend the model to the full genome could be to let the position of the child copy following a $\delta(a)\mathcal{U} + \delta(1-a)\mathcal{E}(\frac{1}{L})$ law. In other words, the child

copy inserts itself anywhere in the genome (uniform distribution), with a probability a, or it inserts itself on the same chromosome than the mother copy (with the distance to the mother following an exponential law) with probability $1 - a$, where a is a new parameter to determine. Nevertheless, this approach raises various questions that must be answered in a further study. Firstly, should $\mathcal{E}(\frac{1}{L})$ represent a number of nucleotides or a proportion of the considered chromosome ? How to redefine the distance built in Part 2.2.2.2 ? etc.

Secondly, as explained in Part 2.4.4, the estimation method needs a large compilation time to give acceptably consistent results. This compilation time increases quickly when the sample size (number of TEs) increases or if we want to estimate more parameters. Therefore, we will test other ways (likelihood maximization, neighbor joining, or Bayesian estimation) to improve our proposal. These types of methods would also allow us to produce some confidence intervals for the estimated parameters. Our long-term objective is to create a useful tool for estimating consistently both parameters and the branching process itself. In other word, our goal is to produce a tool close to phylogenetic tree estimation but adapted to TE constraints. Another possibility of improvement could be to consider the possibility of several roots. For instance, a method of unsupervised classification like Gaussian Mixture model could be applied in order to detect the number of clusters.

In this project, we used RepBase consensus sequences, based on a lot of TE sequences as root sequence. Another possibility could also be to produce an ancestral reconstruction based only on the sequences of the case study. In this way, it would not be necessary to search the data in two different databases. We can also note that in this work, we consider all modifications (*i.e.*, mutations, LTR recombination, and so on) as one single and global deterioration effect. It could be interesting to try to distinguish each effect. Finally the effect of an epigenetic regulation that can affect TE behaviour even if they do not face sequence degradation could be taken into account.

FUNDING

Computations have been performed on the supercomputer facilities of the Mésocentre de calcul de Franche-Comté. This work has been supported by the Transposable Elements project of the Franche-Comté region. The work of S.C. was also sponsored by NMS/IRD project 117480 at NPL.

3

ROC

Ce chapitre décrit les travaux que nous avons effectués sur des données collectées sur le site de Tavaux par l'équipe de chrono-environnement. Ces données sont constituées de caractéristiques physico-chimiques, d'OTUs bactériennes et d'OTUs fongiques collectées sur 16 arbres. Ces 16 arbres sont en l'occurrence 8 saules et 8 peupliers. La moitié de ces arbres (4 saules et 4 peupliers) est située sur une lagune contaminée par divers polluants issus de l'usine chimique Solvay de Tavaux. L'autre moitié (4 saules et 4 peupliers) est située sur une zone naturelle non contaminée. En particulier, cette lagune est affectée par une pollution au mercure qui s'explique par le procédé d'électrolyse à mercure utilisé jusqu'en 2010 pour la fabrication de composés chlorés. L'objectif de cette collaboration avec l'équipe "chrono-environnement" était de proposer des analyses statistiques qui puissent compléter celles qu'ils avaient effectuées l'année précédente sur les mêmes données `zappelini2015diversity` afin d'approfondir la compréhension de l'effet de ces pollutions sur les populations bactériennes et fongiques. Au cours de ces travaux, nous avons décidé de nous concentrer plus précisément sur les résultats apportés par les courbes ROC. Les parties suivantes (depuis "Abstract" jusqu'à la partie 3.4) sont une version d'article telle qu'il a été proposé au journal "Microbial Ecology" en 2017 sous le titre "A ROC-based analytical tool for environmental metabarcoding dataset analysis". Cette version a malheureusement été refusée et l'article a donc été retouché depuis. Néanmoins cette version est plus représentative de mon travail que la retouche c'est pourquoi c'est celle que je présente ici. La partie 3.5 présente d'autre analyses effectuées sur ces données. Plus précisément cette partie présente une visualisation des données via une combinaison ACP-GMM.

Abstract

High-throughput metabarcoding tools based on next generation sequencing are able to produce high volumes of data at an affordable cost. We have developed a receiver operating characteristic (ROC) analytical tool under Python and R that detects and emphasizes the discriminating factors in environmental datasets obtained from a wide metabarcoding analysis. This tool applies a ROC analysis on each variable of a given dataset (e.g., for each operational taxonomic unit) and produces several informative outputs (like the best threshold, the true positive and true negative rates, the area under the curve, etc.) for each variable. These variables are then sorted according to their discrimination power. Such a ROC analysis has been applied to a real metabarcoding dataset related to a Hg-enriched tailing dump. We also performed an extensive comparison with other available

methods and with our previous analysis, and found better performance of the proposal when discriminating and highlighting keystone species in environmental analysis, regardless of their abundance.

3.1/ INTRODUCTION

High-throughput metabarcoding tools based on 454 gottel2011distinct, danielsen2012fungal, tedersoo2014global, op2015impact, Ion Torrent zappelini2015diversity, or Illumina MiSeq schmidt2013illumina, wu2015molecular, foulon2016impact, foulon2016environmental, metabarcoding technologies are able to produce large volumes of data at an affordable cost. In particular, when applied to contaminated soils bell2014linkage, these technologies have been used to reveal changes in microbial communities in the rhizosphere soils and plant roots. They can uncover too dominant species in the rhizosphere and endosphere of tree species under contaminant stress yergeau2015transplanting, azarbad2015microbial, bell2015early, zappelini2015diversity. Knowledge of the plant-associated microbial compartment could be used to help predict the potential recovery of disturbed lands kozdroj2000microflora. Recent data suggests that the soil microbiome may have the greatest impact on plant function during the early stages of revegetation bell2015early. The possible structural changes of indigenous microbial communities by crops constitute a major ecological concern because of the important role that microorganisms have in regulating soil conditions conrad1996soil, jeffries2003contribution. The re-establishment of belowground interactions is required to ensure a successful restoration and the creation of sustainable plant cover. However, little is known about the entirety of the microbial communities that are associated with tree roots in derelict soils. The investigation of microbial diversity and biogeography in contaminated environments is also important to more broadly identify the global environmental drivers of community composition and diversity.

Redundancy analysis (RDA), principal component analysis (PCA), or multi-dimensional scaling (MDS) are widely used in environmental metabarcoding analysis to reveal the relationship between soil and vegetation characteristics with microbial community structure, and to test the significance of each with a permutation test. In a previous paper, we used such a permutation test to establish that the bacterial and fungal communities residing within a tailings dump vs. undisturbed soil samples had significantly different compositions zappelini2015diversity. In another recent article, we used a 2-dimensional non-metric multi-dimensional scaling (NMDS) analysis combined with a permutation test to demonstrate that the site characteristics explained most of the variance in the composition of a fungal community foulon2016impact, foulon2016environmental. Based on PCA analysis, Hong et al. hong2015illumina shown that dominant genera changed in mine sites with the degree of pollution. Lallias et al. lallias2015environmental used MDS analysis coupled with a permutation test to demonstrate that microbial taxa are likely to respond to different environmental drivers and in particular, the hydrodynamics, the salinity range, and the granulometry according to varied life-history characteristics. Based on a PCA analysis and permutation test, Yergeau et al. yergeau2015transplanting highlighted key factors that should be considered when engineering the plant rhizosphere microbiome, including the presence and abundance of keystone species, the diversity

and evenness of the initial inoculum, the ecological differences between fungi and bacteria, the environmental conditions, and the plant growth stage from which the inoculum originates.

Detection of the most site-dependent OTUs is mainly done in the literature by the comparison of relative abundances (cf. our comparison with standard benchmarks). The aim of this work is to implement a new tool to easily achieve such detection. It is based on the so-called ROC curves, allowing to detect the most site-dependent OTUs and to determine, for each OTU, the ideal threshold under which one can assume to be in one site rather than in the other one.

ROC curve analysis was developed in the early 1970s [egan1975signal] and is currently used in medical statistics to determine the best threshold for a diagnostic test [zou2002receiver]. For instance, ROC curves have been used to assess the value of diagnostic tests by providing a standard measure of the ability of a test to correctly classify subjects [morrison2003receiver]. A ROC curve is a graphical representation that allows illustration of the performance of a binary classifier. This graph consists of a representation of the false positive rate in the abscissa and the true positive rate in the ordinate. The curve joins this pair of rates for each possible threshold. In addition to this ability to search the best threshold, ROC curves allow the assignment of a numerical value to the discriminatory power of the binary classifier due to their area under the curve (AUC). The more relevant the classifier is, the larger its ROC AUC is, with a value up to 1 for a perfect classifier. For a larger and more accurate explanation about ROC curve analysis, see [fawcett2006introduction].

For the sake of illustration, we have applied our tool to a real metabarcoding dataset related to a tailings dump generated by the activity of a chlor-alkali industry. Even though these data have already been studied in one of our previous works [zappelini2015diversity], the re-investigation of these data emphasizes new interesting directions of research, which is the second contribution of this article.

3.2/ MATERIAL AND METHOD

3.2.1/ ROC CURVE ANALYSIS : GENERAL CONSIDERATIONS

ROC curve analysis [egan1975signal] is an analysis that determines how accurately a quantitative variable can discriminate a binary variable. In ROC curve analysis, we consider :

- a quantitative variable “Q” and a binary variable “B”,
- a “category 0” and a “category 1”, which are the two categories of the binary variable B, and
- “negative” and “positive” subjects that are in “category 0” and “category 1”, respectively.

Each value “v” taken by the quantitative variable can be considered as a threshold that is able to classify the subjects with the following assertion (which is not always true) : “all subjects that have a quantitative variable Q that is larger or equal to v are positives, and the other ones are negatives”. According to this classification, each subject can be categorized in one of the following cases as illustrated in Table 3.1. The proportion of true positives among category number 1 is called the “true positive rate” (TPR) or “sensitivity”.

TABLE 3.1 – Meaning of the terms : “True positive”, “True negative”, “False positive”, and “False negative” in a ROC curve analysis

	Category 0 (Negative subjects)	Category 1 (Positive subjects)
Q < threshold (subject classified as negative)	True negative	False negative
Q ≥ threshold (subject classified as positive)	False positive	True positive

The proportion of true negative among category 0 is denoted as “true negative rate” (TNR) or “specificity”. Similarly, the proportion of false positives among category 0 and the proportion of false negatives among category 1 are denoted as the “false positive rate” and “false negative rate”, respectively.

The ROC curve goes through every possible combination of true positive rate and false positive rate as exemplified below in Figure 3.1. It always starts at (0,0) (i.e., no true positive and no false positive) and finishes at (1,1) (i.e., each positive subject is a true positive one, and each negative subject is false positive). When the quantitative variable is perfectly discriminating (i.e., there is a threshold such that every subject is well classified), the ROC curve goes through the point (0,1). One of the useful outputs of ROC curve analysis is the ROC area under the curve (AUC). As stated previously, the more relevant a classifier is, the larger its ROC AUC is (1 is for a perfect classifier).

3.2.2/ ROC ANALYSIS IMPLEMENTATION

Our tool receives, as input, a database such that the lines are the quantitative variables and the columns are the subjects. For each variable, the tool determines the threshold that optimizes the sum between the true positive rate and the true negative one.

When the number of subjects is balanced between the two categories, optimizing this sum between the true positive and the true negative rates is equivalent to optimize the number of well-classified subjects (WCS). One can emphasizes that the optimal threshold is not a point value but it covers the interval contained between two consecutive values of the quantitative variable. Thus, borders of this interval are named “inferior threshold” and “superior threshold” (respectively abbreviated “Inf Thres” and “Sup Thres” in Tables 3.2-3.4 and Appendices S1-2). For example, if we consider a perfect classifier, inferior and superior thresholds are respectively the greatest value of Q so that the subject is positive, and the smallest value of Q so that the subject is negative. In this work, the mean between the inferior and superior thresholds is simply named “threshold”. “Delta norm” is the difference between these two values when the variable is standardized. “Delta norm” was also used as a tiebreaker between variables that have the same ROC AUC. The outputted “preference” value (abbreviated “pref” in Tables 3.2-3.4) indicates the category that is supposed to be above the threshold (i.e., the “positive” category).

For each variable, we computed a Wilcoxon test of rank p-value. This test is the most suitable one to determine whether a ROC AUC is significant, because the Wilcoxon statistic and ROC AUC are equivalent `hanley1982meaning`. At a constant sample size, the ROC AUC corresponds to the unique Wilcoxon test of rank p-value, and the Wilcoxon test of rank p-value decreases when the ROC AUC increases. The proposed tool pro-

vides both AUC and Wilcoxon values (e.g., similar to the Pearson's correlation coefficient that is usually provided with its associated p-value). Note that the AUC values represent the strength of the link between the quantitative and binary variables, whereas the Wilcoxon value is for the significance of this link. However, in practice, most software do not use the same formula to compute the Wilcoxon-test. Thus, the equivalence between AUC and Wilcoxon can be highly dependent on the software and options. This is further developed in the section below. Finally, for each variable, we computed the number of nonzero subjects in each category (denoted by #T and #U in Tables 3.2-3.4). This information can be useful for variables with a low AUC, to determine whether the low discriminating power was either because the variable is in both sites or because it is only in a few trees.

3.2.3/ R AND PYTHON IMPLEMENTATION

To rapidly collect all the outputs from our metabarcoding dataset, we produced a dedicated python tool. This tool imports the data (if they are in text format) and directly produces an excel file as an output similar to the tables presented in this article. We have also produced a R tool to generate the same output, if we except that the Wilcoxon test is slightly different in R and in Python. Let us remark that that there is a range of R packages currently available for ROC analysis such as "pROC" `robin2011proc` or "ROCR" `sing2005rocr`. Some solutions also exist under Python like the "roc_curve" function of the "sklearn.metrics" package `scikit-learn`. These libraries allow to perform a ROC analysis between a quantitative and a binary variable. Compared to this state-of-the-art, the particularity of our scripts is that they compute the ROC analysis for each quantitative variable of the database, sort the results, and further group the outputs into a single table that is exported as an excel file. The output tables highlight the most discriminating variables as exemplified in Tables 3.2 to 3.4. The computational details of the parameters set in the package are further described below[1].

3.3/ ROC ANALYSIS APPLIED TO A CASE STUDY

ROC analysis described previously has been applied to our metabarcoding dataset previously studied `zappelini2015diversity`, by defining three different groups of variables : (1) bacteria operational taxonomic units (OTUs), (2) fungal OTUs, and (3) soil physico-chemical properties obtained from the two experimental locations (the tailings dump and undisturbed soils). Figure 3.1 provides examples of ROC curves for the carbon-nitrogen ratios (Fig. 1a), aluminium (Fig. 1b), and soil pH (Fig. 1c). The soil pH ROC curve went through the coordinate (0,1), which was characteristic of a perfectly discriminating variable, whereas the carbon-nitrogen ratio ROC curve was close to the first diagonal (y=x curve in green), which is characteristic of a poorly discriminating variable. The aluminium ROC curve, for its part, shown an intermediate discriminating variable.

1. Note that the variance used to calculate the "Delta norm" is the unbiased variance $\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\bar{x})^2$. This is the formula used by default in R while Python uses the biased variance $\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^2$. Thus, options have been set to force the unbiased variance in Python, leading to two harmonized outputs.

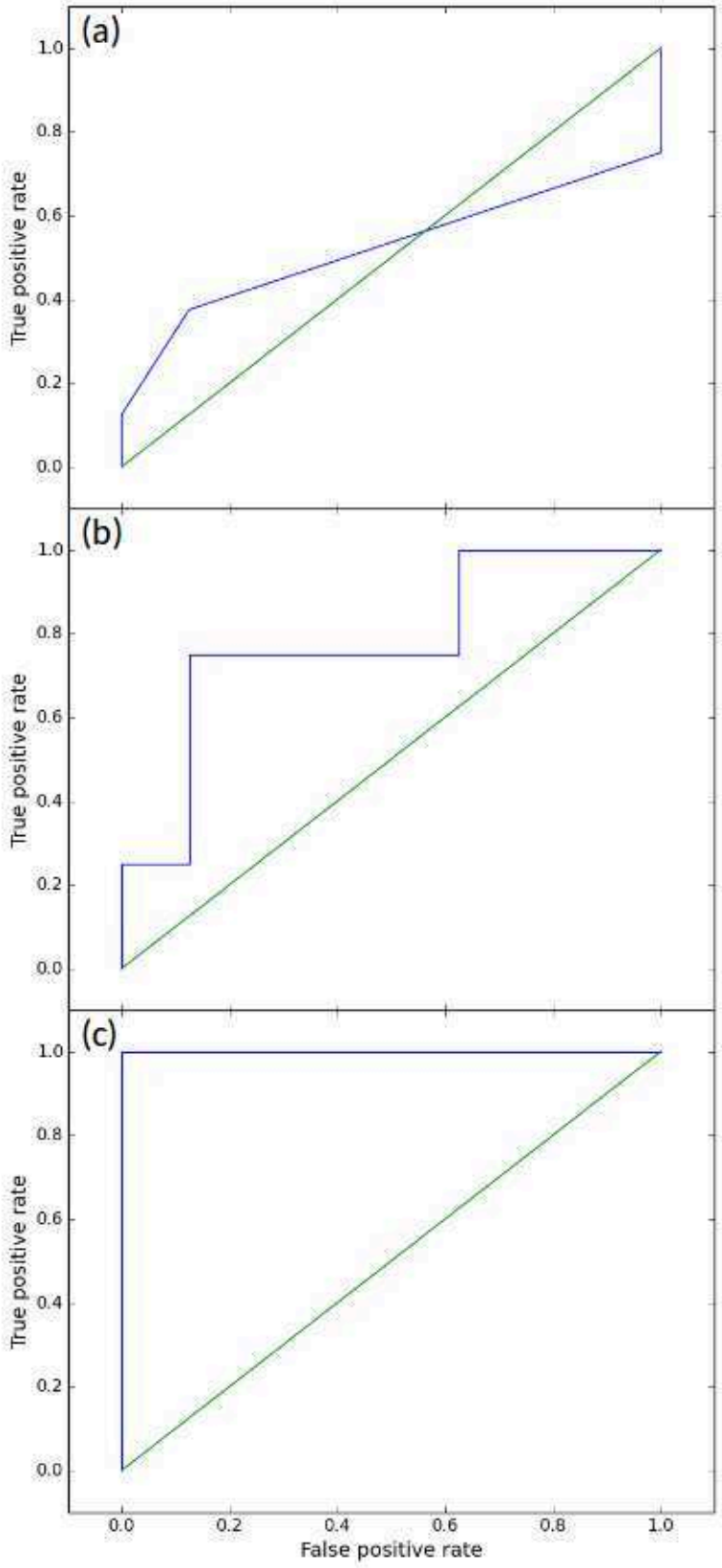

FIGURE 3.1 – ROC curves constructed by plotting the true positive rate and false positive one associated with each unique value of the indicator variable. An indicator variable with a poor discriminatory power (C/N ratio) will have an AUC near 0.5 (c), a variable with an intermediate discriminatory power (Al) will have an AUC close to 0.75 (b), and an indicator variable with a high discriminatory power (pH) will have a curve with an AUC near 1 (a).

ROC AUCs were further computed for each group (i.e., bacterial OTUs, fungal OTUs, and soil physico-chemical variables), and the groups of variables were sorted by decreasing AUCs. Table 3.2 shows the results for all the soil physico-chemical variables, whereas Tables 3.3 and 3.4 show the top 30 most discriminating bacterial and fungal OTUs, respectively. The complete lists of sorted variables for bacterial and fungal OTUs are available as supplementary data (supplemental Tables S1 and S2). In these tables, Delta_norm is used as a tiebreaker between the variables that have the same ROC AUC. This sorting method allowed us to rapidly detect the variables that best discriminate the experimental site.

TABLE 3.2 – ROC AUCs and related parameters of all soil physico-chemical variables. AUC, area under the curve ; Delta norm, difference between the threshold inferior and the threshold superior ; TPR, true positive rate ; TNR, true negative rate ; WCS, well-classified subjects ; Pref, output preference ; Inf Thres, inferior threshold ; Sup Thres, superior threshold ; #T, nonzero subjects in the tailing dump samples ; #U nonzero subjects in the undisturbed soil samples. For each variable, we computed a Wilcoxon test of rank p-value.

Variable	Threshold	Pref	TPR	TNR	Sum	WCS	AUC	Delta norm	Inf Thres	Sup Thres	Wilcoxon	#U	#T
Ph	7.85	Lagoon	1	1	2	16	1.000	0.236	7.70	8.00	0.0008	8	8
Ca	41959	Lagoon	1	1	2	16	1.000	0.230	29683	54234	0.0008	8	8
Na	196	Lagoon	1	1	2	16	1.000	0.164	166	226	0.0008	8	8
Sr	53.0	Lagoon	1	1	2	16	1.000	0.161	44.5	61.5	0.0008	8	8
Carbonate	3.85	Lagoon	1	0.875	1.875	15	0.984	0.030	0.000	7.70	0.0011	1	8
Pb	20.6	Natural	1	0.875	1.875	15	0.969	0.091	20.2	21.1	0.0016	8	8
As	14.0	Lagoon	0.875	1	1.875	15	0.969	0.055	13.8	14.2	0.0016	8	8
Co	5.18	Natural	1	0.875	1.875	15	0.922	0.155	4.98	5.38	0.0046	8	8
Thin silt	447	Lagoon	0.75	1	1.75	14	0.906	0.072	441	453	0.0063	8	8
CEC	96.0	Natural	1	0.625	1.625	13	0.906	0.024	95.0	97.0	0.0063	8	8
P_2O_5	0.185	Lagoon	1	0.875	1.875	15	0.883	0.862	0.110	0.260	0.0101	8	8
CaO	8.83	Lagoon	1	0.875	1.875	15	0.875	0.684	7.16	10.5	0.0117	8	8
Hg	1.49	Lagoon	1	0.875	1.875	15	0.875	0.242	1.23	1.75	0.0117	6	8
P	541	Lagoon	0.875	0.875	1.75	14	0.844	0.119	525	557	0.0209	8	8
Fe	10433	Natural	1	0.625	1.625	13	0.797	0.026	10348	10518	0.0460	8	8
Al	15498	Natural	0.75	0.875	1.625	13	0.781	0.416	13424	17572	0.0587	8	8
Sn	0.951	Lagoon	1	0.625	1.625	13	0.781	0.018	0.936	0.966	0.0587	8	8
Clay	195	Natural	0.75	0.875	1.625	13	0.766	0.191	184	205	0.0742	8	8
Coarse sand	14.0	Natural	0.875	0.625	1.5	12	0.766	0.156	11.0	17.0	0.0742	8	8
Mg	1905	Natural	0.75	0.875	1.625	13	0.750	0.080	1868	1942	0.0929	8	8
Na_2O	0.025	Lagoon	1	0.75	1.75	14	0.750	0.060	0.024	0.025	0.0929	8	8
Large silt	250	Natural	0.625	0.875	1.5	12	0.750	0.023	249	251	0.0929	8	8
Ni	14.1	Natural	0.75	0.75	1.5	12	0.734	0.321	12.6	15.5	0.1152	8	8
Cr	25.1	Natural	0.75	0.875	1.625	13	0.719	0.297	23.0	27.2	0.1415	8	8
Bo	0.780	Lagoon	0.625	0.875	1.5	12	0.719	0.137	0.760	0.800	0.1415	8	8
Cd	0.183	Lagoon	1	0.5	1.5	12	0.703	0.179	0.156	0.209	0.1722	7	8
K_2O	0.315	Lagoon	0.875	0.5	1.375	11	0.695	0.146	0.300	0.330	0.1893	8	8
K	1992	Natural	0.75	0.75	1.5	12	0.688	0.042	1968	2015	0.2076	8	8
OM	36.7	Natural	1	0.25	1.25	10	0.648	0.434	31.5	41.9	0.3184	8	8
Total C	21.2	Natural	1	0.25	1.25	10	0.648	0.432	18.2	24.2	0.3184	8	8
Se	0.490	Natural	0.25	1	1.25	10	0.625	1.478	0.000	0.979	0.4008	2	0
Total N	1.78	Natural	1	0.375	1.375	11	0.617	0.015	1.77	1.79	0.4309	8	8
Mn	135	Lagoon	1	0.5	1.5	12	0.594	0.304	116	155	0.5286	8	8
Thin sand	15.5	Natural	1	0.25	1.25	10	0.578	0.173	10.0	21.0	0.5995	8	8
Sb	0.288	Lagoon	0.5	0.75	1.25	10	0.563	0.663	0.000	0.576	0.6744	2	4
Si	281	Natural	0.5	0.75	1.25	10	0.563	0.138	277	284	0.6744	8	8
Cu	9.94	Natural	0.625	0.625	1.25	10	0.563	0.082	9.71	10.2	0.6744	8	8
MgO	0.150	Lagoon	0.75	0.5	1.25	10	0.531	0.410	0.140	0.160	0.8336	8	8
C/N	11.5	Natural	0.375	0.875	1.25	10	0.523	0.938	11.0	12.0	0.8748	8	8

TABLE 3.3 – ROC AUCs and related parameters of the top 30 most discriminating bacterial OTUs. AUC, area under the curve ; Delta norm, difference between the threshold inferior and the threshold superior ; WCS, well-classified subjects ; Pref, output preference ; Inf Thres, inferior threshold ; Sup Thres, superior threshold ; #T, nonzero subjects in the tailing dump samples ; #U nonzero subjects in the undisturbed soil samples. For each variable, we computed a Wilcoxon test of rank p-value. In the column "Rel ab in U", the number without parenthesis indicates the percentage of the considered OTU in the undisturbed soil (i.e., $100 \times \frac{\text{sequences of this OTU} \in \text{the undisturbed soil}}{\text{all sequences} \in \text{the undisturbed soil}}$) while the number in the parentheses indicates the percentage of the undisturbed soil for the considered OTU (i.e., $100 \times \frac{\text{sequences of this OTU} \in \text{the undisturbed soil}}{\text{sequence of this OTU} \in \text{both sites}}$) , for OTUs that satisfy $\frac{\text{sequences of this OTU} \in \text{the undisturbed soil}}{\text{all sequences} \in \text{the undisturbed soil}} \geq 0.02$ or $\frac{\text{sequences of this OTU} \in \text{the tailings dump}}{\text{all sequences} \in \text{the tailings dump}} \geq 0.02$ in `zappelini2015diversity`. Similar calculations for the tailings dump appear in column "Rel ab in T". Rank, ranking of the most abundant OTUs, as determined by the standard method. The full data set is provided in appendix S1.

OTU	ID	Threshold	Pref	WCS	AUC	Delta norm	Inf Thresh	Sup Thresh	Wilcoxon	#U	#T	Rel Ab in U	Rel Ab in T	rank
Bosea	b'236	2	T	16	1.000	0.746	1	3	0.0008	3	8			
unclassified	b'346	13	T	16	1.000	0.526	8	18	0.0008	5	8			
Tsukamurella	b'533	5.5	T	16	1.000	0.501	4	7	0.0008	4	8			
Iamia	b'458	26.5	T	16	1.000	0.497	21	32	0.0008	8	8			
unclassified	b'423	339.5	T	16	1.000	0.293	311	368	0.0008	8	8	1.8 (25.0)	4.1 (75.0)	7
unclassified	b'214	12.5	T	16	1.000	0.284	11	14	0.0008	6	8			
Methylotenera	b'332	1.5	T	16	1.000	0.282	1	2	0.0008	2	8			
unclassified	b'076	14.5	T	16	1.000	0.279	12	17	0.0008	6	8			
Aeromicrobium	b'503	12	T	16	1.000	0.279	10	14	0.0008	6	8			
unclassified	b'539	2	T	16	1.000	0.277	1	3	0.0008	5	8			
Agrobacterium	b'221	4.5	T	16	1.000	0.271	4	5	0.0008	8	8			
unclassified	b'051	2.5	T	16	1.000	0.251	2	3	0.0008	5	8			
Haliangium	b'373	2.5	T	16	1.000	0.249	2	3	0.0008	4	8			
unclassified	b'081	5.5	T	16	1.000	0.225	5	6	0.0008	6	8			
unclassified	b'424	80.5	T	16	1.000	0.222	73	88	0.0008	8	8			
Cytophaga	b'024	89.5	T	16	1.000	0.205	81	98	0.0008	8	8			
Stenotrophomonas	b'404	1	T	16	1.000	0.198	0	2	0.0008	0	8			
Marmoricola	b'506	26	T	16	1.000	0.191	24	28	0.0008	7	8			
unclassified	b'074	1.5	T	16	1.000	0.174	1	2	0.0008	1	8			
Aminobacter	b'213	4.5	T	16	1.000	0.158	4	5	0.0008	2	8			
Polaromonas	b'311	6.5	T	16	1.000	0.124	5	8	0.0008	5	8			
Pseudomonas	b'395	310	T	16	1.000	0.039	284	336	0.0008	8	8	1.0 (5.0)	16.1 (95.0)	3
Steroidobacter	b'422	25.5	T	15	0.992	0.859	9	42	0.0009	6	8			
Hyphomicrobium	b'242	30	T	15	0.992	0.663	20	40	0.0009	8	8			
Kaistobacter	b'290	90	T	15	0.992	0.406	71	109	0.0009	8	8			
unclassified	b'576	4	T	15	0.984	0.889	2	6	0.0011	6	8			
Lawsonia	b'363	0.5	T	15	0.984	0.478	0	1	0.0011	1	8			
unclassified	b'080	4	T	15	0.984	0.439	3	5	0.0011	4	8			
Pimelobacter	b'508	6.5	T	15	0.984	0.424	3	10	0.0011	7	8			
unclassified	b'112	2.5	T	15	0.984	0.406	2	3	0.0011	7	8			

TABLE 3.4 – ROC AUCs and related parameters of the top 30 most discriminating fungal OTUs. AUC, area under the curve ; Delta norm, difference between the threshold inferior and the threshold superior ; WCS, well-classified subjects ; Pref, output preference ; Inf Thres, inferior threshold ; Sup Thres, superior threshold ; #T, nonzero subjects in the tailing dump samples ; #U nonzero subjects in the undisturbed soil samples. For each variable, we computed a Wilcoxon test of rank p-value. In the column "Rel ab in U", the number without parenthesis indicates the percentage of the considered OTU in the undisturbed soil (i.e., $100 \times \frac{\text{sequences of this OTU} \in \text{the undisturbed soil}}{\text{all sequences} \in \text{the undisturbed soil}}$) while the number in the parentheses indicates the percentage of the undisturbed soil for the considered OTU (i.e., $100 \times \frac{\text{sequences of this OTU} \in \text{the undisturbed soil}}{\text{sequence of this OTU} \in \text{both sites}}$) , for OTUs that satisfy $\frac{\text{sequences of this OTU} \in \text{the undisturbed soil}}{\text{all sequences} \in \text{the undisturbed soil}} \geq 0.02$ or $\frac{\text{sequences of this OTU} \in \text{the tailings dump}}{\text{all sequences} \in \text{the tailings dump}} \geq 0.02$ in `zappelini2015diversity`. Similar calculations for the tailings dump appear in column "Rel ab in T". Rank, ranking of the most abundant OTUs, as determined by the standard method. The full data set is provided in appendix S2.

OTU	ID	Thre shold	Pref	WCS	AUC	Delta norm	Inf Thresh	Sup Thresh	Wilc oxon	#U	#T	Rel Ab in U	Rel Ab in T	rank
unclassified	f'064	15	T	16	1.000	0.232	13	17	0.0008	7	8			
Pyrenochaeta	f'033	110.5	T	16	1.000	0.159	95	126	0.0008	8	8	0.2 (8.0)	2.2 (92.0)	3
unclassified	f'114	2.5	U	16	1.000	0.118	2	3	0.0008	8	5			
Cryptococcus	f'462	118	U	16	1.000	0.040	113	123	0.0008	8	8	2.0 (88.0)	0.3 (12.0)	7
unidentified	f'371	1	U	15	0.992	0.222	0	2	0.0009	8	1			
unidentified	f'435	667	U	15	0.984	0.113	630	704	0.0011	8	8	7.1 (80.0)	2.0 (20.0)	9
Clavulina	f'392	13	U	15	0.984	0.012	12	14	0.0011	8	6			
unidentified	f'063	10.5	T	15	0.977	0.059	10	11	0.0014	7	8			
Ceratobasidium	f'388	4.5	T	14	0.969	0.186	2	7	0.0016	5	8			
Laccaria	f'340	13.5	U	15	0.969	0.125	12	15	0.0016	8	7			
Verticillium	f'258	10	T	15	0.961	0.088	8	12	0.0019	8	8			
Plectosphaerella	f'257	60.5	T	15	0.953	0.340	28	93	0.0023	7	8			
Lactarius	f'424	29	U	14	0.953	0.054	26	32	0.0023	8	8			
unclassified	f'476	127.5	U	15	0.953	0.003	127	128	0.0023	8	8	2.0 (88.0)	0.3 (12.0)	8
Schizothecium	f'281	10.5	T	15	0.945	0.515	8	13	0.0028	4	8			
unclassified	f'248	107	T	15	0.938	0.250	92	122	0.0033	8	8			
Arthrinium	f'250	0.5	T	15	0.938	0.071	0	1	0.0033	0	7			
Cladophialophora	f'068	1.5	U	13	0.930	0.793	0	3	0.0039	8	3			
Trichosporon	f'472	20	U	14	0.930	0.080	17	23	0.0039	8	8			
Rhizoscyphus	f'112	1.5	U	14	0.922	0.242	1	2	0.0046	8	4			
unidentified	f'180	6	U	14	0.922	0.235	5	7	0.0046	8	6			
Fusarium	f'235	63.5	T	14	0.922	0.158	58	69	0.0046	8	8			
unclassified	f'104	1.5	U	15	0.922	0.073	1	2	0.0046	7	2			
Hebeloma	f'322	111	T	15	0.922	0.022	103	119	0.0046	8	8	0.3 (7.0)	4.9 (93.0)	2
unclassified	f'175	31.5	T	14	0.922	0.017	30	33	0.0046	8	8			
unidentified	f'118	16.5	U	14	0.922	0.015	16	17	0.0046	8	7			
Emericellopsis	f'226	0.5	T	14	0.914	0.292	0	1	0.0054	1	7			
Cladosporium	f'007	9.5	T	14	0.914	0.085	8	11	0.0054	8	8			
Volutella	f'240	99	T	14	0.906	0.371	71	127	0.0063	8	8			
Meliniomyces	f'140	0.5	U	14	0.906	0.131	0	1	0.0063	7	1			

For the physico-chemical variables, pH, calcium carbonate (CaCO3) concentration and exchangeable calcium oxide (CaOex), Hg and As were found to be the most discriminating soil parameters (Table 3.2), which is in agreement with our previous study `zappelini2015diversity`. However, ROC curves further indicated that Ca, Na, and Sr were the three additional most relevant parameters to discriminate the two sites with AUC values of 1, which were similar to the AUC of pH.

For the bacterial community, 22 OTUs perfectly discriminated the two sites with an AUC of 1 (Table 3.3). Among these sets of OTUs, 2 were relatively abundant (*unclassified b'423* and *Pseudomonas*) and were already identified as key bacteria by the method used in our previous study `zappelini2015diversity`. However, the ROC curves further indicate that the two sites may also be well discriminated by less represented OTUs, such as *Bosea*, *Methylotenera*, *Agrobacterium*, *Aminobacter*, and *Polaromonas* OTUs and additional unclassified OTUs that belong to the *Proteobacteria*. Additional OTUs from other phyla (*Tsukamurella*, *Iamia*, *Aeromicrobium*, and *Marmoricola*) also showed AUC values of 1 and were also less represented. *Bosea* OTUs, corresponding to the most discriminating bacteria, was represented in each tree from the tailings dump by at least 3 sequences, whereas trees from the undisturbed soil owned, at the most, 1 sequence. If we consider the 30 most discriminating bacterial OTUs, all of them were more abundant on the tailings dump (Table 3.3). If we consider 74 OTUs that lead to a correct classification of 15 or 16 trees, only 11 of them were related to the undisturbed soil (supplemental Tables S1). This result is consistent with our previous analysis `zappelini2015diversity`, which showed that more bacteria OTUs were significantly more present in the tailings dump compared to the undisturbed soil.

For the fungal community, only 4 OTUs perfectly discriminated the two sites with an AUC of 1 (Table 3.4) with the already identified *Pyrenocheta* (which was more present in the tailings dump) and *Cryptococcus* (which was more present in the undisturbed soil, see Table 3.4) OTUs. ROC curves further highlighted the two fungal OTUs that perfectly discriminated the two sites, which were however unclassified OTUs in both soils. In contrast with the bacterial OTUs, among the 30 most discriminating OTUs, 50% discriminated the tailings dump and 50% discriminated the undisturbed soil. With an initial sample size of 16 (8 undisturbed soil and 8 tailings dump), Wilcoxon values of 0.001 and 0.05 were reached for AUC values of approximately 0.99 and 0.79, respectively.

3.4/ COMPARISON WITH STANDARD BENCHMARK

The method used in our previous study `zappelini2015diversity`, to detect the OTUs most impacted by the pollution, consists of selecting the most abundant sequences by retaining OTUs such that $\frac{\text{sequences of this OTU} \in \text{the undisturbed soil}}{\text{all sequences} \in \text{the undisturbed soil}} \geq 0.02$ or $\frac{\text{sequences of this OTU} \in \text{the tailings dump}}{\text{all sequences} \in \text{the tailings dump}} \geq 0.02$. Then, among these sequences, to detect those for which the difference of relative abundance is the most important one, as defined below :

$$100 \times \left| \frac{\text{sequence of this OTU} \in \text{the tailings dump - sequence of this OTU} \in \text{the undisturbed soil}}{\text{sequence of this OTU} \in \text{both sites}} \right|$$

In this standard method, widespread among the scientific community `schmidt2013illumina`, `bell2014linkage`, `tedersoo2014global`,

op2015impact, yergeau2015transplanting, azarbad2015microbial, bell2015early, wu2015molecular, zappelini2015diversity, foulon2016impact, foulon2016environmental, the OTU selection stage is essential. Otherwise, for example, OTUs present in 1 copy on a single tree would appear to have a maximum relative abundance difference of 100%. This selection criterion of most abundant OTUs can appear as arbitrary, while ROC curves do not need such a selection. An OTU present on only one tree will necessarily appear at the bottom of the table of the ROC curves and will be considered as non-discriminating by the Wilcoxon test. Thus the ROC curves allow to consider all the OTUs and to identify the most discriminating ones even among the least abundant ones (such as Bosea), as already stated in the previous section.

Another important aspect of ROC curves is that they allow to provide a criterion for binary classification according to the OTU (for instance, number of *Bosea* sequences $\geq 2 \rightarrow$ tailings dumps), while the standard method does not allow this. We illustrated this important consideration by comparing the data obtained through the classical method, with the ROC approach. In Tables 3.3 and 3.4 and in supplementary Tables S1 and S2, the last three columns correspond to the standard method. In the case where an OTU is abundant enough ($> 2\%$), the number without parenthesis in the "Rel ab in U" column provides the percentage of the considered OTU in the undisturbed soil $\left(i.e., \frac{\text{sequences of this OTU} \in \text{the undisturbed soil}}{\text{all sequences} \in \text{the undisturbed soil}}\right)$ while the number inside the parentheses indicates the percentage of the undisturbed soil for the considered OTU $\left(i.e., \frac{\text{sequences of this OTU} \in \text{the undisturbed soil}}{\text{sequences of this OTU} \in \text{both sites}}\right)$. Similar computations are provided in column "Rel ab in T" for the tailings dump. It can be seen that, logically, the site on which the OTU is preferentially found is the same, regardless of the approach used (ROC curves or standard method). The "Rank" column provides the ranking of the OTU obtained with the classical method, for OTU $> 2\%$. The classical method allows to rank the most important OTUs based on their abundance, whereas the ROC curve analytical tool simply allows to rank (columns "AUC" and "Delta norm") the OTUs, independently of their abundance. This ROC curve analysis implemented with R or Python thus constitutes an interesting tool to discriminate the two sites (e.g., highlighting the bacterial OTUs that were poorly represented, which was exemplified by the *Bosea* case in this study).

RESOURCES AND DATA ACCESSIBILITY

Each code is freely available on the "GitHub" repository, which also contains explanations of the various functions used and examples detailing all the steps required to collect and use data with Python or R.

— for the R codes : https://github.com/SergeMOULIN/ROC_R
— for the Python codes : https://github.com/SergeMOULIN/ROC_python

This work was supported by the French Environment and Energy Management Agency [PROLIPHYT 1172C0053], the Région Franche-Comté [Environnement-Homme-Territoire 2014-069], the Pays de Montbéliard Agglomération [13/070-203-2015], and the French national programme EC2CO-MicrobiEen FREIDI-Hg. C.Z. and S.M. received a PhD grant from the French Ministry of Higher Education and Research. M.C. and C.Z planned and designed the research, performed experiments and produced the data ; S.M. and C.G. computed the scripts and analyzed the data ; C.Z., M.C., S.M. and C.G. wrote the manuscript.

3.5/ COMPLÉMENTS

Dans le cadre de ce projet nous avons également réalisé des combinaisons ACP-GMM pour visualiser les données en deux dimensions et voir comment ces données se répartissent en clusters.

Les figures 3.2, 3.3 et 3.4 représentent respectivement les composants physico-chimiques du sol, les OTUs des bactéries et enfin les OTUs des champignons. Chacune de ces figures est partagée en deux. Dans la partie de gauche, on représente les résultats de la GMM à 2 clusters. C'est-à-dire qu'on colorie en rouge les éléments d'un cluster et en bleu les éléments de l'autre cluster. Dans la partie de droite, on colorie les éléments en fonction de leurs résultats à un test de permutation visant à tester si l'élément est significativement plus présent dans la zone polluée ou dans la zone non polluée. On colorie ici en noir les éléments significativement plus présents dans la lagune polluée que dans la zone naturelle. On colorie en jaune les éléments significativement plus présents dans la zone naturelle que dans la lagune. Et enfin on colorie en bleu les éléments pour lesquels le test n'est pas significatif. Sur ces figures, on constate que tous les éléments "préférant la lagune" se retrouvent systématiquement dans un même cluster, tandis que tous les éléments "préférant le milieu naturel" se retrouvent dans l'autre cluster. Les éléments non significatifs se retrouvent dans les deux clusters. Ceci tend à indiquer que la préférence pour la lagune ou le milieu naturel est un élément essentiel dans la constitution de ces clusters.

Par ailleurs la figure 3.5 indique les coordonnées des centres des clusters pour chaque GMM dans l'espace de dimension 16 (Il y a 16 arbres c'est pourquoi les OTUs des bactéries sont des points d'un espace de dimension 16, tout comme les OTUs des champignons et les composantes physico-chimiques). Dans le nom des arbres, la lettre "P" signifie que l'arbre en question est un peuplier, la lettre "S" signifie que l'arbre est un saule, la lettre "N" signifie que l'arbre est dans la zone naturelle tandis que la lettre "L" signifie que l'arbre se trouve dans la lagune polluée. Ainsi par exemple l'arbre "PL11" désigne un peuplier présent sur la lagune. Pour plus de lisibilité, on a surligné en jaune pour chaque arbre le centre de cluster dont la coordonnée selon cet arbre est la plus importante. On voit bien apparaître, ici également, qu'à chaque fois, un cluster représente les éléments plus présents sur la lagune et l'autre cluster représente les éléments plus présents dans la zone naturelle. Notons que les arbres SN26 et SL6 qui ont un comportement "surprenant" à cet égard avaient déjà été repérés comme atypiques par les analyses précédentes dans l'article `zappelini2015diversity`.

L'intérêt de ces analyses est de montrer que des analyses "sans a priori" telles que l'ACP et la GMM, mettent en évidence la distinction de site (lagune vs zone naturelle) comme un élément structurant de la distribution des OTUs et des composants physico-chimiques. Toutefois, ces analyses laissent une grande place à l'interprétation "subjective" des résultats. C'est pourquoi nous avons préféré axer davantage notre article sur les résultats liés aux courbes ROC.

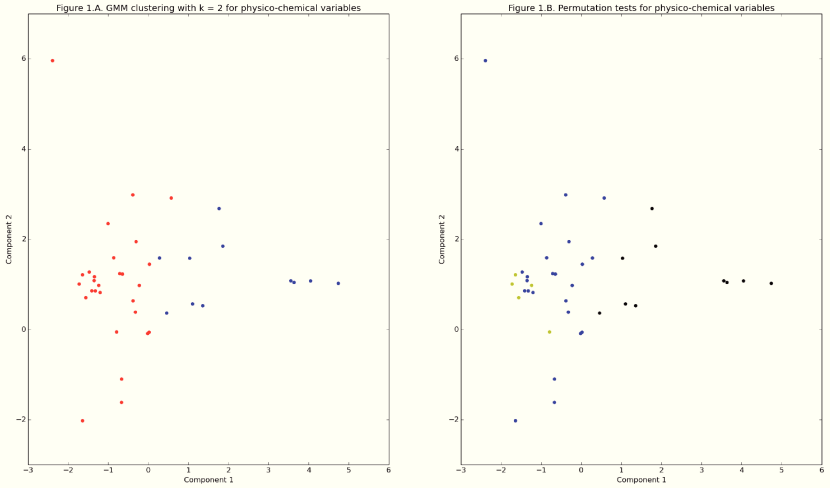

FIGURE 3.2 – GMM et tests de permutations appliqués aux composants physico-chimiques

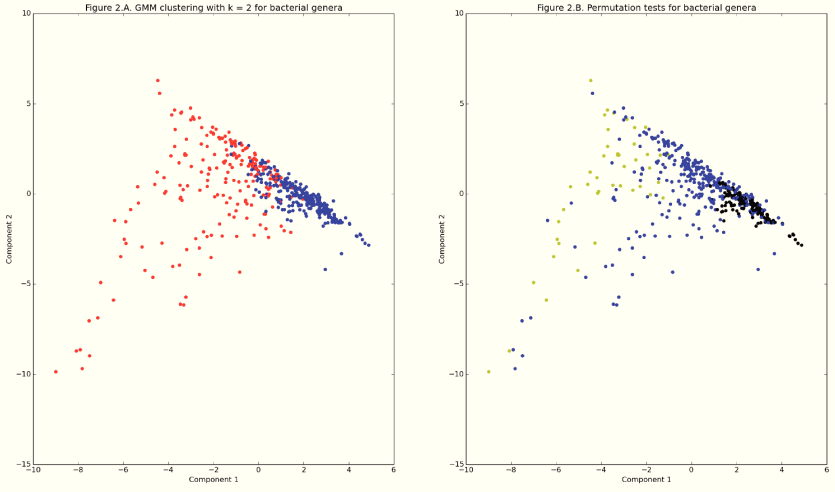

FIGURE 3.3 – GMM et tests de permutations appliqués aux bactéries

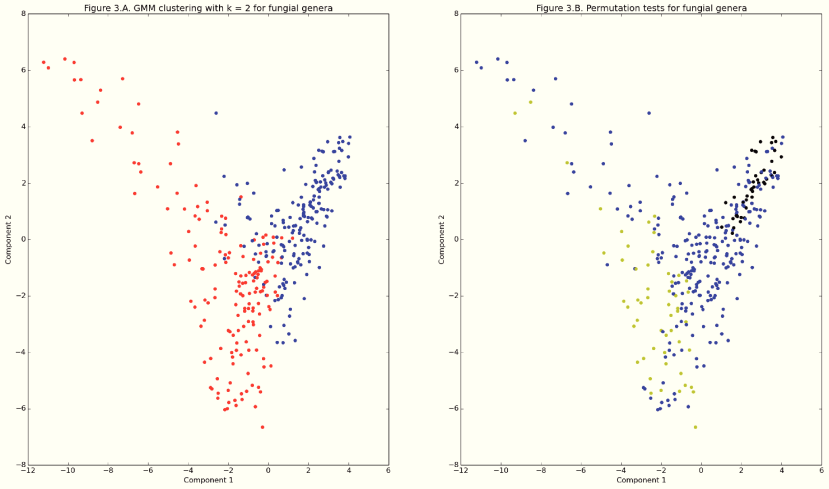

FIGURE 3.4 – GMM et tests de permutations appliqués aux champignons

	Bacteria OTU		Fungal OTU		Physico-chemical properties	
	Center 1	Center 2	Center 1	Center 2	Center 1	Center 2
PL11	1,481	0,679	0,265	1,629	0,885	1,321
PL20	1,025	0,384	0,095	1,595	0,692	1,437
PL21	1,661	0,797	0,973	0,766	0,849	1,492
PL22	1,584	0,894	0,881	1,175	1,077	1,434
SL1	2,028	0,702	0,105	0,998	0,885	2,099
SL23	1,696	0,906	0,282	1,693	0,772	1,615
SL3	1,361	0,524	0,184	1,706	0,642	2,051
SL6	1,074	0,494	0,437	1,689	1,197	0,870
PN13	0,175	0,818	0,838	0,135	0,737	0,285
PN15	0,771	1,824	1,383	0,612	0,867	0,295
PN27	0,778	2,216	1,898	0,559	0,830	0,319
PN8	0,176	1,937	2,246	0,453	0,873	0,354
SN25	0,884	1,165	1,400	0,357	1,474	0,524
SN26	0,495	0,284	1,450	0,428	2,136	1,207
SN31	0,479	1,125	1,724	1,624	1,037	0,355
SN32	0,333	1,250	1,838	0,580	1,047	0,344
effective	278	208	165	186	29	11
cluster color	blue	red	red	blue	red	blue

FIGURE 3.5 – Centre des clusters

4

RÉGRESSION POLYTOMIQUE SPARCE

Cette partie présente nos travaux sur la régression polytomique ordonnée. Ce type de régression s'applique dans les cas où la variable à expliquer est de type polytomique ordonnée (typiquement une tumeur qui aurait plusieurs niveaux de gravité par exemple). Dans ces travaux, nous appliquons à cette régression polytomique ordonnée une pénalité de norme ℓ_1 semblable à celle du LASSO `Tibshirani:JRSSB96`. Les parties de "Abstract" jusqu'à la partie présentent l'article publié dans le cadre de la conférence WABI 18 en août 2018 sous le titre "l1-Penalised Ordinal Polytomous Regression Estimators with Application to Gene Expression Studies" qui résume ces travaux.

ABSTRACT

Qualitative but ordered random variables, such as severity of a pathology, are of paramount importance in biostatistics and medicine. Understanding the conditional distribution of such qualitative variables as a function of other explanatory variables can be performed using a specific regression model known as ordinal polytomous regression. Variable selection in the ordinal polytomous regression model is a computationally difficult combinatorial optimisation problem which is however crucial when practitioners need to understand which covariates are physically related to the output and which covariates are not. One easy way to circumvent the computational hardness of variable selection is to introduce a penalised maximum likelihood estimator based on some well chosen non-smooth penalisation function such as, e.g., the ℓ_1-norm. In the case of the Gaussian linear model, the ℓ_1-penalised least-squares estimator, also known as LASSO estimator, has attracted a lot of attention in the last decade, both from the theoretical and algorithmic viewpoints. However, even in the Gaussian linear model, accurate calibration of the relaxation parameter, i.e., the relative weight of the penalisation term in the estimation cost function is still considered a difficult problem that has to be addressed with caution. In the present paper, we apply ℓ_1-penalisation to the ordinal polytomous regression model and compare several hyper-parameter calibration strategies. Our main contributions are : (a) a useful and simple ℓ_1 penalised estimator for ordinal polytomous regression and a thorough description of how to apply Nesterov's accelerated gradient and the online Frank-Wolfe methods to the problem of computing this estimator, (b) a new hyper-parameter calibration method for the proposed model, and (c) a code which can be freely used that implements the proposed estimation procedure.

4.1/ INTRODUCTION

Ordinal polytomous variables are of paramount importance in bioinformatics where practitioners may have to tackle qualitative but ordered data such as, e.g., the severity of a certain type of cancer Tibshirani:JRSSB96, tibshirani1997lasso, chretien2015investigating, chretien2015using, chretien2016bregman, etc. Understanding how such variables can be explained by other variables such as, e.g., gene expressions, can help the research community investigate the influence of certain genes in the pathology under study. Oftentimes, only a small number of genes are relevant to the statistical modelling and variable selection needs to be performed in order to detect which of them should be ignored and which of them should not. The ordinal polytomous regression model is an adaptation of the classical regression model which is extremely well suited for this type of problem, and the goal of the present paper is to propose efficient approaches to the estimation and variable selection problems for this specific model.

4.1.1/ WHEN THE NUMBER OF COVARIATES EXCEEDS THE NUMBER OF OBSERVATIONS : THE BLESSING OF SPARSITY

One important additional problem in standard gene expression studies is that the number of observations (e.g. patients) is often much smaller than the number of covariates (e.g. genes). In such cases, the problem cannot be expected to be solvable without some additional structure because the number of unknowns is larger than the number of observations. The main structural assumption which is usually made in such cases is that some sparsity property holds. In the exemple of gene expression analysis, it is usually considered natural to assume that only a small number of genes have significant influence on the output under study. Therefore, only a small number of regression coefficients should be nonzero in the estimator, although we cannot know before hand which are the ones which should be selected. Selecting the right variables in regression is often called “support recovery”. Various approaches to variable selection have been proposed in the statistical literature. In practical applications, the most extensively used selection methods are the forward selection and the AIC/BIC information criteria based approaches akaike1998information, schwarz1978estimating miller2002subset. Such methods however, can hardly be applied in situations where the number of covariates, e.g. genes, is large and one usually resorts to convex optimisation based strategies such as the LASSO Tibshirani:JRSSB96 and its generalisations to nonlinear models tibshirani1997lasso, van2008high.

4.1.2/ PREVIOUS WORK ON VARIABLE SELECTION VIA ℓ_1-NORM PENALISATION

Convex optimisation based variable selection approaches are often based on penalised log-likelihood estimation, where the penalisation term is the ℓ_1-norm. In the linear model, it was discovered in candes2007dantzig that under certain specific properties of the design matrix, known as the Restricted Isometry Property, the ℓ_1-norm penalised least ℓ_∞ estimator, aka the Dantzig estimator, would recover the location of the non-zero components exactly. This type of result, was then proven for the ℓ_1-penalised least ℓ_2 estimator, aka the LASSO estimator under weaker assumptions, including incoherence of the design matrix in candes2009near. The work bickel2009simultaneous provided interes-

ting alternative views on the statistical properties of the LASSO and Dantzig estimators which are still extensively used in the current literature on this topic.

Even when neither the Restricted Isometry Property nor the incoherence assumptions are satisfied, the mere computational tractability of ℓ_1-penalisation based estimators makes them the method of choice when the problem size is prohibitively large.

4.1.3/ THE PROBLEM OF HYPER-PARAMETER CALIBRATION

The main advantage of ℓ_1-based penalisation is to reduce the estimation problem to a convex optimisation one if the hyper-parameter, i.e. the relative weight associated with the ℓ_1-penalisation term, is calibrated to an appropriate value. In practice however, finding the right value for this hyper-parameter is often a complicated issue.

Most theoretical works come up with a formula for the hyper-parameter, see e.g. candes2009near. Such types of results are very important because they prove existence of a value of the hyper-parameter that will allow exact support recovery of the sparse regression vector under appropriate, e.g. incoherence assumptions of the design matrix. The theoretical value often gives the right order of dependencies with respect to the dimension of the problem, the standard deviation of the noise, and other important structural parameters, and is therefore a good indicator of how well conditioned the problem is, at least in theory.

In practice, however, the noise level is not known beforehand and therefore, hyper-parameter calibration cannot be performed without joint variance estimation. Reference chretien2014sparse presents efficient methods for solving this joint estimation/calibration problem and present preliminary computational experiments showing practical relevance of the overall approach. The square-root LASSO belloni2011square is another interesting alternative but is sometimes reported to enjoy slightly worse performance in practice.

The usually preferred practical approach to hyper-parameter calibration is Cross Validation arlot2010survey. The Cross-Validation approach is very intuitive and enjoys nice theoretical properties when the number of covariates is smaller than the number of observations. Another drawback of Cross-Validation is the computational burden of re-sampling and computing the LASSO estimator a large number of times. Moreover, Cross-Validation is oriented towards prediction performance rather than accurate support selection. An alternative approach devised in chretien2018hedging, based on the Hedge algorithm of freund1997decision and the stochastic Frank-Wolfe algorithm, was shown to outperform Cross Validation in terms of computational time for the linear model as well.

Recently, giacobino2015quantile devised a very efficient method called Quantile Universal Thresholding for hyper-parameter calibration in the linear model with a view towards efficient variable selection. Extensive numerical experiments provided in giacobino2015quantile show that Quantile Universal Thresholding outperforms Cross-Validation, although Cross-Validation has to be performed when the noise variance is unknown. Fortunately enough, recent work on fast variance estimation, as described e.g. in kennedy2018greedy or based on chretien2018hedging, should however allow to overcome the burden of using Cross-Validation as a subroutine in the Quantile Universal Thresholding procedure of giacobino2015quantile.

4.1.4/ CONTRIBUTIONS OF THE PAPER

The main contributions of the present paper are threefold. The first is to present a ℓ_1-penalised maximum likelihood estimator for the ordered polytomous model and present efficient methods for computing this estimator. The second contribution is an efficient hyper-parameter calibration procedure based on recent work `giacobino2015quantile`. The last contribution is a freely available software implementation which can be downloaded online `lecode`.

4.2/ MATERIEL AND METHOD

4.2.1/ THE MODEL AND THE PENALISED ESTIMATOR

4.2.1.1/ THE STANDARD POLYTOMOUS REGRESSION MODEL

In the ordinal polytomous regression model, the independent qualitative output variables Y_i, $i = 1, \ldots, n$ with Q modalities $m_1, \ldots, m_Q$, are assumed to result from the quantification of a latent continuous variable $Y_i^* = X_i^t\beta^0 + \epsilon_i$, $i = 1, \ldots, n$, where X_i is a p-dimensional vector of covariates and where the residual ϵ_i has logistic cumulative distribution function $\Phi(y) = \frac{\exp(y)}{1+\exp(y)}$. More precisely, setting $-\infty = \gamma_0^0 < \cdots < \gamma_{Q-1}^0 < \gamma_Q^0 = +\infty$, we have $Y_i = m_q$ if and only if $Y_i^* \in]\gamma_{q-1}, \gamma_q]$. For $q = 1, \ldots, Q$, let us denote I_q the subset of $\{1...n\}$ such that $i \in I_q$ if and only if $Y_i = m_q$. Let us denote by γ the vector $\gamma = (\gamma_1, \ldots, \gamma_{q-1})$. The conditional likelihood given $X_1, \ldots, X_n$ for this model is :

$$L_{Y|X}(\beta, \gamma) = \prod_{q=1}^{Q} \prod_{i \in I_q} \left(\Phi\left(X_i^t\beta - \gamma_{q-1}\right) - \Phi\left(X_i^t\beta - \gamma_q\right) \right). \tag{4.1}$$

where X is the $n \times p$ matrix such that X_i is its i^{th} row for all i in 1, ..., n. The conditional log-likelihood is given by

$$l_{Y|X}(\beta, \gamma) = \sum_{i=1}^{n} \sum_{q=1}^{Q} 1_{\{Y_i = m_q\}} \log\left(\Phi\left(X_i^t\beta - \gamma_{q-1}\right) - \Phi\left(X_i^t\beta - \gamma_q\right) \right),$$

or in other words,

$$l_{Y|X}(\beta, \gamma) = \sum_{q=1}^{Q} \sum_{i \in I_q} \log\left(\Phi\left(X_i^t\beta - \gamma_{q-1}\right) - \Phi\left(X_i^t\beta - \gamma_q\right) \right).$$

The parameters of this model are usually estimated using the maximum likelihood principle, i.e., by finding the vector $(\hat{\beta}, \hat{\gamma})$ that maximizes $l_{Y|X}$. Maximization of the log-likelihood is made easy by the well known fact that the conditional log-likelihood function is concave.

The problem with this approach is that it cannot works when p is larger than n because, in this case, the Hessian matrix is easily shown to be singular. The situation where p is larger than n is however frequent in gene expression analysis as in many other problems,

and one needs an estimator which can perform variable selection in such settings with low computational complexity. The next section introduces such an estimator based on ℓ_1 penalisation.

4.2.1.2/ THE PENALISED MAXIMUM LIKELIHOOD ESTIMATOR

One estimator of choice for the type of problem we just described is the ℓ_1-penalised maximum likelihood estimator given by

$$(\hat{\beta}, \hat{\gamma}) \quad \in \quad \text{argmax}_{(b,c)\in\mathbb{R}^p\times\mathbb{R}^{Q-1}} \; l_{Y|X}(b,c) - \lambda \, \|b\|_1, \tag{4.2}$$

where λ is a relaxation parameter. This estimator corresponds exactly to the LASSO in the case where the log-likelihood is the one of the linear model. The main motivation for introducing this estimator is Theorem 1.2 in candes2009near about the LASSO. This theorem states that for a sufficiently sparse β in the linear model $Y = X\beta + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$, the risk of the LASSO estimator is near optimal, i.e. is comparable to the risk obtained with an oracle estimator which would know the support of β ahead of time. Moreover, support recovery is proved to hold with large probability for a vast majority of possible supports.

The assumptions in this theorem are the following :

1. X has low coherence, i.e. the maximum scalar product of two columns of X is less than $A_0/\log(p)$;
2. the support and sign pattern of β have uniform distribution ;
3. the nonzero components of β have magnitude above the noise level times a log factor.

It is therefore natural to expect that an appropriate translation of this result to the case of (ordinal or not) polytomous regression model will hold as well. In the sequel, we will present simulation based results on the penalised conditional likelihood estimator from the view point of variable selection.

4.2.2/ ALGORITHMS

4.2.2.1/ NESTEROV'S ALGORITHM

In Nesterov:Doklady83,Nesterov:MathProg05,becker2011nesta, Nesterov introduced a new approach to convex minimization with possibly non-differentiable functions. Nesterov's method consists of smoothing the non-differentiable function and then apply a refined first order scheme to the problem. The main interest of this approach is that at iteration k, a bound of $O(1/k^2)$ on the error is garanteed, whereas standard gradient methods only garantee $O(1/k)$. Let us now describe a simple version of this method.

The first step is to smooth the ℓ_1-norm function. Notice that, for the vector β, $\|\beta\|_1$ can be written

$$\|\beta\|_1 \quad = \quad \max_{\|u\|_\infty \leq 1} u^t \beta, \tag{4.3}$$

and the maximizer in this expression is simply $\text{sign}(\beta)$. A possible simple smoothing of the ℓ_1-norm is given by

$$\ell_{1,\mu}(\beta) = \max_{\|u\|_\infty \leq 1} u^t\beta - \frac{\mu}{2}\|u\|_2^2. \tag{4.4}$$

Notice that the maximizer u_β^* in (4.4) exists due to continuity and coercivity, and is unique due to the strict convexity of $\|\cdot\|_2^2$. The main interesting feature of this smoothing is the following proposition.

Proposition 4.2.1. *The function $\ell_{1,\mu}$ is differentiable with Lipschitz gradient. Moreover, the gradient is given by*

$$\nabla \ell_{1,\mu} = u_\beta^* \tag{4.5}$$

where u_β^* is the unique maximizer in (4.4) and the Lipschitz constant of the gradient is $L_1 = 1/\mu$.

Proof. See `Nesterov:MathProg05`, Theorem 1. □

With this result in hand, we can present Nesterov's accelerated gradient algorithm for smooth optimisation in Algorithm 1 below. In order to implement the algorithm, one needs to know the Lipschitz constant of the gradient of minus the log-likelihood, which is unknown, and the Lipschitz constant of the smoothed ℓ_1-norm penalty, which is $1/\mu$. In practice, the Lipschitz constant of the gradient of minus the log-likelihood can be estimated by random sampling and computing ratio between the norm of the difference between gradients at sampled points and the norm of the difference of these sample points.

4.2.2.2/ THE FRANK-WOLFE ALGORITHM

The main trick that is needed to implement the Frank-Wolfe algorithm is to reformulate the penalised problem

$$(\hat{\beta}, \hat{\gamma}) \in \text{argmax}_{(b,c)\in\mathbb{R}^p\times\mathbb{R}^{Q-1}} \; l_{Y|X}(b,c) - \lambda \, \|b\|_1. \tag{4.6}$$

as a constrained optimisation problem

$$(\hat{\beta}, \hat{\gamma}) \in \text{argmax}_{(b,c)\in\mathbb{R}^p\times\mathbb{R}^{Q-1}} \; l_{Y|X}(b,c) \text{ with } \|b\|_1 \leq r \tag{4.7}$$

for an appropriate value of r. In this new formulation, the problem of choosing λ is translated into the problem of choosing r.

The second formulation will be used in Section 4.2.2.2, in which the most difficult problem is choose this r value.

The Frank–Wolfe (FW) algorithm is a constrained convex optimisation method proposed by Marguerite Frank and Philip Wolfe in 1956 `frank1956algorithm`. Each iteration of the FW algorithm consists of finding s^k by minimizing $\mathbf{s}^T\nabla f(\mathbf{x}_k)$ subject to $\mathbf{s} \in \mathcal{D}$. We then upgrade $\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{2}{k+2}(\mathbf{s}_k - \mathbf{x}_k)$, where f is the function to minimize, k is the current iteration, and $\mathcal{D}$ is the set on which we want to optimize f. In the case where $\mathcal{D}$ is the hypercube defined by $\|\beta\|_1 \leq r$ (as in our case), determining s_k is simple, since it is the point :

— of coordinate r for the component such that $\nabla_\beta l_{\tilde{Y}|X}(\beta,\gamma)$ is minimal,
— and zero for all the other components.

However, the logistic regression is a special case in which constraints have to be put on β, but not on γ. Practically speaking, the choice has been to alternate iterations of the Frank-Wolfe algorithm (to optimize β with γ fixed) with a simple gradient descent (to optimize γ with β fixed).

4.2.3/ HYPERPARAMETER CALIBRATION

4.2.3.1/ SELECTION OF THE PARAMETER BY AIC

The first implemented method to select the λ parameter is to use the Akaike information criterion (AIC) `akaike1998information`. This AIC is a compromise between the likelihood of the model and the number of non-zero parameters. More precisely, $AIC = -2l_{Y|X}(\beta,\gamma) + 2\|\beta\|_0$, and the goal is to find a set of parameters that minimizes this value. The method of choosing lambda processes in three steps. In the first one, the objective to is determine a penalty $\lambda_\#$ that is large enough to cancel all β components. This objective is realized by using Algorithm 2.

One can then apply, e.g. Nesterov's or the stochastic Frank-Wolfe algorithm with different values of the hyperparameter. One possible set of values is $\lambda_0 = 0$, $\lambda_1 = \frac{\lambda_\#}{50}$, $\lambda_2 = \frac{2\times\lambda_\#}{50}$, ..., $\lambda_{49} = \frac{49\times\lambda_\#}{50}$. The AIC value is then computed for each obtained model and, at the end of the day, the model with smallest AIC is finally selected.

4.2.3.2/ BIC SELECTION

λ is chosen in the same manner than for the AIC method, except for the fact that the value to optimize is, this time, $BIC = -2\, l_{Y|X}(\beta,\gamma) + log(n)\|\beta\|_0$.

4.2.3.3/ ADAPTING THE QUANTILE UNIVERSAL THRESHOLD SELECTION TO ORDINAL POLYTOMOUS REGRESSION

Quantile Universal Threshold (QUT) `giacobino2015quantile` is a simulation-based method. Its objective is to be sure that, if the vector Y to be predicted has no link with the matrix of predictive variables X, then the vector β of the regression coefficients will be the null vector with probability $1-\alpha$, where α is set by the user (α is set to 5% in Section 4.3).

The working principle of QUT is as follows.

— Randomly pick a large number of vectors of the same size than Y. For instance, in the case study of Section 4.3, 100 vectors $\tilde{Y}_1 ... \tilde{Y}_{100}$ are picked as permutations of the original Y vector. That is to say, $\tilde{Y}_i$ has the same number of subjects in each category as the initial vector Y.
— For each random vector $\tilde{Y}_i$, find a λ_i large enough such that, when the ℓ_1-penalised maximum likelihood estimator described in Section 4.2.1.2 is optimized, β is the null vector.
— The obtained λ_i are sorted, and then we select the value such that a proportion $1-\alpha$ of the λ_i is below this threshold.

To speed up the second step of this process, the following property is used. Let us denote by $\lambda_\# = \|\nabla_\beta l_{\tilde{Y}|X}(\beta = \vec{0}, \gamma)\|_\infty$. Thus, an optimisation of the ℓ_1-penalised maximum likelihood estimator with a penalty of $\lambda_\#$ returns $\beta = \vec{0}$.

γ is required in order to compute $\lambda_\#$. However γ is not known, and λ is needed to calculate it. So, a loop has been implemented as in Algorithm 3.

Thanks to the shortcut $\lambda_\# = \|\nabla_\beta l_{\tilde{Y}|X}(\beta = \vec{0}, \gamma)\|_\infty$, the computation time to obtain λ is greatly reduced, leading to the fastest determination of λ (see Section 4.3), as it requires only a few the optimisation of the ℓ_1-penalised maximum likelihood estimator. Note that a version of the QUT whose second step is performed by dichotomy, as in Algorithm 2, has been implemented too, but it underperforms the other methods in terms of computation time.

4.2.3.4/ SELECTION OF THE r PARAMETER BY ONLINE FRANK-WOLFE ALGORITHM

The method follows the procedure described in `chretien2018hedging` with small necessary adjustments in order to accommodate for the specific constraints associated with our estimator. We refer the reader to the associated longer report `chretien2018polytomous` for complete details.

Algorithme 1 : Nesterov's algorithm for penalised log-likelihood estimation

Input An initial point $\theta^{(0)} = (\beta^{(0)}, \gamma^{(0)})$, e.g. $\theta^{(0)} = 0$, the relaxation coefficient λ, the Lipschitz constants L_0 (resp. L_1) of the gradient of - the log-likelihood (resp. of $\ell_{1,\mu}$) and the maximum number of iterations $N \in \mathbb{N}_*$

for $k = 0...N-1$ **do**

Compute $g^{(k)} = \nabla\left(-l(\theta^{(k)}) + \lambda\ell_{1,\mu}(\beta^{(k)})\right)$

Compute $\theta^{(k,1)}$:

$$\theta^{(k,1)} = \text{argmin}_{\tau\in\mathbb{R}^{p+Q-1}}\langle g^{(k)}, \tau - \theta^{(k)}\rangle + \frac{L_0+L_1}{2}\|\tau-\theta^{(k)}\|_2^2.$$

Compute $\theta^{(k,2)}$:

$$\theta^{(k,2)} = \text{argmin}_{\tau\in\mathbb{R}^{p+Q-1}}\Big(\sum_{0\le k'\le k}\frac{1}{2(k'+1)}g^{(k')}, \tau - \theta^{(k')}\rangle\Big) + \frac{L_0+L_1}{2}\|\tau-\theta^{(0)}\|_2^2.$$

Update $\theta^{(k+1)}$:

$$\theta^{(k+1)} = \frac{k+1}{k+3}\,\theta^{(k,1)} + \frac{2}{k+3}\,\theta^{(k,2)}.$$

end for

Output $\widehat{\theta^{(N)}}$.

Algorithme 2 : Find a λ that cancels all β components following a dichotomy approach

V : number of non-zero coefficients of β, as a function of λ.

δ : desired accuracy, set by the user (default value : $\delta = 0.01$).

$\lambda_{max} = 1$

while V(λ_{max}) $\neq 0$ **do**

$\lambda_{max} = \lambda_{max} \times 2$

end while

$\lambda_{min} = \frac{\lambda_{max}}{2}$

if $\lambda_{max} = 1$ **then**

$\lambda_{min=0}$

end if

while $\lambda_{max} - \lambda_{min} \geq \delta$ **do**

$\lambda_{mean} = \frac{\lambda_{max}-\lambda_{min}}{2}$

if $V(\lambda_{mean}) = 0$ **then**

$\lambda_{max} = \lambda_{mean}$

else

$\lambda_{min} = \lambda_{mean}$

end if

end while

Output $\lambda_\# = \lambda_{mean}$.

Algorithme 3 : QUT : successive evaluations of gamma knowing lambda, and of lambda knowing gamma

$\lambda = \sqrt{2} \times log(2 \times \max(p, 1)) \times \max(0.01, std(Y))$ (initialization of λ)
for i = 1 ... 3 **do**
  Choose γ based on the current λ with Nesterov.
  Choose λ based on the current γ with QUT.
end for
Output λ.

4.3/ SIMULATION RESULTS

4.3.1/ DESCRIPTION OF THE EXPERIMENTS

We now assess the practical performance of the proposed methods. For this purpose, we performed various numerical experiments on simulated data. The simulation and testing procedure works as follows.

1. The number of subjects n, the number of variables p, the number of influential variables s, and the underlying cut-off vector γ_0 are set (Section 4.3.2 contains the authors' choices).
2. The vector of underlying parameters β_0 is randomly picked. This vector is of size p such that $p - s$ of its components are null, while the other s components follow a Gaussian law $\mathcal{N}(0, 1)$
3. The matrix X of explanatory variables is then drawn. This is a matrix of size $n \times p$, in which each component follows a law $\mathcal{N}(0, 1)$.
4. The noise vector ϵ of Y^* is drawn. It is of size n, where each component follows a logistic(1,1) law.
5. $Y^* = X\beta_0 + \epsilon$ is computed, and then Y based on Y^* and γ_0.
6. Steps 2, 3, 4, and 5 above allows the construction of a database. They are repeated 50 times, leading to 50 different databases.
7. Each of these 50 databases is divided into a learning sample ($\frac{2}{3}$ of the subjects) and a testing one (the other third).
8. Each of the regression methods listed in Section 4.3.2 is finally applied to the 50 learning samples. The performances of the models are measured on the 50 corresponding test samples based on the criteria defined in Section 4.3.2.

4.3.2/ COMPARISON EXPERIMENTS

The methods we decided to compare are the following.

— λ parameter selection by AIC as in Section 4.2.3.1.
— λ parameter selection by BIC as in Section 4.2.3.2.
— λ parameter selection according to Quantile Universal Threshold, as presented in Section 4.2.3.3.
— The use of the Frank-Wolfe algorithm, to solve the constrained optimization with selection of the r parameter using Online Frank-Wolfe, as defined in Sections 4.2.2.2 and 4.2.3.4.
— The absence of variable selection. That is to say, the model obtained when the likelihood is maximized without penalty. This model is simply named "$\lambda = 0$" in Tables 4.1 and 4.2.
— The model that predicts, for each subject in the test sample, the largest category of the learning sample. It is named "null model" in Tables 4.1 and 4.2, as this is the best possibility if no explanatory variable is taken into account.

$\lambda = 0$ and the null model are only performed to check if the first four methods work well. Indeed, when dealing with the logistic regression, it is important to check if the predictive

model is better than simply placing all patients in the majority category. Moreover, when working on variables selection, it can be useful to check if the obtained model is better than the one with no selection.

Two experiments have been performed. In the first one, $n > p$, there are 50 variables, the learning sample has been constituted by 200 subjects, while the test sample has 100 subjects (see Table 4.1). In the other experiment, $p > n$, the learning sample has 100 subjects, the test one has 50 subjects, and there are 200 variables (Table 4.2). In both cases, the number of significant variables was set to $s = 5$.

We considered $Q = 3$ categories for Y, and we set $\gamma_0 \in [0, 3]$, as unbalanced categories were wanted to complicate the regression problem. With this choice of γ_0, Y_i is in the first category for all $Y_i^* \leq 0$, *i.e.*, for half of the simulated subjects. The Nesterov algorithm runs for 200 iterations, while the Frank-Wolfe one iterates 200 times.

For each method, four performance criteria are studied.

— The percentage of subjects in the test sample which are well ranked by the model fitted on the learning sample. This percentage is named “well classified” in Tables 4.1 and 4.2.

— The average likelihood. That is, the geometric mean of the probabilities that the model fitted to the learning sample assigns the actual categories of subjects in the test sample. This is what we called “average likelihood” in Tables 4.1 and 4.2.

— The average prediction error. That is to say, the average gap between the predicted category and the actual category, named “prediction error” in Tables 4.1 and 4.2.

— The percentage of well-ranked subjects, weighted by the size of the categories. More precisely, we calculate $100 \times \sum_{i=1}^{n} 1_{\text{prediction is right}} \frac{Q \times p}{\#I_{Y_i}}$, where $\#I_{Y_i}$ is the number of subjects in the same category than Y_i. This criterion attaches greater importance to the proper classification of subjects that are in a poorly represented category. It is referenced as “well-classified weighted” in Tables 4.1 and 4.2.

The “average likelihood” and “well-classified weighted” criteria are relevant when classes are very unbalanced (like 98 %, 1 %, and 1 %), which can really occur in practice. In the case study, the “well classified” criterion has been considered first, as this is probably the most natural criterion for not too unbalanced categories like the ones used during our simulations. Tables 4.1 and 4.2 are sorted according to this criterion.

Table 4.1 summarizes the results in the case where the number of subjects in the training sample is 200, the number of subjects in the testing one is 100, and the number of explanatory variables is 50. Table 4.2, for its part, summarizes the results in the case where the number of subjects in the training sample is 100, the number of subjects in the testing one is 50, and the number of explanatory variables is 200.

Wilcoxon tests have also been performed in order to determine if the differences between the methods are statistically significant. Tables 4.3 and 4.4 show the results of these Wilcoxon tests. In the $n_{\text{learning}} = 200, p = 50$ case, the difference between well-classified subjects for BIC (66,7%) and QUT (65,4%) is significant with a p-value of $8,03 \times 10^{-3}$, even if this difference is only equal to 1,3%. Conversely, in the $n_{\text{learning}} = 100, p = 200$ case, the difference between QUT, BIC, OFW is not significant. This case may require more simulated data if we want to separate these methods correctly. Finally, in any cases, QUT, BIC, OFW, and AIC are significantly better than λ =0 and the null model.

TABLE 4.1 – Monte Carlo simulations with $n_{\text{learning}} = 200, p = 50, n_{\text{test}} = 100$

choice of λ (or r)	well classified	average likelihood	prediction error	well classified weighted	Time	λ (or r)	Nb of variables
BIC	66,7	0,48	0,35	59,4	2464,9	14,8	3,9
QUT $\|\|\|\|_\infty$	65,4	0,48	0,36	57,9	53,0	22,0	2,6
AIC	65,3	0,47	0,36	58,6	2458,9	10,2	7,1
OFW	63,3	0,46	0,39	55,2	199,0	7,2	39,6
$\lambda = 0$	60,0	0,44	0,43	55,3	20,1	0,0	50,0
null model	49,0	-	-	-	0	-	0

TABLE 4.2 – Monte Carlo simulations with $n_{\text{learning}} = 100, p = 200, n_{\text{test}} = 50$

choice of λ (or r)	well classified	average likelihood	prediction error	well classified weighted	Time	λ (or r)	Nb of variables
QUT $\|\|\|\|_\infty$	61,0	0,43	0,42	52,4	33,5	16,9	1,3
BIC	60,7	0,44	0,42	54,0	982,9	11,9	3,5
OFW	59,8	0,43	0,42	50,2	72,4	6,4	54,1
AIC	55,4	0,42	0,48	50,1	995,8	8,9	9,2
null model	48,1	—	—	—	0	—	0
$\lambda = 0$	36,7	0,22	0,77	40,0	12,8	0,0	200,0

TABLE 4.3 – Paired Wilcoxon tests associated to Monte Carlo simulations with $n_{\text{learning}} = 200, p = 50, n_{\text{test}} = 100$

	BIC	QUT$\|\|\|\|_\infty$	AIC	OFW	$\lambda = 0$	null model
BIC	-	$8,03 \times 10^{-03}$	$3,69 \times 10^{-03}$	$7,83 \times 10^{-07}$	$1,71 \times 10^{-09}$	7.38×10^{-10}
QUT$\|\|\|\|_\infty$	$8,03 \times 10^{-03}$	-	$7,03 \times 10^{-01}$	$3,36 \times 10^{-03}$	$9,54 \times 10^{-08}$	7.83×10^{-10}
AIC	$3,69 \times 10^{-03}$	$7,03 \times 10^{-01}$	-	$8,99 \times 10^{-04}$	$2,02 \times 10^{-08}$	7.32×10^{-10}
OFW	$7,83 \times 10^{-07}$	$3,36 \times 10^{-03}$	$8,99 \times 10^{-04}$	-	$1,58 \times 10^{-06}$	7.44×10^{-10}
$\lambda = 0$	$1,71 \times 10^{-09}$	$9,54 \times 10^{-08}$	$2,02 \times 10^{-08}$	$1,58 \times 10^{-06}$	-	1.95×10^{-09}
null model	7.38×10^{-10}	7.83×10^{-10}	7.32×10^{-10}	7.44×10^{-10}	1.95×10^{-09}	-

TABLE 4.4 – Paired Wilcoxon tests associated to Monte Carlo simulations with $n_{\text{learning}} = 100, p = 200, n_{\text{test}} = 50$

	QUT	BIC	OFW	AIC	null model	$\lambda = 0$
QUT	-	6.74×10^{-01}	3.77×10^{-01}	2.93×10^{-04}	$8,66 \times 10^{-}09$	9.13×10^{-10}
BIC	6.74×10^{-01}	-	4.86×10^{-01}	5.47×10^{-04}	$1,62 \times 10^{-}08$	1.32×10^{-09}
OFW	3.77×10^{-01}	4.86×10^{-01}	-	2.66×10^{-05}	$1,17 \times 10^{-}08$	1.31×10^{-09}
AIC	2.93×10^{-04}	5.47×10^{-04}	2.66×10^{-05}	-	$1,87 \times 10^{-}05$	3.33×10^{-09}
null model	$8,66 \times 10^{-}09$	$1,62 \times 10^{-}08$	$1,17 \times 10^{-}08$	$1,87 \times 10^{-}05$	-	$6,95 \times 10^{-}07$
$\lambda = 0$	9.13×10^{-10}	1.32×10^{-09}	1.31×10^{-09}	3.33×10^{-09}	$6,95 \times 10^{-07}$	-

4.4/ DISCUSSION

First of all, the four variable selection methods work better than $\lambda = 0$ and the null model. This shows that the algorithms work correctly, and that variable selection is useful. The absence of variable selection is particularly harmful in the case where $p > n$, see Table 4.2. It makes sense because, in this case, the optimisation of the unpenalised likelihood allows an infinite number of solutions. This $p > n$ case is very common in practice.

In the experiment shown in Table 4.1, the BIC works a bit better than the other methods, while in the experiment summarized in Table 4.2, QUT, BIC, and OFW are very close. In terms of computation time, QUT is the most interesting approach. Indeed, as explained in Section 4.2.3.3, this method allows to choose λ by executing the regression only a few times.

4.5/ CONCLUSION

The present paper proposed a new estimator for sparse ordinal polytomous regression in a high dimensional setting together with a strategy for hyper-parameter calibration based on previous results from `giacobino2015quantile`. Performance of the method was assessed via extensive numerical experiments. The forthcoming report `chretien2018polytomous` will include further implementation details, and improvements, and additional numerical results on large real datasets.

FUNDING

Computations have been performed on the supercomputer facilities of the Mésocentre de calcul de Franche-Comté.

4.6/ COMPLÉMENTS

À la suite de la publication ci-dessus, nous avons, à la demande des relecteurs, effectué des essais supplémentaires. En l'occurrence, nous avons d'une part calculé l'efficacité de la validation croisée comme méthode de choix de la pénalisation, et d'autre part relevé pour chaque méthode sa précision dans la reconnaissance des variables influentes à travers son taux de vrai positif et son taux de faux positif.

La colonne "TPR" (true positive rate) des tables 4.5 et 4.6 désigne le pourcentage de variables qui ont été reconnues comme influentes sur l'output parmi celles qui sont effectivement programmées pour influer sur l'output. La colonne "TNR" (true négative rate) désigne le pourcentage de variables qui ont été reconnues comme non influentes sur l'output parmi celles qui sont effectivement programmées pour ne pas influer sur l'output.

La ligne "CV" désigne les résultats pour la cross validation. Il s'agit ici d'une procédure de cross validation à l'intérieur des échantillons d'apprentissage. C'est-à-dire que l'on crée des échantillons d'apprentissage à l'intérieur des échantillons d'apprentissage. Il s'agit d'un découpage 9/10 (apprentissage) vs 1/10 (test). Par exemple pour le cas $n_{\text{learning}} =$

200, $p = 50$, $n_{\text{test}} = 100$, on découpe n_{learning} en 10 sous échantillons de 20 sujets, pour chaque valeur de lambda envisagée on utilise 180 sujets pour apprendre et 20 pour tester dans les 10 configurations possibles. Puis, une fois qu'on a choisi l'hyperparamètre λ, on utilise les 100 sujets de n_{test} pour comparer la cross validation aux autres méthodes.

Concernant le TPR et TNR, comme on pouvait l'imaginer, plus la méthode est sélective (pénalisation forte, nombre de variables retenues faibles), plus la TPR est fort et plus le TNR est faible. Comme toutes ces méthodes ne varient que par le choix de l'hyperparamètre de pénalisation, aucune ne maximise TPR et TNR en même temps. Si on veut maximiser TPR + TNR par exemple, les meilleures méthodes seraient alors OFW dans le cas $p > n$ et AIC dans le cas $n > p$.

Les performances de la validation croisée sont correctes. Il s'agit de la 4^{eme} meilleure méthode pour taux de patients bien classés pour $p > n$ et de la 2^{eme} meilleure pour $n >$. En revanche, c'est de loin la méthode la plus longue à exécuter.

TABLE 4.5 – Monte Carlo simulations with $n_{\text{learning}} = 100, p = 200, n_{\text{test}} = 50$

choice of λ (or r)	correctly ranked	average likelihood	prediction error	CRW	Time	λ (or r)	Nb of variables	TPR	TNR
QUT $\|\|\|\|_\infty$	61.0	0.43	0.42	52.4	33.5	16.9	1.3	23.2	100
BIC	60.7	0.44	0.42	54.0	982.9	11.9	3.5	37.2	99.8
OFW	59.8	0.43	0.42	50.2	72.4	6.4	54.1	73.2	83.7
CV	57.0	0.41	0.47	47.3	7794.3	11.3	4.0	42,7	98,6
AIC	55.4	0.42	0.48	50.1	995.8	8.9	9.2	52.4	97.4
null model	48.1	-	-	33.3	0	-	0	-	-
$\lambda = 0$	36.7	0.22	0.77	40.0	12.8	0.0	200.0	-	-

TABLE 4.6 – Monte Carlo simulations with $n_{\text{learning}} = 200, p = 50, n_{\text{test}} = 100$

choice of λ (or r)	correctly ranked	average likelihood	prediction error	CRW	Time	λ (or r)	Nb of variables	TPR	TNR
BIC	66.7	0.48	0.35	59.4	2464.9	14.8	3.9	58.8	99.6
CV	65.8	0.48	0.36	58.9	17662.1	13.4	5.3	66.8	95.6
QUT $\|\|\|\|_\infty$	65.4	0.48	0.36	57.9	53.0	22.0	2.6	48.4	99.96
AIC	65.3	0.47	0.36	58.6	2458.9	10.2	7.1	74.0	94.8
OFW	63.3	0.46	0.39	55.2	199.0	7.2	39.6	93.2	45.5
$\lambda = 0$	60.0	0.44	0.43	55.3	20.1	0.0	50.0	-	-
null model	49.0	-	-	33.3	0	-	0	-	-

Par ailleurs, durant ces travaux, nous avons également effectué des tests sur une base de donnée concernant le cancer de la vessie. Cette base contenait 34 gènes pour 78 patients. La tumeur présentait 3 niveaux de gravité. Le "défaut" de cette base de données, qui la rend peu adéquate pour notre sujet est que les gènes présents sont des gènes déjà sélectionnés pour leur lien statistique avec ce cancer. Dès lors, si l'on découpe cette base en échantillons d'apprentissages et de tests, la meilleure sélection de variables possible sera de tout garder. C'est ce que l'on voit sur le tableau 4.7 dans lequel la méthode sans sélection de variables ($\lambda = 0$) apparaît comme la meilleure.

Récemment nous nous sommes vu confier une base de données de grande taille (54675 variables, 1656 patients, 4 niveaux de gravité de tumeur). Cette base de données

TABLE 4.7 – Résultats vessie

Choix λ ou r	Bien Classés	Vraisemb moyenne	Erreur de prédiction	Bien classé pondéré	Temps	λ (ou r)	Nb de variables
$\lambda = 0$	63,5	0,45	0,38	64,1	6,4	0,0	34,0
Quantile $\|\|\|\|_\infty$	63,3	0,44	0,39	60,9	16,6	10,3	6,9
OFW	62,8	0,43	0,38	56,9	41,1	6,0	18,6
AIC	61,9	0,43	0,41	59,2	478,4	13,6	3,7
BIC	60,8	0,43	0,42	58,1	474,7	14,5	2,9
Modèle nul	49.2	-	-	-	0	-	0

concerne le cancer du sein, et nous a été confiée par M. Jean Paul Feugas en vu d'une collaboration. Les premiers essais sur cette base font apparaître des bugs, peut-être dus à la taille des données. Je n'ai pas eu le temps de m'y pencher davantage mais cela pourrait être une perspective de continuation possible de ces travaux.

III

CONCLUSION

CONCLUSION

Durant ces trois années de doctorat, nous avons pu aborder différents aspects de la programmation bio-informatique.

Nos travaux les plus aboutis sont probablement ceux qui se situent dans le domaine du clustering de séquences. En effet, nous avons pu dans ce projet effectuer des tests à la fois sur données réelles et sur données simulées. Les tests sur données réelles ont montré une cohérence du clustering obtenu avec l'arbre phylogénétique construit via PhyML ainsi qu'avec la taxonomie proposée par le NCBI. Les résultats sur données simulées ont pu montrer une bonne capacité à retrouver le clustering attendu, cette capacité excédant même celle des outils classiques de clustering de séquences. Ainsi, un prolongement possible de ces travaux pourrait être de rendre l'outil plus facilement accessible en en faisant un package python via pypi par exemple, ou en en faisant un outil utilisable en ligne. Par ailleurs il est aussi envisageable de chercher par de nouveaux tests à optimiser différents paramètres de notre programme, tels la dimension de l'espace dans lequel les Laplacian eigenmaps plongent les données, le nombre de clusters ou le choix de la matrice de similarité.

Concernant nos travaux sur les éléments transposables au sein du génome, nous avons pu fournir une première proposition de modèle de cette propagation et une méthode pour estimer les paramètres de ce modèle. Toutefois, cette méthode par simulation proposée présente ses imperfections, notamment au niveau du temps de calcul qui croit exponentiellement avec le nombre de paramètres du modèle et quadratiquement avec le nombre d'éléments transposables. Pour des travaux futurs, il faudrait sans doute chercher à estimer ces paramètres d'une façon plus mathématique (maximum de vraisemblance, estimation Bayesienne...).

Nos travaux d'analyse de données métagénomiques via courbes ROC ont permis d'extraire de nouveaux résultats de ces données. En particulier, ils ont permis de mettre en évidence des OTUs très discriminantes du site (donc très affectées par la pollution), même lorsque ces OTUs sont présentes en faibles quantité. Les courbes ROC ne sont pas un outil statistique nouveau, et chacun des résultats présenté dans les tableaux présentés dans notre contribution aurait pu être obtenu individuellement avec un outil existant. L'intérêt de notre contribution ici était de rendre l'obtention de tous ces résultats en même temps, triés correctement, de la façon la plus simple possible pour les chercheurs en écologie. Ici, la question est de savoir comment cette proposition sera reçue par cette communauté de chercheurs. Le cas échéant, si la demande existe, une amélioration pourrait être ici aussi de proposer un outil en ligne pour que cet outil soit plus accessible.

Finalement, nos travaux sur la régression polytomique ordonnée pénalisée par norme ℓ_1 ont permis de construire un outil pour résoudre cette régression ainsi qu'une variété de stratégies (AIC, BIC, OFW, QUT) pour choisir la pénalisation adéquate. Les tests effectués sur données simulées ont montré que la plupart de ces stratégies sont raisonnables du point de vue de la précison mais le Quantile Universal Threshold en particulier

se démarque du point de vue de la rapidité de calcul. Une poursuite de ces travaux pourrait être d'effectuer des tests sur des données réelles pertinentes.

BIBLIOGRAPHIE

— mcclintock1950or2 : .....
Barbara McClintock.
The origin and behavior of mutable loci in maize.
Proceedings of the National Academy of Sciences, 36(6) :344–355, 1950.

— finnegan1989eukaryotic : .....
David J Finnegan.
Eukaryotic transposable elements and genome evolution.
Trends in genetics, 5 :103–107, 1989.

— rabbani2015evolution : .....
Mahnaz Rabbani.
Evolution of mobile promoters in prokaryotic genomes.
2015.

— belancio2008mammalian : .....
Victoria P Belancio, Dale J Hedges, and Prescott Deininger.
Mammalian non-ltr retrotransposons : for better or worse, in sickness and in health.
Genome research, 18(3) :343–358, 2008.

— green1988mobile : .....
Melvin M Green.
Mobile dna elements and spontaneous gene mutation.
Banbury Rep, 30 :41–50, 1988.

— wicker2007unified : .....
Thomas Wicker, François Sabot, Aurélie Hua-Van, Jeffrey L Bennetzen, Pierre Capy, Boulos Chalhoub, Andrew Flavell, Philippe Leroy, Michele Morgante, Olivier Panaud, et al.
A unified classification system for eukaryotic transposable elements.
Nature Reviews Genetics, 8(12) :973–982, 2007.

— flybase : .....
flybase.
http ://flybase.org/.

— repbase : .....
repbase.
http ://www.girinst.org/repbase/.

— density : .....
kernel density estimation in r.
https ://stat.ethz.ch/R-manual/R-devel/library/stats/html/density.html.

— munkres_python : .....
Brian Clapper.
munkres 1.0.7 for python.
https ://pypi.python.org/pypi/munkres/, 2008.

— adams2000genome : .....
Mark D Adams, Susan E Celniker, Robert A Holt, Cheryl A Evans, Jeannine D Gocayne, Peter G Amanatides, Steven E Scherer, Peter W Li, Roger A Hoskins, Richard F Galle, et al.
The genome sequence of drosophila melanogaster.
Science, 287(5461) :2185–2195, 2000.

— bartolome2009widespread : .....
Carolina Bartolomé, Xabier Bello, and Xulio Maside.
Widespread evidence for horizontal transfer of transposable elements across drosophila genomes.
Genome Biol, 10(2) :R22, 2009.

— biemont2003influence : .....
Christian BBiémont and Cristina Vieira.
The influence of transposable elements on genome size.
Journal de la Societe de biologie, 198(4) :413–417, 2003.

— bubeck2011introduction : .....
Sébastien Bubeck.
Introduction to online optimization.
Lecture Notes, 2011.

— cleveland1988locally : .....
William S Cleveland and Susan J Devlin.
Locally weighted regression : an approach to regression analysis by local fitting.
Journal of the American Statistical Association, 83(403) :596–610, 1988.

— de2009evolutionary : .....
Nicole De la Chaux and Andreas Wagner.
Evolutionary dynamics of the ltr retrotransposons roo and rooa inferred from twelve complete drosophila genomes.
BMC evolutionary biology, 9(1) :205, 2009.

— egner1981excision : .....
Carol Egner and Douglas E Berg.
Excision of transposon tn5 is dependent on the inverted repeats but not on the transposase function of tn5.
Proceedings of the National Academy of Sciences, 78(1) :459–463, 1981.

— epanechnikov1969non : .....
Vassiliy A Epanechnikov.
Non-parametric estimation of a multivariate probability density.
Theory of Probability & Its Applications, 14(1) :153–158, 1969.

— lander2001initial : .....
Eric S Lander, Lauren M Linton, Bruce Birren, Chad Nusbaum, Michael C Zody, Jennifer Baldwin, Keri Devon, Ken Dewar, Michael Doyle, William FitzHugh, et al.
Initial sequencing and analysis of the human genome.
Nature, 409(6822) :860–921, 2001.

— foster1981three : .....
Timothy J Foster, Victoria Lundblad, Susan Hanley-Way, Shirley M Halling, and Nancy Kleckner.
Three tn10-associated excision events : relationship to transposition and role of direct and inverted repeats.
Cell, 23(1) :215–227, 1981.

— hartl1988unrelated : .....
D L Hartl and S A Sawyer.
Why do unrelated insertion sequences occur together in the genome of escherichia coli ?
Genetics, 118(3) :537–541, 1988.

— drakos2015extinction : .....
Nicole E Drakos and Lindi M Wahl.
Extinction probabilities and stationary distributions of mobile genetic elements in prokaryotes : The birth–death-diversification model.
Theoretical population biology, 106 :22–31, 2015.

— kaminker2002transposable : .....
Joshua S Kaminker, Casey M Bergman, Brent Kronmiller, Joseph Carlson, Robert Svirskas, Sandeep Patel, Erwin Frise, David A Wheeler, Suzanna E Lewis, Gerald M Rubin, et al.
The transposable elements of the drosophila melanogaster euchromatin : a genomics perspective.
Genome Biol, 3(12) :RESEARCH0084, 2002.

— kaplan1985evolution : .....
Norman Kaplan, Tom Darden, and Charles H Langley.
Evolution and extinction of transposable elements in mendelian populations.
Genetics, 109(2) :459–480, 1985.

— moody1988branching : .....
Michael E Moody.
A branching process model for the evolution of transposable elements.
Journal of mathematical biology, 26(3) :347–357, 1988.

— sawyer1987distribution : .....
Stanley A Sawyer, Daniel E Dykhuizen, Robert F DuBose, Louis Green, T Mutangadura-Mhlanga, David F Wolczyk, and Daniel L Hartl.
Distribution and abundance of insertion sequences among natural isolates of escherichia coli.
Genetics, 115(1) :51–63, 1987.

— barbara1987discovery : .....
McClintock Barbara.
The discovery of characterization of transposable elements : the collected papers of Barbara McClintock.
1987.

— sanmiguel1998evidence : .....
Phillip Sanmiguel and Jeffrey L Bennetzen.
Evidence that a recent increase in maize genome size was caused by the massive amplification of intergene retrotransposons.
Annals of Botany, 82(suppl 1) :37–44, 1998.

— needleman1970general : .....
Saul B Needleman and Christian D Wunsch.
A general method applicable to the search for similarities in the amino acid sequence of two proteins.
Journal of molecular biology, 48(3) :443–453, 1970.

— kuhn1955hungarian : .....
Harold W Kuhn.
The hungarian method for the assignment problem.
Naval research logistics quarterly, 2(1-2) :83–97, 1955.

— munkres1957algorithms : .....
James Munkres.
Algorithms for the assignment and transportation problems.
Journal of the Society for Industrial and Applied Mathematics, 5(1) :32–38, 1957.

— lerat2003sequence : .....
Emmanuelle Lerat, Carène Rizzon, and Christian Biémont.
Sequence divergence within transposable element families in the drosophila melanogaster genome.
Genome research, 13(8) :1889–1896, 2003.

— lerat2011comparative : .....
Emmanuelle Lerat, Nelly Burlet, Christian Biémont, and Cristina Vieira.
Comparative analysis of transposable elements in the melanogaster subgroup sequenced genomes.
Gene, 473(2) :100–109, 2011.

— modolo2014new : .....
Laurent Modolo, Franck Picard, and Emmanuelle Lerat.
A new genome-wide method to track horizontally transferred sequences : application to drosophila.
Genome biology and evolution, 6(2) :416–432, 2014.

— smith2007release : .....
Christopher D Smith, ShengQiang Shu, Christopher J Mungall, and Gary H Karpen.
The release 5.1 annotation of drosophila melanogaster heterochromatin.
Science, 316(5831) :1586–1591, 2007.

— morariu08figtree : .....
Vlad I. Morariu, Balaji Vasan Srinivasan, Vikas C. Raykar, Ramani Duraiswami, and Larry S. Davis.
Automatic online tuning for fast gaussian summation.
In *Advances in Neural Information Processing Systems (NIPS)*, 2008.

— emboss : .....
P. Rice, I. Longden, and A. Bleasby.
EMBOSS : The european molecular biology open software suite.
Trends in Genetics, 16(6) :276–277, 2000.

— li2006cd : .....
Weizhong Li and Adam Godzik.
Cd-hit : a fast program for clustering and comparing large sets of protein or nucleotide sequences.
Bioinformatics, 22(13) :1658–1659, 2006.

— edgar2010search : .....
Robert C Edgar.
Search and clustering orders of magnitude faster than BLAST.
Bioinformatics, 26(19) :2460–2461, 2010.

— guindon2005phyml : .....
Stephane Guindon, Franck Lethiec, Patrice Duroux, and Olivier Gascuel.
PHYML online-a web server for fast maximum likelihood-based phylogenetic inference.
Nucleic acids research, 33(suppl 2) :W557–W559, 2005.

— edgar2004muscle : .....
Robert C Edgar.
MUSCLE : multiple sequence alignment with high accuracy and high throughput.
Nucleic acids research, 32(5) :1792–1797, 2004.

— katoh2013mafft : .....
Kazutaka Katoh and Daron M Standley.
MAFFT multiple sequence alignment software version 7 : improvements in performance and usability.
Molecular biology and evolution, 30(4) :772–780, 2013.

— larkin2007clustal : .....
Mark A Larkin, Gordon Blackshields, NP Brown, R Chenna, Paul A McGettigan, Hamish McWilliam, Franck Valentin, Iain M Wallace, Andreas Wilm, Rodrigo Lopez, et al.
Clustal w and clustal x version 2.0.
bioinformatics, 23(21) :2947–2948, 2007.

— belkin2001laplacian : .....
Mikhail Belkin and Partha Niyogi.
Laplacian eigenmaps and spectral techniques for embedding and clustering.
In *NIPS*, volume 14, pages 585–591, 2001.

— chen2007resistance : .....
Haiyan Chen and Fuji Zhang.
Resistance distance and the normalized laplacian spectrum.
Discrete Applied Mathematics, 155(5) :654–661, 2007.

— kopylova2016open : .....
Evguenia Kopylova, Jose A Navas-Molina, Céline Mercier, Zhenjiang Zech Xu, Frédéric Mahé, Yan He, Hong-Wei Zhou, Torbjørn Rognes, J Gregory Caporaso, and Rob Knight.
Open-source sequence clustering methods improve the state of the art.
mSystems, 1(1) :e00003–15, 2016.

— hao2011clustering : .....
Xiaolin Hao, Rui Jiang, and Ting Chen.
Clustering 16s rRNA for OTU prediction : a method of unsupervised bayesian clustering.
Bioinformatics, 27(5) :611–618, 2011.

— rousk2010soil : .....
Johannes Rousk, Erland Bth, Philip C Brookes, Christian L Lauber, Catherine Lozupone, J Gregory Caporaso, Rob Knight, and Noah Fierer.
Soil bacterial and fungal communities across a ph gradient in an arable soil.
The ISME journal, 4(10) :1340–1351, 2010.

— suzek2007uniref : .....
Baris E Suzek, Hongzhan Huang, Peter McGarvey, Raja Mazumder, and Cathy H Wu.
Uniref : comprehensive and non-redundant uniprot reference clusters.
Bioinformatics, 23(10) :1282–1288, 2007.

— day1969estimating : .....
Neil E Day.
Estimating the components of a mixture of normal distributions.
Biometrika, 56(3) :463–474, 1969.

— bilmes1998gentle : .....
Jeff A Bilmes et al.
A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models.
International Computer Science Institute, 4(510) :126, 1998.

— schwarz1978estimating : .....
Gideon Schwarz et al.
Estimating the dimension of a model.
The annals of statistics, 6(2) :461–464, 1978.
— akaike1974new : .....
Hirotugu Akaike.
A new look at the statistical model identification.
IEEE transactions on automatic control, 19(6) :716–723, 1974.
— hillis1992experimental : .....
David M Hillis, James J Bull, et al.
Experimental phylogenetics : Generation of a know phylogeny.
Science, 255(5044) :589, 1992.
— cummings1995sampling : .....
Michael P Cummings, Sarah P Otto, and John Wakeley.
Sampling properties of DNA sequence data in phylogenetic analysis.
Molecular Biology and Evolution, 12(5) :814–822, 1995.
— russo1996efficiencies : .....
CA Russo, Naoko Takezaki, and Masatoshi Nei.
Efficiencies of different genes and different tree-building methods in recovering a known vertebrate phylogeny.
Molecular Biology and Evolution, 13(3) :525–536, 1996.
— zardoya1996phylogenetic : .....
Rafael Zardoya and Axel Meyer.
Phylogenetic performance of mitochondrial protein-coding genes in resolving relationships among vertebrates.
Molecular biology and evolution, 13(7) :933–942, 1996.
— felsenstein1988phylogenies : .....
Joseph Felsenstein.
Phylogenies from molecular sequences : inference and reliability.
Annual review of genetics, 22(1) :521–565, 1988.
— huelsenbeck1995performance : .....
John P Huelsenbeck.
Performance of phylogenetic methods in simulation.
Systematic biology, 44(1) :17–48, 1995.
— nei1996phylogenetic : .....
Masatoshi Nei.
Phylogenetic analysis in molecular evolutionary genetics.
Annual review of genetics, 30(1) :371–403, 1996.
— yang2006computational : .....
Ziheng Yang.
Computational molecular evolution.
Oxford University Press, 2006.
— matiasnotes : .....
Catherine Matias.
Notes de cours : Analyse statistique de graphes.
2015.

— spielman2009spectral : .....
Daniel Spielman.
Spectral graph theory.
Lecture Notes, Yale University, pages 740–0776, 2009.
— chretien2016semi : .....
Stéphane Chrétien, Clément Dombry, and Adrien Faivre.
A semi-definite programming approach to low dimensional embedding for unsupervised clustering.
arXiv preprint arXiv :1606.09190, *, 2016.
— friedman2001elements : .....
Jerome Friedman, Trevor Hastie, and Robert Tibshirani.
The elements of statistical learning, volume 1.
Springer series in statistics Springer, Berlin, 2001.
— biernacki2000assessing : .....
Christophe Biernacki, Gilles Celeux, and Gérard Govaert.
Assessing a mixture model for clustering with the integrated completed likelihood.
IEEE transactions on pattern analysis and machine intelligence, 22(7) :719–725, 2000.
— mclachlan2004finite : .....
Geoffrey McLachlan and David Peel.
Finite mixture models.
John Wiley & Sons, 2004.
— biernacki2003degeneracy : .....
Christophe Biernacki and Stéphane Chrétien.
Degeneracy in the maximum likelihood estimation of univariate gaussian mixtures with EM.
Statistics & probability letters, 61(4) :373–382, 2003.
— wang2014high : .....
Zhaoran Wang, Quanquan Gu, Yang Ning, and Han Liu.
High dimensional expectation-maximization algorithm : Statistical optimization and asymptotic normality.
arXiv preprint arXiv :1412.8729, 2014.
— yi2015regularized : .....
Xinyang Yi and Constantine Caramanis.
Regularized EM algorithms : A unified framework and statistical guarantees.
In *Advances in Neural Information Processing Systems*, pages 1567–1575, 2015.
— vgrm+15:ij : .....
Benoît Valot, Christophe Guyeux, Julien Y Rolland, Kamel Mazouzi, Xavier Bertrand, and Didier Hocquet.
What it takes to be a pseudomonas aeruginosa ? the core genome of the opportunistic pathogen updated.
10(5) :e0126468, may 2015.
— acgmsb+14:ij : .....
Bassam AlKindy, Jean-François Couchot, Christophe Guyeux, Arnaud Mouly, Michel Salomon, and Jacques M. Bahi.
Finding the core-genes of chloroplasts.
Journal of Bioscience, Biochemistery, and Bioinformatics, 4(5) :357–364, 2014.

— buitinck2013api : .....
Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, et al.
Api design for machine learning software : experiences from the scikit-learn project.
arXiv preprint arXiv :1309.0238, 2013.

— knight2007pycogent : .....
Rob Knight, Peter Maxwell, Amanda Birmingham, Jason Carnes, J Gregory Caporaso, Brett C Easton, Michael Eaton, Micah Hamady, Helen Lindsay, Zongzhi Liu, et al.
Pycogent : a toolkit for making sense from sequence.
Genome biology, 8(8) :R171, 2007.

— BLASTClust : .....
sing blastclust to make non-redundant sequence sets.
https://www.ncbi.nlm.nih.gov/Web/Newsltr/Spring04/blastlab.html.

— NCBI : .....
ational center for biotechnology information.
https://www.ncbi.nlm.nih.gov/.

— CD-hitFasterThanBLASTClust : .....
d-hit user's guide.
http ://weizhong-lab.ucsd.edu/cd-hit/wiki/doku.php ?id=cd-hit_user_guide.

— torroni1992native : .....
Antonio Torroni, Theodore G Schurr, Chi-Chuan Yang, Emoke J Szathmary, Robert C Williams, Moses S Schanfield, Gary A Troup, William C Knowler, Dale N Lawrence, and Kenneth M Weiss.
Native american mitochondrial dna analysis indicates that the amerind and the nadene populations were founded by two independent migrations.
Genetics, 130(1) :153–162, 1992.

— he2011laplacian : .....
Xiaofei He, Deng Cai, Yuanlong Shao, Hujun Bao, and Jiawei Han.
Laplacian regularized gaussian mixture model for data clustering.
IEEE Transactions on Knowledge and Data Engineering, 23(9) :1406–1418, 2011.

— bellman2013dynamic : .....
Richard Bellman.
Dynamic programming.
Courier Corporation, 2013.

— bellman2015adaptive : .....
Richard E Bellman.
Adaptive control processes : a guided tour.
Princeton university press, 2015.

— bickel2009simultaneous : .....
Peter J Bickel, Ya'acov Ritov, Alexandre B Tsybakov, et al.
Simultaneous analysis of lasso and dantzig selector.
The Annals of Statistics, 37(4) :1705–1732, 2009.

— lecode : .....
ur python module.
https://github.com/SergeMOULIN/l1-penalised-ordinal-polytomous-regression-estimators.
Accessed : 2018-05-11.

— bunea2007sparsity : .....
Florentina Bunea, Alexandre Tsybakov, Marten Wegkamp, et al.
Sparsity oracle inequalities for the lasso.
Electronic Journal of Statistics, 1 :169–194, 2007.

— Nesterov:MathProg05 : .....
Yurii Nesterov.
Smooth minimization of non-smooth functions.
Mathematical programming, 103(1) :127–152, 2005.

— nesterov:Doklady83 : .....
Yurii Nesterov.
A method of solving a convex programming problem with convergence rate o (1/k2).
Soviet Mathematics Doklady, pages 372–376, 1983.

— candes2006modern : .....
Emmanuel J Candes.
Modern statistical estimation via oracle inequalities.
Acta numerica, 15 :257–325, 2006.

— Massart:LNM07 : .....
Pascal Massart.
Concentration inequalities and model selection, volume 6.
Springer, 2007.

— CandesTao:AnnStat07 : .....
Emmanuel Candes, Terence Tao, et al.
The dantzig selector : Statistical estimation when p is much larger than n.
The Annals of Statistics, 35(6) :2313–2351, 2007.

— Bickel:AnnStat09 : .....
Peter J Bickel, Ya'acov Ritov, Alexandre B Tsybakov, et al.
Simultaneous analysis of lasso and dantzig selector.
The Annals of Statistics, 37(4) :1705–1732, 2009.

— Tibshirani:JRSSB96 : .....
Robert Tibshirani.
Regression shrinkage and selection via the lasso.
Journal of the Royal Statistical Society, Series B, 58 :267–288, 1994.

— candes2009near : .....
Emmanuel J Candès, Yaniv Plan, et al.
Near-ideal model selection by $\ell 1$ minimization.
The Annals of Statistics, 37(5A) :2145–2177, 2009.

— candes2007dantzig : .....
Emmanuel Candes, Terence Tao, et al.
The dantzig selector : Statistical estimation when p is much larger than n.
The Annals of Statistics, 35(6) :2313–2351, 2007.

— akaike1998information : .....
Hirotogu Akaike.
Information theory and an extension of the maximum likelihood principle.
In *Selected Papers of Hirotugu Akaike*, pages 199–213. Springer, 1998.
— frank1956algorithm : .....
Marguerite Frank and Philip Wolfe.
An algorithm for quadratic programming.
Naval Research Logistics (NRL), 3(1-2) :95–110, 1956.
— giacobino2015quantile : .....
Caroline Giacobino, Sylvain Sardy, Jairo Diaz-Rodriguez, and Nick Hengartner.
Quantile universal threshold for model selection.
arXiv preprint arXiv :1511.05433, 2015.
— lafond2015online : .....
Jean Lafond, Hoi-To Wai, and Eric Moulines.
On the online frank-wolfe algorithms for convex and non-convex optimizations.
arXiv preprint arXiv :1510.01171, 2015.
— becker2011nesta : .....
Stephen Becker, Jérôme Bobin, and Emmanuel J Candès.
Nesta : A fast and accurate first-order method for sparse recovery.
SIAM Journal on Imaging Sciences, 4(1) :1–39, 2011.
— tibshirani1997lasso : .....
Robert Tibshirani.
The lasso method for variable selection in the cox model.
Statistics in medicine, 16(4) :385–395, 1997.
— chretien2016bregman : .....
Stéphane Chrétien, Christophe Guyeux, Bastien Conesa, Régis Delage-Mouroux, Michèle Jouvenot, Philippe Huetz, and Françoise Descôtes.
A bregman-proximal point algorithm for robust non-negative matrix factorization with possible missing values and outliers-application to gene expression analysis.
BMC bioinformatics, 17(8) :284, 2016.
— jezequel2015gene : .....
Pascal Jézéquel, Zein Sharif, Hamza Lasla, Wilfried Gouraud, Catherine Guérin-Charbonnel, Loïc Campion, Stéphane Chrétien, and Mario Campone.
Gene-expression signature functional annotation of breast cancer tumours in function of age.
BMC medical genomics, 8(1) :80, 2015.
— chretien2015investigating : .....
Stephane Chretien, Christophe Guyeux, Michael Boyer-Guittaut, Regis Delage-Mouroux, and Francoise Descotes.
Investigating gene expression array with outliers and missing data in bladder cancer.
In *Bioinformatics and Biomedicine (BIBM), 2015 IEEE International Conference on*, pages 994–998. IEEE, 2015.
— chretien2015using : .....
Stéphane Chrétien, Christophe Guyeux, Michael Boyer-Guittaut, Régis Delage-Mouroux, and Françoise Descôtes.
Using the lasso for gene selection in bladder cancer data.
arXiv preprint arXiv :1504.05004, 2015.

— miller2002subset : .....
Alan Miller.
Subset selection in regression.
CRC Press, 2002.
— van2008high : .....
Sara A Van de Geer et al.
High-dimensional generalized linear models and the lasso.
The Annals of Statistics, 36(2) :614–645, 2008.
— chretien2014sparse : .....
Stéphane Chrétien and Sébastien Darses.
Sparse recovery with unknown variance : a lasso-type approach.
IEEE Transactions on Information Theory, 60(7) :3970–3988, 2014.
— belloni2011square : .....
Alexandre Belloni, Victor Chernozhukov, and Lie Wang.
Square-root lasso : pivotal recovery of sparse signals via conic programming.
Biometrika, 98(4) :791–806, 2011.
— arlot2010survey : .....
Sylvain Arlot, Alain Celisse, et al.
A survey of cross-validation procedures for model selection.
Statistics surveys, 4 :40–79, 2010.
— kennedy2018greedy : .....
Christopher Kennedy and Rachel Ward.
Greedy variance estimation for the lasso.
arXiv preprint arXiv :1803.10878, 2018.
— chretien2018hedging : .....
Stephane Chretien, Alex Gibberd, and Sandipan Roy.
Hedging hyperparameter selection for basis pursuit.
arXiv preprint arXiv :1805.01870, 2018.
— chretien2018polytomous : .....
Stephane Chretien, Guyeux Christophe, and Serge Moulin.
l1-penalised ordinal polytomous regression estimators.
arXiv preprint to be submitted, 2018.
— freund1997decision : .....
Yoav Freund and Robert E Schapire.
A decision-theoretic generalization of on-line learning and an application to boosting.
Journal of computer and system sciences, 55(1) :119–139, 1997.
— azarbad2015microbial : .....
Hamed Azarbad, Maria Niklińska, Ryszard Laskowski, Nico M van Straalen, Cornelis AM van Gestel, Jizhong Zhou, Zhili He, Chongqing Wen, and Wilfred FM Röling.
Microbial community composition and functions are resilient to metal pollution along two forest soil gradients.
FEMS microbiology ecology, 91(1) :1–11, 2015.

— bell2015early : .....
Terrence H Bell, Benoît Cloutier-Hurteau, Fahad Al-Otaibi, Marie-Claude Turmel, Etienne Yergeau, François Courchesne, and Marc St-Arnaud.
Early rhizosphere microbiome composition is related to the growth and zn uptake of willows introduced to a former landfill.
Environmental microbiology, 17(8) :3025–3038, 2015.

— bell2014linkage : .....
Terrence H Bell, Saad El-Din Hassan, Aurélien Lauron-Moreau, Fahad Al-Otaibi, Mohamed Hijri, Etienne Yergeau, and Marc St-Arnaud.
Linkage between bacterial and fungal rhizosphere communities in hydrocarbon-contaminated soils is related to plant phylogeny.
The ISME Journal, 8(2) :331, 2014.

— conrad1996soil : .....
Ralf Conrad.
Soil microorganisms as controllers of atmospheric trace gases (h2, co, ch4, ocs, n2o, and no).
Microbiological reviews, 60(4) :609–640, 1996.

— danielsen2012fungal : .....
Lara Danielsen, Andrea Thürmer, Peter Meinicke, Marc Buee, Emmanuelle Morin, Francis Martin, Gilles Pilate, Rolf Daniel, Andrea Polle, and Marlis Reich.
Fungal soil communities in a young transgenic poplar plantation form a rich reservoir for fungal root communities.
Ecology and Evolution, 2(8) :1935–1948, 2012.

— egan1975signal : .....
James P Egan.
Signal detection theory and {ROC} analysis.
1975.

— fawcett2006introduction : .....
Tom Fawcett.
An introduction to roc analysis.
Pattern recognition letters, 27(8) :861–874, 2006.

— foulon2016impact : .....
Julie Foulon, Cyril Zappelini, Alexis Durand, Benoit Valot, Damien Blaudez, and Michel Chalot.
Impact of poplar-based phytomanagement on soil properties and microbial communities in a metal-contaminated site.
FEMS microbiology ecology, 92(10), 2016.

— foulon2016environmental : .....
Julie Foulon, Cyril Zappelini, Alexis Durand, Benoit Valot, Olivier Girardclos, Damien Blaudez, and Michel Chalot.
Environmental metabarcoding reveals contrasting microbial communities at two poplar phytomanagement sites.
Science of the Total Environment, 571 :1230–1240, 2016.

— gottel2011distinct : .....
Neil R Gottel, Hector F Castro, Marilyn Kerley, Zamin Yang, Dale A Pelletier, Mircea Podar, Tatiana Karpinets, ED Uberbacher, Gerald A Tuskan, Rytas Vilgalys, et al.
Distinct microbial communities within the endosphere and rhizosphere of populus deltoides roots across contrasting soil types.
Applied and environmental microbiology, 77(17) :5934–5944, 2011.

— hanley1982meaning : .....
James A Hanley and Barbara J McNeil.
The meaning and use of the area under a receiver operating characteristic (roc) curve.
Radiology, 143(1) :29–36, 1982.

— hong2015illumina : .....
Chen Hong, Yanxiao Si, Yi Xing, and Yang Li.
Illumina miseq sequencing investigation on the contrasting soil bacterial community structures in different iron mining areas.
Environmental Science and Pollution Research, 22(14) :10788–10799, 2015.

— jeffries2003contribution : .....
Peter Jeffries, Silvio Gianinazzi, Silvia Perotto, Katarzyna Turnau, and José-Miguel Barea.
The contribution of arbuscular mycorrhizal fungi in sustainable maintenance of plant health and soil fertility.
Biology and fertility of soils, 37(1) :1–16, 2003.

— kozdroj2000microflora : .....
Jacek Kozdrój.
Microflora of technogenous wastes characterised by fatty acid profiling.
Microbiological research, 155(3) :149–156, 2000.

— lallias2015environmental : .....
Delphine Lallias, Jan G Hiddink, Vera G Fonseca, John M Gaspar, Way Sung, Simon P Neill, Natalie Barnes, Tim Ferrero, Neil Hall, P John D Lambshead, et al.
Environmental metabarcoding reveals heterogeneous drivers of microbial eukaryote diversity in contrasting estuarine ecosystems.
The ISME journal, 9(5) :1208, 2015.

— morrison2003receiver : .....
Ann Michelle Morrison, Kelly Coughlin, James P Shine, Brent A Coull, and Andrea C Rex.
Receiver operating characteristic curve analysis of beach water quality indicator variables.
Applied and environmental microbiology, 69(11) :6405–6411, 2003.

— op2015impact : .....
Michiel Op De Beeck, Bart Lievens, Pieter Busschaert, Francois Rineau, Mark Smits, Jaco Vangronsveld, and Jan V Colpaert.
Impact of metal pollution on fungal diversity and community structures.
Environmental microbiology, 17(6) :2035–2047, 2015.

— schmidt2013illumina : .....
Philipp-André Schmidt, Miklós Bálint, Bastian Greshake, Cornelia Bandow, Jörg Römbke, and Imke Schmitt.
Illumina metabarcoding of a soil fungal community.
Soil Biology and Biochemistry, 65 :128–132, 2013.

— tedersoo2014global : .....
Leho Tedersoo, Mohammad Bahram, Sergei Põlme, Urmas Kõljalg, Nourou S Yorou, Ravi Wijesundera, Luis Villarreal Ruiz, Aída M Vasco-Palacios, Pham Quang Thu, Ave Suija, et al.
Global diversity and geography of soil fungi.
science, 346(6213) :1256688, 2014.

— wu2015molecular : .....
Zhaoxiang Wu, Zhipeng Hao, Yan Zeng, Lanping Guo, Luqi Huang, and Baodong Chen.
Molecular characterization of microbial communities in the rhizosphere soils and roots of diseased and healthy panax notoginseng.
Antonie van Leeuwenhoek, 108(5) :1059–1074, 2015.

— yergeau2015transplanting : .....
Etienne Yergeau, Terrence H Bell, Julie Champagne, Christine Maynard, Stacie Tardif, Julien Tremblay, and Charles W Greer.
Transplanting soil microbiomes leads to lasting effects on willow growth, but not on the rhizosphere microbiome.
Frontiers in microbiology, 6, 2015.

— zappelini2015diversity : .....
Cyril Zappelini, Battle Karimi, Julie Foulon, Laurence Lacercat-Didier, François Maillard, Benoit Valot, Damien Blaudez, David Cazaux, Daniel Gilbert, Etienne Yergeau, et al.
Diversity and complexity of microbial communities from a chlor-alkali tailings dump.
Soil Biology and Biochemistry, 90 :101–110, 2015.

— zou2002receiver : .....
Kelly H Zou.
Receiver operating characteristic (roc) literature research.
On-line bibliography available from :¡ http ://splweb. bwh. harvard. edu, 8000, 2002.

— robin2011proc : .....
Xavier Robin, Natacha Turck, Alexandre Hainard, Natalia Tiberti, Frédérique Lisacek, Jean-Charles Sanchez, and Markus Müller.
proc : an open-source package for r and s+ to analyze and compare roc curves.
BMC bioinformatics, 12(1) :77, 2011.

— sing2005rocr : .....
Tobias Sing, Oliver Sander, Niko Beerenwinkel, and Thomas Lengauer.
Rocr : visualizing classifier performance in r.
Bioinformatics, 21(20) :3940–3941, 2005.

— scikit-learn : .....
F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay.
Scikit-learn : Machine learning in Python.
Journal of Machine Learning Research, 12 :2825–2830, 2011.

— nelson2006news : .....
Emma Nelson, John Manning, and Anthony Sinclair.
News using the length of the 2nd to 4th digit ratio (2d : 4d) to sex cave art hand stencils : factors to consider.
Before Farming, 2006(1) :1–7, 2006.

— dempster1977maximum : .....
Arthur P Dempster, Nan M Laird, and Donald B Rubin.
Maximum likelihood from incomplete data via the em algorithm.
Journal of the royal statistical society. Series B (methodological), pages 1–38, 1977.

— lebret2015rmixmod : .....
Rémi Lebret, Serge Iovleff, Florent Langrognet, Christophe Biernacki, Gilles Celeux, and Gérard Govaert.
Rmixmod : the r package of the model-based unsupervised, supervised and semi-supervised classification mixmod library.
Journal of Statistical Software, 67(6) :241–270, 2015.
— Mixmod : .....
mixmod.
http://www.mixmod.org/spip.php?article1.
— watson1953molecular : .....
James D Watson, Francis HC Crick, et al.
Molecular structure of nucleic acids.
Nature, 171(4356) :737–738, 1953.
— anglin1996semiparametric : .....
Paul M Anglin and Ramazan Gencay.
Semiparametric estimation of a hedonic price function.
Journal of Applied Econometrics, 11(6) :633–648, 1996.
— OverFitting : .....
when to think less (according to math).
https://www.youtube.com/watch?v=sivWzd_AecU.
— cauchy1847methode : .....
Augustin Cauchy.
Méthode générale pour la résolution des systemes d'équations simultanées.
Comp. Rend. Sci. Paris, 25(1847) :536–538, 1847.
— pearson1901liii : .....
Karl Pearson.
Liii. on lines and planes of closest fit to systems of points in space.
The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 2(11) :559–572, 1901.
— steinhaus1956division : .....
Hugo Steinhaus.
Sur la division des corp materiels en parties.
Bull. Acad. Polon. Sci, 1(804) :801, 1956.
— kusters2008dodge : .....
Ulrich Küsters.
Dodge, yadolah : The concise encyclopedia of statistics. new york, 2008.
OR News : das Magazin der GOR, (33), 2008.
— smith1981comparison : .....
Temple F Smith and Michael S Waterman.
Comparison of biosequences.
Advances in applied mathematics, 2(4) :482–489, 1981.
— rusk2010torrents : .....
Nicole Rusk.
Torrents of sequence.
Nature Methods, 8(1) :44, 2010.
— Illumina : .....
illumina.
https://www.illumina.com/techniques/sequencing.html.

— el2007evolution : .....
Elmostafa EL FAHIME and Mly Mustapha ENNAJI.
Évolution des techniques de séquençage.
Les technologies de laboratoire, 2(5), 2007.
— international2004finishing : .....
nternational Human Genome Sequencing Consortium et al.
Finishing the euchromatic sequence of the human genome.
Nature, 431(7011) :931, 2004.
— UCSC : .....
université de californie à santa cruz.
https://genome-euro.ucsc.edu/cgi-bin/hgGateway.
— sur2018modern : .....
Pragya Sur and Emmanuel J Candès.
A modern maximum-likelihood theory for high-dimensional logistic regression.
arXiv preprint arXiv :1803.06964, 2018.
— jacques2014functional : .....
Julien Jacques and Cristian Preda.
Functional data clustering : a survey.
Advances in Data Analysis and Classification, 8(3) :231–255, 2014.

Document réalisé avec LaTeX et :
le style LaTeX pour Thèse de Doctorat créé par S. Galland — http://www.multiagent.fr/ThesisStyle
la collection de paquets tex-upmethodology — http://www.arakhne.org/tex-upmethodology/

Titre : Apport de techniques d'analyse de données pour résoudre des problèmes spécifiques en bio-informatique

Mots clés : Bio-informatique, Statistique, Clustering de séquences génétiques, Éléments transposables, Courbes ROC, LASSO, Régression polytomique ordonnée

Résumé : De nos jours, la quantité de données génétiques séquencées augmente de manière exponentielle sous l'impulsion d'outils de séquençage de plus en plus performants, tels que les outils de séquençage haut débit en particulier. De plus, ces données sont de plus en plus facilement accessibles grâce aux bases de données en ligne. Cette plus grande disponibilité des données ouvre de nouveaux sujets d'étude qui nécessitent de la part des statisticiens et bio-informaticiens de développer des outils adaptés. Par ailleurs, les progrès constants de la statistique, dans des domaines tels que le clustering, la réduction de dimension, ou les régressions entre autres, nécessitent d'être régulièrement adaptés au contexte de la bio-informatique. L'objectif de cette thèse est l'application de techniques avancées de statistiques à des problématiques de bio-informatique. Dans ce manuscrit, nous présentons les résultats de nos travaux concernant le clustering de séquences génétiques via Laplacian eigenmaps et modèle de mélange gaussien, l'étude de la propagation des éléments transposables dans le génome via un processus de branchement, l'analyse de données métagénomiques en écologie via des courbes ROC ou encore la régression polytomique ordonnée pénalisée par la norme l_1.

Title : Use of data analysis techniques to solve specific bioinformatics problems

Keywords : Bioinformatics, Statistic, DNA clustering, Transposable elements, ROC analysis, LASSO, Ordinal polytomous regression

Abstract : Nowadays, the quantity of sequenced genetic data is increasing exponentially under the impetus of increasingly powerful sequencing tools, such as high-throughput sequencing tools in particular. In addition, these data are increasingly accessible through online databases. This greater availability of data opens up new areas of study that require statisticians and bioinformaticians to develop appropriate tools. In addition, constant statistical progress in areas such as clustering, dimensionality reduction, regressions and others needs to be regularly adapted to the context of bioinformatics. The objective of this thesis is the application of advanced statistical techniques to bioinformatics issues. In this manuscript we present the results of our works concerning the clustering of genetic sequences via Laplacian eigenmaps and Gaussian mixture model, the study of the propagation of transposable elements in the genome via a branching process, the analysis of metagenomic data in ecology via ROC curves or the ordinal polytomous regression penalized by the l_1-norm.

Université Bourgogne Franche-Comté
32, avenue de l'Observatoire
25000 Besancon