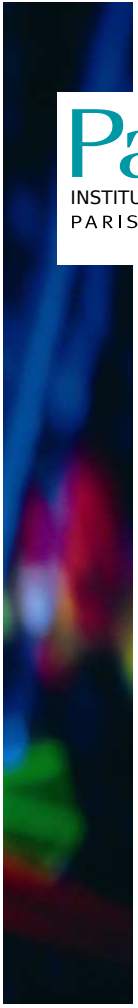
Feature selection from gene expression data : molecular signatures for breast cancer prognosis and gene regulation network inference

Anne-Claire Haury



ParisTech
INSTITUT DES SCIENCES ET TECHNOLOGIES
PARIS INSTITUTE OF TECHNOLOGY

MINES ParisTech

École doctorale n°432 :
Sciences des métiers de l'ingénieur

Doctorat européen ParisTech

T H È S E

pour obtenir le grade de docteur délivré par

l'École nationale supérieure des mines de Paris

Spécialité « Bio-informatique »

présentée et soutenue publiquement par

Anne-Claire HAURY

le 14 décembre 2012

Sélection de variables à partir de données d'expression

– Signatures moléculaires pour le pronostic du cancer du sein et inférence de réseaux de régulation génique –

~ ~ ~

Feature selection from gene expression data

– Molecular signatures for breast cancer prognosis and gene regulatory network inference –

Directeur de thèse : **Jean-Philippe VERT**

Jury

Terence P. SPEED, Professeur émérite, Department of Statistics, University of California, Berkeley	Rapporteur
Florence D'ALCHE-BUC, Professeur, Laboratoire IBISC, Université d'Evry-Val d'Essonne	Rapporteur
Francis BACH, Ingénieur en chef des Mines, Laboratoire d'informatique, Ecole Normale Supérieure	Examinateur
Fabien Reyal, Praticien de centre, Département de chirurgie oncologique, Institut Curie	Examinateur
Jean-Philippe VERT, Maître de recherche, Centre de Bio-Informatique, Mines ParisTech	Examinateur

T H È S E

MINES ParisTech
Centre de Bio-Informatique
35 rue Saint-Honoré, 77300 Fontainebleau, France

To my father.

Faith has been broken, Tears must be cried,
Let's do some living after we die.

The Rolling Stones, *Wild Horses*, 1971

Acknowledgments/Remerciements

Believe me, it does not get easier than this! After days and nights writing this thesis - and obviously years working on it - it is the best of rewards, really, to get to thank the many people who have contributed, one way or another, to making this happen. Because so many are involved, who might not even be aware of it, I feel that it is my duty not to spoil this. Colleagues, friends, family, nothing I could write could ever make up for everything these people have done for me.

Those of you who have ever received an email from me will not be surprised by the length of this section, which I understand is the one place in this dissertation where I should get personal. Because I might have broken this rule anyway, I will give this my very all. So, with the immense fear of forgetting someone and sudden compassion for those who have to accept an Oscar in a five minute talk, I shall begin...

If you are bored already, keywords include: gratitude, couldn't have done it alone, thanks, thanks again, you have no idea how helpful you have been. Now down to it.

To the jury

Let me first thank Terry Speed, Florence d'Alché-Buc, Francis Bach and Fabien Reyal for accepting to be part of a great jury.

I am truly honored that such brilliant people would spend time on my work. I can only hope to have made it somewhat worth their while.

A ceux qui m'ont accueillie

Mes premiers remerciements francophones vont à Jean-Philippe Vert, *a.k.a.* Sensei. Je n'en rajoute pas et peut-être même n'en dis-je pas assez en affirmant que je n'aurais pu rêver un meilleur directeur de thèse. Disponible, à l'écoute, il prend le temps pour chacun, valorise ce qui est bien, et transforme ce qui l'est moins. Je suis d'une incroyable reconnaissance envers celui qui m'a tant appris et ne m'aventurerai pas à lister ici toutes les qualités scientifiques et humaines que je lui trouve, je serai certaine d'en oublier.

Je souhaite également remercier Emmanuel Barillot pour m'avoir ouvert les portes de son unité, mais aussi pour avoir toujours pris le temps de s'intéresser à mon travail et m'avoir donné de nombreux et précieux conseils et avis.

Le CBIO, CBIEN!

A Véronique Stoven, un grand merci, ou plutôt deux: un pour la constructivité de sa critique en sciences et un pour sa rare et profonde humanité - mes mots sont pesés et ma balance n'en supporte pas le poids. A Christian Lajaunie, dont je ne peux suivre le rythme ni en VTT ni en mathématiques, et Isabelle Schmitt, dont je loue la patience, l'écoute et l'efficacité, merci pour votre accueil à Fontainebleau! Ich möchte mich auch bei Thomas Walter herzlich bedanken, da seine unaussprechlichen Menschlich- und Nettigkeit viel geholfen haben, eigentlich wahrscheinlich mehr als er denkt.

Chaque famille a ses anciens, les miens sont jeunes mais pas moins sages. J'adresse de chaleureux remerciements à Fantine Mordelet pour sa gentillesse incomparable, pour m'avoir prise sous son aile, aidée et guidée jusqu'à aujourd'hui (littéralement, merci pour la page de garde!) ainsi qu'à Laurent Jacob, dont j'envie la patience, admire la générosité et apprécie la franchise. Le CBIO, pour moi, c'est aussi d'excellents souvenirs avec Kevin Bleakley, aussi efficace pour faire fonctionner un algorithme que lorsqu'il s'agit de passer une bonne soirée! A Martial Hue, Brice Hoffmann et Mikhail Zaslavskiy, un grand merci également pour leur accueil au CBIO et leur aide au quotidien (dont j'ai certes eu moins besoin le jour où j'ai découvert la commande "man"). Special thanks to Yoshihiro Yamanishi, who might simply be the nicest person in the world and whom I have been missing a lot!

C'est qu'il y a eu du turn-over depuis. Je ne cache pas une petite crainte quand j'ai vu arriver coup sur coup le talentueux Toby Hocking, l'étonnant Pierre Chiche, le charmant Edouard Pauwels et le généreux Matahi Moarii. Puis nous ont rejoint Andrea Cavagnino, Kévin Vervier et Emile Richard, que je regrette de ne pas avoir connu davantage. En effet, j'ai eu peur un instant pour le girl power du CBIO mais il se trouve qu'on est tombés sur les meilleurs. Un peu (beaucoup) geeks, passionnés et surtout adorables. Les garçons, vous allez me manquer! Bien évidemment, j'étais ravie (et rassurée) quand ont débarqué Nelle Varoquaux (et son python), Elsa Bernard (et son humour) et Alice Schoenauer (et sa passion) qui, j'en suis sûre, sauront faire cohabiter geekitude et douceur dans ce petit monde. Je remercie enfin Khalid Jebbari pour avoir bien voulu travailler avec moi.

En résumé, un grand merci au CBIO, où j'ai trouvé une nouvelle petite famille accueillante et d'un grand soutien.

Mon unité

Travailler à Curie, c'est aussi travailler avec l'ensemble des personnes formidables qui ont composé, composent, composeront l'U900 (U pour unité et 900 points pour l'ambiance, qui est notée sur 900). Ils sont légèrement plus nombreux qu'au CBIO, la crainte d'oublier certains d'entre eux m'envahit et je demande pardon d'avance! Pierre Gestraud mérite de ma part de nombreux remerciements mais également tout mon respect pour avoir réussi subrepticement à s'infiltrer au CBIO, bravo! En particular, muchas gracias a Paola Vera-Licona, quien fue un gran apoyo y ayuda en el trabajo, claro, pero tambien durante algunas noches. Parmi mes nombreux collègues, je voudrais remercier pour leur soutien tout au long de ma thèse Alexandre Hamburger, Cécile Laurent, Anne Biton, Fanny Coffin, Laurence Calzone, Simon Fourquet et Bruno Zeitouni. Au quotidien, pour leur aide parfois, leur humour souvent et leur gentillesse surtout, de sincères et chaleureux remerciements à Valentina Boeva, Gautier Stoll, Stéphane Liva, Loredana Martignetti, Patrick Poullet, Bruno Tesson, Séverine Lair, Nicolas Servant, David Cohen, Jennifer Montagne, Isabelle Sanchez, Philippe Hupé, Caroline Paccard, Stuart Pook, Jonas Mandel, Camille Sabbah, Benjamin Sadacca, Isabel Brito, Georges Lucotte, Andrei Zinovyev, Philippe La Rosa, Grégory Duval et Stéphane Cyrulik.

Go bears!

I was lucky enough to spend six months in Berkeley and meet great people there. It is not easy to fit in upon arrival. I would like to thank Sandrine Dudoit from the bottom of my heart for welcoming me, making me feel at home, taking me to the city and introducing me to amazing people. Special thanks go to Terry Speed for taking interest in my work and introducing me to Ljubomir Buturovic, who I am very grateful to for trusting me, as well as Giulia Kennedy and the Veracyte team. I would like to particularly thank Yuval Benjamini for answering many, many, many questions at the Berkeley Espresso, as well as Elizabeth Purdom, Miles Lopes, François Pépin and Davide Risso. But Berkeley was not just about work and I would therefore like to thank all the people who made it easy and fun including Alice Cleynen, Gergana Bounova and Abha Biais from the lab. Many thanks in particular to his swedishness Henrik Bengtsson for the friendship he offered. Some people just know where it's at, which is probably why I met there Laurent Jacob and Julien Mairal, who, for some reason, were not annoyed when I asked about Lasso regularization path for the hundredth time. The rest of my Berkeley friends shall remain surnameless, in case they would not want to be linked up to me. Obviously, I have Noémie, Joan and Clémence to thank quite a lot. It would be very complicated to explain here exactly why, I think they will get it. Finally, many thanks to Aurélie, Jérôme, Jackie, Bahador, Marine, Dennis, Scottie and, last but certainly not least, Magali, for making my american life very fun!

Always around

Le CBIO a un alter ego. Il s'appelle Willow ou Sierra, ce n'est toujours pas très clair pour moi mais je sais en tout cas que j'y ai rencontré beaucoup de personnes bien plus douées que moi en optimisation et autres traitement du signal, et qui plus est suffisamment gentilles pour m'aider, me soutenir, me conseiller. Je tiens donc à remercier en particulier et une seconde fois Julien Mairal, mais également Guillaume Obozinski et Rodolphe Jenatton, qui se souviendront peut-être d'un grand nombre de questions de ma part, probablement sans grand intérêt pour eux! Il n'en fait pas partie mais ne m'en voudra pas de tout confondre, merci à Zaid Harchaoui.

Et il y a ceux avec qui j'ai partagé une salle de cours ou un amphi et qui, sans doute plus par solidarité que par intérêt (mais sait-on jamais), sont venus, quand ils le pouvaient, assister à mes présentations, rendant, par leur présence, l'assemblée bien moins inquiétante ou encore se sont intéressés à mon travail, de loin, de près, rassurants et encourageants. Je pense en particulier à Camille Charbonnier, Armand Joulin, Augustin Lefèvre, Sylvain Robbiano, Rémi Bardenet, Florent Couzinié-Devy et Alexandra Carpentier.

A ceux qui m'ont appris

Je ne serais pas à quelques jours de soutenir une thèse de doctorat sans le soutien, les conseils et les encouragements de ceux qui m'ont poussée dans cette voie et qui m'ont donné confiance en moi. Il est temps, enfin, de remercier les professeurs admirables qui ont cru en moi et dont certains ont bien voulu me confier leurs étudiants pour quelques semestres. En particulier, je suis incroyablement reconnaissante à Jean-Pierre Leca, Jean-Marc Bardet, Annie Millet, Sandie Souchet, Denis Pennequin, Joseph Rynkiewicz et l'ensemble des professeurs croisés à Paris 1 pour leur immense humanité, pour le temps qu'ils m'ont consacré et la passion qu'ils m'ont transmise. Je n'oublie pas Brigitte Augarde, sans qui, m'a-t-on confié, tout se serait arrêté il y a bien longtemps, ni Corinne Aboulker. A l'ENSAE, j'ai eu la chance de rencontrer et d'être soutenue par Nicolas Chopin et Alexandre Tsybakov, qui a bien voulu diriger mon mémoire. Je dois également beaucoup à Eric Gautier, Pierre Neuvial et Pierre-Yves Bourguignon. De mon stage à Pasteur où je me suis plongée pour la première fois dans le vif du sujet, je me dois de remercier en particulier Marie-Agnès Dillies et Pierre Latouche pour m'avoir tant appris, mais également Caroline Proux, Guillaume Soubigou, Jean-Yves Coppée, Matthieu Barthélémy et Odile Sismeiro. Du MVA, je remercie chaleureusement Alain Trouvé, qui a tout mis en œuvre pour rendre mon double cursus plus aisé, et, pour leur pédagogie, leur écoute et leur passion, je remercie Bernard Chalmond, Nicolas Vayatis, Jean-Yves Audibert, Rémi Munos, et, pour les citer une seconde fois, Jean-Philippe Vert et Francis Bach. Il n'est jamais trop tôt pour transmettre une passion et c'est ce qu'ont réussi Pascal Manot à Lakanal, Françoise Bouchardy et Marguerite Mulot au Lycée Franco-Allemand. Enfin, c'est un bac L que j'aurais (peut-être) obtenu si Agnès Alexandre n'avait pas convaincu une capricieuse gamine de 17 ans qu'elle avait

un avenir en sciences.

Les copains d'abord

"Des bateaux, j'en ai pris beaucoup, mais le seul qui ait tenu le coup, qui n'ait jamais viré de bord"... C'est un bateau incroyable, avec un capitaine collégial, un bateau qui ne coulera jamais, j'en mets trois mains au feu, car ses matelots sont parmi les personnes les plus généreuses, fidèles, honnêtes, encourageantes, fortes et protectrices qu'il me sera jamais donné de rencontrer. Sa cale est solide, ses voiles sont hissées depuis plus de 15 ans et il poursuivra tranquillement sa route, désarmant l'ennemi et contournant les obstacles. La sincérité de mes remerciements à ceux qui le dirigent si bien n'a d'égal que l'amour sans limite que je leur porte. Merci, merci pour tout à Béné, Chris, Claire L., Claire M., Cyril, Cyrille, Diane, Franz, Gus, Hugo, Jay, Laure, Line, Lucie, Mathieu, Manu, Mike, Morgane, Nico, Polo, Tarek et Vincent.

A mes petites cigales de la Fourmi, qui, au temps chaud comme au temps froid, ne traverseront jamais la Seine et n'emprunteront probablement jamais le métro en dehors de la ligne 2, ceux dont le moindre défaut est d'avoir tenté de me distraire ces trois dernières années (au moins), ceux qui prêtent peu mais donnent beaucoup, à qui je dois des années de bonheur, un grand merci! Dans le rôle des cigales sont nominés et obtiennent tous le Hauryar du meilleur premier rôle: Alex, Anne, Camille, Flo, Louis et Marion.

Je ne pouvais le citer dans aucun autre paragraphe, qu'à cela ne tienne, en voilà un pour lui seul! Merci à Adrien!

Je tiens également à remercier Franck et Pascal, qui sauront pourquoi.

Finally, I would never ever have gotten there without the help of two true friends, namely Internet and Computer, that have proven to be amazing listeners of great help and besides, that have made themselves available day and night throughout this thesis!

A ma famille

Je remercie ma famille qui, si elle n'est pas bien grande n'en est pas moins le plus incroyable des soutiens. Maman, Fred, comment vous remercier si ce n'est en vous affirmant que vous êtes ma plus grande force! Nanou, Mick, Françoise, Alain, Guillaume, Marion et Cédric, merci pour vos encouragements, et pour avoir cru en moi depuis toujours et probablement bien plus que moi! A Fabrice, que j'ai décidé de remercier dans ce paragraphe, un immense merci pour son aide inconditionnelle, sa générosité et sa patience que je n'aurai jamais.

Mes pensées et mes remerciements vont également à mon arrière grand-mère, et mes grands-parents disparus, qui ont toujours su m'encourager.

Enfin, je tiens à remercier mon père pour m'avoir appris la ténacité, l'honnêteté intellectuelle et la franchise et pour m'avoir transmis, je l'espère, son courage.

Contents

List of Figures

List of Tables

Abstract

Biotechnologies have moved the paradigm of gene expression analysis from a hypothesis-driven to a data-driven approach. Ever since "omics" data have been made available, we are indeed facing a sea of data that we need to mine in order to extract relevant biological information. However, the ease with which it is now possible to obtain genome-wide data is not at all illustrative of the complexity of such data, which is referred to as being *high-dimensional*.

The technology of DNA microarrays, for example, makes it possible to measure the expression of thousands of genes simultaneously and in several conditions. In this thesis we work on two applications involving microarray expression data: *prediction of breast cancer outcome* and *gene regulatory network inference*.

In this thesis, we address these two problems with the same statistical approach, namely the association of *supervised learning* and *feature selection*. Supervised learning refers to elucidating patterns linking some data to a response using a set of examples and to using the inferred patterns to predict the response when new data comes along. Feature selection is the process of extracting relevant information from data possibly clouded by high quantities of noise. Among the feature selection methods we discuss, we are interested in the so-called *Ensemble methods* that consist in incorporating *randomization* techniques within the selection process through the use of *resampling* or *bootstrap*.

Prediction of breast cancer outcome

Being able to predict breast cancer outcome is a pressing issue as the treatment depends on the prognosis. In fact, if there is an indication that a primary cancer will relapse or metastasize, adjuvant chemotherapy after, e.g., a surgery, is the preferred treatment to increase the chances of survival of the patient. However, chemotherapy is only harmful to *good outcome* patients, i.e., in cases where the cancer is not recurrent. Practicians thus have to estimate the chances of recurrence to decide if and when such adjuvant therapy is necessary. Some clinical and histological factors have been helpful to this task, such as the size and grade of the tumor, the age of the patient or the cancer subtype. However, the accuracy of these indicators is not as

high as we would hope.

The accessibility of gene expression information through DNA microarrays has opened new perspectives to this problem. In 2002, Van't Veer and colleagues suggested that mining through microarrays would make it possible to extract *molecular signatures* predictive of the outcome. A signature consists of a list of a few tens of genes believed to contain necessary and sufficient information to assess the likelihood of recurrence of a cancer with some success. These genes are selected based on patterns inferred from the *expression profiles* of the primary tumor using *feature selection methods*. The outcome is then predicted through *supervised learning models*.

This idea appealed to the community and soon tens of new signatures would be published. However, they were only sharing very few genes, if any, which would soon beg questions about the reliability of such models.

The first three chapters of this thesis contribute to this very problem. In Chapter 2, we set up a benchmark in order to compare 32 feature selection methods in light of the accuracy, stability and interpretability of the signatures they find. We show, in particular, that the simplest methods seem to provide the best accuracy/stability trade-off. In Chapter 3, we propose to incorporate biological prior information to the selection procedure using the *Graph Lasso* with the hope of increasing the stability and interpretability of prognosis signatures. We show that adding such prior information can be useful in terms of interpretation, but does not seem to return more accurate or stable signatures. In Chapter 4, we use a different approach, enforcing *structured sparsity* through model penalization by the k-support norm, in order to select genes resembling each other without using any prior information. We propose to add two levels of randomization to the algorithm. Our experiments suggest that the resulting method overperforms previously investigated ones in terms of accuracy. Finally, in Chapter 6, we discuss the current state of research in this field, try to look with hindsight at ten years of findings and investigate the apparent trade-off between accuracy and stability.

Inferring Gene Regulatory Networks

The expression of genes is regulated by proteins called *transcription factors* (TF) that bind to promoter regions and either activate or repress the expression of *target genes* (TG). Because TFs themselves result from genes being transcribed at some point, the patterns of regulation can be represented on a directed graph, where nodes and edges represent genes and regulation directions, respectively. Such a graph is also called a *Gene Regulatory Network* (GRN). Understanding TF-TG interactions has many potential applications in biology and medicine, ranging from *in silico* modeling and simulation of GRN to the identification of new potential drug targets.

GRN inference can be seen as a series of *supervised learning* tasks, where the expression of each TG is predicted by the expression of the TFs regulating it. A typical assumption is the *sparsity* of these networks, in that only few TFs are believed to regulate a TG. Therefore, *feature*

selection can be used to detect the set of TFs interacting with each TG.

It is this approach that we choose to follow in Chapter 5. We describe an algorithm named TIGRESS, with which we participated into the DREAM5 GRN inference challenge. DREAM stands for Dialogue on Reverse Engineering Assessments and Methods and proposes yearly challenges on various biological problems. The GRN inference challenge consisted in inferring three networks, i.e., in predicting probabilities of existence for each edge in each of the three networks. TIGRESS ranked second in the *in silico* network sub-challenge and third overall. The algorithm is based on a popular ℓ_1-penalized method called Lasso on top of which we added two *randomization* layers. In Chapter 5, we analyze the behavior of the algorithm in various situations, assess the impact of the choice of its parameters and benchmark it against state-of-the-art GRN inference methods. Our results are extremely encouraging as to the ability of such *machine learning*-based approaches to solve the GRN inference problem and that of TIGRESS, in particular.

Résumé

De considérables développements dans le domaine des biotechnologies ont modifié notre approche de l'analyse de l'expression génique. En effet, s'il était auparavant nécéssaire de formuler une hypothèse - concernant un gène, par exemple - pour pouvoir la tester, c'est la démarche inverse que l'on suit depuis l'apparition, il y a une quinzaine d'années, des données "omics". Il s'agit désormais d'extraire l'information à partir d'une grande quantité de données, et il est finalement devenu bien plus simple d'obtenir des informations quantitatives à l'échelle du génome que de les analyser. On parle alors de données *en grande dimension*.

Les puces à ADN sont un exemple de ce phénomène. Elles permettent de mesurer l'expression de milliers de gènes de manière simultanée et dans différentes conditions. Dans cette thèse, nous travaillons sur deux applications impliquant l'analyse de telles données : *la prédiction de l'issue du cancer du sein* et *l'inférence de réseaux de régulation génique*.

Bien que ces deux problèmes ne semblent, *a priori*, rien avoir en commun, nous les approchons par la même démarche statistique, à savoir l'association de *l'apprentissage supervisé* et de *la sélection de variables*. L'apprentissage supervisé consiste, à partir d'un ensemble d'exemples, à repérer, ou *apprendre* des liens entre des données et une réponse, de manière à pouvoir prédire la réponse lorsque de nouvelles données se présentent, en utilisant les règles apprises. La sélection de variables se rapporte à un ensemble de méthodes permettant d'extraire l'information *importante* d'une base de données parfois très complexe et potentiellement corrompue par beaucoup de bruit. Parmi les techniques de sélection de variables, nous nous intéressons, entre autres, aux méthodes dites *d'ensemble*, qui consistent à incorporer une part de *randomisation* au processus de sélection, par le biais du *ré-échantillonnage* ou du *bootstrap*.

Prédiction de l'issue du cancer du sein

Le traitement du cancer du sein étant dépendant du *pronostic*, il est capital d'être en mesure de prédire l'issue de la maladie. En effet, si l'on suspecte qu'un cancer primaire donnera lieu à une métastase ou une récurrence, on peut proposer à la patiente une chimio-thérapie adjuvante, c'est-à-dire à la suite d'une opération chirurgicale. Dans de tels cas, ce type de traitement,

destiné à accroître l'espérance de vie de la patiente, est nécessaire. En revanche, dans le cas où le cancer disparaît après l'intervention, la chimio-thérapie est néfaste à la patiente, attaquant uniquement des cellules saines. Les médecins doivent donc estimer la probabilité de récurrence du cancer afin de décider si un tel traitement adjuvant est nécessaire. Des facteurs cliniques et histologiques donnent un certain nombre d'indications et sont un appui pour le pronostic. Cependant, la capacité prédictive de ces estimateurs reste faible.

L'accès aux données d'expression des gènes, grâce à des techniques telles que les puces à ADN, ont ouvert de nouvelles perspectives à ce problème. En 2002, une étude suggérait que de nouveaux indicateurs pouvaient être obtenus en analysant ces puces. Il s'agit d'extraire de ces données ce qu'on appelle des *signatures moléculaires*. Une signature est une liste de quelques dizaines de gènes supposée contenir suffisamment d'information liée au pronostic et qui permettrait donc de calculer la probabilité de récurrence du cancer pour de nouvelles patientes. Ces gènes sont sélectionnés à partir de *profils d'expression géniques* de la tumeur primaire grâce à des méthodes de *sélection de variables*. L'issue est alors prédite par le biais de *modèles d'apprentissage supervisé*.

Cette idée eut immédiatement de l'écho dans la communauté, de telle manière que, très vite, des dizaines de nouvelles signatures furent publiées. Cependant, on se rendit compte qu'elles n'avaient que peu de gènes en commun, ce qui remit en question la fiabilité de ces modèles.

Les trois premiers chapitres de cette thèse sont consacrés aux signatures moléculaires pour le pronostic du cancer du sein. Dans le Chapitre 2, nous proposons une comparaison systématique de 32 méthodes de sélection de variables, du point de vue de leur performance prédictive, de leur stabilité et de leur interprétabilité. En particulier, nous montrons que les méthodes les plus simples semblent fournir le meilleur compromis performance/stabilité. Dans le Chapitre 3, nous proposons d'incorporer à la procédure de sélection de l'information biologique *a priori* par le biais du *Graph Lasso*, dans l'espoir d'accroître la stabilité et l'interprétabilité des signatures. Nous montrons que si l'ajout d'une telle information permet, en effet, d'obtenir des signatures plus interprétables, cela ne semble, en revanche, pas avoir d'effet sur la performance ou la stabilité. Dans le Chapitre 4, nous utilisons une approche différente, à savoir la *parcimonie structurée*, à travers un modèle pénalisé par la norme dite "k-support", dans le but de sélectionner des gènes qui se ressemblent sans cependant utiliser d'information *a priori*. Nous proposons, par ailleurs, d'ajouter deux niveaux de randomisation à la sélection. Il semble, à première vue, que cette approche permette d'accroître la performance des signatures par comparaison aux techniques déjà évaluées. Enfin, dans le Chapitre 6, nous exposons l'état actuel de la recherche dans ce domaine et tentons de prendre du recul sur plus de dix années de publications.

Inférence de réseaux de régulation

L'expression des gènes est régulée par des protéines appelées *facteurs de transcription* (TF), qui, en se fixant sur des séquences en amont des *gènes cibles* (TG), activent ou répriment leur transcription. Les TFs résultant eux-mêmes de gènes ayant été transcrits, ces relations entre gènes peuvent être représentées sur un graphe dirigé, où les noeuds et les arètes représentent respectivement les gènes et les interactions TF-TG. Un tel graphe est appelé *réseau de régulation génique* (GRN). La connaissance et la compréhension des interactions TF-TG ont de nombreuses applications, de la modélisation et la simulation de réseaux *in silico* à l'identification de nouvelles cibles thérapeutiques potentielles.

L'inférence des GRNs à partir de données de puces peut être approchée statistiquement comme une série de tâches *d'apprentissage supervisé*, où l'expression de chaque TG est prédite par l'expression des TFs le régulant. Une hypothèse couramment admise est la *parcimonie* de ces réseaux: on suppose que seul un petit nombre de TFs régule un TG. Il s'agit donc également d'un problème de *sélection de variables* où le but est d'identifier l'ensemble des TFs interagissant avec chaque TG.

Nous adoptons cette approche dans le Chapitre 5. Nous y décrivons un algorithme nommé TIGRESS, avec lequel nous avons participé au challenge d'inférence de réseaux DREAM5. DREAM (Dialogue on Reverse Engineering Assessemnts and Methods) propose chaque année des challenges portant sur différentes questions biologiques. Le problème d'inférence de GRNs consistait à prédire les probabilités d'existence des arètes pour trois réseaux, à partir de données d'expression. TIGRESS a été classée seconde sur le réseau *in silico* et troisième dans l'ensemble. L'algorithme est basé sur le Lasso, une méthode de pénalisation utilisant la norme ℓ_1, à laquelle nous avons ajouté deux niveaux de randomisation. Dans le Chapitre 5, nous analysons le potentiel et les limites de cette méthode dans différentes situations, nous évaluons l'impact de ses paramètres et nous la comparons à l'état de l'art. Les résultats sont très encourageants, validant l'intérêt d'utiliser des méthodes d'apprentissage pour l'inférence des GRNs, et de TIGRESS en particulier.

Chapter 1

Introduction

In this chapter, we aim at providing some mathematical and methodological background on two main concepts present in this thesis, namely *supervised learning* and *feature selection*.

Supervised learning is concerned with the inference of a rule that links some input and output data. The inference is performed on *training samples* for which we are given the output, or the *response*. The aim is to use the learned rule to *predict* the response when new input data is given. The problem of breast cancer outcome prediction, for example, amounts to understanding the relation between the expression of genes and the metastatic status of the cancer. Detailed overviews on supervised learning are given in, e.g., Vapnik (1998), Hastie et al. (2009), Bishop (2006).

Feature selection consists in mining the entire dataset with the goal of identifying the relevant variables, with respect to the problem at hand. Feature selection can be performed either through *subset selection*, i.e., by seeking the best group of features among those available, or through *feature ranking* that consists in ordering the variables by their *importance*. References on feature selection methods include Guyon and Elisseeff (2003), Liu and Motoda (2007).

This chapter is organized as follows. Section 1.1 provides a general introduction to supervised learning and introduces the concept of *regularization*. Section 1.2 is concerned with *feature selection* and *feature ranking*. In Section 1.3, we provide guidelines to correctly *evaluate* and *compare* models. Finally, Section 1.4 provides an overview of the contributions of this thesis.

1.1 Supervised learning to predict and understand

As generally as it gets, *machine learning* can be defined as a set of powerful tools to make sense of data. Through modeling and algorithms, one major goal of the discipline is to extract interesting patterns from the data and take advantage of them to make sensible decisions.

In this thesis, we focus on *supervised learning*, where the response to a specific question is known for a given set of *observations* called *training set* and the aim is to *predict* the response

when new input is given. When the response is discrete, i.e., encoded as classes, the problem is called *classification*. On the other hand, real responses call for *regression* models and algorithms.

This section first introduces supervised learning in a very general way. We then discuss two important concepts in this field, referred to as the approximation/estimation trade-off and the accuracy/interpretability dilemma. The last part of this section focuses on a particular type of models, called *penalized* methods.

1.1.1 Supervised machine learning: problem and notations

The input data consists in n *observations*, each described by a set of p *features*. In the remaining, we will denote the observations by $(x_i)_{i=1...n}$, where for all i, $x_i \in \mathcal{X} \subseteq \mathbb{R}^p$. Concretely, we observe a matrix $\mathbf{X} = (x_{i,j})_{i=1...n,j=1...p}$ consisting of n lines and p columns.

The output data, or *response* is described a vector $\mathbf{Y} = (y_i)_{i=1...n}$. In a *classification* setting, the response consists of C *classes*: $\mathbf{Y} \in \{1...C\}^n$. In the particular case of *binary classification*, $\mathbf{Y} \in \{-1, +1\}^n$. In a *regression* setting, the response is a continuous variable: $\mathbf{Y} \in \mathbb{R}^n$. For instance, predicting the outcome of breast cancer is a binary classification problem, with y_i specifying the outcome ($y_i = +1$ in case of a metastatic event, and -1 otherwise). Predicting the expression level of a gene is a regression problem (expression $y_i \in \mathbb{R}$). For the general case, we write that $y_i \in \mathcal{Y} \subseteq \mathbb{R}$ for all i.

Pairs $(x_i, y_i)_{i=1...n}$ are realizations of the random variables $(X_i, Y_i)_{i=1...n}$ assumed to be i.i.d. from the same unknown joint distribution $\mathbb{P}$ of (X, Y).

We aim at predicting Y from X, i.e., we seek a function f belonging to some set $\mathcal{F} := \{f : \mathcal{X} \to \mathbb{R}, f \text{ measurable}\}$, that we will call a *predictor*, such that $\hat{Y} := f(X)$ is *close* to Y, in some sense. The distance between $f(X)$ and Y is measured by a loss function $l : \mathbb{R} \times \mathcal{Y} \to \mathbb{R}_+$. The smaller $l(f(X), Y)$, the better the prediction. The *performance* of f is measured as the expected loss over $\mathbb{P}$, called *Risk*:

$$R_l(f) = \mathbb{E}_{\mathbb{P}}[l(f(X), Y)]. \tag{1.1}$$

The best f is thus obtained by solving the following *risk functional minimization* problem (Vapnik, 1995)

$$f^* = \arg\min_{f \in \mathcal{F}} R_l(f). \tag{1.2}$$

However, the distribution $\mathbb{P}$ is unknown and the risk $R_l(f)$ is thus not computable. (1.1) is therefore estimated by the *empirical risk*:

$$\hat{R}_l(f) = \frac{1}{n} \sum_{i=1}^{n} l(f(x_i), y_i)$$

and f^* in equation (1.2) approximated by:

$$\hat{f} = \arg\min_{f \in \mathcal{F}} \hat{R}_l(f). \tag{1.3}$$

This program is known as the *empirical risk minimization*.

The first question that arises relates to the choice of l. The second issue is to decide how the set of possible functions $\mathcal{F}$ should be defined.

1.1.2 Loss functions

We are interested in *convex* problems (Boyd and Vandenberghe, 2004) and therefore require that $f \to l(f(x), y)$ be convex for all $(x, y) \in \mathcal{X} \times \mathcal{Y}$.

Square loss

The square loss function is defined as

$$l_{sq}(f(x), y) = (y - f(x))^2.$$

It is mainly used in *regression problems*, such as the *least squares regression* but can also be used in a classification problem. One major advantage of this loss is that it is differentiable everywhere. However, a typical flaw of the square loss is that outliers are heavily penalized, and these penalties enormously affect the empirical risk and hence the answer.

Logistic loss

Commonly used for classification purposes, the *logistic* loss is

$$l_{log}(y, f(x)) = \ln(1 + \exp(-yf(x))).$$

The empirical risk of the logistic loss corresponds to the log-likelihood of the *logistic regression* (Menard, 2001), where it is assumed that $p(Y = 1|X) = \frac{1}{1+\exp(-f(X))} = 1 - \frac{1}{1+\exp(f(X))} = 1 - p(Y = -1|X)$. Therefore, we also have $p(Y = y|X) = \frac{1}{1+\exp(-yf(X))}$ for $y \in \{-1, +1\}$.

Hinge loss

In a classification setting, the hinge loss can also be used:

$$l_{hin}(y, (f(x))) = \max(0, 1 - yf(x)).$$

It should be interpreted from the Support Vector Machine (Vapnik, 1995) viewpoint where $yf(x)$ corresponds to the *margin* that we want to maximize. When $f(x)$ is on the correct side of the separator, i.e., when $sgn(f(x)) = sgn(y)$ and the margin is greater than 1, there is no price to pay. Otherwise, the smaller the margin, the larger the (linear) cost.

Figure 1.1 depicts these three loss functions: the left panel shows the square loss as a function of $y - f(x)$ while the left panel focuses on the classification setting and depicts the logistic and hinge losses. On the latter, we can see how the logistic loss behaves as a softer version of the hinge loss.

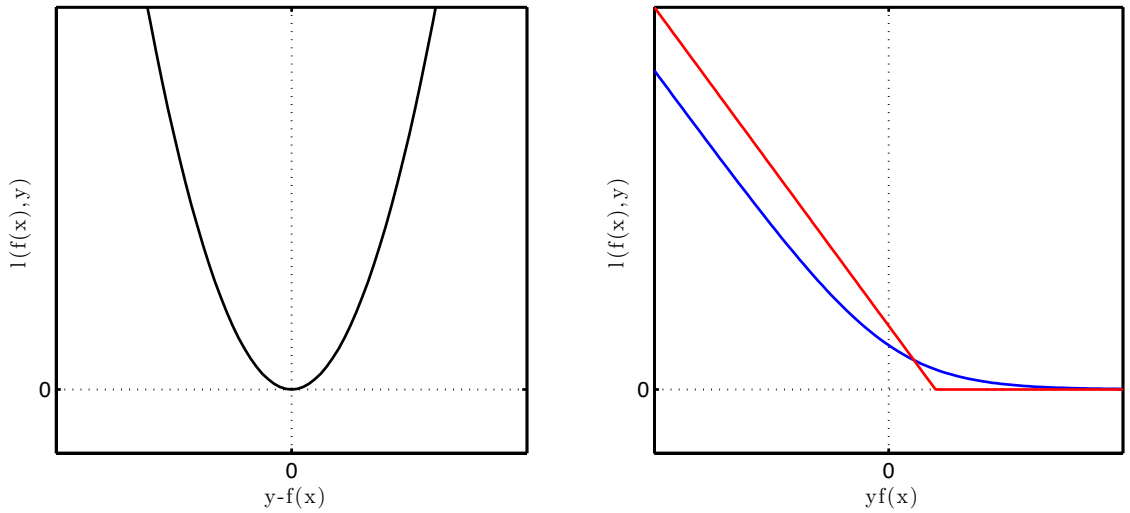


Figure 1.1: **Loss functions**. The left panel shows the square loss. The logistic (blue) and hinge (red) losses are depicted on the right panel.

1.1.3 Defining the complexity of the predictor

Recall that we are looking for the function f that minimizes the empirical risk. We show a two-class classification example on Figure 1.2.

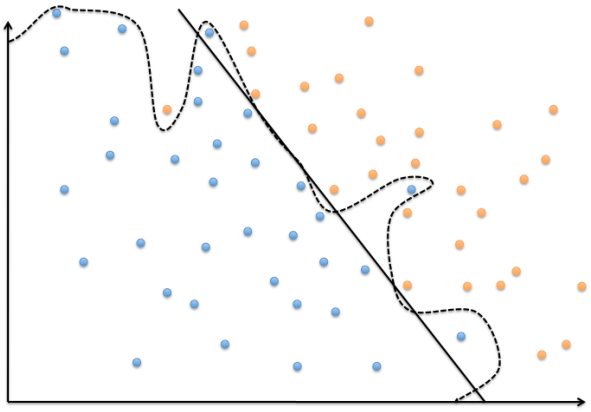


Figure 1.2: **Binary classification problem and overfitting.** The aim to separate the two classes can be achieved with more or less complexity. The dotted separator fits the data perfectly but is prone to *generalizing* poorly. The plain linear separator, on the other hand, allows some training points to be misclassified but will fit new data in a more general and interpretable way.

Each of the two lines depicts a way to separate the blue class and the orange one. The question that we address here is: "which of the two separators is the best?". In this case, there are two main reasons why one should choose the plain simpler classifier. The so-called *approximation/estimation trade-off* is the first and most crucial of them as we develop below.

Depending on the problem, one may also care about the *interpretability* of the solution, which we will discuss as well.

Approximation error and estimation error

One should always keep in mind that the end game is to classify correctly *new data* (Vapnik and Chervonenkis, 1974). A classifier that yields no error on the training set - possibly noisy and containing outliers - is therefore not necessarily the best choice, as it may *generalize poorly* to new points. This behavior is known as *over-fitting*. Another way to see this is to look at the *stability* of the classifier: if one point were to be changed in the training set, how much would it affect the separation? Taking a look at Figure 1.2, we guess that the direction of the plain line would only be slightly modified, if at all. The dotted separator, however, might just surprise us with a very different shape, should one blue point, for the sake of argument, be moved into the middle of the orange crowd.

Formally, there is a need to introduce *restrictions* on f. This is done by searching over some set $\mathcal{H} \subset \mathcal{F}$. Obviously, the chance that $f^* \in \mathcal{H}$ is extremely small. Therefore, the new goal is to find the best possible function $f^*_{\mathcal{H}} \in \mathcal{H}$:

$$f^*_{\mathcal{H}} = \arg\min_{f \in \mathcal{H}} R_l(f). \tag{1.4}$$

It is estimated by

$$\hat{f}_{\mathcal{H}} = \arg\min_{f \in \mathcal{H}} \hat{R}_l(f). \tag{1.5}$$

The error made by restricting the set of possible functions is referred to as the *approximation error* or, with somewhat of a misuse of language, the *bias*.

The *estimation error* of the predictor refers to the difficulty of estimating $f^*_{\mathcal{H}}$ and is sometimes referred to as the *variance*.

Constraining the set will increase the approximation error, as chances are high that the true predictor $f^* \notin \mathcal{H}$. However, it will make the estimation easier to control and thus reduce the estimation error. On the other hand, the less we constrain f, the closest our model to the *true underlying model* assumed to have generated the data. But in this case, we risk having difficulties estimating it.

This trade-off is often referred to as the *approximation/estimation dilemma* - and, with some misuse of language, to the *bias/variance dilemma* - and is illustrated on Figure 1.3 where the total error is decomposed into the bias and the variance terms:

$$\begin{aligned} \text{total error} &= R_l(f_{\mathcal{H}}) - R_l(f^*) \\ &= \underbrace{R_l(f_{\mathcal{H}}) - R_l(f^*_{\mathcal{H}})}_{\text{estimation error}} + \underbrace{R_l(f^*_{\mathcal{H}}) - R_l(f^*)]}_{\text{approximation error}} \end{aligned}$$

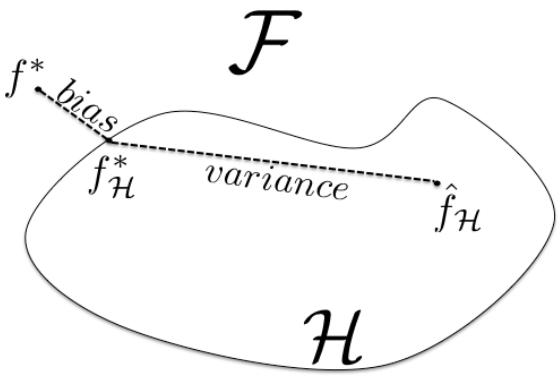


Figure 1.3: **Approximation error and estimation error.** The error made when choosing $\hat{f}_{\mathcal{H}}$ to estimate f^* can be seen as the sum of bias and variance. The bias refers to the approximation error and the variance to the estimation error.

This trade-off is a crucial concept in machine learning with enormous consequences. One way to reduce, if not overcome it, is to think of $\mathcal{H}$ as a set *constraints*, which leads to the definition of *penalized methods*, that we describe in section 1.1.4. By forcing the objective function to meet some constraints, we achieve our goal to restrict the search. We explain in Section 1.1.4 how it can be done in a way that allows us to take into account specificities of our problem.

The interpretability/accuracy dilemma

Although not always a critical point, outputting an interpretable solution can be a requirement in some applications. Breast cancer prognosis, for example, is about predicting the probability of a metastatic event. If the model that does that can also make biological sense, *interdisciplinary applications* are made easier by joining forces to understand the phenomenon. Indeed, assuming one can produce a model that, on top of being performant, makes it clear how much a given gene is involved in breast cancer outcome and in what way, specific biological experiments can be carried out with a higher chance of discovering important patterns. Deciding which variables are relevant is the focus of *feature selection* as we will develop in Section 1.2.

This being said, there is no denying that we might be facing particularly complex systems. For example, *systems biology* studies biological properties whose elaboration might just go beyond our understanding. Leo Breiman argues in Breiman (2001b) that, in many cases, *data models* are not necessarily the right choice. Data modeling consists in assuming a model (linear, logistic, ...) and then infer its parameters. They can be put in opposition to *algorithmic models* where the only assumption is that by some function f that we do not know a thing about, X becomes Y. On problems involving gene expression data, Breiman writes:

> *It requires that [the statisticians] focus on solving the problem instead of asking what data model they can create. The best solution could be an algorithmic model, or maybe a data model, or maybe a combination. But the trick to being a scientist is to*

be open to using a wide variety of tools.

He advertises for *black boxes* such as random forests (Breiman, 2001a), *Neural Networks* (Bishop, 2006) or *Support Vector Machines* (Vapnik, 1995), acknowledging that these algorithms are only slightly more interpretable than nature's black box itself and that the most accurate methods are the less interpretable ones. "The goal is not interpretability", he writes, "but accurate information".

This apparently leaves us with a choice: in an effort to provide both accurate predictions and some understanding to complex biological systems, we might have to choose between the two.

In this thesis, we *combine data and algorithmic models* in the sense of Breiman: starting with simple assumptions, e.g., with a parametric model, we find it necessary to introduce randomness. In particular, we will make intensive use of tools such as *bootstrap* (Efron, 1979) and *bagging* to build *Ensemble algorithms* for feature selection, as developed in Section 1.2.5. In the remaining, we will pay particular attention to investigating the interpretability/accuracy dilemma. In Section6.3, we re-discuss this issue in light of the results presented in the core of this document.

1.1.4 Penalized methods

In the previous section, we introduced the idea of constraining the objective function to a restrained set $\mathcal{H}$. From now on, we will consider objectives that depend on a linear combination of the covariates, i.e., $f(X) = X\omega$ where ω is a vector of weights in $\mathbb{R}^p$. Note that, e.g., the linear and the logistic models belong to this class of functions:

Linear $Y = X\omega + \varepsilon$, ε i.i.d. $\rightsquigarrow \mathcal{N}(0, \sigma^2)$

Logistic $\mathbb{P}(Y = 1|X) = \frac{\exp(-X\omega)}{1+\exp(-X\omega)}$

Constraining f now amounts to constraining ω, i.e, equation (1.3) becomes:

$$\begin{cases} \hat{\omega} = \arg\min_{\omega\in\mathbb{R}^p} \hat{R}_l(\omega) \\ \text{s.t. } \Omega(\omega) \leq T \end{cases} \tag{1.6}$$

where $\Omega : \mathbb{R}^p \to \mathbb{R}$ is a *penalty*. If $\hat{R}_l$ and Ω are both convex and under weak additional assumptions (see, e.g., Boyd and Vandenberghe, 2004, Section 5.2), (1.6) can be written in its Lagrangian form:

$$\hat{\omega} = \arg\min_{\omega\in\mathbb{R}^p} \hat{R}_l(\omega) + \lambda\Omega(\omega) \tag{1.7}$$

Both λ and T control the amount of regularization: when λ increases (resp. when T decreases), the weights ω are more penalized. When $\lambda = 0$ (resp. $T = +\infty$), the model is not penalized.

Depending on the problem, we may want to impose specific conditions on these weights, i.e., specific forms for Ω. In the following, we expose some of these conditions and try to give insight as to when they make sense. More detailed overviews of regularized methods can be found in, e.g., Huang et al. (2009), Bach et al. (2011a).

Enforcing smoothness: ℓ_2 regularization

A common example of a constrained problem is the Tikhonov regularization (Tikhonov, 1963), also known as *Ridge regression*: assume a classical linear setting $Y = X\omega + \varepsilon$. If the design of $\mathbf{X}$ makes it possible to invert $\mathbf{X}^T\mathbf{X}$, the *Ordinary Least Squares* (OLS) solution is:

$$\hat{\omega}_{OLS} = \arg\min_{\omega \in \mathbb{R}^p} ||\mathbf{Y} - \mathbf{X}\omega||_2^2 = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}. \tag{1.8}$$

This model, however, is not always an option, in particular when the number p of variables exceeds the number of examples n, making the matrix $\mathbf{X}^T\mathbf{X}$ singular. Ridge regression (Hoerl and Kennard, 1970) consists in adding the regularization function $\Omega(\omega) = ||\omega||_2^2$ to the empirical risk:

$$\hat{\omega}_{Ridge} = \arg\min_{\omega \in \mathbb{R}^p} ||\mathbf{Y} - \mathbf{X}\omega||_2^2 + \lambda||\omega||^2 = (\mathbf{X}^T\mathbf{X} + \lambda I_p)^{-1}\mathbf{X}^T\mathbf{Y}. \tag{1.9}$$

The first consequence is numerical: for any $\lambda > 0$, $\mathbf{X}^T\mathbf{X} + \lambda I_p$ becomes invertible, independently of the initial design. Second, and most importantly, it enforces the smoothness of the functional, in the sense of Lipschitz. Indeed, the larger λ, the smaller the Lipschitz constant $||\omega||_2$ in the following equation:

$$||f(x_1) - f(x_2)||_2 = ||x_1\omega - x_2\omega||_2 \leq ||\omega||_2||x_1 - x_2||_2. \tag{1.10}$$

Equation (1.10) is extremely valuable from the machine learning point of view: the smoothness of f seems to solve at least the first of the two dilemmas presented in Section 1.1.3. Indeed, when predicting the output y from a new point x, provided it is close - in the sense of the euclidian distance - to some point x_{tr} in the training set, we make sure that our prediction $\hat{y}$ will be close to y_{tr}. The variance of the prediction is therefore reduced, leaving us with only the bias to control. In a sense, we may also see the Ridge penalization as a help to the interpretability issue: because the Ridge solution is more *stable*, i.e., different training sets should lead to similar solutions provided they themselves are similar. It might thus be reasonable to assume that large values of $\hat{\omega}$ correspond to important covariates.

An interesting interpretation is provided by Bayesian statistics, that consider the penalty as a prior distribution on ω: assume that $p(\omega) = \mathcal{N}(0, \frac{1}{\lambda}I_p)$ and $p(Y|X, w) = \mathcal{N}(Xw, \sigma^2)$. Then maximizing the joint likelihood is equivalent to solving problem (1.9). Indeed, assuming $\sigma = 1$:

$$
\begin{aligned}
L(\mathbf{X}, \mathbf{Y}, \omega) &= \exp(-\frac{1}{2}||\mathbf{Y} - \mathbf{X}\omega||_2^2 - \frac{\lambda}{2}||\omega||_2^2) \\
&\propto \exp\left\{\frac{1}{2}(\omega - \Gamma_\lambda \mathbf{X}^T\mathbf{Y})^T\Gamma_\lambda^{-1}(\omega - \Gamma_\lambda \mathbf{X}^T\mathbf{Y})\right\}
\end{aligned}
$$

where $\Gamma_\lambda = (\mathbf{X}^T\mathbf{X} + \lambda I_p)^{-1}$. We thus recover the *posterior* distribution of ω: $\omega|\mathbf{X}, \mathbf{Y} \rightsquigarrow \mathcal{N}(\Gamma_\lambda \mathbf{X}^T\mathbf{Y}, \Gamma_\lambda)$. The maximum *a posteriori* (MAP) is therefore the mean $\Gamma_\lambda \mathbf{X}^T\mathbf{Y}$, corresponding to the Ridge solution (1.9). We can see now how increasing λ will yield an estimator with both a smaller mean and a smaller variance.

The ℓ_2 regularization is best known in the regression setting. However, it has also been studied in the classification setting (Hastie et al., 2009), and in particular in the context of support vector machines (Vapnik, 1995).

Enforcing sparsity: ℓ_1 regularization

Ridge regression constrains the amplitude of the weights in the linear model. However, it does not account for the possible *sparsity* of the data. In recent years, technological progresses have reversed some of the problems faced by the statistical community: it appears that acquiring information on a huge set of variables has been made easier than deciding which variables to experiment with. This seems especially true in bioinformatics, where the behavior of thousands of genes can be evaluated at a cost similar to that of just a few. This new situation makes our approach a little different as, in many cases, we can safely assume that *only some of the genes are relevant* to the problem at hand, the rest just being noise. Under these considerations, it seems only appropriate that the predictor be *sparse*, i.e., the goal is to return a vector ω with only a few non-zero entries relating to the relevant variables.

In this section, we introduce the earliest and still most popular method that enforces sparsity on the weights by penalizing their ℓ_1 norm, namely the Lasso regression.

Recall that the ℓ_1 norm is defined as

$$
||x||_1 = \sum_{j=1}^{p} |x_j|. \tag{1.11}
$$

The *Lasso regression* (Tibshirani, 1996) or *basis pursuit* (Chen et al., 1998) is done by regularizing the weights by $\Omega(\omega) = ||\omega||_1$:

$$
\hat{\omega}_{Lasso} = \arg\min_{\omega \in \mathbb{R}^p} ||\mathbf{Y} - \mathbf{X}\omega||_2^2 + \lambda||\omega||_1 \tag{1.12}
$$

Controlling the sum of absolute values will have a different effect than controlling the sum of squares (ℓ_2 norm). As shown in Figure 1.4, the ℓ_1 norm induces sparsity whereas the ℓ_2 norm only reduces the amplitude of the weights.

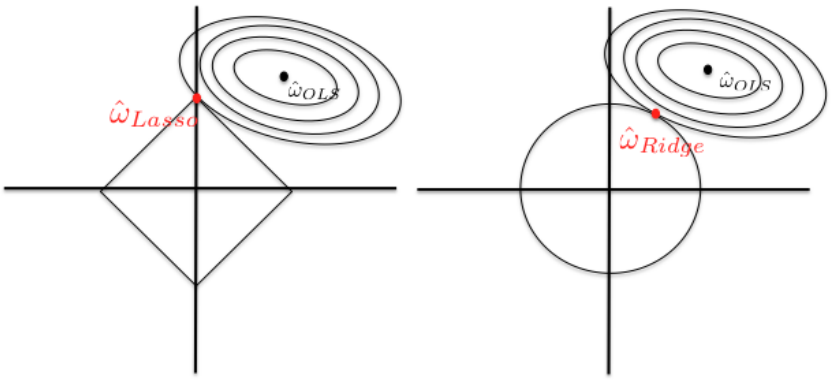


Figure 1.4: **Lasso and Ridge regressions.** In order to fit the constraint, the loss (here the square loss represented by its elliptical curves) approaches the boundaries of the ball. In the ℓ_1 penalization case, it will likely hit a singularity, forcing the solution to be sparse - in the picture, the first component of ω is zero. When we consider the ℓ_2 norm, neither component will be zero, almost surely.

Although Lasso was originally introduced as a regression method, penalization by the ℓ_1 norm has drawn a lot of attention in many communities. In particular, theoretical properties have been studied for the classification setting. This penalization can thus be used with other types of loss functions, including the logistic loss (Shevade and Keerthi, 2003, Koh et al., 2007).

In section 1.2.4, we will further discuss the Lasso as a *feature selection* method and will provide guidelines as to the choice of the penalty parameter λ.

Enforcing structured sparsity: ℓ_1, ℓ_2 and beyond

Lasso regularization appeared as a breakthrough to the statistical community. However, one main pitfall of the algorithm is its undesirable behavior when covariates are correlated. This problem is formally known as the *irrepresentable* condition (Zhao and Yu, 2006). Considering cases where the number of covariates is large - in particular, larger than the number of observations -, there is a high probability that this happens. This is particularly true in the applications that we address: not only are the genes suspected to be correlated, many happen to be biologically linked in a causal way.

When the Lasso is faced with such a correlated design, it will generally choose one covariate among the set of correlated ones and set the others to zero. Ran several times, it might even choose a different variable each time. The reason for that is given intuitively in the following result from Zou and Hastie (2005):

Suppose the extreme case where for some i and j in $\{1...p\}$, $X_i = X_j$ and consider the problem $\hat{\omega} = \arg\min_\omega ||\mathbf{Y} - \mathbf{X}\omega||_2^2 + \lambda\Omega(\omega)$. Then:

1. *If $\Omega(.)$ is strictly convex, $\hat{\omega}_i = \hat{\omega}_j$*

2. If $\Omega(\omega) = ||\omega||_1$*, then* $\hat{\omega}_i\hat{\omega}_j \geq 0$ *and* ω^* *defined by:*

$$\omega_k^* = \begin{cases} \hat{\omega}_k & \text{if} \quad k \neq i, k \neq j \\ s.(\hat{\omega}_i + \hat{\omega}_j) & \text{if} \quad k = i \\ (1-s).(\hat{\omega}_i + \hat{\omega}_j) & \text{if} \quad k = j \end{cases}$$

is also a solution to the problem with some $s \in [0, 1]$

This result emphasizes two important facts: first, the solution to the Lasso is *highly unstable*, especially in such a case as high correlation between some covariates. Second, it does not guarantee that similar covariates will receive similar weights. With this problematic in mind, Zou and Hastie (2005) proposed an improved version of the Lasso, named *Elastic Net*.

Managing correlated variables : the Elastic Net The Elastic Net appears as a compromise between the ℓ_1 and ℓ_2 regularizations and as such has the advantage of yielding a solution that is both sparse and smooth.

The problem to solve is as follows:

$$\omega^{enet} = \arg\min_{\omega \in \mathbb{R}^p} ||\mathbf{Y} - \mathbf{X}\omega||_2^2 + \lambda_1||\omega||_1 + \lambda_2||\omega||_2^2 \tag{1.13}$$

When $\lambda_2 > 0$ the penalty is strictly convex, ensuring the uniqueness of the solution. The following result from Zou and Hastie (2005) sheds light on the behavior of Elastic Net when the design exhibits some correlations:

Let $\rho = corr(X_i, X_j)$ *for some* $(i, j) \in \{1 \dots p\}^2$*. Assume* $\omega_i^{enet}\omega_j^{enet} > 0$*. Then,*

$$\frac{1}{||Y||_1}|\omega_i^{enet} - \omega_j^{enet}| \leq \frac{1}{\lambda_2}\sqrt{2(1-\rho)}$$

Note that when $X_i = X_j$, that is when $\rho = 1$, we do have $\hat{\omega}_i^{enet} = \hat{\omega}_j^{enet}$. Moreover, this result overall confirms the grouping effect of the Elastic Net: whereas the Lasso estimate would only select one of the covariates in a correlated group, the Elastic Net selects all of them.

The Elastic Net problem can in fact be seen as a Lasso-like problem. Indeed, it can be written as:

$$\omega^{enet} = \frac{1}{\sqrt{1+\lambda_2}}\omega^*$$

where ω^* is defined as:

$$\omega^* = \arg\min_{\omega} ||\mathbf{Y}^* - \mathbf{X}^*\omega||_2^2 + \frac{\lambda_1}{\sqrt{1+\lambda_2}}||\omega||_1$$

with $\mathbf{X}^* = \frac{1}{\sqrt{1+\lambda_2}} \begin{pmatrix} \mathbf{X} \\ \sqrt{\lambda_2} I_p \end{pmatrix}$ a matrix of size $(n+p) \times p$ and $\mathbf{Y}^* = \begin{pmatrix} \mathbf{Y} \\ 0 \end{pmatrix}$ a vector of size $n + p$.

Thus, the Elastic Net estimate can be computed using an algorithm that solves the Lasso. However, the estimate as described here might enjoy some performance improvements. Indeed, the authors note that, as empirical evidence confirms, the Elastic Net procedure happens in two steps, each of which introduces unnecessary bias and variance. A way to avoid this is to rescale the estimator as:

$$\omega_{improved}^{enet} = \sqrt{1+\lambda_2}\omega^* = (1+\lambda_2)\omega^{enet}$$

Adding prior knowledge on grouping: the Group Lasso Given particular problems, one might wish to perform a Lasso-like selection among *predefined groups of variables*, as opposed to individual features (Lasso) or correlation-built groups (Elastic Net). For instance, a prior knowledge on the interaction of some genes might lead one to consider them together.

In Yuan and Lin (2006), the authors propose an extension to *factor selection*, introducing a generalization of the Lasso they call *Group Lasso*.

Consider the following regression problem:

$$Y = \sum_{g=1}^{G} \mathbf{X}_g \omega_g + \varepsilon \tag{1.14}$$

where $\mathbf{X}_g$ is of size $n \times p_g$ and ω_g of size $p_g \times 1$ for all $g = 1...G$ such that $\sum_{g=1}^{G} p_g = p$. Each $\mathbf{X}_g$ now represents a group g of p_g variables. Note that these groups do not overlap.

Note that the particular case $p_g = 1$ for all g, (1.14) we are back to the usual setting.

The Group Lasso stands as a generalization of the Lasso procedure considering *block penalization* of these grouped variables:

$$\omega^{GLasso} = \arg\max_{\omega} ||\mathbf{Y} - \sum_{g=1}^{G} \mathbf{X}_g \omega_g||^2 + \lambda \sum_{g=1}^{G} (\omega_g^T \mathbf{K}_g \omega_g)^{1/2} \tag{1.15}$$

where $\mathbf{K}_g$ is a $p_g \times p_g$ matrix representing some relation between the variables in group g.

This penalty enforces sparsity at the *group level*: some $||\omega_g||$ will be set to zero, forcing all variables in g to have a zero weight as well.

Not unlike the Elastic net, the penalty $\Omega(w) = \sum_{g=1}^{G} (\omega_g^T \mathbf{K}_g \omega_g)^{1/2}$ stands as a compromise between the ℓ_1 and the ℓ_2 norms, the major difference being that here the groups are predefined.

The authors note that one should be careful with the choice of $\mathbf{K}_.$. They choose to set them to $\mathbf{K}_g = p_g I_{p_g}$ in order to balance the (groups of) parameters according to their dimension without giving one coordinate any special importance within a group. However one could easily imagine choosing these matrices according to some prior knowledge on the variables. In particular, the off-diagonal terms might be chosen according to the correlation between some covariates within a group.

Note that the Elastic Net and the Group Lasso penalties are special cases of ℓ_1/ℓ_2 regularization and were discussed as such in Jenatton et al. (2009) where the authors derive a general formulation of Ω for this type of regularization.

Managing overlapping groups: the overlapping Group Lasso It is common, in real world examples such as molecular biology, that some variables (genes) belong to several groups. This might be because one gene has several functions (see Sections 2 and 3) or because groups of genes are influenced by the same *transcription factors*.

While the Group Lasso as described in the previous section does not take this possibility into account, the *Overlapping Group Lasso* does. Two such models have been proposed in Jenatton et al. (2009) and Jacob et al. (2009). We focus here on the latter where the authors provide a penalty function that keeps the model sparse at the group level while allowing some variables to belong to both active and inactive groups. This is made possible by considering a different *support* for the model: $supp(\omega) = \left(\bigcup_{g:\omega_g=0} g\right)^c$ where g is meant as a group index, i.e., $g \subset \{1...p\}$ and thus ω_g as a vector of dimension $|g|$. Hence the support is such that some variables are duplicated, e.g., if variable i appears in groups g_1, g_2 and g_3, we consider it three times instead of just one allowing for instance g_1 to be active while g_2 and g_3 remain inactive.

More precisely, if $\mathcal{G}$ is the set of all predefined groups, the penalty considered here can be written:

$$\Omega(\omega) = \inf_{v \in \mathcal{V}_\mathcal{G}, \sum_{g\in\mathcal{G}} v_g = \omega} \sum_{g\in\mathcal{G}} ||v_\mathcal{G}||_2$$

where $\mathcal{V}_\mathcal{G}$ is simply the set of vectors $v = (v_g)_g$ that are such that $supp(v_g) \subset g$. Thus the penalty function takes **group** g into account only once but the **variable** i as many times as the number of groups it belongs to.

We thus seek ω^{OLasso} such that

$$\omega^{OLasso} = \arg\min_{\omega} \sum_{i=1^n} l(y_i, x_i\omega) + \lambda\Omega(\omega) \tag{1.16}$$

The implementation of the algorithm with the logistic loss was adapted from Meier et al. (2008) that uses block coordinate gradient descent and solves the *Logistic Group Lasso*. An interesting application of this norm is its little brother *Graph Lasso*, also presented in Jacob et al. (2009): when features can be represented on a graph, Graph Lasso amounts to Overlapping Group Lasso where the groups consist of connected components of the graph. In Chapter 3, we study the employment of Graph Lasso for genomic signature selection.

Considering all possible groups : the k-support norm When the grouping structure is not known, but we still suspect that some variables should receive a similar treatment, the following is an alternative to the Elastic Net.

Recently, Argyriou et al. (2012) have investigated the k-support - or k-overlap - norm, that is interestingly related to both the Elastic Net and the Overlapping Group Lasso. It is defined as:

$$\Omega_k^{sp}(\omega) = \left(\sum_{i=1}^{k-r-1} (|w|_i^{\downarrow})^2 + \frac{1}{r+1} \left(\sum_{i=k-r}^{p} |w|_i^{\downarrow} \right)^2 \right)^{1/2}$$

where r is the only integer in $\{0, \ldots, k-1\}$ verifying

$$|w|_{k-r-1}^{\downarrow} > \frac{1}{r+1} \sum_{i=k-r}^{p} |w|_i^{\downarrow} \geq |w|_{k-r}^{\downarrow}$$

and $u_i^{\downarrow}$ is the i-th largest element of u with the convention $|u|_0^{\downarrow} = +\infty$.

As $\Omega_1^{sp}(.) = ||.||_1$ and $\Omega_p^{sp}(.) = ||.||_2$, the k-overlap norm can be viewed as a trade-off between Lasso and Ridge penalties, similarly to the Elastic Net.

It can also be seen as the Overlapping Group Lasso with all groups of size up to k as input, as argued in Argyriou:

$$\Omega_k^{sp}(\omega) = \min \left\{ \sum_{I \in \mathcal{G}_k} ||v_I||_2 : supp(v_I) \subseteq I, \sum_{I \in \mathcal{G}_k} v_I = w \right\}$$

Actually, we note that it is equivalent to considering all groups of size exactly k (see Chapter 4).

A fast proximal algorithm (see, e.g., Beck and Teboulle, 2009, Jenatton et al., 2011) makes it possible to compute the solution efficiently and without enumerating all possible groups. In Section 4, we apply it to the breast cancer prognosis problem and investigate the idea of group selection without prior knowledge on the grouping structure.

Other sparsity-inducing penalties Among other sparsity-inducing penalties, we note the *fused Lasso* penalty (Tibshirani et al., 2005), that constrains variables close to each other in some sense to have similar weights, in a pairwise fashion. The fused penalty was adapted for support vector machines and applied to classification of CGH arrays in Rapaport et al. (2008).

Several works relate to group Lasso variants, e.g. Obozinski et al. (2008), where a special case of Group Lasso is investigated in the context of multitask learning, Zhao et al. (2009), who propose a penalty using several norms to account for structures that are both grouped and hierarchical and Jenatton et al. (2009) for a general ℓ_1/ℓ_2 regularization.

1.2 Feature selection and feature ranking

We now turn to the principal focus of this thesis, namely *variable* or *feature selection*. In this section, we will provide some background on the methods we consider in this thesis, that make it possible to *select relevant variables* only from a possibly immense set of available variables. We also explain how feature selection can be performed through *feature ranking*.

Formally, we are looking for a subset $\mathcal{A} \subseteq \{1...p\}$ known as the *active set*. When adequate, we will note $\mathcal{A}_k$ the set containing k active variables.

1.2.1 Overview

A common argument for feature selection is a *reduced computation time*. Limiting the number of features amounts, to some, to preprocessing the data. However, this would not explain why some of us are willing to imagine even more computationally demanding algorithms to perform the selection. Indeed, we pursue two more goals when looking for this relevant subset:

1. **overcome the *curse of dimensionality***. The notion was first introduced in Bellman and Kalaba (1959) in a very intuitive way: considering a hypercube in p dimensions and a grid with equal spaces of size $1/10$, the hypercube contains 10^p points. If $p = 10$, we are looking at 10^{10} points. Now if $p = 20$, we actually consider 10^{20} points. For a classification or regression problem where n is the minimum number of examples necessary in 1 dimension, extending the problem to p dimensions would require n^p examples. However, we are not given more examples and usually have to carry on with $\mathcal{O}(n)$ of them. The main - and worse - consequence of this is overfitting (see Section 1.1.3): because we have much less points than necessary, it is easy to find patterns linking the data to the response, but the chance is high that they be *accidental* and therefore not at all generalizable. As a matter of fact, we are assuming that the data behaves as shown in Figure 1.5, i.e., that there is an optimal number of features maximizing the predictive accuracy.

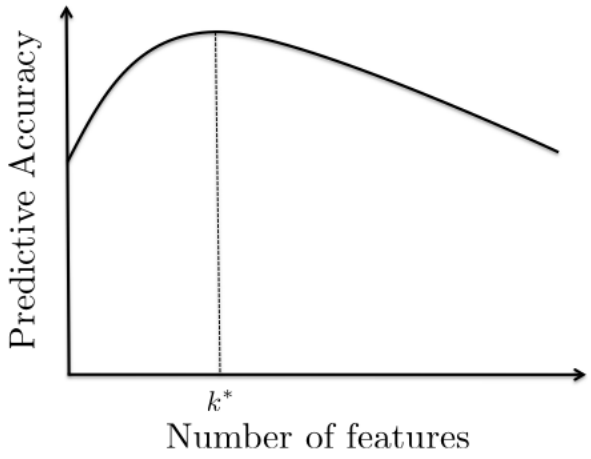


Figure 1.5: **Feature selection and the curse of dimensionality.** One important reason for performing feature selection is the belief that there exists an optimal number of features with respect to the predictive accuracy.

2. **increase interpretability**. As stated in Section 1.1.3, one might want to work with an interpretable model. The complexity of the model itself being equal, reducing the number

of features will certainly make the output less obscure. Moreover, the list of relevant features can be used independently form the predictions. Taking as an example the breast cancer prognosis problem (Section 2), if we are able to extract a relevant set of genes, i.e., a list that, when combined with a classifier, produces a good predictive accuracy, this list can be tested biologically to assess the actual involvement of the genes.

Assuming unlimited computational resources, one could decide for the *best subset selection* algorithm (Furnival and Wilson, 1974, Narendra and Fukunaga, 1977, Hastie et al., 2009), also known as ℓ_0 penalization, that consists in testing each possible subset among the p features. However, the computational complexity of this algorithm is of the order of $\mathcal{O}(2^p)$, making it intractable for problems such as gene selection, where p is usually of the order of some thousands.

There is thus a need to consider less demanding methods. In this section, we present such methods and explain how feature selection can be performed through *feature ranking*. The latter consists in assigning a score to each variable, that represents its *relevance*, yielding a ranked list of variables: the higher the score, the better ranked the variable. The list can then be *thresholded*, returning a subset of features.

We start by describing the three types of methods common to many taxonomies (Saeys et al., 2007) : filter, wrapper and embedded methods. We then detail a few algorithms that perform *feature selection* at the group level, i.e., that choose groups of variables instead of singletons. Finally, we introduce *Ensemble methods*, which we make intensive use of in the core of this thesis.

1.2.2 Filter methods

Univariate filter methods are extremely easy to implement and fast to run. Whether they are parametric or non-parametric, they rely on the same principle: each variable j, $j = 1...p$, is compared to the response, independently of the others, so as to attribute it a *score* s_j, i.e., some type of correlation measure. Features are then *ranked* in a decreasing order and this list is thresholded according to some criterion, such as a given number of features or a significance measure (Zhang and Rajapakse, 2008). The computational complexity of such algorithms is therefore $\mathcal{O}(p)$.

In many applications, and particularly in bioinformatics, it is expected that features work in groups. The downside of *univariate* filters is that they do not take this *structure* into account.

Numerous filter methods exist - including multivariate methods, which we do not describe here. We only present some of them and refer the interested reader to Lazar et al. (2012) for an exhaustive review.

We will focus on the classification setting where $Y \in \{-1, +1\}$. For each feature $j = 1...p$, we write $X_j^- = \{x_{i,j}, y_i = -1, i = 1...n\}$ and $X_j^+ = \{x_{i,j}, y_i = +1, i = 1...n\}$. Moreover, we note $n_j^+ = \sharp\{i : y_i = +1\}$ and $n_j^- = \sharp\{i : y_i = -1\}$.

Among the four methods that we describe below, three (t-test, entropy, Bhattacharyya distance) are *parametric* methods. They assume a gaussian mixture model on the features, i.e., for $j = 1...p$,

$$X_j^+ \rightsquigarrow \mathcal{N}(m_j^+, \sigma_j^+) \text{ and } X_j^- \rightsquigarrow \mathcal{N}(m_j^-, \sigma_j^-).$$

When the sample size is large enough ($n_j^+, n_j^- > 30$ in practice), and the covariates are i.i.d. from any distribution, we can assume that the means of X_j^+ and X_j^- are normally distributed. However, it is worth noting that often, parametric filtering is used without the data verifying these assumptions and, as we show in Haury et al. (2011), with quite some success.

The last method, namely *Wilcoxon sum-rank statistic*, is non-parametric and preferred by some for this reason.

Student's t-test

Student's t-test is a very popular *parametric filter* method. The null hypothesis states that $m_j^+ = m_j^-$ and the test statistic is defined as

$$s_j = \frac{\overline{X}_j^+ - \overline{X}_j^-}{\sqrt{\frac{(\hat{\sigma}_j^+)^2}{n_j^+} + \frac{(\hat{\sigma}_j^-)^2}{n_j^-}}}$$

where $\overline{x}$ and $\hat{\sigma}$ are respectively the empirical mean and standard error of x.

When $\sigma^+ = \sigma^-$, the quantity s_j follows a Student distribution. When variances are unequal, a Welch t-test is used where the degrees of freedom ν are approximated by the Welch-Satterthwaite equation:

$$\nu = \frac{\left(\frac{(\hat{\sigma}_j^+)^2}{n_j^+} + \frac{(\hat{\sigma}_j^-)^2}{n_j^-}\right)^2}{\frac{\left(\frac{(\hat{\sigma}_j^+)^2}{n_j^+}\right)^2}{n_+ - 1} + \frac{\left(\frac{(\hat{\sigma}_j^-)^2}{n_j^-}\right)^2}{n_- - 1}}$$

A p-value can then be computed representing the probability that $m_j^+ = m_j^-$. The smaller the p-value, the smaller the risk we take when deciding that the feature is relevant. A threshold can thus be set on the p-value, e.g., we keep the genes with a p-value smaller than 5%. However, choosing this kind of threshold i) implies the correction of the p-values (Benjamini and Hochberg, 1995) and ii) may lead to lists of different sizes, which can make it difficult to compare methods. In Section 2, we propose to threshold to a given number of genes instead.

Maximum entropy

In its non parametric form, *maximum entropy*, or *Kullback-Leibler divergence* is defined as follows:

$$s_j = \int \ln\left(\frac{P(X_j^+)}{P(X_j^-)}\right) dP(X_j^+) + \int \ln\left(\frac{P(X_j^-)}{P(X_j^+)}\right) dP(X_j^-)$$

It measures a *distance* between the two distributions. The less similar $P(X_j^+)$ and $P(X_j^-)$, the larger the divergence.

Since we assume gaussian distributions on X_j^+ and X_j^-, the entropy becomes:

$$s_j = \frac{1}{2}\left[\left(\frac{\hat{\sigma}_j^+}{\hat{\sigma}_j^-}\right)^2 + \left(\frac{\hat{\sigma}_j^-}{\hat{\sigma}_j^+}\right)^2 + (m_j^+ - m_j^-)^2\left(\left(\frac{1}{\sigma_j^+}\right)^2 + \left(\frac{1}{\sigma_j^-}\right)^2\right) - 2\right]$$

Features are then ranked by decreasing divergence.

Bhattacharyya distance

Similarly to the Kullback-Leibler divergence, we now consider a different distance between the two distributions. The *Battacharyya coefficient*(Guorong et al., 1996) is computed as

$$s_j = -\ln\left(\int \sqrt{P(X_j^+)P(X_j^-)}dx\right)$$

which, under the gaussian assumption, reduces to

$$s_j = \frac{1}{8}\frac{(m_j^+ - m_j^-)^2}{\hat{\sigma}_j^2} + \frac{1}{2}\ln\left(\frac{\hat{\sigma}_j^2}{\sqrt{\hat{\sigma}_j^+\hat{\sigma}_j^-}}\right)$$

where $\hat{\sigma}_j^2 = \frac{(\hat{\sigma}_j^+)^2 + (\hat{\sigma}_j^-)^2}{2}$.

Wilcoxon rank-sum test

Another statistical test that can be used as a filter is the *Wilcoxon rank-sum test*, also known as *Mann-Whitney U-test* and whose statistic is equivalent the the *Area under the ROC curve metric*. It differs from the previous methods in that it is non-parametric, i.e., it does not assume any kind of distribution on the data. The null hypothesis is defined as: $\mathcal{H}_0 : P(X_j^+ > X_j^-) = 0.5$.

For each feature j, the n examples $x_{1,j}, x_{2,j}, ..., x_{n,j}$ are sorted by increasing value so as to assign a rank $R_{i,j}$ to each of them. These ranks are then simply summed for the smallest class, as follows (Wilcoxon, 1945):

$$t_j = \sum_{i:y_i=+1} R_{i,j}.$$

The intuition is to establish whether the ranks of the positive samples are distributed similarly to the ones of the negative samples. This is done by computing:

$$s_j = n_j^+ n_j^- + \frac{n_j^+(n_j^+ - 1)}{2} - t_j$$

which is compared to the Mann-Whitney table values. The distribution of s_j is derived in Mann and Whitney (1947), where the authors use combinatorial probabilities to compute the chances of a sequence showing a given number of $-1/+1$ labels in a row.

1.2.3 Wrapper methods

We now turn to *wrappers*. This class of methods operates over the entire set of features. Using a loss function (see Section 1.1.2) to minimize, they sequentially add or remove variables that optimize this criterion. As a basis for a wrapper method, one may use any type of predictor f and loss function l. The first type of wrappers is known as *forward selection* because it operates by adding features to the active set, one by one. The second type, named *backward elimination*, starts with the entire set of features and removes features one by one.

There are several ways of using these algorithms. *Model selection* criteria can be applied, such as AIC (Akaike, 1973) or BIC (Schwarz, 1978) to decide for the optimal subset size. The optimal subset can also be chosen by estimating the prediction error through *cross-validation* (see Section 1.3.2). Finally, wrappers can be viewed as feature ranking methods as they return *nested subsets* of variables of increasing sizes that can easily be merged into a single ranked list. It is this ranking approach that we consider in the remaining.

Greedy Forward Selection

Starting from the empty subset: $\mathcal{A}_0 = \emptyset$, *stepwise forward selection* consists in *adding one variable at a time* to the active set $\mathcal{A}$ as described in Algorithm 1.

Input: Data $(\mathbf{X}, \mathbf{Y})$ divided into k cross-validation training and test sets, loss function l, predictor f.
Output: Ranked list of the p features.
Initialize: $\mathcal{A}_0 = \emptyset$;
for $t = 1$ *to* p **do**
  $\mathcal{S}_t = \overline{\mathcal{A}_{t-1}}$;
  $r_t = \arg\min_{j \in \mathcal{S}_t} CV_k(l, \mathbf{Y}, f(\mathbf{X}_{j \cup \mathcal{A}_{t-1}}))$;
  $\mathcal{A}_t = \mathcal{A}_{t-1} \cup \{r_t\}$;
end
return r

Algorithm 1: Greedy forward selection

The function CV_k in Algorithm 1 refers to the *k-fold cross-validation* procedure, that we detail in Section 1.3.2.

Assuming that the complexity of CV_k is kc_t at step t, the overall complexity of forward selection to rank all p variables is of the order of $k \sum_{j=1}^{p} c_j(p - j + 1)$ and can be reduced to $k \sum_{j=1}^{q} c_j(p - j + 1)$ if it is run until only q variables are chosen.

This algorithm is also called *Greedy Forward Selection* (GFS) in that, at each step, it chooses the variables that best fits the data and adds it to the current active set as opposed to reconsidering the active set in light of new information, as done by, e.g., exhaustive search, which is, however, a much more expensive algorithm.

Variants of this algorithm include *forward stagewise* selection and Least Angle Regression (Efron et al., 2004). As a modification of the latter, Lasso regression (see Section 1.1.4) is closely related to forward selection.

Backward Elimination

Backward elimination starts with the entire set, i.e., $\mathcal{A}_0 = \{1...p\}$ and removes features sequentially. The idea is essentially to remove the feature or the set of features that least contribute to the current classifier. The most popular method based on this approach is known as *Support Vector Machine Recursive Feature Elimination* (SVM RFE) and was proposed in Guyon et al. (2002). It consists in running a Support Vector Machine (Vapnik, 1998) at each step and removing the feature that exhibits the smallest weight. We describe it in Algorithm 2.

Input: Data $(\mathbf{X}, \mathbf{Y})$, penalization parameter C.
Output: Ranked list of the p features.
Initialize: $\mathcal{A}_0 = \{1...p\}$;
for $t = 1$ *to* p **do**
  $w = SVM(X_{\mathcal{A}_{t-1}}, Y)$;
  $r_t = \arg\min_{j \in \mathcal{A}_{t-1}} w_j^2$;
  $\mathcal{A}_t = \mathcal{A}_{t-1} \setminus \{r_t\}$;
end
$r = flip(r)$;
return r

Algorithm 2: SVM RFE

SVM RFE thus returns a vector r of ranked features. Note that it is flipped in the end because the ranking is obtained backwards. Another version of the algorithm consists in removing entire groups of features at each step.

1.2.4 Embedded methods

Finally, we will consider in this thesis *embedded methods*. They are similar to wrappers and in particular to forward selection. However, they are not as greedy and they restrict the space of the function to find, or - said differently - *guide* the learning process. Among them, we are particularly interested in ℓ_1-penalized methods that we introduced in Section 1.1.4 as well as ℓ_1/ℓ_2 regularized algorithms, that allow to take into account the *structure* of the data. Finally, we will introduce the popular *random forests* algorithm.

In this section, we focus on the ways to transform embedded methods from *subset selection* algorithms into *feature ranking* algorithms.

The Lasso

Lasso (see Tibshirani (1996) and Section 1.1.4) as a feature selection method has been extensively discussed. One main issue remains the choice of the penalty parameter λ. A useful tool is the *regularization path*: across different values of λ within a grid, the values of $\hat{\omega}$ describe a piecewise linear function, as shown in Efron et al. (2004), Rosset and Zhu (2007). Such a path is shown on Figure 1.6. This has two interesting consequences: first, it can make the computation faster, as the weights only have to be evaluated at some points; second, it makes it easier to estimate the optimal value of λ through cross-validation.

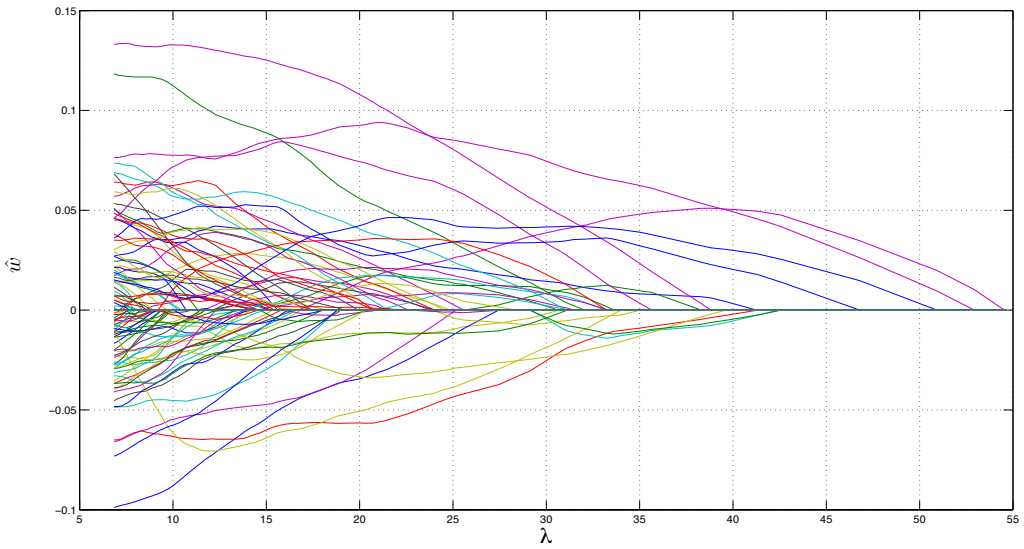


Figure 1.6: **Lasso Regularization Path:** piecewise linearity of the regularization path of Lasso. The regularization parameter λ is shown on the x-axis and the values of $\hat{\omega}$ on the y-axis.

Different types of algorithms exist that can solve the Lasso. Active set methods (Osborne et al., 1999, Efron et al., 2004) perform the regularization in a forward stepwise way: at each step, only one variable is added to (or removed from) the current active set, following the regularization path. Proximal gradient methods (Beck and Teboulle, 2009, Nesterov and Nemirovsky, 1994), on the other hand, do not take advantage of the piecewise linearity of the path: for a given value of λ, the algorithm computes the projection of the gradient onto the ℓ_1 ball. In practice, this projection can be computed using a closed form, named *soft thresholding*, which makes this class of algorithms very performant. A review and empirical comparison of optimization algorithms for Lasso can be read in Yang et al. (2010).

At each step of the *active set algorithm*, one variable either enters or quits the path. Notice the *kinks* on Figure 1.6: it describes the phenomenon of one variable exiting the path to be replaced, at the next step, by a different one that explains the data better. According to Mairal (2010), most designs should not exhibit too many of these kinks, that make both the computation

and the interpretation harder. However, he notes that in extreme correlation cases, it is possible to observe a great number of them.

From this path, we now have to either choose the *best subset* of variables or *rank* them:

Subset selection Run internal cross-validation (see Section 1.3.2) on the λ-grid, choose the value of the parameter that yields the best performance and run the Lasso one more time using this value on the entire training set. This way should ensure a good generalization performance. However, a major drawback is that the number of selected variables cannot be controlled and will likely be different from one dataset to another, with the same value of λ.

Variable ranking Fix a number k of variables to be selected, stop the Lasso at the value of λ for which this number is achieved. Score the k variables according to the value of λ for which they first entered the path or by decreasing absolute value of their weights for the smallest λ.

In the remaining of this thesis, we care for variable ranking and therefore choose the second option. The ranking methods we use are detailed in Chapters 2, 3, 4 and 5.

Note that we discuss methods implying randomization that do not depend on λ in Section 1.2.5, e.g., Stability Selection.

The Elastic Net

We noted in Section 1.1.4 how the Elastic could be viewed as just another Lasso algorithm. Therefore, variable selection through the Elastic Net is done just as described in the previous section.

Random Forests

Also belonging to the class of embedded methods is the extremely popular *Random Forests* algorithm (Breiman, 2001a). Initially, the algorithm was meant for prediction problems as opposed to feature selection. This ensemble algorithm draws subsamples from the data, using *bootstrap* (see Efron (1979) and Section 1.2.5). For each subsample, it trains a tree (Breiman et al., 1984) of given depth, randomly selecting candidate variables at each node and finally aggregates all trees to form the final prediction. It turns out (Breiman, 2001b) that Random Forests can also be used in the feature selection context: *variable importances* can be derived, that represent how much each variable contributed to the final prediction model.

Strobl et al. (2007) review different ways of computing these importances. Formally, for each tree $\mathcal{T}$ in the forest, define $(\mathcal{N}_k)_{k=1\dots|\mathcal{T}|}$ as the nodes of $\mathcal{T}$. The *naive* way to measure variable

importance is to count the number of times each variable was selected for a split:

$$VI(x_j) = \frac{1}{ntree} \sum_{\mathcal{T}} \sum_{k=1}^{|\mathcal{T}|} f(\mathcal{N}_k, x_j),$$

where $f(\mathcal{N}_k, x_j) = 1$ if x_j is selected at node k and 0 otherwise. A more elaborate method is to weigh each node $\mathcal{N}$ by the amount of *uncertainty* it was able to eliminate. Uncertainty measures are computed over all samples. Examples include Shannon entropy and Gini index for classification problems and the variance of the sample for regression problems. Writing $I(\mathcal{N})$ the uncertainty reduction at node $\mathcal{N}$, the weighted variable importance is defined as:

$$VI(x_j) = \frac{1}{ntree} \sum_{\mathcal{T}} \sum_{k=1}^{|\mathcal{T}|} I(\mathcal{N}_k) f(\mathcal{N}_k, x_j).$$

Lastly, one can compute the permutation importance:

$$VI(x_j) = \frac{1}{ntree} \sum_{\mathcal{T}} (err\tilde{OOB}^j_{\mathcal{T}} - errOOB_{\mathcal{T}}),$$

where $errOOB_{\mathcal{T}}$ is the out-of-bag error on tree $\mathcal{T}$ and $err\tilde{OOB}^j_{\mathcal{T}}$ is the OOB error on tree $\mathcal{T}$ after permutation of variable x_j in the OOB sample.

Once the importances are known, variables can be ranked according to this scoring and this list can be thresholded to the first k (Breiman, 2001a). The variable importance measure can also be used in a recursive feature elimination setting (see Section 1.2.3) as proposed in, e.g. Díaz-Uriarte and De Andres (2006). The filter and wrapper approaches have also been combined in Genuer et al. (2010).

In this thesis, we do not directly use Random Forests. However, we found it necessary to introduce them as the algorithm presented in Chapter 4 is inspired by this method. Moreover, the method we compare TIGRESS to in Chapter 5 is based on Random Forests.

Group Feature Selection

We now discuss how to perform feature selection using embedded methods enforcing group selection, e.g. Group Lasso, Overlapping Group Lasso and k-support norm.

These algorithms also return a sparse vector of weights and their regularization path can be computed. However, since groups of variables are added at once, it is possible to score the variables using the values of λ, but some of them will have the same score. Variables can thus be scored by decreasing absolute value of their weights for a given value of the regularization parameter. See Chapters 3 and 4 for details on the ranking procedures we adopt.

1.2.5 Ensemble feature selection

In a high-dimensional context, feature selection methods tend to suffer from a *lack of stability* (see, e.g., Ein-Dor et al., 2005, Michiels et al., 2005, Haury et al., 2011): when the training set

changes slightly, the selected variables might be very different. In order to overcome this pitfall, one solution consists in trying to *stabilize* the list of variables.

One idea is to use bootstrap and resampling : instead of performing feature selection on the entire training set, run the selection method B times, each time on a different subsample drawn with (bootstrap) or without (resampling) replacement. This yields B ranked lists. We assume that the ranked lists are of size p, i.e., that all variables are ranked, from most to least relevant, according to some score $(s_{j,b})_{j=1...p,b=1...B}$. Assume further that $r_{j,b}$ denotes the rank of variable j i the list b. The question remains as to how to aggregate the B lists into one final ranked list. This issue is discussed in, e.g., Boulesteix and Slawski (2009), Abeel et al. (2010), Haury et al. (2011), Meinshausen and Bühlmann (2010). Below is a summary of the aggregation functions that can be used:

- **Methods based on rank averaging**: here we only take into account how the variables were ranked, i.e., the scores s_j are not taken into account further than to compute the ranks. The final list is computed by sorting the following score S_j decreasingly. On Figure 1.7, we show how these methods behave with respect to the ranks.
  - **Averaging the ranks:** for each variable, its ranks are averaged over the B lists, yielding a final ranking $S_j = 1 - \frac{1}{B}\sum_{b=1}^{B} \frac{r_{j,b}}{p}$.
  - **Averaging a decreasing function of the ranks:** in order to give more weight to the variables well ranked in each list, it might be interesting to average a decreasing function of the ranks, such as $S_j = \frac{1}{B}\sum_{b=1}^{B} \exp(-\gamma r_{j,b})$, where γ controls how fast the function decreases.
  - **Stability selection:** this method can be seen as a discontinuous version of the previous one. We compute the frequency of the number of times variable j is chosen among the first k variables: $S_j = \frac{1}{B}\sum_{b=1}^{B} \delta(r_{j,b} \leq k)$.
- **Methods based on score averaging**: the first set of methods does not pay attention to how the scores are distributed. Here we take this information into account. The final list is obtained by sorting the aggregated scores decreasingly.
  - **Averaging the scores:** here, S_j is computed as the average of all B scores for variable j: $S_j = \frac{1}{B}\sum_{b=1}^{B} s_{j,b}$. We give more weight to a variable that has received consistently high scoring.
  - **Averaging a decreasing function of the scores:** similarly as with the ranks, we are giving more weight to variables with a high score: $S_j = \frac{1}{B}\sum_{b=1}^{B} \exp(-\gamma s_{j,b})$.
  - **Stability selection on the scores:** low scores are completely ignored. This is similar to thresholding a list according to p-values: $S_j = \frac{1}{B}\sum_{b=1}^{B} \delta(s_{j,b} \leq s_{thr})$. s_{the} is the threshold. In the case of a filter method where the scores are usually the p-values related to the statistical test, s_{thr} can be set to, e.g. 5%.

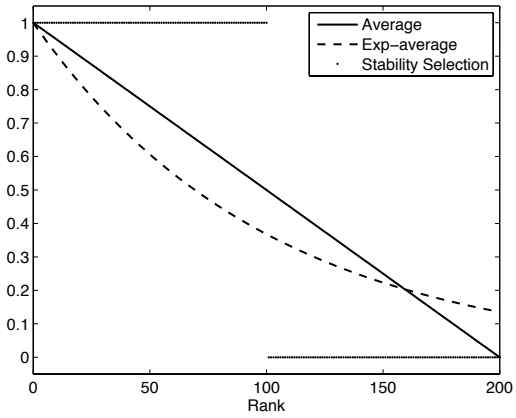


Figure 1.7: **Aggregation methods:** averaging, exponential averaging and stability selection with respect to the rank with $p = 200$, $k = 100$ and $\gamma = 1/100$.

There are, of course, other ways to perform Ensemble feature selection. Instead of bootstrapping the variables, one way is to run several feature selection methods on the same data and to aggregate the lists across the scores or rank produced by these methods. A combination of the two would be to use bootstrap on the samples as well as several selection methods. We do not further discuss these ideas, as we are more interested in how one particular method performs. However, research shows that, in general, wisdom of crowds usually gives good results (Marbach et al., 2012).

On the other hand, we do pay particular attention to a different kind of resampling that involves not only the samples but also the variables. In Meinshausen and Bühlmann (2010), the authors introduce the *randomized Lasso*: at each run b, bootstrap is performed on the samples and, at the same time, a vector $W \in \mathbb{R}^p$ is drawn from a uniform distribution $\mathcal{U}_{[\alpha,1]}$ where $\alpha > 0$. Each variable j is weighted by W_j, and the selection is performed on $\tilde{\mathbf{X}}$ where $\tilde{X}_j = W_{j,b} X_j$. This procedure artificially conditions the chances of variable j being selected during the run, i.e., a variable weighted by a small W_j will have fewer chances of selection. α controls the amount of randomization: the smaller α, the more randomized the variables. In Chapter 4, we take this idea further by strictly preselecting some of the variables for each run. Note that this class of methods are not useful in the context of univariate selection procedures, as the weighting or the preselection does not at all affect their scoring.

1.3 Evaluating and comparing

As developed in section 1.1.3, what we should aim for when training a predictor is a good *generalization ability*, i.e., a good performance on new data. In this section, we describe ways to measure this performance and explain how validation and testing should be undertaken.

1.3.1 Accuracy measures

Depending on the context (regression or classification), several performance measures can be computed. We write $\hat{y}_i$ the value of the prediction for individual i and y_i the true response.

In the regression setting, we mostly use the Mean Squared Error (MSE) : $\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$. In the classification setting, where $y \in \{-1, +1\}$, there are several indicators at disposal. Most classifier return continuous values between, e.g. $x\hat{\omega}$. The prediction $\hat{y}$ is computed by thresholding this value at θ, usually equal to 0. When $x\hat{\omega}$ is greater than θ, $\hat{y} = 1$ and $\hat{y} = -1$ otherwise.

In order to formally describe them, we use the definitions in Table 1.1.

		y		
		-1	$+1$	
$\hat{y}$	-1	TN	FN	$\hat{n}_-$
	$+1$	FP	TP	$\hat{n}_+$
		n_-	n_+	n

Table 1.1: **Classification setting:** four necessary measures. TN : True Negatives; FN: False Negatives; FP: False Positives; TP: True Positives.

- Good classification rate/accuracy: $\frac{TN+TP}{n}$
- Sensitivity/True Positive Rate (TPR)/Recall: $\frac{TP}{n_+}$
- Specificity/True Negative Rate (TNR): $\frac{TN}{n_-}$
- Balanced accuracy: $\frac{TPR+TNR}{2}$
- Precision: $\frac{TP}{\hat{n}_+}$
- AUC/AUROC: Area Under the ROC (Receiver Operating Characteristic) Curve
- AUPR: Area Under the Precision/Recall Curve

The ROC (resp. PR) curve is drawn by successively recomputing TPR and FPR (resp. Precision and Recall) for an increasing discrimination threshold θ. AUC and AUPR are then evaluated by computing the area under these curves. Note that, while AUC is usually an appropriate measure to evaluate any kind of classification problems, AUPR will mostly serve in the context of information retrieval when $n_+ << n_-$, i.e., when one is more interested in the fraction of true positives among the positives that were retrieved (see Chapter 5).

1.3.2 Training, validation and test

In order to properly evaluate the performance of a given algorithm, it is crucial to measure the accuracy on samples that have not been used to train it. Although this might be obvious to readers in the machine learning community, *we cannot emphasize enough the importance of this procedure* as it is commonly misunderstood or spuriously performed, particularly in multi-disciplinary fields such as bioinformatics. See for instance Simon et al. (2003), Ioannidis (2005a), Boulesteix (2010) for opinion papers on that matter. In the latter alarming letter, the author warns the community about the possibly terrible consequences of sloppiness in these regards. In particular, she mentions how assessing the performance of a model wrongly can lead to over-optimism. Obviously, we should be particularly careful as members of the bio-medical research community, as over-optimistic papers can turn into dangerous tools when directly used by practicians.

The correct procedure is known as *training, validation and test* (see, e.g., Hastie et al., 2009). In general,

- the training set contains data that are used for the learning part, i.e., to output a model;
- the validation set is used to perform *model selection*, e.g., to choose the correct parameters (the validation test is therefore not required when only one parameter-free model is considered);
- the test set has performance assessment purposes only.

However, we are now facing a new problem: given one dataset, how should we proceed to divide it into these three sets? In a first attempt to remove some bias, we argue for balancing these sets in the classification setting, i.e., for making sure that they exhibit about the same proportion of positive and negative examples. However, even so, we might still be facing variability: what if we happened to choose some specific sets that do not represent well the entire data? One way to remove most of it is to *repeatedly train and test*: instead of choosing one training/test division, it makes sense to repeat the performance estimation several times and keep the average as the final performance. Note how doing this will additionally make it possible to estimate the variance of the estimation.

1.3.3 k-fold cross-validation

The only procedure that keeps the test sets independent, i.e., non-overlapping, is known as *k-fold cross-validation*. Let us first assume that there is only one parameter-free model to test, that is, we are only dealing with training and test and forget about validation for the moment. The cross-validation procedure goes as follows: the entire set is divided in k folds, as equally sized as possible. Iteratively, the model is trained on $k-1$ folds and tested on the remaining fold. We illustrate this idea on Figure 1.8, with $k = 5$.

Figure 1.8: ***k*-fold cross-validation:** with $k = 5$. The model is iteratively trained on 4/5 (purple) of the data and tested on the remaining 1/5 (orange). The accuracy is thus measured 5 times and finally averaged.

In order to further remove variance, it is common to perform k-fold cross-validation many times: on top of dividing the data into k folds, we repeatedly and randomly choose the k parts, say r times, perform cross-validation on each folding design and finally average the r cross-validation errors. This is known as $r \times k$ cross-validation and, when deciding for this solution, we are probably as careful as it gets!

In many cases, we additionally have to choose a model among many, which often amounts to selecting the right parameter (e.g. λ for Lasso, C for SVM, etc.). This is where the *validation* step enters the picture in what is called *internal cross-validation*: within the cross-validation framework, that we now name *external cross-validation*, we further divide each of the k training set using a k' cross-validation. Therefore, within each external fold, the right model is chosen by averaging the k' internal cross-validation performance measures and choosing the best one. The external training set is then trained one more time, using the best internally chosen model and the final performance is assessed as usual on the external test sets.

1.3.4 Avoiding selection bias

When the model implies feature selection, one must be very careful as to avoid *selection bias* (Ambroise and McLachlan, 2002). In other words, the rule stating that *no example in the test set should have been used to train the model* still applies when we consider feature selection. We give an example in Algorithm 3 as to how to handle both selection and cross-validation.

Example 1. ***Selection with the t-test, training with a SVM.*** *Given a dataset* $(\mathbf{X}, \mathbf{Y})$*, first divide it into k folds, yielding k training sets* $(\mathbf{X}_i^{tr}, \mathbf{Y}_i^{tr})_{i=1...k}$ *and k test sets* $(\mathbf{X}_i^{ts}, \mathbf{Y}_i^{ts})_{i=1...k}$*. Then, run Algorithm 3.*

Input: Training and test data $(\mathbf{X}_i^{tr}, \mathbf{Y}_i^{tr})_{i=1...k}$ and $(\mathbf{X}_i^{ts}, \mathbf{Y}_i^{ts})_{i=1...k}$.

Output: Average performance $\overline{acc}$.

for $i = 1$ *to* k **do**

Rank the p features according to the t-test using $\mathbf{X}_i^{tr}$ and $\mathbf{Y}_i^{tr}$;

Restrain the training set and the test sets to the q most relevant features : $\tilde{\mathbf{X}_i^{tr}}, \tilde{\mathbf{X}_i^{ts}}$;

Train a SVM : $model_i = SVM_C(\tilde{\mathbf{X}_i^{tr}}, \mathbf{Y}_i^{tr})$;

Test it on the test set : $acc_i = perf(model_i, \tilde{\mathbf{X}_i^{ts}}, \mathbf{Y}_i^{ts})$;

end

return *Average performance* $\overline{acc} = 1/k \sum_{i=1}^{k} acc_i$.

Here, function *perf* refers to any of the performance measures presented in Section 1.3.1.

Algorithm 3: Cross-validation using t-test and SVM

1.4 Contributions of this thesis

In this section, we motivate the employment of feature selection methods in bioinformatics. The main contributions of this thesis relate to two main problems: *biomarker discovery for breast cancer outcome prediction* and *gene regulatory network inference*.

In both these problems, we have been working with *gene expression data* obtained from DNA microarrays.

1.4.1 Gene expression data

To define the *expression level* of a gene, we need to go back to the *central dogma of molecular biology* (Lodish et al., 2000): DNA contains both coding and non coding parts. The coding parts are what we call *genes*. All cells contain the same genes. In order for one biological process to take place in one cell, the gene has to be *activated*. Such processes include cell division, metabolism, immune system, etc. More generally, a gene can be thought of as a secret code, consisting of a sequence in which four letters are used. These letters refer to nucleotides (A=Adenin, G=Guanin, T=Thymin, C=Cytosin). Once the sequence is decoded, a particular protein can be synthesized. To achieve the synthesis, two main steps are taken: first, the gene is copied into *messenger RNA*, that exits the nucleus to join the cytoplasm. This step is called *transcription*. Second, the messenger RNA is *translated* into a protein (translation step). This procedure is depicted in Figure 1.9[1]

The transcription can only occur by the means of proteins called *transcription factors* that bind to DNA sequences, enabling or inhibiting their transcription. They are said to *regulate* the transcription. The resulting *phenotype* of a gene being transcribed is called *gene expression*. It can be measured through the quantity of RNA present for each gene. In practice, *microarrays* (Lockhart et al., 2000) can be used, that consist of a glass slide on which RNA is hybridized.

[1]We borrowed this image from `http://ncbi.nlm.nih.gov`.

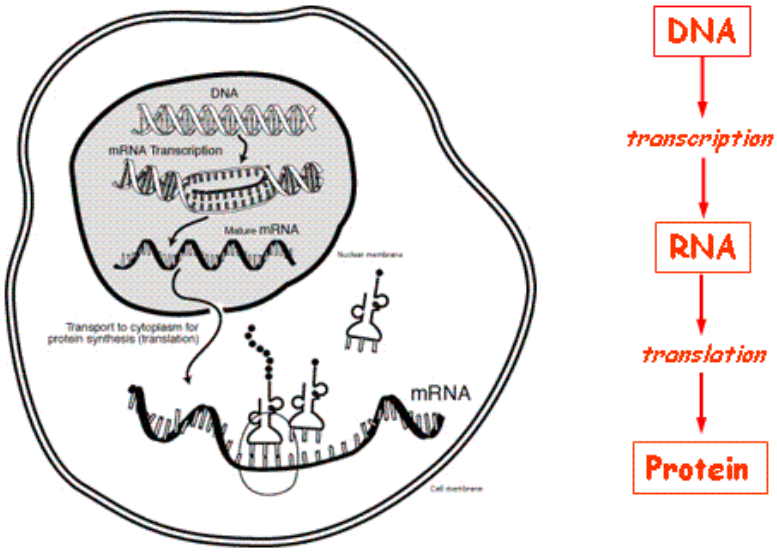


Figure 1.9: **The Central Dogma of Molecular Biology.**

Through image analysis, RNA is then quantified, resulting in what is called *gene expression data*.

It consists of a matrix with n rows and p columns, that we usually name $\mathbf{X}$ (see Section 1.1). Each row corresponds to an experiment, i.e., a measure of the expression of thousands of genes under given experimental conditions.

In the first problem we address, each sample relates to a primary breast tumor, i.e., the expression of thousands of genes is measured from a tumoral cell after a biopsy procedure. In the second, the same genome is measured under different conditions, such as the knock-out of a gene or at different time steps.

However, before being able to analyze the data, one has to preprocess it. Indeed, gene expression data as obtained from microarrays is known to be affected by various sources of bias (see, e.g., Yang et al., 2002, Quackenbush, 2002, Bolstad et al., 2003). Most datasets mentioned in this thesis were normalized using the Robust Multi-array Average procedure (Irizarry et al., 2003).

1.4.2 Biomarker discovery for breast cancer prognosis

Patients that are diagnosed with primary breast cancer usually undergo a preventive or *adjuvant* chemotherapy. However, women with a *good prognostic outcome*, i.e., who will not likely experience a relapse, should not receive this therapy as it is extremely aggressive to healthy cells. Methods based on clinical markers such as tumor grade, age or lymph node status can predict the outcome with some success (see, for example, *Adjuvant! Online*[2]).

[2] `www.adjuvantonline.com`

After a study showed that gene expression was relevant to detecting subclasses of breast cancer (Perou et al., 2000), two publications (van 't Veer et al., 2002, van de Vijver et al., 2002) suggested predicting the outcome from expression profiles. They extracted a *signature*, i.e., a list of genes that they claimed contained prognostic power for breast cancer. Several other signatures were subsequently published, e.g., Wang et al. (2005), Paik (2007).

However it was soon assessed that the published signatures had few or no genes in common and that their accuracy was very dependent on the data they were built on. The entire story is broken down in Section 6.2.

The first three chapters of this thesis present our contributions to this field. In Chapter 2, we provide an exhaustive comparison of feature selection methods to predict breast cancer outcome in light of their *accuracy, stability and interpretability*. We also test Ensemble feature selection and report results from several classification algorithms. This chapter was largely inspired by Haury et al. (2011), a joint work with Pierre Gestraud and Jean-Philippe Vert.

In Chapter 3, we evaluated the benefits of incorporating genetic prior knowledge by the means of a graph. Precisely, we use the Graph Lasso algorithm from Jacob et al. (2009). We also investigated the gain in accuracy and stability when working in an Ensemble framework. This chapter is available online under the reference Haury et al. (2010) in a slightly different form, as joint work with Laurent Jacob and Jean-Philippe Vert.

Chapter 4 presents recent work on the benefits of selecting groups of genes instead of single genes to form the signature. We used the k-support norm (Argyriou et al., 2012) for which we implemented a faster algorithm. We propose to randomize the algorithm on two levels and name the resulting method AVENGER, for Accurate Variable Extraction using the support Norm, Grouping and Extreme Randomization.

1.4.3 Gene Regulatory Network inference

For a gene to be transcribed, the procedure has to be *activated* by a transcription factor, which itself has had to be transcribed first. On the other hand, transcription can be *repressed*, i.e., inactivated by transcription factors as well. This set of relationships can be represented by Gene Regulatory Networks, i.e., directed graphs where nodes are genes and edges signify a regulation. Figure 1.10[3] shows an example of the *E. coli* regulatory network.

One main challenge is the inference of such a network, i.e., the recovery of the edges from gene expression data.

In 2010, we participated into the DREAM5 Network Inference challenge[4] (Marbach et al., 2012). Our method was derived from the Stablity Selection algorithm proposed in Bach (2008), Meinshausen and Bühlmann (2010) and ranked second on the *in vivo* sub-challenge and third overall. We named it TIGRESS for *Trustful Inference of Gene REgulation using Stability Se-*

[3]The image was drawn using Cytoscape (Smoot et al., 2011).

[4]`http://wiki.c2b2.columbia.edu/dream/index.php/The_DREAM_Project`

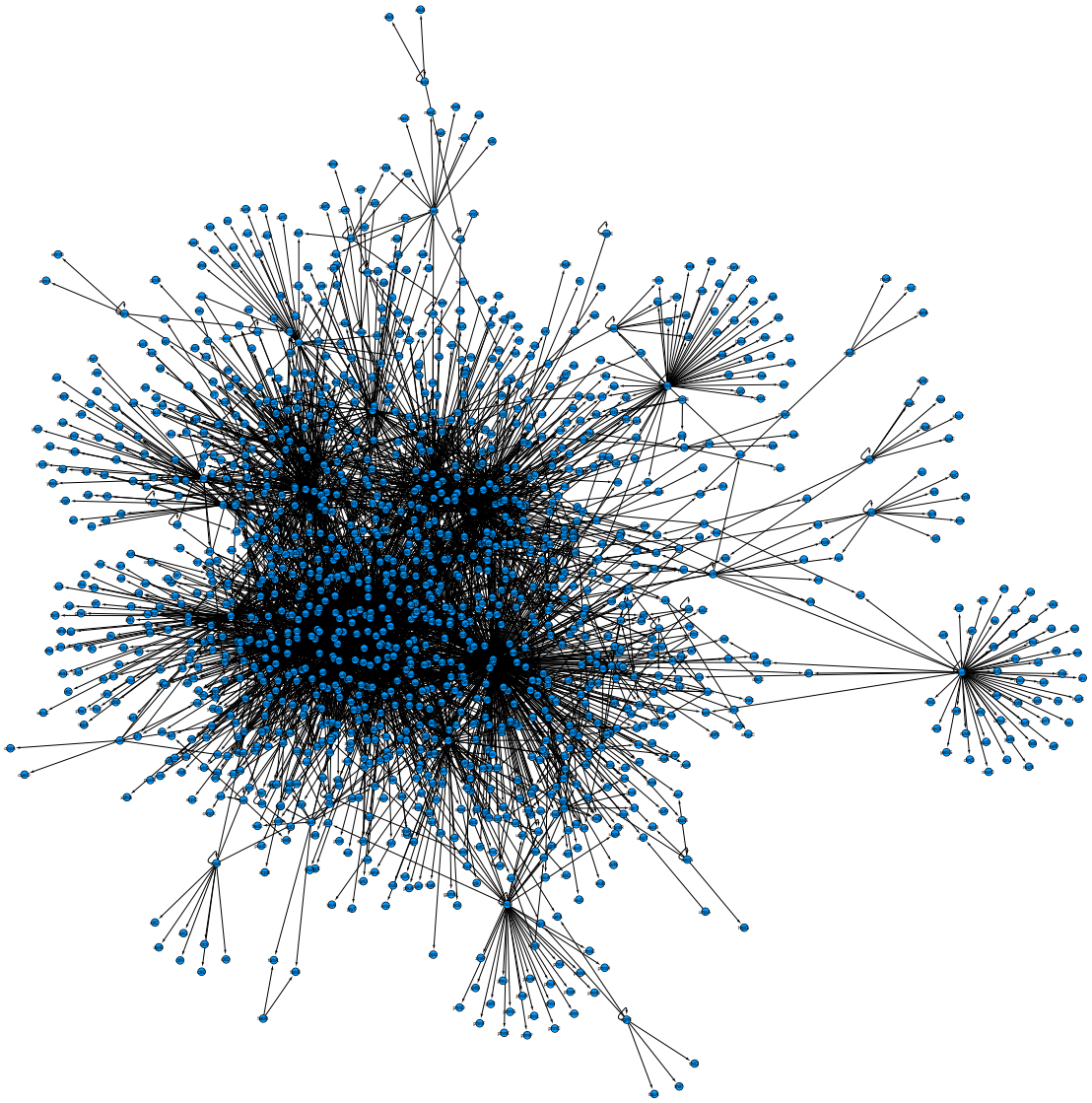


Figure 1.10: **E. coli regulatory Network:** regulation of genes by transcription factors represented as a graph.

lection. In Chapter 5, we present the method, discuss the impact of its parameters and provide results from both DREAM4 and DREAM5 challenges. The chapter was largely inspired by Haury et al. (2012), a joint work with Fantine Mordelet, Paola Vera-Licona and Jean-Philippe Vert, which is currently under review.

Chapter 2

On the influence of feature selection methods on the accuracy, stability and interpretability of molecular signatures

This chapter has been published in a slightly different form in Haury et al. (2011), as joint work with Pierre Gestraud and Jean-Philippe Vert.

Résumé

Ce chapitre traite de la sélection de signatures pour le pronostic du cancer du sein. Il s'agit d'un problème supervisé. On cherche, parmi les milliers de gènes à disposition, les quelques dizaines d'entre eux contenant suffisamment d'information pour établir le pronostic. Ce problème est connu, en particulier, pour son *instabilité*: les signatures obtenues sur différentes bases de données, voire sur différents sous-ensembles d'une même base, ne comportent pas les mêmes gènes. Nous présentons ici une comparaison de 32 méthodes de sélection de variables à partir de données d'expression génique. Nous évaluons chacune d'entre elles en termes de performance prédictive, de stabilité et d'interprétabilité des signatures obtenues. Nous montrons que les méthodes les plus simples semblent les plus efficaces de ces points de vue. En particulier, la sélection par un simple test de Student fournit le meilleur compromis performance/stabilité.

Abstract

Biomarker discovery from high-dimensional data is a crucial problem with enormous applications in biology and medicine. It is also extremely challenging from a statistical viewpoint, but surprisingly few studies have investigated the relative strengths and weaknesses of the plethora of existing feature selection methods. In this study we compare 32 feature selection methods on

4 public gene expression datasets for breast cancer prognosis, in terms of predictive performance, stability and functional interpretability of the signatures they produce. We observe that the feature selection method has a significant influence on the accuracy, stability and interpretability of signatures. Surprisingly, complex wrapper and embedded methods generally do not outperform simple univariate feature selection methods, and ensemble feature selection has generally no positive effect. Overall a simple Student's t-test seems to provide the best results.

2.1 Introduction

Biomarker discovery from high-dimensional data, such as transcriptomic or SNP profiles, is a crucial problem with enormous applications in biology and medicine, such as diagnosis, prognosis, patient stratification in clinical trials or prediction of the response to a given treatment. Numerous studies have for example investigated predictive models based on the expression of so-called *molecular signatures*, i.e., lists of a small number of genes, for the stratification of early breast cancer patients into low-risk or high-risk of relapse, in order to guide the need for adjuvant therapy (see, e.g., Sotiriou and Pusztai, 2009).

While predictive models could be based on the expression of more than a few tens of genes, several reasons motivate the search for short lists of predictive genes. First, from a statistical and machine learning perspective, restricting the number of variables is often a way to reduce over-fitting when we learn in high dimension from few samples and can thus lead to better predictions on new samples. Second, from a biological viewpoint, inspecting the genes selected in the signature may shed light on biological processes involved in the disease and suggest novel targets. Third, and to a lesser extent, a small list of predictive genes allows the design of cheap dedicated prognostic chips.

Published signatures share, however, very few genes in common, raising questions about their biological significance (Ioannidis, 2005b). Independently of differences in cohorts or technologies, Ein-Dor et al. (2005) and Michiels et al. (2005) demonstrate that a major cause for the lack of overlap between signatures is that many different signatures lead to similar predictive accuracies, and that the process of estimating a signature is very sensitive to the samples used in the phase of gene selection. Specifically, Ein-Dor et al. (2006) suggest that many more samples than currently available would be required to reach a descent level of signature stability, meaning in particular that no biological insight should be expected from the analysis of current signatures. On the positive side, some authors noticed that the biological functions captured by different signatures are similar, in spite of the little overlap between them at the gene level (see Shen et al., 2008, Reyal et al., 2008, Wirapati et al., 2008) .

From a machine learning point of view, estimating a signature from a set of expression data is a problem of *feature selection*, an active field of research in particular in the high-dimensional setting (Guyon and Elisseeff, 2003). While the limits of some basic methods for feature selection

have been highlighted in the context of molecular signatures, such as gene selection by Pearson correlation with the output (Ein-Dor et al., 2006), there are surprisingly very few and only partial investigations that focus on the *influence of the feature selection method* on the performance and stability of the signature. Lai et al. (2006) compared various feature selection methods in terms of predictive performance only, and Abeel et al. (2010) suggest that ensemble feature selection improves both stability and accuracy of SVM recursive feature elimination (RFE), without comparing it with other methods. However, it remains largely unclear how "modern" feature selection methods such as the elastic net (Zou and Hastie, 2005), SVM RFE or stability selection (Bach, 2008, Meinshausen and Bühlmann, 2010) behave in these regards and how they compare to more basic univariate techniques.

Here we propose an empirical comparison of a panel of feature selection techniques in terms of accuracy and stability, both at the gene and at the functional level. Using four breast cancer datasets, we observe significant differences between the methods. Surprisingly, we find that ensemble feature selection, i.e., combining multiple signatures estimated on random subsamples, has generally no positive impact, and that simple filters can outperform more complex wrapper or embedded methods.

2.2 Materials and Methods

2.2.1 Feature selection methods

We compare eight common feature selection methods to estimate molecular signatures. All methods take as input a matrix of gene expression data for a set of samples from two categories (good and bad prognosis in our case), and return a set of genes of a user-defined size q. These genes can then be used to estimate a classifier to predict the class of any sample from the expression values of these genes only. Feature selection methods are usually classified into three categories (see Kohavi and John, 1997, Guyon and Elisseeff, 2003): *filter methods* select subsets of variables as a pre-processing step, independently of the chosen predictor; *wrapper methods* utilize the learning machine of interest as a black box to score subsets of variable according to their predictive power; finally, *embedded methods* perform variable selection in the process of training and are usually specific to given learning machines. We have selected popular methods representing these three classes, as described below.

Filter methods

Univariate filter methods rank all variables in terms of relevance, as measured by a score which depends on the method. They are simple to implement and fast to run. To obtain a signature of size q, one simply takes the top q genes according to the score. We consider the following four scoring functions to rank the genes: the *Student's t-test* and *Wilcoxon sum-rank test*, which

evaluate if each feature is differentially expressed between the two classes; and the *Bhattacharyya distance* and *relative entropy* to calculate a distance between the distributions of the two groups. We used the MATLAB Bioinformatics toolbox to compute these scoring functions.

Wrapper methods

Wrapper methods attempt to select jointly sets of variables with good predictive power for a predictor. Since testing all combinations or variables is computationally impossible, wrapper methods usually perform a greedy search in the space of sets of features. We test *SVM recursive feature elimination (RFE)* (Guyon et al., 2002), which starts with all variables and iteratively removes the variables which contribute least to a linear SVM classifier trained on the current set of variables. We remove 20% of features at each iteration until q remain, and then remove them one by one in order to rigourously rank the first q. Following Abeel et al. (2010), we set the SVM parameter C to 1, and checked afterwards that other values of C did not have a significant influence on the results. Alternatively, we test a *Greedy Forward Selection (GFS)* strategy for least squares regression also termed Orthogonal Matching Pursuit, where we start from no variable and add them one by one by selecting each time the one which minimizes the sum of squares, in a 3-fold internal cross-validation setting. This algorithm was implemented in the SPAMS toolbox for Matlab initially published along with Mairal et al. (2010).

Embedded methods

Embedded methods are learning algorithms which perform feature selection in the process of training. We test the popular *Lasso* regression (Tibshirani, 1996), where a sparse linear predictor $\omega \in \mathbb{R}^p$ is estimated by minimizing the objective function $R_l(\omega) + \lambda\|\omega\|_1$, where $R_l(\omega)$ is the empirical risk using the square loss on the training set (considering the two categories as ± 1 values) and $\|\omega\|_1 = \sum_{i=1}^{p} |\omega_i|$. λ controls the degree of sparsity of the solution, i.e., the number of features selected. We fix λ as the smallest value which gives a signature of the desired size q. Alternatively, we tested the elastic net (Zou and Hastie, 2005), which is similar to the Lasso but where we replace the ℓ_1 norm of ω by a combination of the ℓ_1 and ℓ_2 norms, i.e., we minimize $R_l(\omega) + \lambda\|\omega\|_1 + \lambda/2\|\omega\|_2^2$ and $\|\omega\|_2^2 = \sum_{i=1}^{p} \omega_i^2$. By allowing the selection of correlated predictive variables, the elastic net is supposed to be more robust than the Lasso while still selecting predictive variables. Again, we tune λ to achieve a user-defined level of sparsity. For both algorithms, we used the code implemented in the SPAMS toolbox.

2.2.2 Ensemble feature selection

Many feature selection methods are known to be sensitive to small perturbations of the training data, resulting in unstable signatures. In order to "stabilize" variable selection, several authors have proposed to use ensemble feature selection on bootstrap samples: the variable selection

method is run on several random subsamples of the training data, and the different lists of variables selected are merged into a hopefully more stable subset (see, e.g., Bi et al., 2003, Meinshausen and Buehlmann, 2009, Abeel et al., 2010).

For each feature selection method described above, we tested in addition the following three aggregation strategies for ensemble feature selection. We first bootstrap the training samples $B = 50$ times (i.e., draw a sample of size n from the data with replacement B times) to get B rankings $(r^1 ... r^B)$ of all features by applying the feature selection method on each sample. For filter methods, the ranking of features is naturally obtained by decreasing score. For RFE and GFS, the ranking is the order in which the features are added or removed in the iterative process. For Lasso and elastic net, the ranking is the order in which the variables become selected when λ decreases. We then aggregate the B lists by computing a score $S_j = 1/B \sum_{b=1}^{B} f(r_j^b)$ for each gene j as an average function of its rank r_j^b in the b-th bootstrap experiment. We test the following functions of the rank for aggregation:

- *Ensemble-mean (Abeel et al., 2010)*: we simply average the rank of a gene over the bootstrap experiments, i.e., we take $f(r) = p - r$.
- *Ensemble-stability selection (Meinshausen and Bühlmann, 2010)*: we measure the percentage of bootstrap samples for which the gene ranks in the top q, i.e., $f(r) = 1$ if $r \leq q$, 0 otherwise.
- *Ensemble-exponential*: we propose a soft version of stability selection, where we average an exponentially decreasing function of the rank, namely $f(r) = \exp\{-r/q\}$.

Finally, for each rank aggregation strategy, the aggregated list is the set of q genes with the largest score.

2.2.3 Accuracy of a signature

In order to measure the predictive accuracy of a feature selection method, we assess the performance of various supervised classification algorithms trained on the data restricted to the selected signature. More precisely, we test 5 classification algorithms: nearest centroids (NC), k-nearest neighbors (KNN) with $k = 9$, linear SVM with $C = 1$, linear discriminant analysis (LDA) and naive Bayes (BAYES). The parameters of the KNN and SVM methods are fixed to arbitrary default values, and we have checked that no significantly better results could be obtained with other parameters by testing a few other parameters. We assess the performance of a classifier by the area under the ROC curve (AUC), in two different settings. First, on each dataset, we perform a 10-fold cross-validation (CV) experiment, where both feature selection and training of the classifier are performed on 90% of the data, and the AUC is computed on the remaining 10% of the data. This is a classical way to assess the relevance of feature selection of a given dataset. Second, to assess the performance of the signature across datasets, we estimate

a signature on one dataset, and assess its accuracy on other datasets by again running a 10-fold CV experiment where only the classifier (restricted to the genes in the signature) is retrained on each training set. In both cases, we report the mean AUC across the folds and datasets, and assess the significance of differences between methods with a paired ANOVA test.

2.2.4 Stability of a signature

To assess the stability of feature selection methods, we compare signatures estimated on different samples in various settings. First, to evaluate stability with respect to small perturbation of the training set, we randomly subsample each dataset into pairs of subsets with 80% of sample overlap, estimate a signature on each subset, and compute the overlap between two signatures in a pair as the fraction of shared genes, i.e., $|S_1 \cap S_2|/q$. Note that this corresponds to the figure of merit defined by Ein-Dor et al. (2006). The random sampling of subsets is repeated 20 times on each dataset, and the stability values are averaged over all samples. We will refer to this procedure the *soft-perturbation* setting in the remaining. Second, to assess stability with respect to strong perturbation within a dataset, we repeat the same procedure but this time with no overlap between two subsets of samples. In practice, we can only sample subsets of size $N/2$, where N is the number of samples in a dataset, to ensure that they have no overlap. Again, we measure the overlap between the signatures estimated on training sets with no sample in common. We call this procedure the *hard-perturbation* setting. Finally, to assess the stability across datasets, we estimate signatures on each dataset independently, using all samples on each dataset, and measure their overlap. We call this procedure the *between-datasets setting* below.

2.2.5 Functional interpretability and stability of a signature

To interpret a signature in terms of biological functions, we perform functional enrichment analysis by inspecting the signature for over-represented Gene Ontology (GO) terms. This may hint at biological hypothesis underlying the classification (Shen et al., 2008, Reyal et al., 2008). We perform a hypergeometric test on each of the $5,830$ GO biological process (BP) terms that are associated to at least one gene in our dataset, and correct the resulting p-values for multiple testing through the procedure of Benjamini and Hochberg (1995). To assess the *interpretability* of a signature, i.e., how easily one can extract a biological interpretation, we compute the number of GO terms over-represented at 5% FDR. To compare two signatures in functional terms, we first extract from each signature the list of 10 GO terms with the smallest p-values, and compare the two lists of GO terms by the similarity measure of Wang et al. (2007) which takes into account not only the overlap between the lists but also the relationships between GO BP. Finally, to assess the *functional stability* of a selection method, we follow a procedure similar to the one presented in the previous section and measure the mean functional similarity of signatures in the soft-perturbation, hard-perturbation and between-datasets settings.

2.2.6 Data

We collected 4 breast cancer datasets from Gene Expression Omnibus (Barrett et al., 2009), as described in Table 2.1. The four datasets address the same problem of predicting metastatic relapse in breast cancer on different cohorts, and were obtained with the Affymetrix HG-U133A technology. We used a custom CDF file with EntrezGene ids as identifiers (Dai et al., 2005) to estimate expression levels for 12,065 genes on each array, and normalized all arrays with the Robust Multi-array Average procedure (Irizarry et al., 2003).

Table 2.1: **Data**

Dataset name	♯ examples	♯ positives	source
GSE1456	159	40	Pawitan et al. (2005)
GSE2034	286	107	Wang et al. (2005)
GSE2990	125	49	Sotiriou et al. (2006)
GSE4922	249	89	Ivshina et al. (2006)

The four breast cancer datasets used in this study.

2.3 Results

2.3.1 Accuracy

We first assess the accuracy of signatures obtained by different feature selection methods. Intuitively, the accuracy refers to the performance that a classifier trained on the genes in the signature can reach in prediction. Although some feature selection methods (wrapper and embedded) jointly estimate a predictor, we dissociate here the process of selecting a set of genes and training a predictor on these genes, in order to perform a fair comparison common to all feature selection methods. We test the accuracy of 100-gene signatures obtained by each feature selection method, combined with 5 classifiers to build a predictor as explained in the Methods section. Table 2.2 shows the mean accuracies (in AUC) over the datasets as reached by the different combinations in 10-fold cross-validation.

Globally, we observe only limited differences between the feature selection methods, for a given classification method. In particular the selection of a random signature reaches a baseline AUC comparable to that of other methods, confirming results already observed by Ein-Dor et al. (2005). Second, we observe that, among all classification algorithms, the simple NC classifier consistently gives good results compared to other classifiers. We therefore choose it as a default classification algorithm for further assessment of the performance of the signatures below. Figure 2.1 depicts graphically the AUC reached by each feature selection method with NC as a classifier, reproducing the first three lines of Table 2.2. Although the t-test has the best average AUC, the

Table 2.2: **AUC (10-fold cross-validation)**

Class.	Type	Random	t-test	Entropy	Bhatt.	Wilcoxon	SVM RFE	GFS	Lasso	Elastic Net
NC	S	0.62(0.17)	**0.66(0.14)**	0.58(0.15)	0.60(0.15)	0.62(0.15)	0.62(0.15)	0.58(0.15)	0.63(0.15)	0.63(0.15)
	E-M	0.62(0.15)	0.65(0.14)	**0.59(0.15)**	**0.63(0.15)**	0.62(0.15)	**0.63(0.14)**	**0.62(0.13)**	0.61(0.16)	0.63(0.15)
	E-E	0.61(0.15)	0.65(0.14)	0.59(0.15)	0.61(0.16)	0.62(0.15)	0.61(0.15)	0.58(0.13)	0.63(0.13)	0.63(0.14)
	E-S	**0.63(0.14)**	0.65(0.14)	0.58(0.15)	0.61(0.15)	0.62(0.15)	0.63(0.15)	0.59(0.12)	0.63(0.13)	**0.63(0.14)**
KNN	S	0.59(0.16)	0.61(0.15)	0.52(0.11)	0.57(0.13)	0.63(0.15)	0.60(0.15)	0.59(0.13)	0.60(0.17)	0.60(0.17)
	E-M	0.61(0.14)	0.62(0.15)	0.57(0.15)	0.60(0.15)	**0.64(0.16)**	0.62(0.15)	0.61(0.12)	0.61(0.15)	0.60(0.12)
	E-E	0.55(0.13)	0.63(0.15)	0.53(0.10)	0.54(0.10)	0.63(0.16)	0.60(0.17)	0.54(0.16)	0.61(0.14)	0.60(0.17)
	E-S	0.60(0.13)	0.63(0.15)	0.54(0.11)	0.54(0.12)	0.62(0.16)	0.58(0.14)	0.55(0.14)	0.62(0.14)	0.60(0.14)
LDA	S	0.54(0.12)	0.56(0.12)	0.51(0.14)	0.55(0.13)	0.52(0.12)	0.56(0.12)	0.50(0.13)	0.58(0.14)	0.57(0.14)
	E-M	0.53(0.10)	0.55(0.13)	0.55(0.13)	0.58(0.12)	0.56(0.13)	0.60(0.15)	0.52(0.14)	0.59(0.14)	0.60(0.13)
	E-E	0.54(0.13)	0.53(0.15)	0.52(0.15)	0.53(0.11)	0.53(0.14)	0.57(0.13)	0.53(0.15)	0.59(0.12)	0.58(0.13)
	E-S	0.54(0.13)	0.52(0.13)	0.54(0.13)	0.55(0.12)	0.52(0.14)	0.57(0.16)	0.54(0.15)	0.59(0.15)	0.60(0.13)
NB	S	0.57(0.14)	0.60(0.13)	0.58(0.11)	0.58(0.14)	0.57(0.13)	0.56(0.14)	0.54(0.11)	0.59(0.15)	0.59(0.15)
	E-M	0.59(0.13)	0.59(0.14)	0.57(0.14)	0.59(0.13)	0.57(0.13)	0.56(0.13)	0.59(0.12)	0.57(0.15)	0.57(0.14)
	E-E	0.55(0.15)	0.60(0.14)	0.58(0.12)	0.57(0.13)	0.58(0.13)	0.57(0.14)	0.58(0.11)	0.58(0.12)	0.58(0.13)
	E-S	0.58(0.14)	0.60(0.14)	0.57(0.13)	0.57(0.13)	0.58(0.13)	0.56(0.14)	0.58(0.10)	0.58(0.11)	0.58(0.13)
SVM	S	0.56(0.18)	0.56(0.15)	0.55(0.11)	0.55(0.12)	0.54(0.15)	0.62(0.14)	0.51(0.16)	0.62(0.15)	0.62(0.15)
	E-M	0.51(0.15)	0.55(0.14)	0.59(0.16)	0.60(0.13)	0.56(0.13)	0.62(0.15)	0.55(0.16)	0.61(0.16)	0.61(0.16)
	E-E	0.54(0.16)	0.54(0.15)	0.54(0.13)	0.54(0.12)	0.55(0.15)	0.61(0.17)	0.56(0.17)	**0.63(0.13)**	0.62(0.16)
	E-S	0.54(0.17)	0.55(0.18)	0.56(0.12)	0.56(0.12)	0.54(0.14)	0.61(0.16)	0.55(0.17)	0.63(0.14)	0.62(0.16)

AUC obtained for each combination of feature selection and classification method, in 10-fold cross validation and averaged over the datasets. Standard error is shown within parentheses. For each selection algorithm, we highlighted the setting in which it obtained the best performance. The *Type* column refers to the use of feature selection run a single time (S) or through ensemble feature selection, either with the mean (E-M), exponential (E-E) or stability selection (E-S) procedure to aggregate lists.

results vary widely across datasets explaining the large error bars. In fact, a paired ANOVA test detects no method significantly better than the random selection strategy; the only significant differences are observed between t-test, on the one hand, and Entropy and GFS, on the other hand, which have the lowest performances without aggregation. In particular, we observe that ensemble methods for feature selection do not bring any improvement in accuracy in a significant way.

In order to assess how a signature estimated on one dataset behave in another dataset, we report the results for between-datasets experiments in Table 2.3. For each training dataset, we highlight the method with the best results, and report the average results (over the $4 \times 3 \times 10 = 120$ folds) in the last row. In this setting, we barely notice any difference with the cross-validation setting (Table 2.2) and essentially reach the same conclusions, namely that no significant result stands out, except for the t-test to perform overall better than entropy.

In order to check how these results depend on the size of the signature, we plot in Figure 2.2 the AUC of the 9 feature selection methods, with or without ensemble averaging, combined with a NC classifier, as a function of the size of the signature. Interestingly, we observe that in some cases the AUC seems to increase early, implying that fewer than 100 genes may be sufficient to obtain the maximal performance. Indeed, while it is significant that 100-gene signatures

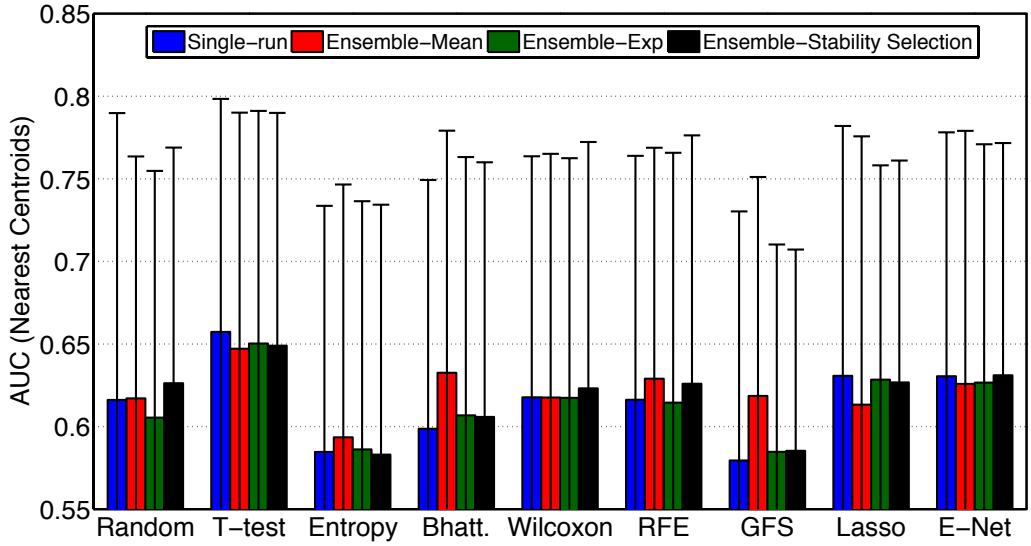


Figure 2.1: **Area under the ROC curve.** Signature of size 100 in a 10-fold CV setting and averaged over the four datasets

Table 2.3: **AUC (between-datasets setting)**

Training data	Type	Random	t-test	Entropy	Bhatt.	Wilcoxon	SVM RFE	GFS	Lasso	Elastic Net
GSE1456	S	0.59(0.10)	**0.63(0.13)**	0.60(0.10)	0.63(0.13)	0.61(0.14)	0.61(0.13)	0.61(0.11)	0.62(0.11)	0.62(0.11)
	E-M	0.60(0.12)	0.63(0.14)	0.60(0.12)	0.61(0.14)	0.61(0.14)	0.61(0.11)	0.60(0.12)	0.63(0.11)	0.60(0.12)
	E-E	0.60(0.13)	0.63(0.13)	0.58(0.10)	0.63(0.12)	0.61(0.13)	0.61(0.11)	0.62(0.12)	0.63(0.11)	0.62(0.11)
	E-S	0.60(0.14)	0.63(0.14)	0.59(0.10)	0.63(0.11)	0.61(0.13)	0.61(0.13)	0.62(0.13)	0.63(0.12)	0.63(0.09)
GSE2034	S	0.62(0.15)	0.62(0.15)	0.57(0.20)	0.59(0.19)	0.58(0.19)	0.60(0.18)	0.62(0.15)	0.63(0.16)	0.63(0.16)
	E-M	0.63(0.17)	0.63(0.15)	0.60(0.15)	0.64(0.16)	0.58(0.19)	0.63(0.17)	0.62(0.16)	0.62(0.16)	0.62(0.16)
	E-E	**0.64(0.14)**	0.63(0.15)	0.56(0.19)	0.58(0.19)	0.59(0.19)	0.63(0.16)	0.60(0.18)	0.61(0.16)	0.61(0.16)
	E-S	0.61(0.17)	0.63(0.16)	0.56(0.17)	0.57(0.19)	0.59(0.19)	0.63(0.15)	0.62(0.17)	0.62(0.16)	0.63(0.16)
GSE2990	S	0.64(0.14)	0.64(0.15)	0.56(0.14)	0.60(0.16)	0.60(0.16)	0.62(0.16)	0.64(0.15)	0.66(0.13)	0.65(0.13)
	E-M	0.61(0.15)	0.66(0.16)	0.59(0.17)	0.65(0.13)	0.58(0.16)	0.65(0.15)	0.62(0.14)	0.64(0.15)	0.64(0.15)
	E-E	0.61(0.14)	**0.66(0.15)**	0.54(0.14)	0.57(0.19)	0.59(0.15)	0.62(0.15)	0.63(0.15)	0.65(0.14)	0.66(0.14)
	E-S	0.62(0.15)	0.66(0.14)	0.55(0.14)	0.57(0.18)	0.60(0.16)	0.64(0.15)	0.63(0.14)	0.65(0.14)	0.65(0.14)
GSE4922	S	0.65(0.15)	0.66(0.15)	0.59(0.16)	0.63(0.14)	0.64(0.16)	0.64(0.14)	0.62(0.12)	0.65(0.14)	0.65(0.14)
	E-M	0.65(0.12)	**0.67(0.15)**	0.64(0.13)	0.66(0.16)	0.65(0.15)	0.64(0.13)	0.65(0.15)	0.66(0.14)	0.64(0.13)
	E-E	0.65(0.15)	0.66(0.15)	0.57(0.16)	0.63(0.15)	0.66(0.15)	0.64(0.12)	0.65(0.13)	0.67(0.13)	0.66(0.14)
	E-S	0.65(0.15)	0.65(0.15)	0.60(0.16)	0.62(0.16)	0.66(0.16)	0.63(0.12)	0.63(0.10)	0.66(0.13)	0.65(0.13)
Average	S	0.62(0.14)	0.64(0.15)	0.58(0.15)	0.61(0.15)	0.61(0.16)	0.62(0.15)	0.62(0.13)	0.64(0.13)	0.64(0.14)
	E-M	0.62(0.14)	**0.65(0.15)**	0.61(0.15)	0.64(0.15)	0.61(0.16)	0.63(0.14)	0.62(0.14)	0.64(0.14)	0.62(0.14)
	E-E	0.62(0.14)	0.64(0.15)	0.56(0.15)	0.60(0.17)	0.61(0.16)	0.63(0.13)	0.62(0.14)	0.64(0.14)	0.64(0.14)
	E-S	0.62(0.15)	0.64(0.15)	0.58(0.15)	0.60(0.16)	0.61(0.16)	0.63(0.14)	0.62(0.14)	0.64(0.14)	0.64(0.13)

AUC obtained with Nearest Centroids when a signature is learnt from one dataset and tested by 10-fold cross-validation on the three remaining datasets. Standard error is shown within parentheses. For each training dataset, we highlighted the best performance. The *Type* column refers to the use of feature selection run a single time (S) or through ensemble feature selection, either with the mean (E-M), exponential (E-E) or stability selection (E-S) procedure to aggregate lists.

perform better than a list of fewer than 10 features ($p < 0.05$ regardless of the method or the setting), signatures of size 50 do not lead to significantly worse performances in general. It is worth noting that some algorithms have an increasing AUC curve in this range of sizes, and we observe no overfitting that may lead to a decreasing AUC when the number of features increases. Random selection was previously shown to give an AUC equivalent to other methods for a large signature, but as we observe on this picture, the fewer genes the larger the gap in AUC.

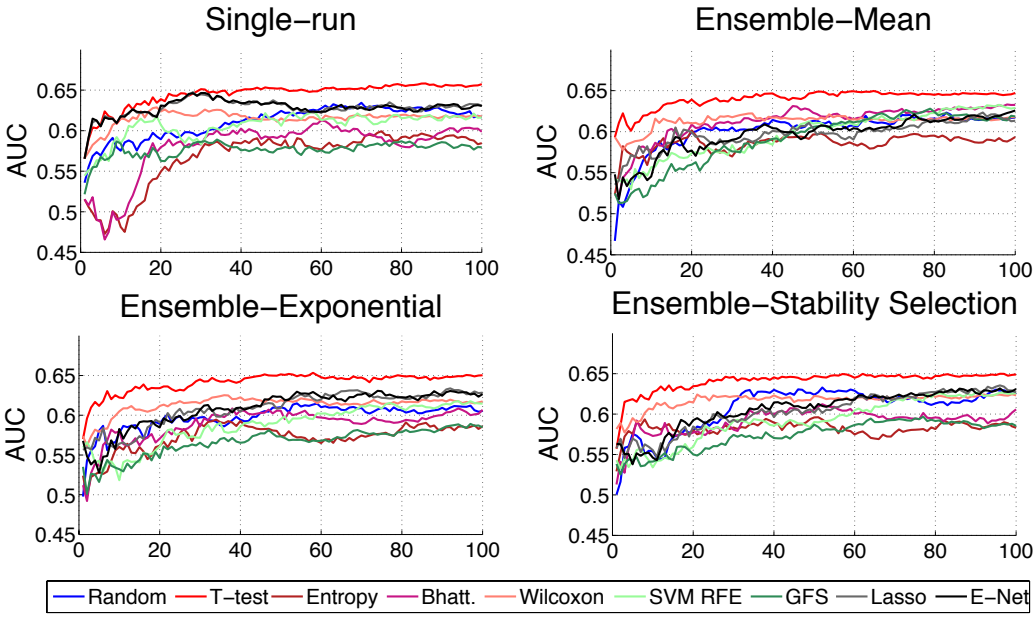


Figure 2.2: **Area Under the ROC Curve.** NC classifier trained as a function of the size of the signature, for different feature selection methods, in a 10-fold CV setting averaged over the four datasets

In order to assess the influence of the number of samples used to estimate the signature, we computed the 10-fold cross-validation AUC (repeated 50 times) reached with a NC classifier as a function of the number of samples in the training set. Figure 2.3 shows the AUC averaged over the four datasets, for each feature selection method, while Figure 2.4 shows the same AUC on each dataset separately.

With no surprise, we observe that the average accuracy clearly increases with the number of samples in the training set, for all methods, and that the relative order of the different methods does not strongly depend on the number of samples. While it is impossible to extrapolate the curve, it is not hard to imagine that it would continue to increase to a certain point. On this plot, t-test clearly outperforms the rest of the methods. However, looking at the behavior of the methods with respect to the size of the training set on each set separately, we note that not only the level of performance but also the relative order between methods strongly depend on the dataset. For example, while t-test outperforms all methods in the GSE4922 dataset,

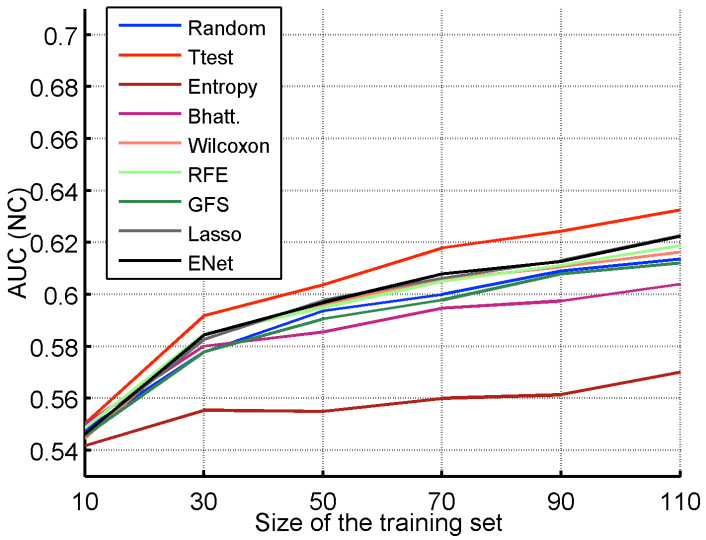


Figure 2.3: **Area Under the ROC Curve.** NC classifier trained as a function of the number of samples in a 50×10-fold CV setting. We show here the accuracy for 100-gene signatures as averaged over the 4 datasets. Note that the maximum value of the x axis is constrained by the smallest dataset, namely GSE2990.

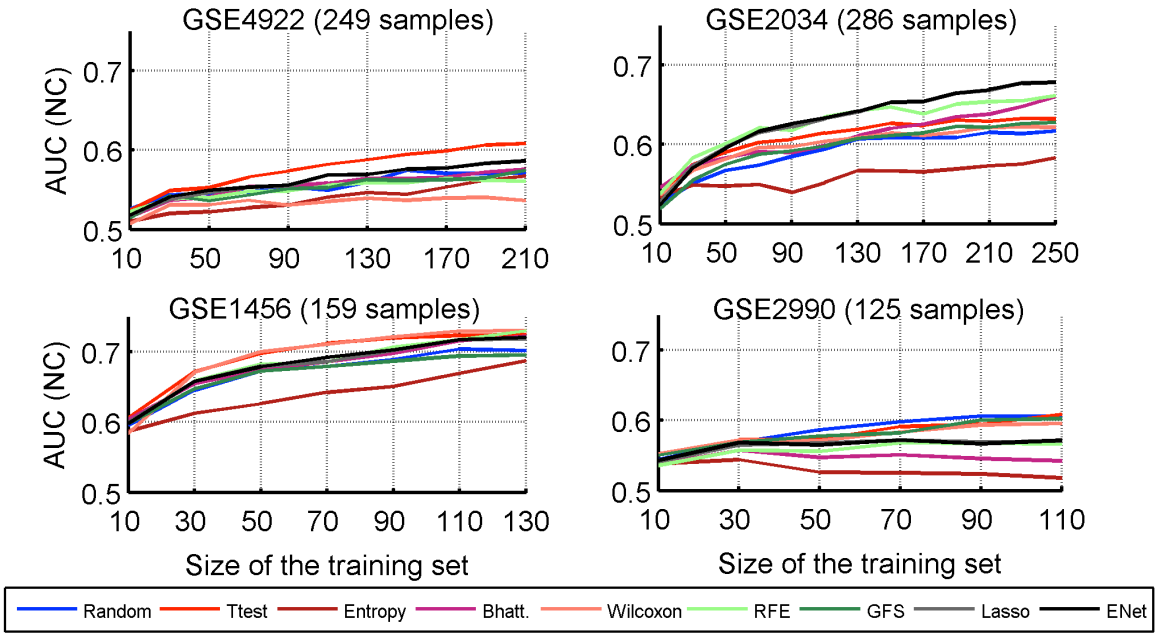


Figure 2.4: **Area Under the ROC Curve.** NC classifier trained as a function of the number of samples in a 50×10-fold CV setting for each of the four datasets. We show here the accuracy for 100-gene signatures.

Lasso and Elastic Net seem to be the best choices in GSE2034. On the other hand, we observe that the best methods on each datasets have not reached their asymptote yet, suggesting by extrapolation that better accuracies could be reached with larger cohorts.

2.3.2 Stability of gene lists

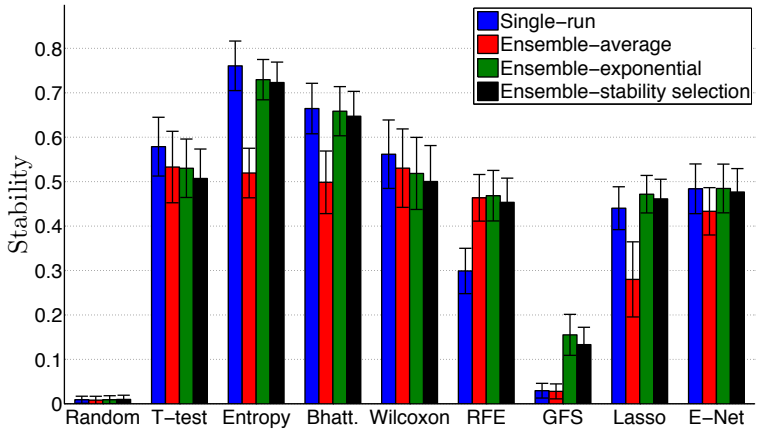
We now assess the stability of signatures created by different feature selection methods at the gene level. Figure 2.5 compares the stability of 100-gene signatures estimated by all feature selection methods tested in this benchmark, in the three experimental settings: soft-perturbation, hard-perturbation and between-datasets settings. The results are averaged over the bootstrap replicates and the four datasets.

It appears very clearly and significantly that filter methods provide more stable lists than wrappers and embedded methods. It also seems that ensemble-exponential and ensemble-stability selection yield much more stable signatures than ensemble-average. It is worth noting that a significant gain in robustness through bootstrap is only observable for relative entropy and Bhattacharyya distance. Interestingly, SVM-RFE seems to benefit from ensemble aggregation in the soft-perturbation setting, as observed by Abeel et al. (2010), but this effect seems to vanish in the more relevant hard-perturbation and between-dataset settings.

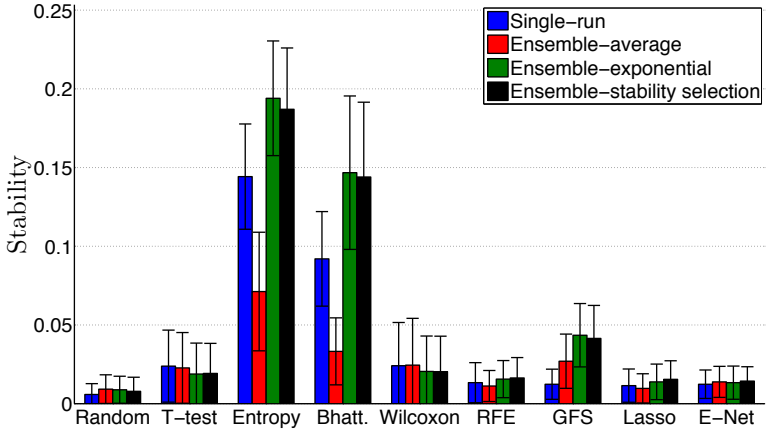
Obviously, subfigures 2.5B) and 2.5C) are very much alike while Figure 2.5A) stands aside. They confirm that the hard-perturbation setting is the best way to estimate the behavior of the algorithms between different studies. The larger stability observed in the between-datasets setting compared to the hard perturbation setting for some methods (e.g., t-test) is essentially due to the fact that signatures are trained on more samples in the between-dataset setting, since no split is required within a dataset .

Figure 2.6 illustrates this difference for one feature selection method. It shows the stability of the t-test in both settings with respect to the number of samples used to estimate signatures. While both curves remain low, we observe like Ein-Dor et al. (2006) a very strong effect of the number of samples. Interestingly, we observe that for very small sample sizes the stability in the hard-perturbation setting is a good proxy for the stability in the between-dataset setting. However, the slope of the hard-perturbation setting stability seems sharper, suggesting that the gap would stretch for larger sample sizes, should the blue curve be extrapolated. These results suggest that i) the main reason for the low stability values is really the sample size and ii) the uniformity of the cohort still plays a role for larger sizes of training sets.

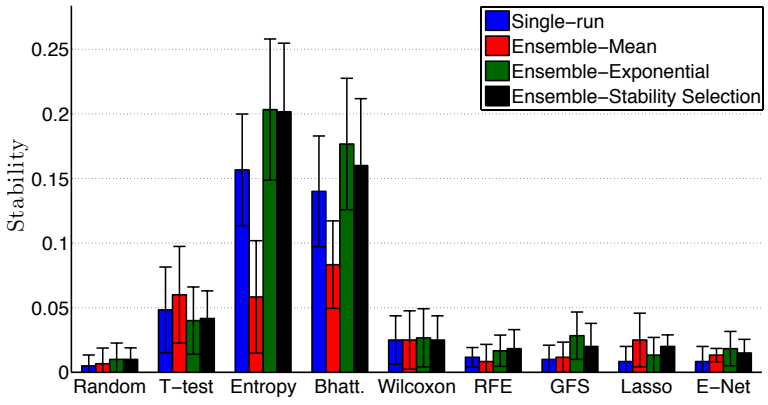
We also observe in Figure 2.7 that the relative stability of the different methods does not depend on the size of the signature over a wide range of values, confirming that the differences observed for signatures of size 100 reveal robust differences between the methods.



(a) Soft-perturbation setting



(b) Hard-perturbation setting



(c) Between-datasets setting

Figure 2.5: **Stability for a signature of size 100.** Average and standard errors are obtained over the four datasets. a) Soft-perturbation setting. b) Hard-perturbation setting. c) Between-datasets setting.

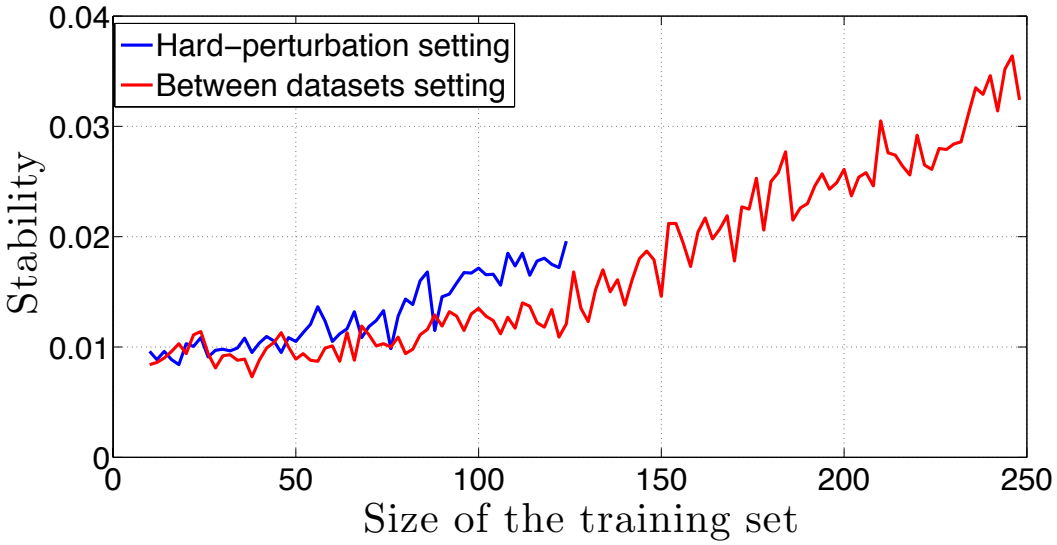


Figure 2.6: **Evolution of stability of t-test signatures** with respect to the size of the training set in the hard-perturbation and the between datasets settings from GSE2034 and GSE4922.

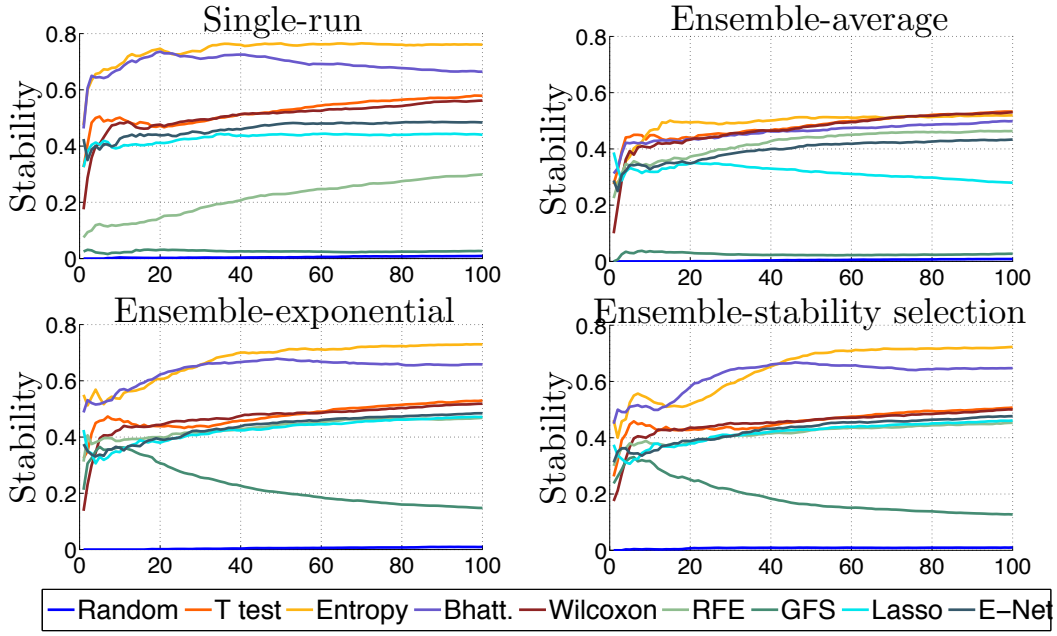


Figure 2.7: **Stability of different methods** in the between-dataset setting, as a function of the size of the signature.

2.3.3 Interpretability and functional stability

Even when different signatures share no or little overlap in terms of genes, it is possible that they encode the same biological processes and be useful if we can extract information about these processes from the gene lists in a robust manner. In the case of breast cancer prognostic

signatures, for example, several recent studies have shown that functional analysis of the signatures can highlight coherent biological processes (see, e.g., Fan et al., 2006, Reyal et al., 2008, Shen et al., 2008, Abraham et al., 2010, Shi et al., 2010). Just like stability at the gene level, it is therefore important to assess the stability of biological interpretation that one can extract from signatures.

First, we evaluate the *interpretability* of signatures of size 100, i.e., the ability of functional analysis to bring out a biological interpretation for a signature.

As shown on Figure 2.8, the four filter methods appear to be much more interpretable than wrappers/embedded methods. However, it should be pointed out that the number of significant GO terms is often zero regardless of the algorithm, leading to large error bars. Ensemble methods do not seem to enhance the interpretability of signatures.

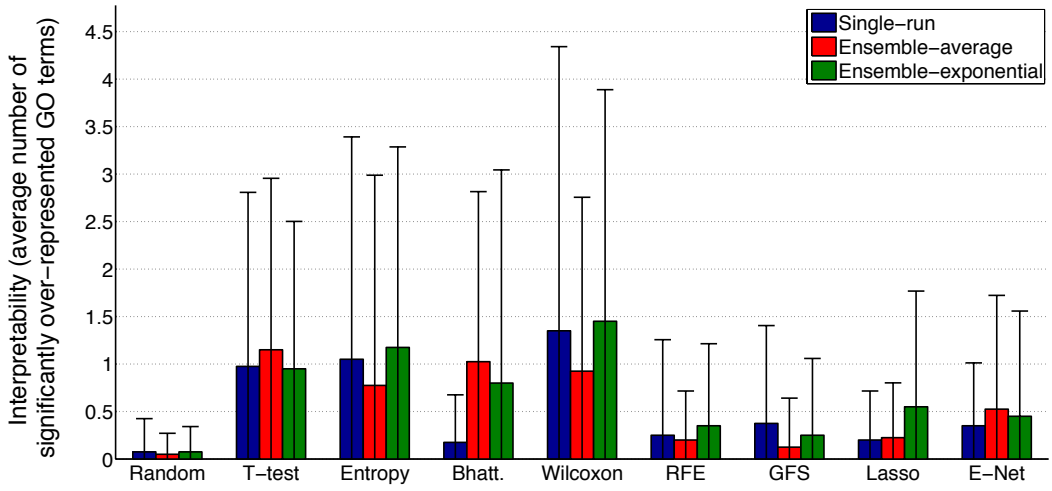
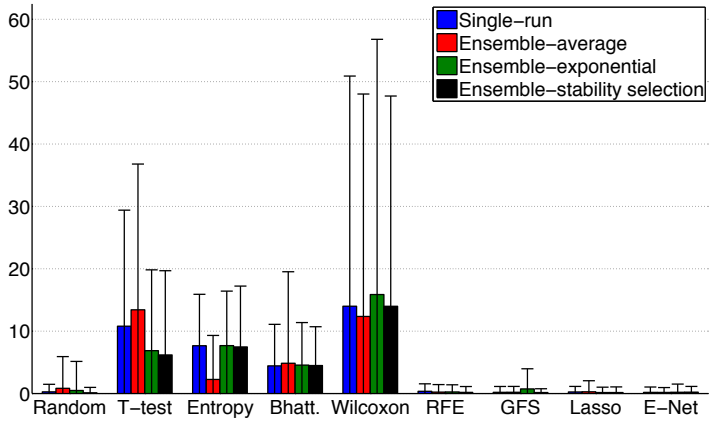


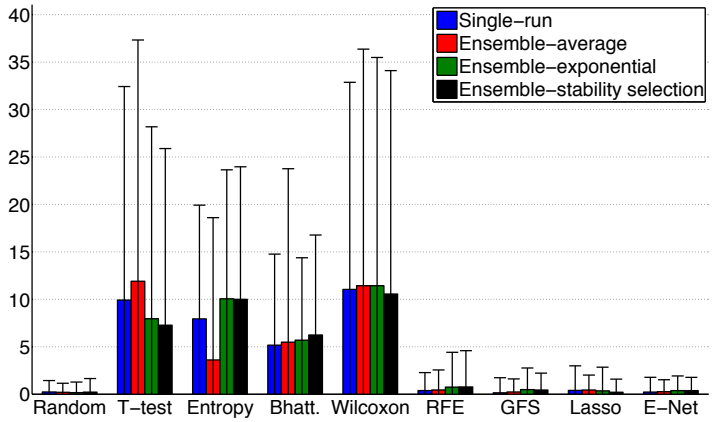
Figure 2.8: **GO interpretability for a signature of size 100.** Average number of GO BP terms significantly over-represented.

Second, we assess on Figure 2.9 the functional stability for all methods in the three settings. While the baseline stability, as obtained by random signatures, is approximatively the same regardless of the setting, we observe that, like stability at the gene level, soft- and hard-perturbation can lead to very different interpretations. This suggests again that the high functional stability obtained by several methods in the soft-perturbation setting is mainly due to the overlap in samples. Hence the hard-perturbation setting seems to be a much better proxy for the between-datasets framework.

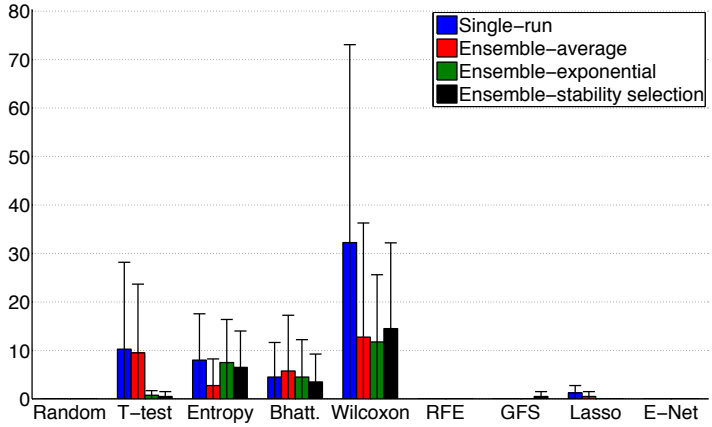
Stability results at the functional level are overall very similar to the results at the gene level, namely, we observe that univariate filters are overall the most stable methods, and that the hard-perturbation setting returns a trustworthy estimate of the inter-datasets stability. In particular, an ANOVA procedure reveals that in the single-run settings, only signatures obtained



(a) Soft-perturbation setting



(b) Hard-perturbation setting



(c) Between-datasets setting

Figure 2.9: **GO stability for a signature of size 100 in the soft-perturbation setting.** Average and standard errors are obtained over the four datasets. a) Soft-perturbation setting. b) Hard-perturbation setting. c) Between-datasets setting.

from filters are significantly more stable than random. We also note that Ensemble-mean never improves the functional stability and that Ensemble-exponential/Ensemble stability selection return more stable signatures than single-run for Entropy and Bhattacharyya as well as for GFS and Lasso although less significantly.

2.3.4 Bias issues in selection with Entropy and Bhattacharrya distance

Gene selection by relative entropy and Bhattacharyya distance is more stable but less accurate than random selection, which suggests a bias in the method which may preferably and consistently select particular genes, not necessarily very predictive. To elucidate this behavior, we investigated the genes selected by these two methods. We noticed that they tend to be systematically expressed at low levels, as shown in Figure 2.10, and that they barely depend on the labels, which explains the high stability but small accuracy.

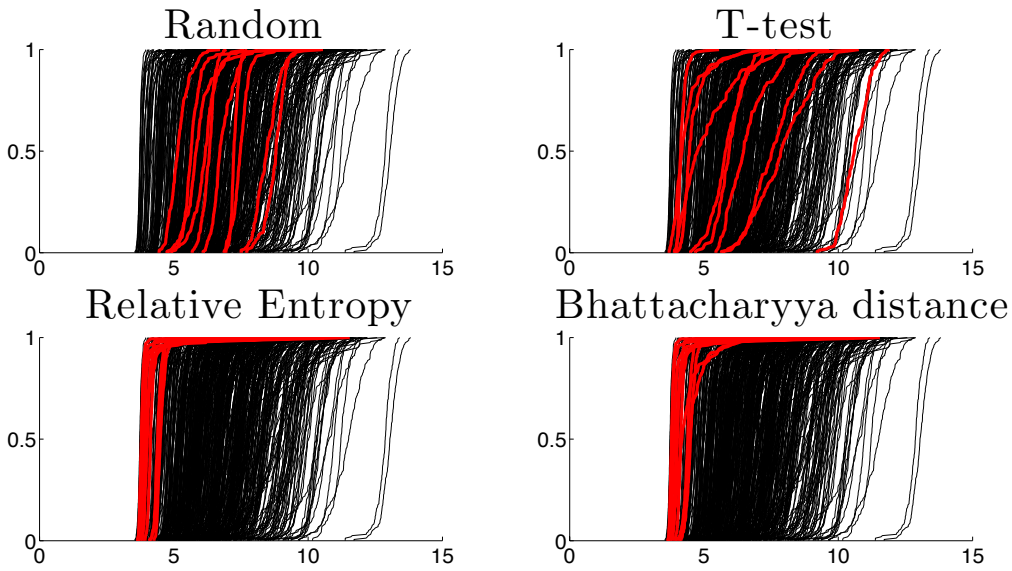


Figure 2.10: **Bias in the selection through entropy and Bhattacharyya distance** Estimated cumulative distribution functions (ECDF) of the first ten genes selected by four methods on GSE1456. They are compared to the ECDF of 500 randomly chosen background genes.

In fact the frequently selected genes systematically show a multimodal yet imbalanced distribution due to the presence of outliers, as illustrated on Figure 2.11. As soon as, by chance, one class of samples contains one or more outliers when the other class doesn't, this type of distribution is responsible for a very high variance ratio between the two classes, thus leading to a very high value of the entropy and Bhattacharyya statistics. It is therefore likely that, although stable and interpretable, the molecular signatures generated by these two methods lead to erroneous interpretation.

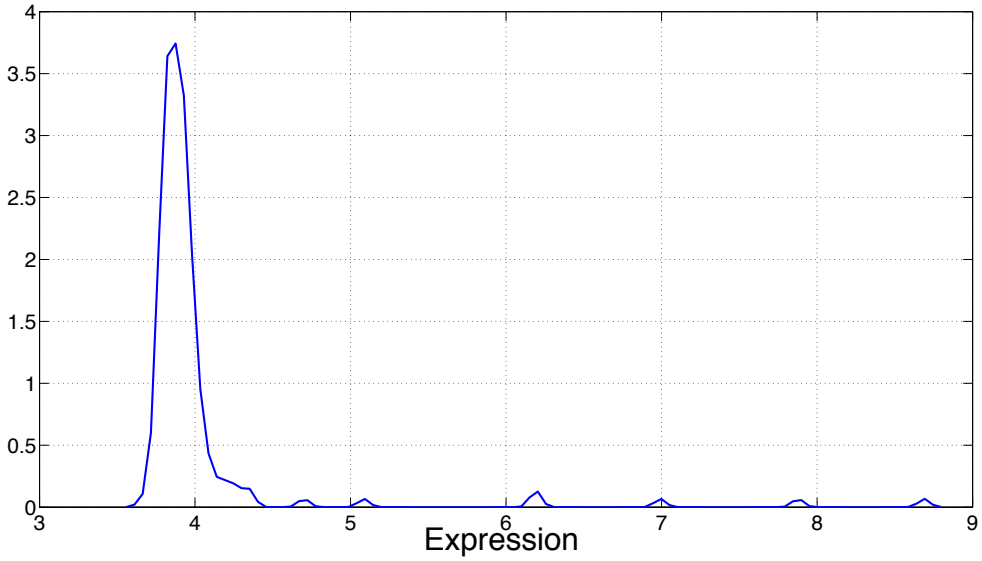


Figure 2.11: **Estimated distribution of the first gene selected by entropy and Bhattacharyya distance.**

2.4 Discussion

We compared a panel of 32 feature selection methods in light of two important criteria: accuracy and stability, both at the gene and at the functional level. Figure 2.12 summarizes the relative performance of all methods, and deserves several comments.

Taking random feature selection as a baseline, we first notice the strange behavior of gene selection by Batthacharyya distance and relative entropy: they are both more stable but less accurate than random selection. A careful investigation of the genes they select allowed us to identify that they tend to select genes with low expression levels, independently of the sample labels. This unwanted behavior can easily be fixed by pre-filtering genes with small variations, but it highlights the danger of blindly trusting a feature selection method, which in this case gives very stable and interpretable signatures.

Second, we observe that among the other methods, only elastic net, Lasso and t-test clearly seem to outperform random in terms of accuracy, and only t-test outperforms it in terms of stability. Overall, t-test gives both the best performance and the best stability. The fact that the Lasso is not stable is not surprising since, like most multivariate methods, it tries to avoid redundant genes in a signature and should therefore not be stable in data where typically many genes encode for functionally related proteins. What was less expected is that neither the elastic net, which was designed exactly to fight this detrimental property of Lasso by allowing the selection of groups of correlated genes, nor stability selection, which is supposed to stabilize the features selected by Lasso, were significantly more stable than the Lasso. In addition, we also

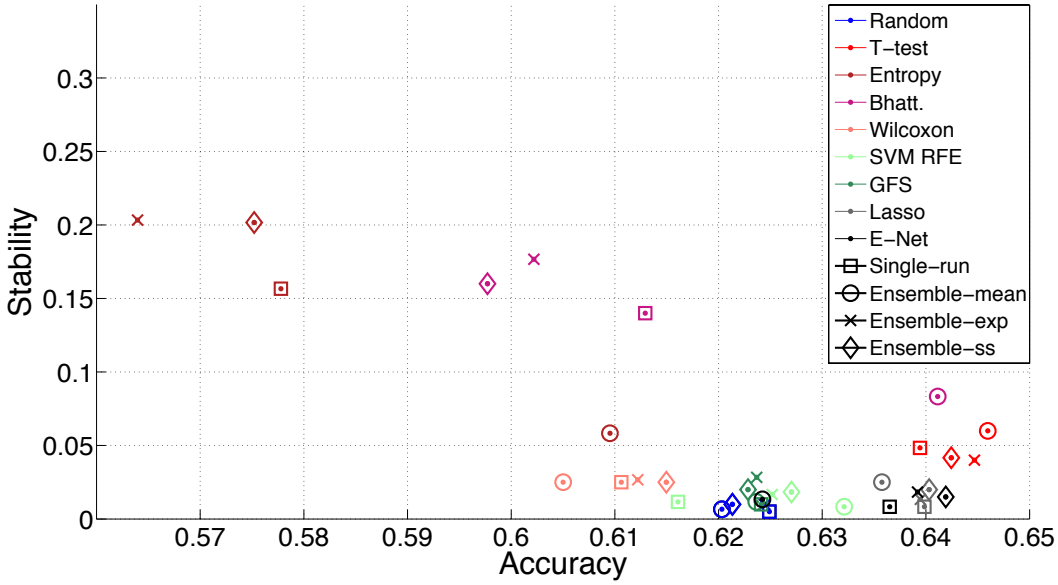


Figure 2.12: **Accuracy/stability trade-off.** Accuracy versus stability for each method in the between-datasets setting. We show here the average results over the four datasets.

found very unstable behaviors at the functional level. This raises questions about the relevance of these methods for gene expression data. Similarly, the behavior of wrapper methods was overall disappointing. SVM RFE and Greedy Forward Selection are neither more accurate, nor more stable or interpretable than other methods, while their computational cost is much higher. Although we observed like Abeel et al. (2010) that SVM RFE can benefit from ensemble feature selection, it remains below the t-test both in accuracy and stability.

Overall we observed that ensemble method which select features by aggregating signatures estimated on different bootstrap samples increased the stability of some methods in some cases, but did not clearly improve the best methods. Regarding the aggregation step itself, we advise against the use of ensemble-average, i.e. averaging the ranks of each gene over the bootstrapped lists, regardless of the selection method. Ensemble-stability selection or ensemble-exponential gave consistently better results. The superiority of the latter two can be explained by the high instability of the rankings, as discussed in Iwamoto et al. (2010).

Regarding the choice of method to train a classifier once features are selected, we observed that the best accuracy was achieved by the simplest one, namely the *nearest centroids* classifier, used, e.g., by Lai et al. (2006), Abraham et al. (2010). An advantage of this classifier is that it does not require any parameter tuning, making the computations fast and less prone to overfitting.

The performance evaluation of gene selection methods must be done carefully to prevent any *selection bias*, which could lead to underestimated error rates as discussed in Ambroise and

McLachlan (2002), Simon et al. (2003). This happens when, for example, a set of genes is selected on a set of samples, and its performance as a signature is then estimated by cross-validation on the same set. In our experimental protocol, we overcome this issue by ensuring that gene selection is never influenced by the test samples on which the accuracy is measured. In the 10-fold cross-validation setting, this means that genes are selected and the classification model is trained 10 times, on the 10 training sets. Alternatively, we also tested the performance of prognostic signature *across* datasets, where selection bias is clearly absent. We barely observed any difference between the 10-fold cross-validation setting and the setting across dataset, in terms of average accuracy, confirming that cross-validation without selection bias is a good way to estimate the generalization performance.

We noticed that evaluating the stability and the interpretability in a soft-perturbation setting may lead to untrustworthy results. The best estimation seems to be obtained in the hard-perturbation setting experiments. The lack of stability between datasets has been explained by four arguments. First data may come from different technological platforms, which is not the case here. Second and third, there are differences in experimental protocols and in patient cohorts, which is indeed the case between datasets; fourth, the small number of sample leads statistical instability. We however obtained very similar stability in the *hard-perturbation* setting (within each dataset) and in the *inter-datasets* results. This suggests that the main source of instability is not the difference in cohorts or experimental protocols, but really the statistical issue of working in high dimension with few samples.

Acknowledgments

We thank Simon Fourquet, Alexandre Hamburger, Laurent Jacob and Fabien Reyal for fruitful discussions.

Chapter 3

Importing prior knowledge: Graph Lasso for breast cancer prognosis.

This chapter has been made available online under a slightly different form (Haury et al., 2010) as joint work with Laurent Jacob and Jean-Philippe Vert.

Résumé

Dans ce chapitre, nous nous intéressons plus particulièrement à la stabilité et l'interprétabilité des signatures moléculaires pour le pronostic du cancer du sein. Nous proposons d'intégrer de l'information biologique *a priori* par le biais du *Graph Lasso*, une méthode pénalisée permettant de traiter des gènes connectés sur un réseau de manière similaire. Par ailleurs, nous proposons de randomiser l'algorithme, c'est-à-dire de construire plusieurs signatures à partir de différents sous-ensembles de données pour finalement les aggréger. Nous montrons que l'apport d'information biologique au processus de sélection permet une meilleure interprétation des signatures. La randomisation, quant à elle, semble accroître leur stabilité. En revanche, aucune de ces méthodes ne parvient à augmenter la performance prédictive des signatures, en comparaison à des modèles plus simples.

Abstract

Molecular signatures for diagnosis or prognosis estimated from large-scale gene expression data often lack robustness and stability, rendering their biological interpretation challenging. Increasing the signature's interpretability and stability across perturbations of a given dataset and, if possible, across datasets, is urgently needed to ease the discovery of important biological processes and, eventually, new drug targets. We propose a new method to construct signatures with increased stability and easier interpretability. The method uses a gene network as side

interpretation and enforces a large connectivity among the genes in the signature, leading to signatures typically made of genes clustered in a few subnetworks. It combines the recently proposed graph Lasso procedure with a stability selection procedure. We evaluate its relevance for the estimation of a prognostic signature in breast cancer, and highlight in particular the increase in interpretability and stability of the signature.

3.1 Background

In recent years a large number of diagnostic, prognostic and predictive molecular signatures have been identified through analysis of genome-wide expression profiles (see, e.g., Golub et al., 1999, Alizadeh et al., 2000, Ramaswamy et al., 2001, van de Vijver et al., 2002). Common signatures involve a few tens of genes whose expression levels allow to classify a sample in a given disease subtype, or assess its prognosis. They have been quickly adopted by the medical community for their ability to provide accurate classification and prediction, and for their direct usefulness in the clinical context. For example, the 70-gene MammaPrint signature is now marketed as a molecular diagnostic test to assess the risk of metastasis for breast cancer (van de Vijver et al., 2002).

Besides their predictive accuracy, signatures should bring useful biological information for further biomedical research, such as the identification of genes or pathways with strong prognostic power which may lead to a new understanding of the underlying biology, and eventually to the identification of new drug targets. However, the signatures proposed in different studies have generally very few genes in common, and it is now well documented that many non-overlapping signatures can have similar predictive accuracy (Ein-Dor et al., 2005). The lack of stability of signatures across datasets can also be observed within a given dataset, as signatures obtained after random perturbations of a given dataset can also have poor overlaps, i.e., lack stability (Abeel et al., 2010). An unfortunate consequence of this lack of stability is that the biological interpretation of possible functions and pathways underlying the signature is difficult *a posteriori*.

To remedy the lack of stability and the difficult interpretation of signatures, several authors have proposed to use side information, such as known biological pathways and gene networks, to analyze expression data and build signatures. For example, Chuang et al. (2007) identifies groups of connected genes in the network (subgraphs) differentially expressed between two conditions; Rapaport et al. (2007) proposed a formulation of support vector machines (SVM) to estimate a predictive model by constraining the weights of connected genes to be similar, allowing to associate positive or negative contributions to regions of the network. These approaches assume that connected genes should contribute similarly to the class prediction, by computing average expression over subnetworks or assuming similar predictive weights of connected genes; however one may argue that this is too strong an hypothesis for many networks.

Here we investigate a related question: how to estimate a molecular signature, typically of

a few tens of genes, that would be "coherent" with a given gene network given *a priori* in the sense that genes in the signature would tend to be connected to each other in the network. Note that here we do not want to constrain connected genes in the signature to have similar weights, we would just like them to be clustered in a limited number of connected components of the graph. The resulting connected components could then be more amenable to biological interpretation than individual genes, and could potentially be more stable across datasets due to the soft constraint induced on the choice of genes.

We assess the relevance of a new method named the *graph Lasso*, proposed recently by Jacob et al. (2009), to automatically learn such a signature given a training set of expression data and a gene network. The graph Lasso is an extension of the Lasso regression (Tibshirani, 1996), a widely-used and state-of-the-art method for feature selection and identification of sparse signature. In graph Lasso, the penalty used in the Lasso is modified to incorporate the gene network information, leading to the selection of features that are often connected to each other. The resulting algorithm is a convex optimization problem, whose unique solution can be found by efficient optimization methods. While the graph Lasso increases the interpretability of the signature by increasing the number of network edges between its components, it may suffer from lack of stability like many other feature selection methods including its cousin the Lasso. Recently randomization and aggregation have been proposed as a powerful way to increase the stability of feature selection methods in large dimension (Abeel et al., 2010). To further increase the stability of the graph Lasso, we propose a procedure akin to stability selection (Bach, 2008, Meinshausen and Bühlmann, 2010) in this context.

We evaluate the relevance of the resulting procedure for the estimation of a prognostic signature in breast cancer. We highlight in particular the increase in interpretability and stability resulting from the incorporation of a large gene network in the graph Lasso procedure, coupled with stability selection.

3.2 Methods

3.2.1 Learning a signature with the Lasso

Given a training set of gene expression data for p genes in n samples belonging to two classes (e.g., good and poor prognosis tumor samples), estimating a discriminative signature is a typical problem of *feature selection* for supervised classification. For example, a popular approach in bioinformatics is to select genes by ranking them according to their correlation with the class information (e.g., van 't Veer et al., 2002). Once genes are selected, it is necessary to estimate a predictive model using these genes only. In this study, we build on a different and increasingly popular approach in statistical learning where the selection of features and the estimation of a predictive model using this features are more tightly coupled. For example, one may look for a model which predicts the outcome as well as possible under the constraint of involving as few

genes as possible. A direct formulation of this joint requirement is :

$$\omega^{sig} = \arg\min_{\omega \in \mathbb{R}^p} \hat{R}_l(\omega) + \lambda \sum_{j=1}^{p} \mathbf{1}_{\{\omega_j \neq 0\}}, \tag{3.1}$$

where $\hat{R}_l$ is the empirical risk associated to a given loss function l and $\mathbf{1}_{\{\omega_j \neq 0\}}$ is 1 if parameter ω_j is non-zero, 0 otherwise, so that the second term counts the number of non-zero elements in ω. If ω contains few zeros, many genes can be involved in the prediction and it is easy to make few errors on the training data, corresponding to small values for $\hat{R}_l$. Conversely, if ω is very sparse, then is becomes more difficult to discriminate the training set correctly. The optimum ω^{sig} is a trade-off between these two extremes. The hyperparameter $\lambda \geq 0$, which must be fixed before optimization, adjusts this tradeoff : at one extreme ($\lambda = 0$) all the genes are involved in the model, and at the other extreme we obtain $\omega^{sig} = 0$ (no gene involved). Now the exact solution of problem (3.1) cannot be computed even for a reasonable number of genes, due to the combinatorial nature of the problem. This motivates the introduction of methods such as the Lasso (Tibshirani, 1996), where the second term is replaced by $\|\omega\|_1 \triangleq \sum_{j=1}^{p} |\omega_j|$. The new problem can be solved exactly, and also results in efficient feature selection.

3.2.2 The Graph Lasso

The group Lasso (Yuan and Lin, 2006) is a useful variant of the Lasso when the features are clustered into groups a priori, and one wishes to select features *by groups*. It replaces the $\|\omega\|_1$ term in the Lasso formulation by $\sum_{g \in \mathcal{G}} \|\omega_g\|$, where $\mathcal{G}$ is the set of groups of variables which should be either all zero or all non-zero. Like $\|\omega\|_1$ approximates the behavior of the count of selected genes, $\sum_{g \in \mathcal{G}} \|\omega_g\|$ approximates the count of groups which have at least one non-zero gene, and leads to solutions where several groups contain only genes at 0, which is exactly equivalent to selecting groups in $\mathcal{G}$ as long as $\mathcal{G}$ is a *partition* of the genes, *i.e.*, that each gene belongs to one and only one group.

When some genes belong to several groups, a situation which arises for example when considering gene pathways as groups, the group lasso does not result anymore in the selection of a union of groups. In Jacob et al. (2009), a generalized version of this penalty was proposed which allows to select unions of pre-defined groups which potentially overlap, *e.g.* the pathways. The overlapping group lasso penalty was empirically shown to select fewer groups than the simple Lasso, and some results were given on its statistical properties, in particular its model selection consistency.

Another interesting case which can be handled by this last penalty is when a graph is defined on the genes, for example to represent biological information such as co-regulation or protein-protein interaction. In this case, finding a signature which is formed by few connected subgraphs instead of a mere list of genes can make the solution more interpretable as it defines new gene sets which are optimal to predict the outcome (Chuang et al., 2007). To obtain this effect, one

can simply use an overlapping group lasso penalty, and define the groups to be the edges of the graph. Since the overlapping group lasso leads to solutions in which a union of groups is selected, and since a union of is more likely to form few connected subgraphs than randomly chosen genes, one can expect that the solution will tend to form connected components. This effect was observed on some simple examples in Jacob et al. (2009). Here we investigate this effect more thoroughly on an outcome prediction problem.

3.2.3 Stability selection

An issue with many feature selection methods, including the Lasso, is their lack of stability in the presence of many highly correlated features, which is to be expected with gene expression. In order to improve stability of feature selection, randomization and aggregation have been proposed as a powerful way to increase the stability of feature selection methods in large dimension (see Bach, 2008, Meinshausen and Bühlmann, 2010, Abeel et al., 2010). The general idea is to repeat the feature selection process on many randomly perturbed training sets (e.g., by bootstrapping the samples in the original training set), and to keep the features that are often selected in this procedure.

We propose a *group selection* procedure to the graph lasso algorithm based on Bach (2008), Meinshausen and Bühlmann (2010). The baseline of this procedure is shown in algorithm 4.

Input: Data $Z = (X, Y)$ divided into a training and a test sets, number of draws $ndraw$, Λ a grid

Output: Probabilities $(\Pi_j^\lambda)_{g=1...pgroups, \lambda \in \Lambda}$

for $i \in \{1...ndraw\}$ **do**
  Draw I a subsample of $\{1...n\}$ of size $[n/2]$ without replacement;
  for $\lambda \in \Lambda$ **do**
    Run a variable selection algorithm on I with regularization parameter λ;
    Store the active set $\mathcal{A}(I, \lambda)$;
  end
end
for $g \in \{1...pgroups\}$ **do**
  for $\lambda \in \Lambda$ **do**
    Compute the *selection probability* $\Pi_g^\lambda = \mathbb{P}(g \in \mathcal{A}(I, \lambda)|I)$;
  end
end

Algorithm 4: Stability Selection

This randomization-based procedure computes the proability Π_g^λ that an edge g is included in the signature for the parameter λ. Figure 3.1 illustrates these probabilities as a function of λ for each edge g. From these probability curves, Meinshausen and Bühlmann (2010) suggests

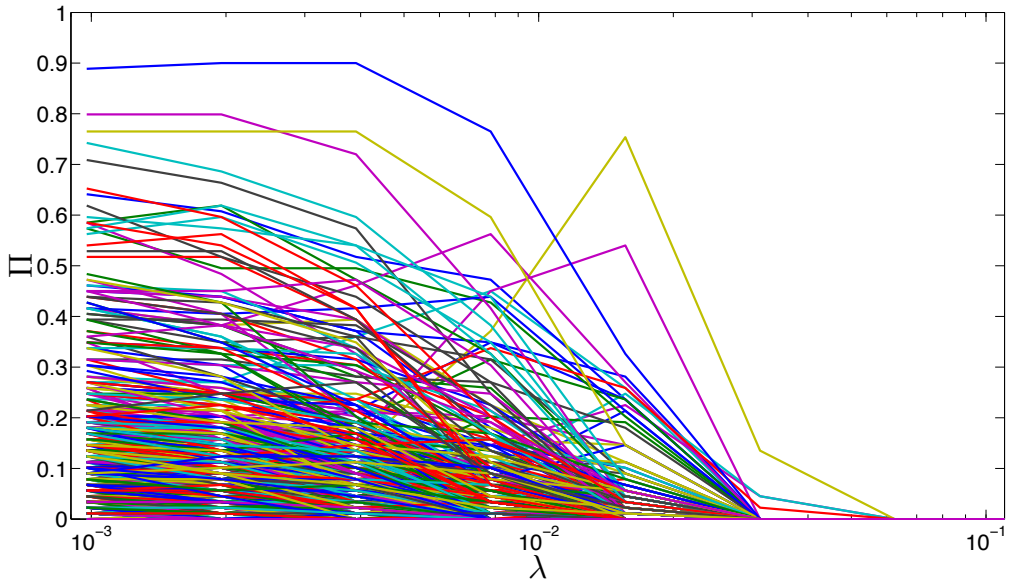


Figure 3.1: **Stability selection scores** for all edges, as a function of λ.

to select the features with the largest maximum probability over λ. While this is a nice way to select groups that are robust to the perturbations of the data, we found it hard to apply. Indeed, computation requires to fix a positive lower bound on λ and the probability for a given group to exceed the threshold increases when λ decreases, adding an extra parameter to be tuned. Therefore, we propose a slightly different way to score the groups according to their stability across the perturbations of our data. For each edge g we define the following score:

$$S_g = \max_{\lambda \in \Lambda} \left(\frac{\Pi_g^\lambda}{\sum_g \Pi_g^\lambda} \right) ,$$

which is intuitively large for a group that often enters the signature very early, while many others are not yet considered as relevant. Note that this scoring function tends to decrease when λ decreases, since more and more groups are selected. Moreover, it constitutes a way not to have to select a value for the regularization parameter. As a matter of fact, figure 3.2 shows the scores that were computed for the groups from figure 3.1. It is clear from this picture that most groups in the final signature are selected for an early λ.

Finally, we obtain a ranked list of edges by decreasing score, which allows us to define signatures of various sizes by selecting the groups whose scores are above a threshold. We note that, without stability selection, Lasso and graph Lasso also provide a ranked list of genes, in the order in which they enter the signature.

3.2.4 Preprocessing

In order to limit the computational burden and discard irrelevant genes we apply the following preprocessing steps each time a signature is built on a training set of gene expression.

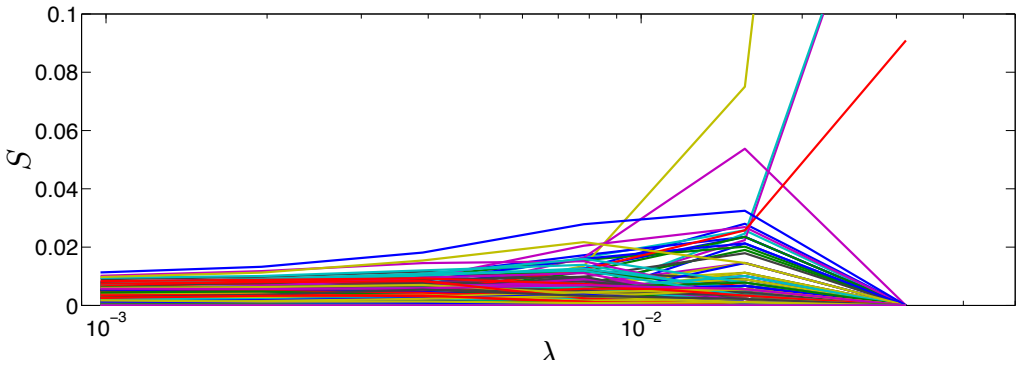


Figure 3.2: **S_g-scores for all edges**, as a function of λ.

- **Scaling.** Each gene is scaled to mean zero and variance one.
- **Outliers.** For each gene, we remove the outliers from the training set, i.e. for each gene g, the examples in set I are removed with $I = \{i, |x_{i,g}| > 1.96\}$. We then compute the correlation between the gene expression and the response.
- **Threshold.** We keep the n_g genes with the greatest correlation with the response. In practice we fix $n_g = 1500$
- **Genes kept.** Among the n_g genes, we discard those that are not connected to any other genes in the gene network. This is to ensure that all genes have the possibility to get connected when the signature is built.

3.2.5 Postprocessing and accuracy computation

Given a signature $\mathcal{A}$, we estimate a predictive model by fitting a logistic regression. The performance is estimated by 5-fold cross-validation, in terms of balanced accuracy, i.e. $(sensitivity + specificity)/2$.

3.2.6 Connectivity of a signature

To quantify whether a set of genes is connected on the gene network, we compute the following connectivity score:

$$C_{\mathcal{A}} = \frac{\text{Size of the greatest connected component}}{\text{Number of genes selected}} \tag{3.2}$$

The larger this score, the more connective the solution. The maximum score 1 is obtained if the active set consists of one and only connected component.

3.3 Data

We work on the Van't Veer breast cancer data set from van de Vijver et al. (2002), and on the Wang dataset from Wang et al. (2005), both restricted to 8,141 genes by Chuang et al. (2007).

The Van't Veer set contains 295 tumors, split into 78 metastatic and 217 non-metastatic ones, while the Wang dataset contains 286 tumors among which 106 are metastatic.

We borrow from Chuang et al. (2007) a human protein-protein interaction network comprising $57,235$ interactions among $11,203$ proteins, integrated fom yeast two-hybrid experiments, predicted interactions via orthology and co-citation, and curation of the literature.

3.4 Results

Throughout this section, we use the Lasso as a baseline method for gene selection, and are interested in the effect of using the graph information and the stability selection on three main quantities. Our first criterion is the predictive accuracy obtained by each algorithm. This accuracy is estimated by the standard 5-fold *cross-validation* procedure, where the data is split into 5 parts, and each part is used to evaluate the performance of a model which is trained on the union of the 4 others. We use the same folding in all the experiments, and make sure that the ratio of metastasic and non-metastasic prognosis is the same across the 5 parts. Second, we consider the *stability* of signatures. This involves both the stability within a dataset with respect to random perturbations of the training set, which we estimate by the number of selected genes that are common to the five folds, and the stability across two different datasets, which we estimate by comparing the signatures estimated on the Van't Veer and on the Wang datasets. Finally, we assess how connected the signature is on the biological graph of Chuang et al. (2007), as an indicator of its interpretability.

3.4.1 Preprocessing facts

Before further investigating the results, it is worth noting that after the preprocessing step where $1,500$ genes are kept in each fold, only 355 genes (connected through 901 edges) appear in the five folds after applying the procedure described in Section 3.2.4 on Van't Veer data. On the Wang dataset, this reduction is even more dramatic : only 145 genes connected by 97 edges are selected in all folds. This illustrates the high instability of the gene selection when changing even partly the set of patients on which the selection is made. This also upper-bounds the stability which is obtained by the learning algorithm, since some genes which are selected on one fold may not be present in another fold in the first place. Since the selection made in preprocessing does not follow the same criterion as the learning algorithm which selects the signature, it is technically possible that some genes would enter the signature if the preprocessing step was skipped. However, it is quite unlikely that the instability which is observed on the pre-processing procedure would be much reduced by directly using the learning algorithm.

Regarding the upcoming assessment of the stability across the datasets, it is worth pointing out that, after pre-processing, the Van't Veer and Wang datasets have only 118 genes in common, connected by 78 edges.

3.4.2 Accuracy

Figures 3.3, 3.4, 3.5 and 3.6 illustrate the 5-fold cross-validation performances on the Van't Veer dataset for the four gene selection algorithms, i.e., respectively the Lasso, the Lasso with stability selection, the graph Lasso and the graph Lasso with stability selection. We plot the balanced accuracy as a function of the size of the signature.

All curves look quite similar. For all methods, we observe that the performance degrades when signature is too small. It appears that the accuracies are overall very similar, *i.e.* neither the use of the graph information through the graph lasso penalty not the stability selection procedure significantly change the performance.

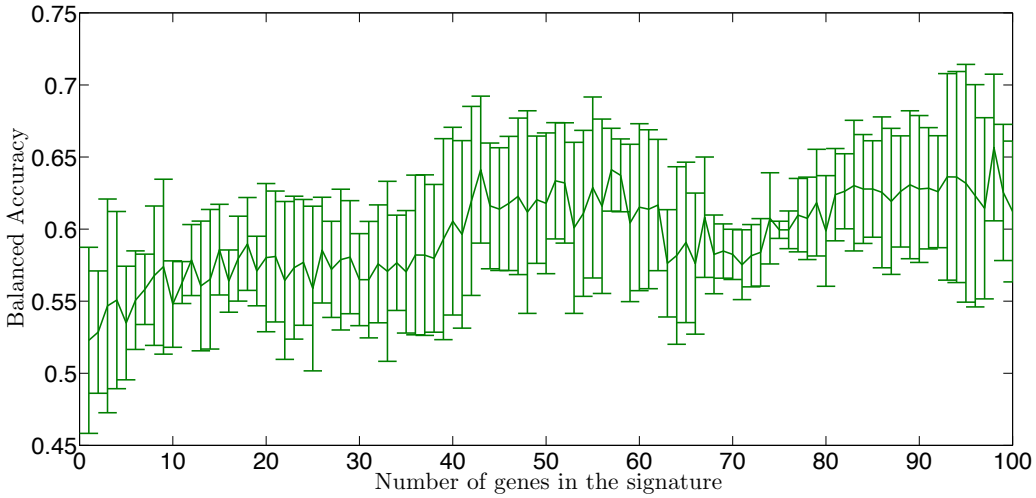


Figure 3.3: **Balanced accuracy** of the unpenalized logistic regression model trained on the signature selected by the Lasso as a function of the size of the signature.

In all cases, signatures with less than 30 genes perform the worst. However, there does not seem to be a clear number of genes that comes out as the best performer. We decide to look further into the four signatures of size 60. It seems a reasonable size according to the signatures proposed in the literature.

For each of these four signatures, we now check whether they are also a useful signature on the independent Wang dataset. We thus train four classifiers on the Wang dataset described in Section 3.3 restricted to the genes present in each of the four signatures obtained on Van't Veer dataset. We also train four classifiers using the same algorithms as the ones used to generate the signatures on the Wang dataset directly. The objective is to assess what we lose when selecting the genes on a different dataset for the four algorithms.

The results obtained are shown on Figure 3.7. They suggest that signatures estimated on the Van't Veer dataset are in fact almost as good on Wang as signatures estimated on Wang itself, if not better in the case of the graph Lasso with stability selection procedure.

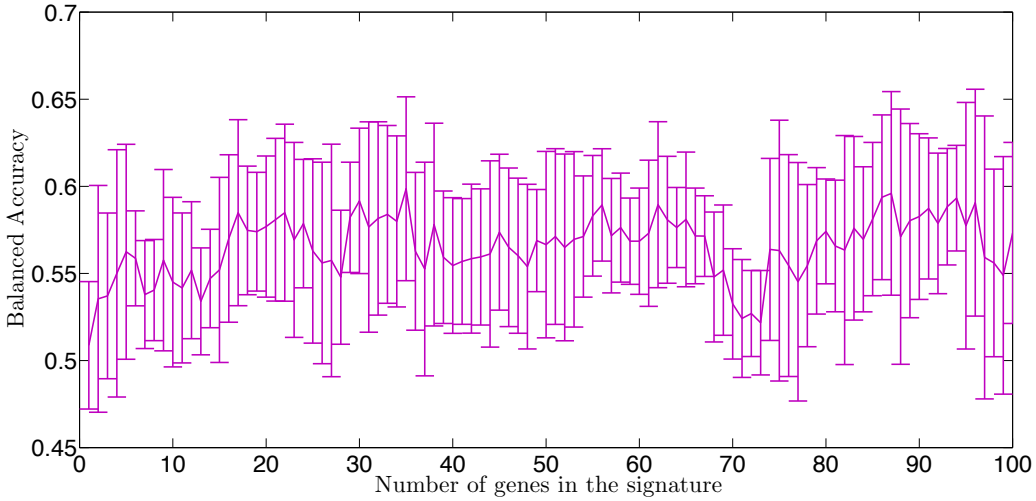


Figure 3.4: **Balanced accuracy** of the unpenalized logistic regression model trained on the signature selected by the Lasso with stability selection, as a function of the size of the signature.

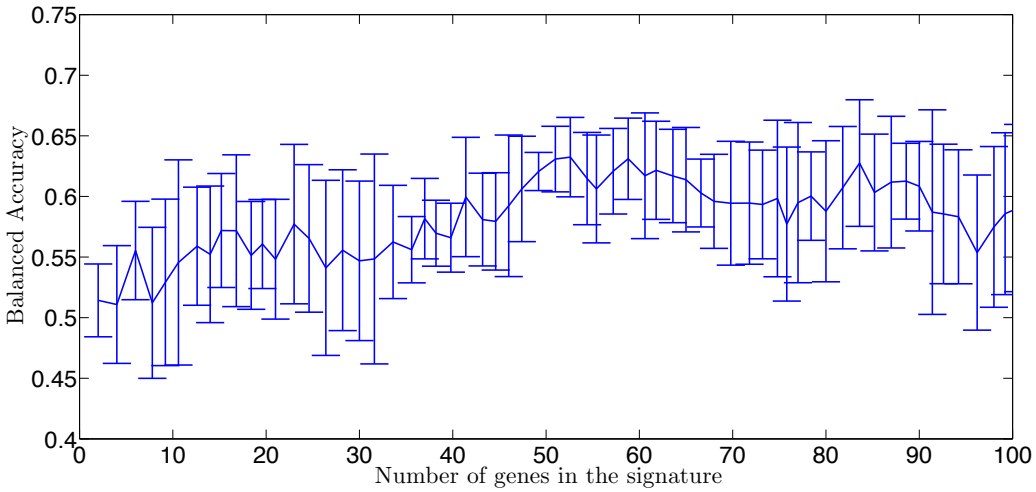


Figure 3.5: **Balanced accuracy** of the unpenalized logistic regression model trained on the signature selected by the graph Lasso as a function of the size of the signature.

3.4.3 Stability

Here we compare the stability of gene selection by the algorithms, i.e. our concerns are both the number of genes selected frequently in the five folds and the intersection of the signatures learnt on two different sets of data.

Figure 3.8 which shows how many genes are in the signatures of 1, 2, 3, 4 or 5 of the five folds, for each algorithm. A stable feature selection method should have more genes occurring five times, and less genes occurring only once.

From a stability point of view, a first improvement over the Lasso is due to the grouping

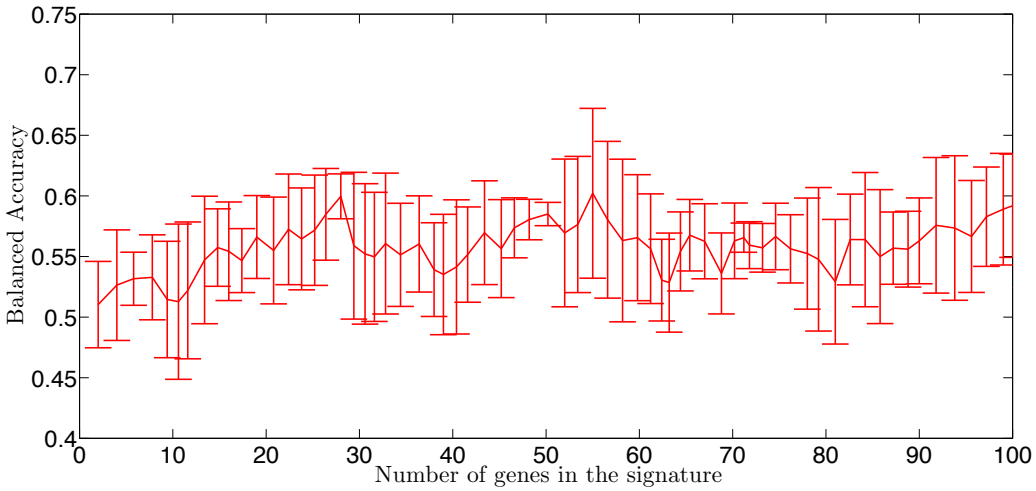


Figure 3.6: **Balanced accuracy** of the unpenalized logistic regression model trained on the signature selected by the graph Lasso with stability selection, as a function of the size of the signature.

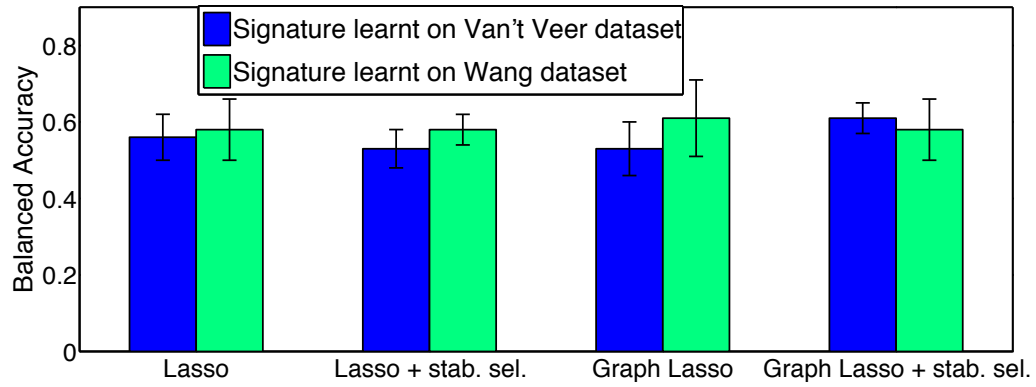


Figure 3.7: **Balanced accuracy on the Wang dataset** when selecting the genes on Wang (green) and Van't Veer (blue) datasets for the four algorithms.

of the variables, as the graph Lasso shows more overlap of more than three folds. However, it appears clearly that stability selection further improve the number of overlaps. Thus, the best stabilization performance is logically obtained by the graph Lasso with stability selection, that combines these two advantages.

Obviously, even though grouping and randomization give better stability results, the solution is still very inconsistent across folds. We believe that this might be due to the heterogeneity of our dataset, more precisely to the fact that there are different tumor subtypes which we consider altogether instead of as many as there are subtypes. However our data set does not allow us to perform such experiences as i) it contains very few samples and dividing this number may lead to very inaccurate predictions and ii) all subtypes are not well represented in the dataset.

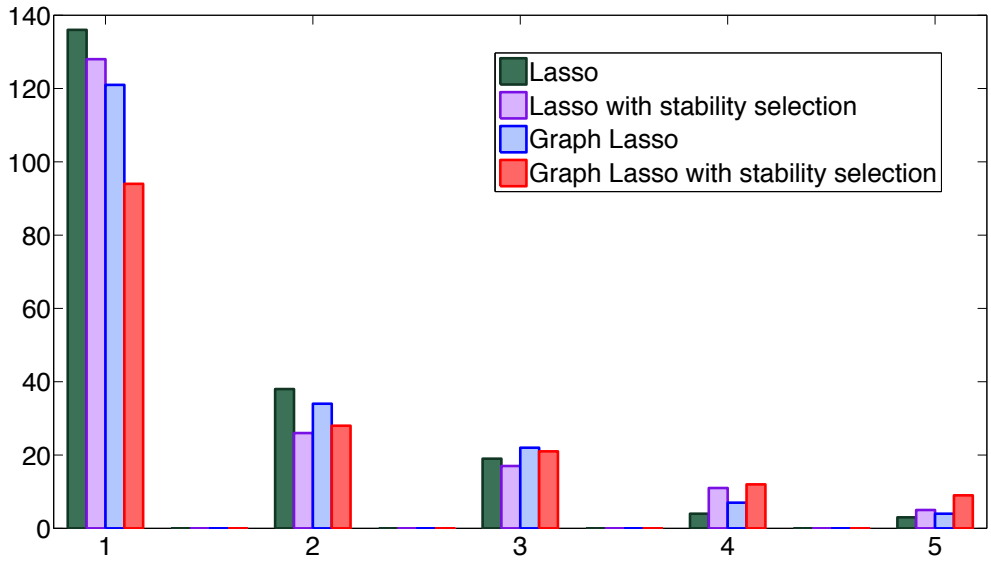


Figure 3.8: **Number of genes** present in exactly $1, 2, 3, 4$ and 5 of the 5 folds for the four algorithms.

A different question is whether these algorithms achieve an overlap between two signatures learnt on different datasets, i.e. for what we may hope in terms of reproducibility or exportability of the signatures. Figure 3.9 sheds some light on this question for it shows the number of genes found in the two signatures from Van't Veer and Wang datasets respectively. While it seems difficult to achieve overlapping with a signature smaller than a few dozen, grouping variables *a priori* still seems to be a way to improve the reproducibility. Randomization does apparently not improve this type of stability.

However, even when we do find some genes overlapping between the two signatures, there are very few of them. We believe that there could be two main explanations for this fact. First, the distribution of the tumor subtypes may be very different from a dataset to another, leading to very different overall expression patterns. Second the normalization of the data also probably plays a disrupting role for the matter of stability.

3.4.4 Connectivity

Given a graph, it may be interesting to look at the connectivity of the solution, i.e. the number and the size of the connected components induced by a selected signature. Recall that we use the scoring function defined by equation (3.2). First, it is worth noting that both the Lasso ran as a single algorithm and the Lasso with stability selection induce very low connectivity (see figure 3.10). However, it seems that using prior information from a graph, e.g. running either a group Lasso algorithm with edges as groups or that same procedure with stability selection greatly improves the connectivity. Note that using stability selection does not significantly improve the

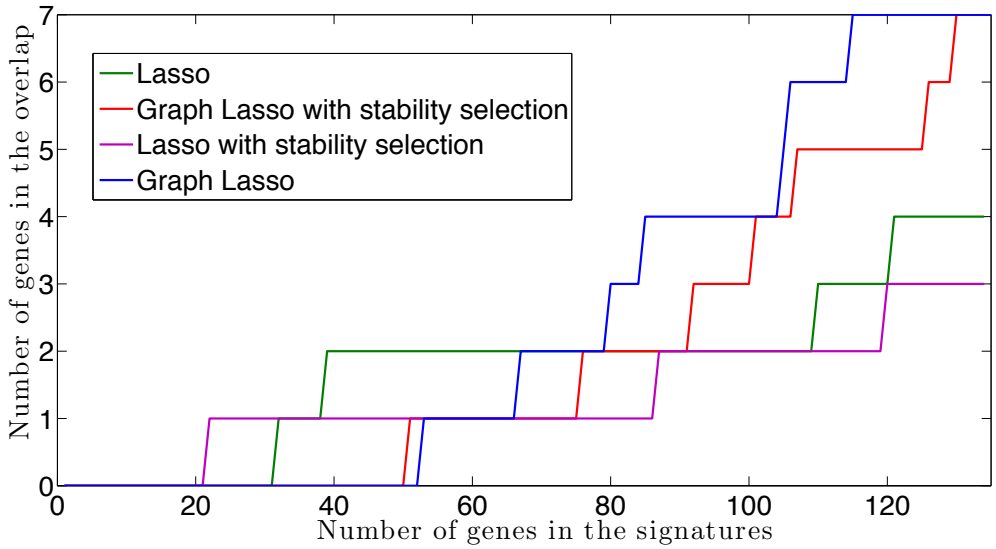


Figure 3.9: **Number of genes** present in both the signature generated on the Van't Veer and the Wang datasets, as a function of the number of genes considered in the signature.

connectivity of the solution. This suggests that mostly the prior is responsible for it i.e. the way to choose the groups, in this case as edges from a graph.

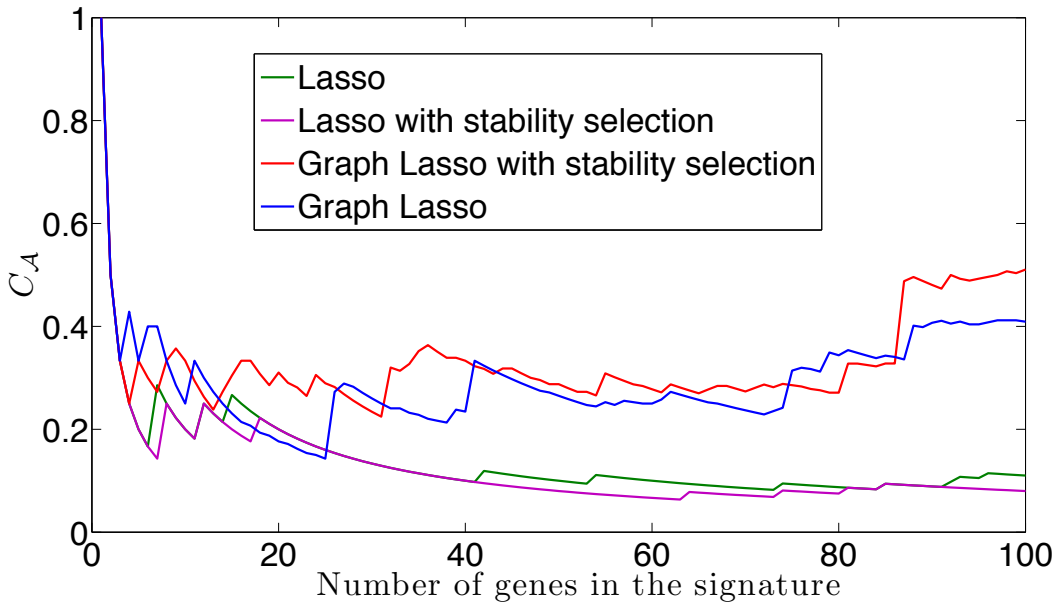


Figure 3.10: **Connectivity index** of the signatures as a function of the number of genes considered in the signature.

Figure 3.12 shows the two 60-genes signatures obtained with the graph Lasso with stability selection and the Lasso.

Obviously, the graph Lasso with stability selection provides a signature that is biologically

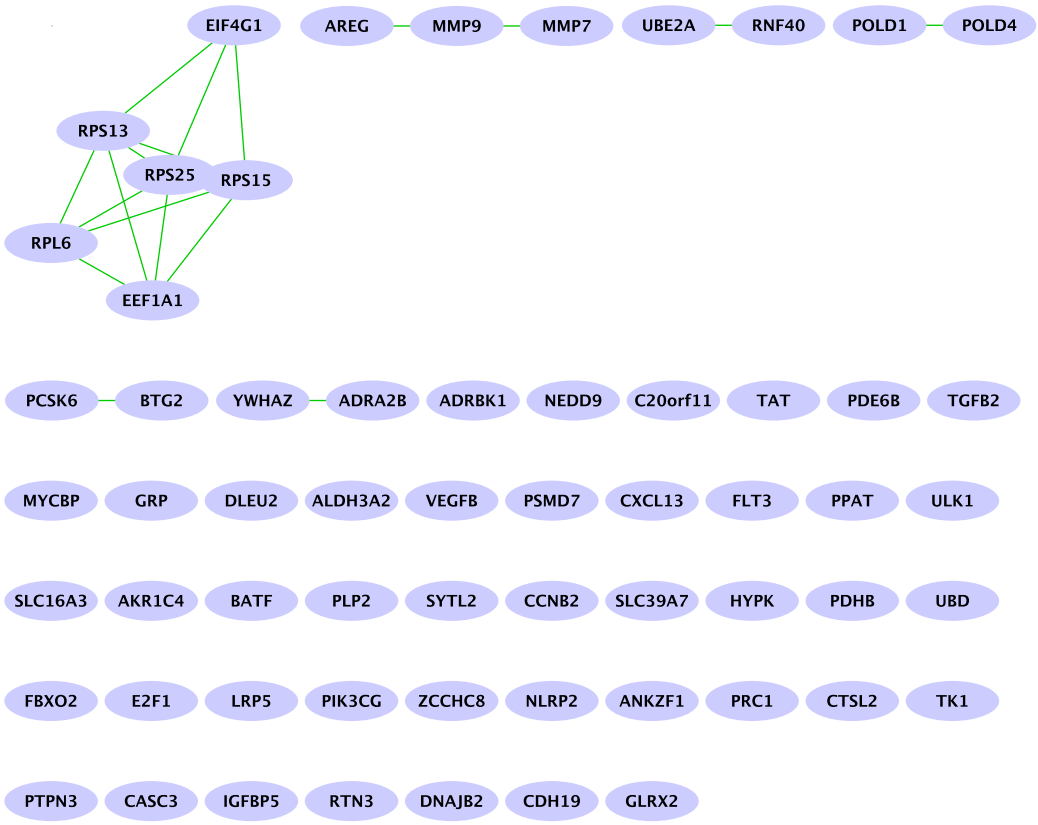


Figure 3.11: **Signature obtained with the Lasso** algorithm.

more relevant than the one chosen with the Lasso. Indeed, the connected components are related to biological processes (see section 3.4.5) and hence make more sense as a whole.

3.4.5 Biological Interpretation

Two main connected components are induced by the signature showed in figure 3.12. The largest includes 20 genes, among which 9 are ribosomal proteins. This component also includes $RAD50$ and $RAD51$, which are two known DNA repair genes that also belong to the ATMPathway (Tumor Suppressor) and the ATRBRCAPathway along with $BRCA1$ and $BRCA2$.

The second largest component almost exclusively contains genes involved in cell cycle, such as transcription factor $E2F1$, cyclins $CCNB2$ and $CCNE2$ or cell division cycle gene $CDC25B$.

Among the 29 genes left in the signature, two more are involved in cell cyle and five belong to known cancer pathways.

The second signature (from the Lasso algorithm) is harder to interpret since many genes are singletons. The largest connected component (of size 6) contains 4 genes from the ribosomes.

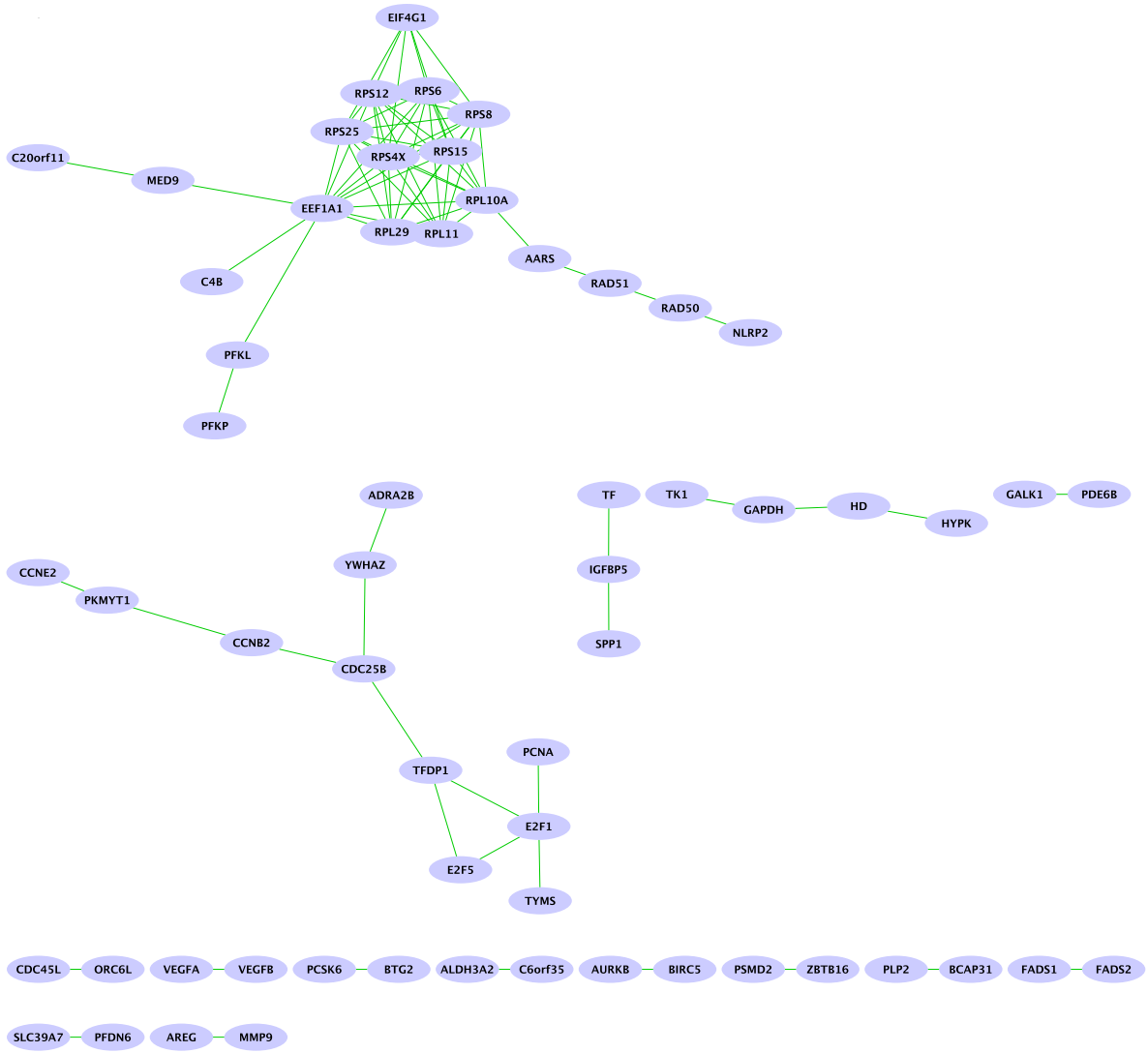


Figure 3.12: **Signature obtained with the graph Lasso** algorithm with stability selection.

6 genes in the rest of the signature are known to be involved in some cancer pathways and 4 belong to the cytokine-cytokine receptor interaction pathway. Overall, the second signature is less interpretable in terms of biological functions than the first one.

These informations were found using both the KEGG pathways and the canonical pathways from MsigDB.

3.5 Discussion

In these experiments we assessed the effect of using a biological graph and stability selection on various characteristics of the solution. A first important remark is that neither of these methods significantly improved the estimated prediction accuracy. On the one hand this is a negative result, as one could have expected that incorporating prior biological information or selecting

more stable signatures would improve the performance. On the other hand, the methods are intended to promote the connectivity of the signature on the graph and making the signature more robust to changes in the set of patients respectively. Each method seems to succeed at the task it is intended for : stability selection tends to produce more stable signatures accross the 5 folds and graph lasso outputs signatures which form a few interpretable connected components on the biological while signature given by the Lasso essentially gives a list of disconnected genes which then have to be interpreted independently. These two improvements are obtained without harming the prediction accuracy, *i.e.*, these methods allow to obtain signatures which are as effective as the one output by the Lasso with the additional benefit of being more stable and more interpretable.

We note however that the signatures obtained remain quite unstable when changing the set of patients (*e.g.* by considering the different folds). A first factor which might explain this variability is the fact that the considered datasets contain several subtypes of breast cancer tumors, some of which (*e.g.* basal versus luminal) are considered by practitioners to be distinct diseases, known to involve distinct biological processes. Finding a unique signature across these different signals may not be possible, and considering different models for the different subtypes, or a global model taking these differences into account may be a better option, although the subtypes are not strictly defined, and very few patients are available for some of them.

Another possible explanation is that there does not exist such a small set of genes which are much more involved than the others in the process of metastasis, *e.g.* that the underlying signal is not sparse at the gene level, so that small changes in the dataset give very different restricted signatures. This of course would not imply that finding a small set of genes with a good predictive power (*e.g.* to build prognosis tools) is hopeless, only that there is no "true" signature and that there is no point to looking for something stable against variations in the dataset. Even in this case, looking for signatures under some constraints which make them suitable for analysis, like the one of being connected on a pre-defined graph may uncover various important aspects of the biological process.

Chapter 4

AVENGER : Accurate Variable Extraction using the support Norm, Grouping and Extreme Randomization

Résumé

Dans ce chapitre, toujours consacré à la découverte de signatures moléculaires pour le cancer du sein, nous introduisons une nouvelle méthode. AVENGER est une méthode d'ensemble pénalisée par la norme dite "k-support". Dans ce travail, nous incorporons deux niveaux de randomisation. Premièrement, à l'instar de méthodes d'ensemble classiques, nous ré-échantillonnons les exemples par le biais du bootstrap. Deuxièmement, nous nous inspirons des forêts aléatoires en ré-échantillonnant également les variables. L'algorithme sélectionne ainsi 500 signatures différentes sur ces sous-ensembles et les aggrège, à la manière de la méthode dite de "stability selection". Par ailleurs, nous proposons l'utilisation d'une méthode d'optimisation différente pour le problème k-support, soit une version adaptative de l'algorithme "ADMM". Nous montrons que cette méthode permet une optimisation bien plus rapide du problème.

Abstract

We are concerned in this work with the selection of molecular signatures to predict breast cancer outcome. Many such signatures have been published, but two main issues arise: i) their predictive performance is limited and ii) it has been shown that they do not overlap. Indeed, no method has been able to output a list of genes that is both predictive and that stays stable when extracted from different subsamples of one single dataset. Models and algorithms have been proposed that intend to overcome this issue. They include Ensemble feature selection, structured sparsity penalization and black boxes. Using the recently introduced k-support norm, we add

two levels of randomization, bootstrapping the samples and subsampling the covariates, in order to pool the advantages that have emerged from these techniques. The resulting algorithm is named AVENGER, for Accurate Variable Extraction using the support Norm, Grouping and Extreme Randomization. Our results suggest an improvement in accuracy, as well as an enhanced stability when larger groups are selected. We also propose to use adaptive Alternating Directions Method of Multipliers (ADMM) to solve the k-support norm penalization problem and show that it significantly fastens the computation.

4.1 Background

We focus in this work on the complex problem of breast cancer outcome prediction from high-dimensional gene expression data. The aim is to compute the probability of a metastatic or relapse event taking place within five years from the first diagnosis (Sotiriou and Pusztai, 2009). On top of good prediction performance, we wish for the model to imply only a few tens of genes picked from the entire genome to build a *signature*. The reasons for enforcing *sparsity* include i) increasing the accuracy of the model, as we look to recover only relevant variables and consider the rest as noise, thus reducing over-fitting; ii) outputting a solution that is interpretable and easy for the biologists to investigate further; iii) speeding up computation by considerably reducing the dimension of the problem.

Many such signatures have been published in the past decade (see, e.g., Golub et al., 1999, Alizadeh et al., 2000, Ramaswamy et al., 2001, van de Vijver et al., 2002), predicting the outcome with some success. However, their predictive performance is limited. Moreover, it has been shown that they do not overlap, i.e. they have few genes in common, (see, e.g., Ein-Dor et al., 2005, Michiels et al., 2005, Fan et al., 2006). Research regarding this pitfall seems to concur: the small number of samples as compared to the high dimension of the problem has been identified as a major reason to this lack of *stability*. Indeed, it was shown (Ein-Dor et al., 2005, Haury et al., 2011) that no method seems to return signatures that are stable with respects to sub samplings within one single dataset. These works beg the question of the biological significance of published signatures.

Possible responses to this problem can be separated into three main ideas : i) attempt to enforce the stability of the signatures by performing Ensemble feature selection (Abeel et al., 2010, Bach, 2008, Meinshausen and Buehlmann, 2009), ii) take into account the structure of the model using group-sparsity regularization, possibly incorporating *prior knowledge* (Jacob et al., 2009, Haury et al., 2010), iii) setting aside the interpretability issue by using black box algorithms, such as Random Forests (Breiman, 2001a). Each of these methods exhibits some success, by increasing either accuracy, stability or interpretability of the signatures. Unfortunately however, no method seems to perform well over these three criteria (Haury et al., 2011).

In this work, we propose to pool these three techniques by using the k-support norm (Ar-

gyriou et al., 2012), to enforce group-sparsity, in an ensemble fashion, e.g., by repeatedly bootstrapping the samples and aggregating many signatures to build the final model. We propose to perform on top what we name *extreme randomization*, following the idea of Random Forests: the variables are also resampled at each run and the selection is thus performed over a small set only. The resulting algorithm is named AVENGER, for Accurate Variable Extraction using the support Norm, Grouping and Extreme Randomization.

In this paper, we also contribute a faster algorithm for learning with the k-support norm, namely, we implemented the Alternating Direction Methods of Multipliers (ADMM) with adaptive step. We show that it greatly improves the computation time.

4.1.1 Material and methods

4.1.2 Structured sparsity: problem and notations

Given a design matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$ (e.g., gene expression) and a binary response vector $\mathbf{Y} \in \{-1, +1\}^n$ (e.g., breast cancer outcome), our aim is to perform classification and feature selection simultaneously, i.e. to recover a sparse vector ω sufficiently informative for $\mathbf{X}\omega$ to be a good predictor of $\mathbf{Y}$. On top of its sparse nature, we assume that ω exhibits particular patterns. In the case of gene expression data, it makes sense to assume that genes are structured in groups, that can possibly overlap. Learning under this assumption is a particular case of the lately often addressed *structured sparsity* problem (see, e.g., Jenatton et al., 2009, Bach et al., 2011b, Huang et al., 2009).

In this work, we are interested in structured sparsity using a *squared penalty* Tomioka and Sugiyama (2008), i.e. in solving the following general problem:

$$\min_{\omega \in \mathbb{R}^p} \hat{R}_l(\omega) + \frac{\lambda}{2}\Omega(\omega)^2$$

where $\hat{R}_l$ is the empirical risk associated with a loss function l. In this paper, we will consider the logistic loss. Ω represents a penalty - in our case a norm - that enforces groupwise recovery of the signal.

4.1.3 The k-support norm

Recently introduced in Argyriou et al. (2012), the k-support norm provides a trade-off between Lasso (Tibshirani, 1996) and Ridge (Hoerl and Kennard, 1970) regularizations. It consists indeed in penalizing the largest weights using the ℓ_2 norms while sparsity is enforced on the remaining of the vector of weights through the ℓ_1 norm. The k-support norm is defined as:

$$\Omega_k^{sp}(w) = \left(\sum_{i=1}^{k-r-1} (|w|_i^{\downarrow})^2 + \frac{1}{r+1} \left(\sum_{i=k-r}^{p} |w|_i^{\downarrow} \right)^2 \right)^{1/2}$$

where r is the only integer in $\{0, \dots, k-1\}$ verifying

$$|w|^{\downarrow}_{k-r-1} > \frac{1}{r+1} \sum_{i=k-r}^{p} |w|^{\downarrow}_i \geq |w|^{\downarrow}_{k-r}$$

and $u^{\downarrow}_i$ is the i-th largest element of u with the convention $|u|^{\downarrow}_0 = +\infty$.

As $\Omega(.)^{sp}_1 = ||.||_1$ and $\Omega(.)^{sp}_p = ||.||_2$, the k-support norm can be viewed as a trade-off between Lasso and Ridge regressions, similarly to the Elastic Net. However, in practice, they are quite different: while the Elastic Net can be viewed as a special case of group selection with predefined groups (singletons and $\{1...p\}$ (Jenatton et al., 2009)), the k-support norm considers implicit groups whose sizes only are known.

It can also be related to the overlapping group Lasso (Jacob et al., 2009) as a special case of this penalty where groups are all subsets of $\{1...p\}$ of size up to k, as argued in Argyriou et al. (2012):

$$\Omega^{sp}_k(w) = \min \left\{ \sum_{G \in \mathcal{G}_k} ||v_G||_2 : supp(v_G) \subseteq G, \sum_{I \in \mathcal{G}_k} v_G = w \right\}. \tag{4.1}$$

Actually, we note that it is equivalent to considering all groups of size exactly k:

Proposition 1. *Let $\mathcal{G}_k$ be the set of all groups of size up to k in $\{1...p\}$ and $\mathcal{G}'_k$ the subset of these groups of size exactly k: $\mathcal{G}'_k = \{G \in \mathcal{G}_k, |G| = k\}$.*

Then, if $|supp(\omega)| \geq k$, $\forall G \in \mathcal{G}_k \setminus \mathcal{G}'_k, v_G = 0$, i.e. $\Omega^{sp}_k(\omega)$ is equivalent to minimizing (4.1) over $\mathcal{G}'_k$.

Proof. The dual norm of $\Omega^{sp}_k(.)$ is : $(\Omega^{sp}_k(\alpha))^* = \max_{G \in \mathcal{G}_k} ||\alpha_G||$ for any $\alpha \in \mathbb{R}^n$ (see Obozinski et al., 2011, Lemma 3). α_G denotes the vector whose entries are the same as α for covariates in G and 0 otherwise. Obviously, $\max_{G \in \mathcal{G}'_k} ||\alpha_G|| = \max_{G \in \mathcal{G}_k} ||\alpha_G||$ and the equality therefore stands for the primal norms. □ □

4.1.4 The primal and dual learning problems

Throughout this paper, we will refer to the primal problem as described by the following equation:

$$\min_{w \in \mathbb{R}^p} F_\lambda(w) = \min_{w \in \mathbb{R}^p} \left\{ \hat{R}_l(w) + \frac{\lambda}{2} \Omega^{sp}_k(w)^2 \right\} \tag{4.2}$$

where $\Omega^{sp}_k(.)$ describes the k-support norm from Argyriou et al. (2012).

Proposition 2. *The dual problem of (4.2) is:*

$$\max_{\alpha \in \mathbb{R}^n} F^*_\lambda(\alpha) = \max_{\alpha \in \mathbb{R}^n} -l^*(\alpha) - \frac{1}{2\lambda} \left(||X^T \alpha||^{(2)}_{(k)} \right)^2$$

where l^* defines the convex conjugate of the loss function l and $||u||^{(2)}_{(k)} = \left(\sum_{i=1}^{k} (|u|^{\downarrow}_i)^2 \right)^{\frac{1}{2}}$, with $u^{\downarrow}_i$ being the i-th largest element of u.

Proof. First note the expression of the dual norm of $\Omega_k^{sp}(.)$ for any $\alpha \in \mathbb{R}^n$:

$$(\Omega_k^{sp}(\alpha))^* = \max_{G \in \mathcal{G}'_k} ||\alpha_G|| = \left(\sum_{i=1}^{k} (|\alpha|_i^{\downarrow})^2 \right)^{\frac{1}{2}} =: ||\alpha||_{(k)}^{(2)}. \tag{4.3}$$

Problem (4.2) can be re-written as:

$$\min_{w \in \mathbb{R}^p, z \in \mathbb{R}^n} \left\{ \hat{R}_l(z) + \tfrac{\lambda}{2} \Omega_k^{sp}(w)^2 \right\} \tag{4.4}$$

$$\text{s.t.} \qquad z = X^T \omega, \tag{4.5}$$

yielding the Lagrangian:

$$\mathcal{L}(\omega, z, \alpha) = \hat{R}_l(z) + \frac{\lambda}{2} \Omega_k^{sp}(\omega)^2 + \alpha(z - X^T \omega). \tag{4.6}$$

The dual is then defined by:

$$\begin{aligned} F_\lambda^*(\alpha) &= \min_{\omega, z} \mathcal{L}(\omega, z, \alpha) \\ &= \min_{\omega} \left\{ \frac{\lambda}{2} \Omega_k^{sp}(\omega)^2 - \alpha^T X \omega \right\} + \min_{z} \left\{ \hat{R}_l(z) + \alpha z \right\} \\ &= -(\frac{\lambda}{2} (\Omega_k^{sp}(-\alpha^T X))^2)^* - l^*(\alpha) \end{aligned}$$

where f^* stands for the Fenchel conjugate of f. We use the result from Boyd and Vandenberghe (2004) stating that if $||.||$ is a norm, then $(\frac{1}{2}||u||^2)^* = \frac{1}{2}(||u||^*)^2$ where $||.||^*$ denotes the dual norm of $||.||$.

Therefore, we have:

$$\sup_w \left\{ \langle \alpha, w \rangle - \frac{\lambda}{2} \Omega_k^{sp}(w)^2 \right\} = \frac{\lambda}{2} (\Omega(\frac{1}{\lambda}\alpha)_k^{sp^*})^2 = \frac{1}{2\lambda} ||\alpha||_{(k)}^{(2)}$$

which concludes the proof. □ □

4.1.5 Optimization

We reimplemented the algorithm as described in Argyriou et al. (2012) and propose to compare it to the Alternating Direction of Method of Multipliers (see Boyd et al., 2011). The two methods rely on the computation of the *proximity operator* of the norm, which we describe first.

Note that, although the authors introduce the k-support norm as a penalization to the square loss, it can be used to regularize any convex loss function with an L-Lipschitz gradient.

Proximal algorithm for $\frac{1}{2L}(\Omega_k^{sp}(.))$

Argyriou et al. (2012) provide an algorithm for the computation of the proximity operator of $\frac{1}{2L}\Omega_k^{sp}(.)^2$, which we reproduce in Algorithm 5. L represents the Lipshitz constant of $\nabla \hat{R}_l$.

Recall that the proximity operator Moreau (1965) of a function f at some point v is defined as:

$$\text{prox}_f(v) = \arg\min_u \left\{ \frac{1}{2}||u-v||^2 + f(u) \right\}$$

Input: $v \in \mathbb{R}^p$

Output: $q = \text{prox}_{\frac{1}{2L}\Omega_k^{sp}(.)^2}(v)$

$z = |v| \downarrow, z_0 = +\infty, z_{d+1} := -\infty;$

Find $r \in \{0, \ldots, k-1\}$, $\ell \in \{k, \ldots, p\}$ such that

$$\begin{aligned} \tfrac{1}{L+1} z_{k-r-1} &> \tfrac{T_{r,\ell}}{\ell-k+(L+1)r+L+1} \geq \tfrac{1}{L+1} z_{k-r} \\ z_\ell &> \tfrac{T_{r,\ell}}{\ell-k+(L+1)r+L+1} \geq z_{\ell+1} \end{aligned}$$

where $T_{r,\ell} := \sum_{i=k-r}^{\ell} z_i;$

$$q_i \leftarrow \begin{cases} \frac{L}{L+1} z_i & \text{if } i = 1, \ldots, k-r-1 \\ z_i - \frac{T_{r,\ell}}{\ell-k+(L+1)r+L+1} & \text{if } i = k-r, \ldots, \ell \\ 0 & \text{if } i = \ell+1, \ldots, d \end{cases}$$

Reorder and change signs of q to conform with v;

return q

Algorithm 5: Proximity operator for the k-support norm

Fast Iterative Shrinkage and Thresholding Algorthim (FISTA)

FISTA (Beck and Teboulle, 2009) is the algorithm originally used to compute the k-support solution in Argyriou et al. (2012). It stands for Fast Iterative Shrinkage-Thresholding Algorithm. Iterative shrinkage-thresholding algorithms can be seen as generalizations of gradient descent algorithms to minimize functions of the form $f(x) + g(x)$ where f and g are convex but g is not necessarily differentiable, e.g. $g = ||.||_1$. This class of methods is efficient when the proximity operator can be computed easily (i.e. there exists a closed form or a fast algorithm to compute it). The improvement of FISTA over ISTA (Daubechies et al., 2004) is the introduction of an auxiliary variable (v below). It thus consists in iterative updates of three variables:

$$\begin{aligned} \theta^{(k+1)} &= \frac{1+\sqrt{1+4(\theta^{(k)})^2}}{2} \\ w^{(k+1)} &= \text{prox}_{\frac{\lambda}{2L}\Omega_k^{sp}(.)^2}\left(v^{(k)} - \frac{1}{L}\nabla \hat{R}_l(v^{(k)}) \right) \\ v^{(k+1)} &= w^{(k)} + \frac{\theta^{(k)}-1}{\theta^{(k+1)}}(w^{(k+1)} - w^{(k)}). \end{aligned}$$

Alternating Direction Method of Multipliers (ADMM)

ADMM (see, e.g., Boyd et al., 2011) is a powerful method that is well adapted to minimization problems of the form:

$$\begin{cases} \min_{x,z} f(x) + g(z) \\ \text{s.t. } Ax + Bz = c \end{cases} \tag{4.7}$$

where, as in FISTA, both f and g are convex but g need not be differentiable.

A particular case is $A = I$ which makes it possible to take advantage of a proximal computation, as we will see below. In a survey about optimization of ℓ_1-penalized models (Yang et al., 2010) showed ADMM to be the fastest algorithm of those in comparison.

We now explain how to adapt the k-support optimization problem to the ADMM framework. Problem (4.2) can be rewritten as follows:

$$\begin{cases} \min_{w,\beta \in \mathbb{R}^p} \hat{R}_l(w) + \frac{\lambda}{2}\Omega^{sp}_k(\beta)^2 \\ \text{s.t. } w = \beta \end{cases}$$

The Augmented Lagrangian (AL) can be written as:

$$\mathcal{L}_\rho(w, \beta, \mu) = \hat{R}_l(w) + \frac{\lambda}{2}\Omega^{sp}_k(\beta)^2 + \mu'(w - \beta) + \frac{\rho}{2}||w - \beta||^2$$

ADMM consists in iteratively updating the values of w, β and μ. At iteration $k+1$, the following updates are performed:

$$\begin{aligned} w^{(k+1)} &= \arg\min_w \left\{ \hat{R}_l(w) + \mu^{(k)T} w + \frac{\rho}{2}||w - \beta^{(k)}||^2 \right\} \\ \beta^{(k+1)} &= prox_{\frac{\lambda}{2\rho}\Omega^{sp}_k(.)^2} \left(w^{(k+1)} + \frac{\mu^{(k)}}{\rho} \right) \\ \mu^{(k+1)} &= \mu^{(k)} + \rho(w^{(k+1)} - \beta^{(k+1)}) \end{aligned}$$

The w-update can be performed either directly or iteratively. If a closed form is available (in the case of, e.g., the squared loss), the direct method can be used. Otherwise, one may always use a Newton or Conjugate Gradient algorithm to solve this step. In order to obtain a faster algorithm, it is possible to limit the number of internal iterations. The algorithm may need more outer iterations to converge, but still be faster.

We use a varying penalty parameter ρ. The role of ρ is to keep the primal and dual residuals within a similar amplitude. Precisely, at step $k+1$,

$$\rho^{(k+1)} = \begin{cases} (1+\tau)\rho^{(k)} & \text{if } ||w^{(k+1)} - \beta^{(k+1)}||_2 > \eta||\beta^{(k+1)} - \beta^{(k)}||_2 \text{ and } k \leq k_{max} \\ \rho^{(k)}/(1+\tau) & \text{if } \eta||w^{(k+1)} - \beta^{(k+1)}||_2 < ||\beta^{(k+1)} - \beta^{(k)}||_2 \text{ and } k \leq k_{max} \\ \rho^{(k)} & \text{otherwise} \end{cases}$$

As suggested in Boyd et al. (2011), He et al. (2000), τ can be chosen to be 1 and η to be 10 in order to keep primal and dual residual norms within a factor 10 from each other. Evidence from He et al. (2000) suggests that allowing ρ to change for the first k_{max} iterations (typically, $k = 50$ or $k = 100$) strongly fastens the convergence.

Relative Duality Gap (RDG) as a common stopping criterion

Following recommendations in Tomioka et al. (2009), we use the RDG as a stopping criterion with tolerance 10^{-2}. At each iteration k, the RDG is defined as :

$$RDG_k = \frac{F_\lambda(w^{(k)}) - F^*_\lambda(\alpha^{(k)})}{F_\lambda(w^{(k)})}. \tag{4.8}$$

Therefore, at each step, we ought to compute $\alpha^{(k)} \equiv \alpha(w^{(k)})$. In fact, a feasible value for the dual variable is $\alpha^{(k)}\nabla\tilde{l}(\mathbf{X}\omega^{(k)})$ (Mairal, 2010), where $\tilde{l}(\mathbf{X}\omega) = \hat{R}_l(\omega)$ (using either the square loss or the logistic loss for l). Note that we do not have to rescale this variable as this correspondence is always true when working with a squared penalty (Tomioka and Sugiyama, 2008).

4.1.6 AVENGER

Introducing extreme variable randomization

In a previous work (Haury et al., 2011), we showed that feature selection methods for signature selection in general lack stability, i.e. that the same method run on two different subsets of the data usually return very different solutions, making it difficult to trust the method to output a stable signature. We empirically compared several feature selection techniques, including their *ensemble variants*. Ensemble methods consist in running a method B times, each time on a different bootstrapped subsample of the training data. This class of methods returns B ranked lists of genes that should be aggregated before thresholded to the desired number of variables. One method to aggregate these lists is called *stability selection* and was introduced in slightly different forms in Bach (2008), Meinshausen and Bühlmann (2010). After the B runs, we compute the frequency with which each variable was ranked within the first k variables. This frequency stands as a score to order all variables in the final list.

Moreover, Meinshausen and Bühlmann (2010) propose what they name the *randomized Lasso*: at each bootstrap run, not only are the examples subsampled but each variable, i.e. each column j of the design matrix, is multiplied by a uniform random weight W_j, i.e., $W_j \rightsquigarrow \mathcal{U}_{[\theta,1]}$ with $0 < \theta \leq 1$ controlling the level of randomization. This method artificially modifies the chances of each variable being selected, as the ones with a lower weight W_j will be less prone to be well ranked.

Following this idea, we propose extreme randomization: at each run, we bootstrap the examples and randomly select only a small proportion of the variables, forcing the algorithm to choose between these. Formally, we draw weights $(W_j)_j$ from a Bernoulli distribution $\mathcal{B}(\pi_{rnd})$ at

each run and on top of the resampling over the examples. We can then compute the frequency of selection of j among the first q variables, conditionally on $W_j = 1$. Variables are then ranked by decreasing value of this frequency to build the final signature, which should be thresholded again to the first q variables.

One immediate advantage of this procedure is the reduction in computation time: evaluating the proximity operator scales as $\mathcal{O}(p(k + \log(p)))$ and one of those is required at each step of the ADMM (or FISTA) algorithm. Choosing π_{rnd} to be of the order of q/p, e.g., $5 \times k$ or $10 \times k$, we reduce the complexity to $\mathcal{O}(kq)$.

Extreme randomization is also designed to make the selection more accurate: assume that gene g is a so-called *hub*, i.e. that it belongs to many groups that have possibly few different genes than g in their overlap. Recovering g in this situation is not an easy task since its importance might be shadowed by the expression of all genes influenced by it. Learning with the k-support norm only will likely not return what we expect, but instead an entire group containing - and in this case depending on - g. By forcing the model to choose from a given set of variables and by resampling this set at each run, we make sure that hubs such as g will be seen in different groups and their frequency of selection over the runs should be high.

After the runs, we chose to aggregate the lists using the stability selection statistic adapted to the extreme randomization case: $s_j = \frac{1}{B}\sum_{b=1}^{B} \delta(r_j^b \leq q, j \in S_b)$, where S_b is the subset of covariates preselected for run b and r_j^b is the rank of variable j in run b.

Learning procedure

In order to select a signature of size q for each of the B bootstrap samples, we choose a decreasing sequence of values for $\lambda \in \Lambda = (2^j)_{j=20,19,\ldots,0}$ and run the ADMM algorithm, systematically initializing it at its previous value, until we reach at least q variables in the active set.

The variables are then ranked by decreasing absolute value of their weights at the end of the path. We threshold this list to the first q variables, which we call a q-gene signature.

We chose $\pi_{rnd} = 5\%$ for a problem of size $p = 12,000$, $q = 100$ and $k = 2, 5, 10, 20, 50$. We perform 500 such resamplings, bootstrapping the variables at the same time to obtain samples of the same size as the training set, i.e. we resample with replacement.

Finally, we aggregate the 500 lists using the stability selection procedure described above.

4.1.7 Accuracy and stability measures

To compute the performance in terms of accuracy and stability of the signatures, we keep the framework that we designed in Chapter 2. We evaluate the accuracy in terms of AUC in a 10-fold cross-validation setting. Our indicator for the stability of the method is the average number of variables in common between two lists obtained on two non-overlapping parts of the training data, repeated 100 times.

4.1.8 Data

Table 4.1 summarizes the attributes of the four breast cancer prognosis datasets we borrowed from Gene Expression Omnibus (Barrett et al., 2011).

Table 4.1: **Data**

Dataset name	♯ examples	♯ positives	source
GSE1456	159	40	Pawitan et al. (2005)
GSE2034	286	107	Wang et al. (2005)
GSE2990	125	49	Sotiriou et al. (2006)
GSE4922	249	89	Ivshina et al. (2006)

The four breast cancer datasets used in this study.

4.2 Results

4.2.1 Convergence

We ran ADMM for several values of k and ρ on GSE1456 restrained to $p = 1000, 2000, 5000, 10000$ variables. Figure 4.1 depicts the Relative Duality Gap (RDG) on a $\log_{10}$ scale as a function of the time for the case $p = 1,000$. Results were similar for the other values of p. Each algorithm was run to $5,000$ iterations. It is clear from this picture that either a large value of ρ or the adaptive procedure perform best. We advise for the adaptive form: although in some cases it is slightly outperformed by a fixed (large) ρ, we claim that it is a better choice across the experiments we made, regardless of the value of p.

4.2.2 Results on breast cancer data

We compared the k-support norm ($k = 2, 5, 10, 20, 50$) to t-test, Lasso and Elastic Net. Each of these methods was also adapted to its extreme randomization variant, as described in Section 4.1.6. A signature of 100 genes was selected on each dataset, using the entire set. It was then used to learn a Nearest Centroids algorithm on each of the three other datasets, in a 10-fold cross validation setting. We averaged the 10×3 resulting AUC values. The signatures were also compared to each other, in a pairwise fashion, to estimate their stability across datasets, i.e. the number of genes in common between two signatures, resulting in 6 values for each algorithm, that were averaged. Figure 4.2 shows the results from the accuracy/stability trade-off point of view.

AVENGER corresponds to the circles for the methods named "kSupport".

First, we observe that, except for kSupport with $k = 5$, embedded methods, i.e., Lasso, Elastic Net and kSupport all seem to benefit from the extreme randomization setting in terms of

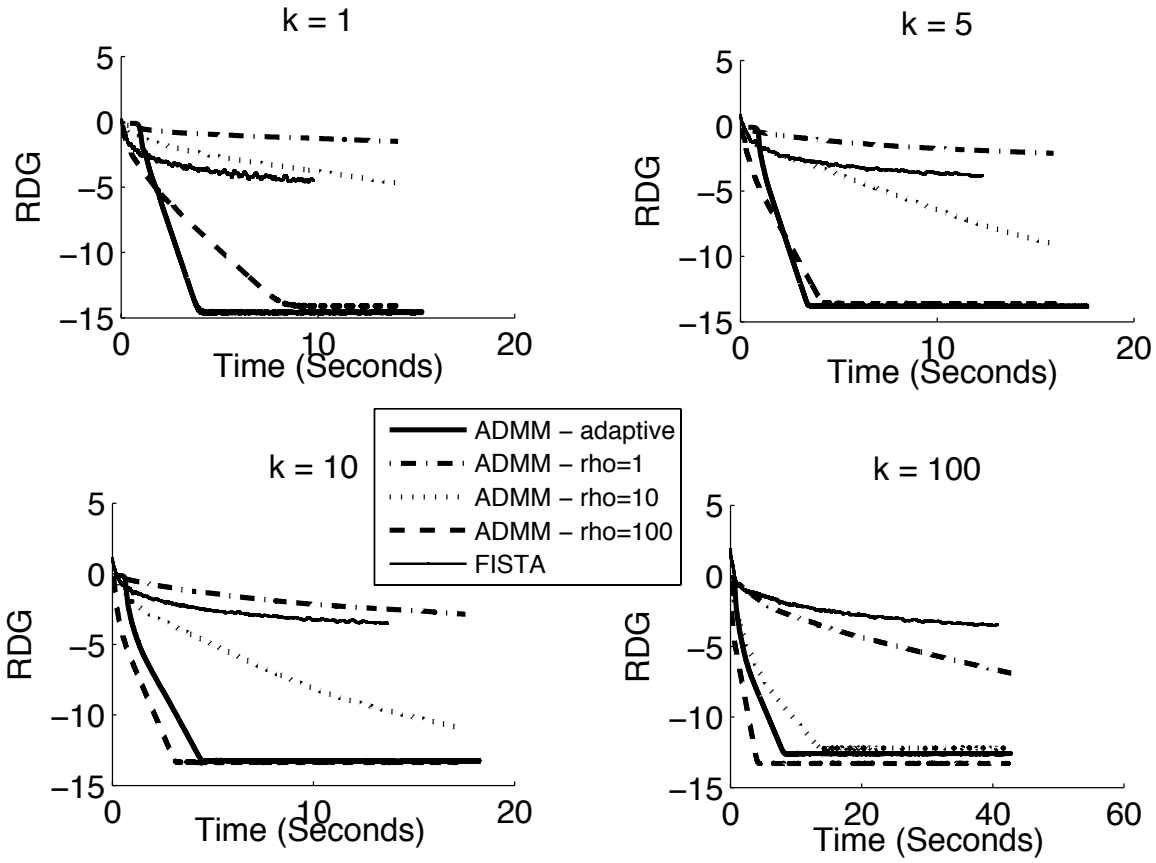


Figure 4.1: **Convergence of the algorithms.** RDG as a function of time for FISTA and different variants of ADMM. All algorithms were run during $5,000$ iterations. Although iterations of FISTA are faster, it converges much slower. Adaptive ADMM seems to be constantly a good choice.

accuracy. Moreover, stability seems to increase with the size of k (recall that Lasso is equivalent to kSupport with $k = 1$).

Figure 4.3 shows the same stability results from a different point of view. The x-axis now shows the *average absolute correlation* of the 100-selected genes, averaged over the four datasets: for each method and each dataset, we computed the correlation between each pair of genes in the signature. Then, we concatenated these correlations over the four datasets and averaged their absolute values. The resulting number is thus an indicator of *how much genes selected by a method are correlated*.

We clearly observe a strong linear relationship between correlation and stability: the more correlated the genes in the signature, the more stable the method. The way some methods are displaced in this picture is not very surprising. It is expected, for example, that Lasso would place at the bottom left and that kSupport with large values of k would find itself at the very opposite, as it is designed to select correlated genes. On the other hand, it is quite surprising to see Elastic Net at the bottom left, as it was also designed to select correlated variables.

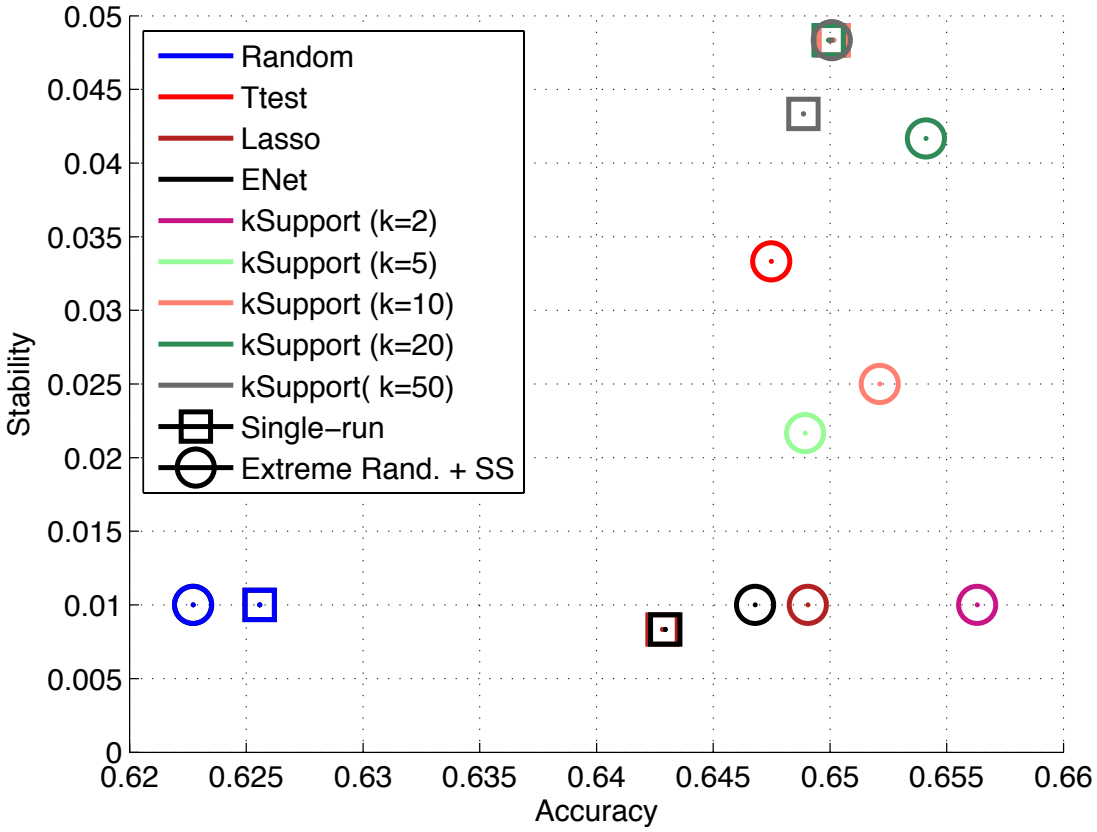


Figure 4.2: **Accuracy and stability** averaged over the four breast cancer datasets.

Interestingly, Ensemble methods generally return less correlated signatures while we noted earlier that they would increase accuracy. It is also worth noting that, e.g., stability selection for Lasso was designed with the main argument that regular Lasso did not care for correlated variables. Running it in an Ensemble setting would thus increase the chances of two similar variables to be selected. This is however not what we observe here.

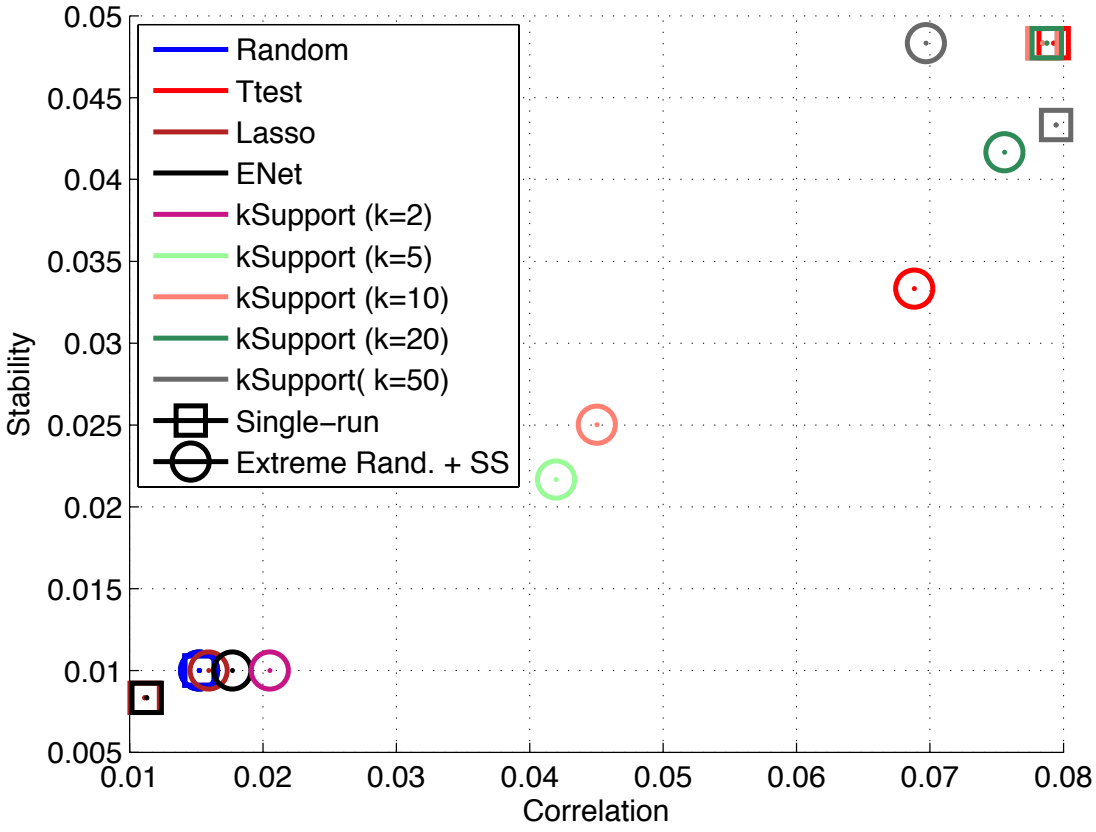


Figure 4.3: **Absolute correlation and stability** averaged over the four breast cancer datasets.

4.3 Discussion

We introduced a feature selection method named AVENGER. AVENGER is based on the k-support norm (Argyriou et al., 2012) and extreme randomization, i.e., we both bootstrap examples and resample genes at each run.

Our second contribution is a faster optimization of the k-support problem through adaptive ADMM. We showed that it outperformed FISTA by far both in computation time and in number of iterations.

Third, we implemented k-support with a logistic loss function, better suited for classification purposes.

We performed experiments on four breast cancer datasets and observed that AVENGER seems to reach the highest accuracy compared to Lasso, Elastic Net and the t-test. Regarding stability, the results are not as we hoped, as the accuracy/stability trade-off that we observed in Haury et al. (2011) keeps holding here.

It also seems that the most stable methods are the ones that select correlated or *redundant* genes. T-test and kSupport with a large value of k belong to these. This is expected for

kSupport, as when k grows, the algorithm allows larger groups to be selected. As for t-test, these results would indicate that the design of gene expression data verifies to some extent the transitivity property of the correlation (Langford et al., 2001), i.e. if X_1 is correlated to Y and X_2 is correlated to Y, it is likely that X_1 and X_2 be correlated.

On the other hand, we observe that Ensemble methods return less redundant signatures than their single-run counterparts. This is somewhat surprising as some of these methods, e.g. Stability Selection for Lasso, were designed to allow for two correlated variables to be selected.

Overall, the results of these studies seem to indicate that AVENGER yields the best accuracy results with a small value of k and the best stability results with a large value of k, similarly to the t-test.

More experiments need however to be carried out to confirm the improvements of AVENGER over other methods. Overall, the results are encouraging, suggesting that pooling randomization on the variables, resampling on the samples and structured feature selection could be a way to reach a higher accuracy.

Chapter 5

TIGRESS : Trustful Inference of Gene REgulation Using Stability Selection

This chapter has been submitted under a slightly different form as (Haury et al., 2012), a joint work with Fantine Mordelet, Paola Vera-Licona and Jean-Philippe Vert.

Résumé

Nous consacrons ce chapitre au problème d'inférence de réseaux de régulation génique à partir de données d'expression. Le mécanisme de transcription se fait par le biais de protéines nommées *facteurs de transcription*. En se fixant en amont de la séquence d'un gène cible, les facteurs de transcription activent ou répriment la transcription du gène. On parle alors de régulation. Les différents mécanismes peuvent être représentés sur un graphe dirigé appelé réseau de régulation. Ce chapitre présente une nouvelle méthode d'inférence de tels réseaux appelée TIGRESS. L'inférence du réseau est considérée comme une série de sous-problèmes, chacun consistant à prédire l'expression d'un gène-cible à partir de celles des facteurs de transcription. Nous utilisons l'hypothèse de *parcimonie* de ces réseaux : chaque gène ne serait régulé que par un nombre limité de facteurs de transcription. Nous traitons ce problème avec un algorithme de sélection de variables, LARS (régression au plus petit angle), auquel nous ajoutons différents niveaux de randomisation. C'est avec cette methode que nous avions participé au challenge d'inférence de réseaux DREAM5. Notre méthode a obtenu la seconde place sur le sous-challenge d'inférence de réseaux simulés et la troisième place dans l'ensemble. Nous évaluons l'impact de ses paramètres dans différentes circonstances, ainsi que ses limitations. Notre analyse confirme l'intérêt d'utiliser des méthodes d'apprentissage pour ce type de problème, et de TIGRESS, en particulier.

Abstract

Inferring the structure of gene regulatory networks (GRN) from a collection of gene expression data has many potential applications, from the elucidation of complex biological processes to the identification of potential drug targets. It is however a notoriously difficult problem, for which the many existing methods reach limited accuracy. In this chapter, we formulate GRN inference as a sparse regression problem and investigate the performance of a popular feature selection method, least angle regression (LARS) combined with stability selection, for that purpose. We introduce a novel, robust and accurate scoring technique for stability selection, which improves the performance of feature selection with LARS. The resulting method, which we call TIGRESS (for Trustful Inference of Gene REgulation with Stability Selection), was ranked among the top GRN inference methods in the DREAM5 gene network inference challenge. In particular, TIGRESS was evaluated to be the best linear regression-based method in the challenge. We investigate in depth the influence of the various parameters of the method, and show that a fine parameter tuning can lead to significant improvements and state-of-the-art performance for GRN inference. TIGRESS reaches state-of-the-art performance on benchmark data, including both *in silico* and *in vivo* (*E. coli* and *S. cerevisiae*) networks. This study confirms the potential of feature selection techniques for GRN inference. Code and data are available on `http://cbio.ensmp.fr/tigress`. Moreover, TIGRESS can be run online through the GenePattern platform (GP-DREAM, `http://dream.broadinstitute.org`).

5.1 Background

In order to meet their needs and adapt to changing environments, cells have developed various mechanisms to regulate the production of the thousands of proteins they can synthesize. Among them, the regulation of gene expression by transcription factors (TF) is preponderant: by binding to the promoter regions of their target genes (TG), TF can activate or inhibit their expression. Deciphering and understanding TF-TG interactions has many potential far-reaching applications in biology and medicine, ranging from the *in silico* modelling and simulation of the gene regulatory network (GRN) to the identification of new potential drug targets. However, while many TF-TG interactions have been experimentally characterized in model organisms, the systematic experimental characterization of the full GRN remains a daunting task due to the large number of potential regulations.

The development of high-throughput methods, in particular DNA microarrays, to monitor gene expression on a genome-wide scale has promoted new strategies to elucidate GRN. By systematically assessing how gene expression varies in different experimental conditions, one can try to *reverse engineer* the TF-TG interactions responsible for the observed variations. Not surprisingly, many different approaches have been proposed in the last decade to solve this GRN reverse engineering problem from collections of gene expression data. When expression data

are collected over time, for example, several methods have been proposed to construct dynamic models where TF-TG interactions dictate how the expression level of a TG at a given time allows to predict the expression levels of its TG in subsequent times (see, e.g., Arkin et al., 1997, Liang et al., 1998, Chen et al., 1999, Akutsu et al., 2000, Yeung et al., 2002, Tegner et al., 2003, Gardner et al., 2003, Chen et al., 2005, Bernardo et al., 2005, Bansal et al., 2006, Zoppoli et al., 2010). When expression data are not limited to time series, many methods attempt to capture statistical association between the expression levels of TG and candidate TF using correlation or information-theoretic measures such as mutual information (Butte et al., 2000, Margolin et al., 2006, Faith et al., 2007) or take explicit advantage of perturbations in the experiments such as gene knock-downs (Rice et al., 2005). The difficulty to separate direct from indirect regulations has been addressed with the formalism of Bayesian networks (see Friedman et al., 2000, Hartemink et al., 2002, Perrin et al., 2003, Friedman, 2004), or by formulating the GRN inference problem as a feature selection problem (Huynh-Thu et al., 2010). Mutual information-based ARACNE (Margolin et al., 2006) was also designed to eliminate redundant edges. We refer to, e.g., Markowetz and Spang (2007), Marbach et al. (2010) for detailed reviews and comparisons of existing methods.

Recent benchmarks and challenges have highlighted the good performance of methods which formalize the GRN inference problem as a regression and feature selection problem, namely, identifying a small set of TF whose expression levels are sufficient to predict the expression level of each TG of interest. The general idea that edges in a directed graph can be discovered node by node was adressed in, e.g.,Meinshausen and Bühlmann (2006). Regarding the GRN inference application, this idea underlies the Bayesian network formalism (Friedman et al., 2000), but is more directly addressed by GENIE3 (Huynh-Thu et al., 2010), a method which uses random forests to identify TF whose expression levels are predictive for the expression level of each TG, and which is now recognized as state-of-the-art on several benchmarks (Huynh-Thu et al., 2010, Marbach et al., 2010). Feature selection with random forests remains however poorly understood theoretically, and one may wonder how other well-established statistical and machine learning techniques for feature selection would behave to solve the GRN inference problem.

In this paper, we investigate the performance of a popular feature selection method, least angle regression (LARS) (Efron et al., 2004) combined with stability selection (Bach, 2008, Meinshausen and Bühlmann, 2010), for GRN inference. LARS is a computationally efficient procedure for multivariate feature selection, closely related to Lasso regression (Tibshirani, 1996). Stability selection consists in running LARS or Lasso many times, resampling the samples and the variables at each run, and in computing the frequency with which each variable was selected across the runs. We introduce a novel, robust and accurate scoring technique for stability selection, which improves the performance of feature selection with LARS. The resulting method, which we call TIGRESS (for Trustful Inference of Gene REgulation with Stability Selection), was ranked among the top GRN inference methods in the DREAM5 gene reconstruction challenge

and was evaluated to be the best linear regression- based method (Marbach et al., 2012). We furthermore investigate in depth the influence of the various parameters of the method, and show that a fine parameter tuning can lead to significant improvements and state-of-the-art performance for GRN inference. Overall this study confirms the potential of state-of-the-art feature selection techniques for GRN inference.

5.2 Methods

5.2.1 Problem formulation

We consider a set of p genes $\mathcal{G} = [1, p]$, including a subset $\mathcal{T} \subset [1, p]$ of transcription factors, among which we wish to discover direct regulations of the form (t, g) for $t \in \mathcal{T}$ and $g \in \mathcal{G}$. We do not try to infer self-regulation, meaning that for each target gene $g \in \mathcal{G}$ we define the set of possible regulators as $\mathcal{T}_g = \mathcal{T} \backslash \{g\}$ if $g \in \mathcal{T}$ is itself a transcription factor, and $\mathcal{T}_g = \mathcal{T}$ otherwise. The set of all candidate regulations is therefore $\mathcal{E} = \{(t, g), g \in \mathcal{G}, t \in \mathcal{T}_g\}$, and the GRN inference problem is to identify a subset of true regulations among $\mathcal{E}$.

For that purpose, we assume we have gene expression measurements for all genes $\mathcal{G}$ in n experimental conditions. Although the nature of the experiments may vary and typically include knock-down or knock-out experiments and even replicates, for simplicity we do not exploit this information and only consider the $n \times p$ data matrix of expression levels X as input for the GRN inference problem. Each row of X corresponds to an experiment and each column to a gene. We assume that the expression data have been pre-processed for quality control and missing values imputation.

In order to infer the regulatory network from the expression data $\mathbf{X}$, we compute a score $s : \mathcal{E} \to \mathbb{R}$ to assess the evidence that each candidate regulation is true, and then predict as true regulation the pairs $(t, g) \in \mathcal{E}$ for which the evidence $s(t, g)$ is larger than a threshold δ. We let δ as a user-controlled parameter, where larger δ values correspond to less predicted regulations, and only focus on designing a significance score $s(t, g)$ that leads to "good" prediction for some values of δ. In other words, we only focus on finding a good ranking of the candidate regulations $\mathcal{E}$, by decreasing score, such that true regulations tend to be at the top of the list; we let the user control the level of false positive and false negative predictions he can accept. Note that such a ranking is the standard prediction format of the DREAM challenge.

5.2.2 GRN inference with feature selection methods

Many popular methods for GRN inference are based on such a score. For example, the correlation or mutual information between the expression levels of t and g along the different experiments is a popular way to score candidate regulations (see, e.g., Butte et al., 2000, Margolin et al., 2006, Faith et al., 2007). A drawback of such direct approaches is that it is then difficult to separate

direct from indirect regulations. For example, if t_1 regulates t_2 which itself regulates g, then the correlation or mutual information between t_1 and g is likely to be large, although (t_1, g) is not a direct regulation. Similarly, if t_1 regulates both t_2 and g, then t_2 and g will probably be very correlated, even if there is no direct regulation between them. In order to overcome this problem, a possible strategy is to post-process the predicted regulations and try to remove regulations likely to be indirect because they are already explained by other regulations (Margolin et al., 2006). Another strategy is, given a target gene $g \in \mathcal{G}$, to *jointly* estimate the scores $s(t, g)$ for all candidate regulators $t \in \mathcal{T}_g$ simultaneously, with a method able to capture the fact that a large score for a candidate regulation (t, g) is not needed if the apparent correlation between t and g is already explained by other, more likely regulations.

Mathematically, the latter strategy is closely related to the problem of *feature selection* in statistics, as already observed and exploited by several authors, e.g., Meinshausen and Bühlmann (2006), Huynh-Thu et al. (2010). More specifically, for each target gene $g \in \mathcal{G}$, we consider the regression problem where we wish to predict the expression level of g from the expression level of its candidate regulators $t \in \mathcal{T}_g$:

$$X_g = f_g(\mathbf{X}_{\mathcal{T}_g}) + \epsilon\,, \tag{5.1}$$

where X_g represents the expression level of the g-th gene across different experiments (modelled as a random variable), $\mathbf{X}_{\mathcal{T}_g} = \{X_t\,, t \in \mathcal{T}_g\}$ is the set of expression levels of the candidate transcription factors for gene g, and ϵ is some noise. Any linear or nonlinear statistical method for regression can potentially be used to infer f_g from the observed expression data. However, we are not directly interested in the regression function f_g, but instead in the identification of a small set of transcription factors which are sufficient to provide a good model for X_g. We therefore need a score $s_g(t)$ for each candidate transcription factor $t \in \mathcal{T}_g$ to assess how likely it is to be involved in the regression model f_g. For example, if we model f_g as a linear function

$$f_g(\mathbf{X}_{\mathcal{T}_g}) = \sum_{t \in \mathcal{T}_g} \beta_{t,g} X_t\,, \tag{5.2}$$

then the score $s_g(t)$ should typically assess the probability that $\beta_{t,g}$ is non-zero (Meinshausen and Bühlmann, 2006). More general models are possible, for example Huynh-Thu et al. (2010) model f_g with a random forest (Breiman, 2001a) and score a predictor $s_g(t)$ with a variable importance measure specific to this model. Once a score $s_g(t)$ is chosen to assess the significance of each transcription factor in the target-gene-specific regression model (5.1), we can combine them across all target genes by defining the score of a candidate regulation $(t, g) \in \mathcal{E}$ as $s(t, g) = s_g(t)$, and rank all candidate regulations by decreasing score for GRN inference.

5.2.3 Feature selection with LARS and stability selection

We now propose a new scoring function $s_g(t)$ to assess the significance of a transcription factor $t \in \mathcal{T}_g$ in the regression model (5.1). Our starting point to define the scoring function is the

LARS method for feature selection in regression (Efron et al., 2004). LARS models the regression function (5.1) linearly, *i*.e. it models the expression of a target gene as a linear combination of the expression of its transcription factors, as in (5.2). Starting from a constant model where no TF is used, it iteratively adds TF in the model to refine the prediction of X_g. Contrary to classical forward stepwise feature selection (Weisberg, 1981, Hastie et al., 2009), LARS does not fully re-optimize the fitted model when a new TF is added to the model, but only refines it partially. This results in a statistically sound procedure for feature selection, akin to forward stage-wise linear regression and the Lasso (Tibshirani, 1996, Hastie et al., 2009), and a very efficient computational procedure. In practice, after L steps of the LARS iteration, we obtain a ranked list of L TF selected for their ability to predict the expression of the target gene of interest. Efficient implementations of LARS exist in various programming languages including R (`lars` package, (Efron et al., 2004)) and MATLAB (`SPAM` toolbox, Mairal et al. (2010)). Since the selection of TF is iterative, LARS has the potential to disregard indirect regulations.

The direct use of LARS to score candidate regulations has, however, two shortcomings. First, LARS can be very sensitive and unstable in terms of selected features when there exist high correlations between different explanatory variables. Second, it only provides a ranking of the TF, for each TG of interest, but does not provide a score $s_g(t)$ to quantify the evidence that a TF t regulates a target gene g. Since we want to aggregate the predicted regulations across all target genes to obtain a global ranking of all candidate regulations, we need such a score.

To overcome both issues, we do not directly score candidate regulations with the LARS, but instead perform a procedure known as *stability selection* (Meinshausen and Bühlmann, 2010) on top of LARS. The general idea of stability selection is to run a feature selection method many times on randomly perturbed data, and score each feature by the number of times it was selected. It was shown that stability selection can reduce the sensitivity of LARS and Lasso to correlated features, and improve their ability to select correct features (Bach, 2008, Meinshausen and Bühlmann, 2010). In addition, it provides a score for each feature, which can then be aggregated over different regression problems, *i.e.* different target genes in our case. More precisely, to score the candidate target genes $t \in \mathcal{T}_g$ of a given target gene g using LARS with stability selection, we fix a (large) number of iterations R, and repeat $R/2$ times the following iterations: we randomly split the experiments into two halves of equal or approximately equal size, we multiply the expression levels of the candidate transcription factors in $\mathcal{T}_g$ on each microarray by a random number uniformly sampled on the interval $[\alpha, 1]$ for some $0 \leq \alpha \leq 1$, and we run the LARS method for $L > 0$ steps on the two resulting reduced and reweighed expression matrices. We therefore perform a total of R LARS runs on randomly modified expression matrices. For each run, the result of LARS after L steps is a ranked list of L TF. After the R runs, we record for each $g \in \mathcal{G}$, $t \in \mathcal{T}_g$ and $l \in [1, L]$ the frequency $F(g, t, l)$ with which the TF t was selected by the LARS in the top l features to predict the expression of gene g. We thus obtain a final score between 0 and 1, 1 meaning that t is always selected by

LARS in the top l features to predict the expression level of g, and 0 that is is never selected. Figure 5.1 displays graphically these frequencies, for a given gene g fixed, all candidate TF in $\mathcal{T}_g$, and $l = 1, \ldots, 15$. When l increases, the frequency $F(g,t,l)$ for fixed g and t is non-decreasing because the LARS method selects increasing sets of TF at each step. In addition, since the total number of TF selected after l LARS steps is always equal to l, taking the average over the R LARS runs leads to the equality $\sum_{t \in \mathcal{T}_g} F(g,t,l) = l$, for any gene g and LARS step l.

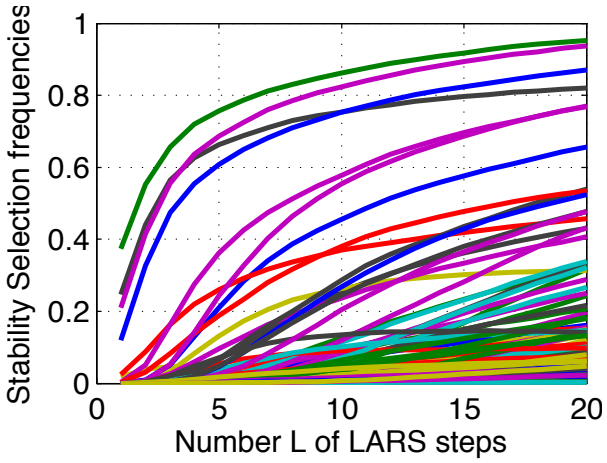


Figure 5.1: **Stability Selection :** Illustration of the stability selection frequency $F(g,t,L)$ for a fixed target gene g. Each curve represents a TF $t \in \mathcal{T}_g$, and the horizontal axis represents the number L of LARS steps. $F(g,t,L)$ is the frequency with which t is selected in the first L LARS steps to predict g, when the expression matrix is randomly perturbed by selecting only a limited number of experiments and randomly weighting each expression array. For example, the TF corresponding to the highest curve was selected 57% of the time at the first LARS step, and 81% of the time in the first two LARS steps.

Once the frequency table $F(g,t,l)$ is computed for $l = 1, \ldots, L$, we need to convert it into a unique score $s(t,g)$ for each candidate pair (t,g). The original stability selection score (Bach, 2008, Meinshausen and Bühlmann, 2010) is simply defined as the frequency of selection in the top L variables, *i.*e.,

$$s_{original}(t,g) = F(g,t,L)\,. \tag{5.3}$$

As suggested by Figure 1, this score may be very sensitive to the choice of L. In particular, if L is too small, many TF may have zero score (because there are never selected in the top L TFs), but when L is too large, several TF may have the same score 1 because they are always selected in the top L TFs. To alleviate this possible difficulty, we propose as an alternative score to measure the *area under each curve* up to L steps, *i.e.* to consider the following *area score*:

$$s_{area}(t,g) = \frac{1}{L} \sum_{l=1}^{L} F(g,t,l)\,. \tag{5.4}$$

It is worth noting that for a given target gene g, the sum of the scores over the potential

transcription factors does not depend on g. Indeed, for any fixed g, there are exactly L TF selected in the first L LARS steps on any randomly modified expression matrix, which implies that the frequencies of selection also sum to L:

$$\sum_t s_{original}(t,g) = \sum_t F(g,t,L) = L\,.$$

Moreover, the area score is also normalized as follows:

$$\sum_t s_{area}(t,g) = \sum_t \frac{1}{L}\sum_{l=1}^{L} F(g,t,l) = \frac{1}{L}\sum_{l=1}^{L} l = \frac{L+1}{2}\,.$$

This shows that the scores output by TIGRESS are naturally normalized per target gene, and we therefore do not consider further normalization before aggregating all scores together across target genes.

The difference between $s_{original}(t,g)$ and $s_{area}(t,g)$ becomes clear if we consider the rank of t in the list produced by LARS in one run as a random variable H_t (with $H_t = 1$ meaning that t is ranked first by LARS). $F(g,t,l)$ is then, by definition, the empirical probability $P(H_t \leq l)$ that H_t is not larger than l. The original score has therefore an obvious interpretation as $P(H_t \leq L)$, which we can rewrite as:

$$s_{original}(t,g) = E\left[\phi_{original}(H_t)\right] \quad \text{with} \quad \phi_{original}(h) = \begin{cases} 1 & \text{if } h \leq L\,, \\ 0 & \text{otherwise.} \end{cases}$$

Interestingly a small computation shows that the area score has a similar probabilistic interpretation:

$$\begin{aligned} s_{area}(t,g) &= \sum_{l=1}^{L} F(g,t,l) \\ &= \sum_{l=1}^{L} P(H_t \leq l) \\ &= \sum_{l=1}^{L}\sum_{h=1}^{l} P(H_t = h) \\ &= \sum_{h=1}^{L} (L+1-h) P(H_t = h) \\ &= E\left[\phi_{area}(H_t)\right]\,, \end{aligned}$$

with

$$\phi_{area}(h) = \begin{cases} L+1-h & \text{if } h \leq L\,, \\ 0 & \text{otherwise.} \end{cases}$$

In other words, both the original and the area scores can be expressed as $E\left[\phi(H_t)\right]$, although with a different function ϕ. While the original score only assesses how often a feature ranks in

the top L, the area score additionally takes into account the value of the rank, with features more rewarded if they are not only in the top L but also frequently with a small rank among the top L. Since s_{area} integrates the frequency information over the full LARS path up to L steps, it should be less sensitive than $s_{original}$ to the precise choice of L, and should allow to investigate larger values of L without saturation effects when several curves hit the maximal frequency of 1. We note that other scores of the form $E\left[\phi(H_t)\right]$ for non-increasing function ϕ could be investigated as well.

5.2.4 Parameters of TIGRESS

In summary, the full procedure for scoring all candidate edges in $\mathcal{E}$, which we call TIGRESS, splits the GRN inference problem into p independent regression problems taking each target gene $g \in \mathcal{G}$ in turn, and scores each candidate regulation (t, g) for a candidate TF $t \in \mathcal{T}_g$ with the original (5.3) or area (5.4) stability score applied to LARS feature selection. In addition to the choice of the scoring method (original or area), the parameters of TIGRESS are (i) the number of runs R performed in stability selection to compute the scores, (ii) the number of LARS steps L, and (iii) the parameter $\alpha \in [0, 1]$ which controls the random re-weighting of each expression array in each stability selection run. Apart from R that should be taken as large as possible to ensure that frequencies are correctly estimated, and is only limited by the computational time we can afford to run TIGRESS, the influence of α and L on the final performance of the method are not obvious. Taking $\alpha = 1$ means that no weight randomization is performed on the different expression arrays, while $\alpha = 0$ leads to maximal randomization. Meinshausen and Bühlmann (2010) advocate that a value between 0.2 and 0.8 is often a good choice. Regarding the choice of L, Meinshausen and Bühlmann (2010) mentions that it has usually little influence on the result, but as discussed above, the choice of a good range of values may not be trivial in particular for the original score. We investigate below in detail how the performance of TIGRESS depends on the scoring method and on these parameters R, α and L.

5.2.5 Performance evaluation

We experimentally compare TIGRESS to several other GRN inference methods. We use the MATLAB implementations of CLR (Faith et al., 2007) and GENIE3 (Huynh-Thu et al., 2010). We run ARACNE (Margolin et al., 2006) using the R package `minet`. We keep default parameter values for each of these methods. Results borrowed from the DREAM5 challenge (Marbach et al., 2012) were directly obtained by each participating team.

Given a gene expression data matrix, each GRN inference method outputs a ranked list of putative regulatory interactions. Taking only the top K predictions in this list, we can compare them to known regulations to assess the number of true positives (TP, the number of known regulations in the top K predictions), false positives (FP, the number of predicted regulations

in the top K which are not known regulations), false negatives (FN, the number of known interactions which are not in the top K predictions) and true negatives (TN, the number of pairs not in the top K predictions which are not known regulations). We then compute classical statistics to summarize these numbers for a given K, including precision ($TP/(TP+FP)$), recall ($TP/(TP+FN)$), and fall-out ($FP/(FP+TN)$). We assess globally how these statistics vary with K by plotting the receiver operating characteristic (ROC) curve (recall as a function of fall-out) and the precision-recall curve (precision as a function of recall), and computing the area under these curves (respectively AUROC and AUPR) normalized between 0 and 1.

For the datasets of DREAM5, we further compute a P-value for the AUROC and AUPR scores, based on all DREAM5 participants' predictions. This method, which was used by the DREAM5 organizers to rank the teams, involves randomly drawing edges from the teams' prediction lists and computing the probabilities of obtaining an equal or larger AUPR (resp. AUROC) by chance. More precisely, random lists are constructed as follows: for each row of the predicted list, an edge at the same position is drawn at random from all predictions. For an ensemble of such random lists, the areas under the curves are computed, allowing to estimate a random distribution. P-values were obtained by extrapolating the resulting histogram. We refer to Marbach et al. (2012) for more details on this scoring scheme. Finally, we compute the so-called *overall score* for a GRN inference method by integrating the AUROC and AUPR P-values as follows:

$$\text{overall score} = \frac{1}{2}\ln(p_{AUPR}\, p_{AUROC})\,. \tag{5.5}$$

5.3 Data

We evaluate the performance of TIGRESS and other GRN inference methods on four benchmark datasets, each consisting of a compendium of gene expression data, a list of known TF, and a gold standard set of verified interactions which we ideally would like to recover from the expression data only. Expression data are either simulated or experimentally measured under a wide range of genetic, drug and environmental perturbations. Table 5.1 summarizes the statistics of these four networks.

Network	**♯ TF**	**♯ Genes**	**♯ Chips**	**♯ Verified interactions**
DREAM5 Network 1 (in-silico)	195	1643	805	4012
DREAM5 Network 3 (*E. coli*)	334	4511	805	2066
DREAM5 Network 4 (*S. cerevisiae*)	333	5950	536	3940
E. coli Network from Faith et al. (2007)	180	1525	907	3812

Table 5.1: **Datasets:** the four datasets used in our experiments.

The first three benchmarks are taken from the DREAM5 challenge (Marbach et al., 2012).

Network 1 is a simulated dataset. Its topology and dynamics were modelled according to known GRN, and the expression data were simulated using the *GeneNetWeaver* software (Schaffter et al., 2011). We refer the interested reader to Marbach et al. (2009, 2010) for more information on this network. The second and third benchmarks are Network 3 and Network 4 of the DREAM5 competition, corresponding respectively to real expression data for *E. coli* and *S. cerevisiae*. Note that we do not use in our experiments Network 2 of DREAM5, because no verified TF-TG interaction is provided for this dataset consisting in expression data for *S. aureus*.

Additionally, we run experiments on the *E. coli* dataset from Faith et al. (2007), which has been widely used as a benchmark in GRN inference literature. The expression data was downloaded from the Many Microbe Microarrays (M^{3D}) database (Faith et al., 2008) (version 4 build 6). It consists in 907 experiments and 4297 genes. We obtained the gold standard data from RegulonDB (Gama-Castro et al., 2011) (version 7.2, May 6th, 2011) that contains 3812 verified interactions among 1525 of the genes present in the microarrays experiments.

As a pre-processing step, we simply mean-center and scale to unit variance the expression levels of each gene within each compendium.

5.4 Results

5.4.1 DREAM5 challenge results

In 2010 we participated to the DREAM5 Network Inference Challenge, an open competition to assess the performance of GRN methods (Marbach et al., 2012). Participants were asked to submit a ranked list of predicted interactions from four matrices of gene expression data. At the time of submission, no further information was available to participants (besides the list of TF), in particular the "true" network of verified interactions for each dataset was not given. After submissions were closed, the organizers of the challenge announced that one network (Network 1) was a simulated network with simulated expression data, while the other expression datasets were real expression data collected for *E. coli* (Network 3) and *S. cerevisiae* (Network 4), respectively. Teams were ranked for each network by decreasing overall score (5.5), and an overall ranking was proposed based on the average of the overall scores over the three networks.

We submitted predictions for all networks with a version of TIGRESS which we did not try to optimize, which we refer to as *Naive TIGRESS* below. Naive TIGRESS is the variant of TIGRESS which scores candidate interactions with the original score (5.3) and uses the arbitrarily fixed parameters $\alpha = 0.2$, $L = 5$, $R_1 = 4,000$, $R_3 = R_4 = 1,000$, where R_i refers to the number of runs for network i. The number of runs were simply set to ensure that TIGRESS would finish within 2 days on a single-core laptop computer. R_1 is larger than R_3 and R_4 because the size of network 1 is smaller than that of networks 3 and 4, implying that each TIGRESS run is faster. The choice $\alpha = 0.2$ followed previous suggestions for the use of stability selection (Meinshausen and Bühlmann, 2010), while the choice $L = 5$ roughly corresponded to the largest

value for which no TF-TG pair had a score of 1.

Naive TIGRESS was among the top GRN prediction methods at DREAM5, ranking second among 29 participating teams in the *in silico* network challenge, and third overall. Table 5.2 summarizes the results of the first three teams as well as two state-of-the-art methods in average overall score.

Method	Network 1			Network 3			Network 4			Overall
	AUPR	*AUROC*	*Score*	*AUPR*	*AUROC*	*Score*	*AUPR*	*AUROC*	*Score*	
GENIE3 Huynh-Thu et al. (2010)	0.291	0.815	104.65	0.093	0.617	14.79	0.021	0.518	1.39	40.28
ANOVerence Küffner et al. (2012)	0.245	0.780	53.98	0.119	0.671	45.88	0.022	0.519	2.21	34.02
Naive TIGRESS	**0.301**	**0.782**	**87.80**	**0.069**	**0.595**	**4.41**	**0.020**	**0.517**	**1.08**	**31.1**
CLR Faith et al. (2007)	0.255	0.773	55.02	0.075	0.590	5.29	0.021	0.516	1.07	20.46
ARACNE Margolin et al. (2006)	0.187	0.763	24.47	0.069	0.572	3.24	0.018	0.504	1.1e-4	9.24
TIGRESS	**0.320**	**0.789**	**105.28**	**0.066**	**0.589**	**3.25**	**0.020**	**0.514**	**0.46**	**36.33**

Table 5.2: **DREAM5 networks results:** AUPR, AUROC and minus the logarithm of related p-values for all DREAM5 Networks and state-of-the-art methods.

The winning method, both *in silico* and overall, was the GENIE3 method of Huynh-Thu et al. (2010). GENIE3 already won the DREAM4 challenge, confirming its overall state-of-the-art performance. It had particularly strong performance on the *in silico* network, and more modest performance on both *in vivo* networks. The ANOVA-based method of Küffner et al. (2012) ranked second overall, with particularly strong performance on the *E. coli* network. Naive TIGRESS ranked third overall, with particularly strong performance on the *in silico* network, improving over GENIE3 in terms of AUPR.

Interestingly, GENIE3 and TIGRESS follow a similar formulation of GRN inference as a collection of feature selection problems for each target gene, and use similar randomization-based techniques to score the evidence of a candidate TF-TG interaction. The main difference between the two methods is that GENIE3 aggregates the features selected by decision trees, while TIGRESS aggregates the features selected by LARS. The overall good results obtained by both methods suggest that this formalism is particularly relevant for GRN inference.

5.4.2 Influence of TIGRESS parameters

In this section, we provide more details about the influence of the various parameters of TIGRESS on its performance, taking DREAM5 *in silico* network as benchmark dataset. Obviously the improvements we report below would require confirmation on new datasets not used to optimize the parameters, but they shed light on the further potential of TIGRESS and similar regression-based method when parameters are precisely tuned.

Starting from the parameters used in Naive TIGRESS ($R = 4,000$, $\alpha = 0.2$ and $L = 5$, original score), we assess the influence of the different parameters by systematically testing the following combinations:

- original (5.3) or area (5.4) scoring method;

- randomization parameter $\alpha \in \{0, 0.1 \ldots, 1\}$;
- length of the LARS path $L \in \{1, 2 \ldots 20\}$;
- number of randomization runs $R \in \{1,000; 4,000; 10,000\}$.

Figure 5.2 summarizes the overall score (5.5) obtained by each combination of parameters on Network 1.

A first observation is that the *area* scoring method consistently outperforms the *original* scoring method, for any choice of α and L. This suggests that, by default, the newly proposed area score should be preferred to the classical original score. We also note that the performance of the area score is less sensitive to the value of α or L than that of the original score. For example, any value of α between 0.2 and 0.8, and any L less than 10 leads to an overall score of at least 90 for the area score, but it can go down to 60 for the original score. This is a second argument in favor of the *area* scoring setting: as it is not very sensitive to the choice of the parameters, one may practically more easily tune it for optimal performance. On the contrary, the window of (α, L) values leading to the best performance is more narrow with the original scoring method, and therefore more difficult to find *a priori*. The recommendation of Meinshausen and Bühlmann (2010) to choose α in the range $[0.2, 0.8]$ is clearly not precise enough for GRN inference. The best overall performance is obtained with $(\alpha = 0.4, L = 2)$ in both scoring settings.

Regarding the relationship between α and L, we observe in Figure 2 a slight positive correlation for the optimal L as a function of α, particularly for the area score. For example, for $R = 10^4$, $L = 2$ is optimal for $\alpha \leq 0.4$, but $L \geq 4$ is optimal for $\alpha \geq 0.8$. The effect is even more pronounced for $R = 4,000$. This can be explained by the fact that when α increases, we decrease the variations in the the different runs of LARS and therefore reduce the diversity of features selected; increasing the number of LARS is a way to compensate this effect by increasing the number of features selected at each run. Another way to observe the need to ensure a sufficient diversity is to observe how the best parameters L and α vary as a function of R (Figure 5.3). It appears clearly that the optimal number of steps L^* decreases when the number of resampling runs increases and stabilizes at 2. This is not a surprising result. Indeed, when more resampling is performed, the chance of selecting a given feature increases. The number N of non zero scores subsequently increases and it thus becomes unnecessary to look further in the regularization path. On the other hand, the value of α^* lies steadily between 0.3 and 0.5, suggesting that the adjustment to the number of bootstraps can mostly be made through the choice of L.

Furthermore, we unsurprisingly observe that increasing the number R of resampling runs leads to better performances. On Figure 5.4, we show the overall score as a function of R with $L = 2$ and $\alpha = 0.4$. We clearly see that, for both scoring methods, increasing the number of runs is beneficial. The performance seems to reach an asymptote only when R becomes larger than $5,000$.

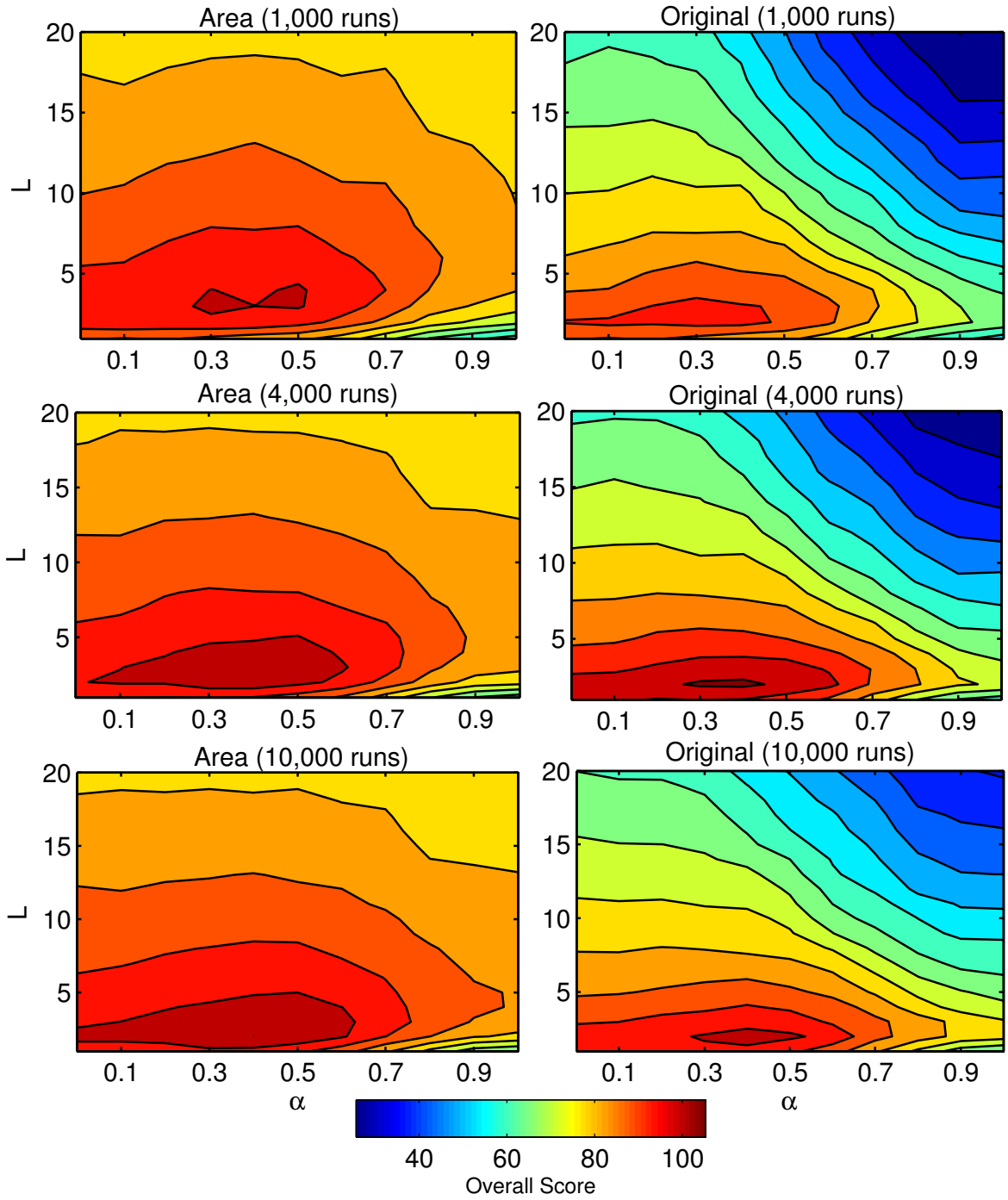


Figure 5.2: **Overall Score for Network 1:** Top plots show the overall score for $R = 4,000$ and bottom plots depict the case $R = 10,000$ for both the area (left) and the original (right) scoring settings, as a function of α and L.

Finally, we were interested in the number of TFs selected per gene. Figures 5.5 and 5.6 show how the distribution of this number changes with respect to the total number of predictions for $L = 2$ and $L = 20$ respectively. We observe a lower variance and a larger median when

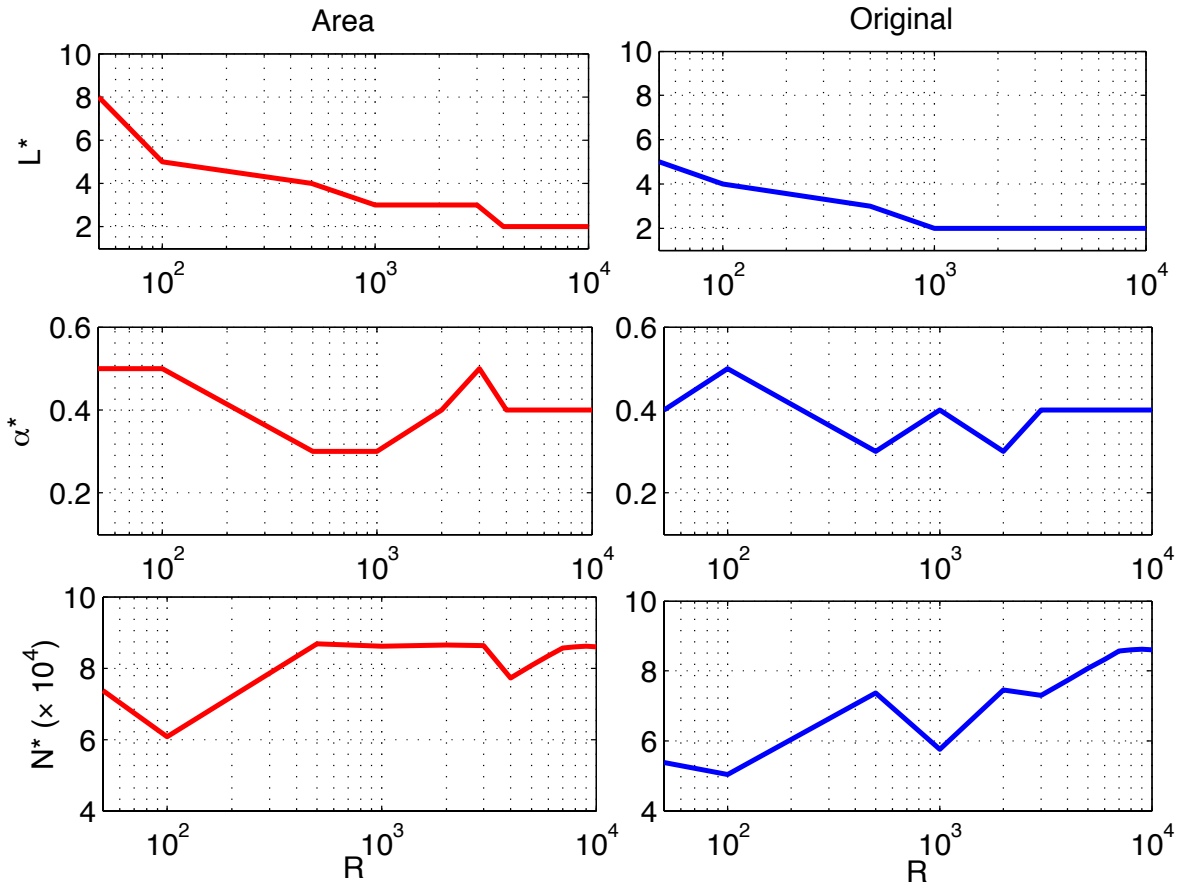


Figure 5.3: **Optimal values of the parameters :** optimal values of parameters L, α and N with respect to the number ofresampling runs.

L is larger, which suggests that choosing a small value for L leads to predicting more variable numbers of interactions per TG whereas a large value will force all TGs to be linked to a similar and higher number of TFs. This observation sheds some light on the choice of L in general, when assumptions can be made on the topology of the network to predict.

5.4.3 Comparison with other methods

Figure 5.7 depicts both the ROC and the Precision/Recall curves for several methods on Network 1. Table 2 summarizes these performances in terms of $AUPR$, $AUROC$ and related p-values as well as the overall score. Here, TIGRESS was run with $\alpha = 0.4$, $L = 2$ and $R = 8,000$ which corresponds to the best performance of the algorithm, as investigated in the previous section.

TIGRESS, as tuned optimally on this network, outperforms all methods in terms of AUPR and all methods but GENIE3 in terms of AUROC. Moreover, the shape of the Precision/Recall curve suggests that the top of the prediction list provided by TIGRESS contains more true edges than other methods. The ROC curve, on the other hand, focuses on the entire list of results. Therefore, we would argue that TIGRESS can be more reliable than GENIE in its first

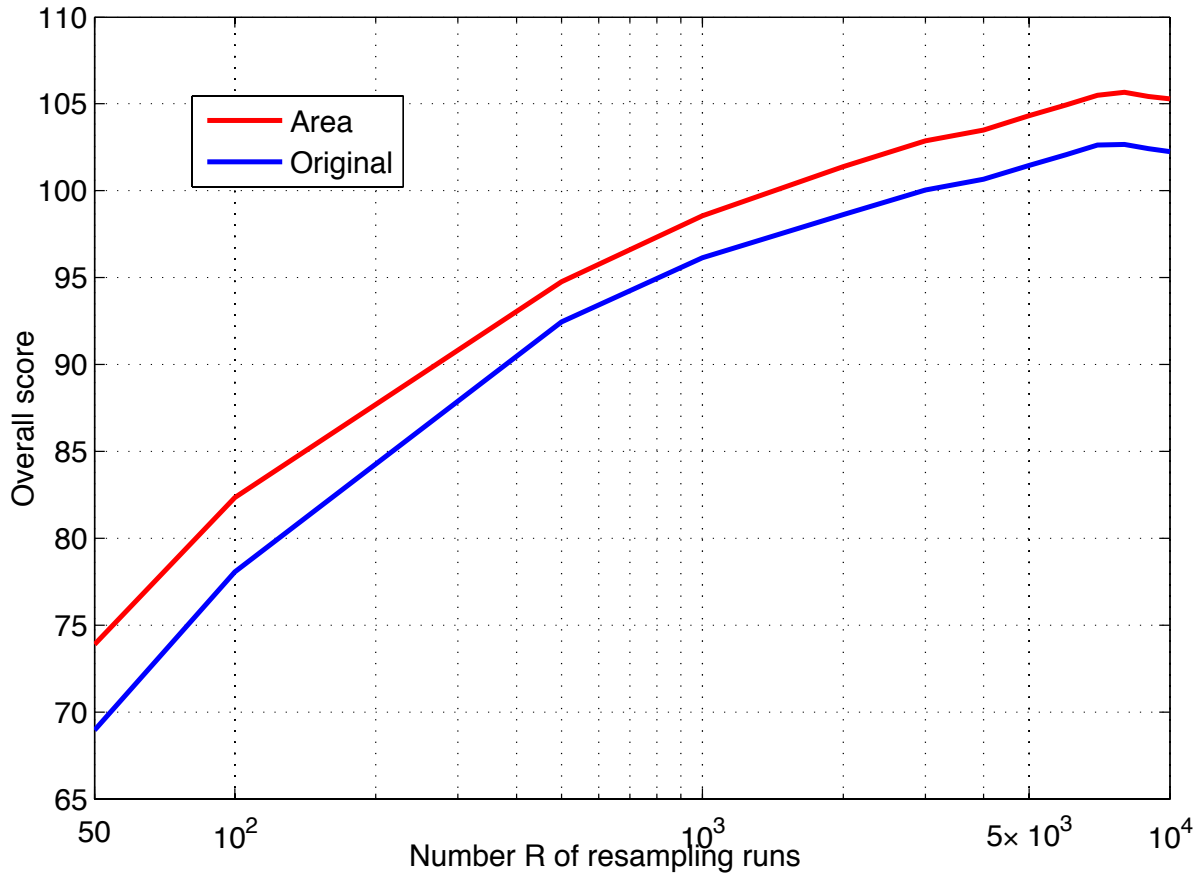


Figure 5.4: **Impact of the number of resampling runs :** overall score as a function of R. In both scoring settings, α and Lwere set to 0.4 and 2, respectively.

predictions but contains overall more errors when we go further in the list.

These results suggest that TIGRESS has the potential to compare with state-of-the-art methods and confirm the importance of correct parameter tuning.

5.4.4 *In vivo* networks results

Since Naive TIGRESS did not perform very well on the *in vivo* networks at the DREAM5 competition (Table 2), we now test on these networks TIGRESS with the best parameters selected on the *in silico* (area score, $\alpha = 0.4$, $L = 2$ and $R = 10,000$). Table 2 also shows the values of AUPR, AUROC, related p-values and overall score for DREAM5 networks 3 and 4 reached by TIGRESS and ROC and P/R curves were drawn on Figures 5.8 and 5.9.

The results on these two networks are overall disappointing: TIGRESS does not do better than Naive TIGRESS. In fact, both sets of results are very weak. Without attempting to re-optimize all parameters for each network, one may wonder whether the parameters chosen using the *in silico* network are optimal for the *in vivo* networks. As a partial answer, Figure 5.10 shows the behavior of the overall score with respect to L for Networks 3 and 4.

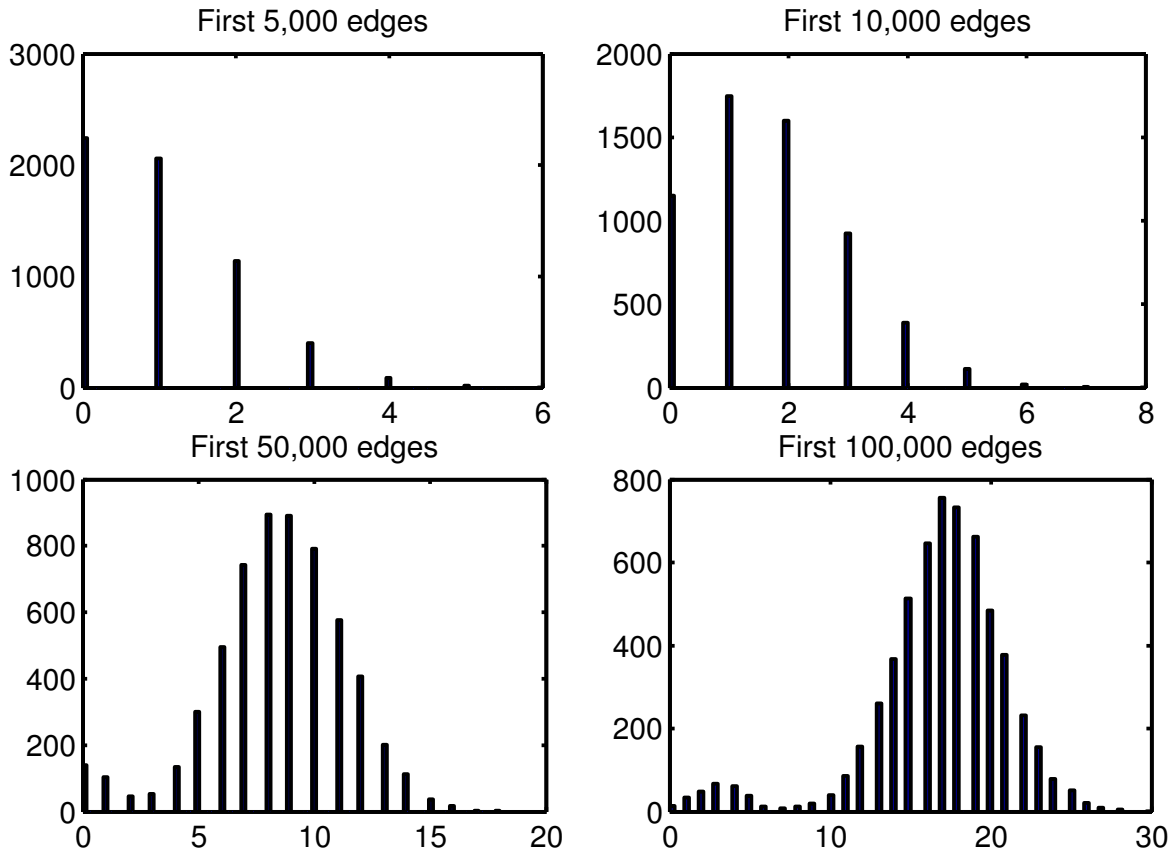


Figure 5.5: **Distribution of the number of TFs selected per gene for L=2 :** histograms of the number of TFs selected per gene with respect to the total number of predictions when $L = 2$.

Interestingly, it seems that a much larger L is preferable in this case, suggesting that one may have to adapt the parameters to the size of the network in terms of number of transcription factors. Indeed, networks 3 and 4 contain respectively 334 and 333 transcription factors, making them much larger than the *in silico* and the *E. coli* networks (195 and 180 TFs respectively), for which a small L leads to a better performance. Choosing $L = 100$ for DREAM5 *in vivo* networks yields much better results. As a matter of fact, TIGRESS obtains the best results on Network 4 with this value of L and doubles its performance on Network 3.

On Figure 5.11 we compare Precision/Recall and ROC curves obtained with TIGRESS with several other algorithms on the *E. coli* network from Faith et al. (2007).

Table 5.3 compares the areas under the curves. TIGRESS is comparable to CLR, while GENIE3 outperforms other methods. However the overall performance of all methods remains disappointing.

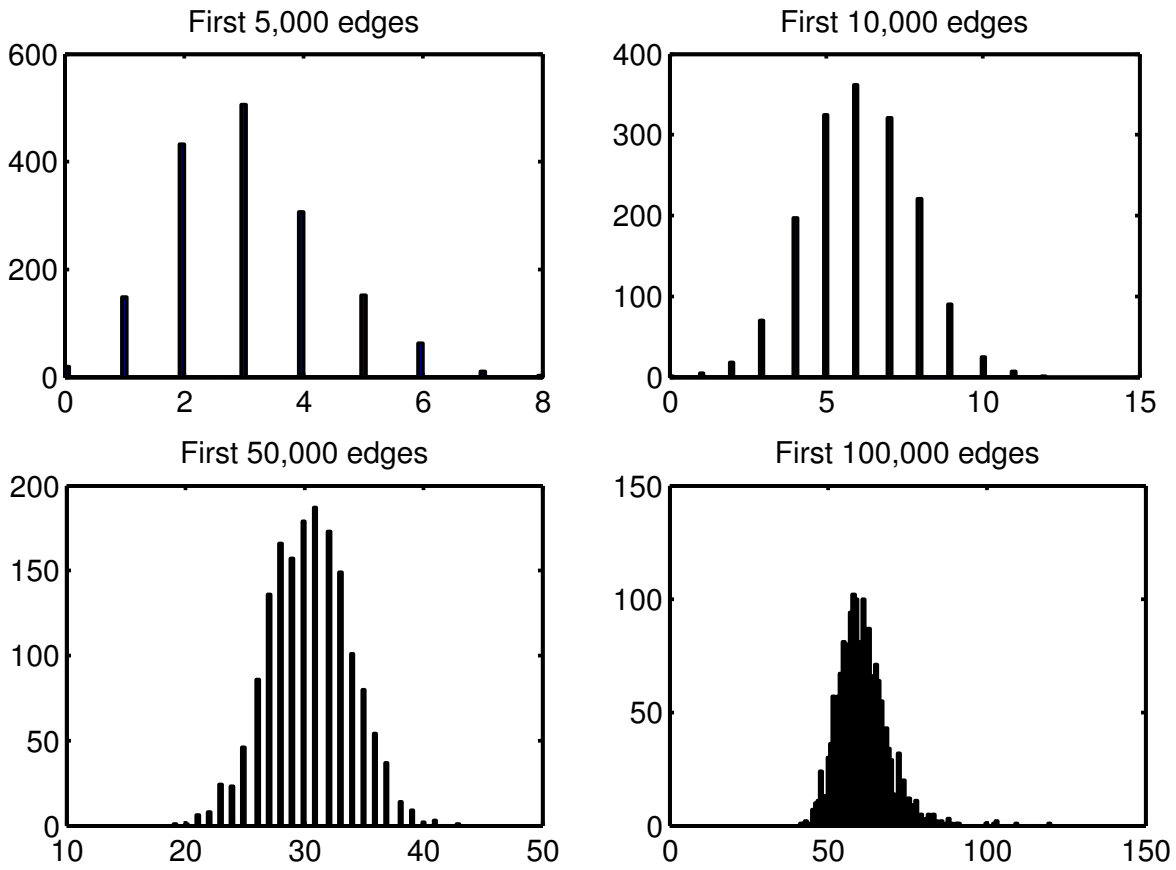


Figure 5.6: **Distribution of the number of TFs selected per gene for L=20 :** histograms of the number of TFs selected per gene with respect to the total number of predictions when $L = 20$.

Method	**AUPR**	**AUROC**	**Score**
TIGRESS	0.0624	0.6026	0.3325
ARACNE	0.0498	0.5531	0.3014
CLR	0.0641	0.6019	0.3330
GENIE3	0.0814	0.6375	0.3594

Table 5.3: ***E. coli* network results:** TIGRESS compared to state-of-the-art methods on the *E. coli* network. Since no p-value can be computed here, the score is simply the average between AUROC and AUPR.

5.4.5 Analysis of errors on *E. coli*

To understand further the advantages and limitations of TIGRESS, we analyze the type of errors it typically makes taking the *E. coli* dataset as example. We analyze FP, *i.e.* cases where TIGRESS predicts an interaction that does not appear in the gold standard GRN.

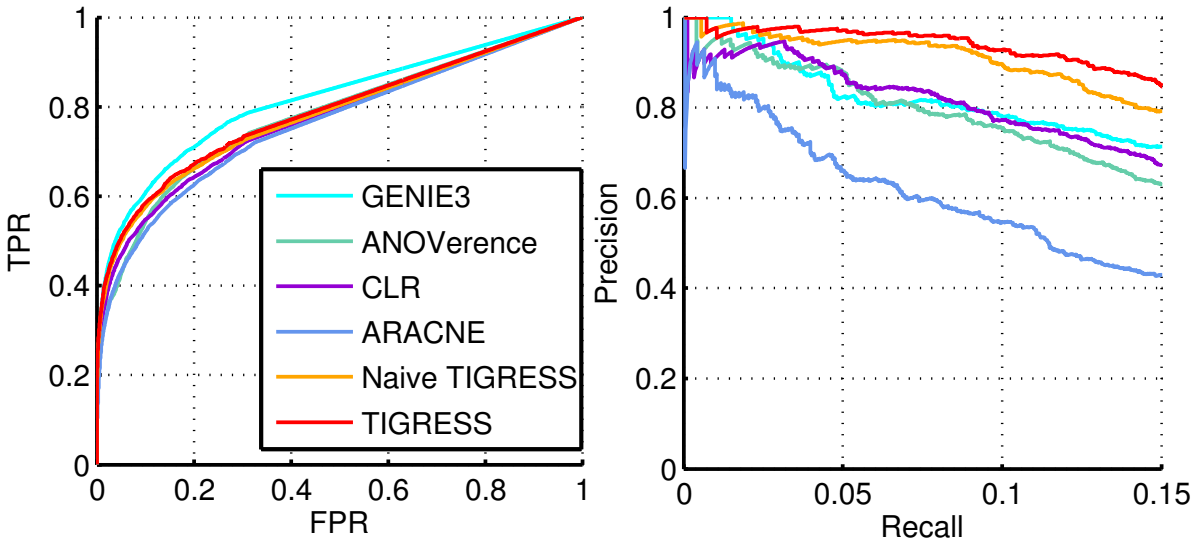


Figure 5.7: **Performance on Network 1:** ROC (left) and Precision/Recall (right) curves for several methods on Network 1.

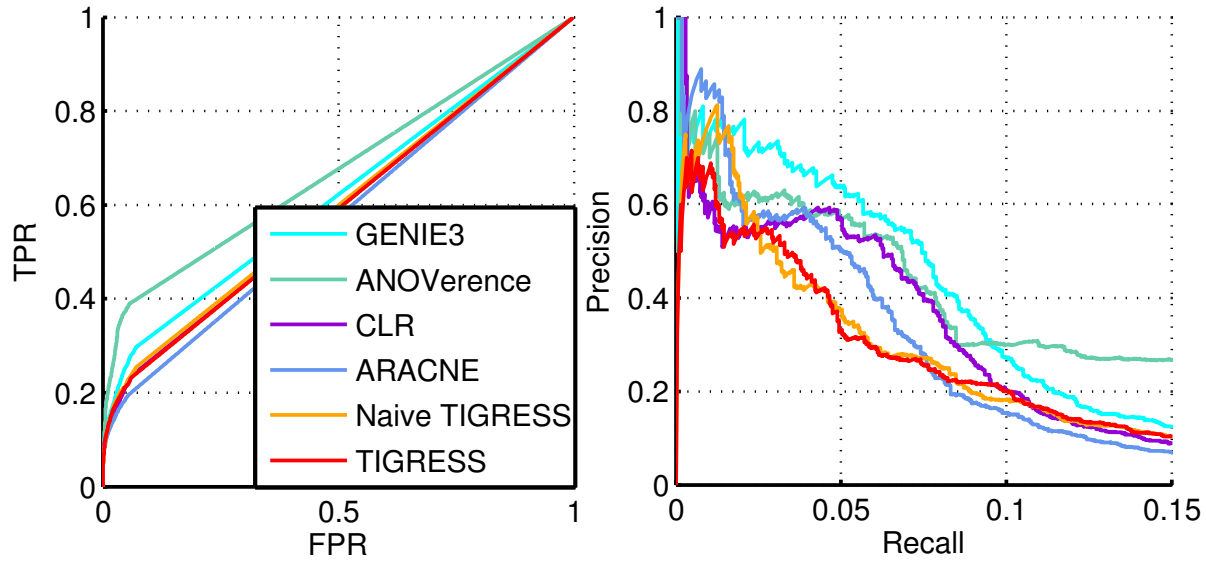


Figure 5.8: **Performance on DREAM5 network 3:** ROC (Left) and Precision/Recall (Right) curves for several methods on DREAM5 network 3.

We focus in particular on quantifying how far a wrongly predicted interaction is from a true one, and introduce for that purpose the notion of distance between two genes as the shortest path distance between them on the gold standard GRN, forgetting about the direction of edges. For two genes $G1$ and $G2$, we call $G1$-$G2$ a *distance-x* link if the shortest path between $G1$ and $G2$ on the true network has length x. Figure 5.12 shows the distribution of these distances for spuriously discovered edges over the gold standard network, *i.e.* the actual proportion of distance-x links, with $x \in \{1, 2, 3, 4, > 4\}$. We write $\hat{p}_x$ the proportion of spurious TF-TG

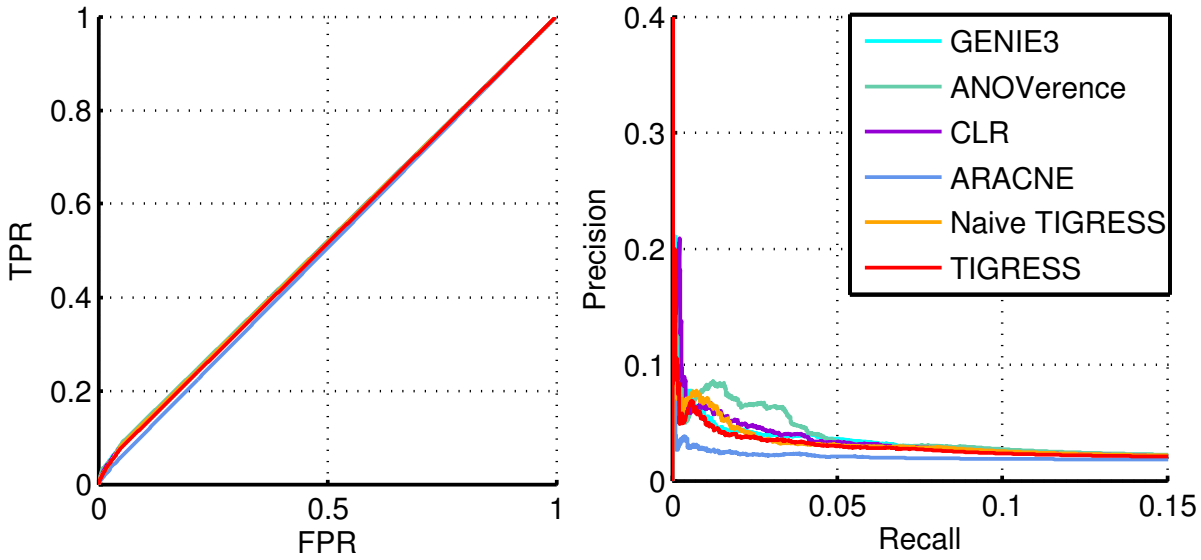


Figure 5.9: **Performance on DREAM5 network 4:** ROC (Left) and Precision/Recall (Right) curves for several methods on DREAM5 network 4.

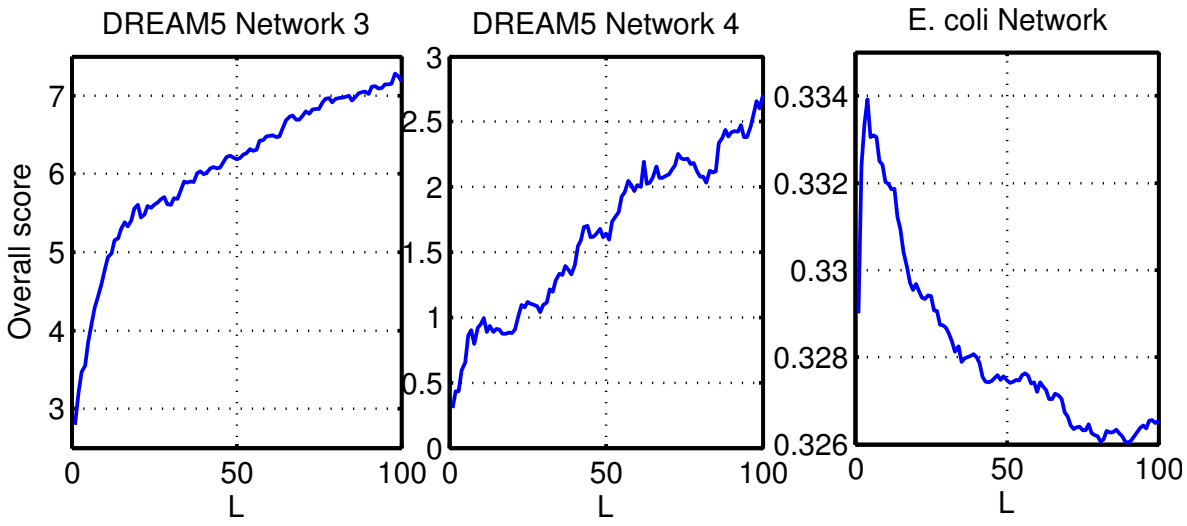


Figure 5.10: ***In vivo* networks results:** overall score with respect to L for DREAM5 networks 3 and 4 and *E. coli* network ($\alpha = 0.4$, $R = 10,000$).

interactions with distance x.

Figure 5.13 depicts the distribution of distance-x proportions among the spuriously detected edges, as a function of the number of predicted edges.

Dotted lines represent the 95% confidence interval around the exact distribution $(\hat{p}_x)_x$. For a given number r of spuriously predicted edges, this interval is computed as

$$\left[\frac{q_{0.025}(\hat{p}_x)}{r}; \frac{q_{0.975}(\hat{p}_x)}{r}\right],$$

where $q_a(\hat{p}_x)$ represents the quantile of order a of a hypergeometric distribution $\mathcal{H}(N_S, \hat{p}_x N_S, r)$

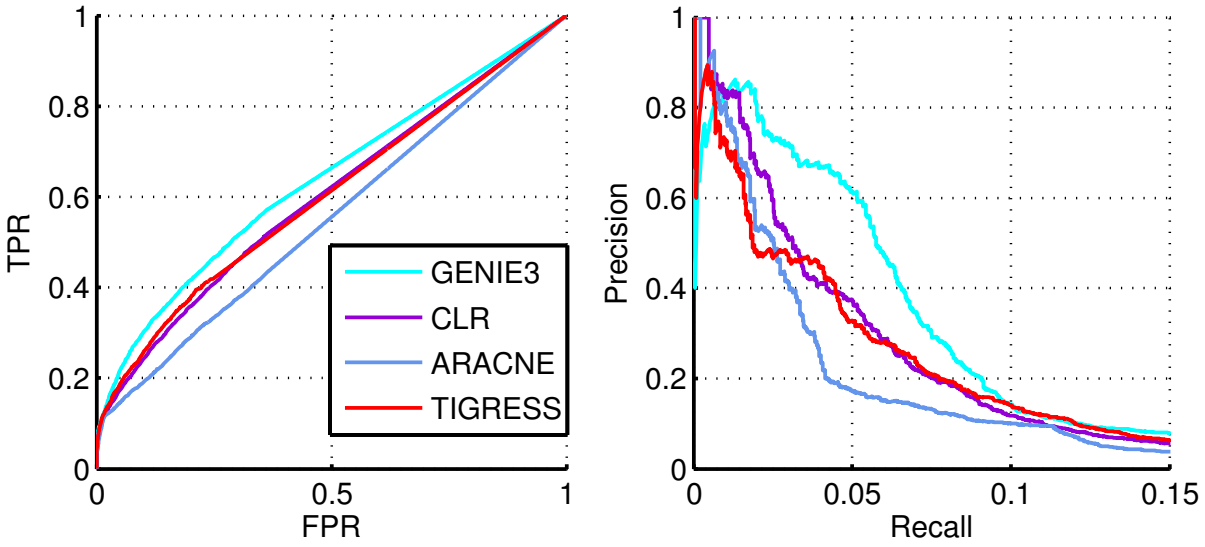


Figure 5.11: **Performance on the *E. coli* network:** ROC (Left) and Precision/Recall (Right) curves for several methods on the *E. coli* dataset.

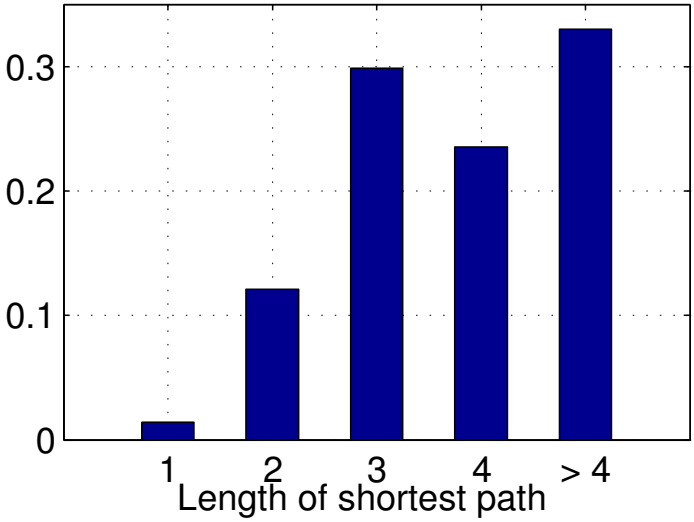


Figure 5.12: **Spurious edges shortest path distribution:** exact distribution of the shortest path between spuriously predicted TF-TG couples.

and N_S is the total number of spuriously predicted edges.

We observe that most of the recovered false positives appear as distance-2 edges in a significantly higher proportion than $\hat{p}_2$ whereas significantly less distance > 4 edges are discovered. These results strongly suggest that most of TIGRESS errors - especially at the top of the list - are indeed sensible guesses, where the two nodes, spuriously discovered with a parent/child relationship are actually separated by only one other node. In Figure 5.14, we detail the three possible patterns observable in this situation.

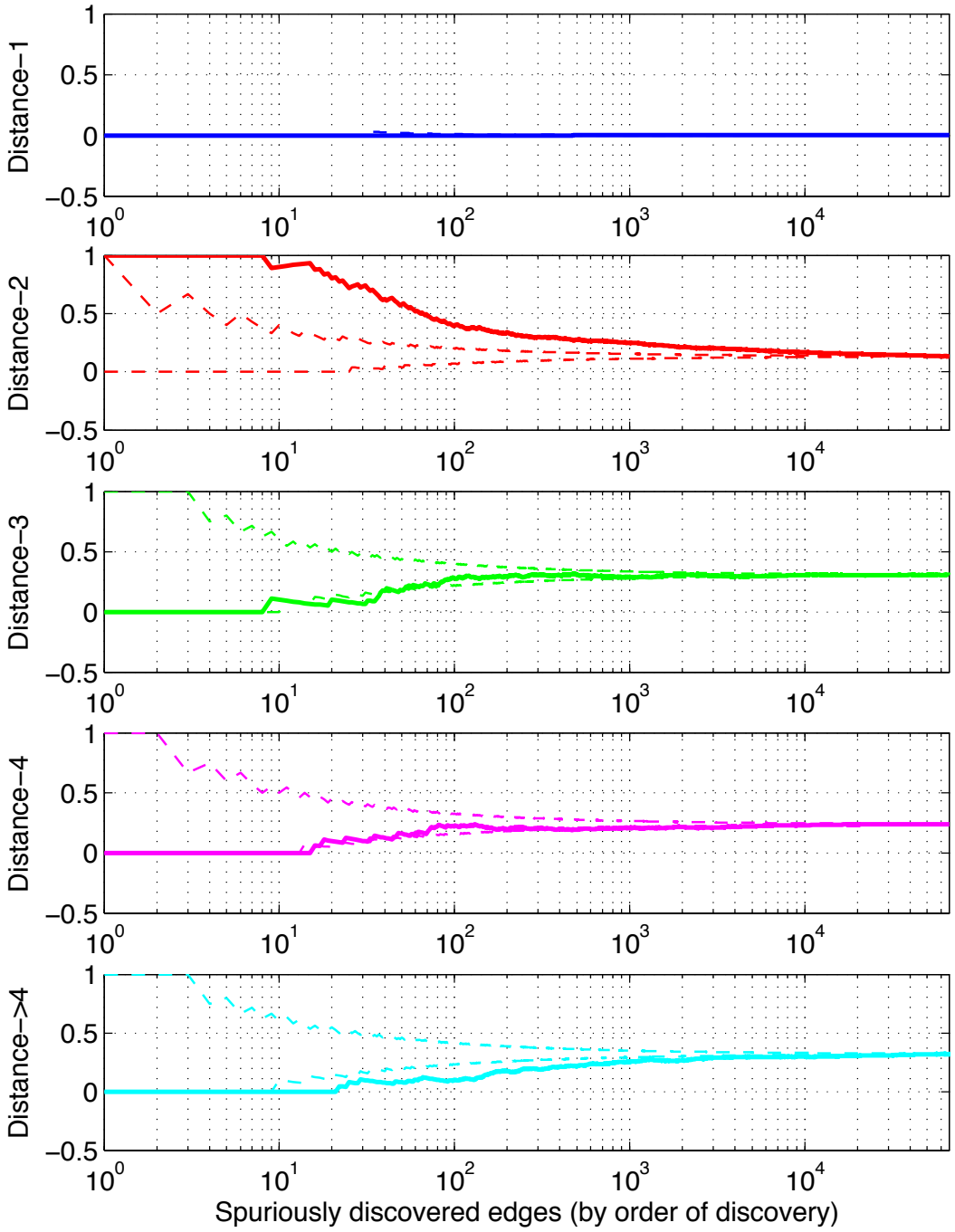


Figure 5.13: **Distribution of the shortest path with respect to the number of predictions:** distribution of the shortest path length between nodes of spuriously detected edges and 95% confidence interval for the null distribution. These edges are ranked by order of discovery.

Figure 5.15 focuses on distance-2 errors. Note that some edges show more than one pattern, e.g., the first spurious edges are both *siblings* and *couples*.

It appears that most of them are *siblings* and can thus be interpreted as spurious feed-

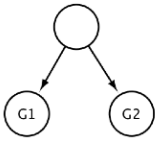
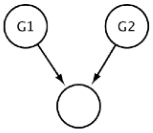
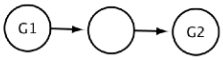
Name	Illustration	Description
Siblings		G1 and G2 have a common parent. They are *siblings*.
Couple		G1 and G2 have a common child. They are a *couple*.
Grandparent/Grandchild		G1 has a child that is a parent of G2. G1 is a *grandparent* of G2.

Figure 5.14: **Distance-2 patterns:** the three possible distance-2 patterns: siblings, couple and grandparent/grandchild relationships.

forward loops. We believe that this can be explained by three main reasons: i) the discovered edges actually exist but have not been experimentally validated yet; ii) there is more of a linear relationship between siblings than between parent and child; iii) some nodes have very correlated expression levels, making it difficult for TIGRESS to tell between the parent and the child.

5.4.6 Directionality prediction : case study on DREAM4 networks

In order to check whether TIGRESS can predict edge directions, we additionnally ran it on the five size 100 multifactorial DREAM4 networks, for which the TFs are not known. The five datasets contain 100 samples and 100 genes. We observe that TIGRESS can indeed perform well in this setting. Table 5.4 shows the results with the default parameter setting ($L = 2$, $\alpha = 0.4$, $R = 10,000$) compared with those of GENIE3, that won the DREAM4 network inference challenge. Without further optimization of the parameters on these networks, TIGRESS achieves a better overall performance than GENIE3.

Method	**Network 1**		**Network 2**		**Network 3**		**Network 4**		**Network 5**		**Overall score**
	AUPR	*AUROC*	*AUPR*	*AUROC*	*AUPR*	*AUROC*	*AUPR*	*AUROC*	*AUPR*	*AUROC*	
GENIE3	0.154	0.745	0.155	0.733	0.231	0.775	0.208	0.791	0.197	0.798	37.482
TIGRESS	0.165	0.769	0.161	0.717	0.233	0.781	0.228	0.791	0.234	0.764	38.848

Table 5.4: **DREAM4 networks results:** results on the 5 DREAM4 size 100 multifactorial networks. The results are shown for the directed setting

Furthermore, we ran the complete analysis of the parameters on these networks, to check whether the optimal parameters change when TFs are unknown. Figure 5.16 shows the overall performance, that is the average performance on all five networks, as a function of α and L.

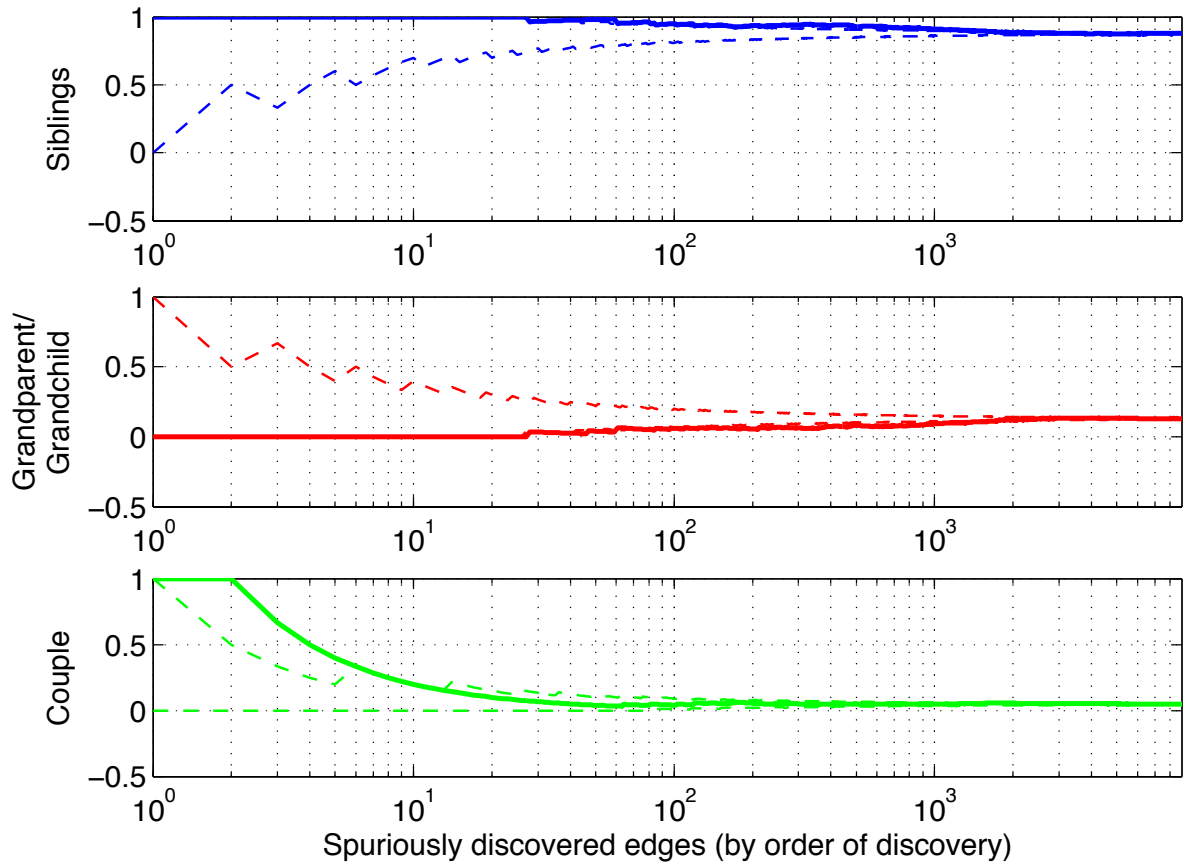


Figure 5.15: **Distribution of distance-2 errors:** distribution of distance 2 errors with respect to the number of predictions. 95% error bars were computed using the quantiles of a hypergeometric distribution.

Given the small size of the networks, it is not surprising that the optimal L is equal to 1. It also seems that the optimal value for α lies in between 0 and 0.1, corresponding to a strong randomization.

5.4.7 Computational Complexity

The complexity of running L LARS steps on a regression problem with q covariates and n samples is $O(nqL + L^3)$ Efron et al. (2004). In our case, q is the number of TF and n is the number of expression arrays, and we pay this complexity for each TG and each resampling. Multiplying by p TG and R resampling, we therefore get a total complexity of order $O(pR(nqL + L^3))$, which boils down to $O(pRnqL)$ in the situation where L is smaller than n and q.

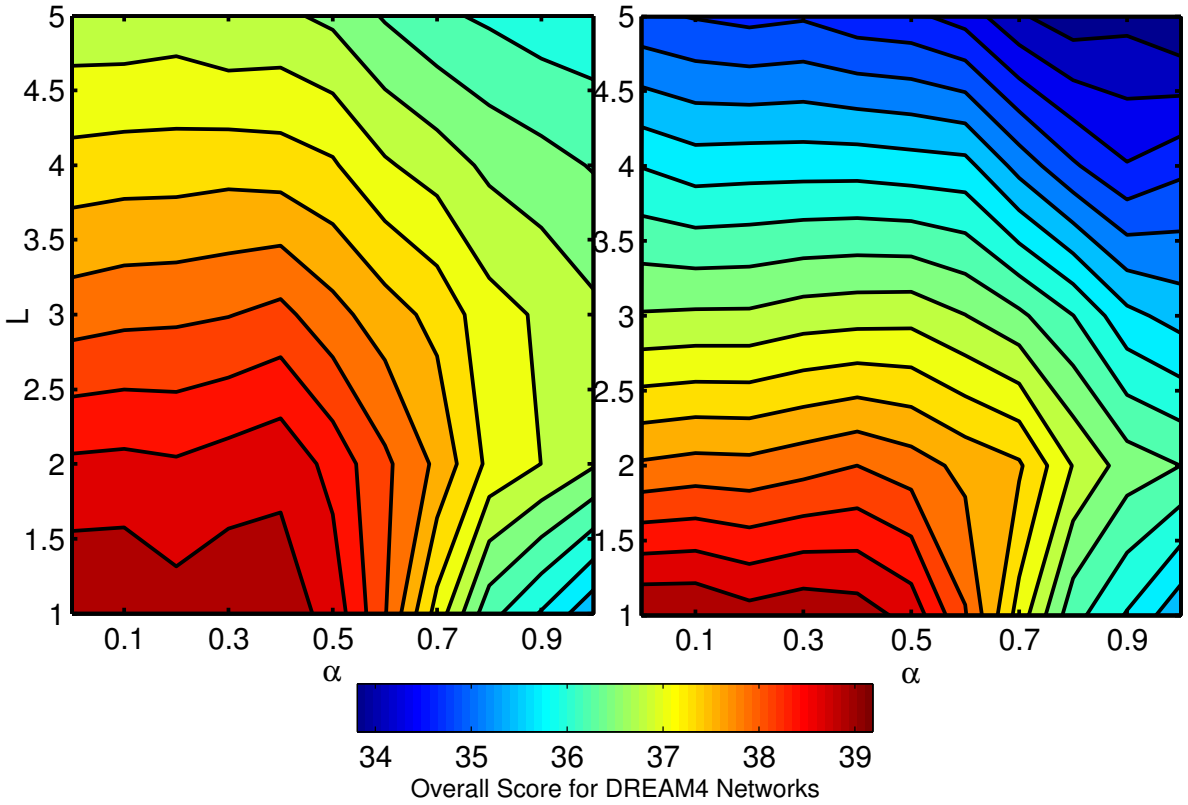


Figure 5.16: **Results on DREAM4 networks:** overall score on the five multifactorial size 100 DREAM4 networks, as a function of α and L.

5.5 Discussion and conclusions

In this paper, we presented TIGRESS, a new method for GRN inference. TIGRESS expresses the GRN inference problem as a feature selection problem, and solves it with the popular LARS feature selection method combined with stability selection. It ranked in the top 3 GRN inference methods at the 2010 DREAM5 challenge, without any parameter tuning. We clarified in this paper the influence of each parameter, and showed that further improvement may result from finer parameter tuning.

We proposed in particular a new scoring method for stability selection, based on the area under the stability curve. It differs from the original formulation of Meinshausen and Bühlmann (2010) which does not take into account the full distribution of ranks of a feature in the randomized feature selection procedure. Comparing the two, we observed that the new area scoring technique yields better results and is less sensitive to the values of the parameters: practically all values of, e.g., the randomization parameter α yield the same performance. Similarly, the choice of the number L of LARS steps to run seems to have much less impact on the performance in this new setting. As we showed, the original and area scores for a feature t can be both expressed in a common formalism as $E\left[\phi(H)\right]$ for different functions ϕ, where H_t is the rank of feature t as

selected by the randomized LARS. It could be interesting to systematically investigate variants of these scores with more general non-increasing functions ϕ, not only for GRN inference but also more generally as a generic feature selection procedure.

Comparing TIGRESS - as tuned optimally - to state-of-the-art algorithms on the *in silico* network, we observed that it achieves a similar performance to that of GENIE3 (Huynh-Thu et al., 2010), the best performer at the DREAM5 challenge. However, TIGRESS does not do as good as this algorithm on *in vivo* networks. GENIE3 is also an ensemble algorithm but differs from TIGRESS in that it uses a non-linear tree-based method for feature selection, while TIGRESS uses LARS. The difference in performance could be explained by the fact that the linear relationship between TGs and TFs assumed by TIGRESS is far-fetched given the obvious complexity of the problem.

A further analysis of our results on the *E. coli* network from Faith et al. (2007) showed that many spuriously detected edges follow the same pattern: TIGRESS discovers edges when in reality the two nodes are *siblings*, and thus tends to wrongly predict feed-forward loops. This result suggests many directions for future work. Among them, we believe that operons, *i.e.* groups of TGs regulated together could be part of the problem. Moreover, it could be that there is more of a linear relationship between siblings than between parent and child, as TFs are known to be operating as *switches*, *i.e.* it is only after a certain amount change in expression of the TF that related TGs are affected. However, it is worth noting that *in vivo* networks gold standards may not be complete. Therefore, the hypothesis that TIGRESS is actually correct when predicting these loops cannot be discarded.

TIGRESS depends on four parameters: the scoring method, the number R of resampling runs, the randomization factor α and the number of LARS steps L. We showed in this paper that changing the value of these parameters can greatly affect the performance and provided guidelines to choose them. It is worth noting, though, that other modifications can be imagined. In particular, one may wonder about the influence of the resampling parameters (with or without replacement, proportion of samples to be resampled). These questions will be tackled in future work.

While it seems indeed more realistic not to restrict underlying models to linear ones, it is fair to say that no method performs very well in absolute values on *in vivo* networks. For example, performances on the *E. coli* network seem to level out at some 64% AUROC and 8% AUPR which cannot be considered satisfying. This suggests that while regression-based procedures such as TIGRESS or GENIE3 are state-of-the-art for GRN inference, their performances seem to hit a limit which probably cannot be outdistanced without some changes. It is worth noting that, as argued in Marbach et al. (2012), combining these methods together leads to improvement, as different sets of interactions are discovered by each method. Another way to overcome these limits may be a change in the global approach such as adding some supervision in the learning process as, e.g., investigated in Mordelet and Vert (2008).

Chapter 6

Discussion

In this chapter, I try to take a step back and examine what I have learned during the past three years and to what conclusions I have personally come regarding the problems I tackled.

In Section 6.1, I discuss how the role of the statisticians has changed - in my view - since the rise of *high-dimensional data* and, in particular, *microarray data*. In Section 6.2, I try to make sense of ten years of research, sometimes contradictory, on signature selection for breast cancer outcome prediction. Finally, in Section 6.3, I explain why I believe that we may have to move on to *black boxes* to solve problems such as those investigated in this thesis.

6.1 Microarray data: the official marriage between biology and statistics

Something old

Quantification and modeling of biological phenomena is nothing recent, really. In 1869, Darwin's cousin, Francis Galton, was the first researcher to attempt a statistical approach to investigating heredity (Galton, 1869). The ethically disturbing use he made of statistics as a means to develop the theory of eugenics probably explains why we tend to forget some of his contributions (among which the definition of correlation and regression!). Together with Karl Pearson (yet another eugenicist), he founded *Biometrika* in 1901 with a clear editorial line:

> *It is intended that* Biometrika *shall serve as a means not only of collecting or publishing under one title biological data of a kind not systematically collected or published elsewhere in any other periodical, but also of spreading a knowledge of such statistical theory as may be requisite for their scientific treatment.*

No later than 1950, Ronald Fisher (and yet another eugenicist) provided the world with what can probably be qualified today as the first bioinformatics study, using a computer to calculate gene frequencies and standard deviations (Fisher, 1950).

Something new

Some fifty years after this first study, microarray technologies rose to be the new paradigm of genetic research. Before we were able to compute expression profiles for the entire genome, it is safe to say that gene expression-based research was performed in a *hypothesis-centered* way: a link between the expression of a gene and a phenotype would have to be hypothesized before being tested. It is quite the opposite that we are witnessing now: the first step is to generate the data, the second is to mine it. There is no need to formulate hypotheses beforehand and nor is it required to know exactly what question(s) we are going to ask the data.

Unquestionably, being able to access this quantity of data is a blessing and molecular signatures have proven efficient in a variety of cases including diagnosis of thyroid nodules (Alexander et al., 2012), early diagnosis of Alzheimer's disease (Ravetti and Moscato, 2008) or cancer (Ramaswamy et al., 2003). It is a blessing to patients, first, who can begin to hope for a personalized treatment through a simple screening. To practicians, as well, to gain insight on a disease that is found to be more and more complicated every day. To biologists, whom this technology allows to reduce their study of a particular disease to individual biomarkers as well as complement their instinct with quantitative evidence. And finally, to statisticians, who have earned a central role in the fight against one very hateful and vicious enemy.

Something borrowed

With this new type of data at hand, statisticians have had to change their ways: classical statistical analysis does not apply to these huge matrices that exhibit more variables than individuals and where noise does not anymore pose as this little adjustment variable but instead possibly clouds most of the observed signal.

But one does have to start somewhere and that is using the knowledge at hand. The generation of statisticians that are currently tackling microarray-based problems were not taught how to handle gene expression data in statistics class and might instead have spent quite a few hours solving maximum likelihood estimation equations by hand or desperately trying to understand the point of statistical testing (well, at least some of us were struggling)... which explains, in my opinion, why many of the current gene expression analysis methods were actually borrowed from the classics.

Many questions have been coming and going through my mind during this thesis: as a statistician who did not know the smallest thing about biology or microarrays three years ago, was I - well, am I - qualified to tackle these problems using a knowledge that I cannot justify as being a right fit? Does it make sense to use a Student's t-test to select relevant genes? Am I out of my mind when applying linear regression models with gaussian noise to data that I know does not even begin to satisfy basic theoretical requirements of such modeling? On the one hand, I do quite agree with the following claim (Freedman, 2006):

> *With models, it is easy to lose track of three essential points: (i) results depend on assumptions, (ii) changing the assumptions in apparently innocuous ways can lead to drastic changes in conclusions, and (iii) familiarity with a model's name is no guarantee of the model's truth. Under the circumstances, it may be the assumptions behind the model that provide the leverage, not the data fed into the model.*

On the other hand, it does not look like there are many satisfactory alternatives. There is only so much we can do, which is our best, and that is a start.

Indeed, we - the community - seem to learn how to adapt to this new type of data in a trial/error fashion. Based on the number of criticizing opinion papers and the increasing amount of reproduced research, it would appear to me that the biostatistical community has grown more and more cautious about taking for granted the results published by their peers. I mean this in a good way here as I believe that this cautiousness is one key to progressively constructing reliable methods.

A further thought brings us back to Section 1.1.3 and the opposite route some groups have taken their research to, deciding against classical statistical modeling and choosing to use black boxes, that require no assumption of any sort. We will come back to this in Section 6.3.

Something blue

As stated earlier, I am not questioning the fact that high-throughput technologies constitute an immense progress to cancer research. However, I must confess that, at some points on my way to writing these words, I have known a few frustrations and felt many doubts.

Truth be told, looking into microarrays to find meaningful information is a difficult task. The contrary would actually be surprising: it is a needle in a haystack problem, where, on top, there is no convincing evidence that the needle actually exists - at least I am not completely convinced.

But I tried. And my background would dictate me to "think statistically", i.e. to use what had been learned or what was expected to guide the model. I was thus thrilled to meet Laurent Jacob and work with him on the application of graph Lasso (Jacob et al., 2009) to breast cancer prognosis signatures (see Sect. 3). The idea immediately made a lot of sense to me: using prior information on gene interactions to improve the signature by forcing it to *mean something*. Of course it would work, it had to, I thought, because it *made sense*. So, we recovered more interpretable signatures to some extent, but there was not the slightest increase in accuracy.

It was also during this first project that I had the most awakening experience about this new science I was discovering: I was evaluating the *interpretability* of this new method we developed by mapping selected genes to the Gene Ontology database. The results were very nice, I thought, as many of the genes in my signature were making sense, biologically speaking, which was validated by very low p-values. The excitement lasted for about five minutes, until I

discovered that I had run the wrong line of my code and that the list of genes I was finding so meaningful had been drawn completely at random. And this is what it took for me to realize that I would have to be extremely careful in the way I would evaluate and interpret my results in the future. It took a few more of these experiences until I would triple-check each and every result and after three years of this somewhat paranoid behavior, I must say that I have learned to consider my results guilty of some bias until proven innocent. Consequently, I do not get carried away by excitement when allegedly good results show up on my screen.

To conclude on this light blue note, when I call this problem *difficult*, I do not only mean it from a statistical point of view: it is difficult to have to discard a method when there is no understanding as to why it does not work as well as expected, let alone the hope at bringing some insight to the hideous ways of cancer that dies with it. And in the case of gene expression profiling, this does unfortunately happen quite a lot.

6.2 Signatures for breast cancer prognosis: ten years of contradictions?

This story begins, ten years ago, with the publication of the pioneer study (van de Vijver et al., 2002) in the *New England Journal of Medicine*, leading to the FDA-approved and now commercialized 70-gene signature *MammaPrint*. What have we learned since?

First of all, it would appear that the idea of predicting breast cancer outcome from expression data has appealed to the community - with a number of good reasons -, as a plethora of research groups have been publishing on the subject for the past decade. Many new signatures were published and five of them are currently available on the US market (Hornberger et al., 2012). In Venet et al. (2011), the authors actually identify no less than 47 signatures in the literature. Evidence is provided in these papers that the published signatures exhibit genes with predictive power regarding breast cancer outcome. But the community would soon discover a disturbing fact: as predictive as each signature could have been proven to be, only little overlap, if any, was detected between two different proposals (see, e.g., Fan et al., 2006, Thomassen et al., 2007).

This is, I believe, when things got a little more complicated: if signatures would not overlap, there had to be an explanation to it. From 2005, some groups decided to find out by directing *meta-studies*. Were the published signatures related on some level? Did the signatures predict the same outcomes? Were their accuracies transferable to other datasets? Were they even predictive, at least more than chance? At this point, I think that it became about agreeing to disagree: molecular signatures have been about as praised as criticized and even meta-studies would start to contradict each other. Digging literature would soon appear to be a tough detective work. As quite a lover of detective stories indeed, I was puzzled: when two different studies simultaneously prove one point and its contrary, does one of them not have to be right? This is really disturbing to a math-head like me. It turns out that, in bio-medical research, the

answer to a yes-or-no question is not necessarily yes or no. In the following, I attempt to break down these apparent contradictions through a set of questions and provide my own opinion on the matter.

Can I get an adequately performing molecular signature to predict breast cancer outcome?

The answer is **yes** -ish (I will detail this below). Yet, I do not recommend asking PubMed this very question with less than a few weeks to spare reading hundreds of papers that seem contradictory.

First of all, performance is actually a very relative measure and it would not mean a thing if I wrote something like : "a 70% AUC is guaranteed with a good method". So let me clarify: for sure, it is possible to obtain a signature that performs better than chance and some of the gene sets that have been published do satisfy this criterion. There is no question about that. However, some papers would make one a little sceptic about that fact. Let us break down some critical research on this issue.

In Ein-Dor et al. (2005), the authors question the unicity of the signature by showing that it is possible to find random sets of genes that have a similar or higher predictive accuracy than MammaPrint on Van't Veer's dataset. Michiels et al. (2005) report similar findings, and add that among the 500 signatures they estimated, only 14 of the 70 genes from MammaPrint were included in more than half of them. The authors write :

> *Five of the seven largest published studies addressing cancer prognosis did not classify patients better than chance. This result suggests that these publications were overoptimistic.*

They raise concerns about learning a signature from a dataset with few samples. In Haibe-Kains et al. (2008b), the authors suggest that a single proliferation gene - AURKA - performs as well as a larger list of genes. More recently, after a four-year fight with bio-medical editors, Venet et al. (2011) were finally able to publish that of the 47 signatures they identified and tested against random sets of genes, 28 did not perform significantly better than the average random signature. They add that with a chance from 5% to 17%, one can find a single gene associated with the outcome and thus somewhat predictive. Finally, we (Haury et al. (2011) and Chapter 2) show that on several datasets, only a few feature selection methods yield significantly better performances than the average random list of genes.

What should we make of these studies? My first opinion is that all of them were well conducted and report accurate results. I do slightly disagree with the quote I reported from the conclusion of Michiels et al. (2005) though: in my opinion, there is no causal implication between the statements "five of the seven largest published studies addressing cancer prognosis did not classify patients better than chance" and "the publications were overoptimistic". Indeed,

their study **did not** invalidate the signature from van 't Veer et al. (2002) in the sense that the accuracy of MammaPrint was not questioned, i.e., it was not shown that MammaPrint's actual accuracy had been overestimated in the original study. The same goes for the other studies. I believe the correct conclusion to be that the baseline is high, i.e. - and we will see below why - random signatures are not that bad, but again, it does not imply that signatures performing similarly or only slightly better than chance are not predictive.

I would like to expand for a moment on the concept of a signature performing *better than chance*. Chance relates the the absence of a pattern. If I were to predict the outcome of breast cancer by tossing a coin for each patient, I could expect to predict about 50% of good outcome. For any given test set, I would thus achieve a predictive accuracy of some 50%. On the other hand, let us now assume that our dataset were composed only of genes that would be completely predictive, i.e. 100% correlated with the outcome, and that our classifier would choose any of these genes, at random. Then, the predictive accuracy would be 100%, regardless of the gene that was chosen. We can see the difference between these two cases, although we both call them *random*. This gets me to my point: it would probably be more accurate to refer to the performance of random lists of genes as the *baseline*, as opposed to the *chance* or *randomness*, which are confusing terms. It would correspond to the information that is carried by expression data, regardless of the selection procedure we apply.

With that in mind, we can go back to the evidence provided in the papers cited earlier. What they really prove, in my opinion, is that some ways of performing variable selection make it possible to capture *more* information from the data than drawing random genes, which do contain information anyway. For example, we report in Haury et al. (2011) that the average AUC of "random" on dataset GSE2034 (Wang et al., 2005) was to be expected around 62%, which is about 12% better than tossing a coin. Supporting this theory is further evidence from Venet et al. (2011) from the abstract of which I borrowed the following quote:

> *Surprisingly, we found that gene expression signatures - unrelated to cancer - of the effect of postprandial laughter, of mice social defeat and of skin fibroblast localization were all significantly associated with breast cancer outcome.*

The authors claim that most of the outcome-related information is also related to proliferation. Thus, any list of genes related to proliferation or more generally, cell cycle activity, would contain some predictive power with respect to breast cancer outcome. The larger the list, the larger the chance of capturing such an information. Currently, there are 1825 gene products associated with Gene Ontology biological process "Cell Cycle", which should be mapped to the array of interest and divided by the total number of genes considered to get an estimate of these odds. A number of studies confirm the implication of proliferation genes in the transcriptome of breast cancer, among which Whitfield et al. (2006), Thomassen et al. (2007), Reyal et al. (2008), Mosley and Keri (2008), Drier and Domany (2011).

To conclude on this part, our goal, of course, is to obtain the most accurate predictors. On some level, it is good news that random genes are somewhat good performers, since we are certain to perform better than chance. This also gives us a baseline. We have no clue, though, as to whether this baseline can be beaten by far. We have shown that doing better is possible. But to what extent? This we do not know.

What are my best options to build a decent predictor from scratch?

In order to build a predictive model, one needs:

- a method that can select a list of genes
- a method that can classify the samples in a binary way
- the will to comply to a few rules to avoid overfitting

I will be brief, as these issues have been discussed extensively in Section 2. In particular, I will not expand on the third point, for which guidelines are provided in the introduction as well.

Our findings from Section 2 (and Haury et al., 2011) are in agreement with many other publications. It would seem that overall, keeping it simple works well. The genes can be selected using a Student's t-test, for example. This method has been vastly employed, starting with van 't Veer et al. (2002) and has been consistently reported as one of the most performant methods (see, e.g., Lai et al., 2006, Zucknick et al., 2008, Popovici et al., 2010, Haury et al., 2011, Staiger et al., 2012). Regarding the classification algorithm, simple works also well, as two of the most popular methods are the *Nearest Centroids classifier* and the very similar *Diagonal Linear Discriminant Analysis*. Again, many publications report consistently good performances of this class of algorithms (see Dudoit et al., 2002, Michiels et al., 2005, Lai et al., 2006, Zucknick et al., 2008, Popovici et al., 2010, Haury et al., 2011, Staiger et al., 2012). It is likely that a similar performance will be achieved by a more complex classifier such as a SVM. However the former methods are in general faster. So why bother?

Yes, simple works, and I must say that it has been quite surprising to me. And somewhat disappointing as I started my PhD with the hope of providing the world with a very clever method requiring prior knowledge (we will discuss this in a later paragraph) that would just *have to do better* than somewhat *naive* procedures. However, from the quantity of evidence at disposal, no doubt is permitted. Let alone that William of Ockham would probably agree.

What type(s) of stability can I expect from a good signature?

If I have to be pessimistic at some point, it is on this part: it seems to me that there is little we can expect regarding the stability of breast cancer outcome signature. I will wrap three concepts in this term:

- gene stability: the overlap between two signatures in terms of genes
- prediction stability: the concordance of two signatures in terms of the predicted outcome
- functional stability: the overlap between two signatures in terms of biological functions

Gene stability has been exhaustively discussed in Sections 2 and 3. We showed that only little overlap is observed, both when the genes are selected on two separate datasets and when they are chosen from subsamples from one single dataset. This claim is supported by, e.g., Ein-Dor et al. (2005, 2006), Staiger et al. (2012). The same goes for commercialized signatures that have very few genes in common, if any (Thomassen et al., 2007). Because of the high correlation of many genes on the array, I believe today, after three years of giving it all of my best shots, that we will not achieve any decent gene stability, especially if we work on datasets that contain 100 times more genes than samples. But this begs a question: should we care? In a world where two signatures would successfully predict the exact same outcome for a given set of patients and, - even better - contain genes that we can consider as "working together", I believe that we should not. Recall that the primary goal is to predict well and the secondary goal is to interpret the signatures we obtain. In Simon (2008), the author points out - in quite a subtle way - that as we are looking for a signature, our knowledge on the disease is too limited for us to know what we should expect to find:

> *Accurate and robust predictive classifiers should not be rejected because we do not understand the underlying biology or because the particular genes used in the prediction may not be unique.*

Of course, all things being equal, we would prefer a method that would systematically output the same genes. However, things are being everything but equal, as we will see next.

Many groups have been interested in the prediction stability - a problem that we have not addressed. Haibe-Kains et al. (2008a), Fan et al. (2006) and Thomassen et al. (2007) compare respectively 3, 5 and 8 published signatures on test datasets of respective sizes 198, 295 and 60. All three studies conclude that some of them show significant agreement in the prediction. In Thomassen et al. (2007), the authors emphasize that "four of [the five] models [they tested] resulted in similar predictions - for example, each model assigned the same samples to the poor-outcome groups." Different results emerged in Reyal et al. (2008), where the authors perform a comparison of nine prognostic signatures on $1,127$ samples. They show that more than half of the samples have at least one discordant class assignment over the nine signatures, both in the good and poor outcome groups. The much larger set used by Reyal et al. (2008) makes the evidence from this paper more convincing, suggesting that, indeed, published signatures are not concordant in terms of prediction.

A number of studies have addressed the question of functional stability between signatures. As stated earlier, signatures have been shown to contain proliferation-related genes. However, as

demonstrated by Venet et al. (2011), we are bound to find these and therefore, we are interested in discovering other pathways that contain predictive power. In Shen et al. (2008), the authors claim functional convergence of four published signatures over several other pathways than proliferation. However, this claim is only supported by very poor enrichment scores. Reyal et al. (2008) build *enlarged signatures incorporating functional informations* revealing a high accuracy of classifiers trained using a combination of the Immune Response and the RNA splicing modules. Yu et al. (2007) had already reported enrichment of the Immune Response module for ER positive tumors.

The latter question brings us to discussing the incorporation of prior knowledge, such as pathways or graphical information, in the signatures.

Should we integrate prior knowledge on the genes?

On the subject of including prior knowledge to the prognosis signatures, studies could not disagree more.

Using a PPI graph as input, Chuang et al. (2007) average the expression of genes connected on a *subnetwork* of the graph and perform feature selection among these new meta-features. Lee et al. (2008) perform the same averaging using pathways from the MsigDB database and thus do not use the connectivity information. Both studies claim an increase in classification performance as well as in stability of the signatures with respect to *single genes* classifiers, i.e. regular signatures containing genes as opposed to groups or averaged features.

On the other hand, Abraham et al. (2010) create meta-genes by averaging the expression of genes over pathways and show that the performance is similar to that of single gene signatures. Recently, Staiger et al. (2012) developed a framework to compare single genes and network-based approaches to conclude that the latter brought no improvement on performance or stability. In chapter 3 (Haury et al., 2010), we showed that incorporating graph information did not improve the performance or the stability of the signature. We only noted an increase in the interpretability of the solution, as it would output genes connected on the graph.

To explain the difference in the results, Staiger et al. (2012) hypothesize that both Chuang et al. (2007) and Lee et al. (2008) reported biased results as they did not properly separate the training and test sets. Moreover, Staiger et al. (2012) studied the matter over six different breast cancer datasets in contrast to the former two studies that used only two, one of which for training and the second as both training and test. From this evidence and personal experience, my conclusion is that network- or pathway-based classifiers do not improve the classification performance or the stability of the signature.

6.3 From data models to black boxes

As microarray data and gene expression profiling became more and more popular, so did feature selection methods. Figure 6.1 shows the number of articles published until october 2012 that meet the following search criteria in PubMed: ("feature selection" OR "variable selection" OR "biomarker discovery") AND ("microarray" OR "microarrays" OR "gene expression profiling").

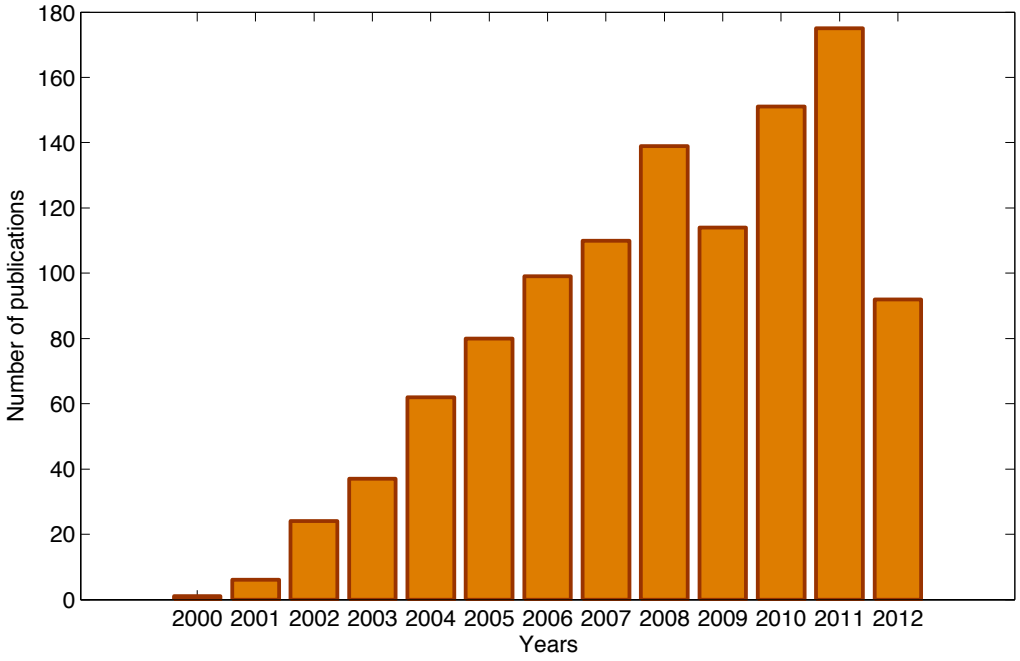


Figure 6.1: **Evolution of the number of publications** related to feature selection from microarray data until october 2012.

The quest for stability

I believe that one reason for the explosion of the number of papers related to this issue has been the very one that motivated this thesis, namely the need for some "closure", in a sense. As stated earlier, the role of the statistician has changed in some drastic way: given a gene expression matrix, we are, as the authors of Iwamoto et al. (2010) put it, "lost in a sea of data". And as it would soon be made clear, at least in some applications, models were drowning. In particular, they were not *stable*, not at the gene level, not at the prediction level, not at the functional level. Now this can be a difficult fact to accept, *assuming* that what we are looking for is there somewhere and we cannot find it. Indeed, we can view stability as a *consistence* measure: if we have developed the right selection method, then it should be consistently returning comparable outputs over the datasets we feed it. Consequently, from this reasoning, these outputs should be correct.

With this in mind, many groups have tackled this *stability* or *robustness* issue, (see, e.g., Davis et al., 2006, Kalousis et al., 2007, Zucknick et al., 2008, Yu et al., 2008, Boulesteix and Slawski, 2009, He and Yu, 2010, Abeel et al., 2010)

It is worth noting that not one single dataset borrowed in this thesis for experimentation has i.i.d. observations - which is probably one of the main reasons why so many state-of-the-art algorithms fail in this context. In particular, the requirement that observations should be *identically distributed* is not met in, e.g., the signature selection datasets, since they are composed of samples from different cancer subtypes or were obtained through different experimental protocols. Correlation between gene expression levels was discussed throughout this thesis as a major issue for feature selection. Causality, on the other hand, was not, at least not explicitly. At first glance, whether gene expression changes are causes or consequences of, e.g., a metastatic event, could seem of low interest for prediction purposes. Consider a simple example where g_1 has a direct bearing to the response and changes in g_2 and g_3 are two possible consequences of changes in g_1 in different situations. Both g_2 and g_3 could still have a great predictive power. However, since our observations are not identically distributed, isn't it possible that such secondary causal relations would be affected such that g_1 would be selected on some subset along with the true cause g_2 and that the same g_2 would be chosen simultaneously with g_3 on another one?

From my point of view, and in a way that seems only natural, the complexity of the feature selection methods published in this field has escalated and often with the goal of improving the stability of the methods. In particular, since the techniques were not stable, a good idea has been to *embed* stability in them. If we design a method to return the variables that are consistently selected over different subsamples of the data, or by different methods, then by construction, it should be more stable itself, i.e., produce similar results when the training set changes.

Accuracy or stability: which one is it?

While looking for stability, we seem to be facing a stability/accuracy trade-off, which makes sense in at least two ways:

The bias/variance trade-off for feature selection:

following Han and Yu (2010), assume a feature selection method f that produces a *relevance score* r for a given feature X. We are interested in the behavior of r accross changes in the dataset D. D can be seen as a random variable with distribution $p(D)$. What we call *stability or robustness* can be seen as the opposite of the following quantity

$$\text{Var}(X) = \mathbb{E}[r(D) - \mathbb{E}(r(D))]^2$$

which represents the *variance* of f under $p(D)$. In other words, it answers the question "how much does r change when D changes?". Similarly, assuming that r^* is the *true* relevance score

of X, the bias of f under $p(D)$ is by definition

$$\text{Bias}(X) = \mathbb{E}(r(D)) - r^*$$

and relates to the accuracy of f, answering the question "are the features correctly chosen over D?". And just as in the usual setting, the total error (relatively to the relevance of X) can be decomposed into the bias and the variance. Over the entire set of p features, we thus have:

$$\frac{1}{p}\sum_{j=1}^{p}\text{Err}(X_j) = \frac{1}{p}\sum_{j=1}^{p}\text{Bias}^2(X_j) + \frac{1}{p}\sum_{j=1}^{p}\text{Var}(X_j).$$

This decomposition sheds light on the trade-off between predictive accuracy and the stability we observed in Chapters 2 and 4. Indeed, it makes sense that the correct features (i.e., a low bias) would yield a better *predictive accuracy* although the two concepts are different.

Stability is linked with redundancy:

embedding stability within a feature selection method by, e.g., resampling or bootstrapping the examples to obtain several variable subsets and aggregate them, can be seen as a way to allow a method to select correlated variables. A well-known pitfall of Lasso (Tibshirani, 1996) is that, when it faces two correlated variables, it chooses one of them. Stability Selection (Bach, 2008, Meinshausen and Bühlmann, 2010) was designed with one argument being that it would allow the design to violate the *irrepresentable condition* (Zhao and Yu, 2006). In other words, it would make it possible to select correlated/redundant variables. Meinshausen and Bühlmann (2010) also motivate the algorithm with a signature selection example, and suggest that Stability Selection will improve the (no suspense here) stability of the signature.

This begs a question: when we look for *stability*, are we actually looking for methods that will select *redundant covariates*?

Some experiments I showed in Chapter 4 seem to indicate that the redundancy of variables selected by a method and the stability of said method have indeed created a loving (and linear) relationship. The more stable the method, the more redundant the features it selects, and that's that.

Some works (Koller and Sahami, 1996, Yu and Liu, 2004, Peng et al., 2005) advertise for *low redundancy* methods on the basis that redundant features do not improve prediction accuracy.

Chasing our tail yet? If stability implies redundancy and redundancy implies lower accuracy, then it seems difficult to simultaneously obtain stability and accuracy.

Accuracy it is!

I am now convinced: accuracy is what we should focus on, no matter how little we understand why or how we obtain it. I have not always been convinced. Actually, it is fair to say that I have changed my mind over the years, as more and more evidence would challenge my initial beliefs

until I would finally give up on them. I remember having an argument, once, with a good friend of mine. "Signatures will never be stable", he said, "this is a done deal, forget it". I was very upset that day, as the perspective of spending three years on a "done" problem did not quite appeal to me. As of today, I think that he was right. However, I am happy I did go on and keep investigating the issue, as I feel that I can make sense of it now. Well, to some extent.

So, black boxes? In a recent talk about Random Forests, Gérard Biau, who has been working on theoretical aspects of this algorithm (Biau, 2012), confessed he still did not completely understand why it worked so well. Breiman himself was not sure (Breiman, 2001b). Yet, it does work well. For instance, in Chapter 5, we mentioned that the method working best on the GRN inference problem was based on Random Forests (Huynh-Thu et al., 2010).

Looking back at it, why TIGRESS provided good performances is not clear either. Basically, we started from a linear model. Subsequently, we randomized the observations, we randomized the features and, all in all, I believe that we obtained a box, that, if not black, is at least dark gray. We showed how increasing randomization would increase the accuracy. However, I am not sure why. In Chapter 4, I tried *extreme randomization* which resulted in a higher accuracy as well. But the layers of randomization make the reason for this incomprehensible to me.

Here is my final thought: if there were a strong signal in the data, we would need no such thing as randomization to obtain models that would be *accurate, stable and irredundant*, which is obviously what we are seeking. But it appears that there is not, or, at least, that it is not strong enough. Maybe the data does simply not contain the necessary information to our prediction problems. Or maybe it does, or it would, if we had access to many more samples. Either way, my point is that, until then, I believe that we have to let go of interpretability and understanding and focus on the performance. Actually, I will let Leo Breiman have the final word since what I have just understood, he had already written about some ten years ago:

> *The point of a model is to get useful information about the relation between the response and predictor variables. Interpretability is a way of getting information. But a model does not have to be simple to provide reliable information about the relation between predictor and response variables.*

Bibliography

Abeel, T., Helleputte, T., Van de Peer, Y., Dupont, P., and Saeys, Y. (2010). Robust biomarker identification for cancer diagnosis with ensemble feature selection methods. *Bioinformatics*, 26:392–398.

Abraham, G., Kowalczyk, A., Loi, S., Haviv, I., and Zobel, J. (2010). Prediction of breast cancer prognosis using gene set statistics provides signature stability and biological context. *BMC Bioinformatics*, 11(1):277.

Akaike, H. (1973). Information theory and an extension of the maximum likelihood principle. In N., P. B. and F., C., editors, *Proc. of the 2nd Int. Symp. on Information Theory*, pages 267–281.

Akutsu, T., Miyano, S., and Kuhara, S. (2000). Algorithms for identifying Boolean networks and related biological networks based on matrix multiplication and fingerprint function. *J. Comput. Biol.*, 7(3-4):331–343.

Alexander, E. K., Kennedy, G. C., Baloch, Z. W., Cibas, E. S., Chudova, D., Diggans, J., Friedman, L., Kloos, R. T., LiVolsi, V. A., Mandel, S. J., et al. (2012). Preoperative diagnosis of benign thyroid nodules with indeterminate cytology. *New England Journal of Medicine*, 367(8):705–715.

Alizadeh, A. A., Eisen, M. B., Davis, R. E., Ma, C., Lossos, I. S., Rosenwald, A., Boldrick, J. C., Sabet, H., Tran, T., Yu, X., Powell, J. I., Yang, L., Marti, G. E., Moore, T., Hudson, J., Lu, L., Lewis, D. B., Tibshirani, R., Sherlock, G., Chan, W. C., Greiner, T. C., Weisenburger, D. D., Armitage, J. O., Warnke, R., Levy, R., Wilson, W., Grever, M. R., Byrd, J. C., Botstein, D., Brown, P. O., and Staudt, L. M. (2000). Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature*, 403(6769):503–511.

Ambroise, C. and McLachlan, G. (2002). Selection bias in gene extraction on the basis of microarray gene-expression data. *Proc. Natl. Acad. Sci. USA*, 99(10):6562–6566.

Argyriou, A., Foygel, R., and Srebro, N. (2012). Sparse prediction with the k-overlap norm. *arXiv preprint arXiv:1204.5043*.

Arkin, A., Shen, P., and Ross, J. (1997). A test case of correlation metric construction of a reaction pathway from measurements. *Science*, 277(5330):1275–1279.

Bach, F., Jenatton, R., Mairal, J., and Obozinski, G. (2011a). Optimization with sparsity-inducing penalties. *Foundations and Trends® in Machine Learning*, 4(1):1–106.

Bach, F., Jenatton, R., Mairal, J., and Obozinski, G. (2011b). Structured sparsity through convex optimization. *arXiv preprint arXiv:1109.2397*.

Bach, F. R. (2008). Bolasso: model consistent Lasso estimation through the bootstrap. In Cohen, W. W., McCallum, A., and Roweis, S. T., editors, *Proceedings of the 25th international conference on Machine learning*, volume 308 of *ACM International Conference Proceeding Series*, pages 33–40, New York, NY, USA. ACM.

Bansal, M., Della Gatta, G., and Bernardo, D. (2006). Inference of gene regulatory networks and compound mode of action from time course gene expression profiles. *Bioinformatics*, 22(7):815–822.

Barrett, T., Troup, D., Wilhite, S., Ledoux, P., Evangelista, C., Kim, I., Tomashevsky, M., Marshall, K., Phillippy, K., Sherman, P., et al. (2011). Ncbi geo: archive for functional genomics data sets - 10 years on. *Nucleic acids research*, 39(suppl 1):D1005–D1010.

Barrett, T., Troup, D., Wilhite, S., Ledoux, P., Rudnev, D., Evangelista, C., Kim, I., Soboleva, A., Tomashevsky, M., Marshall, K., et al. (2009). Ncbi geo: archive for high-throughput functional genomic data. *Nucleic acids research*, 37(suppl 1):D885–D890.

Beck, A. and Teboulle, M. (2009). A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Img. Sci.*, 2(1):183–202.

Bellman, R. and Kalaba, R. (1959). On adaptive control processes. *Automatic Control, IRE Transactions on*, 4(2):1–9.

Benjamini, Y. and Hochberg, Y. (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. *J. R. Stat. Soc. Ser. B*, 57:289–300.

Bernardo, D., Thompson, M. J., Gardner, T. S., Chobot, S. E., Eastwood, E. L., Wojtovich, A. P., Elliott, S. J., Schaus, S. E., and Collins, J. J. (2005). Chemogenomic profiling on a genome-wide scale using reverse-engineered gene networks. *Nat. Biotechnol.*, 23(3):377–383.

Bi, J., Bennett, K., Embrechts, M., Breneman, C., and Song, M. (2003). Dimensionality reduction via sparse support vector machines. *J. Mach. Learn. Res.*, 3:1229–1243.

Biau, G. (2012). Analysis of a random forests model. *The Journal of Machine Learning Research*, 98888:1063–1095.

Bishop, C. (2006). *Pattern recognition and machine learning*. Springer.

Bolstad, B., Irizarry, R., Åstrand, M., and Speed, T. (2003). A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics*, 19(2):185–193.

Boulesteix, A. (2010). Over-optimism in bioinformatics research. *Bioinformatics*, 26(3):437–439.

Boulesteix, A. and Slawski, M. (2009). Stability and aggregation of ranked gene lists. *Briefings in bioinformatics*, 10(5):556–568.

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press, New York, NY, USA.

Breiman, L. (2001a). Random forests. *Mach. Learn.*, 45(1):5–32.

Breiman, L. (2001b). Statistical modeling: The two cultures (with comments and a rejoinder by the author). *Statistical Science*, 16(3):199–231.

Breiman, L., Friedman, J., Stone, C., and Olshen, R. (1984). *Classification and regression trees*. Chapman & Hall/CRC.

Butte, A. J., Tamayo, P., Slonim, D., Golub, T. R., and Kohane, I. S. (2000). Discovering functional relationships between RNA expression and chemotherapeutic susceptibility using relevance networks. *Proc. Natl. Acad. Sci. USA*, 97(22):12182–12186.

Chen, K.-C., Wang, T.-Y., Tseng, H.-H., Huang, C.-Y. F., and Kao, C.-Y. (2005). A stochastic differential equation model for quantifying transcriptional regulatory network in Saccharomyces cerevisiae. *Bioinformatics*, 21(12):2883–2890.

Chen, S. S., Donoho, D. L., and Saunders, M. (1998). Atomic decomposition by basis pursuit. *SIAM J. Sci. Comput.*, 20(1):33–61.

Chen, T., He, H. L., and Church, G. M. (1999). Modeling gene expression with differential equations. *Pac. Symp. Biocomput.*, 4:29–40.

Chuang, H.-Y., Lee, E., Liu, Y.-T., Lee, D., and Ideker, T. (2007). Network-based classification of breast cancer metastasis. *Mol. Syst. Biol.*, 3:140.

Dai, M., Wang, P., Boyd, A., Kostov, G., Athey, B., Jones, E., Bunney, W., Myers, R., Speed, T., Akil, H., et al. (2005). Evolving gene/transcript definitions significantly alter the interpretation of genechip data. *Nucleic acids research*, 33(20):e175–e175.

Daubechies, I., Defrise, M., and De Mol, C. (2004). An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics*, 57(11):1413–1457.

Davis, C., Gerick, F., Hintermair, V., Friedel, C., Fundel, K., Küffner, R., and Zimmer, R. (2006). Reliable gene signatures for microarray classification: assessment of stability and performance. *Bioinformatics*, 22(19):2356–2363.

Díaz-Uriarte, R. and De Andres, S. (2006). Gene selection and classification of microarray data using random forest. *BMC bioinformatics*, 7(1):3.

Drier, Y. and Domany, E. (2011). Do two machine-learning based prognostic signatures for breast cancer capture the same biological processes? *PloS one*, 6(3):e17795.

Dudoit, S., Fridlyand, J., and Speed, T. (2002). Comparison of discrimination methods for classification of tumors using gene expression data. *J. Am. Stat. Assoc.*, 97:77–87.

Efron, B. (1979). Bootstrap methods: another look at the jackknife. *Ann. Stat.*, 7(1):1–26.

Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004). Least angle regression. *Ann. Stat.*, 32(2):407–499.

Ein-Dor, L., Kela, I., Getz, G., Givol, D., and Domany, E. (2005). Outcome signature genes in breast cancer: is there a unique set? *Bioinformatics*, 21(2):171–178.

Ein-Dor, L., Zuk, O., and Domany, E. (2006). Thousands of samples are needed to generate a robust gene list for predicting outcome in cancer. *Proc. Natl. Acad. Sci. USA*, 103(15):5923–5928.

Faith, J., Driscoll, M., Fusaro, V., Cosgrove, E., Hayete, B., Juhn, F., Schneider, S., and Gardner, T. (2008). Many microbe microarrays database: uniformly normalized affymetrix compendia with structured experimental metadata. *Nucleic Acids Res.*, 36(Database issue):D866–D870.

Faith, J. J., Hayete, B., Thaden, J. T., Mogno, I., Wierzbowski, J., Cottarel, G., Kasif, S., Collins, J. J., and Gardner, T. S. (2007). Large-scale mapping and validation of Escherichia coli transcriptional regulation from a compendium of expression profiles. *PLoS Biol.*, 5(1):e8.

Fan, C., Oh, D., Wessels, L., Weigelt, B., Nuyten, D., Nobel, A., van't Veer, L., and Perou, C. (2006). Concordance among gene-expression-based predictors for breast cancer. *N. Engl. J. Med.*, 355(6):560.

Fisher, R. (1950). Gene frequencies in a cline determined by selection and diffusion. *Biometrics*, 6(4):353–361.

Freedman, D. (2006). Statistical models for causation what inferential leverage do they provide? *Evaluation Review*, 30(6):691–713.

Friedman, N. (2004). Inferring cellular networks using probabilistic graphical models. *Science*, 303(5659):799.

Friedman, N., Linial, M., Nachman, I., and Pe'er, D. (2000). Using Bayesian networks to analyze expression data. *J. Comput. Biol.*, 7(3-4):601–620.

Furnival, G. and Wilson, R. (1974). Regressions by leaps and bounds. *Technometrics*, 16(4):499–511.

Galton, S. (1869). *Hereditary genius*. Macmillan and Company.

Gama-Castro, S., Salgado, H., Peralta-Gil, M., Santos-Zavaleta, A., Muñiz-Rascado, L., Solano-Lira, H., Jimenez-Jacinto, V., Weiss, V., García-Sotelo, J. S., López-Fuentes, A., Porrón-Sotelo, L., Alquicira-Hernández, S., Medina-Rivera, A., Martínez-Flores, I., Alquicira-Hernández, K., Martínez-Adame, R., Bonavides-Martínez, C., Miranda-Ríos, J., Huerta, A. M., Mendoza-Vargas, A., Collado-Torres, L., Taboada, B., Vega-Alvarado, L., Olvera, M., Olvera, L., Grande, R., Morett, E., and Collado-Vides, J. (2011). RegulonDB version 7.0: transcriptional regulation of Escherichia coli K-12 integrated within genetic sensory response units (gensor units). *Nucleic Acids Res.*, 39(suppl 1):D98–D105.

Gardner, T. S., Bernardo, D., Lorenz, D., and Collins, J. J. (2003). Inferring genetic networks and identifying compound mode of action via expression profiling. *Science*, 301(5629):102–105.

Genuer, R., Poggi, J., and Tuleau-Malot, C. (2010). Variable selection using random forests. *Pattern Recognition Letters*, 31(14):2225–2236.

Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P., Coller, H., Loh, M. L., Downing, J. R., Caligiuri, M. A., Bloomfield, C. D., and Lander, E. S. (1999). Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537.

Guorong, X., Peiqi, C., and Minhui, W. (1996). Bhattacharyya distance feature selection. In *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*, volume 2, pages 195–199. IEEE.

Guyon, I. and Elisseeff, A. (2003). An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182.

Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Mach. Learn.*, 46(1/3):389–422.

Haibe-Kains, B., Desmedt, C., Piette, F., Buyse, M., Cardoso, F., Van't Veer, L., Piccart, M., Bontempi, G., and Sotiriou, C. (2008a). Comparison of prognostic gene expression signatures for breast cancer. *BMC Genomics*, 9:394.

Haibe-Kains, B., Desmedt, C., Sotiriou, C., and Bontempi, G. (2008b). A comparative study of survival models for breast cancer prognostication based on microarray data: does a single gene beat them all? *Bioinformatics*, 24(19):2200–2208.

Han, Y. and Yu, L. (2010). A variance reduction framework for stable feature selection. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 206–215. IEEE.

Hartemink, A., Gifford, D., Jaakkola, T., and Young, R. (2002). Using graphical models and genomic expression data to statistically validate models of genetic regulatory networks. In Altman, R. B., Dunker, A. K., Hunter, L., Lauerdale, K., and Klein, T. E., editors, *Proceedings of the Pacific Symposium on Biocomputing 2002*, pages 422–433. World Scientific.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The elements of statistical learning: data mining, inference and prediction*. Springer, 2 edition.

Haury, A., Jacob, L., and Vert, J. (2010). Increasing stability and interpretability of gene expression signatures. *arXiv preprint arXiv:1001.3109*.

Haury, A.-C., Gestraud, P., and Vert, J.-P. (2011). The influence of feature selection methods on accuracy, stability and interpretability of molecular signatures. *PLoS One*, 6(12):e28210.

Haury, A.-C., Mordelet, F., Vera-Licona, P., and Vert, J.-P. (2012). Tigress: Trustful inference of gene regulation using stability selection. *BMC Systems Biology*, 6(1):145.

He, B., Yang, H., and Wang, S. (2000). Alternating direction method with self-adaptive penalty parameters for monotone variational inequalities. *Journal of Optimization Theory and applications*, 106(2):337–356.

He, Z. and Yu, W. (2010). Stable feature selection for biomarker discovery. *arXiv preprint arXiv:1001.0887*.

Hoerl, A. E. and Kennard, R. W. (1970). Ridge regression : biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67.

Hornberger, J., Alvarado, M., Rebecca, C., Gutierrez, H., Tiffany, M., and Gradishar, W. (2012). Clinical validity/utility, change in practice patterns, and economic implications of risk stratifiers to predict outcomes for early-stage breast cancer: A systematic review. *Journal of the National Cancer Institute*, 104(14):1068–1079.

Huang, J., Zhang, T., and Metaxas, D. (2009). Learning with structured sparsity. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 417–424. ACM.

Huynh-Thu, V. A., Irrthum, A., Wehenkel, L., and Geurts, P. (2010). Inferring regulatory networks from expression data using tree-based methods. *PLoS One*, 5(9):e12776.

Ioannidis, J. (2005a). Why most published research findings are false. *PLoS medicine*, 2(8):e124.

Ioannidis, J. P. A. (2005b). Microarrays and molecular research: noise discovery? *Lancet*, 365(9458):454.

Irizarry, R. A., Hobbs, B., Collin, F., Beazer-Barclay, Y. D., Antonellis, K. J., Scherf, U., and Speed, T. P. (2003). Exploration, normalization, and summaries of high density oligonucleotide array probe level datas. *Biostatistics*, 4(2):249–264.

Ivshina, A., George, J., Senko, O., Mow, B., Putti, T., Smeds, J., Lindahl, T., Pawitan, Y., Hall, P., Nordgren, H., et al. (2006). Genetic reclassification of histologic grade delineates new clinical subtypes of breast cancer. *Cancer research*, 66(21):10292–10301.

Iwamoto, T., Pusztai, L., et al. (2010). Predicting prognosis of breast cancer with gene signatures: are we lost in a sea of data? *Genome medicine*, 2(11):81.

Jacob, L., Obozinski, G., and Vert, J.-P. (2009). Group lasso with overlap and graph lasso. In *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning*, pages 433–440, New York, NY, USA. ACM.

Jenatton, R., Audibert, J.-Y., and Bach, F. (2009). Structured variable selection with sparsity-inducing norms. Technical Report 0904.3523, arXiv.

Jenatton, R., Mairal, J., Obozinski, G., and Bach, F. (2011). Proximal methods for hierarchical sparse coding. *J. Mach. Learn. Res.*, 12(Jul):2297–2334.

Kalousis, A., Prados, J., and Hilario, M. (2007). Stability of feature selection algorithms: a study on high-dimensional spaces. *Knowledge and information systems*, 12(1):95–116.

Koh, K., Kim, S., and Boyd, S. (2007). An interior-point method for large-scale l1-regularized logistic regression. *Journal of Machine learning research*, 8(8):1519–1555.

Kohavi, R. and John, G. (1997). Wrappers for feature selection. *Artificial Intelligence*, 97(1-2):273–324.

Koller, D. and Sahami, M. (1996). Toward optimal feature selection.

Küffner, R., Petri, T., Tavakkolkhah, P., Windhager, L., and Zimmer, R. (2012). Inferring gene regulatory networks by ANOVA. *Bioinformatics*.

Lai, C., Reinders, M. J. T., van't Veer, L. J., and Wessels, L. F. A. (2006). A comparison of univariate and multivariate gene selection techniques for classification of cancer datasets. *BMC bioinformatics*, 7(1):235.

Langford, E., Schwertman, N., and Owens, M. (2001). Is the property of being positively correlated transitive? *The American Statistician*, 55(4):322–325.

Lazar, C., Taminau, J., Meganck, S., Steenhoff, D., Coletta, A., Molter, C., de Schaetzen, V., Duque, R., Bersini, H., and Nowé, A. (2012). A survey on filter techniques for feature selection in gene expression microarray analysis. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 9(4):1106–1119.

Lee, E., Chuang, H.-Y., Kim, J. W., Ideker, T., and Lee, D. (2008). Inferring pathway activity toward precise disease classification. *PLoS Comput Biol*, 4(11):e1000217.

Liang, S., Fuhrman, S., and Somogyi, R. (1998). REVEAL, a general reverse engineering algorithm for inference of genetic network architectures. *Pac. Symp. Biocomput.*, 3:18–29.

Liu, H. and Motoda, H. (2007). *Computational methods of feature selection*. Chapman & Hall/CRC.

Lockhart, D., Winzeler, E., et al. (2000). Genomics, gene expression and dna arrays. *NATURE-LONDON-*, pages 827–836.

Lodish, H., Berk, A., Zipursky, S., Matsudaira, P., Baltimore, D., and Darnell, J. (2000). Molecular cell biology. *New York*.

Mairal, J. (2010). *Sparse coding for machine learning, image processing and computer vision*. PhD thesis, École normale supérieure de Cachan-ENS Cachan.

Mairal, J., Bach, F., Ponce, J., and Sapiro, G. (2010). Online learning for matrix factorization and sparse coding. *J. Mach. Learn. Res.*, 11:19–60.

Mann, H. and Whitney, D. (1947). On a test of whether one of two random variables is stochastically larger than the other. *The annals of mathematical statistics*, 18(1):50–60.

Marbach, D., Costello, J., Küffner, R., Vega, N., Prill, R., Camacho, D., Allison, K., the DREAM5 Consortium, Kellis, M., Collins, J., and Stolovitzky, G. (2012). Wisdom of crowds for robust gene network inference. *Nat. Methods*, 9(8):796–804.

Marbach, D., Prill, R. J., Schaffter, T., Mattiussi, C., Floreano, D., and Stolovitzky, G. (2010). Revealing strengths and weaknesses of methods for gene network inference. *Proc. Natl. Acad. Sci. USA*, 107(14):6286–6291.

Marbach, D., Schaffter, T., Mattiussi, C., and Floreano, D. (2009). Generating realistic in silico gene networks for performance assessment of reverse engineering methods. *J. Comput. Biol.*, 16(2):229–239.

Margolin, A. A., Nemenman, I., Basso, K., Wiggins, C., Stolovitzky, G., Dalla Favera, R., and Califano, A. (2006). ARACNE: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular contexts. *BMC Bioinformatics*, 7 Suppl 1:S7.

Markowetz, F. and Spang, R. (2007). Inferring cellular networks - a review. *BMC Bioinformatics*, 8(Suppl 6):S5.

Meier, L., van de Geer, S., and Bühlmann, P. (2008). The group lasso for logistic regression. *J. R. Stat. Soc. Ser. B*, 70(1):53–71.

Meinshausen, N. and Buehlmann, P. (2009). Stability selection.

Meinshausen, N. and Bühlmann, P. (2006). High dimensional graphs and variable selection with the lasso. *Ann. Stat.*, 34:1436–1462.

Meinshausen, N. and Bühlmann, P. (2010). Stability selection. *J. R. Stat. Soc. Ser. B*, 72(4):417–473.

Menard, S. (2001). *Applied logistic regression analysis*, volume 106. Sage Publications, Incorporated.

Michiels, S., Koscielny, S., and Hill, C. (2005). Prediction of cancer outcome with microarrays: a multiple random validation strategy. *Lancet*, 365(9458):488–492.

Mordelet, F. and Vert, J.-P. (2008). SIRENE: Supervised inference of regulatory networks. *Bioinformatics*, 24(16):i76–i82.

Moreau, J.-J. (1965). Proximité et dualité dans un espace hilbertien. *Bulletin de la S.M.F.*, 93:273–299.

Mosley, J. and Keri, R. (2008). Cell cycle correlated genes dictate the prognostic power of breast cancer gene lists. *BMC Medical Genomics*, 1(1):11.

Narendra, P. and Fukunaga, K. (1977). A branch and bound algorithm for feature subset selection. *Computers, IEEE Transactions on*, 100(9):917–922.

Nesterov, Y. and Nemirovsky, A. (1994). Interior point polynomial methods in convex programming: Theory and applications. *SIAM*.

Obozinski, G., Jacob, L., and Vert, J.-P. (2011). Group lasso with overlaps: the latent group lasso approach. Technical report, arXiv.

Obozinski, G., Wainwright, M. J., and Jordan, M. I. (2008). Union support recovery in high-dimensional multivariate regression. Technical Report 0808.0711v1, arXiv.

Osborne, M. R., Presnell, B., and Turlach, B. A. (1999). On the lasso and its dual. *Journal of Computational and Graphical Statistics*, 9:319–337.

Paik, S. (2007). Development and clinical utility of a 21-gene recurrence score prognostic assay in patients with early breast cancer treated with tamoxifen. *Oncologist*, 12(6):631–635.

Pawitan, Y., Bjöhle, J., Amler, L., Borg, A., Egyhazi, S., Hall, P., Han, X., Holmberg, L., Huang, F., Klaar, S., et al. (2005). Gene expression profiling spares early breast cancer patients from adjuvant therapy: derived and validated in two population-based cohorts. *Breast Cancer Research*, 7(6):R953.

Peng, H., Long, F., and Ding, C. (2005). Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(8):1226–1238.

Perou, C. M., Sørlie, T., Eisen, M. B., van de Rijn, M., Jeffrey, S. S., Rees, C. A., Pollack, J. R., Ross, D. T., Johnsen, H., Akslen, L. A., Fluge, O., Pergamenschikov, A., Williams, C., Zhu, S. X., Lønning, P. E., Børresen-Dale, A. L., Brown, P. O., and Botstein, D. (2000). Molecular portraits of human breast tumours. *Nature*, 406(6797):747–752.

Perrin, B., Ralaivola, L., Mazurie, A., Bottani, S., Mallet, J., and d'Alche Buc, F. (2003). Gene networks inference using dynamic bayesian networks. *Bioinformatics*, 19(suppl 2):ii138–ii148.

Popovici, V., Chen, W., Gallas, B., Hatzis, C., Shi, W., Samuelson, F., Nikolsky, Y., Tsyganova, M., Ishkin, A., Nikolskaya, T., et al. (2010). Effect of training-sample size and classification difficulty on the accuracy of genomic predictors. *Breast Cancer Res*, 12(1):R5.

Quackenbush, J. (2002). Microarray data normalization and transformation. *Nat Genet*, 32 Suppl:496–501.

Ramaswamy, S., Ross, K. N., Lander, E. S., and Golub, T. R. (2003). A molecular signature of metastasis in primary solid tumors. *Nat. Genet.*, 33(1):49–54.

Ramaswamy, S., Tamayo, P., Rifkin, R., Mukherjee, S., Yeang, C., Angelo, M., Ladd, C., Reich, M., Latulippe, E., Mesirov, J., Poggio, T., Gerald, W., Loda, M., Lander, E., and Golub, T. (2001). Multiclass cancer diagnosis using tumor gene expression signatures. *Proc. Natl. Acad. Sci. USA*, 98(26):15149–15154.

Rapaport, F., Barillot, E., and Vert, J.-P. (2008). Classification of arrayCGH data using fused SVM. *Bioinformatics*, 24(13):i375–i382.

Rapaport, F., Zynoviev, A., Dutreix, M., Barillot, E., and Vert, J.-P. (2007). Classification of microarray data using gene networks. *BMC Bioinformatics*, 8:35.

Ravetti, M. G. and Moscato, P. (2008). Identification of a 5-protein biomarker molecular signature for predicting alzheimer's disease. *PloS one*, 3(9):e3111.

Reyal, F., van Vliet, M. H., Armstrong, N. J., Horlings, H. M., de Visser, K. E., Kok, M., Teschendorff, A. E., Mook, S., van't Veer, L., Caldas, C., Salmon, Remy, R. J., van de Vijver, M. J., and Wessels, L. F. A. (2008). A comprehensive analysis of prognostic signatures reveals the high predictive capacity of the proliferation, immune response and RNA splicing modules in breast cancer. *Breast Cancer Res.*, 10(6):R93.

Rice, J., Tu, Y., and Stolovitzky, G. (2005). Reconstructing biological networks using conditional correlation analysis. *Bioinformatics*, 21(6):765–773.

Rosset, S. and Zhu, J. (2007). Piecewise linear regularized solution paths. *Annals of Statistics*, 35(3):1012–1030.

Saeys, Y., Inza, I., and Larrañaga, P. (2007). A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19):2507–2517.

Schaffter, T., Marbach, D., and Floreano, D. (2011). Genenetweaver: in silico benchmark generation and performance profiling of network inference methods. *Bioinformatics*, 27(16):2263–2270.

Schwarz, G. (1978). Estimating the dimension of a model. *Annals of Statistics*, 6:461–464.

Shen, R., Chinnaiyan, A. M., and Ghosh, D. (2008). Pathway analysis reveals functional convergence of gene expression profiles in breast cancer. *BMC Medical Genomics*, 1(1):28.

Shevade, S. and Keerthi, S. (2003). A simple and efficient algorithm for gene selection using sparse logistic regression. *Bioinformatics*, 19(17):2246–2253.

Shi, W., Bessarabova, M., Dosymbekov, D., Dezso, Z., Nikolskaya, T., Dudoladova, M., Serebryiskaya, T., Bugrim, A., Gyuryanov, A., Brennan, R. J., Shah, R., Dopazo, J., Chen, M., Deng, Y., Shi, T., Jurman, G., Furnlanelle, C., Thomas, R. S., Corton, J. C., Tong, W., Shi, L., and Nikolsky, Y. (2010). Functional analysis of multiple genomic signatures demonstrates that classification algorithms choose phenotype-related genes. *Pharmacogenomics J.*, 10(4):310–323.

Simon, R. (2008). Lost in translation problems and pitfalls in translating laboratory observations to clinical utility. *European journal of cancer (Oxford, England: 1990)*, 44(18):2707.

Simon, R., Radmacher, M., Dobbin, K., and McShane, L. (2003). Pitfalls in the use of dna microarray data for diagnostic and prognostic classification. *Journal of the National Cancer Institute*, 95(1):14–18.

Smoot, M., Ono, K., Ruscheinski, J., Wang, P., and Ideker, T. (2011). Cytoscape 2.8: new features for data integration and network visualization. *Bioinformatics*, 27(3):431–432.

Sotiriou, C. and Pusztai, L. (2009). Gene-expression signatures in breast cancer. *N. Engl. J. Med.*, 360(8):790–800.

Sotiriou, C., Wirapati, P., Loi, S., Harris, A., Fox, S., Smeds, J., Nordgren, H., Farmer, P., Praz, V., Haibe-Kains, B., Desmedt, C., Larsimont, D., Cardoso, F., Peterse, H., Nuyten, D., Buyse, M., de Vijver, M. J. V., Bergh, J., Piccart, M., and Delorenzi, M. (2006). Gene expression profiling in breast cancer: understanding the molecular basis of histologic grade to improve prognosis. *J Natl Cancer Inst*, 98(4):262–272.

Staiger, C., Cadot, S., Kooter, R., Dittrich, M., Müller, T., Klau, G., and Wessels, L. (2012). A critical evaluation of network and pathway-based classifiers for outcome prediction in breast cancer. *PloS one*, 7(4):e34796.

Strobl, C., Boulesteix, A., Zeileis, A., and Hothorn, T. (2007). Bias in random forest variable importance measures: Illustrations, sources and a solution. *BMC bioinformatics*, 8(1):25.

Tegner, J., Yeung, M. K. S., Hasty, J., and Collins, J. J. (2003). Reverse engineering gene networks: integrating genetic perturbations with dynamical modeling. *Proc. Natl. Acad. Sci. USA*, 100(10):5944–5949.

Thomassen, M., Tan, Q., Eiriksdottir, F., Bak, M., Cold, S., and Kruse, T. (2007). Comparison of gene sets for expression profiling: prediction of metastasis from low-malignant breast cancer. *Clinical Cancer Research*, 13(18):5355–5360.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *J. R. Stat. Soc. Ser. B*, 58(1):267–288.

Tibshirani, R., Saunders, M., Rosset, S., Zhu, J., and Knight, K. (2005). Sparsity and smoothness via the fused lasso. *J. R. Stat. Soc. Ser. B Stat. Methodol.*, 67(1):91–108.

Tikhonov, A. (1963). Solution of incorrectly problems and the regularization method. *Soviet Mathematics Doklady*, 4:1035–1038.

Tomioka, R. and Sugiyama, M. (2008). Sparse learning with duality gap guarantee. Technical report, Department of Computer Science; Graduate School of Information Science and Engineering, Tokyo Institute of Technology, 152-8552, Tokyo, Japan.

Tomioka, R., Suzuki, T., and Sugiyama, M. (2009). Super-linear convergence of dual augmented-lagrangian algorithm for sparsity regularized estimation. *arXiv preprint arXiv:0911.4046*.

van de Vijver, M. J., He, Y. D., van't Veer, L. J., Dai, H., Hart, A. A. M., Voskuil, D. W., Schreiber, G. J., Peterse, J. L., Roberts, C., Marton, M. J., Parrish, M., Atsma, D., Witteveen, A., Glas, A., Delahaye, L., van der Velde, T., Bartelink, H., Rodenhuis, S., Rutgers, E. T., Friend, S. H., and Bernards, R. (2002). A gene-expression signature as a predictor of survival in breast cancer. *N. Engl. J. Med.*, 347(25):1999–2009.

van 't Veer, L. J., Dai, H., van de Vijver, M. J., He, Y. D., Hart, A. A. M., Mao, M., Peterse, H. L., van der Kooy, K., Marton, M. J., Witteveen, A. T., Schreiber, G. J., Kerkhoven, R. M., Roberts, C., Linsley, P. S., Bernards, R., and Friend, S. H. (2002). Gene expression profiling predicts clinical outcome of breast cancers. *Nature*, 415(6871):530–536.

Vapnik, V. and Chervonenkis, A. Y. (1974). Teoriya raspoznavaniya obrazov: Statisticheskie problemy obucheniya. (Russian) [Theory of Pattern Recognition: Statistical Problems of Learning]. Moscow: Nauka.

Vapnik, V. N. (1995). *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA.

Vapnik, V. N. (1998). *Statistical Learning Theory*. Wiley, New-York.

Venet, D., Dumont, J., and Detours, V. (2011). Most random gene expression signatures are significantly associated with breast cancer outcome. *PLoS computational biology*, 7(10):e1002240.

Wang, J., Du, Z., Payattakool, R., Yu, P., and Chen, C. (2007). A new method to measure the semantic similarity of GO terms. *Bioinformatics*, 23(10):1274.

Wang, Y., Klijn, J., Zhang, Y., Sieuwerts, A., Look, M., Yang, F., Talantov, D., Timmermans, M., Meijer-van Gelder, M., Yu, J., Jatkoe, T., Berns, E., Atkins, D., and Foekens, J. (2005). Gene-expression profiles to predict distant metastasis of lymph-node-negative primary breast cancers. *Lancet*, 365(9460):671–679.

Weisberg, S. (1981). *Applied linear regression*. Wiley, New-York.

Whitfield, M., George, L., Grant, G., and Perou, C. (2006). Common markers of proliferation. *Nature Reviews Cancer*, 6(2):99–106.

Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83.

Wirapati, P., Sotiriou, C., Kunkel, S., Farmer, P., Pradervand, S., Haibe-Kains, B., Desmedt, C., Ignatiadis, M., Sengstag, T., Schütz, F., Goldstein, D. R., Piccart, M., and Delorenzi, M.

(2008). Meta-analysis of gene expression profiles in breast cancer: toward a unified understanding of breast cancer subtyping and prognosis signatures. *Breast Cancer Res.*, 10(4):R65.

Yang, A., Ganesh, A., Sastry, S., and Ma, Y. (2010). Fast l1-minimization algorithms and an application in robust face recognition: a review. In *Proceedings of the International Conference on Image Processing*, pages 1849–1852.

Yang, Y. H., Dudoit, S., Luu, P., Lin, D. M., Peng, V., Ngai, J., and Speed, T. P. (2002). Normalization for cdna microarray data: a robust composite method addressing single and multiple slide systematic variation. *Nucleic Acids Res.*, 30(4).

Yeung, M. K. S., Tegnér, J., and Collins, J. J. (2002). Reverse engineering gene networks using singular value decomposition and robust regression. *Proc. Natl. Acad. Sci. USA*, 99(9):6163–6168.

Yu, J., Sieuwerts, A., Zhang, Y., Martens, J., Smid, M., Klijn, J., Wang, Y., and Foekens, J. (2007). Pathway analysis of gene signatures predicting metastasis of node-negative primary breast cancer. *BMC cancer*, 7(1):182.

Yu, L., Ding, C., and Loscalzo, S. (2008). Stable feature selection via dense feature groups. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 803–811. ACM.

Yu, L. and Liu, H. (2004). Efficient feature selection via analysis of relevance and redundancy. *The Journal of Machine Learning Research*, 5:1205–1224.

Yuan, M. and Lin, Y. (2006). Model selection and estimation in regression with grouped variables. *J. R. Stat. Soc. Ser. B*, 68(1):49–67.

Zhang, Y. and Rajapakse, J. (2008). *Machine learning in bioinformatics*, volume 4. Wiley.

Zhao, P., Rocha, G., and Yu, B. (2009). The composite absolute penalties family for grouped and hierarchical variable selection. *The Annals of Statistics*, 37(6A):3468–3497.

Zhao, P. and Yu, B. (2006). On model selection consistency of lasso. *J. Mach. Learn. Res.*, 7:2541.

Zoppoli, P., Morganella, S., and Ceccarelli, M. (2010). Timedelay-aracne: Reverse engineering of gene networks from time-course data by an information theoretic approach. *Bmc Bioinformatics*, 11(1):154.

Zou, H. and Hastie, T. (2005). Regularization and variable selection via the Elastic Net. *J. R. Stat. Soc. Ser. B*, 67:301–320.

Zucknick, M., Richardson, S., Stronach, E., et al. (2008). Comparing the characteristics of gene expression profiles derived by univariate and multivariate classification methods. *Stat Appl Genet Mol Biol*, 7(1):7.

Sélection de variables à partir de données d'expression.

Signatures moléculaires pour le pronostic du cancer du sein et inférence de réseaux de régulation génique.

Résumé : De considérables développements dans le domaine des biotechnologies ont modifié notre approche de l'analyse de l'expression génique. En particulier, les puces à ADN permettent de mesurer l'expression des gènes à l'échelle du génome, dont l'analyse est confiée au statisticien.

A partir de ces données dites en grande dimension, nous contribuons, dans cette thèse, à l'étude de deux problèmes biologiques. Nous traitons ces questions comme des problèmes d'apprentissage statistique supervisé et, en particulier, de sélection de variables, où il s'agit d'extraire, parmi toutes les variables - gènes - à disposition, celles qui sont nécessaires et suffisantes pour prédire la réponse à une question donnée.

D'une part, nous travaillons à repérer des listes de gènes, connues sous le nom de signatures moléculaires et supposées contenir l'information nécessaire à la prédiction de l'issue du cancer du sein. La prédiction des événements métastatiques est en effet cruciale afin d'évaluer, dès l'apparition de la tumeur primaire, la nécessité d'un traitement par chimio-thérapie adjuvante, connue pour son agressivité. Nous présentons dans cette thèse trois contributions à ce problème. Dans la première, nous proposons une comparaison systématique des méthodes de sélection de variables, en termes de performance prédictive, de stabilité et d'interprétabilité biologique de la solution. Les deux autres contributions portent sur l'application de méthodes dites de parcimonie structurée (graph Lasso et k-support norm) au problème de sélection de signatures. Ces trois travaux discutent également l'impact de l'utilisation de méthodes d'ensemble (bootstrap et ré-échantillonnage).

D'autre part, nous nous intéressons au problème d'inférence de réseau génique, consistant à déterminer la structure des interactions entre facteurs de transcription et gènes cibles. Les premiers sont des protéines ayant la faculté de réguler la transcription des gènes cibles, c'est-à-dire de l'activer ou de la réprimer. Ces régulations peuvent être représentées sous la forme d'un graphe dirigé, où les noeuds symbolisent les gènes et les arêtes leurs interactions. Nous proposons un nouvel algorithme, TIGRESS, classé troisième lors du challenge d'inférence de réseaux DREAM5 en 2010. Basé sur l'algorithme LARS couplé à une stratégie de ré-échantillonnage, TIGRESS traite chaque gène cible séparément, en sélectionnant ses régulateurs, puis assemble ces sous-problèmes pour prédire l'ensemble du réseau.

Enfin, nous consacrons le dernier chapitre à une discussion ayant pour objectif de replacer les travaux de cette thèse dans un contexte bibliographique et épistémologique plus large.

Mots clés : Apprentissage statistique, Sélection de variables, Signatures moléculaires, Réseau de régulation

Feature selection from gene expression data.

Molecular signatures for breast cancer prognosis and gene regulatory network inference.

Abstract: Important developments in biotechnologies have moved the paradigm of gene expression analysis from a hypothesis-driven to a data-driven approach. In particular, DNA microarrays make it possible to measure gene expression on a genome-wide scale, leaving its analysis to statisticians.

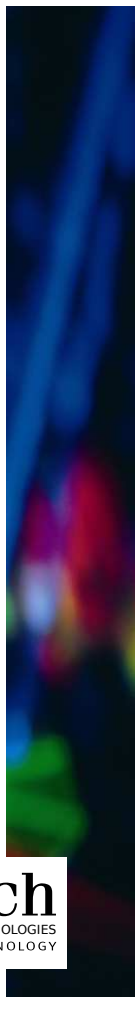
From these high-dimensional data, we contribute, in this thesis, to two biological problems. Both questions are considered from the supervised learning point of view. In particular, we see them as feature selection problems. Feature selection consists in extracting variables - here, genes - that contain relevant and sufficient information to predict the answer to a given question.

First, we are concerned with selecting lists of genes, otherwise known as molecular signatures and assumed to contain the necessary amount of information to predict the outcome of breast cancer. It is indeed crucial to be able to estimate the chances for future metastatic events from the primary tumor, in order to evaluate the relevance of having the patient undergo an aggressive adjuvant chemotherapy. In this thesis, we present three contributions to this problem. First, we propose a systematic comparison of feature selection methods in terms of predictive performance, stability and biological interpretability of the solution they output. The second and third contributions focus on applying so-called structured sparsity methods (here graph Lasso and k-overlap norm) to the signature selection problem. In all three studies, we discuss the impact of using so-called Ensemble methods (bootstrap, resampling).

Second, we are interested in the gene regulatory network inference problem that consists in determining patterns of interaction between transcription factors and target genes. The formers are proteins that regulate the transcription of target genes in that they can either activate or repress it. These regulations can be represented as a directed graph, where nodes symbolize genes and edges depict their interactions. We introduce a new algorithm named TIGRESS, that granted us the third place at the DREAM5 network inference challenge in 2010. Based on the LARS algorithm and a resampling procedure, TIGRESS considers each target gene independently by inferring its regulators and finally assembles individual predictions to provide an estimate of the entire network.

Finally, in the last chapter, we provide a discussion that attempts to place the contributions of this thesis in a broader bibliographical and epistemological context.

Keywords: Machine learning, Feature selection, Molecular signatures, Gene regulatory networks



MINES ParisTech

ParisTech
INSTITUT DES SCIENCES ET TECHNOLOGIES
PARIS INSTITUTE OF TECHNOLOGY