Analyse supervisée multibloc en grande dimension

Hadrien Lorenzo

École doctorale
Sociétés, politique, santé publique

université de BORDEAUX

Inserm

Inria

THÈSE PRÉSENTÉE

POUR OBTENIR LE GRADE DE

DOCTEUR DE L'UNIVERSITÉ DE BORDEAU

École doctorale n° 545 : Sociétés, politique, santé publique

Spécialité doctorale « Santé publique, option Biostatistique »

par **Hadrien LORENZO**

Analyse supervisée multibloc en grande dimension

sous la direction de **Rodolphe THIÉBAUT** et **Jérôme SARACCO**

présentée et soutenue publiquement le mercredi 27 novembre 2019

devant le jury composé de :

Pr.	**Anne RUIZ-GAZEN**	(1)	Présidente du jury
Pr.	**Anne-Laure BOULESTEIX**	(2)	Rapporteure
Pr.	**François HUSSON**	(3)	Rapporteur
MCf, HDR	**Marie CHAVENT**	(4,5)	Examinatrice
Pr.	**Arthur TENENHAUS**	(6)	Examinateur
Pr.	**Jérôme SARACCO**	(4,7)	Codirecteur
Pr.	**Rodolphe THIÉBAUT**	(8)	Directeur

(1) Toulouse School of Economics, Université de Toulouse 1 Capitole, 21 allée de Brienne, 31000 Toulouse, France. (2) Ludwig-Maximilians-University Munich, Allemagne. (3) IRMAR, Unité de Mathématiques Appliquées, Agrocampus Ouest, Rennes. (4) INRIA Bordeaux Sud-Ouest, Équipe CQFD, 33400, Talence, France. (5) Université de Bordeaux, IMB, UMR 5251, 33400, Talence, France. (6) L2S, UMR CNRS 8506,CNRS–CentraleSupélec–Universite Paris-Sud, Université Paris-Saclay, 3 rue Joliot-Curie, 91192 Gif-sur-Yvette, France. (7) ENSC - Bordeaux INP, IMB, UMR 5251, 33400, Talence, France. (8) INSERM, ISPED, centre INSERM U-897-Épidémiologie-Biostatistique, Bordeaux, 33000, FRANCE.

REMERCIEMENTS

L'encadrement facilite la lecture et accompagne le travail, qu'importe la qualité de ce dernier. C'est à la fois une extension et une transition. En effet, il ouvre intellectuellement vers un cadre plus général mais le limite aussi physiquement. Cette limitation devient elle-même constructive lorsqu'elle donne à apprécier le point de vue des encadrants, experts de l'art dans lequel le néophyte manque de se noyer et ceci à plusieurs reprises. C'est avec fierté et honneur que je remercie Rodolphe Thiébaut et Jérôme Saracco d'avoir assumé et assuré l'encadrement de cette thèse, de lui avoir imaginé cette coloration interdisciplinaire et d'en avoir permis sa réalisation. Cette thèse a aussi été ma thèse et je les remercie d'avoir pris de leur temps et de leur énergie pour moi. Pour que ce ne soit pas uniquement un résultat universitaire mais aussi un achèvement humain associé à une évolution personnelle. Je ressens ceci et je ne ferai pas le détail des moments passés, des thèmes abordés ou des conclusions imaginées car leur implication va plus loin que l'observation de quelques réalisations. C'est leur processus entier que je remercie, déterministiquement.

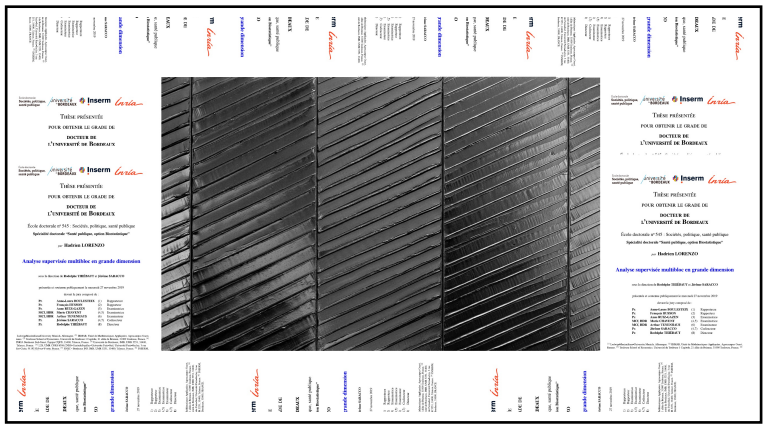
Je souhaite aussi remercier Madame Anne-Laure Boulesteix et Monsieur François Husson d'avoir accepté de rapporter les résultats compilés dans le présent manuscrit. Merci d'avoir pris de leur précieux temps à la relecture de ce travail et d'en avoir rédigé des commentaires précis, judicieux et constructifs. Je tiens aussi à remercier l'ensemble des membres de ce jury à commencer par Mesdames Anne Ruiz-Gazen et Marie Chavent. Je tiens à remercier Monsieur Arthur Tenenhaus qui fut l'initiateur de mon rapprochement avec l'ISPED et de l'équipe dirigée par Rodolphe Thiébaut et qui est aujourd'hui présent dans ce jury. Il a énormément contribué à mon ouverture scientifique bien sûr, mais aussi à mon bien-être alors que j'étais étudiant dans l'école, aujourd'hui disparue, Supélec. Je me permets aussi de remercier ici Monsieur Laurent Le Brusquet pour son aide dans mes débuts de jeune chercheur. Arthur Tenenhaus a aussi été, au côté de Madame Hélène Jacqmin-Gadda, membre de mon comité de suivi de mi-thèse. Je les en remercie tous deux chaleureusement.

La patience et l'attention déployés par Madame Misbah Razzaq et Monsieur David Alexandre-Trégouët nous ont permis de produire des résultats d'intérêt pour les experts, merci d'avoir cru en cette méthode, alors balbutiante et de continuer à la promouvoir. Merci aussi à Madame Marie-Pierre Ellies d'avoir participé à cette évolution et avec qui je fus très heureux de travailler, à quand le prochain projet ?

Je tiens aussi à remercier Madame Laura Richert, qui fut impressionnante de pragmatisme, de pédagogie et de patience devant mes nombreuses erreurs et imprécisions. Aussi Monsieur Boris Hejblum avec qui j'eus la joie de faire mes armes sur un jeu de données qui me poursuit encore aujourd'hui. Plus largement merci à tous les membres de l'équipe SISTM et de l'équipe Biostat pour leurs accompagnements, dans le travail, sur des skis, agrippé à des prises ou à un comptoir. Au bureau John Snow je dirais aussi merci, à ceux qui sont partis et à ceux qui sont toujours là et avec qui j'ai pu partager des moments d'exception au travail mais aussi et surtout en dehors, Laura, Solenne, Myriam, Irène, Alexandra, Chloé, Perrine, Henri, Loïc, Jean-Noël, Aaron, Bruno, Marie, Virginie mais aussi à Robin, Anaïs, Corentin, Camille, Morgane et à ceux que je ne connais pas encore assez. Merci bien entendu à Madame Sandrine Darmigny, pour tout !

À tous ceux qui m'ont aidé dans la rédaction de ce manuscrit, à commencer par mes parents qui battirent la campagne à plusieurs reprises pour en imprimer les différentes versions et à mon père qui, je le remercie, ne s'étonna pas trop de voir tant de barbarismes dans une de mes productions. Merci encore à vous deux, j'ai vos goûts et vos instructions, je m'en inspire et j'avance. Merci aussi à Bastien, Guillaume et Rémi pour leurs nombreuses relectures de manuscrit, actes volontaires et appréciés ! Merci à Valentin qui relut la première version de l'article de méthode (*Si si, il l'a fait...*) ainsi qu'à Simon, Sylvain et Arnaud pour ces débats scientifiques. Avant tout ce sont mes amis. Si ce ne sont pas que les sciences, je veux aussi remercier la chaleur de Laurianne, Françoise, Emma, Olivier, Édouard, Costanza, Nico, Victor, Marianne, Guillaume, Marie, Audrey, François et de tous ces gens qui ont fait de mon cadre de vie un épanouissement. Merci encore à Denys et Oulfa, pour ces balades, ces soutenances de physiques fondamentale et cycliste et cet amour.

Enfin je voudrais remercier Anna, un fondamental dans ma vie, une montagne de patience et d'amour. Merci pour tout ce que tu me donnes et que je m'efforce de te renvoyer malgré mes maladresses. Et alors que nous étions deux en arrivant à Bordeaux, nous sommes aujourd'hui trois. Mais ceci relève d'une tout autre affaire...



Tentative manquée d'encadrement d'une œuvre de l'artiste Pierre Soulages.

COLLABORATIONS, TRAVAUX ET ENSEIGNEMENTS

Collaborations

— International AIDS Vaccine Initiative, Londres, Royaume-Uni : Jonathan Hare.

— University Medical Center Hamburg-Eppendorf, 1st Department of Medicine, 20246, Hambourg, Allemagne : Anne Rechtien.

— Equipe SISTM, INSERM U1219, INRIA BSO, Bordeaux, France : Laura Richert, Solenne Delahaye, Henri Bonnabau et Boris Hejblum.

— Equipe Vintage, INSERM U1219, Bordeaux, France : Misbah Razzaq, Felipe Leal Valentim et David-Alexandre Trégouët.

— Equipe Biomarqueurs, INRA, Bordeaux, France : Marie-Pierre Ellies.

Articles publiés

— Rechtien, A.*, L. Richert*, H. Lorenzo, ..., R. Thiébaut, M. Altfled, M. Addo. 2017, «Systems Vaccinology Identifies an Early Innate Immune Signature as a Correlate of Antibody Responses to the Ebola Vaccine rVSV-ZEBOV», *Cell reports*, vol. 20, n^{o}9, p. 2251–2261.,

— Lorenzo, H., R. Misbah, J. Odeber, P.-E. Morange, J. Saracco, D.-A. Trégoüet et R. Thiébaut. 2019, «High-dimensional multi-block analysis of factors associated with thrombin generation potential», *2019 IEEE 32nd International Symposium on Computer-Based Medical Systems (CBMS)*, IEEE, p. 453–458,

— Lorenzo, H.*, Ellies-Oury, M.-P.* , C. Denoyelle, J. Saracco et B. Picard. 2019, «An original methodology for the selection of biomarkers of tenderness in five different muscles», *Foods*, vol. 8, n^{o}6, p. 206,

Articles soumis

— Lorenzo, H., J. Saracco et R. Thiébaut. 2019, «Supervised learning for multi-block incomplete data», *arXiv preprint arXiv :1901.04380*,

*. Même contribution des auteurs.

— Lorenzo, H., J. Saracco et R. Thiébaut. 2019, «ddsPLS : A Package to Deal with Multiblock Supervised Problems with Missing Samples in High Dimension»,

Communications orales dans un congrès international

— Lorenzo, H., R. Thiébaut et J. Saracco. 2018, «Multi-block high-dimensional lasso penalized analysis with imputation of missing data applied to postgenomic data in an Ebola vaccine trial», *Annual workshop on Statistical Methods for Post Genomic Data - SMPGD 2018*, Montpellier, France.

— Lorenzo, H., J. Saracco et R. Thiébaut. 2019, «High-Dimensional Multi-Block Analysis of Factors Associated with Thrombin Generation Potential», *32nd IEEE CBMS International Symposium on Computer-Based Medical Systems – CBMS2019*, Cordoue, Espagne.

Communications orales dans un congrès national

— Lorenzo, H., J. Saracco et R. Thiébaut. 2018, «Une PLS multivoie parcimonieuse avec observations manquantes pour données hétérogènes», *Journées de Statistique 2018 - Société Française de Statistique (SFdS)*, EDF Lab Paris Saclay, France.

Communications affichées

— Lorenzo, H., J. Saracco et R. Thiébaut. 2019, «Apprentissage supervisé pour données massives multi-blocs incomplètes», *JED 2019 - EDSP2*, Bordeaux, France.

— Hare, J., C. Streatfield, J. Yates, W. Kilembe, S. Lakhi, H. Lorenzo, ... et J. Gilmour. 2016. «CD4 T-cell Counts More Closely Associate with the Cytokine Profiles Observed in Acutely Infected Volunteers than Viral Load or Replicative Capacity», *AIDS research and human retroviruses* (Vol. 32, pp. 170-170). 140 Huguenot street, 3rd fl, New Rochelle, NY 10801 USA : Mary Ann Liebert, Inc.

Séminaires

— Séminaires de biostatistique, BPH U1219, Bordeaux, France.

— Séminaires de l'équipe SISTM, BPH U1219, Bordeaux, France.

— Séminaires des doctorants de l'ISPED, BPH U1219, Bordeaux, France.

Enseignements

— M2 - Biostatistiques, ISPED, Bordeaux, France :
  — 2017-2018 : Cours et travaux dirigés : 26h.
  — 2018-2019 : Cours et travaux dirigés : 6h.
— Double cursus Ecole Santé-Sciences, équivalent M1, ISPED, Bordeaux, France :
  — 2018-2019 : Cours et travaux dirigés : 24h.
— 3A ingénieur - Biostatistiques, ENSAI, Rennes (Bruz), France :
  — 2018-2019 : Projet de génétique : 21h.
— École d'été de l'ISPED, Bordeaux, France :
  — 2017-2018 : Cours et travaux dirigés : 6h.
— DU BDSI (Big data et statistique pour l'ingénieur), ENSC, Bordeaux, France :
  — 2018-2019 : Cours et travaux dirigés : 11h.

Encadrements

— Valentin Kochegarov, stage de 3 mois, mai à juillet 2019.
« *Etude des méthodes de l'analyse de l'expression différentielle des données de sequençage d'ARN à haut débit.* »

Développements logiciels

— Package *data-driven sparse PLS* (ddsPLS) :
  — Version R stabilisée sur le CRAN et la vignette associée.
  — Version R en développement sur GitHub via

    `https://github.com/hlorenzo/ddsPLS`.

  — Version Python en développement sur GitHub via

    `https://github.com/hlorenzo/py_ddspls`.

— Site informatique personnel (R blogdown) via

`https://hadrienlorenzo.netlify.com/`.

TABLE DES MATIÈRES

RÉSUMÉ EN FRANÇAIS

L'apprentissage statistique consiste à apprendre à partir de données mesurées dans un échantillon d'individus et cherche à prédire la grandeur d'intérêt chez un nouvel individu. Dans le cas de la vaccination, ou dans d'autres cas dont certains présentés dans ce manuscrit, le nombre de variables mesurées dépasse le nombre d'individus observés, c'est un cas dégénéré d'analyse statistique qui nécessite l'utilisation de méthodes spécifiques. Les propriétés des algorithmes de régularisation permettent de gérer ces cas. Il en existe plusieurs types en fonction de la structure des données considérées et du problème qui sont étudiés.

Dans le cas de ce travail, l'objectif principal a été d'utiliser l'information disponible à l'issue de décompositions en éléments propres des matrices de covariances transformées via un opérateur de seuillage doux. Cette solution est particulièrement peu coûteuse en termes de temps de calcul et permet la sélection des variables d'intérêt.

Nous nous sommes centrés sur les données qualifiées d'hétérogènes, c'est à dire issues de jeux de données qui sont provenant de sources ou de technologies distinctes. On parle aussi de données multiblocs. Les coûts d'utilisation de certaines technologies pouvant être prohibitifs, il est souvent choisi de ne pas acquérir certaines données sur l'ensemble d'un échantillon, mais seulement sur un sous-échantillon d'étude. Dans ce cas, le jeu de données se retrouve amputé d'une partie non négligeable de l'information. La structure des données associée à ces défauts d'acquisition induit une répartition elle-même multibloc de ces données manquantes, on parle alors de données manquantes par blocs. Le second objectif de notre méthode est de gérer ces données manquantes par blocs en s'appuyant sur l'information à prédire, ceci dans le but de créer un modèle prédictif qui puisse gérer les données manquantes aussi bien pour les données d'entraînement que pour celles de test. Cette méthode emprunte au seuillage doux afin de sélectionner les variables d'intérêt et ne nécessite que deux paramètres à régler qui sont le nombre de composantes et le nombre de variables à sélectionner parmi les covariables. Ce paramétrage est classiquement réalisé par validation croisée.

La méthode développée a fait l'objet de simulations la comparant aux principales méthodes existantes. Elle montre d'excellents résultats en prédiction et en termes de temps de calcul. Elle a aussi été appliquée à plusieurs jeux de données.

RÉSUMÉ EN ANGLAIS

Statistical learning objective is to learn from observed data in order to predict the response for a new sample. In the context of vaccination, the number of features is higher than the number of individuals. This is a degenerate case of statistical analysis which needs specific tools. The regularization algorithms can deal with those drawbacks. Different types of regularization methods can be used which depends on the data set structure but also upon the question.

In this work, the main objective was to use the available information with soft-thresholded empirical covariance matrix estimations through SVD decompositions. This solution is particularly efficient in terms of variable selection and computation time.

Heterogeneous typed data sets (coming from different sources and also called multiblock data) were at the core of our methodology. Since some data set generations are expensive, it is common to down sample the population acquiring some types of data. This leads to multi-block missing data patterns. The second objective of our methodology is to deal with those missing values using the response values. But the response values are not present in the test data sets and so we have designed a methodology which permits to consider both the cases of missing values in the train or in the test data sets.

Thanks to soft-thresholding, our methodology can regularize and select features. This estimator needs only two parameters to be fixed which are the number of components and the maximum number of features to be selected. The corresponding tuning is performed by cross-validation.

According to simulations, the proposed method shows very good results comparing to benchmark methods, especially in terms of prediction and computation time. This method has also been applied to several real data sets associated with vaccine, thomboembolic and food researches.

AVANT-PROPOS ET NOTATIONS

L'apprentissage statistique des données a émergé de façon prodigieuse très récemment, il prend naissance dans les années 60 avec les travaux de l'école russe et notamment de Vapnik. L'apprentissage statistique acquiert alors des justifications théoriques. À cette période s'opère un changement dans les méthodes statistiques puisque l'objectif de généraliser les modèles statistiques à des échantillons de taille finie prend forme. Des risques empiriques sont développés qui ambitionnent de rendre compte de l'erreur de *surapprentissage* en se basant sur les informations du jeu de donnée d'entraînement, formé d'un nombre fini n d'individus. Celles-ci étant intégrées dans une modélisation du processus observé de complexité k, où k est très souvent associé au nombre de paramètres à estimer dans le modèle. Cette période très prolifique en terme de production méthodologique verra l'essor de la *régularisation*.

La *régularisation* apparaît afin de résoudre le cas des problèmes sous-déterminés (*i.e.* $n << p$ où p est le nombre de variables explicatives). L'exemple classique est celui de la résolution du problème des moindres carrés ordinaires (MCO) qui est formulable via le problème d'optimisation suivant

$$\min_{\mathbf{b}\in\mathbb{R}^p} ||\mathbf{y} - \mathbf{X}\mathbf{b}||_2^2, \tag{1}$$

où $\| \bullet \|_2$ symbolise l'opérateur de norme euclidienne $\mathbf{y}$ et $\mathbf{X}$ sont respectivement un vecteur de longueur n et une matrice $n \times p$, décrivant n réalisations indépendantes. $\mathbf{X}$ est décrite par p variables telles que pour chacune des réalisations $i \in [\![1, n]\!]$, $y_i = X_i^T\mathbf{b} + \epsilon_i$ où ϵ_i est une erreur d'observation. La résolution de ce problème, dans le cas où le rang de la matrice $\mathbf{X}$ vaut n conduit à $\hat{\mathbf{b}}^{(MCO)} = \left(\mathbf{X^T X}\right)^{-1}\mathbf{X^T y}$. Mais lorsque le problème n'a pas de solution exacte, si $n < p$ notamment ce qui rend la matrice $X^T X$ non inversible, c'est ce que l'on appelle la *grande dimension*.

Afin de résoudre ce problème qualifié de *dégénéré*, une méthode a été introduite (connue aujourd'hui sous le nom de régularisation de Tikhonov) et dont la mise en pratique la plus simple est la régularisation Ridge. Cette régularisation propose une solution approchée du problème (1) au moyen d'un paramètre appelé *paramètre de régularisation* et noté ici λ. La solution approchée devient alors

$$\hat{\mathbf{b}}^{(Ridge)}(\lambda) = \left(\mathbf{X}^T\mathbf{X} + \lambda\mathbb{I}_p\right)^{-1}\mathbf{X}^T\mathbf{y},$$

où $\mathbb{I}_p$ est la matrice identité $p \times p$. Le coefficient $\lambda > 0$ a plusieurs interprétations mais c'est avant tout la valeur ajoutée à chacune des valeurs propres de la matrice $\mathbf{X}^T\mathbf{X}$ qui a pour effet de rendre la matrice $\mathbf{X}^T\mathbf{X} + \lambda\mathbb{I}_p$ inversible.

De nombreuses autres régularisations ont été développées sur le schéma d'un problème d'optimisation qui est la somme de deux termes. Le premier représente l'*attache aux données*,

$||\mathbf{y} - \mathbf{Xb}||_2^2$, alors que le second, $\lambda\,||\mathbf{b}||_2^2$ dans le cas de la régularisation Ridge, est souvent associé à la *complexité* du modèle.

En plus de la régularisation, la sélection de variables peut être recherchée et les méthodes Lasso permettent de répondre à cet objectif. La résolution de ce genre de problèmes passe par l'utilisation de *fonctions de seuillages*, fonctions qui seuillent leurs arguments en fonction d'un ou plusieurs paramètres associés.

Les cas de réponses multivariées ($\mathbf{y}$ prend alors la forme d'une matrice $\mathbf{Y}$ à n lignes et q colonnes) impliquent de devoir modifier les problèmes d'optimisation. La méthode des moindres carrés partiels (PLS) permet de gérer les données présentant une matrice objectif $\mathbf{Y}$ multidimensionnelle. Au moyen de variables latentes, comme le fait l'analyse en composantes principales (*ACP*), cette méthode s'affranchit des contraintes de grande dimension tout en construisant des modèles de prédiction. L'utilisation de contraintes spécifiques a permis de gérer la sélection de variables pour le modèle PLS.

Un niveau encore au-dessus en terme de complexité structurelle nous amène à considérer les données dont les variables peuvent être naturellement groupées. On parle alors de données multiblocs. Un cas particulier intéressant survient lorsque les blocs sont décrits par différentes modalités d'un même groupe de variables, on appelle ceci les données multivoies et donc multiblocs. Les données prises à différents temps forment différentes modalités par exemple. Les données de séquençage de l'ARN à différents temps après vaccination contre le virus Ebola sont des données qui forment un jeu de données multiblocs. Cette remarque renvoie aux données rVSV Ebola-ZEBOV analysées pendant cette thèse et qui associent à cette structure complexe approximativement 30% de données manquantes pour seulement une vingtaine d'individus, l'objectif étant alors de prédire la réponse anticorps ($\mathbf{Y}$) formée de $q = 5$ variables quantitatives. Ces données manquantes sont structurées en blocs coïncidant avec la structure intuitive des données, due aux technologies employées ainsi qu'aux différents échantillonnages. La figure 1 détaille ce genre de données où les données manquantes sont représentées par des lignes rouges, où les $(p_t)_{t=1\ldots T}$ sont les dimensions des T blocs $(\mathbf{X}_t)_{t=1\ldots T}$. Ces données manquantes sont dites manquantes par blocs.

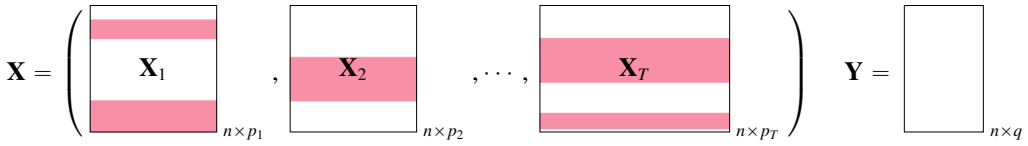


FIGURE 1 – Structure de données multiblocs supervisées avec données manquantes par blocs, symbolisées par les zones colorées

Le faible échantillonnage de l'étude implique de ne pouvoir construire qu'un nombre réduit de composantes. Or les types de données concernées, notamment des données tanscriptomiques, s'expriment au moyen d'une large quantité de variables, où les variables prépondérantes ne sont pas généralement associées au problème considéré. Dans ce contexte il devient impératif de ne construire qu'un nombre très restreint de composantes associées à $\mathbf{Y}$.

Le problème principal de cette thèse s'est naturellement dessiné autour du dimensionnement d'une méthode permettant la gestion de données manquantes par blocs en prenant en compte une matrice objectif $\mathbf{Y}$. On appelle ceci l'imputation supervisée par blocs.

Les méthodes envisagées et les recherches effectuées ont permis de produire une méthode capable de gérer un problème supervisé en grande dimension avec sélection de variables, tenant compte de données manquantes et d'une structure multibloc. Seulement 2 paramètres doivent être fixés dans cette méthode. Le package *ddsPLS* (*), rédigé en R, est disponible sur le CRAN et implémente des fonctions de visualisation associés aux modèles de prédiction. Ce package met à disposition une fonction de validation croisée permettant d'optimiser les deux paramètres de ce modèle, et les méthodes de visualisation associées. Une vignette d'introduction à la méthode est aussi présente sur la page CRAN du package.

Notations supplémentaires

Dans la suite nous choisirons la convention de symboliser les éléments de $\mathbb{R}$ par des lettres en italiques, les vecteurs, ligne ou colonnes, par des lettres en italiques gras et les matrices par des lettres majuscules en gras. Les listes de matrices, ou les ensembles plus complexes, seront le plus souvent symbolisées par des lettres stylisées se rapportant aux éléments qu'elles contiennent.

Organisation de la thèse

Le présent manuscrit est composé de 5 chapitres suivis d'une Annexe elle-même divisée en quatre sections. Le chapitre 1 reprend les différents éléments associés à la compréhension des problèmes régularisés en grande dimension avec des contraintes de sélection de variables et/ou structurelles, du type multibloc. Ceci s'associe à une étude des méthodes d'imputation dans le cadre d'algorithmes d'espérance-maximisation (*EM*) (†) dans le cas de données mono/multiblocs, supervisées ou non supervisées.

Le chapitre 2 présente le fonctionnement de la méthode ddsPLS pour la gestion de problèmes supervisés multiblocs avec données manquantes par échantillons et contraintes de sélection de variables. Cette méthode s'appuie sur un article soumis (Lorenzo et collab., 2019b) dans lequel des simulations numériques ont été effectuées ainsi qu'une application sur des données réelles. Une section complémentaire permet d'ajouter des résultats théoriques et de simulations sur l'algorithme considéré dans le contexte d'une absence de données manquantes. Cette partie est entièrement rédigée en anglais par soucis de cohérence avec l'article qui y est associé.

Le chapitre 3 propose un descriptif du package ddsPLS, rédigé en R et disponible sur le CRAN, développé pendant cette thèse et qui permet d'utiliser la méthode éponyme.

Le chapitre 4 se concentre sur deux nouvelles applications de la méthode qui ont donné lieu à des valorisations au travers d'articles. Le premier jeu de données traite de la prédiction

(*). https://CRAN.R-project.org/package=ddsPLS

(†). On dit que ces méthodes sont « typées-EM » si la démonstration mathématique du caractère EM de cette méthode n'est pas démontrée.

de trois métriques d'une dynamique protéique pour plusieurs centaines de participants atteints de thrombose veineuse et décrits par des données hétérogènes. Les données sont structurées en blocs de différentes natures. La proportion de données manquantes dans le bloc contenant le plus de variables est de pratiquement 70% et pose un challenge d'un point de vue méthodologique. Le deuxième jeu de données décrit 10 vaches au travers d'une vingtaine de marqueurs protéiques sur 5 morceaux de viandes différents. On repère alors une structure multivoie à 5 modalités pour 10 individus et 20 variables. L'étude de ces deux jeux de données a permis de modifier l'algorithme de façon à prendre en compte plusieurs aspects, comme l'initialisation des données manquantes et la visualisation des résultats pour aider l'interprétation.

Le chapitre 5, permet de conclure sur le travail effectué dans le cadre de cette thèse et d'ouvrir de nouvelles pistes de recherche.

L'annexe décrit des outils importants utilisés pendant ce travail de thèse ainsi que des exemples ou remarques argumentées.

CHAPITRE 1

INTRODUCTION

> *Why do we need a theory of consistency if our goal is to construct algorithms for a small (finite) sample size ?*
>
> Vladimir Naumovitch Vapnik, An Overview of Statistical Learning Theory, 1999

Sommaire

Un jeu de données d'entraînement, noté $\mathcal{D}_n$ dans la suite de ce manuscrit, est formé de n réalisations indépendantes du couple (X, Y) où X de dimension p correspond aux covariables et Y de dimension q correspond aux variables réponse. Les couples $(X_i, Y_i)_{i\in[\![1,n]\!]}$ sont supposés suivre une même loi $\mathbb{P}$ inconnue. L'objectif de l'apprentissage statistique est de fournir une estimation d'un modèle permettant de prédire les valeurs de Y pour des échantillons de test indépendants de $\mathcal{D}_n$ en ayant estimé un modèle de prédiction sur le jeu de données d'entraînement $\mathcal{D}_n$.

La première étape de ce type d'analyse est de décrire un modèle statistique des données. Par exemple le modèle de la régression à bruit ϵ centré de variance σ_ϵ^2 peut s'écrire

$$Y = f(X) + \epsilon,$$

et sera considéré dans la totalité de ce manuscrit en supposant f déterministe.

Afin de pouvoir estimer la fonction f inconnue, il faut déterminer un estimateur de cette fonction qui sera entraîné sur les données $\mathcal{D}_n$, on le note $\hat{f}_n$. Afin de sonder un maximum de fonctions, l'analyste propose souvent une famille de M estimateurs possibles, que l'on peut représenter par $\left(\hat{f}_n^{(m)}\right)_{m\in[\![1,M]\!]}$. Il en existe de différentes sortes en fonction de la structure des données, des capacités de calcul et des désirs d'interprétabilité émis par les experts du domaine. On détaillera dans ce manuscrit différents estimateurs pouvant s'appliquer dans le contexte très particulier où $n < p$.

Une fois les différents estimateurs évalués, il faut choisir le plus efficace. Cette efficacité est mesurée par un risque d'erreur empirique, noté $\widehat{\mathcal{R}}_n$, et permet la validation d'un estimateur en le comparant aux autres. On note alors l'estimateur optimal $\hat{f}_{opt} = \arg\max_{m\in[\![1,M]\!]} \widehat{\mathcal{R}}_n\left(\hat{f}_n^{(m)}\right)$. Cette étape est dépendante de l'utilisation d'un échantillon de test sur lequel mesurer cette erreur. Ces deux facettes sont aussi largement décrites dans ce manuscrit et contextualisées au cas $n < p$.

La Figure 2 reprend donc le principe de l'apprentissage statistique qui est d'intégrer cinq classes d'objets que sont les données d'entraînement, les modèles, les estimateurs, les critères de validation et les données de test, de façon à obtenir un estimateur $\hat{f}_{opt}$ satisfaisant aux exigences des experts.

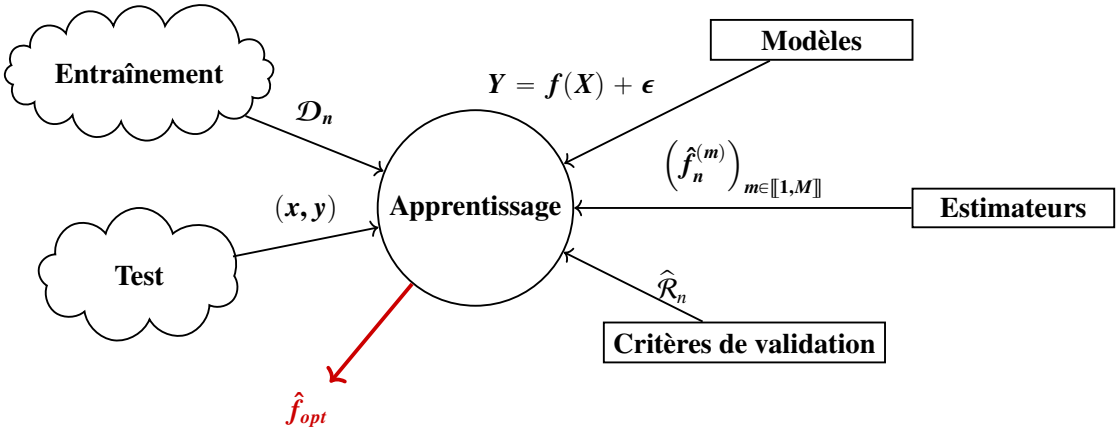


FIGURE 2 – Schéma de principe de l'apprentissage statistique où les nuages représentent les sources aléatoires d'information, les rectangles les informations fournies par l'analyste et le cercle le processus d'apprentissage mis en place par l'analyste, ceci afin de produire l'estimateur optimal $\hat{f}_{opt}$.

Dans les sections qui suivent nous allons détailler les principes associés aux critères de validation, aux estimateurs et aux modèles, qui prennent en compte l'aléa des sources de données, ceci en offrant quelques exemples permettant de clarifier l'énoncé. Peu de démonstrations sont fournies, dans le cas contraire des renvois sont faits vers des annexes ou des articles. Un cas d'étude bien connu permet de clore cette section en proposant un schéma d'apprentissage suivant le genre de schéma décrit par la Figure 2.

La fin de cette partie se concentre sur les principes de données manquantes et de données structurées en blocs afin de détailler les travaux réalisés autour des données manquantes dans le contexte supervisé pour données multiblocs.

1.1 Repères historiques

Dans la première moitié du siècle dernier, la science des données s'est concentrée sur les méthodes d'inférence où l'objectif était de tester des hypothèses statistiques sur des jeux de données de faibles dimensions, soit de l'ordre $(n \propto 10, p < 10)$. Les méthodes les plus utilisées sont alors les méthodes linéaires. Les générations de données sont alors planifiées et l'idée de les réutiliser dans un contexte exploratoire ne se fait qu'aux alentours des années 70.

L'ordinateur rend possible le stockage mais aussi la génération de nouvelles données. Les technologies s'affinant, les bases de données s'enrichissent et se complexifient. Il n'est plus alors question de cantonner l'analyse des données recueillies aux questions de recherche qui sont à leur origine, la dimension p augmente dans les jeux de données et aussi la richesse de l'information qu'ils contiennent. C'est la naissance de l'apprentissage statistique. La méthode ACP (Hotelling, 1933) et ses équivalents non linéaires tels que l'ACP à noyaux (*kernelPCA*) (voir par exemple Scholkopf et Smola, 2001) offrent des possibilités basées sur des résultats mathématiques théoriques qui sont très appréciés. Le début des années 2000 est un moment paradoxale d'un point de vue des données. En effet, les p des données grandissent du fait de l'amélioration des technologies pourtant le n n'augmente pas en conséquence, laissant les analystes avec des problèmes insolvables du fait de ce déséquilibre trop marqué entre le n et le p. Vapnik (1999b) pose alors la question :

Why do we need a theory of consistency if our goal is to construct algorithms for a small (finite) sample size ?

La réponse de ce dernier est d'affirmer que les développements théoriques sont une nécessité à la généralisation des résultats. La théorie qu'il développe introduit le principe de régularisation qui défend l'idée qu'un estimateur dont seulement l'erreur d'approximation cherche à être minimisée ne peut conduire qu'à des estimateurs à très fortes variances et donc à haut risque d'erreur. C'est le surrapprentissage du modèle, ce qui équivaut à apprendre des spécificités de certains individus exotiques de l'échantillon d'entraînement et non pas des caractéristiques globales partagées par tous ces individus. La régularisation passe par la recherche de ce compromis qui permet de lisser suffisamment les estimateurs pour ne conserver que l'information commune. On parle de compromis entre le *biais* et la *variance*, ce qui sera explicité plus loin. Les méthodes telles que Ridge (Hoerl et Kennard, 1970) ou Lasso (Tibshirani, 1996) semblent être tout indiquées dans ce contexte.

Ces méthodes font intervenir des paramètres qui ne sont pas directement interprétés par l'utilisateur dans le modèle de prédiction mais qui doivent être fixés, ce sont les hyperparamètres. Le principe du risque fonctionnel de Vapnik (1992) puis ceux de validation croisée Geisser (1975) sont utilisés dans cet objectif.

C'est dans ce contexte que se place le travail rapporté dans ce manuscrit. D'autres approches tiennent aujourd'hui compte d'une modification profonde de l'acquisition et de la gestion des données. En effet, les bases de données sont mises à jour de façon continue et l'optimisation des modèles doit prendre en compte cette dynamique. Il faut donc inclure cette information fraîche sans pénaliser trop fortement l'information déjà acquise. C'est un autre cas de *grande dimension*.

La prochaine section part des travaux de Vapnik afin d'introduire les outils utilisés dans le reste du manuscrit.

1.2 Risque et compromis

Pour une classe de fonctions $f \in \mathcal{F}$, Vapnik (1992) a décrit ce que l'on appelle le risque fonctionnel. Supposons (X, Y) un couple de deux vecteurs aléatoires réels avec (x, y) une réalisation quelconque de loi jointe $\mathbb{P}(x, y)$ inconnue, alors on définit

$$\text{le risque fonctionnel par } \mathcal{R}(f) = \mathbb{E}\left(L\left(y, f(x)\right)\right) = \int_{\mathcal{X}\times\mathcal{Y}} L\left(y, f(x)\right) \mathbb{P}(x, y) dxdy, \quad (1.1)$$

où le choix de la fonction de perte L est laissé à l'analyste. On peut citer par exemple la perte quadratique $L : (y, \hat{y}) \rightarrow ||y - \hat{y}||_2^2$ où $. \rightarrow ||.||_2$ est la norme euclidienne sur $\mathcal{Y}$ qui est utilisée en régression. La *hinge loss*, $L : (y, \hat{y}) \rightarrow max(0, 1 - y\hat{y})$, pour les problèmes de classification binaire, $y \in \{-1, 1\}$ et qui est utilisée dans le contexte de la classification. Puisque $\mathbb{P}(x, y)$ est inconnue, il a été introduit

$$\text{le risque empirique noté } \hat{\mathcal{R}}_n(f) = \mathbb{E}_n\left[L\left(y, f(x)\right) | \{x, y\} \in \mathcal{D}_n\right] = \frac{1}{n}\sum_{i=1}^{n} L\left(y_i, f(x_i)\right), \quad (1.2)$$

estimé sur l'échantillon d'apprentissage $\mathcal{D}_n$.

Parmi ses résultats importants, Vapnik a montré l'intérêt du principe de minimisation du risque empirique (principe ERM) qui postule que pour tout échantillon $\mathcal{D}_n$, la fonction $\hat{f}_n$ qui minimise l'équation (1.2), est une bonne approximation de f, fonction qui minimise l'équation (1.1). Ceci moyennant une majoration qui va dépendre, de façon croissante de la complexité de l'espace des fonctions et de façon décroissante du nombre d'individus au travers de la fonction ϕ, ce qui est formulé grâce à l'équation

$$\mathcal{R}(f) < \hat{\mathcal{R}}_n(f) + \phi\left(\sqrt{\frac{h}{n}}, \eta\right), \quad (1.3)$$

où h se rapporte à la complexité de la fonction f et η la probabilité associée à ce risque empirique (Vapnik et Chervonenkis, 2015).

1.2.1 Minimisation du Risque Structurel

L'étude du second terme de la partie de droite de l'équation (1.3) a conduit l'auteur à définir une méthode de recherche de solution nommée minimisation du risque structurel (SRM). Cette méthode nécessite que l'utilisateur définisse des ensembles de fonctions $H_1, \cdots, H_M$ (pour un $M \in \mathbb{N}^\star$ fixé par l'utilisateur) tel que $H_1 \subset \cdots \subset H_M$. Dans chaque ensemble l'utilisateur doit trouver la fonction solution de la minimisation de l'équation (1.2) et ensuite évaluer le *risque structurel*, $\phi\left(\sqrt{\frac{h}{n}}, \eta\right)$, associé à cette fonction pour ce jeu de données d'entraînement. La solution finalement choisie est celle qui minimise le *risque structurel* parmi les M solutions minimisant le *risque empirique* dans chaque espace H_m, $\forall m \in [\![1, M]\!]$.

Cette méthode, théoriquement robuste, s'avère difficile à mettre en œuvre. En effet, selon Cornuéjols et Miclet (2010, pp.712-713), la première étape de définition des espaces H_k emboîtés demande une connaissance *a priori* sur le domaine et donc un biais subjectif. De plus, comme l'avoue aussi l'auteur (voir Vapnik, 1999a, pp.99) dans le cas de la grande dimension, la convergence des fonctions cibles peut être très lente car le degrés de régularité des ensembles de fonctions considérés devient lui-même important. En prenant le cas des fonctions polynomiales, l'auteur démontre que l'augmentation du nombre de dimensions induit une divergence du nombre de paramètres, ce qu'il appelle *Curse of Dimensionality*, *Fléau de la dimensionnalité* en français, voir Vapnik (1998, Theorème 6.8, pp.251–252). Dans ce dernier cas l'auteur propose de passer par des approximations locales de fonctions, adaptant le SRM au travers d'une méthode nommée *Structural Risk Minimization Principle for Local Function Estimation* (*Minimisation du Risque Structurel Local pour l'Estimation Local de Fonctions*), en admettant que, localement, toute fonction puisse être approchée par des polynômes de degrés 0, 1, 2 ou 3 et en rappelant aussi que la théorie classique des approximations locales passe par des fonctions constantes, voir Vapnik (1998, pp.268–269).

Bien que le SRM semble être abandonné de nos jours pour les raisons qui viennent d'être citées, le principe de pénalisation du risque empirique pour recherche d'un optimum a été conservé. Les méthodes de régularisation qui seront abordées dans ce manuscrit sont des descendants de cette école. De plus, l'équation (1.3) est très utilisée de nos jours comme fondement théorique à l'utilisation des méthodes de ré-échantillonnage pour la sélection de modèles, comme la validation croisée, ce sur quoi nous reviendrons dans ce manuscrit, voir section 1.4.2.

1.2.2 Décomposition du risque

On désigne par *oracle* la fonction $f^\star$ qui est solution du problème dans l'hypothèse où $\mathbb{P}$ est connue et en utilisant une fonction de perte L quelconque dans l'équation (1.1) :

$$\text{la fonction oracle par } f^\star \quad = \quad \arg\min_{f \in \mathcal{F}^\star} \mathcal{R}(f), \tag{1.4}$$

où l'ensemble fonctionnel $\mathcal{F}^\star$ est inconnu mais comprend la fonction $f^\star$ avec de plus $\mathcal{F} \subset \mathcal{F}^\star$.

Remarque 1.1 ($\mathcal{F}$ et $\mathcal{F}^\star$)**.** *L'ensemble $\mathcal{F}$ est l'ensemble auquel l'analyste a accès grâce à sa modélisation du processus. C'est par exemple $\mathbb{R}^p$ pour modéliser une variable Y en fonction de p covariables au travers d'un modèle linéaire classique sans ordonnée à l'origine. Dans la réalité le processus étudié est souvent bien plus complexe l'ensemble $\mathcal{F}^\star$ représente l'espace*

hypothétique à partir duquel l'information est prise pour produire $f^\star$. C'est donc naturellement que nous avons $\mathcal{F} \subset \mathcal{F}^\star$.

L'oracle est, bien entendu, inconnue puisque l'échantillon $\mathcal{D}_n$ est fini et que $\mathcal{F}^\star$ est inconnu. Dans ce qui suit nous introduisons

$$\text{la fonction oracle sur } \mathcal{F} \text{ par } g^\star = \arg\min_{f\in\mathcal{F}} \mathcal{R}(f). \tag{1.5}$$

On peut alors décomposer

$$\mathcal{R}\left(\hat{f}_n\right) - \mathcal{R}(f^\star) \quad = \quad \mathcal{R}\left(\hat{f}_n\right) - \mathcal{R}(g^\star) \quad + \quad \mathcal{R}(g^\star) - \mathcal{R}(f^\star) \ .$$

La première différence à droite de l'égalité décrit l'erreur d'estimation et la seconde décrit l'erreur d'approximation.

Intuitivement, mais cela n'a pas vertu de démonstration, lorsque la complexité de $\mathcal{F}$ est faible, les risques associés à $g^\star$ et $\hat{f}_n$ vont être assez proches et tout deux assez éloignés du risque de $f^\star$. Inversement, lorsque l'ensemble $\mathcal{F}$ va se complexifier, $g^\star$ va se rapprocher de $f^\star$ pour s'éloigner de $\hat{f}_n$ puisque ce dernier n'a pas accès à l'information de $\mathbb{P}$. La section suivante approfondit la description du risque en l'appliquant à une fonction de perte bien particulière, la perte quadratique

1.2.3 Compromis biais-variance du risque quadratique

Si l'on suppose maintenant que L est la fonction de perte quadratique, Geman et collab. (1992) ont démontré que l'espérance du risque de $\hat{f}_n$ peut se décomposer en trois termes comme suit

$$\mathbb{E}\left[\mathcal{R}\left(\hat{f}_n\right)\right] \quad = \quad \text{bruit}\left(\hat{f}_n\right) \quad + \quad \text{biais}^2\left(\hat{f}_n\right) \quad + \quad \text{var}_{est}\left(\hat{f}_n\right) \ , \tag{1.6}$$

où le terme « bruit $\left(\hat{f}_n\right)$ » est irréductible, car associé à l'erreur d'observation, alors que les deux autres termes sont associés à la complexité de $\mathcal{F}$. Nous allons démontrer cette relation pour ensuite interpréter chacun des termes. Cette démonstration permet de se familiariser avec les outils de l'apprentissage statistique.
La première sous-section est occupée par la démonstration à proprement parler alors que la seconde offre l'estimation qui est souvent utilisée dans la pratique, celle du risque empirique.

Cas général

Soit $\mathcal{D}_n$ un échantillon d'entraînement tiré aléatoirement et x un échantillon de test aussi choisi aléatoirement, donc indépendant de $\mathcal{D}_n$, mais fixé. On peut noter $\hat{y} = \hat{f}_n(x)$ la prédiction de y sachant x et $\mathcal{D}_n$. Deux processus aléatoires indépendants sont en jeu ici, à x fixé, qui sont associés

— au processus d'échantillonage des données d'entraînement, que nous symboliserons par $\mathcal{D}_n$ et $\mathbb{E}_n[\bullet]$ pour les calculs d'espérance associés.

— au processus d'échantillonage des données de test, que nous symboliserons par y et $\mathbb{E}_y[\bullet]$ pour les calculs d'espérance associés.

Remarque 1.2. *Ce point est fondamental dans le cas de l'apprentissage statistique. La suite de cette preuve s'appuie sur l'indépendance de ces deux sources d'aléa afin de produire une décomposition du risque comme une somme de termes. Chaque terme pouvant être associé à une des sources d'aléa mais aussi à l'approximation qui est faite sur l'estimateur.*
Si l'on raisonne géométriquement, puisque ces processus aléatoires sont indépendants, l'utilisation d'une norme telle que celle induite par le risque quadratique décompose le risque selon ces directions. C'est un cas d'application du théorème de Pythagore. Ce que nous allons détailler dans cette section en utilisant une terminologie statistique uniquement.
Cette remarque est à rapprocher de la remarque 1.4 qui considère une nomre matricielle.

Le calcul de l'espérance que l'on peut avoir sur l'erreur $\left(f - \hat{f}_n\right)^2$, aussi appelé « risque quadratique espéré » par exemple, est défini en intégrant toutes les possibilités acceptables de $\mathcal{D}_n$ et de f. On peut ainsi écrire

$$\mathbb{E}\left[\mathcal{R}(\hat{f}_n)\right] = \mathbb{E}_{\mathcal{D}_n,y}\left[(y-\hat{y})^2\right] = \mathbb{E}_{n,y}\left[(y-\hat{y})^2\right],$$

en se permettant d'alléger les notations via $\mathcal{D}_n \mapsto n$. En utilisant la notion d'oracle présentée plus haut au travers de $y^\star$, qui s'interprète comme l'espérance de y sachant x, on a donc $y^\star = \mathbb{E}_y[y|x]$ et on obtient

$$\begin{aligned}\mathbb{E}\left[\mathcal{R}(\hat{f}_n)\right] &= \mathbb{E}_{n,y}\left[(y-y^\star+y^\star-\hat{y})^2\right] \\ &= \mathbb{E}_{n,y}\left[(y-y^\star)^2\right] + \mathbb{E}_{n,y}\left[(y^\star-\hat{y})^2\right] \\ &\quad -2\times\mathbb{E}_{n,y}\left[(y-y^\star)(y^\star-\hat{y})\right],\end{aligned}$$

où l'on peut remarquer que le terme $(y-y^\star)$ est indépendant de $\mathcal{D}_n$ ce qui simplifie la première espérance en $\mathbb{E}_y\left[(y-y^\star)^2\right]$. De la même manière, le terme $(y^\star-\hat{y})$ est indépendant de y, la seconde espérance devient alors $\mathbb{E}_n\left[(y^\star-\hat{y})^2\right]$. Pour les mêmes raisons, le terme croisé peut se développer en un produit de deux termes :

$$\mathbb{E}_y\left[(y-y^\star)\right]\mathbb{E}_n\left[(y^\star-\hat{y})\right].$$

Or l'oracle est sans biais donc le premier terme s'annule et donc ce produit s'annule par intégrité. Il reste au total

$$\mathbb{E}\left[\mathcal{R}(\hat{f}_n)\right] = \mathbb{E}_y\left[(y-y^\star)^2\right] + \mathbb{E}_n\left[(y^\star-\hat{y})^2\right], \tag{1.7}$$

où l'on interprète le premier terme comme l'erreur d'observation, qui est irréductible car associée au processus aléatoire d'observation de y. Ce terme est appelé $\mathrm{var}(bruit)$ dans l'équation (1.6). Si l'on considère maintenant l'espérance de $\hat{y}$ sur l'ensemble des jeux de données $\mathcal{D}_n$ accessibles, que nous noterons $g^\star = \mathbb{E}_n[\hat{y}]$ en reprenant la notion d'oracle restreinte à $\mathcal{F}$ vue plus haut, il est possible de réécrire le dernier terme de l'équation (1.7) tel que

$$\begin{aligned}\mathbb{E}_n\left[(y^\star-\hat{y})^2\right] &= \mathbb{E}_n\left[(y^\star-g^\star+g^\star-\hat{y})^2\right] \\ &= \mathbb{E}_n\left[(y^\star-g^\star)^2\right] + \mathbb{E}_n\left[(g^\star-\hat{y})^2\right] \\ &\quad -2\times\mathbb{E}_n\left[(y^\star-g^\star)(g^\star-\hat{y})\right],\end{aligned}$$

dans l'esprit de ce qui a été fait pour la précédente décomposition, où les espérances ne sont plus associées qu'à $\mathcal{D}_n$. Le premier des trois termes à droite de l'équation calcule une espérance

d'un terme indépendant de l'échantillon $\mathcal{D}_n$, il vient $\mathbb{E}_n\left[\left(y^\star - g^\star\right)^2\right] = \left(y^\star - g^\star\right)^2$. Le troisième terme, pour les mêmes raison peut se réécrire en

$$\begin{array}{rcl} \mathbb{E}_n\left[\left(y^\star - g^\star\right)\left(g^\star - \hat{y}\right)\right] & = & \left(y^\star - g^\star\right)\mathbb{E}_n\left[g^\star - \hat{y}\right] \quad = \quad \left(y^\star - g^\star\right) \times \left(g^\star - \mathbb{E}_n\left[\hat{y}\right]\right) \\ & = & \left(y^\star - g^\star\right) \times \left(g^\star - g^\star\right) \quad = \quad 0 \end{array}$$

Au total on peut écrire, pour un échantillon d'apprentissage $\mathcal{D}_n$, les grandeurs détaillées dans l'équation (1.6) conduisent à l'établissement des relations présentées dans la liste (1.8).

Termes d'erreur du risque empirique

$$\begin{array}{lcl} \text{Erreur de bruit} & : & \text{bruit}\left(\hat{f}_n\right) = \mathbb{E}_x\left[\mathbb{E}_y\left[\left(f(x) - f^\star(x)\right)^2\right]\right] \\ \text{Erreur d'approximation} & : & \text{biais}^2\left(\hat{f}_n\right) = \mathbb{E}_x\left[\left(f^\star(x) - g^\star(x)\right)^2\right] \\ \text{Erreur d'estimation} & : & \text{var}_{est}\left(\hat{f}_n\right) = \mathbb{E}_x\left[\mathbb{E}_{\mathcal{D}_n}\left[g^\star(x) - \hat{f}_n(x)\right]\right] \end{array} \tag{1.8}$$

Cas particulier du risque empirique

Nous avons déjà défini le risque empirique au travers de l'équation (1.2), par utilisation de l'estimateur de la moyenne on peut décrire chaque terme d'erreur. Ce qui est résumé dans la liste (1.9).

Termes d'erreur du risque empirique

$$\begin{array}{lcl} \text{Erreur de bruit} & : & \widehat{\text{bruit}}\left(\hat{f}_n\right) = \frac{1}{n}\sum_{i=1}^n \left(f\left(x_i\right) - f^\star\left(x_i\right)\right)^2 \\ \text{Erreur d'approximation} & : & \widehat{\text{biais}^2}\left(\hat{f}_n\right) = \frac{1}{n}\sum_{i=1}^n \left(f^\star(x_i) - g^\star(x_i)\right)^2 \\ \text{Erreur d'estimation} & : & \widehat{\text{var}}\left(\hat{f}_n\right) = \frac{1}{n}\sum_{i=1}^n \left(g^\star(x_i) - \hat{f}_n(x_i)\right)^2 \end{array} \tag{1.9}$$

1.2.4 Interprétation de l'espérance du risque quadratique

Cette section utilise la forme (1.6), dont les termes sont détaillés dans l'équation (1.9), afin de décrire le comportement du risque quadratique. Cette section s'appuie sur la figure 3 comme aide à l'interprétation.

D'après la décomposition (1.6) et la figure 3, pour un sous-espace fonctionnel $\mathcal{F}$ donné, les variables $\mathcal{D}_n$ et f admettent des réalisations observables via $\hat{f}_n$ et f représentées par les points bleus et violets respectivement. Une observation de f, notée f_1 est représentée en vert ainsi qu'une observation de $\hat{f}_{n_1}$ en bleue foncée. Ces deux points sont sensibles aux deux processus $\mathcal{D}_n$ et f qui sont les seuls à introduire de l'aléatoire dans cette expérience. La longueur du trait double représente l'erreur quadratique commise par l'utilisateur en construisant son modèle sur $\mathcal{D}_{n_1}$ et f_1.

En espérance, l'utilisateur risque une erreur qu'il est possible de décomposer en trois termes, qui sont :

— L'**erreur d'approximation**, ou **biais au carré**, constante à $\mathcal{F}$ fixée mais qui diminue pour une complexité de l'estimateur qui augmente, $g^\star$ se rapproche alors de $f^\star$. Ce terme est particulièrement lourd à estimer, comme démontré dans la section 1.2.5, qui fait suite à la section actuelle.

— L'**erreur de bruit**. Qui s'obtient en sommant sur les réalisations de f, les points en violet sur la figure. La sphère violette de rayon $E_f[(f^\star - f)^2]$ est une représentation de l'erreur d'observation, centrée sur $f^\star$. On remarque que cette forme n'est pas modifiée par le choix de $\mathcal{D}_n$, on dit que cette erreur est irréductible car associée seulement au processus d'observation.

— L'**erreur d'estimation** sur $\mathcal{D}_n$ de la fonction objectif. Plus le modèle est complexe et plus le nombre de paramètres va augmenter ce qui a pour effet de faire croître cette erreur. Plus intuitivement, l'espace concentré dans le cercle de rayon $E_n[(g^\star - \hat{f}_n)^2]$ représente la volatilité des erreurs de modèles, si $\mathcal{F}$ avait été choisi plus petit, une droite dans la vision schématique, l'espace associé aurait été plus faible et donc l'erreur d'approximation aurait été moins importante. Inversement, en prenant $\mathbb{R}^3$, l'espace enclos dans la sphère de rayon $E_n[(g^\star - \hat{f}_n)^2]$ aurait été plus grande et donc l'erreur aurait grimpé.

Ces développements montrent que la complexité des espaces de fonctions considérés ainsi que la profondeur d'échantillonnage ont un impact direct sur le risque associé. Le cas particulier de la perte quadratique a uniquement été traité mais Domingos (2000) a démontré que ce genre de décomposition peut se retrouver pour de nombreuses fonctions de perte.

La section suivante développe le calcul des estimateurs du risque quadratique empirique et du biais associé. On montre que ce dernier est particulièrement onéreux à calculer.

1.2.5 Estimations du risque quadratique et du biais associé

Supposons l'échantillon aléatoire $\mathcal{D}_n$ sur lequel il est possible de construire l'estimateur $\hat{f}_n$, alors afin de construire un estimateur du risque quadratique et un estimateur du biais^2, nous allons construire un premier estimateur de $g^\star$ que l'on rappelle pouvoir définir comme

$$g^\star(.) \quad = \quad \mathbb{E}_n\left[\hat{f}_n(.)\right],$$

où la notation « $\hat{f}_n(.)$ » permet d'expliciter une fonction applicable à l'échantillon « . » et dont l'estimation des paramètres se fait sur $\mathcal{D}_n$.

Dans la suite, nous notons $D_n = \{\{x_{n,1}, y_{n,1}\}, \ldots, \{x_{n,n}, y_{n,n}\}\}$ une réalisation de la variable aléatoire $\mathcal{D}_n$. Supposons un échantillon à N observations indépendantes de $\mathcal{D}_n$ que l'on notera

$$\mathbf{D}_N = \left\{D_n^{(1)}, \ldots, D_n^{(N)}\right\} \quad \text{et} \quad \forall j \in [\![1, N]\!], \quad D_n^{(j)} = \left\{\left\{x_{n,1}^{(j)}, y_{n,1}^{(j)}\right\}, \ldots, \left\{x_{n,n}^{(j)}, y_{n,n}^{(j)}\right\}\right\}.$$

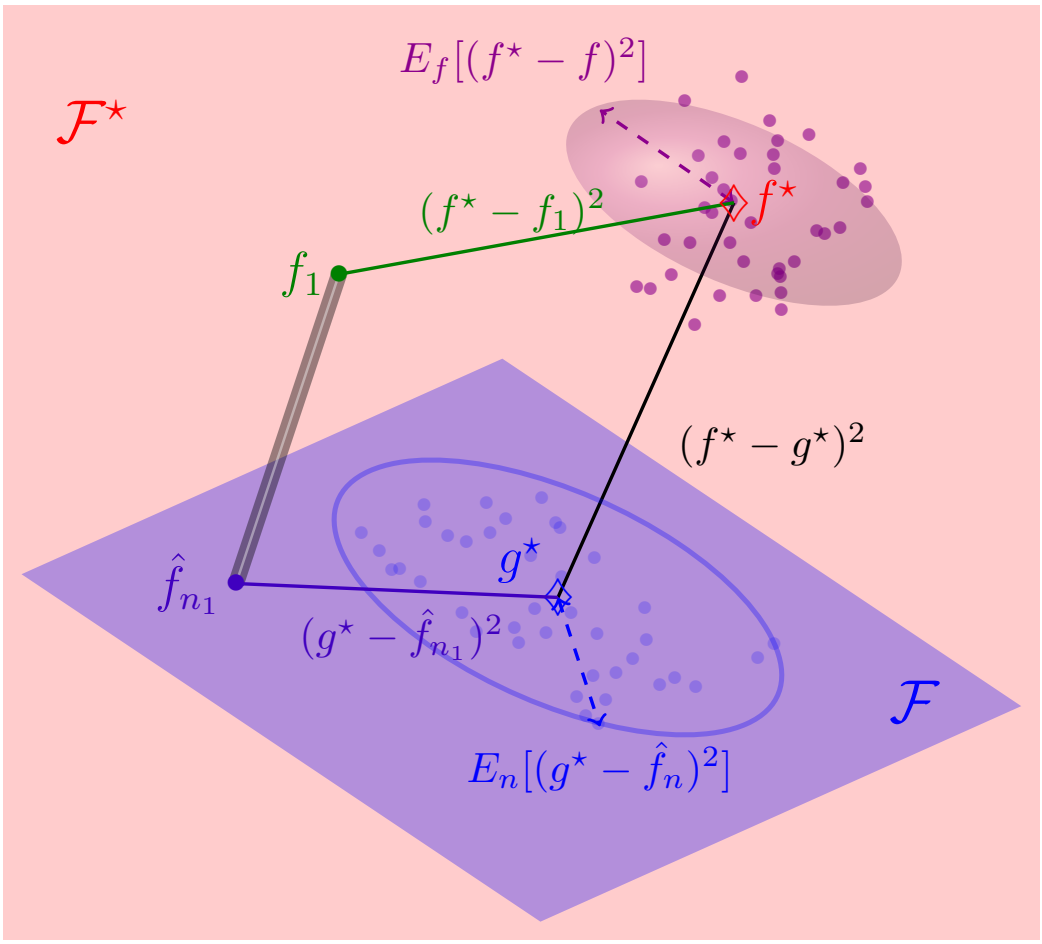


FIGURE 3 – Représentation de l'oracle $f^\star$ et de l'oracle restreint à $\mathcal{F}$ noté $g^\star$. En rouge l'espace $\mathcal{F}^\star$ (ici de dimension apparente égale à 3) et en bleu l'espace des modèles accessibles (de dimension apparente égale à 2) symbolisé par un plan. Les différents points représentent des tirages de $\mathcal{D}_n$ et f, un couple $(f_1, \hat{f}_{n_1})$ symbolise un tirage particulier. Les points et les sphères associées représentent d'autres tirages et les variances associées pour chacun des deux processus. L'utilisateur commet l'erreur quadratique empirique symbolisée par le trait double en considérant l'expérience $(f_1, \mathcal{D}_{n_1})$. Il peut espérer un risque quadratique qui correspond à la somme $1^o)$ des erreurs de bruit symbolisées par les longueurs vertes, $(f^\star - f_1)^2$ pour f_1, en sommant sur f, $2^o)$ des erreurs d'estimation symbolisées par les longueurs bleues foncées, $\left(g^\star - \hat{f}_{n_1}\right)^2$ pour $\mathcal{D}_{n_1}$, en sommant sur $\mathcal{D}_n$ et $3^o)$ du biais du à l'approximation symbolisée par le trait noir, $(f^\star - g^\star)^2$.

Il est alors possible d'estimer l'espérance de $g^\star$, que l'on notera $\hat{g}^\star_N$, via l'estimateur de

$$\text{la moyenne empirique de } g^\star \quad : \quad \hat{g}^\star_N(.|\mathbf{D}_N) \quad = \quad \frac{1}{N}\sum_{j=1}^{N} \hat{f}_n\left(.|D_n^{(j)}\right), \tag{1.10}$$

où il est donc nécessaire de calculer sur $\mathbf{D}_n$ entier.
L'espérance du risque empirique peut être classiquement approchée grâce à la moyenne empirique sur l'échantillon $\mathbf{D}_N$, que l'on note $\mathbf{R}_n\left(\hat{f}_n|\mathbf{D}_n\right)$ dans la suite et que l'on formalise via

$$\begin{array}{c}\text{la moyenne empirique de l'estimateur du risque empirique}\\ \mathbf{R}_N\left(\hat{f}_n|\mathbf{D}_N\right) = \frac{1}{N}\sum_{j=1}^{N} \hat{\mathcal{R}}_n\left(\hat{f}_n|D_n^{(j)}\right).\end{array} \tag{1.11}$$

En rappelant que le biais^2 de l'estimateur $\hat{f}_n$ peut être écrit

$$\text{biais}^2\left(\hat{f}_n\right) \quad = \quad \frac{1}{n}\sum_{i=1}^{n}\left(f^\star(x_i) - g^\star(x_i)\right)^2,$$

et ainsi une estimation de son espérance, que nous noterons **biais**^{2} peut être réalisée en utilisant la moyenne empirique et l'équation (1.10) et qui se met sous la forme de

$$\begin{array}{c}\text{la moyenne empirique du biais quadratique}\\ \mathbf{biais}^2_N\left(\hat{f}_n|\mathbf{D}_N\right) = \frac{1}{nN}\sum_{j=1}^{N}\sum_{i=1}^{n}\left(f^\star(x_{n,i}^{(j)}) - \hat{g}^\star_N\left(x_{n,i}^{(j)}|\mathbf{D}_N\right)\right)^2,\end{array} \tag{1.12}$$

après utilisation de l'estimateur proposé pour $g^\star$ via l'équation (1.10).

Les estimateurs (1.12) et (1.11) sont utilisés dans la sous-section 1.3.4 pour détailler le comportement de l'estimateur Lasso. En effet $g^\star$ n'admet pas, dans le cas général, de forme analytique.

La section suivante développe le cas du modèle linéaire avec résolution par MCO qui accompagnera l'ensemble de cette lecture.

1.2.6 Cas particulier du problème linéaire

Le modèle linéaire suppose qu'il existe un vecteur $\mathbf{b}$ à p lignes qui soit associé au vecteur aléatoire y. Comme jusqu'à présent, $\mathbf{b}$ est déterministe et on suppose une erreur de mesure (ou d'observation) ϵ centrée de variance σ^2_ϵ additive de façon à pouvoir écrire le modèle de mesure (ou d'observation) suivant

$$y \quad = \quad X^T\mathbf{b} \quad + \quad \epsilon,$$

Soit un échantillon $\mathcal{D}_n$ tiré aléatoirement, il est possible d'utiliser un formalisme matriciel en notant la matrice des réponses $\mathbf{y} \in \mathbb{R}^n$, le vecteur des covariables $\mathbf{X} \in \mathbb{R}^{n\times p}$ et le vecteur de bruit $\mathbf{e} \in \mathbb{R}^n$. Il vient alors le modèle matriciel

$$\mathbf{y} = \mathbf{X}\mathbf{b} + \mathbf{e}. \tag{1.13}$$

Le modèle étant défini, il est possible d'estimer, pour une nouvelle observation de test $\mathbf{x}_1$ tiré aléatoirement, les grandeurs $f^\star(\mathbf{x}_1)$ et $f(\mathbf{x}_1)$ telles que :

$$\begin{array}{rcl} f^\star(\mathbf{x}_1) & = & \mathbf{x}_1\mathbf{b}, \\ f(\mathbf{x}_1) & = & \mathbf{x}_1\mathbf{b} + \epsilon_1, \end{array}$$

en notant ϵ_1 le bruit d'observation.

Solution des moindres carrés ordinaires

Le problème des MCO cherche à minimiser la perte quadratique du modèle (1.13), ce qui revient à écrire

$$\hat{\mathbf{b}}_n^{(MCO)} = \arg\min_{\mathbf{b}\in\mathbb{R}^p} \|\mathbf{y} - \mathbf{X}\mathbf{b}\|_2^2 .$$

Ce problème a pour solution

$$\text{l'estimateur des MCO noté } \hat{\mathbf{b}}_n^{(MCO)} = \mathbf{X}^+\mathbf{y}, \tag{1.14}$$

où l'on note $\mathbf{X}^+$ la pseudo-inverse de Moore-Penrose [(a)] de $\mathbf{X}$. Comme précédemment, la fonction ayant été estimée sur l'échantillon d'entraînement, il est possible d'estimer, pour un échantillon de test $\mathbf{x}_1$ tiré aléatoirement, les grandeurs $g^\star(\mathbf{x}_1)$ et $\hat{f}_n^{(MCO)}(\mathbf{x}_1)$ telles que :

$$\begin{aligned} g^\star(\mathbf{x}_1) &= \mathbf{x}_1\mathbf{b}, \\ \hat{f}_n^{(MCO)}(\mathbf{x}_1) &= \mathbf{x}_1\mathbf{b} + \mathbf{x}_1\mathbf{X}^+\boldsymbol{\epsilon} \end{aligned}$$

voir l'annexe B.1 pour les démonstrations de ces résultats.

Biais, variance et cas dégénérés

Il est possible de connaître les termes d'erreurs pour l'échantillon $\mathbf{x}_1$ au travers de termes décrits ci-dessus. La décomposition en trois termes du biais quadratique de l'estimateur des MCO est démontrée en annexe B.1 et rappelée dans la liste (1.15).

Termes d'erreur pour l'estimateur MCO

$$\begin{aligned} &\text{Erreur de bruit} &:\ &\text{bruit}\left(\hat{b}_n^{(MCO)}\right) &=\ &\sigma_\epsilon^2 \\ &\text{Erreur d'approximation} &:\ &\text{biais}^2\left(\hat{b}_n^{(MCO)}\right) &=\ &0 \\ &\text{Erreur d'estimation} &:\ &\text{var}_{est}\left(\hat{b}_n^{(MCO)}\right) &=\ &p\sigma_\epsilon^2 \text{Trace}\left(\mathbf{X}^{+T}\mathbf{X}^+\right) \end{aligned} \tag{1.15}$$

L'espérance du risque empirique s'écrit comme la somme des trois termes associés aux trois erreurs, soit

$$\mathbb{E}\left[\mathcal{R}\left(\hat{b}_n^{(MCO)}\right)\right] = \sigma_\epsilon^2 + p\sigma_\epsilon^2 \text{Trace}\left(\mathbf{X}^{+T}\mathbf{X}^+\right).$$

On déduit de cette étude deux choses :

— l'estimateur des moindres carrés est sans biais,

— la variance de l'estimateur des MCO diverge si la matrice échantillon $\mathbf{X}$ n'est plus inversible.

(a). L'Annexe A permet de rappeler les principes de pseudo-inverse, de pseudo-inverse de Moore-Penrose ainsi que de décomposition en valeurs singulières (*SVD*) (Pearson, 1901). La connaissance de ces outils est nécessaire à la poursuite de la lecture. Voir notamment le Théorème A.3 pour l'intérêt de la pseudo-inverse de Moore-Penrose à la résolution du problème des MCO généralisée à tous les rangs de la matrice $\mathbf{X}$.

Le *conditionnement*, qui peut par exemple se calculer en prenant le rapport de la plus grande valeur singulière sur la plus petite, pour une matrice $\mathbf{X}$ donnée, permet d'évaluer la difficulté d'inverser numériquement une matrice. Si le conditionnement de la matrice diverge, alors la variance de l'estimateur des MCO diverge aussi.

Si l'on suppose un design orthogonal (voir Remarque 1.3) qui correspond à $\mathbf{X}^T\mathbf{X} = n\mathbb{I}_p$, il vient que $\mathbf{X}^{+T}\mathbf{X}^+ = \frac{1}{n}\mathbb{I}_n$ et donc Trace $\left(\mathbf{X}^{+T}\mathbf{X}^+\right) = 1$ et alors

$$\text{l'espérance du risque de l'estimateur des MCO pour un design orthogonal} \quad \mathbb{E}\left[\mathcal{R}_{ortho}\left(\hat{\mathbf{b}}_n^{(MCO)}\right)\right] = \sigma_\epsilon^2(1+p). \tag{1.16}$$

Le risque quadratique est donc une fonction affine de la complexité du modèle linéaire représentée par p.

Remarque 1.3 (Le design orthogonal)**.** *L'astuce du design orthogonal suppose que la matrice de covariance empirique est égale à l'identité, soit $\mathbf{X}^T\mathbf{X} = n\mathbb{I}_p$ et donc que les variables de $\mathcal{D}_n$ ne sont pas corrélées. Cette hypothèse semble très forte mais permet plusieurs choses comme*

- *d'offir des ordres de grandeurs comme ici avec l'équation* (1.16) *pour les MCO mais aussi pour le Ridge comme nous le verrons dans l'équation* (1.21)*.*
- *De résoudre des équations non linéaires en offrant des conditions de variables découplées, comme nous le verrons dans la section sur le Lasso au paragraphe 1.3.4.*

C'est donc une hypothèse à l'ordre 0 si l'on suppose la variable X aléatoire sans autre hypothèse de distribution.

Conclusion

On dit assez naturellement que l'estimateur de MCO est non biaisé. Du point de vue du risque quadratique, on peut aussi dire que l'estimateur des MCO est mauvais dès que le conditionnement de la matrice $\mathbf{X}$ diminue.

Le théorème de Gauss-Markov montre que cet estimateur est celui qui minimise sa variance pour un biais nul, de ce point de vue l'estimateur des MCO est parfois appelé *Best Linear Unbiased Estimator* (*BLUE*). L'idée de la régularisation est donc de permettre un biais d'estimation afin de réduire la variance de l'estimateur en question pour les cas de mauvais conditionnement de $\mathbf{X}$. On entend souvent les termes de « matrices dégénérées » et de « problèmes mal posés ».

1.3 Grande dimension et régularisation

Nous considérons le risque empirique, défini dans l'équation (1.2) dans le cas particulier du risque quadratique, qui peut donc s'écrire comme

$$\text{le risque empirique quadratique } \hat{\mathcal{R}}_n(f) = \frac{1}{n}\sum_{i=1}^{n}(Y_i - f(X_i))^2 \tag{1.17}$$

sur l'échantillon $\mathcal{D}_n$ en supposant que la variable aléatoire Y associée aux observations de la réponse est vectorielle, de dimension $q = 1$ pour simplifier les notations, et que l'on cherche à l'expliquer au moyen d'un modèle faisant intervenir la fonction $f : \mathbb{R}^p \to \mathbb{R}$ qui s'applique la variable aléatoire X vectorielle de dimension p. Les hypothèses d'indépendance sont quant à elles définies en introduction.

1.3.1 Enjeux de la grande dimension

La grande dimension pose deux problèmes principaux dans le cas du modèle linéaire.
Le premier est associé à ce qui a été montré au travers de l'équation (1.16), à savoir que la variance de l'estimateur des MCO augmente linéairement avec p, ce qui implique selon toute probabilité de mauvaises performances en prédiction. p représentant le nombre de variables dans le modèle, il faut donc sélectionner les variables à inclure dans le modèle.

Il a été observé que le mauvais conditionnement de la matrice $\mathbf{X}^T\mathbf{X}$ induisait cette divergence de l'erreur d'estimation. S'il n'est pas recherché de sélectionner des variables, il peut être raisonnable de choisir de modifier cette matrice en modifiant son spectre. C'est la *régularisation*.
Nous allons dans cette section détailler différentes méthodes objectivant de résoudre ces deux problèmes. Les deux premières méthodes y parviennent en sélectionnant les variables.

1.3.2 Solution « brute force »

Soit le modèle de régression linéaire résolu par la méthode des MCO. Le modèle présente un nombre de paramètres de l'ordre de p. Supposons que l'utilisateur désire construire un estimateur des MCO à $k \leqslant n$ variables, il y a C_p^k possibilités.

Si maintenant tous les modèles possibles doivent être testés, ce qui est la méthode la plus efficace, le nombre de possibilités (2^p) limite son utilisation puisque les valeurs prises par p sont de l'ordre de plusieurs dizaines de milliers. Nous ne retiendrons donc pas cette méthode dans ce manuscrit.

1.3.3 Méthodes pas à pas

Naturellement, on peut considérer les méthodes de sélection où pour le modèle courant on peut choisir d'ajouter la variable utile ou inversement de retirer la moins utile. L'utilité d'une variable dans un modèle peut être évaluée au moyen d'un des critères de qualité détaillés dans la section 1.4. A l'étape $k \in [\![1, p]\!]$, $p - k$ modèles sont testés, ce qui implique de tester $p(p-1)/2$ modèles au total. Ce peut être laborieux et ne sera pas une méthode de choix.

1.3.4 Solutions régularisées

Les problèmes mal posés sont étudiés depuis le siècle dernier mais c'est Tikhonov (1943) qui a en premier développé un cadre théorique capable de répondre à ce problème. C'est ce que l'on appelle la *régularisation*. Ceci se caractérise par une modification du problème de minimisation (1.17) au travers d'une version *pénalisée*, ou *régularisée*, du problème initial de minimisation du risque empirique appelé

$$\text{risque empirique pénalisé et noté} \qquad \min_{f\in\mathcal{F}} \quad \hat{\mathcal{R}}_n(f) + \lambda\|f\|_{\mathcal{F}}, \tag{1.18}$$

où $\|.\|_{\mathcal{F}}$ est une fonctionnelle définie sur $\mathcal{F}$ et à valeur dans $\mathbb{R}^+$ et est appelée pénalité fonctionnelle. Le coefficient λ est appelé coefficient de régularisation. La fonctionnelle $\|.\|_{\mathcal{F}}$ est classiquement un produit scalaire sur l'espace $\mathcal{F}$. Le choix de cette fonctionnelle traduit les objectifs de l'analyse. En effet, certaines fonctionnelles permettent de forcer des variables ou des groupes de variables à être exclus ou bien conservés dans le modèle. Ces fonctionnelles peuvent se baser sur une structure connue des données, des groupes de variables par exemple.

On peut remarquer que ce genre de décomposition est formé par la somme de deux termes correspondants à la variance du modèle pour le terme de droite et au biais pour le terme de gauche. Ce n'est pas sans rappeler les résultats sur la décomposition du risque structurel étudiée dans la partie 1.2. Le compromis est lui-même recherché au moyen du coefficient de pondération λ ou le terme de gauche peut être associé à la variance de l'estimateur et le terme de droit à son biais.

La régularisation fait en quelque sorte le pari de :

— légèrement augmenter l'erreur d'approximation en ajoutant quelques dimensions au problème au travers de paramètre λ et des paramètres de la fonctionnelle $\|.\|_{\mathcal{F}}$. Dans le cas linéaire, cela revient à augmenter la complexité telle que

$$p \to p + 1 + \#\left\{\text{paramètres de la fonctionnelle } \|.\|_{\mathcal{F}}\right\},$$

— tout en réduisant l'erreur d'estimation, grâce à un choix judicieux de la fonctionnelle $\|.\|_{\mathcal{F}}$ et du paramètre λ.

Les « $1 + \#\left\{\text{paramètres de la fonctionnelle } \|.\|_{\mathcal{F}}\right\}$ » paramètres supplémentaires sont appelés *hyperparamètres* ou *paramètres de régularisation*. Certains de ces hyperparamètres sont interprétables, comme c'est le cas en analyse bayésienne, et peuvent être fixés par la connaissance que l'on a *a priori* des données mais d'autres nécessitent un travail de validation pour être fixés. Les outils de validation ont déjà été présentés (voir section 1.4), les prochaines sous-sections s'intéressent à la description des méthodes de régression les plus courantes.

La régularisation Ridge

On la doit à Hoerl et Kennard (1970) et la dénomination de régularisation $\mathcal{L}_2$ est aussi rencontrée. Cette régularisation peut s'appliquer que la réponse $\mathbf{Y}$ soit unidimensionnelle ($q = 1$) ou multidimensionnelle ($q > 1$). On note alors

$$\text{l'estimateur Ridge, } \forall\lambda > 0 : \hat{\mathbf{B}}_n^{(Ridge)} = \left(\mathbf{X}^T\mathbf{X}/n + \lambda\mathbb{I}\right)^{-1}\mathbf{X}^T\mathbf{Y}/n. \tag{1.19}$$

On peut remarquer que ce modèle permet l'inversion de la matrice $\mathbf{X}^T\mathbf{X}$ en ajoutant λ à chacune de ses valeurs propres. Cette méthode ne cherche pas à sélectionner de variables.

L'annexe B.2 permet d'estimer les termes d'erreur du risque quadratique pour cet estimateur qui sont rappelés dans la liste (1.20).

Termes d'erreur pour l'estimateur Ridge

$$\begin{array}{lcll} \text{Erreur de bruit} & : & \text{bruit}\left(\hat{b}_n^{(Ridge)}\right) & = \sigma_\epsilon^2 \\ \text{Erreur d'approximation} & : & \text{biais}^2\left(\hat{b}_n^{(Ridge)}\right) & = \lambda^2 \mathbf{b}^T \left(\mathbf{X}^T\mathbf{X}/n + \lambda\mathbb{I}\right)^{-2} \mathbf{b} \\ \text{Erreur d'estimation} & : & \text{var}_{est}\left(\hat{b}_n^{(Ridge)}\right) & = {\sigma_\epsilon}^2 \text{Trace}\left(\left(\mathbf{X}^T\mathbf{X}/n + \lambda\mathbb{I}\right)^{-2} \mathbf{X}^T\mathbf{X}\right)/n \end{array} \tag{1.20}$$

pour cette étude nous supposerons $q = 1$. Dans l'équation (B.5) rappelée ici

$$\begin{aligned} \hat{\mathcal{R}}\left(\hat{\mathbf{b}}_n^{(Ridge)}\right) &= {\sigma_\epsilon}^2 + \lambda^2 \mathbf{b}^T \left(\mathbf{X}^T\mathbf{X}/n + \lambda\mathbb{I}\right)^{-2} \mathbf{b} \\ &+ {\sigma_\epsilon}^2 \text{Trace}\left(\left(\mathbf{X}^T\mathbf{X}/n + \lambda\mathbb{I}\right)^{-2} \mathbf{X}^T\mathbf{X}\right)/n, \end{aligned}$$

le premier terme représente l'erreur due au bruit, le second terme est associé à l'erreur d'estimation (variance du modèle) et le terme suivant est lié à l'erreur d'approximation (biais du modèle au carré). Le terme central est croissant vis à vis du paramètre λ alors que le terme de droite est décroissant par rapport à cette même variable. On peut aussi supposer travailler à design orthogonal [(b)] et réécrire l'équation (B.5)

$$\hat{\mathcal{R}}\left(\hat{\mathbf{b}}_n^{(Ridge)}\right)_{orthogonal} = {\sigma_\epsilon}^2 + \mathbf{b}^T\mathbf{b}\frac{\lambda^2}{(1+\lambda)^2} + {\sigma_\epsilon}^2 p\frac{1}{(1+\lambda)^2}. \tag{1.21}$$

On peut alors noter l'erreur d'approximation (le troisième terme) à son homologue dans le cas des MCO qui vaut ${\sigma_\epsilon}^2 p$. Dans le cas du Ridge, un terme multiplicatif permet de régulariser l'action délétère de p dans le terme d'erreur d'approximation et qui rend le modèle efficace en grande dimension.
La figure 4 présente les différentes composantes du risque quadratique pour un design orthogonal.

On observe un net coude du risque quadratique, au point ⊠, qui correspond au modèle optimal repéré par λ_{opt}. A droite de ce coude l'erreur de biais diminue moins rapidement que n'augmente l'erreur d'approximation, c'est le surapprentissage. A l'inverse, à gauche du coude, l'erreur d'approximation chute très rapidement, bien plus rapidement que n'augmente l'erreur d'estimation, c'est l'apprentissage.

Cette régularisation est appliquée à l'exemple du jeu de données *MNIST* dans la section 1.5.

La régularisation Lasso

Lasso vient de l'anglais *L*east *A*ngle *S*hrinkage and *S*election *O*perator (voir Tibshirani, 1996). La fonctionnelle associée s'écrit comme la somme des valeurs absolues des coefficients de régression dans le cas où $q = 1$ seulement. On écrit alors

$$\|f\|_{\mathcal{F}} = \|f\|_1 = \sum_{j=1}^{p} |b_j|.$$

(b). Comme réalisé pour les MCO où l'on suppose que $\mathbf{X}^T\mathbf{X} = n\mathbb{I}$, voir la sous-section 1.2.6.

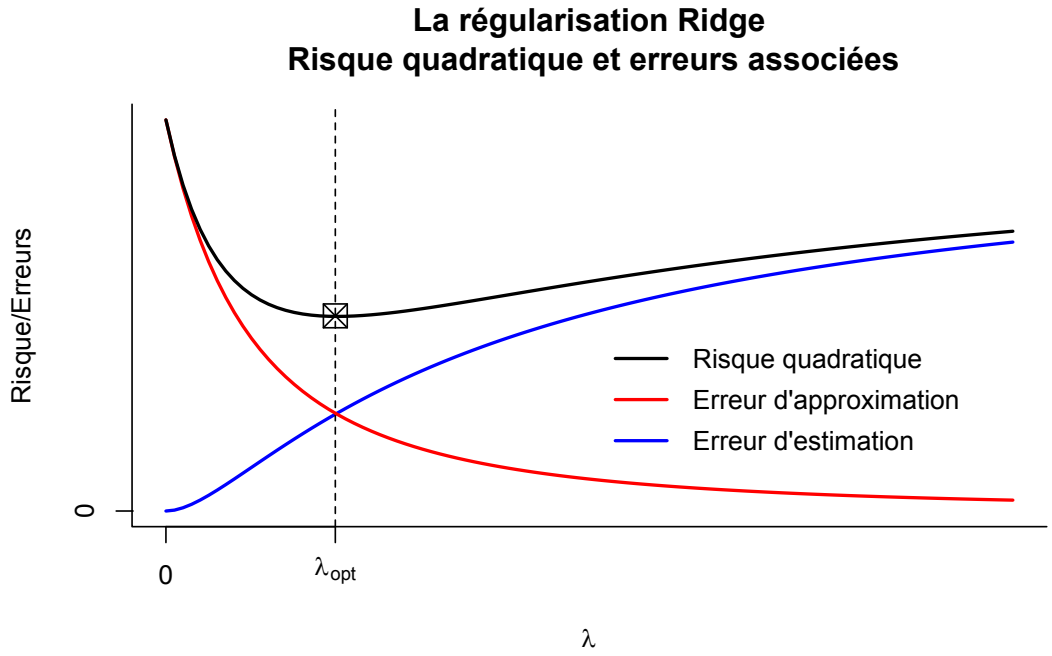


FIGURE 4 – Décomposition du risque quadratique pour l'estimateur régularisé Ridge dans le cas du problème de régression linéaire avec un bruit d'observation centrée. Le symbole ⊠ correspond au minimum de risque quadratique.

C'est un problème non convexe du fait de la présence d'un terme pris en norme 1, la solution n'est donc pas forcément unique. Le cas du design orthogonal permet de résoudre complètement le problème associé, ce qui est présenté en annexe B.3.

Cas du design orthogonal Le design orthogonal correspond à une matrice de covariance empirique égale à l'identité. Les coefficients associés à chaque variable sont donc intuitivement traités de façon séparée. La solution Lasso utilise alors la fonction de seuillage doux (représentée sur la figure 6, page 35) telle que, en reprenant l'annexe B.3,

$$\hat{b}_j^{(Lasso)} = \begin{cases} \hat{b}_j^{(MCO)} - \lambda \text{sign}\left(\hat{b}_j^{(MCO)}\right) & si \left|\hat{b}_j^{(MCO)}\right| > \lambda \\ 0 & sinon \end{cases}, \tag{1.22}$$

où les notations sont explicitées dans l'annexe associée, bien qu'elles soient assez transparentes dans ce cas. Le Lasso, dans ce cas, part de l'estimateur des MCO et rétrécit ses coefficients de façon linéaire d'un coefficient λ jusqu'à tous les annuler. On remarque que le modèle perd en variables lorsque λ grandit. Le modèle Lasso opère donc de la sélection de variables.

Cas général Le cas général n'admet pas de solution exacte mais plusieurs algorithmes ont été proposés, parmi lesquels une modification de l'algorithme de régression à corrélations minimales hiérarchique (*LARS*) (voir EFRON et collab., 2004). La principale idée de cet algorithme étant d'inclure, ou de retirer, une variable du modèle courant en fonction de son intérêt pour le

modèle résultant, en terme d'explication de l'information résiduelle. Les étapes sont rappelées ici :

- l'algorithme part d'un modèle vide, aucune variable n'est sélectionnée,
- la plus petite valeur de λ est estimée de façon à introduire la variable la plus corrélée avec $\mathbf{y}$ dans le modèle,
- le modèle avance dans la direction de cette variable, en réduisant λ, avec le plus grand pas possible, jusqu'à ce que la corrélation de ce modèle avec le résidu corresponde à la corrélation d'une autre variable avec ce même résidu, ou bien qu'un coefficient jusqu'alors non nul s'annule
  - si un coefficient s'annule, la variable en question est retirée du modèle et un nouveau modèle est calculé avec le λ courant et les variables restantes,
  - si un coefficient supplémentaire cesse d'être nul, un nouveau modèle est créé et avance alors dans la direction formée par la bissectrice (d'où le terme de LARS) de ces variables,
- le modèle réitère jusqu'à $\lambda = 0$.

La troisième étape est en réalité résolue de façon exacte en calculant les pentes des droites à chaque point d'ajout d'une variable, on dit que le Lasso résout exactement le chemin de régularisation. Cet algorithme est intrinsèquement associé à une perte quadratique et d'autres algorithmes ont été proposés pour s'adapter à des fonctions de coût différentes quoique convexes comme le fait la méthode développé par Rosset et collab. (2004). Cette méthode utilise les propriétés d'agrégations d'une famille de *weak learners* afin d'obtenir de meilleurs résultats en prédiction, c'est le *boosting*. L'optimisation numérique est effectuée par descente de gradient.
Puisqu'il n'y a pas de forme générale de la solution Lasso, il est impossible d'offrir une estimation du risque quadratique, comme cela a été fait pour les méthodes présentées jusqu'à présent. Des recherches sont réalisées afin de majorer ce risque, on parle d'« inégalié oracle ». Cette dernière est formulée par Koltchinskii (2009) dans le cas du problème linéaire pour $p > n$

$$\frac{\left\|\mathbf{X}\left(\hat{\mathbf{b}}^{(Lasso)} - \mathbf{b}\right)\right\|_2^2}{n} \leqslant const. \frac{\sigma_\epsilon^2 \log(p)}{n} s_0,$$

où s_0 est le nombre de paramètres sélectionnés par le modèle courant et *const.* est une constante à déterminer. On peut remarquer que l'espérance de ce risque pour l'estimateur des MCO est $\sigma_\epsilon^2 p/n$, voir équation (1.16). La dépendance en p est linéaire dans le cas du MCO et logarithmique dans le cas du *Lasso*, ce dernier semble donc être moins atteint par le « fléau de la dimension ».

Afin d'intégrer de façon intuitive le comportement du Lasso, des simulations ont été réalisées avec pour modèle $Y = b_1 X_1 + b_2 X_2 + \epsilon$ où $\epsilon \sim \mathcal{N}(0, 0.8^2)$, $b^\star = (1, 0.5)^T$ et $n = 1000$. On se propose de tirer aléatoirement $N = 300$ échantillons pour lesquels on teste différentes $k_\lambda = 6$ valeurs des paramètres de régularisation Lasso et Ridge. On suppose de plus que $X_1 \sim \mathcal{N}(0, 1)$ et $X_2 \sim \mathcal{N}(0, 1)$ et une corrélation théorique $\rho = 0$ [(c)] puis $\rho = 0.5$. Les $\mathbf{Y}$ sont standardisés de façon à utiliser les moyennes et variances dans la phase de test. La figure 5 montre les résultats des estimations pour ces expériences.

Cette figure est divisée en trois lignes. La première et la seconde correspondant aux cas $\rho = 0$ et $\rho = 0.5$ respectivement pour le Ridge à gauche et le Lasso à droite. La troisième ligne correspond à des estimations du risque empirique et du biais quadratique des biais des estimateurs Ridge et Lasso. On propose dans la Remarque 1.2.5 un estimateur du biais quadratique, noté $\mathbf{R}_N\left(\hat{f}_n|\mathbf{D}_n\right)$ voir l'équation (1.12). Ainsi qu'un estimateur du risque empirique quadratique, noté $\mathbf{biais}^2_N\left(\hat{f}_n|\mathbf{D}_n\right)$ voir l'équation (1.11).
Sur cette figure, on note alors :

— $f^\star$, l'oracle qui est représenté par des points noirs ●.

— $\hat{f}_n$, les estimateurs qui sont représentés par les points colorés (●, ●, ●, ●, ●, ●) par pénalisation croissante.

— $g^\star$, les espérances des estimateurs qui sont représentées par des croix encerclées et colorées (⊕, ⊕, ⊕, ⊕, ⊕, ⊕) par pénalisation croissante. De plus, le lieu des $g^\star$ pour ces données est symbolisé par un trait en pointillé.

— $\bar{f}_n$, les moyennes empiriques des estimateurs qui sont représentées par des losanges colorés (♦, ♦, ♦, ♦, ♦, ♦) par pénalisation croissante.

Les modèles en magenta correspondent à des modèles non régularisés, le λ est donc nul. On peut remarquer que les estimateurs moyens sont alignés sur la droite joignant l'origine et l'oracle pour le Ridge (c'est bien une droite du fait du design orthogonal) alors que le Lasso a tendance à plus dévaloriser la coordonnée b_2, moins importante dans le modèle de l'oracle car $b_2 < b_1$. Le Lasso pénalise donc plus que le Ridge. De plus, dans le cas du Ridge on perçoit très nettement la régularisation par un étalement des points lorsque le λ se rapproche de 0.

L'échantillonnage choisi sur les coefficients de régularisation permet de considérer que le biais est approximativement le même (qui correspond à la distance euclidienne élevée au carré entre le point $g^\star$ et $f^\star$) entre un estimateur Lasso et un estimateur Ridge pour une couleur donnée. On remarque alors que la variance de l'estimateur Lasso semble être plus importante que celle de l'estimateur Ridge pour un biais donné.

Plus précisément, la figure présente sur la troisième ligne montre que l'estimateur Lasso admet un plus grand risque lorsque la corrélation entre les covariables est élevée, quoique $\rho = 0.5$ soit une corrélation raisonnable.

(c). Cette orthogonalité est intégrée dans les approches classiques afin de connaître l'ordre de grandeur des erreurs rencontrées, comme pour le modèle linéaire pour obtenir l'équation (1.16).

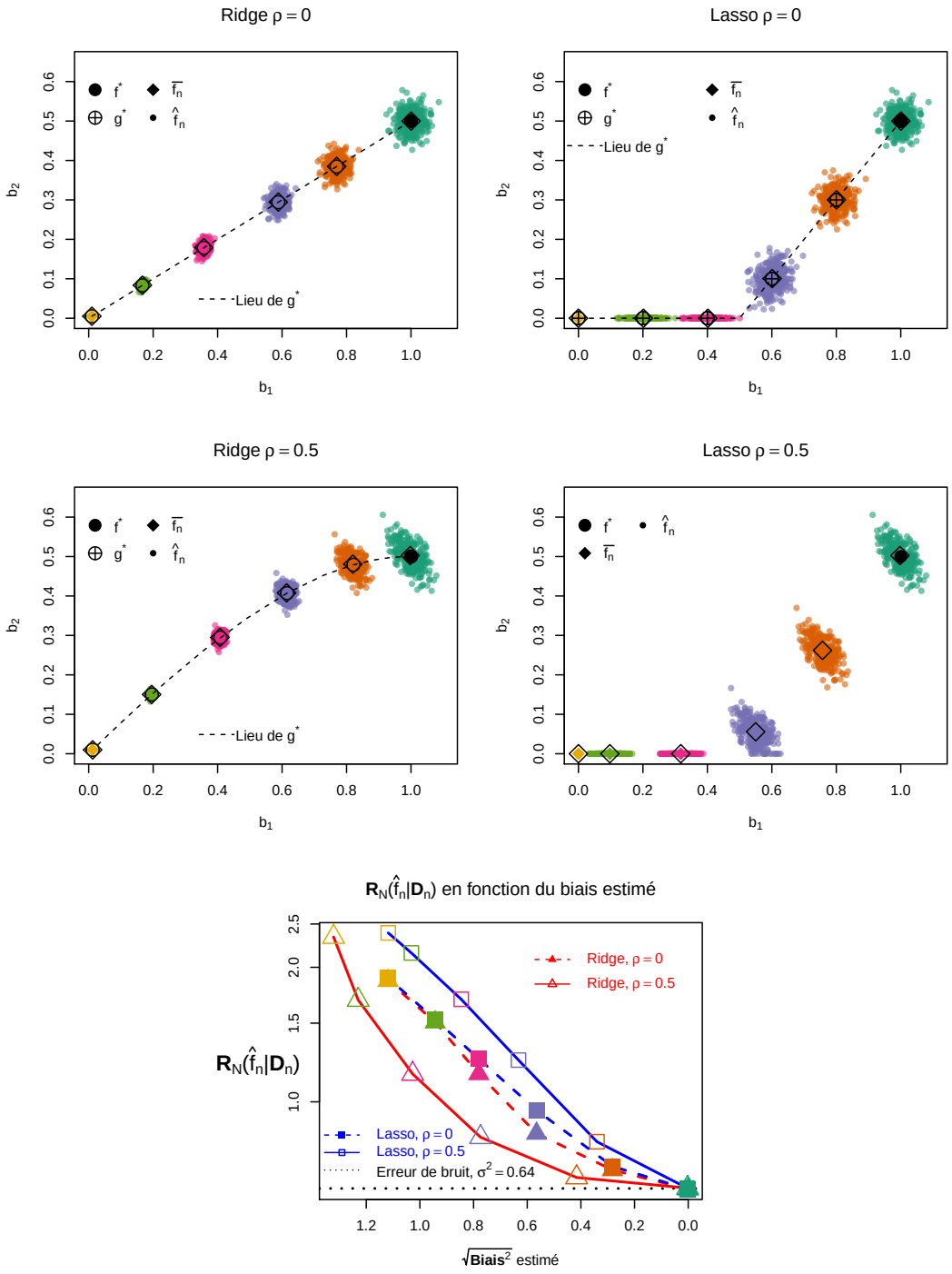


FIGURE 5 – Estimateurs Lasso et Ridge où $f^\star = (1, 0.5)^T$ avec un bruit additif $\mathcal{N}(0, 0.8^2)$ pour $n = 1000$ observations par tirage pour un total de $N = 300$ tirages où $X_1 \sim \mathcal{N}(0,1)$ et $X_2 \sim \mathcal{N}(0,1)$ avec une corrélation théorique $\rho = 0$ pour la première ligne et $\rho = 0.5$ pour la seconde. Chaque couleur correspond à une valeur différente de coefficient de régularisation, différente entre le Lasso et le Ridge. La troisième ligne correspond à la représentation des estimations du risque empirique, $\mathbf{R}_N\left(\hat{f}_n|\mathbf{D}_n\right)$, en fonction de l'estimation du biais quadratique associé, noté $\mathbf{biais}^2_N\left(\hat{f}_n|\mathbf{D}_n\right)$.

L'approche Lasso permet la régularisation du problème par sélection de variables. Les méthodes mises en places sont très performantes d'un point de vue calculatoire et cette méthode est préférée en pratique aux méthodes pas à pas et « brute force ».
Le principal reproche qui est fait à la méthode Lasso est son comportement dans le cas de variables très corrélées. Dans ce cas deux comportements sont à déplorer à savoir 1) l'augmentation de la variance des estimateurs et 2) une mauvaise sélectivité. Le Lasso est quelque part trop sélectif (voir Zhao et Yu, 2006, à ce propos). De plus, cette pénalisation ne s'applique que lorsque la réponse $\mathbf{Y}$ est unidimensionnelle.

La régularisation Elastic-net

Comme énoncé précédemment, le Lasso souffre de la présence de fortes corrélations parmi ses covariables pour ce qui est de l'efficacité en sélection. Afin de résoudre ce problème, Zou et Hastie (2005) ont proposé de coupler la procédure Lasso à la procédure Ridge via un coefficient de régularisation $\alpha \in [0, 1]$ entre les deux fonctionnelles de chacune des deux méthodes

$$\|f\|_{\mathcal{F}} = \alpha \sum_{j=1}^{p} |b_j| + (1-\alpha) \sum_{j=1}^{p} b_j^2,$$

le problème ainsi créé se nomme *Elastic Net* pour la souplesse qu'il introduit dans la pénalisation entre le Ridge et le Lasso.

En effet, cette nouvelle pénalisation permet la sélection de variables dont les différences linéaires sont réduites grâce à la partie ridge. Cette méthode demande la calibration d'un second paramètre, α, en plus du paramètre λ. Cette pénalisation ne s'applique que lorsque la réponse $\mathbf{Y}$ est unidimensionnelle.

Les régularisations PLASM et group-lasso

Un cas courant en analyse de données fait intervenir des groupes de variables. Supposons que le jeu de données considéré ait ses variables compartimentées en T groupes de variables, on peut alors noter chaque vecteur de coefficients correspondant

$$\forall t \in [\![1, T]\!],\ \mathbf{b}_t \in \mathbb{R}^{p_t},$$

où p_t devient naturellement le nombre de variables dans le groupe t. Bakin (1999) a développé et étudié, pendant sa thèse, sur la pénalisation

$$\|f\|_{\mathcal{F}} = \sum_{t=1}^{T} \mathbf{b}_t^T \mathbf{b}_t.$$

cette pénalisation fut nommée Probing Least Absolute Squares Modelling (*PLASM*) par son auteur. Elle est directement associée à la pénalisation de Lasso groupé (*gLasso* pour *group-Lasso*) de Yuan et Lin (2006) qui s'écrit

$$\|f\|_{\mathcal{F}} = \sum_{t=1}^{T} \sqrt{p_t} \sqrt{\mathbf{b}_t^T \mathbf{b}_t}.$$

Ces derniers auteurs ont alors développé un nouvel algorithme de calcul qu'ils ont testé sur des données réelles ainsi qu'au travers de simulations. Cette méthode permet de sélectionner des

groupes de variables en mettant tous les coefficients des groupes non sélectionnés à zéro et ne prend pas en compte la variance des données pour valoriser un groupe par rapport à un autre. Cet algorithme souffre pourtant d'une relative lenteur des calculs et ne peut s'appliquer que lorsque la réponse $\mathbf{Y}$ est unidimensionnelle.

La régularisation sparse group-lasso

Afin d'introduire de la parcimonie à l'intérieur même d'un groupe, Simon et collab. (2013) ont développé la méthode de Lasso groupé parcimonieux (*sgLasso* pour sparse group-Lasso). Le terme de régularisation s'écrit

$$\|f\|_{\mathcal{F}} = (1-\alpha)\sum_{t=1}^{T}\sqrt{\mathbf{b}_t^T\mathbf{b}_t} + \alpha\sum_{t=1}^{T}\|\mathbf{b}_t\|_1,$$

où $\alpha \in [\![0,1]\!]$ permet de régler l'importance de chacun des deux termes. Si $\alpha = 0$ alors nous sommes en présence d'un gLasso et si $\alpha = 1$, on reconnaît le Lasso classique.

Cette pénalisation assure bien la sélection inter-groupes et intra-groupes. Les auteurs proposent des simulation pour étayer cet argument. L'algorithme passe par une descente de gradient, accélérée grâce aux résultats de Nesterov (2013).

1.3.5 Régularisation par méthodes à composantes

La solution de Moore-Penrose a été développée afin de contourner la non inversibilité de la matrice de $\mathbf{X}$ pour le problème des MCO, voir l'équation (1.14). La solution ainsi définie est résolue en projetant sur la totalité de l'espace engendré par $\mathbf{X}$ pourtant, certaines composantes ne sont pas associées au problème d'explication de $\mathbf{Y}$ et les inclure dans le modèle ne semble donc pas être un choix parcimonieux.

Cette section détaille deux méthodes qui permettent de sélectionner dans l'ensemble des composantes de $\mathbf{X}$ celles qui sont les plus adaptées au problème de prédiction de $\mathbf{Y}$.

La Régression sur Composantes Principales (*PCR*)

La méthode de Régression en Composantes Principales (*PCR*), voir Hotelling (1957), consiste à construire un estimateur des MCO sur les R premières directions principales de l'ACP de $\mathbf{X}$.

Ainsi, en supposant que $R \leqslant \text{rang}(\mathbf{X})$, si l'on note $\mathbf{V}$ la matrice réelle $p \times p$ des vecteurs singuliers droits de la matrice $\mathbf{X}$. La projection de $\mathbf{X}$ sur ses R premiers sous-espaces propres peut s'écrire $\mathbf{T}_R = \mathbf{X}\mathbf{V}\mathbf{\Delta}_R$, où $\mathbf{\Delta}_R$ est la matrice réelle $p \times R$ où tous les éléments sont nuls sauf les R sur la diagonale qui sont égaux à 1. On peut alors écrire le modèle linéaire associé à la régression de $\mathbf{Y}$ sur $\mathbf{T}_R$:

$$\mathbf{Y} = \mathbf{T}_R\mathbf{B}_{intermediaire} + \mathbf{E}, \tag{1.23}$$

où $\mathbf{E}$ est l'erreur du modèle d'obsevation avec des observations indépendantes de moyenne nulle et de variance $\sigma_\epsilon^2\mathbb{I}_q$. Comme le rang de $\mathbf{T}_R$ est R par hypothèse, la matrice $\mathbf{T}_R^T\mathbf{T}_R$ est inversible et le modèle (1.23) est estimable par les MCO selon

$$\hat{\mathbf{B}}_{intermediaire} = \left(\mathbf{T}_R^T\mathbf{T}_R\right)^{-1}\mathbf{T}_R^T\mathbf{Y} = \left(\mathbf{\Delta}_R^T\boldsymbol{D}^T\boldsymbol{D}\mathbf{\Delta}_R\right)^{-1}\mathbf{\Delta}_R\boldsymbol{D}\boldsymbol{U}^T\mathbf{Y} = \left(\mathbf{\Delta}_R^T\boldsymbol{D}^T\boldsymbol{D}\mathbf{\Delta}_R\right)^{-1}\mathbf{\Delta}_R^T\boldsymbol{D}\boldsymbol{U}^T\mathbf{Y}.$$

En remarquant que $\mathbf{T}_R\mathbf{B}_{intermediaire} = \mathbf{X}\mathbf{V}\mathbf{\Delta}_R\mathbf{B}_{intermediaire}$, il vient le modèle de régression, équivalent au modèle (1.23), s'écrit

$$\mathbf{Y} = \mathbf{X}\mathbf{B} + \mathbf{E}, \tag{1.24}$$

où $\mathbf{B} = \mathbf{V}\mathbf{\Delta}_R\mathbf{B}_{intermediaire}$ et donc en utilisant l'estimateur des MCO proposé précédemment, il vient

$$\hat{\mathbf{B}}^{(PCR,R)} = \mathbf{V}\mathbf{\Delta}_R\hat{\mathbf{B}}_{intermediaire} = \mathbf{V}\mathbf{\Delta}_R\left(\mathbf{\Delta}_R^T\boldsymbol{D}^T\boldsymbol{D}\mathbf{\Delta}_R\right)^{-1}\mathbf{\Delta}_R^T\boldsymbol{D}\boldsymbol{U}^T\mathbf{Y} = \mathbf{V}\mathbf{D}_{(PCR)}^{\sim 1}\mathbf{U}^T\mathbf{Y},$$

en notant $\mathbf{D}_{(PCR)}^{\sim 1} = \mathbf{\Delta}_R\left(\mathbf{\Delta}_R^T\boldsymbol{D}^T\boldsymbol{D}\mathbf{\Delta}_R\right)^{-1}\mathbf{\Delta}_R^T$. En se référant à la définition de la solution généralisée des MCO proposée au travers du Théorème A.3, il est possible de réécrire la solution des MCO sous la forme

$$\hat{\mathbf{B}}^{(MCO)} = \mathbf{V}\mathbf{D}^{\sim 1}\mathbf{U}^T\mathbf{Y},$$

où l'on peut isoler le terme associé au spectre de $\mathbf{X}$ dans chacun des deux estimateurs, à savoir :

$$\begin{array}{ll} \text{pour les MCO} & : \quad \mathbf{D}^{\sim 1}, \\ \text{pour la PCR} & : \quad \mathbf{D}_{(PCR)}^{\sim 1}. \end{array}$$

Donc la PCR réalise une transformation sur le spectre de $\mathbf{X}$ en créant une nouvelle matrice de mêmes dimensions que la matrice à laquelle elle s'applique et qui retourne les R premiers coefficients diagonaux de cette dernière en forçant les suivants à être nuls.
Pour toute matrice inversible $\mathbf{V}$, on définit la norme matricielle suivante [(d)]

$$\mathbf{X} \to \|\mathbf{X}\|_{\mathbf{V}} = \sqrt{\text{Trace}\left(\mathbf{X}^T\mathbf{V}\mathbf{V}^T\mathbf{X}\right)},$$

et le produit scalaire associé $< \bullet, \bullet >_{\mathbf{V}}$. On peut ainsi évaluer le biais de l'estimateur en se référant au modèle (1.24) :

$$\begin{aligned} \|\hat{\mathbf{B}}^{(MCO)} - \hat{\mathbf{B}}^{(PCR,R)}\|_{\mathbf{V}}^2 &= \|\mathbf{V}\left(\mathbf{D}^{\sim 1} - \mathbf{D}_{(PCR)}^{\sim 1}\right)\mathbf{D}\mathbf{V}^T\mathbf{B}\|_{\mathbf{V}}^2 + \|\mathbf{V}\left(\mathbf{D}^{\sim 1} - \mathbf{D}_{(PCR)}^{\sim 1}\right)\mathbf{U}^T\mathbf{E}\|_{\mathbf{V}}^2 \\ &+ 2 < \mathbf{V}\left(\mathbf{D}^{\sim 1} - \mathbf{D}_{(PCR)}^{\sim 1}\right)\mathbf{D}\mathbf{V}^T\mathbf{B}, \mathbf{V}\left(\mathbf{D}^{\sim 1} - \mathbf{D}_{(PCR)}^{\sim 1}\right)\mathbf{U}^T\mathbf{E} >_{\mathbf{V}}. \end{aligned}$$

Si l'on passe à l'espérance sur la matrice d'erreur $\mathbf{E}$ il vient que le dernier terme s'annule. Le second terme est associé à l'erreur d'estimation, on notera alors que $\mathbb{E}_{\mathbf{E}}\left[\mathbf{E}\mathbf{E}^T\right] = \sigma_\epsilon^2\mathbb{I}_n$ par indépendance des erreurs d'observation dans l'échantillon d'entraînement. Le premier terme quant à lui rend compte de l'approximation de modèle, c'est le biais de l'estimateur. On note alors

$$\mathbb{E}_{\mathbf{E}}\left[\|\hat{\mathbf{B}}^{(MCO)} - \hat{\mathbf{B}}^{(PCR,R)}\|_{\mathbf{V}}^2\right] = \|\left(\mathbf{D}^{\sim 1} - \mathbf{D}_{(PCR)}^{\sim 1}\right)\mathbf{D}\mathbf{V}^T\mathbf{B}\|_{\mathbf{V}}^2 + \sigma_\epsilon^2\|\left(\mathbf{D}^{\sim 1} - \mathbf{D}_{(PCR)}^{\sim 1}\right)\mathbf{U}^T\|_{\mathbf{V}}^2, \tag{1.25}$$

on remarque que ces termes ont un interprétation semblable aux termes d'erreur d'estimation et d'erreur d'approximation du risque quadratique, voir la sous-section 1.2.4 à ce propos et la Remarque 1.4 qui suit. Il est alors possible de décomposer plus en avant chaque terme mais nous en resterons là. On remarque que la PCR a un effet sur l'erreur d'approximation et sur l'erreur d'estimation.

(d). Nous laissons le soin au lecteur de démonter que cet fonction est bien une norme.

On dit que la méthode PCR shrink le spectre de $\mathbf{X}$, c'est un effet de la régularisation. Cet estimateur a été très utilisé lorsque les dimensions des données étaient raisonnables, aujourd'hui il ne serait pas déraisonnable d'imaginer un cas de figure où la condition de rang formulée plus haut ne pourrait pas être activée sans que l'information dans $\mathbf{X}$ associée à $\mathbf{Y}$ ne soit décrite. C'est principalement pour cette raison que, dans la pratique, cette méthode n'est pas utilisée dans le contexte de la grande dimension.

Remarque 1.4 (Lien avec le risque quadratique.)**.** *Le calcul qui vient d'être effectué, ainsi que celui qui est présenté dans la sous-section 1.3.6, permet de majorer le biais des estimateurs régularisés. Ce travail est à rapprocher de ce qui a été montré dans le cas du calcul de l'erreur de biais*^{2} *associé au risque quadratique pour l'estimateur Ridge, voir Annexe B.2.*
Par rapport au calcul du risque effectué plus tôt, nous avons alors étudié une seconde dimension d'aléa, celle qui est associé à la prédiction sur un nouvel échantillon et qui amenait un troisième terme dans l'équation finale correspondant à l'erreur de bruit. Cette remarque fait écho à ce qui a été décrit dans la Remarque 1.2 et sera reprise dans la sous-section 1.3.6.

La Régression PLS

Cette partie étudie plus en détail le modèle PLS, pour *Partial Least Squares* signifiant Moindres Carrés Partiels en opposition à la solution des MCO, qui permet de gérer la grande dimension dans des problèmes supervisés avec une réponse $\mathbf{Y}$ multidimensionnelle.

Les matrices de covariance sont essentielles dans l'étude des liens linéaires qui existent entre groupes de variables décrivant les mêmes individus. Nous avons par exemple observé en section 1.3.4 que la régularisation Ridge équivaut à un modification de la matrice de covariance dans l'écriture de la solution. La solution du problème des Moindres Carrés Ordinaire, dans le cas non dégénéré, s'écrit comme présenté dans l'équation (1.14), soit le produit de l'inverse d'une matrice de covariance par une autre matrice de covariance.

L'inversion matricielle est problématique dans le cas de problèmes mal posés, nous allons maintenant présenter un modèle qui permet de ne pas réaliser cette inversion pour produire un modèle de régression en s'appuyant sur l'étude des matrices de covariance. Ces modèles s'appellent les modèles PLS.

La méthode PLS a été introduite par Wold au début des années 90. Wold et collab. (2001) réalisent un état de l'art des nombreux ingrédients théoriques et pratiques qui ont été apportés à cette méthode. L'algorithme mis en jeu construit séquentiellement plusieurs axes principaux en commençant par celui qui explique un maximum du lien entre les matrices $\mathbf{X}$ et $\mathbf{Y}$. R axes sont ainsi construits, où R est fixé par l'utilisateur. La définition de chacun de ces axes passe par la résolution d'un problème d'optimisation, le problème (1.27) que nous décrirons plus loin, qui est à chaque fois appliqué à un jeu de données égal au jeu de données correspondant à la composante précédente mais diminué de l'information de cette dernière. Cette diminution est appelée "déflation" et les matrices résultantes sont appelées matrices déflatées.

Cette méthode recherche, dans la matrice de covariance $\mathbf{X}^T\mathbf{Y}$, les R directions de plus forte occurence. C'est une adaptation de l'ACP au cas supervisé de l'analyse de données. On dit aussi que la PLS recherche les sous-espaces de forte représentativité dans $\mathbf{X}$ et dans $\mathbf{Y}$ séparément

mais aussi dans $\mathbf{X}^T\mathbf{Y}$.

La PLS construit un modèle de régression multidimensionnel (p et q peuvent être supérieurs à 1) et fut initialement proposée pour résoudre des problèmes de chimiométrie dans lesquels un bloc de covariables $\mathbf{X}$ doit expliquer un bloc $\mathbf{Y}$ dans lesquels plusieurs dimensions semblent être partagées.

Bien que la PLS soit bien connue depuis maintenant plusieurs décennies, nous avons remarqué qu'il était difficile de discerner les différents paramètres de cette analyse. Nous avons donc pris la liberté de l'ordre dans lequel les différents paramètres du modèle sont présentés à commencer par

— les *scores* dont l'importance est primordiale puisqu'ils concentrent l'information recherchée pour chaque axe.

— La *déflation* doit être présentée en second puisqu'elle permet de passer d'un axe à un second en évitant toute redondance, elle équivaut donc à un procédé d'orthonormalisation sur les scores.

— Les *poids* (*loadings*, en anglais) permettent ensuite de construire chaque composante sur les sous-espaces orthogonaux aux sous-espaces précédents. Il convient ici de remarquer que nous avons pris la liberté de définir deux types de poids, les *poids classiques* et les *poids spéciaux*. Les premiers étant associés aux matrices transformées par les déflations successives et les seconds aux matrices initiales.

— Le problème d'optimisation est finalement introduit, celui-là même qui est classiquement présenté en premier.

— Enfin le modèle de régression qui passe de la définition des scores à la description matricielle, soit la matrice $\mathbf{B}$ dans le modèle hypothétique $\mathbf{Y} \approx \mathbf{XB}$.

L'aspect régularisant de la PLS est cité et une introduction à la méthode des moindres carrés partiels parcimonieux (*sPLS*) est faite.

Les *scores* On notera par la suite R le nombre de dimensions différentes que l'algorithme va chercher à décrire. Les deux blocs de données sont projetés sur ces R dimensions pour former R différents *scores* décrivant à chaque fois les n individus, les notations sont alors

$$\begin{array}{rcl} \text{scores du bloc } \mathbf{X} & : & \left\{\mathbf{t}^{(1)}, \cdots, \mathbf{t}^{(R)}\right\}, \\ \text{scores du bloc } \mathbf{Y} & : & \left\{\mathbf{s}^{(1)}, \cdots, \mathbf{s}^{(R)}\right\}. \end{array}$$

Les matrices résultantes $\mathbf{T} = \left[\mathbf{t}^{(1)}, \cdots, \mathbf{t}^{(R)}\right] \in \mathbb{R}^{n\times R}$ et $\mathbf{S} = \left[\mathbf{s}^{(1)}, \cdots, \mathbf{s}^{(R)}\right] \in \mathbb{R}^{n\times R}$. Pour chaque bloc, les scores sont construits sur les résidus des blocs, eux-mêmes résiduels, dans la régression sur le score du bloc $\mathbf{X}$ pour une dimension $r \in [\![1, R]\!]$ donnée $\mathbf{t}^{(r)}$. Ainsi nous avons le résultat de normalité $\forall (r, r') \in [\![1, R]\!]^2 | r \neq r',\ {\mathbf{t}^{(r)}}^T \mathbf{t}^{(r')} = 0$.

La *déflation* Les informations décrites sur chaque score doivent être orthogonales pour le bloc $\mathbf{X}$. Afin de réaliser ceci, il a été choisi d'utiliser la méthode de *déflation*, qui met à jour les blocs $\mathbf{X}$ et $\mathbf{Y}$ pour le rang r. On note alors, $\forall r \in [\![1, R]\!]$, $\mathbf{X}^{(r)}$ et $\mathbf{Y}^{(r)}$ les matrices résiduelles sur

lesquelles sont construits les scores $\mathbf{t}^{(r)}$ et $\mathbf{s}^{(r)}$. Avec pour convention que $\mathbf{X}^{(1)} = \mathbf{X}$ et $\mathbf{Y}^{(1)} = \mathbf{Y}$.

La PLS cherchant à créer un modèle de régression multiple de $\mathbf{Y}$ sur $\mathbf{X}$, il a été choisi comme convention d'utiliser une déflation sur les scores de $\mathbf{X}$, soit $\forall r \in [\![1, R]\!]$, le score $\mathbf{t}^{(r)}$ permet de passer de $\mathbf{X}^{(r)}$ à $\mathbf{X}^{(r+1)}$ et de $\mathbf{Y}^{(r)}$ à $\mathbf{Y}^{(r+1)}$. On définit alors

$$\text{les matrices de régression, } \forall r \in [\![1, R]\!], \begin{cases} \text{de } \mathbf{X}^{(r)} \text{ sur } \mathbf{t}^{(r)} & : \mathbf{b}_X^{(r)} = {\mathbf{t}^{(r)}}^T \mathbf{X}^{(r)} / {\mathbf{t}^{(r)}}^T \mathbf{t}^{(r)}, \\ \text{de } \mathbf{Y}^{(r)} \text{ sur } \mathbf{t}^{(r)} & : \mathbf{b}_Y^{(r)} = {\mathbf{t}^{(r)}}^T \mathbf{Y}^{(r)} / {\mathbf{t}^{(r)}}^T \mathbf{t}^{(r)}, \end{cases}$$

qui sont des matrices ligne. obtenues en minimisant les moindres carrés ordinaires sur les modèles de régression linéaires suggérés. Dès lors on définit

$$\text{les équations de déflation, } \forall r \in [\![1, R-1]\!], \begin{cases} \mathbf{X}^{(r+1)} = \mathbf{X}^{(r)} - \mathbf{t}^{(r)} \mathbf{b}_X^{(r)}, \\ \mathbf{Y}^{(r+1)} = \mathbf{Y}^{(r)} - \mathbf{t}^{(r)} \mathbf{b}_Y^{(r)}. \end{cases} \tag{1.26}$$

Les *poids* Il existe deux types de *poids* dans la PLS, $\forall r \in [\![1, R]\!]$, qui vont chacun conduire à la création du score $\mathbf{t}^{(r)}$. Pourtant ils ne s'appliquent pas aux mêmes objets, nous définissons les poids dits *classiques* et les poids dits *spéciaux*. Cette partie décrit ces deux familles de poids.

— **Les *poids classiques*** : Ces poids, $\forall r \in [\![1, R]\!]$, sont dérivés de chacun des blocs courants $\mathbf{X}^{(r)}$ tels que

$$\begin{aligned} \mathbf{t}^{(r)} &= \mathbf{X}^{(r)} \mathbf{u}^{(r)} \\ \mathbf{s}^{(r)} &= \mathbf{Y}^{(r)} \mathbf{v}^{(r)} \end{aligned}.$$

Ce sont donc les deux familles de vecteurs de dimensions p et q respectivement notées $\left(\mathbf{u}^{(r)}\right)_{r \in [\![1,R]\!]}$ et $\left(\mathbf{v}^{(r)}\right)_{r \in [\![1,R]\!]}$.

— **Les *poids spéciaux*** : Ces poids, $\forall r \in [\![1, R]\!]$, sont dérivés de chacun des blocs initiaux $\mathbf{X}$ tels que

$$\mathbf{t}^{(r)} = \mathbf{X} \mathbf{u}^{\star(r)}.$$

On ne définit ici qu'une seule famille de vecteurs p-dimensionnels $\left(\mathbf{u}^{\star(r)}\right)_{r \in [\![1,R]\!]}$.

l'équation (1.26) permet, par application d'un raisonnement par récurrence, de démontrer la relation suivante qui lie chacun des $\mathbf{u}^{\star(r)}$ aux autres paramètres du modèle. Soit donc

$$\forall r \in [\![1, R]\!],\ \mathbf{u}^{\star(r)} = \left(\mathbb{I} - \mathbf{u}^{(1)} \mathbf{b}_X{}^{(1)}\right) \cdots \left(\mathbb{I} - \mathbf{u}^{(r)} \mathbf{b}_X{}^{(r)}\right) \mathbf{u}^{(r)}$$

Les poids spéciaux sont essentiels pour construire un modèle de prédiction généralisable à un nouveau jeu de données (voir section 1.3.5) alors que les poids classiques sont issus du problème d'optimisation que nous allons voir maintenant.

Les R problèmes d'optimisation $\forall r \in [\![1, R]\!]$, le problème d'optimisation est le suivant

$$\mathbf{u}^{(r)} = \underset{\mathbf{u}\in\mathbb{R}^p,\ \mathbf{u}^T\mathbf{u}=1}{\arg\max} \left\| {\mathbf{Y}^{(r)}}^T \mathbf{X}^{(r)} \mathbf{u} \right\|^2, \tag{1.27}$$

qui peut être résolu en utilisant le multiplicateur de $\mathcal{L}$agrange (voir Théorème A.1) tel que réalisé dans la preuve du Théorème A.4 (nous laisserons les lecteurs intéressés consulter l'annexe A associée à ces résultats). Il est ainsi possible de définir $\mathbf{v}^{(r)}$ tel que

$$\mathbf{v}^{(r)} = \begin{cases} \dfrac{{\mathbf{Y}^{(r)}}^T \mathbf{X}^{(r)} \mathbf{u}^{(r)}}{\left\| {\mathbf{Y}^{(r)}}^T \mathbf{X}^{(r)} \mathbf{u}^{(r)} \right\|} & \text{si} \quad \left\| {\mathbf{Y}^{(r)}}^T \mathbf{X}^{(r)} \mathbf{u}^{(r)} \right\| \neq 0 \\ 0 & \text{sinon} \end{cases}$$

La recherche des R sous-espaces associés aux R problèmes (1.27) est historiquement résolue par à un algorithme alterné dit des moindres carrés partiels non linéaires itératifs (*NIPALS*) (voir Wold et collab., 1989). Cette méthode se base sur la méthode de la puissance itérée (aussi appelé algorithme de Von Mises et détaillé dans l'Annexe C, page 189) qui permet de diagonaliser une matrice semi-définie positive en mettant en évidence les valeurs propres et vecteurs propres associés. Il fonctionne de façon séquentielle en commençant par le sous-espace associé à la valeur propre qui a la plus grande amplitude. Dans le cas de la PLS, seul le premier espace est construit pour chaque composante.

Le modèle de régression La famille des R scores $\left\{\mathbf{t}^{(1)}, \cdots, \mathbf{t}^{(R)}\right\}$ ainsi construite est orthogonale et définit un sous-espace fortement associé à $\mathbf{Y}$, c'est donc naturellement que $\mathbf{Y}$ est décomposé sur ce sous-espace avec

$$\mathbf{Y} = \mathbf{t}^{(1)}\mathbf{b}_Y^{(1)} + \cdots + \mathbf{t}^{(R)}\mathbf{b}_Y^{(R)} + \mathbf{E},$$

où $\mathbf{E}$ est un terme orthogonal au sous-espace engendré par la famille des scores. En réécrivant cette dernière équation et en vertu de la définition des *poids spéciaux*, il vient :

$$\begin{array}{rcll} \mathbf{Y} & = & \mathbf{X}\mathbf{u}^{\star(1)}\mathbf{b}_Y^{(1)} + \cdots + \mathbf{X}\mathbf{u}^{\star(R)}\mathbf{b}_Y^{(R)} & + \mathbf{E}, \\ & = & \mathbf{X}\sum_{r=1}^{R} \mathbf{u}^{\star(r)}\mathbf{b}_Y^{(r)} & + \mathbf{E}. \end{array}$$

On note alors $\mathbf{B}^{(r)} = \mathbf{u}^{\star(r)}\mathbf{b}_Y^{(r)}$, et $\mathbf{B} = \sum_{r=1}^{R} \mathbf{u}^{\star(r)}\mathbf{b}_Y^{(r)}$ et il est possible d'écrire le modèle de régression suivant

$$\mathbf{Y} = \mathbf{X}\mathbf{B} + \mathbf{E},$$

où $\mathbf{E}$ doit être de norme minimale, ce qui est laissé à l'appréciation de l'utilisateur et réglé grâce au paramètre R.

Un modèle de régularisation La méthode PLS, telle que décrite précédemment, ne dépend que du paramètre R qui correspond au nombre de *composantes* à construire. C'est un paramètre entier positif non nul. L'algorithme a donc tendance à ajouter de l'information du jeu de données initial lorsque le coefficient R grandit, c'est la définition de la *régularisation*. Pourtant la démonstration de ce résultat n'a pas été aisée et il a fallu attendre les travaux réalisés par De Jong (1995) pour en avoir une preuve théorique formelle qui se base sur des arguments géométriques élémentaires et élégants.

Le Modèle PLS Parcimonieux et à Variables Groupées Le modèle PLS a été présenté dans les sections précédentes, il a aussi été vu que ce modèle permet de réduire les coefficients afin d'affiner la description de la structure des données, c'est la régularisation. Pourtant cet aspect nécessaire de l'analyse de données de grande dimension ne permet pas la sélection des variables engagées dans le modèle construit. La régularisation Lasso, dans le contexte du problème MCO, permet la sélection de variables. L'esprit de cette régularisation a été, pour la première fois, transposé à l'analyse en composante principale au travers de la méthode Simplified Component Technique-Lasso (*SCoTLASS*) proposée par Jolliffe et collab. (2003) qui permet de sélectionner les variables les plus associées à chaque composante, par composante. La PLS obtient sa première version parcimonieuse grâce aux travaux de Lê Cao et collab. (2008) en reprenant la méthode SCoTLASS, c'est la méthode sPLS. Cette méthode permet de fixer le nombre de variables à sélectionner par composante du côté des covariables mais aussi du côté de la réponse, à supposer que la réponse soit multidimensionnelle. Cette solution est implémentée dans le package R nommé mixOmics (Lê Cao et collab., 2009) et est téléchargeable depuis le CRAN.

Zou et collab. (2006) ont montré, ce qui a été repris ensuite par Chun et Keleş (2010), que la solution SCoTLASS n'est pas assez parcimonieuse, il est aussi souvent pointé du doigt que cette solution n'est pas assez convexe. Afin de proposer une solution plus parcimonieuse, Chun et Keleş (2010) proposent un nouveau modèle, plus parcimonieux. Ce dernier ajoute un paramètre qui permet de valoriser la partie convexe du critère d'optimisation tout en conservant sa sélectivité.

Puisque les deux méthodes décrites ici sont assez similaires, nous avons décidé de présenter la première historiquement (soit celle développée par Lê Cao et collab., 2008). Le critère d'optimisation

$$\left(\mathbf{u}^{(r)}, \mathbf{v}^{(r)}\right) = \underset{(\mathbf{u},\mathbf{v})\in\mathbb{R}^p\times\mathbb{R}^q,\ \mathbf{u}^T\mathbf{u}=\mathbf{v}^T\mathbf{v}=1}{\arg\min} \left|\left|\mathbf{Y}^{(r)^T}\mathbf{X}^{(r)} - \mathbf{v}\mathbf{u}^T\right|\right|_F^2 + \lambda_x||\mathbf{u}||_1 + \lambda_y||\mathbf{v}||_1,$$

remplace celui décrit dans l'équation (1.27), où les coefficients λ_x et λ_y sont fixés par l'utilisateur et permettent de choisir le nombre de variables à sélectionner dans le modèle ainsi construit. Les étapes suivantes, à savoir la déflation et la création du modèle de régression, sont les mêmes que dans le cas de la PLS classique.

Il existe d'autres dérivées de la méthode PLS, comme les méthodes des moindres carrés partiels groupés (*gPLS*) ou des moindres carrés partiels parcimonieux groupés (*sgPLS*) introduites par Liquet et collab. (2015) mais la description de ce genre de méthode nécessite l'introduction aux méthodes multiblocs, ce qui est réalisé dans la section suivante

1.3.6 Rétrécissement (*shrinkage*) et régularisation par seuillage

Dans les développements précédents, nous avons rencontré plusieurs méthodes de régularisation dont les plus connues sont le Ridge et le Lasso [(e)]. Ces méthodes rétrécissent leurs estimateurs (« *shrink* » en anglais) qui permet de stabiliser le risque quadratique, en plus de rendre les problèmes d'optimisation solvables.

(e). Mais aussi la PCR, voir la sous-section 1.3.5 pour une démonstration.

Dans le cas du Lasso, en considérant un design orthogonal, le rétrécissement est observé au travers de l'équation (1.22) et la fonction élémentaire de rétrécissement est la fonction de seuillage doux, notée s_λ pour un seuil λ, plus largement détaillée dans les sections suivantes. On peut alors noter

$$\text{l'estimateur du Lasso pour un design orthogonal } \mathbf{b}_{n,ortho}^{(Lasso)} = s_\lambda\left(\mathbf{b}_n^{(MCO)}\right). \tag{1.28}$$

Dans le cas du Ridge, en utilisant l'estimateur tel que décrit dans l'équation (1.19) que l'on peut mettre sous la forme

$$\hat{\mathbf{B}}_n^{(Ridge)} = \mathbf{V}\left(\mathbf{D}^T\mathbf{D}/n + \lambda\mathbb{I}\right)^{-1}\mathbf{D}^T\mathbf{U}^T\mathbf{Y}/n,$$

où les matrices $(\mathbf{U},\mathbf{D},\mathbf{V})$ sont définies dans le Corollaire A.1 ainsi que $\mathbf{D}^{\sim 1}$ afin de construire la pseudo-inverse de Moore-Penrose de $\mathbf{X}$: $\mathbf{X}^+ = \mathbf{V}\mathbf{D}^{\sim 1}\mathbf{U}^T$. Cette dernière matrice est aussi utilisée dans l'équation (1.14) pour décrire la solution généralisée des MCO $\hat{\mathbf{B}}_n^{(MCO)} = \mathbf{V}\mathbf{D}^{\sim 1}\mathbf{U}^T\mathbf{Y}$. Comme réalisé dans la sous-section 1.3.5, il est possible d'estimer le biais de l'estimateur du Ridge et ainsi

$$\mathbb{E}_{\mathbf{E}}\left[\|\hat{\mathbf{B}}^{(MCO)} - \hat{\mathbf{B}}^{(Ridge)}\|_{\mathbf{V}}^2\right] = \|\left(\mathbf{D}^{\sim 1} - \mathbf{D}_{(Ridge)}^{\sim 1}\right)\mathbf{D}\mathbf{V}^T\mathbf{B}\|_{\mathbf{V}}^2 + \sigma_\epsilon^2\|\left(\mathbf{D}^{\sim 1} - \mathbf{D}_{(Ridge)}^{\sim 1}\right)\mathbf{U}^T\|_{\mathbf{V}}^2, \tag{1.29}$$

où l'on a noté $\mathbf{D}_{(Ridge)}^{\sim 1} = \left(\mathbf{D}^T\mathbf{D}/n + \lambda\mathbb{I}\right)^{-1}\mathbf{D}^T/n$. Cette décomposition est similaire dans sa structure à celle décrite pour l'estimateur PCR. Dans son comportement, le coefficient λ étant continue, cette régularisation offre plus de possibilités de régularisation. Tout comme pour le cas PCR, voir l'équation (1.25), l'équation (1.29) montre que l'estimateur Ridge opère un seuillage sur le spectre de $\mathbf{X}$. C'est un cas de « *shrink* » classique en régularisation. L'absence de condition sur le rang de la matrice $\mathbf{X}$ à l'emploi de cette méthode

Il existe plusieurs seuillages utilisés pour régulariser les modèles, les deux sous-sections qui suivent décrivent les méthodes associées à ce genre d'opérateur ainsi que la capacité de ces méthodes à reconstruire l'information désirée.

Opérateurs généralisés

D'autres types de régularisation ont été étudiés, par exemple Johnstone et Lu (2004) définissent l'Analyse en Composante Principale Parcimonieuse (*sparseACP*). La solution sparseACP décrite par Johnstone et Lu (2004) repose sur la résolution de l'ACP et passe par la construction d'une base orthogonale des vecteurs propres de $\mathbf{X}^T\mathbf{X}$, cette solution nécessite le seuillage des éléments propres avant projection des valeurs résultantes sur les sous-espaces propres correspondants. Cette méthode utilise le seuillage dur qui, pour un paramètre $\lambda > 0$ fixé par l'utilisateur, annule les valeurs qui lui sont inférieures en valeur absolue et laisse inchangées les autres, voir figure 6.

Cette méthode induit une parcimonie sur la base de projections choisies par l'opérateur. L'hypothèse est donc de considérer qu'un phénomène avec une faible représentativité sur l'espace de projection est non informationnel et doit être supprimé. D'autres fonctions de seuillage

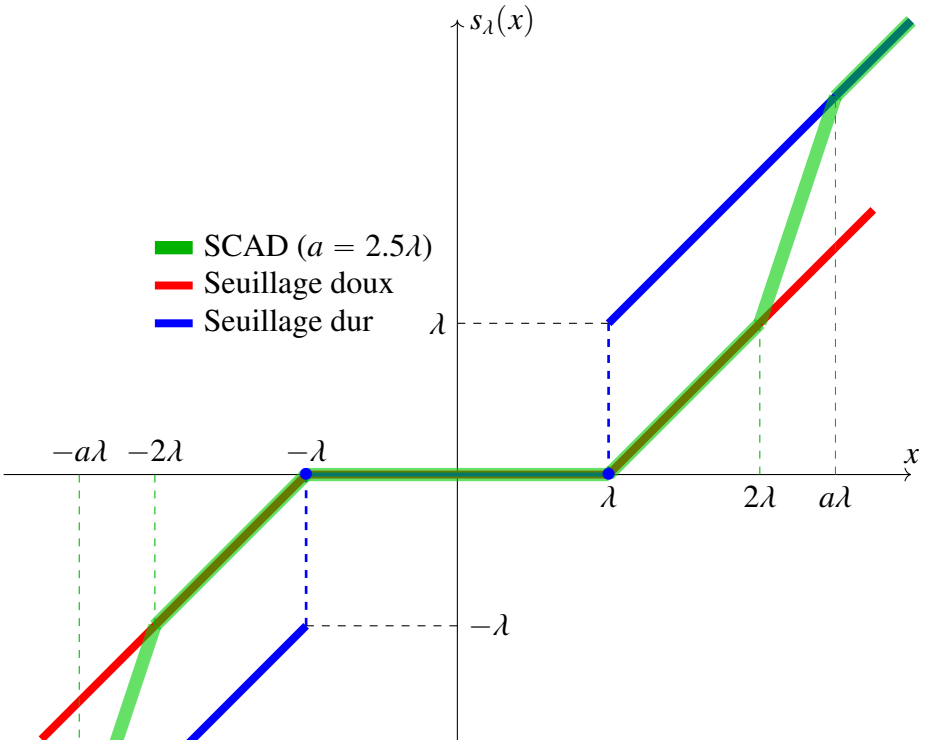


FIGURE 6 – Différents seuillages

ont été étudiées, par exemple par ROTHMAN et collab. (2009), la classe des opérateurs de seuillage appelée "*Generalized Thresholding Operators*" est ainsi décrite. Elle admet certaines propriétés qui sont, en notant $\lambda > 0$ le paramètre de seuillage, $s_\lambda : \mathbb{R} \to \mathbb{R}$ une fonction de cette classe et $\forall x \in \mathbb{R}$:

— $|s_\lambda(x)| \leqslant |x|$,
— $|s_\lambda(x)| = 0 \; \forall \; |x| < \lambda$,
— $|s_\lambda(x) - x| \leqslant \lambda$.

Cette famille contient le seuillage dur, déjà rencontré et visible en bleu sur la figure 6, mais aussi le seuillage doux, visible en rouge sur la figure 6. Ce dernier admet une propriété de continuité, il apparaît dans l'écriture de la différentiation du problème de pénalisation Lasso. On voit sur cette courbe que le seuillage doux ne permet pas une bonne reconstruction des coefficients dont la valeur absolue est supérieure à λ, par contre il permet une forte régularisation pour les faibles coefficients (puisque $\forall |x| \leqslant \lambda, \; s_\lambda(x) = 0$). Des solutions ont été pensées pour rattraper ce défaut dont le seuillage dur, mais ce dernier admet une discontinuité en $|x| = \lambda$ et une régularisation nulle $\forall |x| > \lambda$. D'autres seuillages de la forme "*Generalized Thresholding Operators*" ont été étudiés qui soient de classe C^0 au moins et qui montrent une « bonne » reconstruction des grands coefficients. Parmi eux le seuillage Smoothly Clipped Absolute Deviation Penalty (*SCAD*), visible en vert sur la figure 6 et paramétré par deux coefficients (a, λ) où $a > 2$ permet d'ajuster la douceur de transition d'une zone de découpe ($\forall |z| < \lambda$) à une zone où le signal ne doit pas être modifié ($\forall |z| > a\lambda$). Cette méthode est particulièrement adaptée lorsqu'une erreur importante bruite le signal mais que ce dernier est discernable au moyen d'une méthode d'analyse harmonique par exemple, l'auteur cite un travail réalisé sur des transformés en ondelettes, voir FAN (1997) à ce propos. Le seuillage Lasso adaptatif (*aLasso*) permet aussi ce genre de fenêtrage sur les coefficients, voir ZOU (2006). L'idée est alors de pénaliser le seuillage de chaque coefficient

par une valeur fournie par l'utilisateur avec une plus forte pénalisation pour les coefficients les plus faibles. Il est souvent conseillé d'utiliser comme pondération la valeur du coefficient de régression obtenu par les MCO. Ces deux dernières méthodes permettent de retirer l'erreur de reconstruction du Lasso pour les grands coefficients, du au seuillage doux, mais pénalisent de façon continue l'ensemble des plages de valeurs de coefficients.

Consistance en grande dimension

Comme développé en section 1.3, la régularisation permet de décrire des structures de données dans le contexte de la grande dimension. Alors, l'estimateur de la matrice de covariance empirique est un mauvais estimateur de la matrice de covariance populationnelle (voir Bickel et collab., 2008, par exemple). La consistance du problème de l'ACP a été étudiée en proposant de résoudre

$$\text{le problème sparseACP de rang 1 noté} \quad \underset{\mathbf{u}\in\mathbb{R}^p,\ s.t.\ \|\mathbf{u}\|_2=1,\ \|\mathbf{u}\|_0\leqslant k}{\arg\max} \quad \mathbf{u}^{\mathbf{T}}\hat{\mathbf{\Sigma}}\mathbf{u}, \tag{1.30}$$

où k est la cardinalité maximale de l'ensemble des variables à coefficients non nuls et $\hat{\mathbf{\Sigma}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i\mathbf{x}_i{}^T$ est la matrice de covariance empirique si les individus sont centrés.

Parmi les méthodes développées pour répondre à ce problème, la décomposition SVD de la matrice de covariance seuillée sur sa diagonale a montré d'excellentes performances (voir Johnstone et Lu, 2009), méthode appelée seuillage de la diagonale (*DT*). Un algorithme, développé par d'Aspremont et collab. (2005), utilise la relaxation du problème (1.30) réécrit en considérant l'ensemble des matrices semi-définies positives, algorithme dit de programmation positive (*SDP*). Le travail réalisé par Krauthgamer et collab. (2015) montre que la solution de d'Aspremont et collab. (2005) ne parvient pas, dans l'hypothèse d'une matrice de rang unitaire, à reconstruire l'information recherchée. Ils montrent aussi que l'algorithme DT, quoique beaucoup plus simple que celui développé par d'Aspremont et collab. (2005), montre des performances similaires.

Partant de ce constat, les auteurs proposent de tester les performances de l'algorithme qui consiste à faire la SVD de la matrice de covariance seuillée, algorithme nommé seuillage de covariance (*CT*), et proposé initialement par Bickel et collab. (2008). Contrairement à l'algorithme DT, tous les éléments de la matrice sont ici seuillés. Cette méthode présente une complexité algorithmique plus faible que celle de d'Aspremont et collab. (2005), Krauthgamer et collab. (2015) montrent son intérêt sur des exemples pratiques et Deshpande et Montanari (2016) apportent des preuves théoriques de la consistance de cette méthode, dans l'hypothèse non plus de rang unitaire mais de rang $R \geqslant 1$ de la matrice de covariance même dans le cas dégénéré de la grande dimension où $n << p$.

1.4 Choix de modèle

L'analyste a accès à un jeu de données $\mathcal{D}_n$ et aux fonctions candidates, c'est à lui de choisir le meilleur modèle. Le compromis entre biais et variance, détaillé dans la partie précédente, doit se faire sans pouvoir accéder à de nouveaux échantillons et sans donc pouvoir obtenir d'erreur

de test sur un échantillon indépendant.

Nous avons vu que le risque empirique permettait un choix de modèle mais il peut y avoir des contraintes à son utilisation, notamment lors du calcul du biais, voir section 1.2.5. De plus, en fonction du contexte des données, les dimensions des données, la question associée, il peut être intéressant de pénaliser certains aspects du modèle plutôt que d'autres, comme un critère de sélection de variables dans le cas d'une problématique de sélection. Pour toutes ces raisons, des critères autres que biais ont été introduits et la première section présente certains d'entre eux.

De même, puisqu'il n'est pas possible d'avoir accès à d'autres échantillons que l'échantillon d'entraînement, une réalisation de $\mathcal{D}_n$, il n'est pas possible d'obtenir l'erreur d'estimation de chaque modèle. L'analyste a recours à des simulations d'échantillonage afin de recréer ce processus, nous appelons ceci le rééchantillonnage. La seconde section détaille les méthodes les plus utilisées à cette fin.

1.4.1 Critères de choix

Dans cette section nous supposerons Y est une variable aléatoire et non plus un vecteur aléatoire. Le problème est donc un problème de régression unidimensionnel.

Le coefficient de détermination

C'est le premier critère utilisé historiquement, on l'appelle aussi R^2. Il s'applique à l'étude des estimateurs du modèle linéaire. Pour un modèle linéaire dont on estime la matrice de régression $\hat{\mathbf{b}}_n$, on définit trois grandeurs

— la Somme des Carrés des Résidus (*SCR*), qui est la somme des carrés des résidus correspondant à la variance qui n'est pas expliquée par le modèle :

$$\begin{aligned} SCR &= \left\| f(\mathbf{X}) - \hat{f}_n(\mathbf{X}) \right\|_2^2 \\ &= \left\| \mathbf{Y} - \hat{\mathbf{Y}} \right\|_2^2 . \end{aligned}$$

— La Somme des Carrés Totale (*SCT*), qui est la somme des carrés des variations des observations de Y autour de sa moyenne correspondant à la variance totale des observations :

$$SCT = \left\| \mathbf{Y} - \bar{y}\mathbf{1}_n \right\|^2 ,$$

avec $\bar{y} = \sum_{i=1}^n y_i$ et $\mathbf{1}_n$ est le vecteur de taille n formé uniquement de 1.

— La Somme des Carrés Expliqués (*SCE*), qui est la somme des carrés des variations des estimations de Y autour de la moyenne des observations et correspondant à la variance totale expliquée :

$$SCE = \left\| \hat{\mathbf{Y}} - \bar{y}\mathbf{1}_n \right\|^2 .$$

On remarque alors que

$$SCT = SCR + SCE,$$

et l'on définit

$$\text{le coefficient de détermination par } R^2 = \frac{SCE}{SCT} = 1 - \frac{SCR}{SCT} = 1 - \frac{\left\| \mathbf{Y} - \hat{\mathbf{Y}} \right\|_2^2}{\left\| \mathbf{Y} - \bar{y}\mathbf{1}_n \right\|^2}.$$

Lorsque sa valeur est proche de 0, on dit que le modèle est mauvais et lorsqu'il est proche de 1 l'inverse. Pour interprétation, supposons que les observations de la réponse sont centrées et réduites, ce qui n'est pas une forte hypothèse, dans ce cas il vient $R^2 = 1 - SCR/n$. On remarque aussi que, dans le cas de l'estimateur des MCO,

$$SCR \quad = \quad \|\mathbf{Y} - \hat{\mathbf{Y}}\|_2^2 \quad = \quad \|\mathbf{Y} - \mathbf{X}\hat{\mathbf{b}}_n^{(MCO)}\|_2^2 \quad = \quad \|\left(\mathbb{I} - \mathbf{X}\mathbf{X}^+\right)\mathbf{Y}\|_2^2,$$

un classique calcul de variance montre que, en notant $\hat{\epsilon} = \mathbf{Y} - \hat{\mathbf{Y}}$, $\text{var}(\hat{\epsilon}) = \sigma_\epsilon^2\left(\mathbb{I} - \mathbf{X}\mathbf{X}^+\right)$, en remarquant que $\left(\mathbb{I} - \mathbf{X}\mathbf{X}^+\right)$ est un projecteur. Ainsi,

$$\begin{aligned} \mathbb{E}[\|\hat{\epsilon}\|_2^2] &= \mathbb{E}[\text{Trace}(\hat{\epsilon}\hat{\epsilon}^T)] \\ &= \text{Trace}(\mathbb{E}[\hat{\epsilon}\hat{\epsilon}^T]) = \text{Trace}(\text{var}[\hat{\epsilon}]) \\ &= \sigma_\epsilon^2(n-p), \end{aligned} \tag{1.31}$$

car $\left(\mathbb{I} - \mathbf{X}\mathbf{X}^+\right)$ est un projecteur de rang p. Ainsi on peut formuler l'espérance du R^2

$$\mathbb{E}\left[R^2\right] = \frac{p}{n},$$

et le coefficient de détermination est sensible de façon linéaire au nombre de variables dans le modèle. Ceci induit de ne pas pouvoir comparer deux modèles dont le nombre de variables n'est pas le même.

Pour cette raison, il a été recherché des méthodes permettant de régulariser le critère d'ajustement aux données par un terme permettant de pénaliser les modèles à grands p.

Le critère de R^2 ajusté $R_a^2 = R^2 - \dfrac{p(1-R^2)}{n-p-1}$

tente de rattraper ce défaut.

Le C_p de Mallows

Afin de rattraper l'effet néfaste du nombre de paramètres dans le cas du R^2, Mallows (1973) a proposé une version pénalisée du SCR appelée aujourd'hui

$$C_p \text{ de Mallows} : C_p \;=\; \frac{SCR}{\hat{\sigma}_\epsilon^2} \;-\; n \;+\; 2p. \tag{1.32}$$

D'après l'équation (1.31), nous avons pu remarquer que le SCR admettait pour espérance $\sigma_\epsilon^2(n-p)$, ainsi lorsque le modèle construit ne viole pas les hypothèses d'indépendance du bruit du modèle, le C_p devrait coller à p. Le C_p est donc utilisé en le comparant à p, plus ce dernier est proche de p et plus le modèle est bon. Ce critère demande de calculer un estimateur de la variance σ_ϵ^2, ce qui peut être un choix difficile devant le grand nombre d'estimateurs existants, La littérature valorise l'estimateur non biaisé de la variance dans ce cas.

Le critère AIC

Le critère d'information d'Akaike (*AIC*), développée par Akaike (1973), s'applique aux problèmes estimés par maximisation de vraisemblance. Il pénalise la vraisemblance du problème considéré directement par le nombre de paramètres à estimer, noté p ici, soit

$$AIC = -2\log\tilde{L} + 2p,$$

où $\tilde{L}$ est la vraisemblance maximisée. On reconnaît une pénalisation de l'erreur d'estimation par le nombre de paramètres du modèle, c'est donc de nouveau une recherche de compromis de type biais-variance qui est faite.

Le critère BIC

Cherchant à inclure les contraintes de faible échantillonnage, Schwarz et collab. (1978) ont développé le critère d'information bayésienne (BIC). Il se base sur un formalisme bayésien et recherche le modèle le plus probable *a posteriori*. Il se formule

$$BIC = -2\log\tilde{L} + p\log(n),$$

lorsqu'aucun *a priori* n'est pris sur la distribution du modèle.

Le BIC diffère de l'AIC par le terme « $p(\log(n) - 2)$ », ainsi dès lors que $\ln\ n > 2$, soit pour $n \leqslant 8$, le BIC assure une pénalisation plus importante que l'AIC, ce qui implique qu'en pratique le BIC aura tendance à conserver des modèles plus parcimonieux que l'AIC. Ces 2 critères ont été affinés en d'autres critères. Leur principal limite est de ne pouvoir être utilisable que lorsqu'une vraisemblance est associée au problème considéré, ce qui n'est pas forcément le cas.

1.4.2 Méthodes de rééchantillonnage

Ces méthodes se basent sur la majoration du risque par son risque empirique ajouté d'une fonction dépendant d'un paramètre de complexité, ce qui a été vu au tournant de l'équation (1.3) et du principe ERM. Elles cherchent à estimer l'espérance du risque empirique (1.2) d'un estimateur $\hat{f}$ de f sur la réalisation D_n qui suit la loi de $\mathcal{D}_n$, discutée dans les parties précédentes. L'objectif, qui est aussi la contrainte, étant de se servir uniquement de D_n pour estimer cette grandeur.

La validation simple

Le principe est de découper l'ensemble $\mathcal{D}_n$ en deux sous-échantillons non vides. Le premier ensemble est qualifié d'ensemble d'apprentissage $\mathcal{D}_n^a$ avec I_n^a les indices correspondant dans $[\![1,n]\!]$. Le second l'ensemble de validation $\mathcal{D}_n^v$ avec I_n^a ses indices. On a forcément $I_n^a \cup I_n^v = [\![1,n]\!]$ et $I_n^a \cap I_n^v = \varnothing$. On définit ainsi

l'estimateur « simple » du risque empirique de $\hat{f}$ sur $\mathcal{D}_n$ via

$$\hat{\mathcal{R}}^{(Simple)}\left(\hat{f}, \mathcal{D}_n, I_n^v\right) = \frac{1}{\#\left\{I_n^v\right\}} \sum_{i \in I_n^v} L\left(y_i, \hat{f}_{\mathcal{D}_n^a}(X_i)\right). \tag{1.33}$$

Il vient alors directement que

$$\text{l'espérance de l'estimateur simple du risque empirique de } \hat{f} \text{ sur } \mathcal{D}_n \text{ vaut} \quad \mathbb{E}\left[\hat{\mathcal{R}}^{(Simple)}\left(\hat{f}, \mathcal{D}_n, I_n^v\right)\right] = \mathbb{E}\left[\mathcal{R}_{\mathbb{P}}\left(\hat{f}, \mathcal{D}_n^a\right)\right], \tag{1.34}$$

qui ne dépend que de n_a, le nombre d'individus dans l'échantillon d'apprentissage. On peut démontrer que ce biais diminue avec le nombre d'individus dans l'échantillon d'apprentissage n_a.

D'après Arlot (2018) par exemple, on définit une forme compacte de

$$\text{la variance de l'estimateur simple du risque empirique de } \hat{f} \text{ sur } \mathcal{D}_n \quad \text{var}\left[\hat{\mathcal{R}}^{(Simple)}\left(\hat{f}, \mathcal{D}_n, I_n^v\right)\right] = \frac{1}{n_v}\mathbb{E}\left[\text{var}\left[L\left(\hat{f}\left(\mathcal{D}_n^a\right)\right)\right]\right] + \text{var}\left[\mathcal{R}\left(\hat{f}\left(\mathcal{D}_n^a\right)\right)\right],$$

en notant $n_v = \#\{I_n^v\}$. On remarque ainsi que n_v fait décroître cette variance. L'estimateur de validation simple semble donc être stabilisé lorsque n_v grandit. Nous pouvons démontrer (voir Arlot, 2018, par exemple) que le second terme de cette somme diminue avec n_a.

L'estimateur de validation simple se stabilise donc lorsque les nombres d'individus dans les échantillons d'entraînement et de test augmentent.

L'estimateur simple du risque empirique est basé sur l'estimation d'un seul modèle de prédiction. Ceci le rend sujet aux variations de ce dernier, que l'on sait importantes dans le cas de la grande dimension grâce aux développements précédents. La solution de faire augmenter n_a ou n_v n'est pas acceptable puis n est fixé. L'idée est de découper $\mathcal{D}_n$ en plusieurs sous-échantillons d'entraînement et de test, ce que nous développons dans la section suivante.

La validation croisée à K *plis*

Cette méthode répond à la nécessité de stabiliser l'estimateur de validation simple du risque empirique. C'est la méthode la plus connue, elle a été introduite par Geisser (1975). En anglais *K-folds cross-validation*, *fold* signifiant pli.

Par rapport à la définition précédente, il convient de diviser $\mathcal{D}_n$ en $K \in [\![1, n]\!]$ sous-ensembles disjoints notés, $\forall k \in [\![1, K]\!]$, I_n^k et de cardinalités approximativement égales. K étant fixé par l'utilisateur. En reprenant les notations de la définition de l'estimateur de validation simple du risque empirique, on note

$$\text{l'estimateur à } K \text{ plis du risque empirique de } \hat{f} \text{ sur } \mathcal{D}_n \text{ via} \quad \hat{\mathcal{R}}^{(K)}\left(\hat{f}, \mathcal{D}_n, \left(I_n^k\right)_{k\in[\![1,K]\!]}\right) = \frac{1}{K}\sum_{k=1}^{K}\hat{\mathcal{R}}^{(Simple)}\left(\hat{f}, \mathcal{D}_n, I_n^k\right),$$

où le terme de la somme est l'estimateur simple défini au travers de l'équation (1.33).

On parle de Leave-One-Out (*LOO*), introduite de façon concomitante par Stone (1974); Allen (1974); Geisser (1975), lorsque le nombre de plis vaut $K = n$ et le découpage se fait en prenant une observation par pli. Dans ce cas précis il n'y a pas d'aléatoire dans la définition des

plis et l'estimateur présente un biais minimum par rapport aux cas $K < n$.

En supposant que les I_n^k ont tous pour cardinalité n_k et que $\mathcal{D}_n^a$ est formé de $n_a = n - n_k$ observations indépendantes de même loi $\mathbb{P}$, nous montrons que l'espérance de cet estimateur est la même que dans le cas de la validation simple, voir équation (1.34). L'estimation de la variance de l'estimateur du risque de validation croisée à K plis est quant à elle plus délicate et ne se prête pas à une forme générale.

L'expertise de l'analyste, dans le cas de la validation croisée à K plis, est de régler le paramètre K en recherchant un compromis entre biais et variance. Plusieurs solutions existent, il est par exemple possible d'augmenter le K, ce qui va augmenter n_a pour chaque pli, afin de réduire le biais. Ce qui augmente les temps de calcul.

La validation par « languette de botte » ou « *bootstrap* » en anglais

Le « bootstrap » permet d'évaluer l'espérance et la variance de cet estimateur en recréant un nombre B d'échantillons « bootstrap » choisis par tirages aléatoires équiprobables sur $\mathcal{D}_n$ de n observations avec remises, voir par exemple (voir par exemple Breiman et Spector, 1992) à ce sujet, on note $\mathcal{D}_n^{(B)}$ la variable aléatoire associée. Le paramètre B vaut typiquement quelques centaines. En notant alors $\mathcal{U}_{[\![1,B]\!]}$ la loi uniforme sur $[\![1, B]\!]$. Alors

l'estimateur bootstrap, à B réplications, du risque empirique de $\hat{f}$ sur $\mathcal{D}_n$ via

$$\begin{aligned}\hat{\mathcal{R}}^{(B)}\left(\hat{f}_n, \mathcal{D}_n, \left(I_n^b\right)_{b\in[\![1,B]\!]}\right) &= \frac{1}{nB}\sum_{b=1}^{B}\sum_{i\in I_n^b} L\left(y_i, \hat{f}_{\mathcal{D}_n^{(B)}}(X_i)\right)\\ &\text{où} \quad \forall b \in [\![1,B]\!],\ I_n^b \sim \mathcal{U}_{[\![1,B]\!]}.\end{aligned}$$

Or cet estimateur, puisque les données d'apprentissage sont aussi potentiellement utilisées pour le test, conduit à des modèles trop complexes et donc à des cas de surapprentissage (voir par exemple Efron, 1983). Ce dernier propose la méthode de « *leave-one-out bootstrap* » en remarquant que $\mathcal{D}_n^{(B)}$ contient en moyenne 63.2% de $\mathcal{D}_n$ et calcule le risque sur les échantillons qui n'ont pas été mis dans $\mathcal{D}_n^b$, on appelle $\overline{\mathcal{D}_n^b}$ le complémentaire de $\mathcal{D}_n^b$ dans $\mathcal{D}_n$ et $\overline{I_n^b}$ le complémentaire de I_n^b dans $[\![1, n]\!]$. On appelle cette erreur

l'erreur « *out-of-bag* » en anglais pour « en dehors du sac »

$$\begin{aligned}\hat{\mathcal{R}}^{(B)}\left(\hat{f}_n, \mathcal{D}_n, \left(I_n^b\right)_{b\in[\![1,B]\!]}\right) &= \frac{1}{B}\sum_{b=1}^{B}\frac{1}{\#\{\overline{\mathcal{D}_n^b}\}}\sum_{i\in\overline{I_n^b}} L\left(y_i, \hat{f}_{\mathcal{D}_n^{(B)}}(X_i)\right)\\ &\text{où} \quad \forall b \in [\![1,B]\!],\ I_n^b \sim \mathcal{U}_{[\![1,B]\!]}\\ &, \quad I_n^b \text{ est le complémentaire de } \overline{I_n^b} \text{ dans } [\![1,n]\!]\\ &\text{et} \quad \mathcal{D}_n^b \text{ est le complémentaire de } \overline{D_n^b} \text{ dans } [\![1,n]\!].\end{aligned} \tag{1.35}$$

Il a été remarqué un biais de cet estimateur est important qui équivaut à une validation croisée à 2 plis. Efron (1983) propose une correction de cet estimateur en le pénalisant par le risque empirique, voir l'équation (1.2) sur les proportions espérées des données d'apprentissage et de test, soit

l'estimateur .632 du risque empirique de $\hat{f}$ sur $\mathcal{D}_n$

$$\begin{aligned}\hat{\mathcal{R}}^{(.632)}\left(\hat{f}_n, \mathcal{D}_n, \left(I_n^b\right)_{b\in[\![1,B]\!]}\right) &= 0.368\hat{\mathcal{R}}\left(\hat{f}_n, \mathcal{D}_n\right) + 0.632\hat{\mathcal{R}}^{(B)}\left(\hat{f}_n, \mathcal{D}_n, \left(I_n^b\right)_{b\in[\![1,B]\!]}\right)\\ \text{où} \quad & \forall b \in [\![1,B]\!],\ I_n^b \sim \mathcal{U}_{[\![1,B]\!]}\\ , \quad & I_n^b \text{ est le complémentaire de } \bar{I_n^b} \text{ dans } [\![1,n]\!]\\ \text{et} \quad & \mathcal{D}_n^b \text{ est le complémentaire de } \bar{D_n^b} \text{ dans } [\![1,n]\!].\end{aligned}$$

Qui peut aussi être adapté en cas de surapprentissage important via

le taux d'erreur sans information $\gamma = \frac{1}{n^2}\sum_{i=1}^n\sum_{j=1}^n L\left(y_i, \hat{f}_{\mathcal{D}_n}\left(X_i\right)\right)$

et

le taux de surapprentissage relatif $\hat{R} = \dfrac{\mathcal{R}^{(B)}\left(\hat{f}_n, \mathcal{D}_n, \left(I_n^b\right)_{b\in[\![1,B]\!]}\right) - \hat{\mathcal{R}}\left(\hat{f}_n, \mathcal{D}_n\right)}{\gamma - \hat{\mathcal{R}}\left(\hat{f}_n, \mathcal{D}_n\right)}$,

pour créer

l'estimateur .632+ du risque empirique de $\hat{f}$ sur $\mathcal{D}_n$

$$\begin{aligned}\hat{\mathcal{R}}^{(.632+)}\left(\hat{f}_n, \mathcal{D}_n, \left(I_n^b\right)_{b\in[\![1,B]\!]}\right) &= \frac{0.368(1-R)\hat{\mathcal{R}}\left(\hat{f}_n, \mathcal{D}_n\right)}{1-0.368R}\\ &+ \frac{0.632\hat{\mathcal{R}}^{(B)}\left(\hat{f}_n, \mathcal{D}_n, \left(I_n^b\right)_{b\in[\![1,B]\!]}\right)}{1-0.368R}\\ \text{où} \quad & \forall b \in [\![1,B]\!],\ I_n^b \sim \mathcal{U}_{[\![1,B]\!]}\\ , \quad & I_n^b \text{ est le complémentaire de } \bar{I_n^b} \text{ dans } [\![1,n]\!]\\ \text{et} \quad & \mathcal{D}_n^b \text{ est le complémentaire de } \bar{D_n^b} \text{ dans } [\![1,n]\!].\end{aligned}$$

Qui lui-même souffre de problèmes très bien étudiés par la communauté mais que nous ne traiterons pas ici.

Remarque 1.5 (Bagging, forêts aléatoires et sélection de variables.)**.** *L'erreur « en dehors du sac » a d'autres applications notamment dans l'étude des forêts aléatoires de* Breiman *(2001) qui utilise cette erreur comme mesure de l'apprentissage de ses forêts en fonction du nombre d'arbres construits. L'agrégation des modèles de « bootstrap » permet justement d'ajouter le comportement d'un nouvel arbre de décision au comportement de la forêt qui est en construction. La forêt aléatoire se construit donc itérativement au moyenne de cette méthode d'aggrégation appelée « bagging », pour « bootstrap aggregating ». Ces résultats sont détaillés par* Breiman *(1996).*
L'erreur « en dehors du sac » est très utile pour déterminer l'importance d'une variable dans le modèle des forêts aléatoires. L'idée est de perturber une des variables de l'échantillon « en dehors du sac » et de regarder l'impact sur l'erreur de cet échantillon. On peut ainsi ajouter une mesure d'importance de chaque variable sur le modèle construit. L'utilisation de ce genre de mesure a poussé Genuer *et collab. (2010) à décrire un algorithme de sélection de variable pour les forêts aléatoires. Algorithme implémenté dans un paquet R sous le nom de VSURF, voir* Genuer *et collab. (2015).*

Autres méthodes de validation

D'autres méthodes que la *Validation-Croisée Généralisée* (GCV) existent, dans le cas des méthodes linéaires, détaillée par Golub et collab. (1979), qui permet de réduire le coût calculatoire de l'ensemble des K estimateurs à un seul en estimant la matrice $\mathbf{X}\left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T$. Ce manuscrit ne décrira pas plus ces méthodes. La validation croisée est le critère étudié par la suite. La connaissance de ce critère et son évolution en fonction de la complexité des modèles permet de fixer ces derniers.

1.5 Application à l'étude du jeu de données *MNIST*

Cette section permet d'observer l'impact d'hyperparamètres de régularisation au travers d'un exemple de reconnaissance de caractères. Dans ce cas le jeu de données est divisé en 2, un pour l'entraînement et un pour le test. Cet exemple est donné à titre introductif de l'intérêt des méthodes de régularisation, nous ne nous concentrerons donc pas sur la discussion du nombre de plis de la validation croisée. Nous utiliserons ici un seul pli.

Cette partie est aussi un moyen d'introduire les problèmes de classification que nous n'avons pas pu aborder plus tôt. Ces derniers sont abordés de façon pédagogique et ne nécessitent pas de connaissances autres que celles détaillées jusqu'à présent dans ce manuscrit.

1.5.1 Présentation du jeu de données

MNIST est un recueil de plusieurs dizaines de milliers de caractères manuscrits représentant les chiffres de 0 à 9 au travers d'images carrés d'une vingtaine de pixels de côté en niveaux de gris. Toutes ces images sont labellisées 0 à 9 en fonction du chiffre qu'elles représentent (voir LeCun et collab., 1998, par exemple). Cette base de données a beaucoup servi la recherche en reconnaissance automatique de formes, d'abord au moyen de modèles linéaires, puis grâce à des modèles non linéaires et aujourd'hui très complexes. MNIST a été étudié pour explorer des méthodes supervisées mais plus souvent non supervisées (voir Becht et collab., 2019; van der Maaten et Hinton, 2008, pour l'étude de méthodes appliquées à des problématiques non supervisées en biologie).

Nous avons construit un jeu de données d'entraînement de 200 images représentant autant de chiffres "2" que de chiffres "5". Le jeu de données de test est constitué de 10 chiffres "2" et 10 chiffres "5". Aucune image n'apparaît plus d'une fois dans les deux jeux de données. Le jeu de données d'entraînement et le jeu de données de test sont indépendants. Par rapport à ce qui a été expliqué dans la section 1.4.2, ceci correspond à un processus de validation simple.

1.5.2 Analyse non supervisée sans décision

On considère le problème régularisé non supervisé suivant

$$\min_{\mathbf{B}\in\mathbb{R}^{p\times p}} \|\mathbf{X}-\mathbf{X}\mathbf{B}\|_F^2 + \lambda\|\mathbf{B}\|_F^2, \tag{1.36}$$

où $\lambda \in \mathbb{R}^\star$. C'est un modèle linéaire d'auto-encodeur régularisé Ridge et l'estimateur associé s'écrit

$$\hat{\mathbf{B}} \quad = \quad \left(\mathbf{X}^T\mathbf{X}+\lambda\mathbb{I}\right)^{-1}\mathbf{X}^T\mathbf{X} \tag{1.37}$$

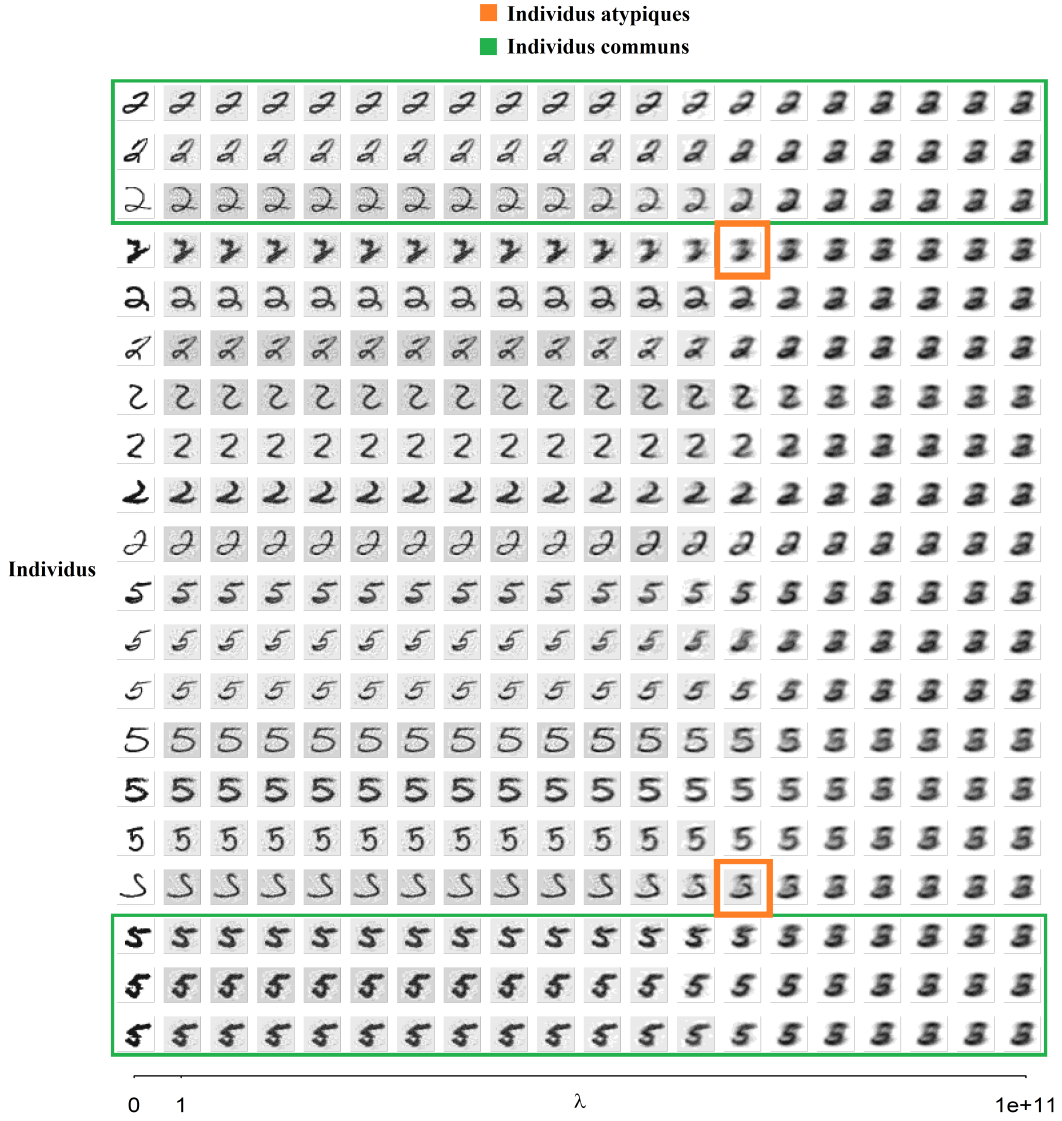


FIGURE 7 – Résolutions du problème (1.36) pour $\lambda \in [1, 10^{11}]$ variant horizontalement sur un échantillon test de 10 individus de chacune des 2 classes utilisées pour l'apprentissage. Avec 200 individus utilisé pour l'apprentissage. La première colonne correspondant aux individus bruts.

La figure 7 présente les encodages de l'échantillon de test pour quelques estimateurs correspondant à un coefficient de régularisation $\lambda \in [1, 10^{11}]$ où la première colonne est l'échantillon initial.

Les premières colonnes représentent les données avec le plus faible niveau de régularisation à l'inverse de ce qui se passe à l'extrême droite où les individus sont alors difficilement discernables, sauf peut-être en s'appuyant sur les barres verticales supérieures, plus ou moins intenses dans chacun des 2 cas.
Certains individus sont plus atypiques que d'autres, voir les deux images encadrées en orange par exemple. En effet, on trouve alors deux motifs que l' œil humain ne saurait reconnaître. Ceci s'oppose à ce qui se passe pour les individus les plus communs, en quelque sorte, leur régulari-

sation semble passer par une déformation moins douloureuse. Il faut regarder à ce propos les six individus encadrés en vert.

On comprend que ce genre d'analyse est très importante pour « connaître » les données mais ne permet pas de conclure de façon quantitative, à prendre une décision, ce qui est nécessaire en classification.

1.5.3 Utilisation d'une couche décisionnelle

Il convient de créer une règle décisionnelle qui puisse associer à une image courante une classe d'appartenance. En notant $n_a = 200$, p le nombre de pixels par image et $\mathbf{X}_i$ l'image vectorisée numéro i avec

— $\forall i \in [\![1, n_a/2]\!], \mathbf{X}_i$ est de la classe "2",

— $\forall i \in [\![n_a/2+1, n_a]\!], \mathbf{X}_i$ est de la classe "5".

Pour un nouvel individu d'image vectorisée $\mathbf{x}$ on définit alors la distance choisie de ce dernier à la classe $c \in \{2,5\}$, projetée via la matrice $\hat{\mathbf{B}}$ comme

$$d(\mathbf{x}, c, \hat{\mathbf{B}}) = \frac{\left|\left|\mathbf{x}\hat{\mathbf{B}} - \boldsymbol{\mu}_{\hat{\mathbf{B}}}^{(c)}\right|\right|_2}{\frac{1}{p}\sum_{j=1}^{p} \sigma_j^{(c)}}, \tag{1.38}$$

où $\boldsymbol{\mu}_{\hat{\mathbf{B}}}^{(c)}$ est le centre de masse des individus de la classe c observés dans l'espace régularisé, soit $\boldsymbol{\mu}_{\hat{\mathbf{B}}}^{(c)} = \frac{1}{n_a}\sum_{i=1}^{n_a} \mathbf{X}_i\hat{\mathbf{B}}$ et $\sigma_j^{(c)}$, $\forall j \in [\![1,p]\!]$ est l'écart-type de chacun des pixels estimé sur les individus de la classe c dans l'espace initial. On suppose que le dénominateur est non nul, ce qui n'est pas une hypothèse très forte. Ce dénominateur permet de dévaloriser les classes qui présentent peu de variations. Ainsi si $\mathbf{x}$ est plus proche d'une classe c_1 qui est très compacte, cette distance pourrait l'associer préférentiellement à la classe c_2, i.e. $d(\mathbf{x}, c_1, \mathbf{B}) > d(\mathbf{x}, c_2, \mathbf{B})$. La règle de décision devient

$$classe(\mathbf{X}, \hat{\mathbf{B}}) = \underset{c\in\{2,5\}}{\arg\min}\, d(\mathbf{X}, c, \hat{\mathbf{B}}).$$

La courbe bleue de la figure 8 présente l'erreur estimée sur 20 individus de test par cette procédure pour les vingt coefficients de régularisation discutés précédemment ($\lambda \in 10^{[0,11]}$). Un point supplémentaire a été ajouté qui représente le cas $\mathbf{B} = \mathbb{I}$, pris par convention pour $\lambda = 0$, labellisé **Non régularisé**. La courbe rouge sera discutée plus tard. On remarque que la régularisation permet une nette chute de l'erreur dès les plus faibles niveaux de régularisation. En effet le modèle sans régularisation implique 42.5% d'erreur de test alors que les modèles suivants, et pour une bonne proportion de la fenêtre de recherche, les erreurs de test valent 7.5%. Lorsque la régularisation devient trop importante, l'erreur remonte. On dit que le modèle est surcontraint et ne parvient plus à apprendre des données. Par rapport à ce qui a été discuté dans les précédentes parties, on peut dire que le modèle a oublié les caractéristiques communes au jeu de données d'entraînement.

L'algorithme recherche l'information la plus abondante dans le jeu de données sans vérifier que cette dernière soit en lien avec la question posée : la discrimination des deux classes. Non seulement les modèles créés ne sont pas capables de rechercher dans la direction adéquat mais en plus ils décrivent des dimensions très riches, car la matrice recherchée, $\hat{\mathbf{B}}$, admet un grand

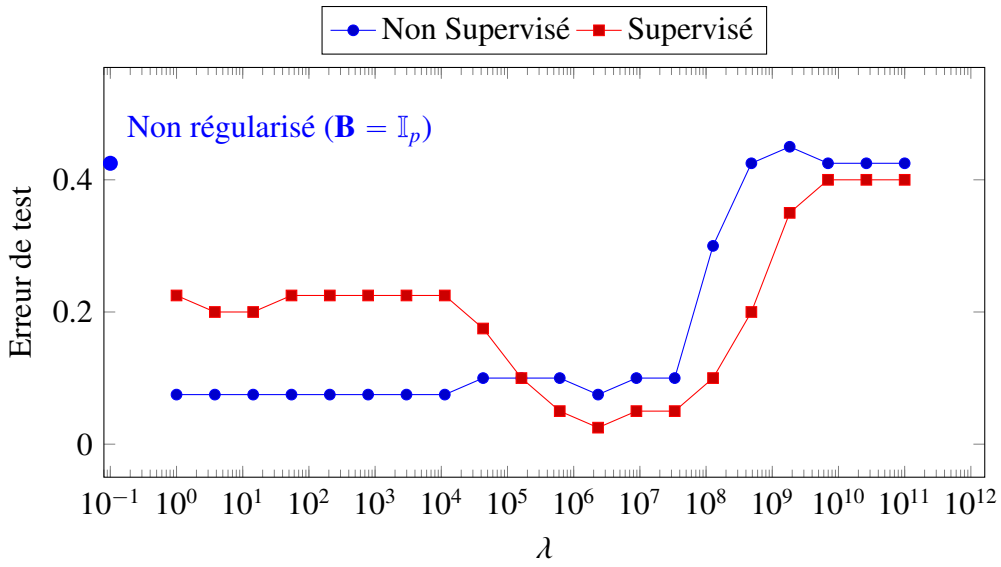


FIGURE 8 – Erreur empirique sur 20 individus de test du jeu de données *MNIST*

nombre de coefficients, p^2, avec un temps de calcul important. Il convient donc de rechercher l'information de façon supervisée afin de produire un modèle plus efficace et plus rapide.

1.5.4 Méthode supervisée

Le problème d'auto-encoder permet la compression de l'information mais ne prend pas en compte l'information à prédire. Nous considérons le problème des MCO régularisé Ridge suivant

$$\min_{\mathbf{B}\in\mathbb{R}^{p\times q}} \|\mathbf{Y} - \mathbf{XB}\|_F^2 + \lambda\|\mathbf{B}\|_F^2, \tag{1.39}$$

où $\mathbf{Y}$ code l'information de classe de chacun des individus, dans notre cas, c'est une matrice à deux colonnes où chaque ligne correspond à un individu et la colonne à la classe d'appartenance de l'individu en question, colonne une pour la classe "2" et colonne deux pour la classe "5".
Le même protocole que celui employé dans la section non supervisée de cet exemple a été appliqué. Les taux d'erreur sont visibles en rouge sur la figure 7 où le même principe de décision que dans le travail précédent a été utilisé. On voit que l'erreur est plus faible pour une partie de la fenêtre de régularisation, autour de $\lambda = 10^6$, pour un minimum valant 2.5%, soit une seule erreur sur les quarante individus testés. C'est un chiffre 5 qui pose problème, il ressemble au quatrième chiffre en partant du bas de la figure 7. La zone des faibles régularisations correspond à des erreurs autour de 2% et les erreurs dans le cas d'une forte régularisation sont proches du cas non supervisé. De plus, les temps de calculs sont bien moins importants que dans le cas non supervisé, ici seulement $p \times 2$ coefficients doivent être estimés par modèle.

1.5.5 Conclusion de l'exemple

Cet exemple montre l'intérêt des méthodes régularisées et supervisées dans le cas d'un problème de classification pour plusieurs raisons :

— Les problèmes supervisés permettent de :
  — Réduire les temps de calcul lorsque $p > q$, ce qui est toujours le cas en pratique.

— Réaliser de meilleurs prédictions, car les sous-espaces construits sont directement associés avec la réponse.

— La régularisation permet de dégager l'information parasitaire des individus exotiques au profit de l'information commune.

1.6 Les méthodes multiblocs

Le terme de bloc renvoie à une structuration particulière des données qui permet de diviser le jeu de données en groupes de variables cohérents. On distingue un sous-type de données multiblocs qui se rencontrent lorsque les différents groupes décrivent des données d'un même type, recueillis au travers de différentes modalités. C'est le cas par exemple d'un même groupe de données mesurées à différents temps. On emploie souvent la terminologie "données multivoies" pour caractériser ce genre de données.

Les deux sous-sections suivantes présentent les approches retenues dans l'étude des données multiblocs puis des données multivoies.

1.6.1 Cas général

On suppose qu'il existe un nombre $T \in \mathbb{N}^\star$ de blocs, sous forme de matrices $(\mathbf{X}_1, ..., \mathbf{X}_T) \in \mathbb{R}^{n\times p_1} \times \cdots \times \mathbb{R}^{n\times p_T}$ (où $(p_1, \cdots, p_T) \in \mathbb{N}^{\star T}$ et $n \in \mathbb{N}^\star$) servant à expliquer les variables réponse et décrivant les mêmes individus dont les observations sont mises sous forme d'une matrice $\mathbf{Y} \in \mathbb{R}^{n\times q}$.

Le contexte PLS se prête naturellement à l'application de méthodes multiblocs. "SW-Harald-HW multiblock algorithm" par Wold (1984) est la première mention de ce genre de méthode. Quelques années plus tard, Wangen et Kowalski (1989) ont détaillé l'approche des moindres carrés partiels multiblocs (*MBPLS*). Cette méthode s'appuie sur le critère d'optimisation suivant

$$\begin{aligned} \max_{(\mathbf{u},\boldsymbol{\beta},\mathbf{v})} \quad & \sum_{t=1}^{T} \beta_t \mathbf{v}^T \mathbf{Y}^T \mathbf{X}_t \mathbf{u}_t \\ \text{s.t.} \quad & \mathbf{u}_t^T \mathbf{u}_t = \mathbf{v}^T \mathbf{v} = \boldsymbol{\beta}^T \boldsymbol{\beta} = 1, \end{aligned} \tag{1.40}$$

où les coefficients β_t permettent de mettre en commun l'information issue de chacun des T blocs $\mathbf{X}_t$ au travers de la projection sur les sous-espaces définis par les *poids* $\mathbf{u}_t$ respectivement. Ce qui permet de construire la *super-composante*

$$\sum_{t=1}^{T} \beta_t \mathbf{X}_t \mathbf{u}_t. \tag{1.41}$$

MBPLS est optimisée au moyen de l'algorithme NIPALS.

Des étapes de déflations comme pour la PLS classique sont utilisées. Une littérature riche a été nourrie par des discussions au sujet du type de déflation à utiliser. Le principe général de la déflation est exposé dans la sous-section 1.3.5. La déflation telle que présentée au travers de l'équation (1.26) permet de retirer l'information d'un des blocs à ce même bloc. Ce qui a l'avantage donc de ne pas introduire dans un bloc, l'information provenant des autres

blocs. Westerhuis et collab. (1997) ont montré que la déflation de tous les blocs sur la super-composante (1.41) présentait d'intéressants résultats en prédiction. Westerhuis et Smilde (2001) discute de ces différents aspects.

Qin et collab. (2001) ont démontré que la méthode MBPLS est en réalité équivalente à une PLS classique réalisée sur la concaténation des T blocs si l'on divise chaque bloc par le nombre de variables de chaque bloc correspondant [(f)]. La méthode MBPLS a été implémentée dans le package ade4 [(g)] en considérant la déflation sur la super-composante discutée au travers de l'équation (1.26).

Le lien entre cette méthode et l'analyse des redondances (*RDA*) (voir van den Wollenberg, 1977) a été mis en évidence par Bougeard et collab. (2011). L'analyse RDA repose sur un problème proche de celui décrit par l'équation (1.40) à ceci près que les contraintes de norme unitaire associées aux poids $\mathbf{u}_t$ sont remplacées par des contraintes de norme unitaire sur les scores, soit le problème suivant

$$\begin{array}{ll} \max\limits_{(\mathbf{u},\boldsymbol{\beta},\mathbf{v})} & \sum_{t=1}^{T} \beta_t \mathbf{v}^T \mathbf{Y}^T \mathbf{X}_t \mathbf{u}_t \\ \text{s.c.} & (\mathbf{X}_t \mathbf{u}_t)^T \mathbf{X}_t \mathbf{u}_t = \mathbf{v}^T \mathbf{v} = \boldsymbol{\beta}^T \boldsymbol{\beta} = 1. \end{array} \tag{1.42}$$

Cette méthode a été généralisée à l'analyse canonique des corrélations (*CCA*) détaillée par Hotelling (1936), au moyen d'un paramètre $\eta_t \in [0,1]$ qui permet de valoriser le côté corrélation de la PLS où le côté covariance de la PLS au moyen de la nouvelle contrainte

$$\eta_t (\mathbf{X}_t \mathbf{u}_t)^T \mathbf{X}_t \mathbf{u}_t + (1-\eta_t) \boldsymbol{u}_t^T \boldsymbol{u}_t = 1, \tag{1.43}$$

qui peut être associée à une régularisation de type Ridge, voir Tenenhaus et Tenenhaus (2011). Cette méthode porte le nom d'analyse canonique des corrélations généralisée régularisée (*RGCCA*). Une contrainte de parcimonie a aussi été introduite, au moyen de l'astuce du Lasso, afin de permettre la sélection de variables, c'est l'analyse canonique des corrélations généralisée parcimonieuse (*SGCCA*) (voir Tenenhaus et collab., 2014). Les auteurs de ces deux travaux assurent la convergence de leurs algorithmes, qui sont de type NIPALS.

Les méthodes gPLS et sgPLS permettent aussi de gérer l'aspect multibloc, tout comme la méthode MBPLS mais en introduisant des contraintes de type PLASM et gLasso pour définir le modèle gPLS (voir Liquet et collab., 2015). Le modèle sgPLS est lui-même décrit par Liquet et collab. (2015) comme une transposition du critère employé par Simon et collab. (2013) pour définir le sgLasso. Ces méthodes permettent la prise en compte de données multiblocs en PLS en introduisant des contraintes de groupe, elles sont accompagnées de résultats théoriques et d'applications à des jeux de données réels.

Les méthodes présentées dans cette section partagent en commun de tenter de résoudre un problème d'optimisation unique. La méthode MBPLS par exemple va résoudre le problème (1.40). En notant $\mathcal{L} = \sum_{t=1}^{T} \beta_t \mathbf{v}^T \mathbf{Y}^T \mathbf{X}_t \mathbf{u}_t$ le critère minimisé par ce problème, $\forall t \in [\![1,T]\!]$, $\mathbf{t}_t = \mathbf{X}_t \mathbf{u}_t$, $\mathbf{s} = \mathbf{Y}\mathbf{v}$, $\mathbf{t} = [\mathbf{t}_1, \cdots, \mathbf{t}_T]$ et en résolvant le problème (1.40) au moyen de la méthode du lagrangien, on arrive au système à $T+2$ équations suivant

(f). A condition que toutes les variables aient la même variance.
(g). Disponible sur le CRAN via https://cran.r-project.org/package=ade4.

$$\left\{\begin{array}{rcl:rcl} \partial_{\mathbf{u}_1}\mathcal{L} & = & 0 & \beta_1\mathbf{v}^T\mathbf{Y}^T\mathbf{X}_1 & = & \lambda_1\mathbf{u}_1 \\ \cdots & & & & & \\ \partial_{\mathbf{u}_T}\mathcal{L} & = & 0 & \beta_T\mathbf{v}^T\mathbf{Y}^T\mathbf{X}_T & = & \lambda_T\mathbf{u}_T \\ \partial_{\mathbf{v}}\mathcal{L} & = & 0 & \mathbf{Y}^T\mathbf{t}\,\boldsymbol{\beta} & = & \lambda_{\mathbf{v}}\mathbf{v} \\ \partial_{\boldsymbol{\beta}}\mathcal{L} & = & 0 & \mathbf{s}^T\mathbf{t} & = & \lambda_{\boldsymbol{\beta}}\boldsymbol{\beta} \end{array}\right. ,$$

où $\lambda_1, \cdots, \lambda_T, \lambda_{\mathbf{v}}, \lambda_{\boldsymbol{\beta}}$ sont les coefficients de Lagrange et qui est souvent résolut par un algorithme alterné du type NIPALS. Cet algorithme résout ligne par ligne ce système en mettant à jour les variables estimées aux étapes précédentes. C'est donc du fait de la présence de toutes les variables à estimer à toutes les lignes de ce système que l'algorithme est amené à réestimer de façon itérative ce système. Ce genre d'algorithme peut aboutir à des calculs longs voire même qui ne convergent pas dans le contexte des données manquantes comme observé dans Lorenzo et collab. (2019b).

D'autres méthodes, qui n'ambitionnent pas de résoudre des problèmes uniques tels que celui qui vient d'être décrits, permettent de résoudre des problèmes multiblocs comme la méthode de structuration de tableaux à trois indices de la statistique (*STATIS*) (L'Hermier des Plantes, 1976) ou de l'analyse factorielle multiple (*AFM*) (Escofier et collab., 1984). Ces méthodes sont particulièrement rapides puisqu'elles fonctionnent sur le principe de fournir tout d'abord une description de chaque bloc séparément puis une description de la réunion de ces descriptions, ce qui évite les itérations algorithmiques à l'origine des problèmes de lenteurs algorithmiques. On pourrait ainsi imaginer l'algorithme

$$\left\{\begin{array}{rcl} \mathbf{v}_1^T\mathbf{Y}^T\mathbf{X}_1 & = & \lambda_1\mathbf{u}_1 \\ \cdots & & \\ \mathbf{v}_T^T\mathbf{Y}^T\mathbf{X}_T & = & \lambda_T\mathbf{u}_T \\ \mathbf{Y}^T\mathbf{t}\mathbf{w} & = & \lambda_{\mathbf{v}}\mathbf{v} \\ \boldsymbol{\gamma}^T\mathbf{s}^T\mathbf{t} & = & \lambda_{\boldsymbol{\beta}}\boldsymbol{\beta} \end{array}\right. , \tag{1.44}$$

où chaque étape est bien indépendante des étapes suivantes. Un œil averti reconnaît qu'à chaque ligne, le coefficient à droite de chaque égalité est en fait le premier vecteur singulier droit des matrices $\mathbf{Y}^T\mathbf{X}_1, \cdots, \mathbf{Y}^T\mathbf{X}_T, \mathbf{t}^T\mathbf{Y}$ et $\mathbf{s}^T\mathbf{t}$. Ce genre de décomposition peut donc être interprété de façon hiérarchique en partant des données pour se diriger vers les composantes. Ceci permet un meilleur interprétabilité des modèles en partant du plus "réelle", à savoir les données, pour aboutir jusqu'au plus "abstrait", les composantes.

1.6.2 Les données multivoies

On appelle données multivoies des données dont les n individus ont été mesurés T fois au travers des mêmes p variables pour différentes modalités. Ces modalités peuvent être des instants ou des longueurs d'onde d'observation par exemple. On choisit souvent la représentation tensorielle, ou cubique. Où l'objet des données est formé par construit, dans le cas de l'analyse de données classique, par trois dimensions correspondant aux trois dimensions de l'étude, (individus,variables,modalités) par exemple. Cet objet possède des opérations et des propriétés dont certaines vont être, sans rentrer dans le détail, présentées dans ce qui suit. On parle aussi de T-voies où T désigne le nombre de modalités.

Méthodes non supervisées

Des méthodes non supervisées ont permis l'analyse de ce genre de données. La méthode Tucker3 (voir Kroonenberg, 1983) permet la description de données à 3-modalités grâce au modèle suivant

$$\forall (i,j,t) \in [\![1,n]\!] \times [\![1,p]\!] \times [\![1,T]\!],\ x_{ijt} = \sum_{p'=1}^{P'} \sum_{q=1}^{Q} \sum_{r=1}^{R} a_{ip'} b_{jq} c_{tr} g_{p'qr} + e_{ijt}, \tag{1.45}$$

où n, p et T sont les cardinalités de chacun des modes. Pour reprendre l'exemple ci-dessus, n serait le nombre d'individus, p le nombre de variables et T le nombre de répétitions. De plus P', Q et R représentent le nombre d'axes dans chaque modalité. $A = \left(a_{ip'}\right)$, $B = \left(b_{jq}\right)$ et $C = \left(c_{tr}\right)$ sont les matrices de composantes de chacune des modalités et $G = \left(g_{pqr}\right)$ est appelée matrice dite "centrale". $E = \left(e_{ijt}\right)$ correspond aux approximations du modèle. Ce modèle est estimé en minimisant l'erreur quadratique des erreurs de modèle. La matrice $\mathbf{G}$ permet de pondérer certaines composantes par rapport aux autres. On dit de ce modèle qu'il généralise l'ACP au contexte des données 3-voies.

L'analyse factorielle parallèle (*PARAFAC* ou *CP*) (voir Harshman, 1970; Carroll et Chang, 1970) à 3 modalités se rapproche du modèle Tucker3 dans le cas où $P' = Q = R$ et où les éléments de la matrice centrale sont tous nuls sauf les éléments diagonaux qui sont égaux à 1. On obtient ainsi le modèle [(h)]

$$\forall (i,j,t) \in [\![1,n]\!] \times [\![1,p]\!] \times [\![1,T]\!],\ x_{ijt} = \sum_{r=1}^{R} a_{ir} b_{jr} c_{tr} + e_{ijt}, \tag{1.46}$$

où la signification des paramètres a été détaillée plus haut. Si le nombre de composantes choisi n'est pas trop important au regard de la somme des rangs de chacune des matrices $\mathbf{A}$, $\mathbf{B}$ et $\mathbf{C}$, soit plus précisément si

$$rang(\mathbf{A}) + rang(\mathbf{B}) + rang(\mathbf{C}) \leqslant 2R + 2,$$

alors la décomposition correspondante est unique (voir Kruskal, 1977). L'équation (1.46) peut aussi s'écrire au moyen de l'opérateur de *CP*, ainsi le tenseur d'ordre 3 résultant se note

$$[\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$$

L'algorithme initialement employé se base sur un algorithme alterné de type NIPALS. L'algorithme de Gauss-Newton a aussi été utilisé dans plusieurs méthodes d'estimation et l'astuce de Levenberg-Marquardt (voir Marquardt, 1963) a été employée dans les cas dégénérés (Tomasi et Bro, 2005, voir). Voir Tomasi et Bro (2006) pour un comparatif de neuf méthodes différentes.

Méthodes supervisées

Les méthodes PARAFAC/CP et Tucker3 ne permettent pas de répondre aux questions posées par des jeux de données supervisés. La méthode PLS à N voies (*NPLS*) (voir Bro, 1996), gère des jeux de données où la partie des covariables est formée par des données multivoies. NPLS est une méthode qui recherche les *poids*

(h). On décrit ici le modèle PARAFAC/CP à trois modalités mais il est possible de construire des modèles à N modalités pour N entier aussi grand que désiré.

— $\mathbf{v}$, associé à la matrice $\mathbf{Y}$,

— $\mathbf{w}^J$, associé à la seconde modalité de $\mathbf{X}$,

— $\mathbf{w}^K$, associé à la troisième modalité de $\mathbf{X}$.

Il est aussi défini la matrice

$$\mathbf{Z}(\mathbf{v}) = \left((\mathbf{Y}\mathbf{v})^T \mathbf{X}_{jk} \right) \in \mathbb{R}^{J \times K}, \tag{1.47}$$

où $\mathbf{X}_{jk} \in \mathbb{R}^{I \times 1}$ s'interprète comme l'ensemble des réalisations des I valeurs de la première modalité pour les modalités deux et trois valant i et j respectivement. En supposant ici que la première modalité représente les *individus*, la seconde les *variables* et la troisième les *réalisations temporelles* ; ceci dans un seul objectif d'interprétabilité. Chaque élément de la matrice $\mathbf{Z}$ représente la covariance entre la représentation des *individus* côté $\mathbf{Y}$ via sa projection sur l'axe courant $\mathbf{v}$ et la représentation des mêmes *individus* côté $\mathbf{X}$ pour la *variable* au *temps* k.

Le problème de la NPLS s'écrit alors

$$\begin{array}{ll} \max\limits_{(\mathbf{v},\mathbf{w}^J,\mathbf{w}^K)} & \left(\mathbf{w}^J\right)^T \mathbf{Z}(\mathbf{v})\mathbf{w}^K \\ \text{s.c.} & ||\mathbf{v}||_2^2 = \left|\left|\mathbf{w}^J\right|\right|_2^2 = \left|\left|\mathbf{w}^K\right|\right|_2^2 = 1, \end{array} \tag{1.48}$$

ceci par composante. La déflation étant assurée par les transformations suivantes

$$\left\{ \begin{array}{lcll} \mathbf{s} & = & \mathbf{Y}^J\mathbf{v} & \in \mathbb{R}^{I\times 1} \\ \mathbf{T} & = & \left(\left(\mathbf{w}^J\right)^T \mathbf{X}_i \mathbf{w}^K \right) & \in \mathbb{R}^{I\times 1} \\ \mathbf{b} & = & \left(\mathbf{T}^T\mathbf{T}\right)^{-1}\mathbf{T}^T\mathbf{s} & \in \mathbb{R}^{I\times 1} \\ \mathbf{X}_i & \leftarrow & \mathbf{X}_i - t_i\mathbf{w}^J\left(\mathbf{w}^K\right)^T & \\ \mathbf{Y} & \leftarrow & \mathbf{Y} - \mathbf{T}\mathbf{b}\mathbf{v}^T & \end{array} \right. ,$$

où $\mathbf{T} = (t_i)$ est le score du modèle côté covariable et $\mathbf{s}$ côté réponse. $\mathbf{b}$ est la matrice de régression. Il est possible de montrer que la méthode $3-PLS$ est équivalente à la méthode PARAFAC/CP appliqué au tenseur à trois dimensions des matrices de covariances entre $\mathbf{X}$ et $\mathbf{Y}$, voir Hanafi et collab. (2015).

Le problème (1.49) présente un exemple de problème de fusion de données couplant la factorisation d'un tenseur $\mathcal{X}$ et d'une matrice $\mathbf{Y}$

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{V}} |||\mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!]|||^2 + \left|\left|\mathbf{Y} - \mathbf{A}\mathbf{V}^T\right|\right|^2, \tag{1.49}$$

on appelle ce problème la factorisation tensorielle d'une collection de matrices (*CMTF*), voir par exemple les travaux de Acar et collab. (2011b, 2013). Les matrices $\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{V}$ ont R colonnes, ce qui correspond au nombre de composantes recherchées. La matrice $\mathbf{A}$ permet de lier les deux parties du critère global en décrivant les n individus, elle a donc n lignes. La matrice $\mathbf{B}$ correspond aux variables de $\mathcal{X}$ et donc $\mathbf{B} \in \mathbb{R}^{p\times R}$. La matrice $\mathbf{C}$ correspond aux voies de $\mathcal{X}$ et donc $\mathbf{C} \in \mathbb{R}^{T\times R}$. La matrice $\mathbf{V}$ correspond aux variables de $\mathbf{Y}$ et donc $\mathbf{V} \in \mathbb{R}^{q\times R}$. La norme tensorielle employée dans le premier terme de l'équation (1.49) équivaut à $\mathcal{X} \rightarrow |||\mathcal{X}||| = \sqrt{\sum_{i=1}^n \sum_{j=1}^p \sum_{t=1}^T \mathcal{X}_{ijt}^2}$, directement inspirée de la norme euclidienne. Ce problème est classiquement résolu en utilisant un algorithme alterné, du type NIPALS, par rapport à $\mathbf{A}$ en initialisant $\mathbf{B}$ puis $\mathbf{C}$ et $\mathbf{V}$ puis par rapport à $\mathbf{B}$, $\mathbf{C}$ et enfin $\mathbf{V}$. Plusieurs résultats (voir Tomasi et Bro,

2006; Acar et collab., 2011a, par exemple) ont montré que l'algorithme alterné de résolution du problème PARAFAC/CP n'est plus adapté lorsque le nombre de composantes recherchées est supérieur au vrai nombre de composantes, alors que les méthodes *all-in-one* qui passent pas des méthodes de descente de gradient sur tous les blocs simultanément souffrent moins de ce problème. Ce problème porte le nom d'*overfactoring*, ou *sur-factorisation* en français.

Remarque

Des méthodes existent qui permettent de coupler les approches multiblocs et multivoies (voir Smilde et collab., 2000) mais ces méthodes dépassent le cadre de ce manuscrit et sont donc seulement citées à titre informatif.

1.7 Données manquantes

Le terme de données manquantes est associé à une information non répertoriée dans le jeu de données alors qu'elle devrait s'y trouver.

Des conditions peuvent apparaître en fonction de l'information que l'on a sur le processus de perte de données et la première section (voir 1.7.1, page 52) de cette partie permet de mettre un cadre théorique sur ces conditions. La seconde section permet de répertorier un certain nombre de méthodes utilisées pour la gestion de données manquantes (voir section 1.7.2, page 52). Cette partie ne vise pas à l'exhaustivité et a plutôt tendance à restreindre les solutions envisagées par les différents auteurs à celles qui correspondent au cadre de notre étude.

1.7.1 Processus de perte de données

Sous certaines conditions, des valeurs ne sont pas observées. Ce phénomène est présent dans toutes les branches de l'analyse de données et a été formellement décrit pour la première fois par Rubin (1976). Il s'appuie sur une description du processus de genèse des données manquantes. Il décrit ainsi trois cas possibles :

— MCAR (*Missing Completely At Random* pour *Manquant Complètement Aléatoire*), si la probabilité de perte de l'information est la même pour toutes les observations et donc ne dépend ni des données observées ni des autres données manquantes.

— MAR (*Missing At Random* pout *Manquant Aléatoire*), si la probabilité que la donnée soit manquante est associée à une ou plusieurs autres variables, observées.

— MNAR (*Missing Not At Random* pout *Manquant Non Aléatoire*), si la probabilité de ne pas observer cette valeur dépend de cette valeur et ou dépend de variables non observées.

Dans la suite de l'exposé nous nous concentrerons sur les données MCAR. Les données étudiées au moyen de la méthode discutée dans ce manuscrit vérifient les hypothèses MCAR. De futurs développements pourraient nous conduire à discuter de la robustesse de la méthode aux hypothèses MAR.

1.7.2 Prise en compte des données manquantes et imputation

Une façon courante de gérer les données manquantes est de supprimer les échantillons pour lesquels certaines valeurs ne sont pas observées, voir Josse et collab. (2009). Cette méthode

conduit à la réduction de la puissance statistique, sans biais si ce sont des données manquantes selon un processus MCAR. Le jeu de données associé à cette solution est qualifié de *cas complet*.

Dès lors que le nombre d'individus est faible, il ne convient pas de supprimer des données pour résoudre le problème des données manquantes, au risque de réduire encore la taille de l'échantillon. Une approche consiste à prédire les données manquantes, c'est l'*imputation*. On dit que l'imputation est simple lorsque la donnée est imputée une fois et qu'elle est multiple lorsqu'un certain nombre de jeux de données est généré avec des imputations différentes pour les données manquantes. L'imputation multiple répond à un impératif de conserver une certaine variabilité dans les données imputées afin de ne pas déformer les distributions, ceci est particulièrement important dans le cas des données MAR ou MNAR (voir Schafer et Graham, 2002, Figure 3).

Les modèles d'imputation sont très variés, allant de l'imputation à valeur constante (e.g. imputation par la moyenne) à des imputations conditionnelles issues de modèles multidimensionnels.

Cette partie est divisée en deux sections où nous discutons tout d'abord des solutions imaginées pour l'imputation à une valeur constante et ensuite de méthodes plus perfectionnées qui permettent de prendre en compte les éventuels liens avec d'autres variables et groupes de variables.

1.7.3 Imputation à valeur constante

L'imputation par une valeur constante d'une variable ayant des valeurs manquantes est très utilisée, la moyenne par exemple. L'intérêt de cette dernière est de ne pas modifier la moyenne de la variable imputée. Par contre, elle déforme la distribution de façon importante si la quantité de données manquantes devient non négligeable par rapport à la quantité de sujets. En plus de déformer les distributions marginales, Schafer et Graham (2002) montrent l'effet néfaste de ce genre d'imputation sur les structures de corrélation entre variables. La médiane peut être utilisée pour atténuer l'effet d'individus extrêmes mais ne saurait endiguer les autres effets néfastes discutés ici.

Afin de minimiser les effets délétères de l'imputation sur les distributions et sur les structures de corrélation, des méthodes d'imputation perfectionnées ont été développées et nous allons ici discuter de celles qui nous semblent être les plus importantes par rapport à notre problématique.

1.7.4 Imputation avec ajustement sur les autres variables

Le lien qu'une variable entretient avec une ou plusieurs autres variables, indépendamment du caractère manquant de certaines de ses réalisations est une information dont il faut tenir compte pour estimer la valeur des données manquantes.

Des modèles statistiques bien précis, qui peuvent être modifiés par l'imputation, rendent compte de la nature de ces intéractions entre les variables. Le fait d'imputer à une valeur constante, comme vu dans le paragraphe précédent, a tendance à déformer les distributions as-

sociées aux paramètres d'intérêt. Afin de résoudre ce problème il est possible d'imputer les données manquantes en faisant l'hypothèse d'un lien entre les variables.

Ce lien, et les paramètres statistiques associés, peuvent être estimés seulement sur les données observées. Ce qui présente le désavantage de ne pas fonctionner si beaucoup de données sont manquantes.

Une autre solution peut être d'estimer les données manquantes par itérations successives en ré-estimant de la même façon les modèles statistiques associés aux liens entre les variables. Les données manquantes étant elles-mêmes ré-estimées en maximisant un critère associé aux modèles statistiques et aux données observées. Un cadre algorithmique a été développé par Dempster et collab. (1977) afin de mettre en œuvre cette idée, il s'agit de l'algorithme EM, dont une description formelle est fournie en Annexe D, page 191 ainsi qu'un cas d'application simple où l'on peut apprécier son fonctionnement de façon pratique au travers de simulations. Cette dernière partie ne présente pas un travail novateur mais un support de réflexion à la méthodologie EM. La section suivante détaille son utilisation dans les méthodes modernes d'imputation et l'inspiration qu'il continue à insuffler dans des méthodes même si l'étiquette d'algorithme EM n'est pas formellement apposée.

Des méthodes d'imputation EM ou inspirées de l'algorithme EM

Comme nous l'avons déjà dit, l'algorithme EM est un cadre très général qui permet la mise en place de procédures d'imputation et de nombreuses méthodes l'utilisent. Nous allons voir dans cette partie des exemples d'utilisation de cette méthode.

Le concours Netflix, démarré en 2006 et clos en 2010, est sûrement l'évènement qui aura apporté le plus de visibilité au problème des données manquantes. Un jeu de données annonçant les préférences pour des centaines de milliers de films de dizaines de milliers d'utilisateurs était proposé et l'objectif des participants au concours était d'estimer les préférences des participants pour les films qu'ils n'ont pas notés, ce qui est présenté comme un problème de données manquantes. La spécificité étant que la proportion de données manquantes approche les 99%, ce qui est assez intuitif puisqu'il est très difficile pour un même spectateur de voir tous les films proposés par la plate-forme. Ceci implique notamment que le rang de la matrice des notes de tous les individus est très faible devant ses dimensions. L'algorithme développé par Mazumder et collab. (2010) a permis d'obtenir d'excellents scores au concours Netflix. Ce dernier s'appelle *Soft-Impute* (pour *Imputation douce* en anglais) et résout le problème

$$\min_{\mathbf{Z}} \frac{1}{2} ||P_\Omega(\mathbf{X}) - P_\Omega(\mathbf{Z})||_F^2 + \lambda ||\mathbf{Z}||_\star, \tag{1.50}$$

où λ est un coefficient de régularisation réel positif, P_Ω l'opérateur matricielle qui met à 0 les coefficients de la matrice considérée lorsque la position du coefficient concerné correspond à un coefficient non observé dans la matrice initiale avec données manquantes. Cette matrice est $\mathbf{X}$ dans le problème actuel et la matrice $\mathbf{Z}$ est la matrice approchée grâce à la résolution de ce problème. L'opérateur $||.||_\star$ associée au second terme, le terme de régularisation, est la *norme nucléaire*, c'est la somme des valeurs singulières de la matrice considérée. Ce problème, qui semble remarquablement complexe, admet pour solution, lorsqu'il n'y a aucune donnée man-

quante,

$$\mathbf{U}S_\lambda(\mathbf{D})\mathbf{V}^T,$$

où S_λ est l'opérateur de seuillage doux détaillé dans la section 1.3.6 et les matrices $(\mathbf{U}, \mathbf{V}, \mathbf{D})$ correspondant aux vecteurs singuliers gauches, droits et à la matrice diagonale complétée des valeurs singulières de la matrice décrite.

Bien que les auteurs de *Soft-Impute* ne revendiquent pas que cet algorithme soit un algorithme EM, ils proposent de nombreux résultats de convergence d'un critère approché de leur problème d'optimisation. Hastie et collab. (2015) détaillent les propriétés d'un algorithme proposé par Rennie et Srebro (2005) qui est une version alternée de l'algorithme *Soft-Impute*, à savoir

$$\min_{\mathbf{Z}} \frac{1}{2} \left|\left|P_\Omega\left(\mathbf{X} - \mathbf{A}\mathbf{B}^T\right)\right|\right|_F^2 + \frac{\lambda}{2}\left(||\mathbf{A}||_F^2 + ||\mathbf{B}||_F^2\right), \tag{1.51}$$

qui est résolu alternativement en $\mathbf{A}$ puis en $\mathbf{B}$, l'autre étant respectivement fixée. Ce problème est connu sous le nom de factorisation de matrice à marge maximale (*MMMF*) où $\mathbf{A}$ et $\mathbf{B}$ sont contraintes pour avoir des rangs inférieurs à un entier r fixé par l'opérateur. Hastie et collab. (2015) démontrent aussi la convergence d'un critère associé à une erreur de prédiction sur l'échantillon d'entraînement. Cette erreur ressemble terriblement à une vraisemblance partielle comme ce qui se fait dans les approches EM et leur démonstration de la croissance majorée de ce critère leur permet d'assurer que l'algorithme est très "Typé-EM". Les auteurs comparent cette méthode (`softImpute-ALS` dans l'article) à la version non alternée de *Soft-Impute* ainsi que l'algorithme qui consiste à estimer de façon alternée, et jusqu'à convergence, les matrices $\mathbf{A}$ et $\mathbf{B}$ en résolvant des problèmes multilinéaires (`ALS` dans l'article). Ils démontrent que la supériorité de `softImpute-ALS` en terme de temps de calcul sur un jeu de données de notes de films, `MovieLens 100k` ainsi que sur le jeu de données de `NetFlix`. Cet algorithme est celui auquel nous nous comparons dans la partie 2 et est implémenté dans le package softImpute, disponible sur le CRAN [(i)].

Parmi les méthodes permettant l'imputation dans le contexte de données monoblocs non supervisées, la méthode missForest (voir Stekhoven et Bühlmann, 2011) utilise les forêts aléatoires. C'est un algorithme de "Typé-EM" où chaque variable présentant des données manquantes est imputée au moyen de l'ensemble des autres variables. Les variables ayant initialement des données manquantes comme celles qui n'en ont ont pas sont utilisées. Cet algorithme demande un temps de calcul important si la proportion de données manquantes ou si le nombre de variables est important. Cet algorithme a aussi donné lieu à la création d'un package, missForest, disponible sur le CRAN [(j)].

La méthode NIPALS emprunte aux puissances itérées, exposées dans l'Annexe C, afin d'estimer les données manquantes d'un jeu de données par reconstruction inverse au moyen des axes définis par l'ACP. Cette méthode nécessite une initialisation des données manquantes. La moyenne des données présentes pour la variable considérée est souvent utilisée comme initialisation. Un algorithme "Typé-EM" est utilisé en alternant l'estimation des axes principaux, via la méthode ACP, puis l'estimation des données manquantes par projection de la matrice initiale sur les axes estimés en ne modifiant que les données manquantes initialement.

(i). https://CRAN.R-project.org/package=softImpute
(j). https://CRAN.R-project.org/package=missForest

Le package mixOmics, dans le cas de l'ACP, permet l'imputation de jeux de données au moyen de la méthode NIPALS. Les auteurs du package mixOmics proposent aussi d'utiliser la méthode NIPALS dans le contexte de problèmes supervisés tels que la PLS [(k)]. Cette solution semble poser deux problèmes qui sont :

— de forcer l'imputation à se faire sur les sous-espaces construits sur une méthode non supervisée, alors que le problème est supervisé. Dans le cas où la puissance statistique est faible il devient nécessaire d'introduire une parcimonie sur le nombre de composantes estimées et une méthode non supervisée ne peut pas, en général, faire ressortir des sous-espaces associés à la variable d'intérêt du problème [(l)].

— de rendre impossible toute méthode de validation du type validation croisée. En effet, les données imputées contiennent l'information associée à la totalité du jeu de données et notamment des données qui vont être utilisées comme jeu de données de test.

Remarque 1.6. *L'imputation non supervisée dans le contexte supervisé. Dans le cas de données de grande dimension, le nombre de composantes est potentiellement important. Si de plus le problème est supervisé, le modèle de prédiction doit fouiller dans ces nombreuses composantes afin de mettre en avant celles qui répondent le mieux au problème posé. Si des données sont manquantes dans ce contexte, il est nécessaire de considérer les composantes d'intérêt du problème dans le processus d'imputation. En effet, une imputation utilisant des informations non associées à l'information que l'on cherche à prédire conduit à une perte de puissance statistique. Ce qui n'est pas à négliger, cela d'autant plus que la taille de l'échantillon est faible.*

Josse et Husson (2012) se sont appuyés sur les résultats de Kiers (1997) afin de gérer les données manquantes en ACP. Ce dernier propose alors l'algorithme, qui prend aujourd'hui le nom d'iterative PCA qui consiste à initialiser les données manquantes à des valeurs d'intérêt, puis d'estimer les composantes principales sur les données complétées pour imputer les données manquantes grâce à la projection des données estimées à l'itération précédente sur les R composantes ainsi créées. Cet algorithme est un algorithme EM selon Josse et collab. (2009). Josse et Husson (2012) ont développé une ACP régularisée sur le shéma de l'iterative PCA afin d'éviter les problèmes de surapprentissage.
Cette méthode se base sur deux paramètres de pénalisation. Le nombre de composantes, qui doit être estimé par validation croisée ou par validation croisée généralisée, moins coûteuse en terme de temps calcul que la validation croisée classique. L'autre paramètre est associé à une approche probabiliste du problème de l'ACP, détaillée par Caussinus (1986). Le coefficient en question permet de réduire la variance de chacune des R composantes considérées en leur retirant une portion de variance correspondant aux composantes restantes. Le coefficient est assimilé à un coefficient ridge et ne nécessite pas le concours de l'utilisateur. Les points forts de cette méthode d'imputation régularisée sont

— de ne pas contraindre l'utilisateur à la recherche d'une valeur optimale pour le second coefficient,

— d'offrir un cadre théorique précis grâce à une définition d'un algorithme EM. En se basant sur les descriptions de Caussinus (1986); Roweis (1998); Tipping et Bishop (1999).

La généralisation à des jeux de données multitableaux hétérogènes de cette méthode a été réalisée par Husson et Josse (2013) où les auteurs utilisent le formalisme de l'AFM. Les auteurs

(k). Voir le tutoriel associé http://mixomics.org/methods/missing-values/.
(l). Voir la **Remarque sur l'imputation non supervisée dans le contexte supervisé**.

rappellent que l'AFM peut prendre la forme d'une ACP dont chaque variable est pondérée par l'inverse de la racine carrée de la première valeur propre du groupe de variables dont elle fait partie. De plus, les variables catégorielles sont dichotomisées afin des les intégrer au jeu de données continues . Les auteurs définissent alors une AFM itérative qui utilise l'ACP itérative, définie plus haut, appliquée à cette version dichotomisée puis pondérée du jeu de données initial. Cette méthode ne nécessite donc que le paramétrage du nombre de composante, ce qui doit être fait par validation croisée selon les auteurs. Le package *missMDA* (voir Josse et Husson, 2016) implémente ces différentes méthodes.

D'autres travaux permettent de gérer l'aspect multibloc du point de vue multivoie, ce qui est présenté dans la section 1.6.2 page 49, mais en prenant en compte les données manquantes. Tomasi et Bro (2005) comparent deux algorithmes adaptant la méthode PARAFAC/CP au moyen d'algorithmes « Typé-EM » où la différence se fait principalement via l'algorithme de résolution, à savoir

— un algorithme alternée nommé imputation simple par alternance (*ALS-SI*),

— un algorithme de résolution pondéré (mise à 0 du poids des données manquantes) adapté de la méthode de Levenberg–Marquardt (voir Marquardt, 1963), c'est la méthode paraFAC pour Données INcomplètes (*INDAFAC*).

La comparaison de ces méthodes de résolution se fait sur la base de simulations et sur l'application à des données réelles. Un léger avantage est donné à la méthode INDAFAC quoique les dimensions des données (relativement faibles, $30 \times 30 \times 30$ pour les nombres d'*individus*, de *variables* et de *modes*) ne soient pas au désavantage des méthodes globales de descente de gradient qui souffrent de la grande dimension [(m)].

Acar et collab. (2011b) proposent une autre résolution du problème PARAFAC/CP formalisée au travers du problème

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} |||P_\Omega\left(\mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!]\right)|||^2, \tag{1.52}$$

où l'opérateur de pondération P_Ω est introduit dans l'équation (1.50). Ce dernier problème, dénommé PARAFAC Optimal Pondéré (*CP-WOPT*), est adapté au problème CMTF avec données manquantes dans la partie tensorielle $\mathcal{X}$ via

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{V}} |||P_\Omega(\mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!])|||^2 + ||\mathbf{Y} - \mathbf{A}\mathbf{V}^T||^2, \tag{1.53}$$

lui-même baptisé factorisation tensorielle optimale pondérée d'une collection de matrices (*CMTF-WOPT*). Acar et collab. (2013) étudient le modèle de simulation

$$\begin{array}{rcccc} \mathcal{X} & = & \alpha\left[\!\left[\mathbf{A}^{(1)},\mathbf{A}^{(2)},\mathbf{A}^{(3)}\right]\!\right] & + & (1-\alpha)\left[\!\left[\mathbf{B}^{(1)},\mathbf{B}^{(2)},\mathbf{B}^{(3)}\right]\!\right] \\ \mathbf{Y} & = & \alpha\mathbf{A}^{(1)}\mathbf{V}^T & + & (1-\alpha)\mathbf{C}^{(1)}{\mathbf{C}^{(2)}}^T \end{array},$$

où $\alpha \in [0,1]$ contrôle le niveau d'association entre $\mathcal{X}$ et $\mathbf{Y}$, via la matrice $\mathbf{A}^{(1)}$. Dans ce contexte il semblerait que la méthode CMTF-WOPT ait un meilleur comportement lorsque le niveau

(m). La méthode ALS-SI est reprise par Hubert et collab. (2012) en la modifiant de façon à la rendre plus robuste aux valeurs extrêmes (les individus associé sont appelés *outliers* en anglais). L'algorithme résultant semble tenir ses promesses d'après les auteurs.

d'association est important, α proche de 1, même pour de grandes proportions de données manquantes, jusqu'à des proportions de 90%.

L'erreur considérée représente la qualité de reconstruction du tenseur $\mathcal{X}$ ceci afin de pouvoir quantifier l'influence de l'information apportée par $\mathbf{Y}$ sur cette reconstruction. Ce que nous pouvons associer à la question suivante.

L'information introduite par la matrice $\mathbf{Y}$ est-elle vraiment utile à la reconstruction du tenseur $\mathcal{X}$?

Cette question centralise l'analyse sur la qualité de l'imputation. Or il convient de considérer des cas où l'imputation n'est qu'un moyen de pouvoir efficacement répondre à la question de la prédiction de $\mathbf{Y}$. Se pose aussi la question de la validation des hyperparamètres du modèle, le nombre de composantes principalement. Classiquement, comme présenté par Josse et Husson (2012), la validation croisée est utilisée dans ce cas. Il convient pourtant de gérer la présence de données manquantes pour les données d'entraînement d'un modèle.

Louwerse et collab. (1999) décrivent deux méthodes permettant d'estimer les erreurs en validation croisée de modèles multivoies, en se basant sur le modèle Tucker3, voir Section 1.6.2 page 50. La première solution utilise un algorithme "Typé-EM" baptisé pédagogiquement EM-Tucker3. La seconde solution n'utilise pas un algorithme "Typé-EM" mais des approximations des matrices composantes du modèle Tucker3 en supprimant de façon alternée des individus puis des variables et enfin des voies. Ce modèle s'appelle *Leave-Bar-Out* (*LBO*). A chaque étape, les 3 matrices sur 4 qui ne sont pas associées au mode dont des échantillons ont été supprimées approximent les matrices associées dans le modèle global. Cette méthode permet de reconstruire 8 approximations différentes du tenseur pour lequel les données seraient manquantes dans les individus les variables et les voies de façon simultanée. Nous laissons le lecteur attentif consulter le tableau 2 page 499 des travaux de Louwerse et collab. (1999) détaillant ces modèles. Cet algorithme peut sembler complexe car, de prime abord, combinatoire mais il se justifie en réalité complètement par le modèle Tucker3. Les deux méthodes sont comparées sur la base de la somme des carrés résiduels de l'erreur en prédiction (*PRESS*) qui est la somme des erreurs quadratiques des valeurs estimées du tenseur. Pourtant le PRESS semble induire de l'*overfactoring*, discuté précédemment, comme décrit par Osten (1988) qui proposent d'utiliser la *statistique W* qui pénalise l'erreur de PRESS pour R composantes en prenant en compte l'erreur de PRESS à $R-1$ composantes et le nombre de coefficients ajoutés par cette R^{ieme} composante.

Les deux méthodes de validation croisée sont testées sur des données simulées et montrent de réelles capacités d'aide à l'estimation des paramètres du modèle Tucker3. Le critère de PRESS permet de choisir plus simplement un modèle optimal. Le modèle "Typé-EM" semble offrir de meilleurs résultats en terme de précision mais il est bien plus demandant en termes de temps de calcul et pour cette raison l'algorithme LBO est préféré par les auteurs. Ces deux méthodes permettent de réaliser la validation croisée du modèle Tucker3 en présence de données manquantes mais ne permettent pas de résoudre ce problème dans le cas supervisé, le cas du problème CMTF par exemple.

D'autres résultats (voir par exemple van Ginkel et Kroonenberg, 2017) permettent de comparer les approches "Typées-EM" telles que celles que nous venons de décrire et des méthodes d'imputation multiple, mais cela dans un mode non supervisé. Pourtant l'imputation multiple

n'est pas considérée dans le travail de cette thèse et de futurs développements pourraient nous amener à considérer leur utilisation.

Le tableau 1.1 répertorie les estimateurs considérés dans le travail de cette introduction et, dans chaque cas, énonce des propriétés essentielles.

	Supervisée ?	**Régularisante ?**	**Parcimonieuse ?**	**À composantes ?**	**Multiblock ?**	**Multivoie ?**	**Gère les *NA* ?**
MCO	v	·	·	·	·	·	·
ACP	·	v	·	v	·	·	·
NIPALS	·	v	·	v	·	·	v
sparseACP	·	v	·	v	·	·	·
Ridge (MCO)	v	v	·	·	·	·	·
Lasso (MCO)	v	v	v	·	·	·	·
Elastic-net (MCO)	v	v	v	·	·	·	·
PLASM (gLasso) (MCO)	v	v	v	·	·	·	·
sgLasso (MCO)	v	v	v	·	·	·	·
PCR	v	v	·	v	·	·	·
PLS	v	v	·	v	·	·	·
gPLS	v	v	·	v	·	·	·
sgPLS	v	v	·	v	·	·	·
MBPLS	v	v	·	v	v	·	·
NPLS	v	v	·	v	v	v	·
RDA	·	v	·	v	v	·	·
PLS-PM	·	v	·	v	v	·	·
RGCCA	·	v	·	v	v	·	·
SGCCA	·	v	v	v	v	·	·
AFM	·	v	·	v	v	·	·
PARAFAC/CP	·	v	·	v	·	v	·
Tucker3	·	v	·	v	·	v	·
Soft-Impute	·	v	·	v	·	·	v
imputeMFA	·	v	·	v	v	·	v
INDAFAC	·	v	·	v	·	v	v$^{(1)}$
CP-WOPT	·	v	·	v	·	v	v$^{(1)}$
CMTF	v	v	·	v	·	v	·
CMTF-WOPT	v	v	·	v	·	v	v$^{(1)}$
ddsPLS	v	v	v	v	v	·	v$^{(2)}$

$^{(1)}$ Pondération des *NA* à 0 dans l'estimation du modèle via l'opérateur P_Ω (voir problème (1.50) par exemple).
$^{(2)}$ Données manquantes par blocs.

TABLE 1.1 – Principales méthodes considérées dans ce travail ainsi que leurs propriétés générales.

1.8 Problématique de la thèse

Les références associées aux différents domaines de l'analyse que nous venons de survoler sont tout aussi foisonnantes et riches que ne saurait être maigre le rendu qui vient d'en être fait. Le choix de diriger cette introduction dans la direction des problèmes supervisés avec données manquantes pour structures multibloc est associé avec la problématique de cette thèse.

CMTF-WOPT semble être la méthode dont les hypothèses de travail s'approchent le plus de notre problématique. Pourtant cette dernière ne permet pas :

— de prendre en compte la structure hétérogène des données (qui peut donc être multivoies et multivoies hétérogène, sans distinction),

— d'imputer les données manquantes car elle les « met de côté »,

— de sélectionner des variables d'intérêt.

La contrainte de sélection des variables a trouvé sa solution dans l'utilisation du seuillage doux. La méthode Lasso ayant été mise de côté pour son mauvais comportement dans le contexte de données corrélées, comme présenté par exemple au travers de la figure 5. Qui sont les données les plus présentes dans les champs d'application touchés par le fléau de la grande dimension $n << p$. Les méthodes adaptées telles que l'Elastic-net n'ont pas non plus été considérées puisqu'elles amenaient un cout combinatoire supplémentaire associé aux hyperparamètres supplémentaires.

Il a de plus été remarqué (voir Lorenzo et collab., 2019b, tableau 5) que certaines méthodes d'imputation EM ou « Typées-EM », souffraient de non convergence dans certains cas. C'est pour cette raison qu'il a été fait le choix d'un algorithme particulièrement simple pour la prise en compte de l'aspect multibloc. A savoir

— trouver simplement des sous-espaces propres à chaque bloc,

— capitaliser les information précédentes afin de produire une description multibloc de la réponse $\mathbf{Y}$.

Ce genre d'algorithme a été rapidement décrit au travers du système (1.44). En plus d'alléger les calculs, ce genre d'algorithmes, comme l'AFM ou STATIS, permet des interprétations efficaces des paramètres des modèles construits.

En plus de ces précédentes contraintes, l'algorithme doit gérer la présence de données manquantes dans le jeu de données d'entraînement, comme le proposent Louwerse et collab. (1999) dans un contexte non supervisé. Mais des données manquantes peuvent aussi être présentes dans le jeu de données de test, jeu de données pour lequel l'information $\mathbf{Y}$ n'est pas accessible. Dans ce dernier cas, l'algorithme doit aussi être en mesure d'estimer les données manquantes. L'algorithme mis en place pour répondre à cette question, nommé Koh-Lanta, détaillé dans Lorenzo et collab. (2019b) et associé à ce manuscrit en partie 2, page 63. La méthode ddsPLS, qui nécessite des jeux de données complets, utilise l'algorithme Koh-Lanta afin de gérer les données manquantes. On se permet ici de donner une idée de l'algorithme Koh-Lanta, un algorithme en deux étapes qui correspondent à l'imputation des données d'entraînement et des données de test respectivement. Soit plus précisément :

— *l'étape des tribus*. Une étape itérative, « Typé-EM » donc. Les données manquantes sont initialisées dans chaque bloc. On forme autant de modèles monoblocs ddsPLS qu'il y a

de blocs. La matrice **Y** est utilisée comme covariable et chacun des blocs comme matrice réponse. Une fois les blocs complétés, un modèle ddsPLS globale est construit sur ces derniers avec cette fois-ci la matrice **Y** comme réponse. C'est le modèle objectif à l'interaction courante. De ce dernier modèle on note les variables sélectionnées. La prochaine étape consiste à rééstimer les données manquantes avec les modèles monoblocs ddsPLS, comme déjà réalisé plus haut, mais en ne conservant que les variables sélectionnées par le modèle objectif. Puis on reconstruit ce modèle objectif et l'algorithme réitère jusqu'à convergence des paramètres.

— *L'étape de réunification*. Une étape non itérative. Considérer chaque individu séparément, s'il n'a pas de données manquantes, prédire de façon classique. S'il est le lieu de données manquantes, découper ses réalisations en un ensemble de données manquantes et un ensemble de données non manquantes. L'idée est alors d'estimer les données manquantes en construisant un modèle ddsPLS avec pour matrice réponse la concaténation des blocs associés aux données manquantes de cet individu. Les covariables considérées dans ce modèle sont alors les scores associés aux blocs de données présentes. L'idée est donc de prendre l'information de ces scores qui est supposée être associée avec la direction de la réponse, via le jeu d'entraînement. Ainsi l'information qui serait introduite dans les données manquantes ne peut être qu'associée avec cette direction, celle que l'on cherche à prédire.

La partie suivante détaille cette méthode, différentes simulations qui ont été réalisées ainsi que l'application à un jeu de données réelles multiblocs de régression multiple présentant des données manquantes qui est nouveau, ainsi que deux jeux de données monoblocs sans données manquantes qui lui est bien connu par la communauté et qui présente un problème de classification à 3 classes et un problème de régression multiple.

CHAPITRE 2

LA MÉTHODE *DDSPLS*

Sommaire

Dans ce chapitre nous présentons la méthode ddsPLS qui permet la gestion de données structurées en blocs, supervisées, avec données manquantes par blocs. La méthode Koh-Lanta est aussi décrite, c'est l'algorithme dédié à la gestion des données manquantes afin d'imputer à la fois dans le jeu de données d'entraînement et dans le jeu de test.

Les fonctionnalités de cette méthode sont discutées théoriquement et au travers de simulations et d'applications sur données réelles, ceci au travers d'un article soumis, voir section 2.1. Des compléments à cet article sont disponibles en section 2.2. Le package R associé est présenté au chapitre 3 et d'autres applications sont quant à elles présentées au chapitre 4.

Remarque Notons que le cas de la classification est géré en appliquant une méthode de classification classique (`logit()` ou `lda()`) aux scores construits grâce à la dichotomisation des labels des individus (matrice représentant les n individus dont chaque colonne affiche la probabilité d'appartenance à chaque classe, donc soit 0 soit 1 dans le jeu de données d'entraînement). C'est l'astuce utilisée pour adapter la PLS aux problèmes de classification. Elle porte le nom d'analyse discriminante par PLS (*PLS-DA*).

2.1 Supervised Learning for Multiblocks Incomplete Data

Supervised Learning for Multi-Block Incomplete Data

Hadrien Lorenzo
hadrien.lorenzo@u-bordeaux.fr
SISTM INRIA BSO, Inserm-U1219 BPH
146 rue Lo Saignat 33076 Bordeaux cedex, France

Jérôme Saracco
jerome.saracco@inria.fr
CQFD INRIA Bordeaux Sud-Ouest-France, CNRS (UMR5251)
200 Avenue de la Vieille Tour, 33405 Talence, France

Rodolphe Thiébaut
rodolphe.thiebaut@u-bordeaux.fr
SISTM INRIA BSO, Inserm-U1219 BPH, Vaccine Research Institute
8 rue du Général Sarrail 94010 Créteil cedex, France

October 15, 2019

Abstract

In the supervised high dimensional settings with a large number of variables and a low number of individuals, one objective is to select the relevant variables and thus to reduce the dimension. That subspace selection is often managed with supervised tools. However, some data can be missing, compromising the validity of the sub-space selection. We propose a Partial Least Square (PLS) based method, called Multi-block Data-Driven sparse PLS "mdd-sPLS", allowing jointly variable selection and subspace estimation while training and testing missing data imputation through a new algorithm called Koh-Lanta. This method was challenged through simulations against existing methods such as mean imputation, nipals, softImpute and imputeMFA. In the context of supervised analysis of high dimensional data, the proposed method shows the lowest prediction error of the response variables. To our knowledge this is the only method combining data imputation and response variable prediction. The efficacy of the supervised multi-block mdd-sPLS method increases with the intra-block and inter-block correlations. The application to a real data-set from a rVSV-ZEBOV Ebola vaccine trial revealed interesting and biologically relevant results. The method is

implemented in a **R**-package available on the **CRAN** and a **Python**-package available on **pypi** with corresponding vignettes.

Keywords

Variable selection; Supervised learning; PLS regression; Missing data; Multi-block data.

1 Introduction

Heterogenous data can be represented as blocks where each block represents one type of data leading to multiblock analysis. In a supervised analysis, several blocks **X** could be used to explain one block **Y**. This representation is particularly useful when dealing with methods for reducing high-dimensional data. Missing data may also happen in high-dimensional settings where the number of variables is very large but not fully completed. Among the many different methodological solutions, of which a description can be found in Bertsimas et al. (2018), this paper focuses on singular value decomposition (**SVD**)-based methods with simple imputation of missing data. The **SVD**-based imputation methods assume that the eigenvectors are not too much influenced by the missing values and also can be denoised by alternating **SVD**-decompositions and imputations through linear combinations of the current eigenvectors until convergence is reached (Troyanskaya et al., 2001; Hastie et al., 2015). The **mean** imputation is used as a reference method and leads to good results in many situations. Hastie et al. (2015) propose a particularly fast ridge-regression and **SVD** soft-thresholding alternated method, designed for the mono-block context. Husson and Josse (2013) develop a multi-block method called **imputeMFA** that uses multiple axis decomposition with self-tunable ridge penalization in unsupervised context. The **nipals** algorithm, for **N**onlinear **I**terative **Pa**rtial **L**east **S**quares firstly detailed by Wold (1966), allows dealing with missing observations (see for example Nelson et al., 1996) in the context of unsupervised problems with principal component analysis (**PCA**) but also in supervised problems with the Partial Least Squares (**PLS**). Other methods not considered in this paper are not based on **SVD** such as **missForest** developed by Stekhoven and Bühlmann (2011).

All the methods existing so far and presented above are two-steps approaches where the data are first imputed and then the imputed data set is analyzed using any unrelated statistical tool. Our objective was to develop a method able to impute missing covariates and predict the response in the same time, for a multi-block supervised data set with potential missing data in the covariate part.

In the following, matrices are written with bold capital letters and vectors with bold lower-case letters. For any matrix $\mathbf{M}$, $\mathbf{m}^{(i)}$ denotes its i^{th}-column. Let $\mathbf{X} \in \mathbb{R}^{n\times p}$ and $\mathbf{Y} \in \mathbb{R}^{n\times q}$, be respectively the covariate matrix and the response matrix, describing n individuals through p (resp. q) variables. Unless otherwise stated, data matrices $\mathbf{X}$ and $\mathbf{Y}$ are supposed to be standardized (zero-mean and unit-variance). The **PLS** approach maximizes the covariance between both of the projections of $\mathbf{X}$ and $\mathbf{Y}$ on their proper weights denoted by $\mathbf{u} \in \mathbb{R}^p$ and $\mathbf{v} \in \mathbb{R}^q$ respectively. The underlying optimization problem can be written as

$$\begin{aligned} \max_{(\mathbf{u},\mathbf{v})} \quad & \mathbf{u}^T\mathbf{X}^T\mathbf{Y}\mathbf{v} \\ \text{s.t.} \quad & \mathbf{u}^T\mathbf{u} = \mathbf{v}^T\mathbf{v} = 1, \end{aligned} \tag{1}$$

for the current axis decomposition. The **nipals** permits to solve that problem. If further axes are needed, deflations are successively performed to remove the information carried by previous axes before solving the problem (1) on the corresponding residual matrices. It has been shown that (1) is equivalent to find eigenvector linked to the largest eigen-value of $\mathbf{X}^T\mathbf{Y}\mathbf{Y}^T\mathbf{X}$ such as reformulated by Höskuldsson (1988).

To deal with the variable selection problem in **PLS**, two existing methods are presented hereafter. They are based on the Lasso formulation which shrinks the $\mathcal{L}_1$-norm of the weights (see Tibshirani, 1996). Based on the SCoTLASS solution to **sparse PCA**, proposed by Jolliffe et al. (2003), Lê Cao et al. (2008) considered a $\mathcal{L}_1$-norm penalization of the $\mathbf{X}$ and $\mathbf{Y}$ weights, introducing two $\mathcal{L}$agrangian coefficients which are fixed by the user, denoted as Lasso parameters. The associated approach has been implemented in the **R**-package **mixOmics**, see Lê Cao et al. (2009), and is denoted as **classic-sPLS** in the following. As pointed out by Zou et al. (2006) and recalled by Chun and Keleş (2010), problem from Lê Cao et al. (2008) is not convex and solutions are not in practice sufficiently sparse. Chun and Keleş (2010) proposed an alternative formulation to mitigate those drawbacks. Introducing a parameter κ, they can reduce the concave part of the original problem and so reduces its impact on the global optimization problem. The authors performed simulations showing that a low κ indeed provides "a numerically easier optimization problem" but no general result was given. The main drawback is the computational cost due to the number of parameters. Furthermore, their problem allows to select only on the $\mathbf{X}$ data set while Lê Cao et al. (2008) allow to select variables on both data sets $\mathbf{X}$ and $\mathbf{Y}$. This is clearly a limit if the number q of variables in $\mathbf{Y}$ is large and so tends to draw uncorrelated subspaces.

Other variable selection methods have been recently studied, such as the **SVD** decomposition of thresholded variance matrices, developed to tackle the **sparse PCA** question. d'Aspremont et al. (2005) have developed an elegant convex re-

laxed optimization problem and detailed an algorithm to solving it. Subsequently, Amini and Wainwright (2008) have compared that **SDP** solution to the Diagonal Thresholding (**DT**) method, developed by Johnstone and Lu (2004) and Johnstone and Lu (2009). This method showed comparable results with higher computational efficiency. Different types of threshold operators are considered and studied for example by Rothman et al. (2009) and Johnstone and Lu (2009) and more recently by Cai and Liu (2011). Deshpande and Montanari (2016) detailed an algorithm in which the **SVD** is performed on the soft-thresholded variance-covariance matrix based on results of Krauthgamer et al. (2015) for which element-wise hard-thresholding was considered. Strong theoretical results about selectivity and consistency exist for those approaches and Deshpande and Montanari (2016) have proved the consistency of the soft-thresholding covariance matrix as an estimator of the covariance matrix in the high-dimensional context, $n << p$, and using a spiked model hypothesis on the covariable $\mathbf{X}$ for which the authors seeks a sparse estimation of the variance-covariance matrix. In the present work, the matrix $\mathbf{Y^T X}$, which is at the core of the **PLS** problem through its **SVD**, has been modified under soft-thresholding manipulations which is justified by some of the works cited above and detailed below.

In regards of the context of multi-block high-dimensional data, several supervised approaches inspired by the PLS have been proposed. The covariate part is defined through T different matrices $\mathbf{X}_1, ..., \mathbf{X}_T$ describing the same n individuals. The adaptation of the **PLS** method to the multi-block structure has been initially proposed through the "SW-Harald-HW multi-block algorithm" by Wold (1984). A few years later Wangen and Kowalski (1989) detailed this approach today known as **MBPLS** (for Multi-Block Partial Least Square) through the optimization problem

$$\begin{aligned} \max_{(\mathbf{u},\boldsymbol{\beta},\mathbf{v})} \quad & \sum_{t=1}^{T} \beta_t \mathbf{v}^T \mathbf{Y}^T \mathbf{X}_t \mathbf{u}_t \\ \text{s.t.} \quad & \mathbf{u}_t^T \mathbf{u}_t = \mathbf{v}^T \mathbf{v} = \boldsymbol{\beta}^T \boldsymbol{\beta} = 1, \end{aligned}$$

where the $\beta_t' s$ gathering the information from the T blocks $\mathbf{X}_t$ via their weight $\mathbf{u}_t$, and they make it possible to build the super-component $\sum_{t=1}^{T} \beta_t \mathbf{X}_t \mathbf{u}_t$. **MBPLS** is a **nipals** flavored method, using deflation procedures to obtain further axes. Initially the deflation of each block was made on its proper component, but Westerhuis et al. (1997) have shown the interest, in terms of prediction, to deflate each block on the super-component. Many authors have decided to challenge that question of deflation (see for example Westerhuis and Smilde, 2001). Supposing that each block has been divided by its square root number of variables, Qin et al. (2001) have demonstrated the similarities of the **MBPLS** problem with a classical **PLS** problem. Westerhuis and Smilde (2001) have rewritten the **MBPLS** problem by re-weighting a

standard **PLS** model built on the concatenated matrix of the T blocks. Bougeard et al. (2011) implemented the **MBPLS** algorithm in the **R**-package **ade4** with the super-component deflation version of Westerhuis and Smilde (2001). Bougeard et al. (2011) bind the **MBPLS** problem to the **RA** problem (Redundancy Analysis) defined through the same criterion as **PLS**, covariance maximization of the covariates projection and response projections, but here weights $(\mathbf{u}_t)_{t\in[\![1,T]\!]}$ are not directly constrained while components $(\mathbf{X}_t\mathbf{u}_t)_{t\in[\![1,T]\!]}$ are constrained to $\mathcal{L}_2$-norm equal to 1. Their solution uses a regularization by convexly balance the power of the variance-covariance matrix of each block towards the identity matrix, and permitting to solve the **MBRA** (Multi-Block Redundancy Analysis) problem in the context of badly conditioned matrices. That solution has been generalized to the canonical correlation analysis by Tenenhaus and Tenenhaus (2011) and the corresponding method is called **RGCCA**. A sparse version of that method has been developed and detailed by Tenenhaus et al. (2014) using $\mathcal{L}_1$-norm regularization of the weights. Those methods use **nipals** typed algorithms and the authors demonstrated their monotonically convergences.

Here, we propose a method to deal with variable selection on multi-block data structured with a specific case of missing data (some entire block rows are missing). The proposed approach which stands for **mdd-sPLS** for multi-block data-driven sparse PLS, the term data-driven has been chosen because of the highly interpretable nature of the penalization parameter, it corresponds to the minimum correlation accepted between **X** and **Y** variables to put it in the model. It shows good theoretical performances on variable selection and regularization capacities. Simulations and applications to real data sets exhibit its practical, interpretable and numerical interests in comparison to four baseline methods in the presence of missing samples. The main originality of the proposed method is to deal with missing data while finding the best predictors of an outcome in the context of multiblock data.

The paper is organized as follows. Section 2 describes the covariance thresholding sparse **PLS**, called **ct-sPLS** when there are no missing values. Useful theoretical results are provided in this context. Section 3 details the proposed multi-block approach (**mdd-sPLS**) and the chosen algorithm to deal with missing data, denoted **Koh-Lanta**. Section 4 studies the numerical behavior of the **mdd-sPLS** approach through simulations. Section 5 provides results on a real data set from an Ebola rVSV phase 1 vaccine trial. Concluding remarks are given in Section 6. The method is implemented in the **R**-package `https://cran.r-project.org/package=ddsPLS` and in the **Python**-package `https://pypi.org/project/py_ddspls/`. Some examples are tested and are documented in a **R**-vignette at `https://hadrienlorenzo.netlify.com/html/ddspls` and in a **Python**-vignette at `https://pypi.org/project/py_ddspls/`.

2 Covariance-Thresholding Sparse PLS (ct-sPLS)

In this section, we focus on the mono-block context associated with the two data matrices $\mathbf{X} \in \mathbb{R}^{n\times p}$ and $\mathbf{Y} \in \mathbb{R}^{n\times q}$ without missing data. First, let us define the soft-thresholding operator, applied term to term to a matrix, as

$$S_\lambda : \quad x \to \text{sign}(x)(|x| - \lambda)_+,$$

where $\lambda \in [0,1]$ is a regularization parameter and $(.)_+ = \max(0,.)$. Thus, the matrix $S_\lambda(\mathbf{Y}^T\mathbf{X}/(n-1))$ is the soft-thresholded version of the empirical correlation matrix between $\mathbf{X}$ and $\mathbf{Y}$ (since these matrices are assumed to be standardized) with respect to the threshold λ.

The proposed **ct-sPLS** problem, written for a R-dimensional decomposition, is defined as

$$\begin{array}{ll} \max\limits_{\mathbf{U}\in\mathbb{R}^{p\times R}} & \sum_{r=1}^{R} ||S_\lambda\left(\frac{\mathbf{Y}^T\mathbf{X}}{n-1}\right)\mathbf{u}^{(r)}||_2^2 \\ \text{s.t.} & \mathbf{U}^T\mathbf{U} = \mathbb{I}_R. \end{array} \tag{2}$$

The solution of problem (2) is

$$\mathbf{U} = \mathbf{SVD}_R\Big(S_\lambda\left(\frac{\mathbf{Y}^T\mathbf{X}}{n-1}\right)\Big) =: \mathbf{ct\text{-}sPLS}(\mathbf{X},\mathbf{Y},\lambda,R) \in \mathbb{R}^{p\times R},$$

where $\mathbf{SVD}_R(\mathbf{M})$ gives the R first right-singular-vectors of $\mathbf{M}$. The R components rely on the same regularization parameter λ, which can be obtained via cross-validation. Note that there is no deflation step for the construction of the R different axes constructions. Similarly, let $\mathbf{V} \in \mathbb{R}^{q\times R}$ be the R first left singular vectors of $S_\lambda(\mathbf{Y}^T\mathbf{X}/(n-1))$. The associated components of $\mathbf{X}$ and $\mathbf{Y}$ are respectively denoted by $\mathbf{T} = \mathbf{XU} \in \mathbb{R}^{n\times R}$ and $\mathbf{S} = \mathbf{YV} \in \mathbb{R}^{n\times R}$.

It is relevant to have a regression model in the context of supervised learning in order to explain $\mathbf{Y}$ by $\mathbf{X}$. In the following a linear approximation, $\mathbf{Y} \approx \mathbf{XB}$, is shown to be reasonable, where $\mathbf{B}$ is constructed using the **ct-sPLS** components $\mathbf{T}$ and $\mathbf{S}$.

Some theoretical results are formulated hereafter. Section 2.1 proves that **ct-sPLS** method is a sparse method. Section 2.2 provides consistency results. This implies the proposed method indeed performs regularization over the data and also allows building the associated regression model based on the **ct-sPLS** components.

2.1 Sparsity

The following theorem illustrates the ability of the **ct-sPLS** method to provide a sparse solution $\mathbf{U}$.

Theorem 2.1. *Let* $(\mathbf{X}, \mathbf{Y}) \in \mathbb{R}^{n\times p} \times \mathbb{R}^{n\times q}$ *assumed to be standardized. Let* $\lambda \in [0,1]$ *such as* $\mathbf{S} = S_\lambda\left(\frac{\mathbf{X}^T\mathbf{X}}{n-1}\right)$ *is not null and* $\mathbf{u} \in \mathbb{R}^p$ *and* $\mathbf{v} \in \mathbb{R}^q$ *respectively the right and left singular vectors associated to the same, non null, singular value denoted* $\sqrt{\theta}$*. Then, we have*

$$\forall i = [\![1, p]\!] : \quad < \mathbf{s}^{(i)}, \mathbf{v} >= 0 \iff u_i = 0$$

where u_i *stands for the* i^{th}*-element of* $\mathbf{u}$ *and* $\mathbf{s}^{(i)}$ *stands for the* i^{th}*-column of* $\mathbf{S}$*.*

Proof. According to the definition of $\mathbf{u}$, $\mathbf{v}$ and $\sqrt{\theta}$, we get $u_i = 0 \iff (\mathbf{S}^T\mathbf{S}\mathbf{u})_i = \theta u_i = 0$. Moreover, $(\mathbf{S}^T\mathbf{S}u)_i = {\mathbf{s}^{(i)}}^T\mathbf{S}\mathbf{u} =< \mathbf{s}^{(i)}, \mathbf{S}\mathbf{u} >$. Since $\mathbf{S}\mathbf{u} = \sqrt{\theta}\mathbf{v}$, by definition of the left and right singular vectors, we get $< \mathbf{s}^{(i)}, \mathbf{S}\mathbf{u} >=< \mathbf{s}^{(i)}, \sqrt{\theta}\mathbf{v} >= \sqrt{\theta} < \mathbf{s}^{(i)}, \mathbf{v} >= 0 \iff < \mathbf{s}^{(i)}, \mathbf{v} >= 0$. □

In Theorem 2.1, the nullity of an element of weights $\mathbf{u}$ associated with $\mathbf{X}$ is the only case considered. Equivalent results can straightforwardly obtained for the weights $\mathbf{v}$ associated with $\mathbf{Y}$ considering $\mathbf{S}^T$ instead of $\mathbf{S}$. This implies that the **ct-sPLS** method simultaneously selects variables in the $\mathbf{X}$ part and in the $\mathbf{Y}$ part. The condition obtained in the previous theorem is computationally unacceptable for the user and in practice **ct-sPLS** user would appreciate an upper-bound to the cardinality of $\mathbf{u}$ and $\mathbf{v}$, the number of variables selected for the $\mathbf{X}$ part and for the $\mathbf{Y}$ part respectively. The next corollary indicates that for all λ in $[0,1]$, there exists an easy to compute upper bound to the number of variable selected for $\mathbf{X}$ and for $\mathbf{Y}$. Let $\mathbf{Card}(\mathbf{u}) = \#\{\text{Non null elements in } \mathbf{u}\}$.

Corollary 2.1.1. *Under assumptions of Theorem 2.1, we have*

$$\mathbf{Card}(\mathbf{u}) \leqslant p - \#\{\textit{Null columns of } S_\lambda\left(\frac{\mathbf{X}^T\mathbf{X}}{n-1}\right)\},$$
$$\mathbf{Card}(\mathbf{v}) \leqslant q - \#\{\textit{Null rows of } S_\lambda\left(\frac{\mathbf{X}^T\mathbf{X}}{n-1}\right)\}.$$

Proof. Applying Theorem 2.1 and counting cases where the i^{th} column (respectively the j^{th} row) of $S_\lambda\left(\frac{\mathbf{X}^T\mathbf{X}}{n-1}\right)$ are filled with null elements only, then the corresponding $\mathbf{u}$ (respectively $\mathbf{v}$) coefficients are equal to 0, which demonstrates the corollary. □

The previous corollary is applied, for a given λ, to any singular-space of $S_\lambda(\frac{\mathbf{X}^T\mathbf{X}}{n-1})$. This is only data-dependent, under the λ user appreciation. Note that the cardinality of any axis of $\mathbf{U}$ and $\mathbf{V}$ is not monotonic. In Appendix A, a simulation study exhibits a particular case, for $R = 1$ considering the singular-space associated with the largest singular-value, in which the cardinality of $\mathbf{u}$ is not monotonic.

2.2 Consistency

Under additional (theoretical) assumptions, we will show that it is relevant to consider a regression model in order to explain $\mathbf{Y}$ by $\mathbf{X}$ using the **ct-sPLS** components $\mathbf{T} = \mathbf{YV}$ and $\mathbf{S} = \mathbf{XU}$.

As described by Johnstone and Lu (2004), let us consider that $\mathbf{X}$ and $\mathbf{Y}$ follow two spiked covariance models:

$$\begin{cases} \mathbf{X} = \mathbf{L}\boldsymbol{\Omega}_x^{1/2}\mathbf{U}_{mod}^T + \mathbf{E}_x \\ \mathbf{Y} = \mathbf{L}\boldsymbol{\Omega}_y^{1/2}\mathbf{V}_{mod}^T + \mathbf{E}_y \end{cases}, \tag{3}$$

where $\boldsymbol{\Omega}_x$ and $\boldsymbol{\Omega}_y$ are R-dimensional diagonal matrices with strictly positive diagonal elements, $\mathbf{U}_{mod} \in \mathbb{R}^{p\times R}$ and $\mathbf{V}_{mod} \in \mathbb{R}^{q\times R}$ are two matrices with orthonormal columns, $\mathbf{L} \in \mathbb{R}^{n\times R}$ is a matrix where elements are i.i.d. standard Gaussian random effects, $\mathbf{E}_x \in \mathbb{R}^{n\times p}$ (resp. $\mathbf{E}_y \in \mathbb{R}^{n\times q}$) is a matrix such that each row follows the standard multivariate normal distribution $N_p(0, \mathbb{I}_p)$ (resp. $N_q(0, \mathbb{I}_q)$) and the n rows are independent and mutually independent noise vectors. Note that $\mathbf{L}$ introduces a common structure to $\mathbf{X}$ and $\mathbf{Y}$ models. The intuition developed by Deshpande and Montanari (2016) is applied to $\mathbf{Y^T X}/(n-1)$ (instead of $\mathbf{X^T X}/n - \mathbb{I}_p$ in the original paper) and permits to use their Theorem 1 inferring the consistency of $S_\lambda(\mathbf{Y^T X}/(n-1))$ to the population covariance $\mathbf{V}_{mod}\boldsymbol{\Omega}_y^{1/2}\boldsymbol{\Omega}_x^{1/2}\mathbf{U}_{mod}^T$. That result is of prior importance because it implies that the solution of the optimization problem (2) is such that

- $S_\lambda(\mathbf{Y^T X}/(n-1))\mathbf{U}$ and $\mathbf{Y^T XU}$ span approximately the same R-dimensional subspace of $\mathbb{R}^q$,
- $S_\lambda(\mathbf{X^T Y}/(n-1))\mathbf{V}$ and $\mathbf{X^T YV}$ span approximately the same R-dimensional subspace of $\mathbb{R}^p$.

Therefore $\mathbf{U}$ and $\mathbf{V}$ permit to find a cross-subspace of $\mathbb{R}^n$ such as $\mathbf{S} = \mathbf{XU}$ and $\mathbf{T} = \mathbf{YV}$ span approximately the same R-dimensional subspace of $\mathbb{R}^n$. From this result, it is then possible to construct a regression model to explain $\mathbf{Y}$ from $\mathbf{X}$ such as $\mathbf{Y} \approx \mathbf{XB}$. The matrix $\mathbf{B}$, obtained from the **ct-sPLS** components $\mathbf{T}$ and $\mathbf{S}$, is detailed in the multi-block framework, see Section 3.2.

3 Multi-Block Data-Driven sparse PLS (mdd-sPLS)

In this section, let us focus on the multi-block context associated with the T data matrices $\mathbf{X}_t \in \mathbb{R}^{n\times p_t}, t = 1\ldots, T$ and a single response matrix $\mathbf{Y} \in \mathbb{R}^{n\times q}$. Here missing samples might be in any of the T blocks $\mathbf{X}_t$, which means that any row of any block of covariates can be missing, but not in the response matrix $\mathbf{Y}$.

The idea of the multi-block data-driven sPLS (**mdd-sPLS**) is to use **ct-sPLS** properties to find structure between the $\mathbf{X}_t$'s and the response matrix $\mathbf{Y}$ including variable selection in the context of missing samples. Contrary to some existing multi-block approaches, the **mdd-sPLS** does not consider correlations between covariate blocks covariances but only covariate block $\mathbf{X}_t$ and response block $\mathbf{Y}$ covariances. Two sequential optimization problems are then defined, in terms of intra/inter-blocks weights, particularly stable and efficient in terms of prediction. Note that the proposed method is non "component-wise" in the sense that all the axes are built together and no deflation is used as to answer variance constraint problems discussed in the introduction and inherent to the deflation. Since missing samples can be both in the train sample and in the test sample, a new specific algorithm, called **Koh-Lanta** hereafter, has been developed. Note that this algorithm has been designed for the **mdd-sPLS** approach but can be extended to any supervised multi-block method.

Sections 3.1 and 3.2 provide the description of the **mdd-sPLS** method. The corresponding algorithms are given in Section 3.3. The imputation algorithm **Koh-Lanta** is described in Section 3.4. The computational complexity of **mdd-sPLS** is studied in Section 3.5. Finally the adaptation to classification (rather than regression) is discussed and illustrated in Section 3.6.

3.1 Description

Let $R \in [\![0, \min(\min_{t\in[\![1,T]\!]}(\text{rank}(\mathbf{X}_t)), \text{rank}(\mathbf{Y}))]\!]$. The **mdd-sPLS** approach splits into three steps.

- STEP 1: SOLVE INDEPENDENTLY THE T CT-SPLS PROBLEMS BASED ON $\mathbf{X}_t$ AND $\mathbf{Y}$.

  This provides the weights $(\mathbf{U}_t)_{t\in[\![1,T]\!]}$

  $$\mathbf{U}_t := \textbf{ct-sPLS}(\mathbf{X}_t, \mathbf{Y}, \lambda, R) = \textbf{SVD}_R(\mathbf{M}_t) \in \mathbb{R}^{p_t \times R},$$

  where $\mathbf{M}_t = S_\lambda\left(\frac{\mathbf{Y}^T\mathbf{X}_t}{n-1}\right)$.

- STEP 2: COMBINE INFORMATION FROM THE T BLOCKS.

  Let $\mathbf{Z} = [\mathbf{M}_1\mathbf{U}_1, \cdots, \mathbf{M}_T\mathbf{U}_T]$. In order to combine information from all the blocks and to recover a common R-dimensional subspace, we introduce the super-weights $(\underline{\boldsymbol{\beta}}_t)_{t\in[\![1,T]\!]} \in \mathbb{R}^{R\times R}$ associated with block t defined as

  $$\underline{\boldsymbol{\beta}} = \left[\underline{\boldsymbol{\beta}}_1^T, \cdots, \underline{\boldsymbol{\beta}}_T^T\right]^T = \textbf{SVD}_R(\mathbf{Z}) \in \mathbb{R}^{RT\times R}.$$

- STEP 3: PREDICT $\mathbf{Y}$ USING A REGRESSION MODEL.

The weights $(\mathbf{U}_t)_{t\in[\![1,T]\!]}$ and the super-weights $(\underset{\sim}{\boldsymbol{\beta}}_t)_{t\in[\![1,T]\!]}$ are used to build the matrices $(\mathbf{B}_t)_{t\in[\![1,T]\!]}$ approximating the regression model

$$\mathbf{Y} \approx \sum_{t=1}^{T} \mathbf{X}_t \mathbf{B}_t \in \mathbb{R}^{n\times q}, \tag{4}$$

This step is detailed in the following section.

Remark. *If there exists a covariate matrix $\mathbf{X}_{t*}$ such that $S_\lambda\big(\mathbf{Y}^T\mathbf{X}_{t*}/(n-1)\big) = \mathbf{0}$, the corresponding solution to the first step of **mdd-sPLS** is the null matrix. The same process is applied if more that one matrix is null. If all the matrices are null, the solution is the empty solution.*

3.2 Details on Method's Third Step

The underlying regression model is constructed with the same statistical assumptions as described by Johnstone and Lu (2004), already used in (3) in the monoblock (**ct-sPLS**) context and denoted as spike covariance models

$$\begin{cases} \mathbf{X}_1 = \mathbf{L}\boldsymbol{\Omega}_1^{1/2}\mathbf{U}_{1,mod}^T + \mathbf{E}_1 \\ \quad\vdots \\ \mathbf{X}_T = \mathbf{L}\boldsymbol{\Omega}_T^{1/2}\mathbf{U}_{T,mod}^T + \mathbf{E}_T \\ \mathbf{Y} = \mathbf{L}\boldsymbol{\Omega}_y^{1/2}\mathbf{V}_{mod}^T + \mathbf{E}_y \end{cases}$$

where $(\boldsymbol{\Omega}_t)_{t\in[\![1,T]\!]}$ and $\boldsymbol{\Omega}_y$ are R-dimensional diagonal matrices with strictly positive diagonal elements. $(\mathbf{U}_{t,mod} \in \mathbb{R}^{p_t\times R})_{t\in[\![1,T]\!]}$ and $\mathbf{V}_{mod} \in \mathbb{R}^{q\times R}$ are matrices with orthonormal columns. $\mathbf{L} \in \mathbb{R}^{n\times R}$ is a matrix where elements are i.i.d. standard Gaussian random effects, $(\mathbf{E}_t \in \mathbb{R}^{n\times p_t})_{t\in[\![1,T]\!]}$ (respectively $\mathbf{E}_y \in \mathbb{R}^{n\times q}$) are matrices such that each row follows the standard multivariate normal distribution $(N_{p_t}(0,\mathbb{I}_{p_t}))_{t\in[\![1,T]\!]}$ (respectively $N_q(0,\mathbb{I}_q)$) and the n rows are independent and mutually independent noise vectors. Let us mention that the matrix $\mathbf{L}$ does not depend of t and thus introduces a common structure between the $\mathbf{X}_t$'s and $\mathbf{Y}$ models. Moreover let us recall that the matrix $\boldsymbol{\beta}$ of super-weights gathers information from the different covariates corresponding to the different blocks.

Under these assumptions, using Theorem 1 from Deshpande and Montanari (2016) and extending the context of Section 2.1 to T soft-thresholded matrices $S_\lambda(\mathbf{Y}^T\mathbf{X}_t/(n-1))$, Theorem 1 insures that $\mathbf{U}_t \in \mathbb{R}^{p_t\times R}$ (the optimal R-dimensional right-singular matrix of $S_\lambda(\mathbf{Y}^T\mathbf{X}_t/(n-1))$) is close to the optimal R-dimensional right-singular matrix of $\mathbf{Y}^T\mathbf{X}_t$.

Let us introduce the following notations

$$\begin{cases} \mathbf{U}_{t,super} &= \mathbf{U}_t\boldsymbol{\beta}_t \in \mathbb{R}^{p_t\times R} \\ \mathbf{T}_{super} &= \sum_{t=1}^{T} \mathbf{X}_t\mathbf{U}_{t,super} \in \mathbb{R}^{n\times R} \\ \mathbf{V}_{super} &= norm_2(\mathbf{Z}\boldsymbol{\beta}) \in \mathbb{R}^{q\times R} \\ \mathbf{S}_{super} &= \mathbf{Y}\mathbf{V}_{super} \in \mathbb{R}^{n\times R} \end{cases},$$

where $norm_2$ is the function that returns the columns normalized to a $\mathcal{L}_2$-norm equal to 1 if the corresponding column is non null and 0 otherwise.

The aim is to provide weights that favor the most predictive directions $\mathbf{u}_t^{(r)}$ for $\mathbf{Y}$. The matrix $\mathbf{U}_{t,super}$ is the super-weight corresponding to the t^{th}-block. The matrix $\mathbf{T}_{super}$ is the super-component for covariate part, which is the most predictive of $\mathbf{Y}$. The matrix $\mathbf{V}_{super}$ is the weight enabling to build the component $\mathbf{S}_{super}$ of the response part. The component $\mathbf{T}_{super}$ and $\mathbf{S}_{super}$ describes the n individuals from the point of view of the covariates and the response. Let us write the regression model

$$\mathbf{S}_{super} = \mathbf{T}_{super}\mathbf{B}_{0,mod} + \mathbf{E}_{ort}, \tag{5}$$

where $\mathbf{E}_{ort} \in \mathbb{R}^{n\times R}$ a residual matrix and $\mathbf{B}_{0,mod} \in \mathbb{R}^{R\times R}$ the matrix of parameters. Freedom is given to the user to determine if the model is sufficiently informative. Let us introduce $\mathbf{V}_{ort} = \mathbf{SVD}_R(\mathbf{T}_{super})$ and $\boldsymbol{\Delta}_{ort}$ the diagonal matrix filled with the corresponding square singular values. Using Moore-Penrose pseudo-inverse of ${\mathbf{T}_{super}}^T\mathbf{T}_{super}$, the parameters $\mathbf{B}_{0,mod}$ can be estimated by

$$\mathbf{B}_0 = \mathbf{V}_{ort}\boldsymbol{\Delta}_{ort}^{-1}\mathbf{V}_{ort}^T\mathbf{T}_{super}^T\mathbf{S}_{super},$$

which is the best solution in the regression problem (5) according to Penrose (1956). One can therefore rewrite (5) as

$$\mathbf{S}_{super} = \mathbf{Y}\mathbf{V}_{super} \approx \sum_{t=1}^{T} \mathbf{X}_t\mathbf{U}_{t,super}\mathbf{B}_0.$$

And so

$$\mathbf{Y} \approx \mathbf{Y}\mathbf{V}_{super}\mathbf{V}_{super}^T \approx \sum_{t=1}^{T} \mathbf{X}_t\mathbf{U}_{t,super}\mathbf{B}_0\mathbf{V}_{super}^T.$$

This approximation is discussed in the next remark. Finally, by identification with (4), one obtains

$$\mathbf{B}_t = \mathbf{U}_{t,super}\mathbf{B}_0\mathbf{V}_{super}^T. \tag{6}$$

Remark. *Since the application* $\mathbf{Y} \rightarrow \mathbf{Y}\mathbf{V}_{super}\mathbf{V}_{super}^T$ *is a projection on the common subspace between the response and the covariate matrices, it is usual (see for*

example Manne, 1987) to write, instead of (6)*, the approximation* $\mathbf{Y} \approx \sum_{t=1}^{T} \mathbf{X}_t \mathbf{B}_t$ *already introduced in (4). This notation takes into account that no better regression model is accessible in the context of* ***PLS*** *modelling. The proposed method allows to know which variable of the* $\boldsymbol{Y}$ *part is indeed predicted by the model, since the Theorem 2.1 insures the* **mdd-sPLS** *method selects variables both in the* $\mathbf{X}_t$*'s and in* $\boldsymbol{Y}$*.*

3.3 Algorithms of the Multi-Data-Driven-sPLS

First, the algorithm of the **ct-sPLS** method is provided in Algorithm 1. Then, the algorithm of **mdd-sPLS** approach is described in Algorithm 2. In the following, $\odot$ is the Hadamard product $\mathbf{A}//\mathbf{B}$ denotes the term-to-term division operator of matrices (or vectors) $\mathbf{A}$ and $\mathbf{B}$. Let $\mathbf{1}_n$ denote the n-dimensional vector of 1's and *diag* which returns, for a given square matrix $\mathbf{X}$, the row vector of the diagonal elements of $\mathbf{X}$. The notation $\mathcal{M}[.]$ makes it possible to extract any attribute of the considered object obtained from **ct-sPLS** or **mdd-sPLS** method, the available attributes correspond to the outputs of the corresponding algorithms.

Algorithm 1 ct-sPLS

procedure CT-sPLS($\mathbf{X}, \mathbf{Y}, \lambda, R$)
  $(\boldsymbol{\mu}_x, \boldsymbol{\mu}_y) \leftarrow \frac{1}{n}(\mathbf{1}_n^T \mathbf{X}, \mathbf{1}_n^T \mathbf{Y})$
  $(\boldsymbol{\sigma}_x^2, \boldsymbol{\sigma}_y^2) \leftarrow \frac{1}{n-1}\Big(\mathbf{1}_n^T\big((\mathbf{X} - \mathbf{1}_n\boldsymbol{\mu}_x) \odot (\mathbf{X} - \mathbf{1}_n\boldsymbol{\mu}_x)\big), \mathbf{1}_n^T\big((\mathbf{Y} - \mathbf{1}_n\boldsymbol{\mu}_y) \odot (\mathbf{Y} - \mathbf{1}_n\boldsymbol{\mu}_y)\big)\Big)$
  $\mathbf{X} \leftarrow (\mathbf{X} - \mathbf{1}_n\boldsymbol{\mu}_x)//(\mathbf{1}_n\boldsymbol{\sigma}_x)$
  $\mathbf{Y}_i \leftarrow (\mathbf{Y}_i - \boldsymbol{\mu}_y)//(\mathbf{1}_n\boldsymbol{\sigma}_y)$
  $\mathbf{M} \leftarrow S_\lambda\big(\frac{\mathbf{Y}^T\mathbf{X}}{n-1}\big)$
  $\mathbf{U} \leftarrow \mathbf{SVD}_R(\mathbf{M})$
  $\mathbf{V} \leftarrow norm_2(\mathbf{MU})$
  return $(\mathbf{U}, \mathbf{V}, \mathbf{M}, \boldsymbol{\mu}_x, \boldsymbol{\sigma}_x, \boldsymbol{\mu}_y, \boldsymbol{\sigma}_y, \mathbf{X}, \mathbf{Y})$
end procedure

Let $\mathcal{M}$ be the **mdd-sPLS** model built on train data sets

$$(\mathbf{X}_1^{(train)}, \ldots, \mathbf{X}_T^{(train)}, \mathbf{Y}^{(train)}).$$

Given test data sets $\mathbf{X}^{(test)} := (\mathbf{X}_1^{(test)}, \ldots, \mathbf{X}_T^{(test)})$ of m individuals, the prediction operator, denoted by $\mathcal{P}$, allows to estimate the response $\mathbf{Y}^{(test)}$ by

$$\mathcal{P}(\mathbf{X}^{(test)}, \mathcal{M}) = \Big(\sum_{t=1}^{T}\big((\mathbf{X}_{t,i}^{(test)} - \mathcal{M}[\boldsymbol{\mu}_x]_t) \odot (\mathcal{M}[\boldsymbol{\sigma}_y]//\mathcal{M}[\boldsymbol{\sigma}_x]_t)\big)\mathcal{M}[\mathcal{B}]_t + \mathcal{M}[\boldsymbol{\mu}_y]\Big)_{i \in [\![1,m]\!]},$$

Algorithm 2 mdd-sPLS

procedure MDD-sPLS($\mathbf{X} = \{\mathbf{X}_t\}_{t\in[\![1,T]\!]}, \mathbf{Y}, \lambda, R$)
  for $t \in [\![1,T]\!]$ **do**
    $\mathcal{M}_t \leftarrow$ `ct-sPLS`$(\mathbf{X}_t, \mathbf{Y}, \lambda, R)$
  end for
  $(\boldsymbol{\mu}_x, \boldsymbol{\mu}_y) \leftarrow \left((\mathcal{M}_t[\boldsymbol{\mu}_x])_{t\in[\![1,T]\!]}, \mathcal{M}_1[\boldsymbol{\mu}_y]\right)$
  $(\boldsymbol{\sigma}_x, \boldsymbol{\sigma}_y) \leftarrow \left((\mathcal{M}_t[\boldsymbol{\sigma}_x])_{t\in[\![1,T]\!]}, \mathcal{M}_1[\boldsymbol{\sigma}_y]\right)$
  $\mathbf{M} \leftarrow \left(\mathcal{M}_t[\mathbf{M}]\right)_{t\in[\![1,T]\!]}$
  $\boldsymbol{\beta} \leftarrow \mathbf{SVD}_R(\mathbf{Z})$
  $\overline{\forall} t \in [\![1,T]\!], \quad \mathbf{U}_{t,super} \leftarrow \mathcal{M}_t[\mathbf{U}]\underline{\boldsymbol{\beta}}_t$
  $\mathbf{T} \leftarrow \left(\mathbf{X}_t\mathbf{U}_{t,super}\right)_{t\in[\![1,T]\!]}$
  $\mathbf{T}_{super} \leftarrow \sum_{t=1}^{T} \mathbf{X}_t\mathbf{U}_{t,super}$
  $\mathbf{V}_{super} \leftarrow norm_2(\mathbf{Z}\underline{\boldsymbol{\beta}})$
  $\mathbf{S}_{super} \leftarrow \mathbf{Y}\mathbf{V}_{super}$
  $\mathbf{V}_{ort} \leftarrow \mathbf{SVD}_R(\mathbf{T}_{super})$
  $\boldsymbol{\Delta}_{ort} \leftarrow (\mathbf{T}_{super}\mathbf{V}_{ort})^T\mathbf{T}_{super}\mathbf{V}_{ort}$
  $\mathbf{B}_0 \leftarrow \mathbf{V}_{ort}\boldsymbol{\Delta}_{ort}^{-1}\mathbf{V}_{ort}^T\mathbf{T}_{super}^T\mathbf{S}_{super}$
  $\mathcal{B} \leftarrow \left(\mathbf{U}_{t,super}\mathbf{B}_0\mathbf{V}_{super}^T\right)_{t\in[\![1,T]\!]}$
  $\mathbf{U} \leftarrow (\mathcal{M}_t[\mathbf{U}])_{t\in[\![1,T]\!]}$
  $\mathbf{V} \leftarrow (\mathcal{M}_t[\mathbf{V}])_{t\in[\![1,T]\!]}$
  return $(\mathbf{U}, \mathbf{V}, \mathbf{T}, \mathbf{T}_{super}, \mathbf{S}_{super}, \mathcal{B}, \mathbf{M}, \boldsymbol{\mu}_x, \boldsymbol{\sigma}_x, \boldsymbol{\mu}_y, \boldsymbol{\sigma}_y)$
end procedure

where $\mathbf{X}_{t,i}^{(test)}$ denotes the row vector containing the information in the data set $\mathbf{X}^{(test)}$ relative to block t and individual i. The treatment of missing values is described in the next section through the **Koh-Lanta** algorithm. In the mono-block case without missing values, the classical regression data set, called **liver toxicity**, has been used to illustrate the behavior of the method and the corresponding results are provided in Appendix B.

3.4 Koh-Lanta Algorithm: Impute Train & Test Data Sets

For most machine learning procedures, the user splits the available data into a train data set used to build the model (in order to recover the underlying structure) and a test data set allowing to evaluate the validity and the precision of the model. Let m denote the number of individuals in the train data set. Without loss of generality the m first individuals among the n individuals are in the train data set. In the following, mathematical symbols powered by a o concern the train part and mathematical

symbols powered by b concern the test part. Missing samples can appear in the train and/or in the test data set. A visual presentation of the data is given in Figure 1.

For any data set $\mathbf{X}$ the following notation $\mathbf{X}_{\texttt{blocks|indiv|variables}}$ allows to extract the information from $\mathbf{X}$ relative to the blocks indexed by `blocks`, to the individuals indexed `indiv` and to the variables indexed by `variables`. The use of a dot as index for any of those indices means that all the indexes are taken into account. For instance, $\mathbf{X}_{\texttt{blocks|indiv|.}}$ takes all the variables for blocks in `blocks` and individuals in `indiv`. To distinguish train and test parts of the data set, the following notations are introduced

Train part			Test part		
$\mathcal{I}^o$	$=$	$\{i = 1, \ldots, m\}$	$\mathcal{I}^b$	$=$	$\{i = (m+1), \ldots, n\}$
$\mathcal{I}_t^o$	$=$	$\{i \in \mathcal{I}_o \vert \mathbf{X}_{t\vert i\vert .}$ is missing$\}$	$\mathcal{I}_t^b$	$=$	$\{i \in \mathcal{I} \vert \mathbf{X}_{t\vert i\vert .}$ is missing$\}$
$\mathcal{J}_t^o$	$=$	$\{i \in \mathcal{I}_o \vert \mathbf{X}_{t\vert i\vert .}$ is present$\}$	$\mathcal{J}_t^b$	$=$	$\{i \in \mathcal{I} \vert \mathbf{X}_{t\vert i\vert .}$ is present$\}$

Moreover let us also define $\forall i \in [\![1, n]\!], \quad K_i = \{t \in [\![1, T]\!] \vert \mathbf{X}_{t\vert i\vert .}$ is missing$\}$.

The objective of the **Koh-Lanta** algorithm is to impute predicted values in place of missing samples in the train data set and in the test data set. Two stages have been designed to solve that problem. The first stage, denoted as "The Tribe Stage", imputes in the train data set. It uses an EM based algorithm, alternating between estimating the general model $\mathcal{M}^\star$ and using that model for imputation of $\mathbf{X}^\star_{.\vert \mathcal{I}^o\vert .}$. The second stage, denoted as "The Reunification Stage", allows predicting the potential missing values of the test data set. Those two stages are detailed below.

3.4.1 Train-Data Imputation and Model Construction: The Tribe Stage

The "Tribe Stage" can be described thanks to the following algorithm.

0. $\forall t \in [\![1, T]\!], \mathbf{X}^\star_{t\vert \mathcal{I}_t^o\vert .}$ are imputed to the mean variables estimated on $\mathbf{X}^\star_{t\vert \mathcal{J}_t^o\vert .}$,
1. Model construction:
   - $\mathcal{M}^\star \leftarrow$ Mdd-sPLS$(\mathbf{X}^\star_{.\vert \mathcal{I}^o\vert .}, \mathbf{Y}_{\mathcal{I}^o}, \lambda, R)$,
   - $\mathcal{V}^\star \leftarrow \left\{\mathcal{V}_t^\star = \{j \in 1..p_t \vert \mathcal{M}^\star[\mathbf{U}]_t \neq 0\}\right\}_{t \in [\![1,T]\!]}$,
2. Imputation process, $\forall t \in [\![1, T]\!] \vert \mathcal{I}_t \neq \emptyset$:
   - $\mathcal{M}_t \leftarrow$ Mdd-sPLS$(\mathcal{M}^\star[\mathbf{S}_{super}]_{t\vert \mathcal{J}_t^o}, \mathbf{X}_{t\vert \mathcal{J}_t^o\vert \mathcal{V}_t^\star}, \lambda, R)$,
   - $\mathbf{X}^\star_{t\vert \mathcal{I}_t^o\vert \mathcal{V}_t^\star} \leftarrow \mathcal{P}(\mathcal{M}^\star[\mathbf{S}_{super}]_{t\vert \mathcal{I}_t^o}, \mathcal{M}_t)$
3. Back to 1. until convergence of $\mathcal{M}^\star[\mathbf{T}_{super}]$.
4. Return $(\mathcal{M}^\star, \mathbf{X}^\star, \mathcal{V}^\star)$

Note that $\mathcal{M}^\star$ is always learned with $\mathbf{X}^\star_{.|\mathcal{I}^o|.}$ as predictors and $\mathbf{Y}_{\mathcal{I}^o}$ as response variables. In the imputation process, only variables in $\mathcal{V}^\star$ are considered because the objective is to impute, for a given block, only the variables on which the learning has been done. So this imputation stage takes into account only the best features, the best elements for each block, for each tribe. This is why this step is called "The Tribe Stage".

3.4.2 Test-Data Imputation and Prediction: The Reunification Stage

The "Reunification Stage" can be described thanks to the following algorithm.

- $\forall i \in \mathcal{I}^b$:
  1. $\mathcal{M}_i \leftarrow \text{Mdd-sPLS}(\mathcal{M}^\star[\mathbf{T}]_{\bar{K}_i|\mathcal{I}^0}, \mathbf{X}_{K_i|\mathcal{I}^0|\mathcal{V}^\star_{K_i}}, \lambda, R)$
  2. $\mathbf{X}^\star_{K_i|i|\mathcal{V}^\star_{\bar{K}_i}} \leftarrow \mathcal{P}(\mathcal{M}^\star[\mathbf{T}]_{\bar{K}_i|i}, \mathcal{M}_i)$
  3. $\mathbf{X}^\star_{\cdot|i|\cdot} = \cup\{\mathbf{X}_{\bar{K}_i|i|\cdot}, \mathbf{X}^\star_{K_i|i|\cdot}\}$
- $\mathbf{Y}_{\mathcal{I}^b} \leftarrow \mathcal{P}(\mathbf{X}^\star_{\cdot|\mathcal{I}^b|\cdot}, \mathcal{M}^\star)$
- Return $\mathbf{Y}_{\mathcal{I}^b}$.

where $\mathcal{M}^\star$, $\mathbf{X}^\star$ and $\mathcal{V}^\star$ were obtained in the "Tribe Stage". This step is based on the model $\mathcal{M}^\star$ which is then used to impute missing values of the test data set and then to predict its response.

3.5 Computational Complexity

In that section, for sake of simplicity, let p denote the number of covariates in the $\mathbf{X}_t$'s blocks. Only the highest operations in terms of computational time have been taken into account hereafter, operations with at least quadratic terms in q or p. Plus it is supposed that the number R of components computed is largely smaller than q, p or even n.

Let us focus on the case where there are no missing values. The **mdd-SPLS** algorithm needs in its first step the computation of T **ct-sPLS**, only covariance and **SVD** computations are greedy, the corresponding complexity is then $O(nTqp + Tpq\min(p,q))$=$O(Tpq(n + min(p,q))$. In its second step, one **SVD** is performed and the associated complexity is $O(Tpq\min(p,q))$. In the third step, another **SVD** is performed to get regression parameters but the corresponding complexity is $O(n)$ and so negligible against other. Finally, the total computational complexity is

$$O\big(Tqp(n + \min(p,q))\big).$$

The computation of the T **SVD** in the first step clearly dominates the computational complexity. Note that different cases may arise:

- $n >> \max(q,p)$: this is the classical context of data analysis. The total complexity is $O(nTpq)$, which implies linearity in each of the parameters. Let us remark that the behavior is the same if $n >> \min(q,p)$.
- $p >> n$: this is the high dimensional context. If
  - $n >> q$, the total complexity becomes $O(nTqp)$ which is again linear in each of the parameters.

– $n << q$, the total complexity is now $O(Tq^2p)$ which is quadratic in q and linear in T and p.

Figure 2 shows computation times of the **mdd-SPLS** algorithm according to different values of the parameters n, T, p and q. For each set of parameters, 20 simulations were performed. The observed numerical results is clearly consistent with the previous theoretical total complexity. The numerical/theoretical results of the case "$p >> n$ and $q > p$" are not provided here but are similar to the case "$p >> n$ and $q < p$" with linear behavior in q and quadratic behavior in p.

3.6 Adaptation to the Classification Case

If the response variables are categorical, the regression model so far discussed for numerical response variables must be adapted to the problem of classification. **PLS** methods were showed to be efficient through a **Logistic Discrimination** model. The underlying methodology is based on a classic **PLS** on the dummy-standardized variables of the response matrix to build a convenient number of components. Those components are used to estimate a **Logistic Regression** model discriminating original classes, see for example Sjöström et al. (1986) for **PLS-Discriminant Analysis** and Hosmer and Lemeshow (1989) for the **Logistic Regression**. Sabatier et al. (2003) showed that this method is biased. Clemmensen et al. (2011) recommend using a standard **Linear Discriminant Analysis** (**LDA**) in that context. Because slightly better results were found with the **LDA** method, it has been conserved in the **mdd-sPLS** implementation. The **mdd-sPLS** components are thus built on the standardized complete disjunctive coding of the class membership of the individuals for R components, which are the covariates of the **LDA** model and where class memberships are responses. Appendix C provides an illustration of the **mdd-sPLS** approach in this classification framework using the usual data set **Penicillium YES data set**.

4 Simulation Study

Previous sections have presented **mdd-sPLS**, a supervised multi-block method which allows regularization and variable selection. A regression model has also been defined. An algorithm to deal with missing samples, which means that a row of a block might be missing in the train or in the test data set, has been proposed. Simulations have been performed to explore the performances of the **mdd-sPLS** method in term of prediction errors according to:

- increasing proportion of missing values,
- decreasing number of individuals,

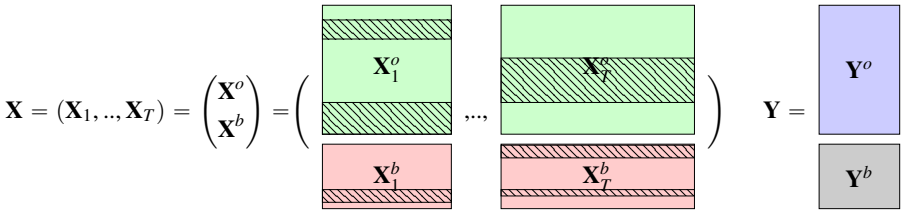


Figure 1: Structure of a T-blocks multi-block data set for train (resp.test) part, denoted by "o" (resp. 'b") symbol. Missing values are symbolized by hatched areas.

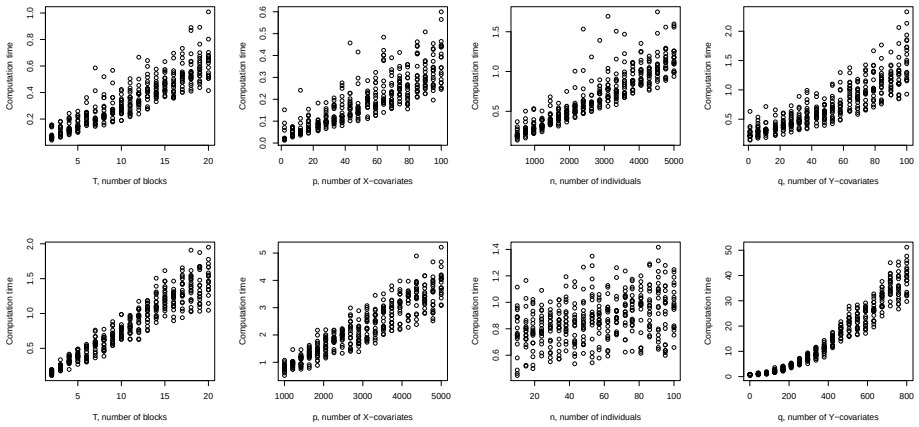


Figure 2: Computation times in both of the discussed regimes against T, p, n and q. First row describes cases when $n >> \max(p,q)$ with $(n,T,q,p) = (1000,10,10,100)$ if not varying. Second row describes cases when $p >> n$ with $(n,T,q,p) = (20,10,10,1000)$ if not varying.

- decreasing correlations between the blocks $\mathbf{X}_t, t = 1, \ldots, T$.

Computation times and convergence successes have also been studied. The data generating process is described in Section 4.1. Competing methods (for imputation and prediction) are presented in Section 4.2. Notice that the **mdd-sPLS** (with **Koh-Lanta** algorithm) method is the only method that handles the missing data and makes the prediction of the response at the same time, denoted as "all-in-one method". The other methods are called "two-steps methods" because they first handle missing values and then build a model to predict response. The methodological process used for the "two-steps methods" is detailed and takes into account the choices and comments of the corresponding authors. All numerical results of the simulation study are presented in Section 4.3.

4.1 Data Generating Process

Let us consider a multi-block context with $T = 10$ blocks of covariates. The $\mathbf{X}_t$'s are generated with **inter-block** relationships (i.e. links between the different blocks) and **intra-block** relationships (i.e. links between the different variables within a block).

- Each block $\mathbf{X}_t$ is composed of $D = 4$ groups of covariates. The number of covariates in each group is equal to 40.
- For $d \in [\![1, D-1]\!]$, the group d of block $\mathbf{X}_t$ is linearly linked to the corresponding group d of the other blocks; the **inter-block** linear correlation parameter is denoted ρ_t.
- For a given group $d \in [\![1, D-1]\!]$ and a given block t, the variables are linearly linked through the **intra-block** linear correlation parameter, denoted ρ_d.
- For a given block t, the D^{th} group is not linked either to the other groups of the block t nor to the groups of the other blocks.

To resume, the $\mathbf{X}_t$'s data sets can be represented as

$$\mathbf{X} = \Big(\overbrace{\underbrace{\clubsuit \cdots \clubsuit}_{\text{Group 1}} \cdots \underbrace{\spadesuit \cdots \spadesuit}_{\text{Group } D}}^{\mathbf{X}_1} \overbrace{\bullet \cdots \bullet}^{\mathbf{X}_{2 \text{ to T-1}}} \overbrace{\underbrace{\clubsuit \cdots \clubsuit}_{\text{Group 1}} \cdots \underbrace{\spadesuit \cdots \spadesuit}_{\text{Group } D}}^{\mathbf{X}_T} \Big),$$

where the card game symbols represent variables with different links. To generate the n observations of all the covariates of the T blocks, the multivariate normal distribution has been used with a null vector as mean and a covariance matrix of size $T \times D \times 40 = 1600$respecting the conditions mentioned above.

The response matrix $\mathbf{Y}$ must be designed with links to the covariates of the blocks $\mathbf{X}_1, \ldots, \mathbf{X}_T$.

- The block $\mathbf{Y}$ must be restricted to $q = 1$ variable since the Lasso method, the main prediction benchmark method in the simulation study, only works for univariate response.

- $\mathbf{Y}$ is linked to 5 of the 10 blocks. In each of those blocks, only a number of variables denoted θ, randomly chosen in $\Theta = \{4, 8, 12, 16, 20, 24, 28, 32, 36, 40\}$, is indeed taken into account. The $40 - \theta$ other variables of that group are filled with Gaussian noises.

  This process allows to simulate strongly correlated data sets if θ is high for each of the 5 blocks. Inversely, if the different θ's are small, the first left singular vector tends to describe less common information.

- The response variable is then obtained as the first left singular vector of the **SVD** applied to the matrix containing only the informative covariates and is then naturally linearly linked to those informative covariates.

For generation of the missing values, a random process is used to delete some rows (observations) of some blocks $\mathbf{X}_t$, corresponding to a proportion of missing values fixed a priori. A constraint has been taken into account: there must be at least one block of non missing values for each individual.

Figure 3 shows the correlation matrix of all the covariates of the $T = 10$ blocks bound together and the right column corresponds to the correlations with the response matrix $\mathbf{Y}$, using a data set simulated with $n = 100$, $\rho_t = 0.9$, $\rho_d = 0.9$ and 30% of missing values. Note that the calculation of the correlations is based only on the non missing values.

In the simulation study of Section 4.3, various values for the parameters n, ρ_t, ρ_d and the proportion of missing values were considered.

4.2 Competing Methods

mdd-sPLS (with **Koh-Lanta** algorithm) was compared to several competing methods. Since existing approaches are mostly "two-steps methods", it is therefore necessary to choose **imputation** method and **prediction** method.

The `imputation` methods selected for this simulation study were:

- **mean**: this is the simplest way to impute. For a given covariate in a given block, the missing values are estimated with the mean of the non missing values of this covariate.

- **softImpute**, see Hastie et al. (2015): the imputation is based on the use of fast-ALS dedicated to estimate missing values in a single-block context. Hence, the block structure of the data set is ignored.
- **imputeMFA**, from the package `missMDA`, see Husson and Josse (2013): the underlying algorithm takes into account the block structure.
- **nipals**, suggested by **mixOmics** authors among other sources: this imputation step is followed by a **classic-sPLS**. The protocol suggested by the authors is detailed on `http://mixomics.org/methods/missing-values/`.

Note that the last three imputation methods look for a **SVD**-modified componentwise structure of the data, as in the proposed **mdd-sPLS** (with **Koh-Lanta** algorithm). However, those **imputation** baseline methods are not supervised and so the number of axes must be tuned by the user.

For the **prediction** step, three methods were selected:

- **mdd-sPLS**: the proposed method applied to the imputed data set (i.e. without missing values).
- $\mathcal{L}$**asso**: the well-known $\mathcal{L}_1$-penalized prediction method which is easily usable if the response variable is univariate.
- **classic-sPLS**: as previously mentioned, this approach is used once the **nipals** algorithm has been used to impute missing values.

From these different methods of imputation and prediction, we will compare the numerical behavior of the following 8 methodologies:

1. **mdd-sPLS** with **Koh-Lanta** algorithm,
2. **nipals** + **classic-sPLS**,
3. **imputeMFA** + **mdd-sPLS**,
4. **imputeMFA** + $\mathcal{L}$**asso**,
5. **softImpute** + **mdd-sPLS**,
6. **softImpute** + $\mathcal{L}$**asso**,
7. **mean** + **mdd-sPLS**,
8. **mean** + $\mathcal{L}$**asso**.

In order to properly evaluate the performance of the different methodologies, a learning (train) sample and a test sample should be considered. The given data set is then splitted into a **train** data set, $(\mathbf{X}^{(\mathbf{train})},\mathbf{Y}^{(\mathbf{train})})$, and a **test** data set, $(\mathbf{X}^{(\mathbf{test})},\mathbf{Y}^{(\mathbf{test})})$. Let us discuss the strategy of imputation of the **train** data set and the **test** data set for the considered methodologies.

- The **Koh-Lanta** algorithm allows to deal with missing values in the **train** and **test** data sets.
- **mean**: the missing values have been estimated as the mean of the $\mathbf{X}^{(\mathbf{train})}$ variables. They are used to impute $\mathbf{X}^{(\mathbf{train})}$ and $\mathbf{X}^{(\mathbf{test})}$ data sets.
- **imputeMFA**: the underlying method is used to impute $\mathbf{X}^{(\mathbf{train})}$, but cannot be applied to $\mathbf{X}^{(\mathbf{test})}$ imputation. Thus the missing values of $\mathbf{X}^{(\mathbf{test})}$ was imputed to the means, estimated from $\mathbf{X}^{(\mathbf{train})}$.

- **softImpute**: even if the authors consider a mono-block problem, it is possible to build a prediction model of imputation (using the `softImpute` function), which is used to estimate $\mathbf{X}^{(\mathbf{test})}$ from the imputed $\mathbf{X}^{(\mathbf{train})}$ (using the `complete` function). Note that, apart from the proposed "all-in-one" method (**mdd-sPLS** with **Koh-Lanta** algorithm), **softImpute** is the only method reusing the eigen-spaces constructed on $\mathbf{X}^{(\mathbf{train})}$ to impute $\mathbf{X}^{(\mathbf{train})}$ and $\mathbf{X}^{(\mathbf{test})}$.
- **nipals**: $\mathbf{X}^{(\mathbf{train})}$ is imputed with the `nipals` function from **mixOmics** package. The number of components has been arbitrarily fixed to `ncomp`=3. As for **missMDA**, there is no particular reason to reuse the eigen-spaces built to impute the $\mathbf{X}^{(\mathbf{train})}$'s missing values to predict the $\mathbf{X}^{(\mathbf{test})}$'s missing values. Thus the $\mathbf{X}^{(\mathbf{test})}$'s missing values are imputed to the mean of the $\mathbf{X}^{(\mathbf{train})}$ data set. Note that the estimation of the **classic-sPLS** model is based on the imputed $\mathbf{X}^{(\mathbf{train})}$ and $\mathbf{Y}^{(\mathbf{train})}$.

4.3 Simulation Results

The simulation study splits into five parts in order to evaluate:

- the effect of the proportion of missing values,
- the effect of the number of individuals,
- the effect of the inter-block correlation structure,
- the effect of the intra-block correlation structure,
- the computation time and the convergence (of the underlying algorithm) efficiency.

The error considered is the leave-one-out cross-validation *root mean squared error in prediction*, denoted RMSEP. For the **mdd-sPLS**, eight different values were tested for λ and the one with the lowest RMSEP error is selected. For the $\mathcal{L}$**asso**, the `glmnet` package is used to select the `lambda.1se` regularization coefficient as proposed by the authors when the low sample size is small. For **nipals**, **softImpute** and **imputeMFA**, the number of components is fixed to 3. Moreover, eight different values were tested for the **softImpute** parameter and the most accurate was selected.

The various scenarios considered are inspired by real case problems. For each of the simulation settings, 20 data sets were generated from the data generating process describes in Section 4.1. Then the eight methodologies presented in section 4.2 were applied to each of the data sets and the associated RMSEP were calculated.

4.3.1 Effect of the Proportion of Missing Values

For the eight methodologies considered, Table 1 provides the RMSEP errors for eight different proportions of missing values from 2%, to 60% when the data gen-

erating process is based on $\rho_d = 0.9$, $\rho_t = 0.9$ (i.e. strong inter/intra-blocks correlations) with $n = 100$. Figure 4 shows the performances of the methods.

	Method / Prop. of NA		2%	5%	8%	10%	15%
	Imputation	Prediction					
RMSEP	mdd-sPLS (**with** Koh-Lanta)		0.243 ± 0.0535	0.259 ± 0.0560	**0.255 ± 0.0588**	**0.260 ± 0.0582**	**0.282 ± 0.0610**
	nipals	classic-sPLS	0.251 ± 0.0514	0.282 ± 0.0545	0.296 ± 0.0523	0.313 ± 0.0510	0.366 ± 0.0540
	imputeMFA	mdd-sPLS	0.253 ± 0.0542	0.283 ± 0.0555	0.291 ± 0.0567	0.307 ± 0.0549	0.347 ± 0.0553
	imputeMFA	Lasso	0.218 ± 0.0460	0.259 ± 0.0495	0.269 ± 0.0425	0.292 ± 0.0442	0.335 ± 0.0484
	softImpute	mdd-sPLS	0.251 ± 0.0531	0.281 ± 0.0547	0.290 ± 0.0550	0.304 ± 0.0531	0.347 ± 0.0554
	softImpute	Lasso	**0.215 ± 0.0445**	**0.255 ± 0.0471**	0.267 ± 0.0403	0.289 ± 0.0431	0.332 ± 0.0465
	Mean	mdd-sPLS	0.253 ± 0.0541	0.284 ± 0.0553	0.292 ± 0.0566	0.308 ± 0.0551	0.348 ± 0.0557
	Mean	Lasso	0.219 ± 0.0455	0.260 ± 0.0495	0.271 ± 0.0430	0.293 ± 0.0437	0.337 ± 0.0477

	Method / Prop. of NA		20%	30%	40%	50%	60%
	Imputation	Prediction					
RMSEP	mdd-sPLS (**with** Koh-Lanta)		**0.300 ± 0.0625**	**0.315 ± 0.0488**	**0.362 ± 0.0598**	**0.413 ± 0.0555**	**0.519 ± 0.0639**
	nipals	classic-sPLS	0.407 ± 0.0475	0.475 ± 0.0439	0.561 ± 0.0474	0.639 ± 0.0412	0.747 ± 0.0418
	imputeMFA	mdd-sPLS	0.380 ± 0.0516	0.426 ± 0.0480	0.488 ± 0.0536	0.544 ± 0.0473	0.634 ± 0.0480
	imputeMFA	Lasso	0.379 ± 0.0525	0.437 ± 0.0578	0.516 ± 0.0615	0.584 ± 0.0618	0.688 ± 0.0688
	softImpute	mdd-sPLS	0.379 ± 0.0519	0.425 ± 0.0475	0.487 ± 0.0539	0.541 ± 0.0469	0.624 ± 0.0471
	softImpute	Lasso	0.378 ± 0.0518	0.437 ± 0.0556	0.514 ± 0.0588	0.582 ± 0.0576	0.676 ± 0.0638
	Mean	mdd-sPLS	0.381 ± 0.0523	0.426 ± 0.0479	0.489 ± 0.0539	0.544 ± 0.0465	0.628 ± 0.0473
	Mean	Lasso	0.382 ± 0.0524	0.441 ± 0.0548	0.517 ± 0.0596	0.584 ± 0.0579	0.679 ± 0.0643

Table 1: Effect of proportion of missing values on the RMSEP. 100 simulation results for ($\rho_d = 0.9, \rho_t = 0.9$) and $n = 100$ individuals. The main statistics are given for each method with (mean ± std). Bolded results correspond to best results for a given proportion of missing values.

For small proportions of missing values, all the methodologies provide very similar results with a very slight advantage to the **Lasso** based regression methods. When the proportion of missing values increases, two methods behaved differently compared to the others (Figure 4). From 20% of missing values onward, The **mdd-sPLS** with Koh-Lanta gave clearly better results than the others methods. When the proportion of missing values is at 50% or more, **nipals + classic-sPLS** methods provides poorer results compared to all the other approaches.

4.3.2 Effect of the Number *n* of Individuals

For the eight methodologies considered, Table 2 provides the RMSEP errors for different numbers n of individuals ($n = 100, 50, 20$) when the data generating process is based on $\rho_d = 0.9$, $\rho_t = 0.9$ (i.e. strong inter/intra-blocks correlations) with a proportion of missing values equal to 30%. Table 2 provides RMSEP error results for these different numbers n of individuals.

The **mdd-sPLS** with **Koh-Lanta** leads to the smallest RMSEP error for the three different sample size n. The other **mdd-sPLS**-based methods have better

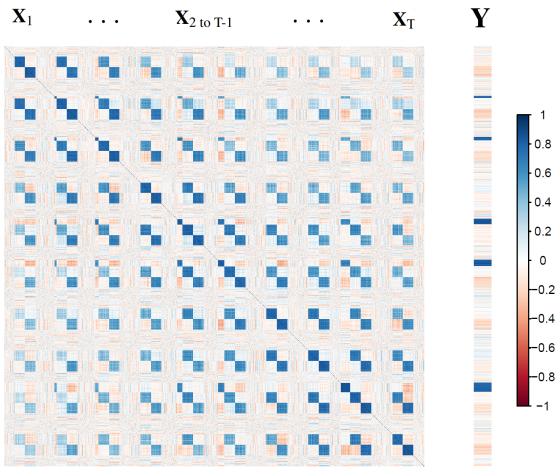


Figure 3: Example of a empirical correlation matrix calculated on a data set simulated with $n = 100$, $T = 10$, $\rho_t = 0.9$, $\rho_d = 0.9$ and 30% of missing values. Correlation scale: from red (-1) to blue (+1)

Figure 4: Effect of proportion of missing values on the RMSEP, barplot of Table 1 results.

	Method \ # individuals		100	50	20
	Imputation	Prediction			
RMSEP	mdd-sPLS (**with** Koh-Lanta)		**0.308 ± 0.0528**	**0.335 ± 0.0643**	**0.445 ± 0.139**
	nipals	classic-sPLS	0.491 ± 0.0504	0.497 ± 0.0665	0.566 ± 0.124
	imputeMFA	mdd-sPLS	0.420 ± 0.0530	0.440 ± 0.0648	0.521 ± 0.119
	imputeMFA	Lasso	0.438 ± 0.0599	0.506 ± 0.0852	0.714 ± 0.169
	softImpute	mdd-sPLS	0.420 ± 0.0525	0.436 ± 0.0644	0.505 ± 0.119
	softImpute	Lasso	0.436 ± 0.0581	0.496 ± 0.0824	0.656 ± 0.162
	Mean	mdd-sPLS	0.421 ± 0.0527	0.440 ± 0.0666	0.528 ± 0.125
	Mean	Lasso	0.440 ± 0.0593	0.504 ± 0.0866	0.721 ± 0.188

Table 2: Effect of sample size on the RMSEP. 100 simulation results for ($\rho_d = 0.9, \rho_t = 0.9$) and 30% of missing values. The main statistics are given for each method with (mean ± std). Bolded results correspond to best results for a given number of individuals per sample.

behavior than the **Lasso**-based methods regardless of the imputation method chosen. Finally, the "two-steps method" **softImpute + mdd-sPLS** has the second best performance.

4.3.3 Effect of the Inter-Block Correlations

As presented before, the response variable in $\mathbf{Y}$ was correlated to some covariates of some X_t's blocks with the intensity ρ_t. Moreover the correlation between different $\mathbf{X_t}$'s blocks was also equal to ρ_t. The performances of the methods were evaluated according to the parameter ρ_t with $\rho_t \in \{0.9, 0.7, 0.5, 0.3\}$. Simulation results are provided in Table 3 and are plotted in Figure 9.

For any ρ_t, the **mdd-sPLS with Koh-Lanta** method is the most accurate one according to the RMSEP. For low values of ρ_t, the **mdd-sPLS**-based methods showed better behaviors than the **Lasso**-based ones.

4.3.4 Effect of the Intra-Block Correlations

The following simulations evaluate the impact of a varying intra-correlation on the overall error **RMSEP**. The other parameters have been fixed to $\rho_t = 0.9$ and 30% of missing values. Simulations results are provided in Table 4 and are plotted in Figure 10.

Among the four simulated settings, the case $\rho_d = 0.9$ corresponds to an already discussed one, see Figure 4. It is interesting to see the stability of the results for those new simulations. The data set with $\rho_d = 0.7$ shows that all the method are

	Method \ ρ_t		0.9	0.7	0.5	0.3
	Imputation	Prediction				
RMSEP	mdd-sPLS **with** Koh-Lanta		**0.312 ± 0.0516**	**0.528 ± 0.0801**	**0.662 ± 0.0948**	**0.752 ± 0.0896**
	nipals	classic-sPLS	0.470 ± 0.0451	0.602 ± 0.0662	0.699 ± 0.0779	0.766 ± 0.0751
	imputeMFA	mdd-sPLS	0.421 ± 0.0487	0.572 ± 0.0710	0.678 ± 0.0848	0.756 ± 0.0822
	imputeMFA	Lasso	0.438 ± 0.0537	0.598 ± 0.0740	0.724 ± 0.103	0.816 ± 0.0975
	softImpute	mdd-sPLS	0.420 ± 0.0492	0.570 ± 0.0706	0.677 ± 0.0853	0.754 ± 0.0824
	softImpute	Lasso	0.433 ± 0.0533	0.591 ± 0.0723	0.718 ± 0.102	0.813 ± 0.100
	Mean	mdd-sPLS	0.421 ± 0.0492	0.572 ± 0.0713	0.679 ± 0.0858	0.756 ± 0.0827
	Mean	Lasso	0.436 ± 0.0538	0.598 ± 0.0743	0.724 ± 0.105	0.818 ± 0.0996

Table 3: Effect of inter-block correlation on the RMSEP. 100 simulation results for $\rho_d = 0.9$, $\rho_t \in \{0.3, 0.5, 0.7, 0.9\}$, $n = 100$ individuals and 30% of missing values. The main statistics are given for each methodology with (mean ± std). Bolded results correspond to the best ones for a given value of ρ_t.

	Method \ ρ_d		0.3	0.5	0.7	0.9
	Imputation	Prediction				
RMSEP	mdd-sPLS **with** Koh-Lanta		**0.399 ± 0.054**	**0.346 ± 0.0563**	**0.317 ± 0.0482**	**0.312 ± 0.0516**
	nipals	classic-sPLS	0.538 ± 0.0524	0.499 ± 0.0492	0.485 ± 0.0453	0.47 ± 0.0451
	imputeMFA	mdd-sPLS	0.477 ± 0.0534	0.443 ± 0.0533	0.433 ± 0.0468	0.421 ± 0.0487
	imputeMFA	Lasso	0.565 ± 0.0691	0.504 ± 0.0608	0.472 ± 0.0554	0.438 ± 0.0537
	softImpute	mdd-sPLS	0.476 ± 0.0524	0.443 ± 0.0541	0.433 ± 0.0467	0.42 ± 0.0492
	softImpute	Lasso	0.565 ± 0.0661	0.503 ± 0.0591	0.472 ± 0.0531	0.433 ± 0.0533
	Mean	mdd-sPLS	0.476 ± 0.0527	0.443 ± 0.0541	0.434 ± 0.0469	0.421 ± 0.0492
	Mean	Lasso	0.572 ± 0.0697	0.508 ± 0.0604	0.476 ± 0.0533	0.436 ± 0.0538

Table 4: Effect of intra-block correlation on the RMSEP. 100 simulation results for $\rho_t = 0.9$, $\rho_d \in \{0.3, 0.5, 0.7, 0.9\}$, $n = 100$ individuals and 30% of missing values. The main statistics are given for each methodology with (mean ± std). Bolded results correspond to the best ones for a given value of ρ_d.

equivalent (≈ 0.31) except **mdd-sPLS (with Koh-Lanta)** for which the error is lower (≈ 0.27). For higher intra-block correlations, among the baseline imputation methods, the **Lasso** based prediction methods are more efficient than **mdd-sPLS** ones but **mdd-sPLS (with Koh-Lanta)** show lowest errors. For lower intra-block correlations, $\rho \in \{0.3, 0.5\}$, among the baseline imputation methods, the **Lasso** based prediction methods are less efficient than **mdd-sPLS** but **mdd-sPLS (with Koh-Lanta)** still lead to better results. It is also interesting to notice that the **nipals** approach has equivalent results than **mdd-sPLS** prediction based methods with the baseline imputation methods in all features.

4.3.5 Computation Time and Convergence Quality

Regarding the convergence of the various methods, the **mean** imputation method is not concerned by this numerical aspect since it is based on only one step of imputation. For the other methodologies, once imputation stages have no further effects on subspace estimation, we considered that the imputation process has converged. The convergence criterion was defined as the stabilization of estimations in the last estimated subspace with a threshold value set to 10^{-9} and the maximum number of iterations to 100. More precisely, let us specify for each method the concerned matrix:

- **mdd-sPLS**: the matrix $\mathbf{T}_{super}\mathbf{V}_{ort}$, which is defined in the algorithm of the method.
- **softImpute**: the matrix U of the left-singular vectors (Hastie et al., 2015, Algorithm 2.1),
- **imputeMFA**: the matrix $\mathbf{U}$ of the left-singular vectors (Josse and Husson, 2016, Chapter 3.1),
- **nipals**: the matrix of the components $\mathbf{t_k}$ (Wold et al., 1983, Algorithm 3c). Since the $\mathbf{t}_k$'s are obtained by deflation, the test of convergence is done on each component separately. If one of the components does not converge, we consider that the algorithm did not converge and if all the components converge, then the number of iterations is the mean of the total number of iterations.

100 simulated data sets have been generated with $T = 10$ blocks of $p = 160$ covariates (with $\rho_d = 0.9$ and $\rho_t = 0.9$), 30% of missing values and $n \in \{100, 50, 20\}$. For each proposed method, results on convergence rate and number of iterations are presented in the first two parts of Table 5. The prediction errors have also been computed and are represented in Figure 11.

nipals and **mdd-sPLS** with **Koh-Lanta** get 100% convergence. **imputeMFA** almost always converged while for **softImpute** almost 30% of imputation processes did not converge when $n = 100$. Concerning the number of iterations, denoted # *iterations* in Table 5, the **mdd-sPLS** with **Koh-Lanta** only needs 3 iterations before converging. **softImpute** shows a high number of iterations. **nipals** needed less iterations to converge. **imputeMFA** used an average of ~ 30 iterations with a large standard deviation relatively to the other methods.

Computations have been performed on `Intel® Xeon® CPU E5-2690 v2, 3.00GHz` processors. Concerning the computation time, one notice that the **Mean** process naturally is the fastest. This intuitive result is followed by the **mdd-sPLS**

	Method \ # individuals		100	50	20
	Imputation	Prediction			
Conv. rate	mdd-sPLS with Koh-Lanta		**100 %**	**100 %**	**100 %**
	nipals		**100 %**	**100 %**	**100 %**
	imputeMFA		99.4 %	98.8 %	99.8 %
	softImpute		71.5 %	85.9 %	92.3 %
# iterations	mdd-sPLS with Koh-Lanta		**3 ± 0**	**3 ± 0**	**3 ± 0**
	nipals		42.6 ± 8.64	39.2 ± 8.29	48.7 ± 7.71
	imputeMFA		27.0 ± 10.1	29.7 ± 11.5	31.2 ± 8.91
	softImpute		71 ± 13.3	66.6 ± 12.9	72.7 ± 12.5
Time (s)	mdd-sPLS with Koh-Lanta		**0.662 ± 0.209**	**0.343 ± 0.0403**	**0.315 ± 0.0550**
	nipals	classic-sPLS	33.0 ± 5.27	18. ± 3.77	22.1 ± 3.48
	imputeMFA	mdd-sPLS	9.44 ± 3.39	3.93 ± 1.45	3.12 ± 0.600
	softImpute	mdd-sPLS	2.00 ± 0.984	0.849 ± 0.155	1.14 ± 0.175
	Mean	mdd-sPLS	**0.0124 ± 0.00215**	**0.00683 ± 0.00069**	**0.00410 ± 0.000469**

Table 5: Effect of number of individuals. 100 simulation results for $T = 10$ blocks of $p = 160$ covariates (with $\rho_d = 0.9$ and $\rho_t = 0.9$) and for 30% of missing values. The main statistics (over the 100 simulations) are given for each method with (mean $\pm$ std). That table is divided in three parts. The first part (lines 1 to 4) corresponds to the convergence rate for each imputation method. The second part (lines 5 to 8) corresponds to the number of iterations for each imputation method, only in case of convergence. The Mean imputation method (that works in 1 iteration and thus always converges) is not taken into account in those two parts. The third part (lines 9 to 13) corresponds to the computation time for each method. *mean*'s are calculated not only over the 100 simulated data sets but also over the number of individuals (indicated in the column and having an impact on leave-one-out computations) in order to "standardize" the results to the time to that of creating a single model one model. Bolded results correspond to best results for a given number n of individuals.

with **Koh-Lanta** approach for which the computation time lasts ~ 0.5 seconds. On the contrary the **nipals** method lasts within tens of seconds, **imputeMFA** is faster but still lasts within seconds (from 3.1 to 9.4 seconds). The **softImpute** method is faster, less than 2 seconds. Not surprisingly, the computation time of almost all methods decreased as the number n of individuals decreased, with the exception of the **nipals** and **softImpute** algorithms.

4.3.6 Conclusion from the Simulations

In comparison with the other competing methods, **mddsPLS** with **Koh-Lanta** clearly exhibits very good performances in terms of predictive capacities in the

context of a large proportion of missing values and small number n of individuals. This is shown in the context of strongly correlated blocks as well as in the context of low inter-block information correlation (small ρ_t). Another set of simulations show the robustness of the results for low ρ_d and low ρ_t and is presented in Appendix G.

5 Real Data Application: the Ebola rVSV-ZEBOV Data Set

The current work was inspired by this real data application.

5.1 The Data Set

The application is an early phase vaccine trial evaluating the rVSV-ZEBOV Ebola vaccine already studied by Rechtien et al. (2017). As many modern early vaccine trials it includes small number of participants (here, $n = 18$) with heterogeneous and high dimensional data sets carrying a lot of information through numerous covariates allowing a deep evaluation of the response to the vaccine.

More specifically, for each participant, the gene expression in whole blood by RNA-seq and the cellular functionality by cytometry have been measured at four different days $\in \{0, 1, 3, 7\}$ after vaccination. Genes of interest were pre-selected by removing those with a variance less than 0.2 leading to 18 301 genes included in the following analysis. The cellular functionality consisted in the characterization of *Natural killers*, *Dendritic cells* and *Cytokines*, covering a total of 129 variables. So, $T = 8$ blocks $\mathbf{X}_t$ of covariates were available, see Table 6 for the number of covariates in each $\mathbf{X}_t$'s blocks.

Moreover, the antibody responses against the Gueckedou strain by ELISA have been measured at days $\in \{28, 56, 84, 180\}$ after vaccination, so $\mathbf{Y} \in \mathbb{R}^{18\times 4}$. The aim of the analysis was to find the best predictors of the antibody responses among the gene expression (transcriptome) and the cellular functionality.

Recall that the **mdd-sPLS** method works with standardized variables. This standardization step implies that the information contained in the variance is not taken into account for each of the variables.

Because of the sample quality constraints, gene expression was not available in about 30% of cases leading to missing values. For example, Table 7 shows the absence (in blue) of all the RNA-Seq values for a particular individual (in columns) for a particular day (in rows) depicting around 30% of missing values. The data set used in that table is available on the NCBI repository.

Type	RNA-SEQ				Cellular functionality			
Block	$\mathbf{X}_1$	$\mathbf{X}_2$	$\mathbf{X}_3$	$\mathbf{X}_4$	$\mathbf{X}_5$	$\mathbf{X}_6$	$\mathbf{X}_7$	$\mathbf{X}_8$
Day	0	1	3	7	0	1	3	7
$\#(variables)$	10279	10134	9082	9670	129	129	129	129

Table 6: Number of covariates per block $\mathbf{X}_t$.

Individuals	7	5	9	1	15	10	14	4	2	12	17	16	8	18	13	11	3	6
Day 0: $\text{Vec}(\mathbf{X}_1^T)^T$																		
Day 1: $\text{Vec}(\mathbf{X}_2^T)^T$																		
Day 3: $\text{Vec}(\mathbf{X}_3^T)^T$																		
Day 7: $\text{Vec}(\mathbf{X}_4^T)^T$																		

Table 7: Missing values (in blue) in the Ebola rVSV-ZEBOV RNA-Seq data sets $\mathbf{X}_t$, $t = 1, \ldots, 4$, where the notation Vec stands for the Vec operator. For a given individual and a given day, all the corresponding values are missing.

5.2 Statistical Analysis

Four **mdd-sPLS**-based methodologies were compared through **MSEP** (means square error of prediction) calculated by leave-one-out cross-validation:

- **mdd-sPLS** with **Koh-Lanta**,
- two-step approach: imputation to the **mean** + **mdd-sPLS**,
- two-step approach: imputation with **softImpute** + **mdd-sPLS**,
- two-step approach: imputation with **imputeMFA** + **mdd-sPLS**.

Figure 5 focuses on **mdd-sPLS** with **Koh-Lanta** and **mean** + **mdd-sPLS** and shows the number of times each response variable has been selected for every optimal λ value. All comparisons are provided in Table 9. Since the **softImpute** method uses random initialization and does not converge systematically, a variability appears in the prediction errors, here depicted by the $(mean \pm std)$ notation. All the methods led to the selection of the day 56 response variable in the model, the only variable that was always selected by the four methods. **mdd-sPLS** with **Koh-Lanta** clearly retains two response variables in the model: day 56 and day 84.

Table 9 shows the final model, selected as minimum for day 56, with $\lambda \simeq 0.866$ which implies that only correlations with one of the responses over that value will be considered in that model. The **mdd-sPLS** with **Koh-Lanta** approach was highly selective as it kept three covariates while the other methods kept 15 variables (see

Method		Leave-One-Out prediction error								
		Day 28		Day 56		Day 84		Day 180		Mean
Imputation	Prediction	RMSEP	#	RMSEP	#	RMSEP	#	RMSEP	#	RMSEP
mdd-sPLS with Koh-lanta $\lambda = 0.8654$		**1.027**	4/18	**0.6143**	18/18	**0.9426**	17/18	**1.029**	1/18	**0.9035**
Mean	mdd-sPLS $\lambda = 0.863$	1.028	2/18	0.6312	18/18	1.041	6/18	**1.029**	0/18	0.9326
softImpute	mdd-sPLS $\lambda = 0.8566667$	1.029±0	(0 ± 0)/18	0.6326 ± 0.03795	(18 ± 0)/18	1.027 ± 0.002191	(4.4 ± 0.8)/18	**1.029±0.0001374**	(0.3 ± 0.5)/18	0.9294
imputeMFA	mdd-sPLS $\lambda = 0.852222$	1.028	3/18	0.6899	18/18	1.026	7/18	**1.029**	0/18	0.9433

Table 8: Results of the leave-one-out cross-validation prediction errors applied to the rVSV data set. The last column gives the mean error. The # symbol represents the number of times each variable is selected in the cross-validation process, among 18 different models built in the cross-validation process (since $n = 18$).

Rechtien et al., 2017, Figure S5 from supplementary materials). As mentionned before, the selection over the **Y** part was also efficient and kept 2 response variables in the model: the antibody levels at day 56 and day 84.

Finally, the selected covariates were biologically meaningful. The three genes ($\text{TIFA}_{day\ 1}$, $\text{SLC6A9}_{day\ 3}$, $\text{FAM129B}_{day\ 3}$) selected through the proposed approach were also selected by Rechtien et al. (2017) as the three top genes in the sensibility analysis realized with bootstrap analysis. Note that the other three methodologies selected the same three covariates except for the **softImpute+mdd-sPLS** methods which did not select $\text{SLC6A9}_{day\ 3}$, the corresponding results are not provided here. For each selected covariate in Table 9, the absolute value of the product between the corresponding weight and super-weight gives a measure of its impact in the model. For $\text{TIFA}_{day\ 1}$, this value was equal to 0.961, for $\text{SLC6A9}_{day\ 3}$ equal to 0.107 and for $\text{FAM129B}_{day\ 3}$ equal to 0.255. The interpretation was that $\text{TIFA}_{day\ 1}$ was the most important covariate while $\text{FAM129B}_{day\ 3}$ was the second most important one and $\text{SLC6A9}_{day\ 3}$ was the third most important one. Correlations, considering only present samples, between $\text{TIFA}_{day\ 1}$ and Gueckedou strain on days $\in \{28, 56, 84, 180\}$ are respectively equal to 0.83, 0.96, 0.90 and 0.81.

It is also interesting to interpret the parsimony of the antibody response measurements by selecting day 56 and day 84 and not response measurements at day 28 and day 180. This reflects probably the fact that once established, the antibody response is quite stable over every individual and therefore does not need many repeated measurements to be characterized.

6 Conclusion

The **mdd-sPLS** method is a **SVD**-based method (without iteration process) dedicated to multi-block supervised analysis. The **Koh-Lanta** algorithm deals with missing values in the *train* sample but also in the test sample and is implemented

	Block	Variable	Weights	Super-weights
X	Genes on day 1	TIFA	1	-0.961
	Genes on day 3	SLC6A9	0.388	0.277
		FAM129B	0.922	
Y	Response	Gueckedou on day 56	-0.924	×
		Gueckedou on day 84	-0.382	

Table 9: Ebola rVSV phase I/II constructed mdd-sPLS model for $\lambda = 0.8654$. 4 blocks of gene expression and 4 blocks of cellular functionality, one for each of the days $\{0, 1, 3, 7\}$, have been introduced but only days 1 and 3 blocks of gene expression have been selected. Also, in the response block, only days 56 and 84 have been selected. In columns are represented the weights which denote $u_t^{(1)}$ for the $\mathbf{X}$ blocks and $v^{(1)}$ for the $\mathbf{Y}$ block and also the super-weights for the $\mathbf{X}$ blocks and denoted by $\beta_t^{(1)}$. Only one dimension was found interesting here.

in the **mdd-sPLS** method. The proposed method shows very good performance on simulated data sets and gave relevant results in the real data application. This approach allows to make variable selection and missing values imputation. The missing data context is limited to entire rows of missing values for certain blocks and can be generalized to any position of missing values by adjusting missing values thanks to known values through a linear model for example. Most of the results, described in this paper, relate to regression problem but the method can also be applied to classification problem.

The **mdd-sPLS** including **Koh-Lanta** algorithm method has been implemented in:

- a **R**-package accessible on the **CRAN**, `https://cran.r-project.org/package=ddsPLS`, with a **R**-vignette at `https://hadrienlorenzo.netlify.com/html/ddspls`,
- a **python**-package accessible on **PyPi**, `https://pypi.org/project/py_ddspls/`, with a **Python**-vignette at `https://pypi.org/project/py_ddspls/`.

Acknowledgments

The authors would like to thank Franois Husson, Arthur Tenenhaus and Julie Josse for helpful discussions. Hadrien Lorenzo is supported by a 2016 Inria-Inserm thesis grant *Médecine Numérique* (for *Digital Medicine*).

A Monotonicity of the Weight Cardinality: a Counter Example

The remaining question is about the potential decreasing of the number of variables selected per component. In other words, is the number of null coefficients of a given component a decreasing function of λ? The answer is no as we will see through the following counterexample.

A sample of $n = 100$ individuals is generated with the following correlation structure between a 9-dimensional covariate and a two-dimensional response:

$\frac{\mathbf{Y}^T\mathbf{X}}{n-1} =$

	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9
Y_1	1.00	-0.06	-0.10	0.07	0.09	0.15	0.16	0.14	0.22
Y_2	-0.08	0.98	0.29	-0.18	0.25	0.02	0.04	-0.01	-0.03

.

The 1^{st} and the 2^{nd} $\mathbf{X}$-variables are clearly well correlated respectively with the 1^{st} and the 2^{nd} $\mathbf{Y}$-variables. Let us denote by $\mathbf{u}$, respectively $\mathbf{v}$, the first right, respectively left, eigenvector of the soft-thresholded covariance matrix $S_\lambda(\frac{\mathbf{Y}^T\mathbf{X}}{n-1})$ for any positive λ. Figure 6 shows the real cardinalities (**black lines**) and the upper bound cardinalities (red lines) of $\mathbf{u}$ and $\mathbf{v}$ weights which correspond the application of Corollary 2.1.1. Vertical lines (discontinuous blue lines) symbolize the vanishing of a coefficient of the current matrix $S_\lambda(\frac{\mathbf{Y}^T\mathbf{X}}{n-1})$, depending on λ. When $\lambda \in [0.1, 0.14]$, $\mathbf{Card}(\mathbf{u})$ increases, this corresponds to an area in which $\mathbf{Card}(\mathbf{u})$ might take the value 9, in that part all the columns are different from 0, except for $\lambda = 0.14$, where the variable X_8 "disappears". Let us zoom on those particular points:

$$u(\lambda = 0.12) \approx (0, 0.97, 0.19, -0.07, 0.15, 0, 0, 0, 0)^T \rightarrow \mathbf{Card}(\mathbf{u}(\lambda = 0.12)) = 4$$
$$u(\lambda = 0.13) \approx (0.99, 0, 0, 0, 0, 0.023, 0.034, 0.011, 0.10)^T \rightarrow \mathbf{Card}(\mathbf{u}(\lambda = 0.13)) = 5,$$

This is due to the fact that the order of the components associated with the first two-dimensional eigenspace is defined through the $\mathcal{L}_2$-norm of the components. However, the $\mathcal{L}_1$-shrinkage of the coefficients based on λ can change this order since the power of both the first two components are very close to each other, , and only in that case. A way of avoiding this kind of reversal would be to change the soft-thresholding operation with a more $\mathcal{L}_2$-shrinkage flavored operation such as the **SCAD operation**, see for example Fan and Li (2001). But in real cases, the first components are not often sufficiently close in the $\mathcal{L}_2$-norm sense to observe this kind of reversal. Thus, it was decided to keep the soft-thresholding operator in the **mdds-PLS** method.

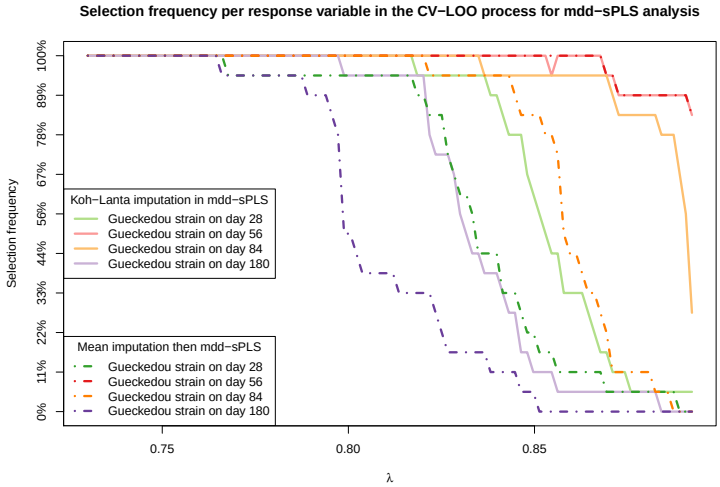


Figure 5: Proportion of selection of each $\mathbf{Y}$ variable for both proposed methods on the rVSV data set through Leave-One-Out Cross-Validation. Dotted lines show mean-imputation results and bolded lines full mdd-sPLS with the Koh-Lanta algorithm.

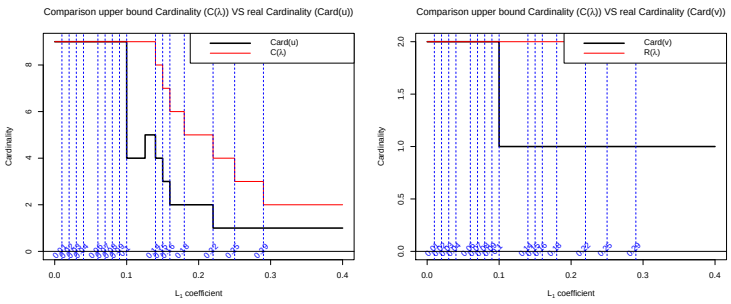


Figure 6: Behaviors of the cardinalities and their upper bounds, defined with corollary 2.1.1, for a simulation case, first component.

B Regression Example: the Liver Toxicity Data Set

In the liver toxicity data set (see Heinloth et al., 2004) $n = 64$ male rats of the inbred strain Fisher 334 were exposed to non toxic (50 or 150 mg/kg), moderately toxic (1500 mg/kg) or severely toxic (2000 mg/kg) doses of acetaminophen (paracetamol) in a controlled experiment. The values of $\mathbf{X} \in \mathbb{R}^{64\times 3116}$ are RNA measures and the values of $\mathbf{Y} \in \mathbb{R}^{64\times 10}$ are clinical measures of markers for liver injury. There are no missing values. A comparison of **classic-sPLS** and the proposed **mdd-sPLS** is given in Figure 7. The first two graphics provide the results of leave-one-out cross-validation for the associated tuning parameters: the number of variables for **classic-sPLS** and the parameter λ for **mdd-sPLS**. The third graphic gives the number of occurrences of each of the $\mathbf{Y}$'s response variables versus λ. These 10 response variables are plotted independently, so one curve represents the behavior of one response variable depending on the model and the regularization parameter chosen. Let us comment the results of these two approaches.

- **classic-sPLS**. According to the suggestion of Lê Cao et al. (2008), the $keep_Y$ parameter has arbitrarily been fixed to 2. With this choice, 8 response variables are removed systematically from the built models in the cross-validation process. One hope that the same two response variables are almost always selected, but this is clearly not the case. Indeed, if it were, 8 response variables should have a RMSEP value around 1 (the mean-prediction error, since the variables were standardized), which means that the model does not take into account those variables. However, this observation is only valid for only 5 response. This is problematic because the obtained sparse model strongly depends on this arbitrary choice of the $keep_Y$ parameter and thus may provide a wrong model for both selection and prediction.

- **mdd-sPLS**. The graphic of the RMSEP errors versus λ clearly shows that 5 response variables are already estimated to the mean when $\lambda = 0.4$ (with corresponding RMSEP errors closed to 1) while the five others are still estimated by the model at this stage. Then, as the regularization parameter λ increases, the variability of the RMSEP errors increases since the model has less and less information in the underlying soft-thresholded matrix. By carefully studying this graph, only 2 response variables (the 2 bottom ones) show real learning interest, observable through their decreasing curves while the other curves are increasing. The decreasing part reaches a minimum for $\lambda \approx 0.85$. One can see that this value coincides with the moment when the 3^{rd} response variable, in terms of increasing RMSEP ranking, reaches to 1, the symbolic limit of the error. This is equivalent to say that the model doesn't select that variable.

The graphic of the occurrences (per response variable) in the estimated models built in the cross-validation process also reinforces the user's choice to only select two response variables.

Variable		A_43_P14131	A_42_P620915	A_43_P11724	A_42_P802628	A_43_P10606	A_42_P675890	A_43_P23376	A_42_P758454	A_42_P578246	A_43_P17415	A_42_P610788	A_42_P840776	A_42_P705413	A_43_P22616	Mean RMSEP(LOO)	Min RMSEP(LOO)
classic-sPLS	$k_X = 12$	-0.6	-0.52	0.17	-0.12	-0.14	-0.18	-0.21	-0.18	-0.14	-0.33	-0.07	-0.26			0.78	0.34
mdd-sPLS	$\lambda = 0.845$	-0.6	-0.52	0.17	-0.12	-0.14	-0.18	-0.21	-0.18	-0.14	-0.33	-0.07	-0.26	-0.03	-0.01	0.88	0.36
	$\lambda = 0.9$	-0.86	-0.51													0.89	0.41

Table 10: Results for classic-sPLS and mdd-sPLS methods: selected genes with their corresponding estimated weights. The two last columns provide the mean and the minimum of the RMSEP errors calculated during the cross-validation process.

Table 10 provides results of **X**'s weights obtained for **classic-sPLS** and **mdd-sPLS** with two choices of λ. Those models have been retained according to Figure 7. **classic-sPLS** based on **mixOmics** R package selects 12 genes (with an optimal parameter $keep_X$ obtained by cross-validation) and 2 response variables (with the parameter $keep_y$ arbitrarily set to 2 by the user). This approach provides the lower cross-validation leave-one-out errors. For the **mdd-sPLS** method, one can clearly see that the best model, in terms of minimum RMSEP error is not the sparsest one (with 14 genes selected including the 12 genes selected by **classic-sPLS**). But, looking at the degree of sparsity in **Y** also permits to select $\lambda = 0.9$ as a good candidate. For that value of λ (very close to the optimal one), the number of genes selected goes from 14 to 2 which is a very good model in terms of sparsity. For $\lambda = 0.9$, **mdd-sPLS** (with the first component only) provides an excellent selection simultaneously in **X** and **Y**, with two variables selected in each matrix and a parameter λ close to its optimal value in terms of cross-validation leave-one-out errors.

C Classification Example: the Penicillium YES Data Set

The **Penicillium YES** data set (available in the `sparseLDA` package) is a classification data set describing three Penicillium species: melanoconodium, polonicum, and venetum. In this data set of size $n = 36$ (with the three balanced groups),

$p = 3542$ covariates were extracted from multi-spectral images with 18 spectral bands: $\mathbf{X} \in \mathbb{R}^{36\times3542}$ and $\mathbf{Y} \in \mathbb{R}^{36\times3}$ where the three columns of $\mathbf{Y}$ are the indicator variables of the groups). More details are available by Clemmensen et al. (2011) where the interest of the **Sparse Discriminant Analysis** method is highlighted. The **Sparse Discriminant Analysis** method needed only 2 covariates to perfectly predict the assignment to one of the three groups. A leave-one-out cross-validation has been performed for each fold $i = 1, \ldots, 12$, the i^{th} triplet of melanoconodium, polonicum, and venetum, to optimize the parameter λ. The **mdd-sPLS** method (with $\lambda = 0.956$ for example) permits to select 4 different covariates, 2 on each of the two components, with a perfect assignment rate. These two components are plotted in Figure 8 and the separation of the three groups is clearly visible.

D Effect of Varying Inter-Block Correlation

The case of varying inter-block correlation has been analysed in that part. The Figure 9 summarizes the corresponding results. Other parameters have been respectively fixed to $\rho_d = 0.9$, 30% of missing values, 100 individuals per simulated data set and 100 simulations per ρ_t.

In the context of strongly correlated blocks, **mdd-sPLS (with Koh-Lanta)** seems to have a better behavior than the other methods where **nipals + classic-sPLS** is slightly less efficient than other methods where **mdd-sPLS** prediction based methods are lighlty better than **Lasso** prediction based methods. As the correlation between blocks shrinks, the **mdd-sPLS** prediction based methods and **mdd-sPLS (with Koh-Lanta)** show equivalent results, plus, **nipals + classic-sPLS** is almost as good as those methods. Finally in that low correlation context, the **Lasso** prediction based methods, are less efficient than other methods.

E Effect of Varying Intra-Block Correlation

The case of varying intra-block correlation has been analysed in that part. The Figure 10 summarizes the corresponding results. Other parameters have been respectively fixed to $\rho_t = 0.9$, 30% of missing values, 100 individuals per simulated data set and 100 simulations per ρ_d.

Whatever is the intensity of the intra-block correlation, **mdd-sPLS (with Koh-Lanta)** shows better results in terms of prediction error. Furthermore the **mdd-sPLS** prediction based methods are the second ranked methods. The third position is given to the **Lasso** prediction based methods for strong intra-block correlations and to **nipals + classic-sPLS** for low intra-block correlations.

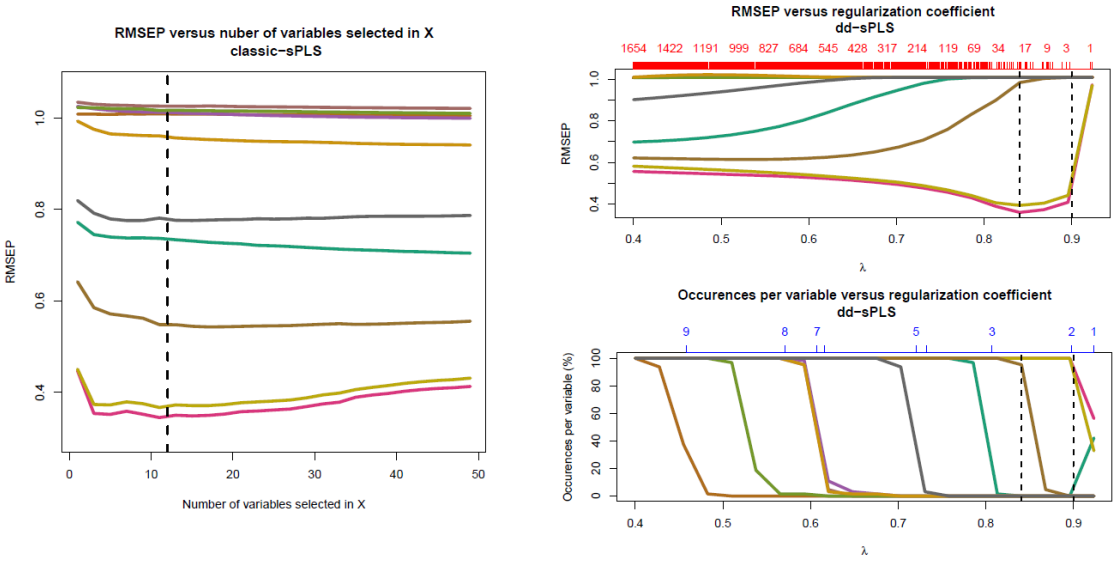


Figure 7: Results of Cross-Validation Leave One Out for both the **classic-sPLS** and **mdd-sPLS** methods for the Liver Toxicity data set.

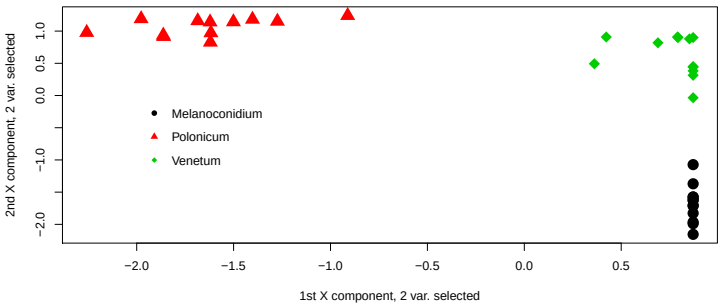


Figure 8: Results of the application of the **mdd-sPLS** to the Penicillium YES data set.

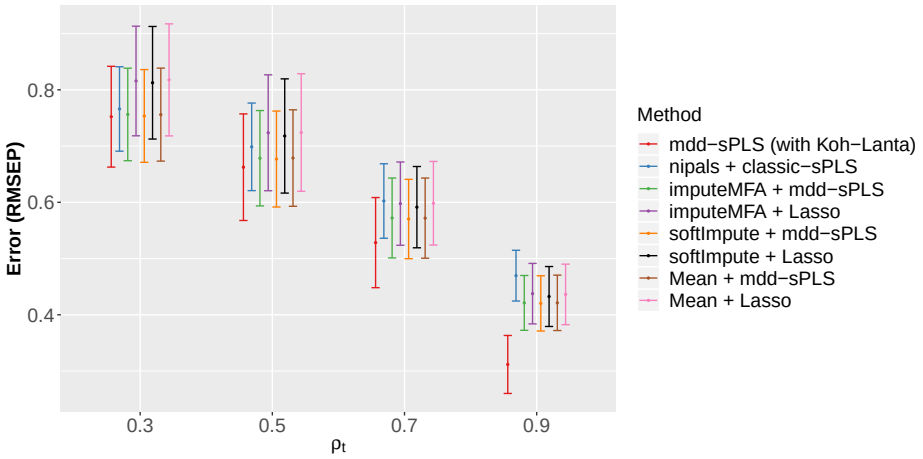


Figure 9: Effect of varying ρ_t on the RMSEP, barplot of Table 3 results. For fixed values of $\rho_d = 0.9$, 30% of missing values, 100 individuals per simulated data set and a total of 100 simulated data sets

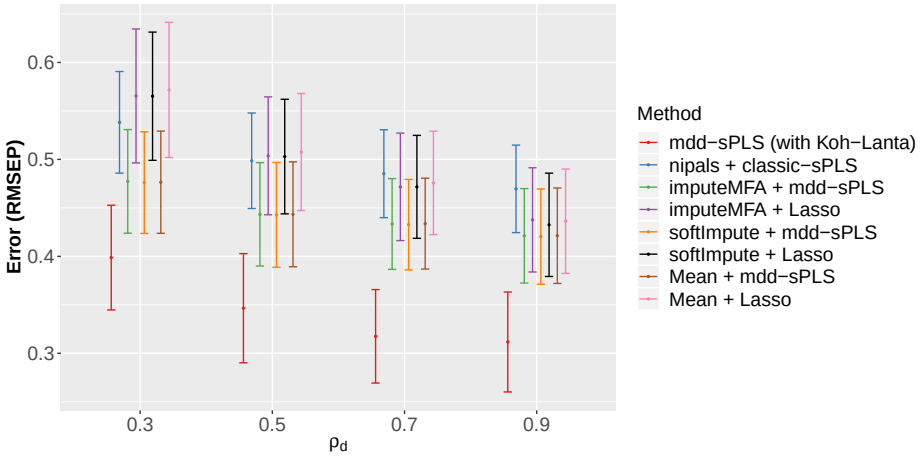


Figure 10: Effect of of varying ρ_d on the RMSEP, barplot of Table 4 results. For fixed values of $\rho_t = 0.9$, 30% of missing values, 100 individuals per simulated data set and a total of 100 simulated data sets

F Effect of Varying Number of Individuals

The case of varying number of individuals has been analysed in that part. Figure 11 summarizes the corresponding results. Other parameters have been respectively fixed to $\rho_t = 0.9$, $\rho_d = 0.9$, 30% of missing values, 100 individuals per simulated data set and 100 simulations per n.

Three cases have been considered. For a low number of individuals, **mdd-sPLS (with Koh-Lanta)** has the smallest error, the **mdd-sPLS** prediction based methods and the **nipals + classic-sPLS** method are slightly less precise and the **Lasso** prediction based methods are fewer precise. As the number of individuals increase, the **nipals + classic-sPLS** method seems to be less precise than other methods while **mdd-sPLS (with Koh-Lanta)** finally gets very better results.

G Effect of Low Intra-Block and Inter-Block Simulations

The following simulations permit to appreciate the validity of the proposed method if there is little information to catch between blocks (low inter-block correlation) and inside blocks (low intra-block correlation).

Here, the proportion of missing values has been fixed to 30% and the number of individuals to $n = 100$. The intra-block correlation has been fixed to $\rho_d = 0.3$ and the inter-block correlation to $\rho_t = 0.5$. Figure 12 shows the results for 100 simulated data sets.

For that level of information, the all methods show equivalent performances, with an error close to ≈ 0.73, except for the **Lasso** based prediction methods, for which the error is higher ≈ 0.83.

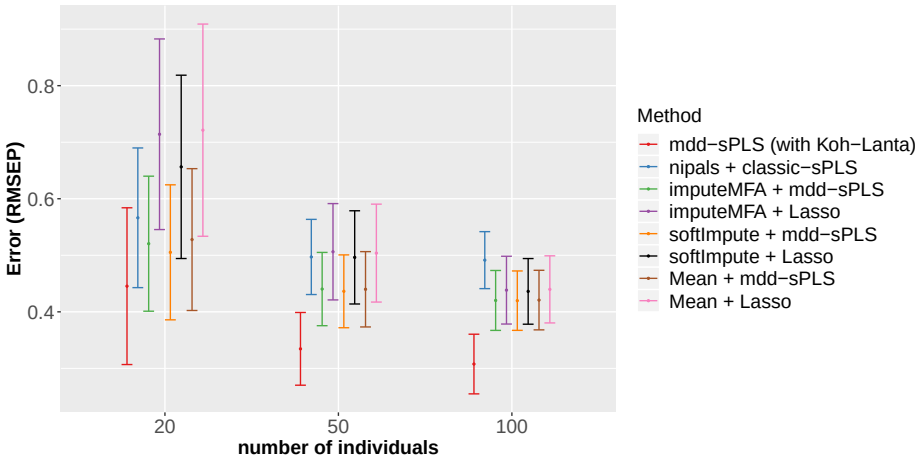


Figure 11: Effect of of varying number of participants on the RMSEP, barplot of Table 5 results. For fixed values of $\rho_t = 0.9$, $\rho_d = 0.9$, 30% of missing values and a total of 100 simulated data sets.

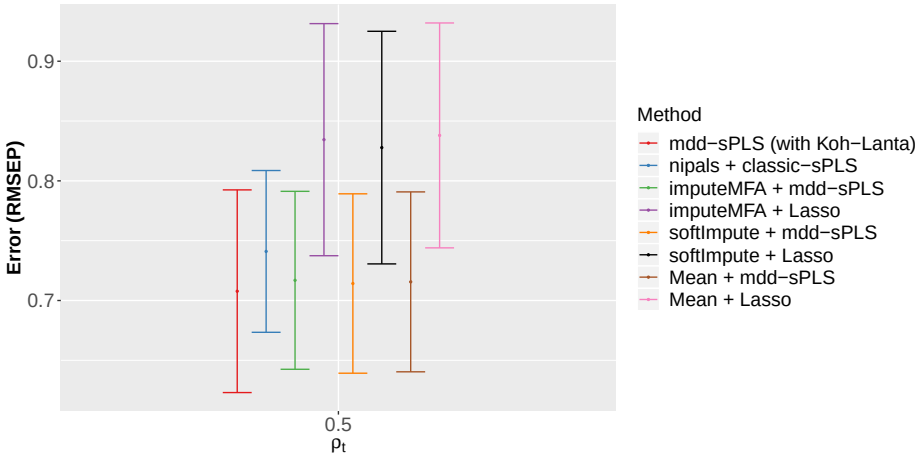


Figure 12: Effect of low intra-block correlation, $\rho_d = 0.3$, and low inter-block correlation, $\rho_t = 0.5$, on the overall errors for fixed values of of missing values, 30%, for 100 individuals per simulated data set and a total of 100 simulated data sets.

References

Arash A Amini and Martin J Wainwright. High-dimensional analysis of semidefinite relaxations for sparse principal components. In Information Theory, 2008. ISIT 2008. IEEE International Symposium on, pages 2454–2458. IEEE, 2008.

Dimitris Bertsimas, Colin Pawlowski, and Ying Daisy Zhuo. From predictive methods to missing data imputation: An optimization approach. Journal of Machine Learning Research, 18(196):1–39, 2018. URL http://jmlr.org/papers/v18/17-073.html.

Stéphanie Bougeard, El Mostafa Qannari, Coralie Lupo, and Mohamed Hanafi. From multiblock partial least squares to multiblock redundancy analysis. a continuum approach. Informatica, 22(1):11–26, 2011.

S van Buuren and Karin Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in r. Journal of statistical software, pages 1–68, 2010.

Tony Cai and Weidong Liu. Adaptive thresholding for sparse covariance matrix estimation. Journal of the American Statistical Association, 106(494):672–684, 2011.

Hyonho Chun and Sündüz Keleş. Sparse partial least squares regression for simultaneous dimension reduction and variable selection. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 72(1):3–25, 2010.

Line Clemmensen, Trevor Hastie, Daniela Witten, and Bjarne Ersbøll. Sparse discriminant analysis. Technometrics, 53(4):406–413, 2011.

Alexandre d'Aspremont, Laurent E Ghaoui, Michael I Jordan, and Gert R Lanckriet. A direct formulation for sparse pca using semidefinite programming. In Advances in neural information processing systems, pages 41–48, 2005.

Yash Deshpande and Andrea Montanari. Sparse pca via covariance thresholding. Journal of Machine Learning Research, 17(141):1–41, 2016. URL http://jmlr.org/papers/v17/15-160.html.

Jianqing Fan and Runze Li. Variable selection via nonconcave penalized likelihood and its oracle properties. Journal of the American statistical Association, 96 (456):1348–1360, 2001.

Trevor Hastie, Rahul Mazumder, Jason D Lee, and Reza Zadeh. Matrix completion and low-rank svd via fast alternating least squares. Journal of

Machine Learning Research, 16(1):3367–3402, 2015. URL http://jmlr.org/papers/volume16/hastie15a/hastie15a.pdf.

Alexandra N Heinloth, Richard D Irwin, Gary A Boorman, Paul Nettesheim, Rickie D Fannin, Stella O Sieber, Michael L Snell, Charles J Tucker, Leping Li, Gregory S Travlos, et al. Gene expression profiling of rat livers reveals indicators of potential adverse effects. Toxicological Sciences, 80(1):193–202, 2004.

Agnar Höskuldsson. Pls regression methods. Journal of chemometrics, 2(3):211–228, 1988.

DW Hosmer and Stanley Lemeshow. Applied logistic regression. 1989. New York: Johns Wiley & Sons, 1989.

François Husson and Julie Josse. Handling missing values in multiple factor analysis. Food quality and preference, 30(2):77–85, 2013.

Iain M Johnstone and Arthur Yu Lu. Sparse principal components analysis. Unpublished manuscript, 7, 2004.

Iain M Johnstone and Arthur Yu Lu. On consistency and sparsity for principal components analysis in high dimensions. Journal of the American Statistical Association, 104(486):682–693, 2009.

Ian T Jolliffe, Nickolay T Trendafilov, and Mudassir Uddin. A modified principal component technique based on the lasso. Journal of computational and Graphical Statistics, 12(3):531–547, 2003.

Julie Josse and François Husson. missmda: a package for handling missing values in multivariate data analysis. Journal of Statistical Software, 70(1):1–31, 2016.

Robert Krauthgamer, Boaz Nadler, Dan Vilenchik, et al. Do semidefinite relaxations solve sparse pca up to the information limit? The Annals of Statistics, 43 (3):1300–1322, 2015.

Kim-Anh Lê Cao, Debra Rossouw, Christele Robert-Granié, and Philippe Besse. A sparse pls for variable selection when integrating omics data. Statistical applications in genetics and molecular biology, 7(1), 2008.

Kim-Anh Lê Cao, Ignacio González, and Sébastien Déjean. integromics: an r package to unravel relationships between two omics data sets. Bioinformatics, 25(21):2855–2856, 2009.

Rolf Manne. Analysis of two partial-least-squares algorithms for multivariate calibration. Chemometrics and Intelligent Laboratory Systems, 2(1-3):187–197, 1987.

Philip RC Nelson, Paul A Taylor, and John F MacGregor. Missing data methods in pca and pls: Score calculations with incomplete observations. Chemometrics and intelligent laboratory systems, 35(1):45–65, 1996.

Roger Penrose. On best approximate solutions of linear matrix equations. In Mathematical Proceedings of the Cambridge Philosophical Society, volume 52, pages 17–19. Cambridge University Press, 1956.

S Joe Qin, Sergio Valle, and Michael J Piovoso. On unifying multiblock analysis with application to decentralized process monitoring. Journal of chemometrics, 15(9):715–742, 2001.

Anne Rechtien, Laura Richert, Hadrien Lorenzo, Gloria Martrus, Boris Hejblum, Christine Dahlke, Rahel Kasonta, Madeleine Zinser, Hans Stubbe, Urte Matschl, Ansgar Lohse, Verena Krähling, Markus Eickmann, Stephan Becker, Selidji Todagbe Agnandji, Sanjeev Krishna, Peter G. Kremsner, Jessica S. Brosnahan, Philip Bejon, Patricia Njuguna, Marylyn M. Addo, Claire-Anne Siegrist, Angela Huttner, Marie-Paule Kieny, Vasee Moorthy, Patricia Fast, Barbara Savarese, Olivier Lapujade, Rodolphe Thiébaut, Marcus Altfeld, and Marylyn Addo. Systems vaccinology identifies an early innate immune signature as a correlate of antibody responses to the ebola vaccine rvsv-zebov. Cell Reports, 20 (9):2251–2261, 09 2017. ISSN 2211-1247. doi: 10.1016/j.celrep.2017.08.023. URL http://dx.doi.org/10.1016/j.celrep.2017.08.023.

Adam J Rothman, Elizaveta Levina, and Ji Zhu. Generalized thresholding of large covariance matrices. Journal of the American Statistical Association, 104(485): 177–186, 2009.

Robert Sabatier, Myrtille Vivien, and Pietro Amenta. Two approaches for discriminant partial least squares. In Between data science and applied data analysis, pages 100–108. Springer, 2003.

Michael Sjöström, Svante Wold, and Bengt Söderström. Pls discriminant plots. In Pattern Recognition in Practice, Volume II, pages 461–470. Elsevier, 1986.

Daniel J Stekhoven and Peter Bühlmann. Missforestnon-parametric missing value imputation for mixed-type data. Bioinformatics, 28(1):112–118, 2011.

Arthur Tenenhaus and Michel Tenenhaus. Regularized generalized canonical correlation analysis. Psychometrika, 76(2):257, 2011.

Arthur Tenenhaus, Cathy Philippe, Vincent Guillemot, Kim-Anh Le Cao, Jacques Grill, and Vincent Frouin. Variable selection for generalized canonical correlation analysis. Biostatistics, 15(3):569–583, 2014.

Robert Tibshirani. Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society. Series B (Methodological), pages 267–288, 1996.

Olga Troyanskaya, Michael Cantor, Gavin Sherlock, Pat Brown, Trevor Hastie, Robert Tibshirani, David Botstein, and Russ B Altman. Missing value estimation methods for dna microarrays. Bioinformatics, 17(6):520–525, 2001.

LE Wangen and BR Kowalski. A multiblock partial least squares algorithm for investigating complex chemical systems. Journal of chemometrics, 3(1):3–20, 1989.

Johan A Westerhuis and Age K Smilde. Deflation in multiblock pls. Journal of chemometrics, 15(5):485–493, 2001.

Johan A Westerhuis, Pierre MJ Coenegracht, and Coenraad F Lerk. Multivariate modelling of the tablet manufacturing process with wet granulation for tablet optimization and in-process control. International journal of Pharmaceutics, 156 (1):109–117, 1997.

Herman Wold. Estimation of principal components and related models by iterative least squares. Multivariate analysis, pages 391–420, 1966.

S Wold. Three pls algorithms according to sw. In Proc.: Symposium MULDAST (multivariate analysis in science and technology), pages 26–30, 1984.

Svante Wold, Harold Martens, and H Wold. The multivariate calibration problem in chemistry solved by the pls method. In Matrix pencils, pages 286–293. Springer, 1983.

Hui Zou, Trevor Hastie, and Robert Tibshirani. Sparse principal component analysis. Journal of computational and graphical statistics, 15(2):265–286, 2006.

2.2 Is the block structure interesting with no missing values ?

Here is explored the behavior of the ddsPLS method in the context of non missing samples. More precisely, one would like to know whether or not the block structure impacts the regression matrices if there are no missing values.
Indeed, the ddsPLS permits easy interpretations due to its hierarchical estimation, particularly for the explained variances per component or per block or per variable. This point is largely discussed in the part 3, dedicated to the presentation of the R package ddsPLS.
The following section tries to answer that question in a theoretical way. The first following section concentrates on the case where the number R of components, of the ddsPLS model built by the user, is equal to the number of variables in the response matrix, q, in a regression problem.
The next section, based on numerical simulations, answers the same question if the number of axes of the model is not equal to q.

Remark 2.1. *If some blocks have a low number of covariates regarding to the number of axes the user wants to build, say $\exists t \in [\![1,T]\!] | p_t < R$, then the convention is to build the p_t components for that block using the ddsPLS algorithm and to complete the $R - p_t$ other components with null components.*

2.2.1 For Complete Models, *i.e.* when $R=q$

In that section we prove that under some hypotheses, a ddsPLS model built on a multiblock data set is equivalent to a ddsPLS model built on the monoblock data set corresponding to the column-concatenated covariate matrix data set, if the number R of components desired by the user is equal to the number q of variables in the response matrix. Let us first consider the following lemma.

Lemma 2.1. $\forall (n,q,p) \in (\mathbb{N}^\star)^3$, $\forall (\mathbf{X},\mathbf{Y}) \in \mathbb{R}^{n\times p} \times \mathbb{R}^{n\times q}$ *with no missing entries and* $\forall (\lambda, R) \in [0,1] \times [\![1, min(n,q,p)]\!]$, *let*

$$\mathcal{M} := \texttt{mdd-sPLS}(\mathbf{X},\mathbf{Y},\lambda,R),$$

then $\mathcal{M}\left[\underline{\boldsymbol{\beta}}\right] = \mathbb{I}_R$.

Proof. In the following, all the elements refer to output from object $\mathcal{M}$. By definition, $\underline{\boldsymbol{\beta}} = \mathbf{SVD}_R(\mathbf{MU})$, where $\mathbf{M} = S_\lambda\left(\frac{\mathbf{Y}^T\mathbf{X}}{n-1}\right)$, but $\mathbf{U} = \mathbf{SVD}_R(\mathbf{M})$. Using the unicity of the R-dimensional SVD decomposition of a matrix, under ± 1 multiplicative coefficients it becomes that $\mathbf{SVD}_R(\mathbf{MU})$ is a diagonal matrix with diagonal elements randomly equal to ± 1. For the sake of simplicity we choose the one which is equal to the identity. This is $\underline{\boldsymbol{\beta}}$ and this permits to conclude the demonstration. □

Theorem 2.1. *Let be defined,* $\forall (n,q,p_1,\cdots,p_T) \in (\mathbb{N}^\star)^{T+2}$, $\forall (\mathbf{X}_1,\cdots,\mathbf{X}_T,\mathbf{Y}) \in \mathbb{R}^{n\times p_1}\times\cdots\times \mathbb{R}^{n\times p_T} \times \mathbb{R}^{n\times q}$ *with no missing entries,* $\forall \lambda \in [0,1]$ *and if $R = q$, one defines*

- $\mathcal{M}_m := \texttt{mdd-sPLS}(\mathbf{X} = \{\mathbf{X_t}\}_{t\in[\![1,T]\!]}, \mathbf{Y}, \lambda, R = q)$,
- $\mathcal{M}_1 := \texttt{mdd-sPLS}(\mathbf{X} = [\mathbf{X_1},\cdots,\mathbf{X_T}], \mathbf{Y}, \lambda, R = q)$.

Then we have

$$\mathcal{M}_1\big[\mathbf{U}\big] = \underbrace{\mathcal{M}_m\big[\mathbf{U}\big]}_{\cdots\cdots\cdots}\mathcal{M}_m\big[\underline{\boldsymbol{\beta}}\big], \tag{2.1}$$

where "`ell` $\rightarrow$ $\underbrace{\texttt{ell}}_{\cdots}$*" is the function that concatenates along the diagonal the elements of the list "*`ell`*".*

Proof. Let be defined $\forall t \in [\![1,T]\!]$, $\mathbf{M}_t = S_\lambda\left(\frac{\mathbf{Y}^T\mathbf{X}_t}{n-1}\right)$ and $\mathbf{M} = [\mathbf{M}_1, \cdots, \mathbf{M}_T]$. Both of those matrices have a rank at most equal to q. Let be denoted by $\mathbf{U}_t = \mathcal{M}_m\big[\mathbf{U}\big]_t$ and $\mathbf{U} = \underbrace{\mathcal{M}_m\big[\mathbf{U}\big]}_{\cdots\cdots\cdots}$.
Finally let $\mathbf{Z} = \mathbf{M}\mathbf{U}$, which also equals $\mathbf{Z} = [\mathbf{M}_1\mathbf{U}_1, \cdots, \mathbf{M}_T\mathbf{U}_T]$.
Since $rank(\mathbf{M}_t) \leqslant q$ and $R = q$, then $\forall t \in [\![1,T]\!]$, $\mathbf{M}_t\mathbf{U}_t\mathbf{U}_t^T = \mathbf{M}_t$ and so $\mathbf{Z}\mathbf{U}^T = \mathbf{M}\mathbf{U}\mathbf{U}^T = \mathbf{M}$.
Under a norm-1 multiplicative real constant, $\mathbf{M}$ admits the following common *SVD* decomposition $\mathbf{Z} = \underline{\boldsymbol{\alpha}}\boldsymbol{D}\underline{\boldsymbol{\beta}}^T$, where $\boldsymbol{D}$ is a positive-diagonal matrix and both of the other matrices are column orthogonal. $\boldsymbol{\beta}$ is the super-weight of the **mdd-sPLS** decomposition parameterized by $(\mathbf{X} = \{\mathbf{X_t}\}_{t\in[\![1,T]\!]}, \mathbf{Y}, \overline{\lambda}, R = q)$.
Finally $\mathbf{Z}\mathbf{U}^T = \mathbf{M} = \underline{\boldsymbol{\alpha}}\boldsymbol{D}\underline{\boldsymbol{\beta}}^T\mathbf{U}^T = \underline{\boldsymbol{\alpha}}\boldsymbol{D}(\mathbf{U}\underline{\boldsymbol{\beta}})^T$ and under the same unicity arguments it becomes that $\mathbf{U}\underline{\boldsymbol{\beta}} = \mathbf{SVD}_R(\mathbf{M})$.
With $\overline{\mathbf{U}} := \underbrace{\mathcal{M}_m\big[\mathbf{U}\big]}_{\cdots\cdots\cdots}$ and $\underline{\boldsymbol{\beta}} := \mathcal{M}_m\big[\underline{\boldsymbol{\beta}}\big]$ applying lemma 2.1 to the model $\mathcal{M}_1$, this is clear that $\mathcal{M}_1\big[\mathbf{U}\big]\mathcal{M}_1\big[\underline{\boldsymbol{\beta}}\big] = \mathcal{M}_1\big[\mathbf{U}\big]$, and the proof is complete. □

This Theorem admits the following corollary.

Corollaire 2.1. *Under the hypotheses of Theorem 2.1, we have*

$$\mathcal{M}_1\big[\mathbf{B}\big] = \underline{\underline{\mathcal{M}_m\big[\mathbf{B}\big]}},$$

where "`ell` $\rightarrow$ $\underline{\underline{\texttt{ell}}}$*" is the function that concatenates along the rows the elements of the list "*`ell`*".*

Proof. The proof of that corollary uses the definition of $\mathcal{B}$ given in Step 3 of the ddsPLS algorithm detailed in the previous section where one can see that $\mathcal{M}_1\big[\mathbf{T}^\star\big] = \mathcal{M}_m\big[\mathbf{T}^\star\big]$ by application of Theorem 2.1 and by definition of $\mathbf{T}^\star$. □

Those results show that, in the context of non missing values, there is an equivalence between the multiblock structured models and the corresponding monoblock ones if the number of axes built equals the number q of response variables. The next section looks for the behavior of the ddsPLS method when $R < q$ context through simulation results.

2.2.2 For Incomplete Models, *i.e.* when *R<q*

It has been chosen to consider simulations to illustrate that case. Let us consider a 2-block supervised data-sets $(\mathbf{X}_1, \mathbf{X}_2, \mathbf{Y}) \in \mathbb{R}^{n\times p_1} \times \mathbb{R}^{n\times p_2} \times \mathbb{R}^{n\times q}$, where $q = q_1 + q_2$, such as

$$\begin{array}{rcl} \mathbf{Y}^{[\![1,q_1]\!]} & = & \mathbf{X}_1^{[\![1,q_1]\!]} + \boldsymbol{\epsilon}_1, \\ \mathbf{Y}^{[\![q_1+1,q_1+q_2]\!]} & = & \mathbf{X}_2^{[\![1,q_2]\!]} + \boldsymbol{\epsilon}_2, \end{array}$$

where $\boldsymbol{\epsilon}_1$ and $\boldsymbol{\epsilon}_2$ are n realisations of Gaussian multivariate variables $\mathcal{N}_q\left(\mathbf{0}, 0.05^2\mathbb{I}_q\right)$. The notation $\mathbf{Mat}^{[\![a,b]\!]}$ considers variables of **Mat** for indices between a and b. The covariate blocks are realisations of two independent Gaussian multivariate variables X_1 and X_2 such as

$$\begin{array}{rcl} X_1 & \sim & \mathcal{N}_{p_1}\left(\mathbf{0}, \mathbb{I}_{p_1}\right), \\ X_2 & \sim & \mathcal{N}_{p_2}\left(\mathbf{0}, \mathbb{I}_{p_2}\right). \end{array}$$

multiblock ddsPLS models, denoted as $\mathcal{M}_m$, and their associated monoblock models $\mathcal{M}_1$ are built for each simulated data sets taking $R \in [\![1, q]\!]$. Two descriptors are monitored :

— **The norm of differences of regression matrices**, which can be written

$$\left|\left|\mathcal{M}_1\big[\mathbf{B}\big] - \underline{\underline{\mathcal{M}_m\big[\mathbf{B}\big]}}\right|\right|_o ,$$

where ". $\rightarrow ||.||_o$" is the one norm (maximum absolute column sum), which is the default norm of the R function `norm`.

— **The monoblock to multiblock differences of explained variances**, which is the subtraction of the variance explained of the $\mathbf{Y}$ by the multiblock model to the monoblock model. If this is a positive value, the monoblock explains more variance than the multiblock value, and vice versa. The explained variance of $\mathbf{A}$ by $\mathbf{B}$ is defined as

$$f : (\mathbf{A}, \mathbf{B}) \longrightarrow \begin{cases} \frac{\left|\left|\mathbf{A}\left(\mathbf{A}^T\mathbf{A}\right)^+\mathbf{A}^T\mathbf{B}\right|\right|_F}{||\mathbf{B}||_F} & \text{if} \quad \mathbf{B} \neq \mathbf{0}, \\ 0 & \text{else,} \end{cases}$$

where ". $\rightarrow ||.||_F$" is the Frobenius norm. In that context a positive difference of 0.01 means that the monoblock model permits to explain 1% more variance than the multiblock model.

Effect of covariate block dimensions

Samples are simulated according to that scenario modifying p_1 and p_2 with $n = 100$ and $q_1 = q_2 = 10$ fixed for now. 100 simulation have been considered for different values of p_1 and p_2 and some corresponding results, through boxplots are detailed in figure 9.

The bottom left corner (red boxplots) represents the norm of regression matrix differences and top left corner (blue boxplots) represents the monoblock to multiblock differences of explained variances. These corners are divided in rows and columns corresponding to different values for p_1 and p_2 where x-axis of each plot represents values given to R. As proved in the next section, for both of the corners, the case $R = q$ corresponds to differences equal to 0. Let us remark that in the case $p_1 = p_2 = 10$, models are equivalent as soon as $R \geqslant 10$.

In the red corner, plots present model differences which are more variable as quantities p_1 and p_2 increase.

In the blue corner, let us remind that if the values are negative, then the multiblock model explains more variance of $\mathbf{Y}$ and if the values are positive then the associated monoblock model explains more variance of $\mathbf{Y}$. One might notice different cases which are :

1. If dimensions are large ($\min(p_1, p_2) = 1000$) then the differences are lower than in low dimensional cases. This corresponds to the last column of the blue corner.
2. If p_1 or p_2 is equal to 100, then monoblock models always outperforms multiblock models in terms of variance explained. Especially if p_1 and p_2 are very different.
3. If $p_1 \in \{20, 50\}$ and $p_2 \in \{10, 20, 50\}$, then the "curves" are minimum for $R = 10$ and this corresponds to negative values. In those contexts, multiblock models explain more variance than the associated monoblock models.

Moreover the variability, as a function of R, is always decreasing, even if the proportion of explained variances are around 1% of difference between the multiblock and its associated

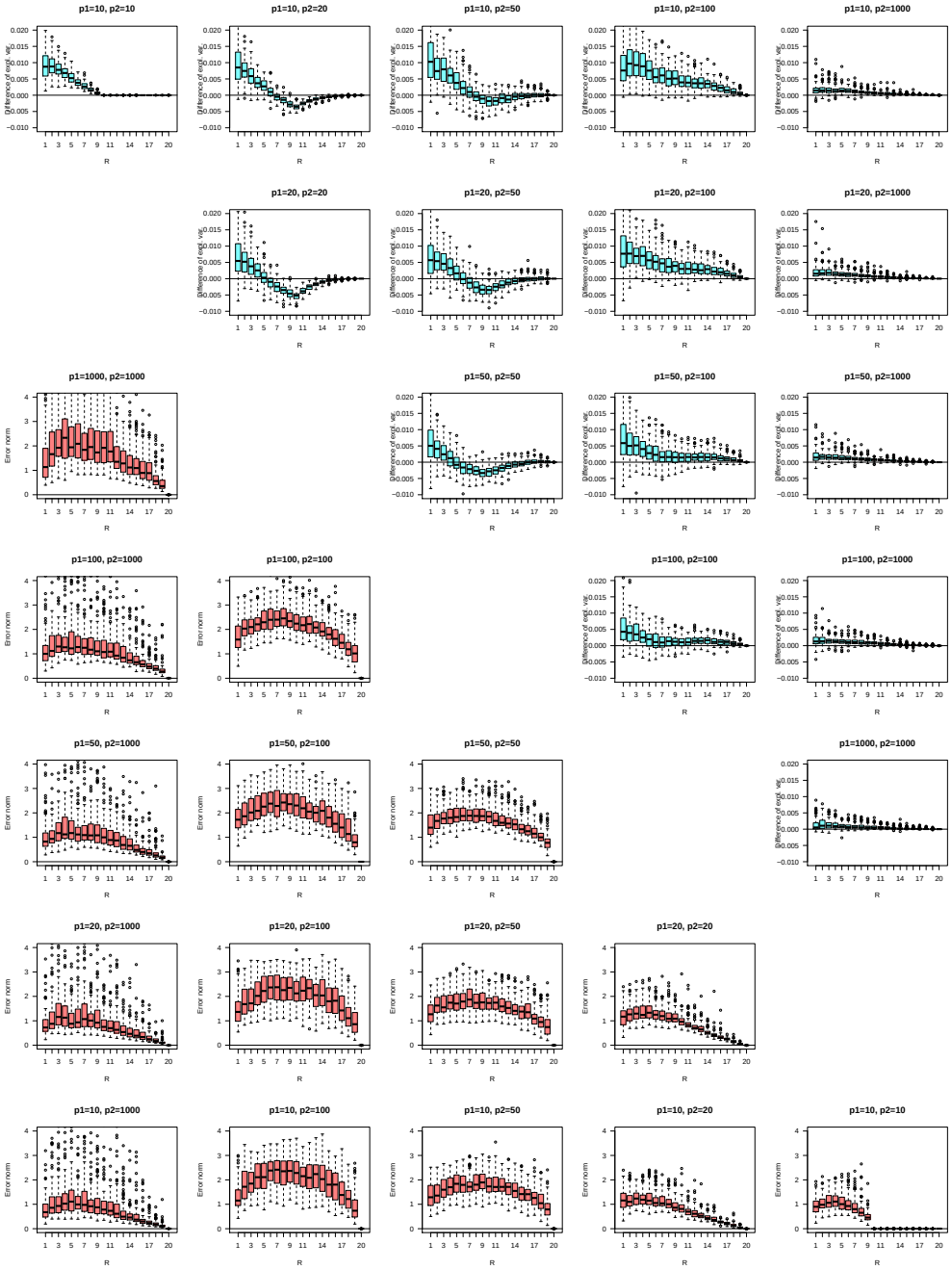


Figure 9 – "The norm of differences of regression matrices" in bottom left corner and filled in red : the higher it is the more different are the estimated regression matrices from both of the approaches. "The monoblock to multiblock differences of explained variances" in top right corner and filled in blue : a positive value, respectively negative value, means monoblock, respectively multiblock, performs better. Parameters are $(p_1, p_2) \in \{10, 20, 50, 100, 1000\}^2$ through 100 simulations per set of parameters for $q_1 = q_2 = 10$ and $n = 100$.

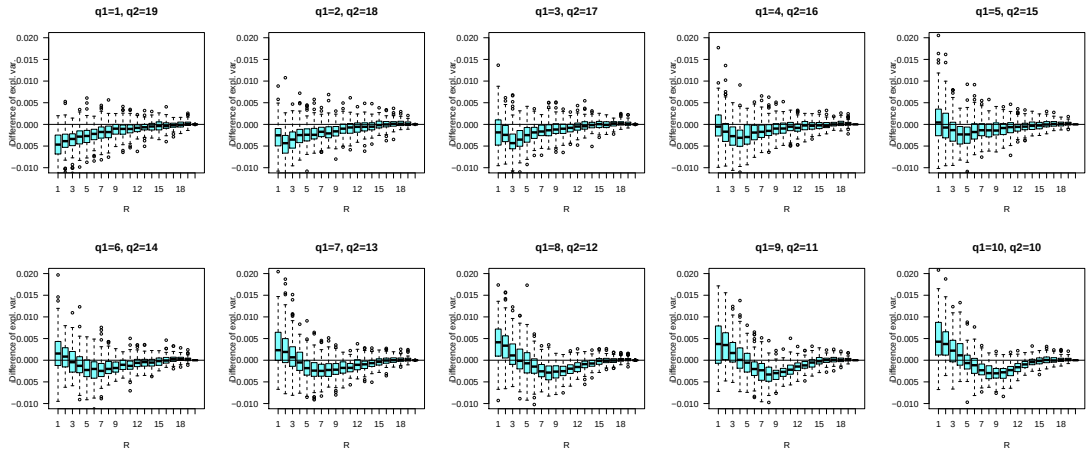


Figure 10 – "The monoblock to multiblock differences of explained variances" where a positive value, respectively negative value, means monoblock, respectively multiblock, performs better. Parameters are $p_1 = p_2 = 50$, through 100 simulations per set of parameters for $q_1 \in [\![1, 10]\!]$, and $q_2 = 20 - q_1$ and $n = 100$.

monoblock methods, it seems that there are different regimes corresponding to better behaviors of the multiblock or its associated monoblock description. In the case 3, it seems that the differences of explained variances are minimal when $R \approx 10$.

The next paragraphs go deeper in those problems.

When the multiblock outperforms its associated monoblock description ?

Previous simulations have showed interesting behaviors for $p_1 \in \{20, 50\}$ and $p_2 \in \{10, 20, 50\}$. That paragraph details results for $p_1 = p_2 = 50$ and $q_1 \in [\![1, 10]\!]$, $q_2 = 20 - q_1$. The same number of 100 simulations is performed per set of parameters.
Figure 10 gathers results regarding to the **monoblock to multiblock differences of explained variances**. In that context, the curve shows a clearer minimum for $R = 10$.
Figure 10 is divided in two rows corresponding to increasing q_1 and so decreasing q_2. In fact, it corresponds to a equalization of the presence of each covariate block in the response matrix. For the top left plot, 19 over 20 of the total response variables are from one covariate block while for the bottom right plot, both of the covariate blocks are present in the response in equal proportions. In those 10 plots, the minimum of the curves corresponds, approximately, to the minimum of q_1 and q_2, and this corresponds to cases where the multiblock approach outperforms its associated monoblock approach. As mentioned in the previous paragraph, when the contribution of each covariate block to the response is balanced, a low number of components is associated with a better behavior of the monoblock models.

Effect of sample size

Many *scenarii* might be performed to study the effect of the sample size n. Here are considered the worst scenario for the ddsPLS method according to the blue corner representations of Figure 9, and so $p_1 = 10$, $p_2 = 100$. In Section 2.2.2, it has been observed that a balanced influence from both blocks (i.e. $q_1 = q_2$) seems to promote monoblock descriptions and so q_1

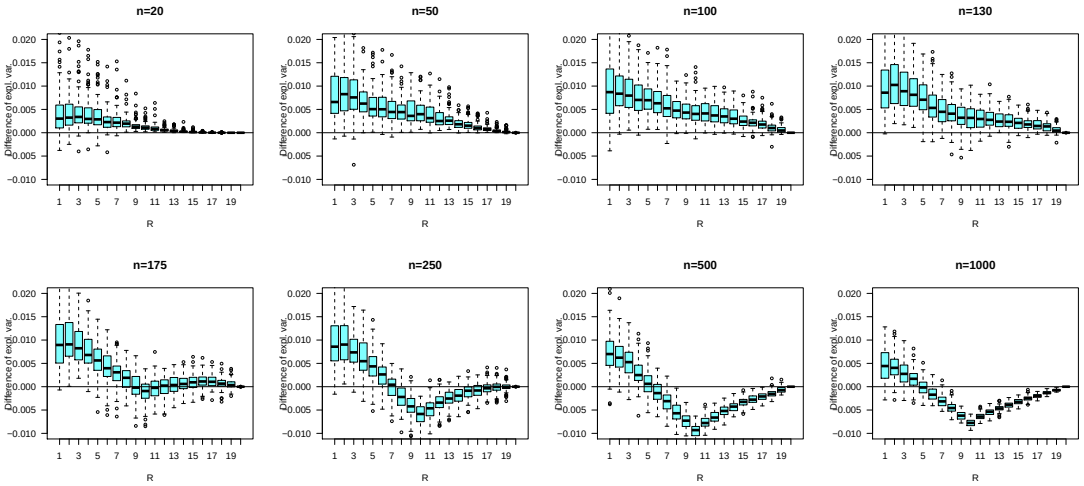


Figure 11 – "The monoblock to multiblock differences of explained variances" where a positive value, respectively negative value, means monoblock, respectively multiblock, performs better. Parameters are such as n is equal to 20, 50, 100, 130, 175, 250, 500 and 1000 through 100 simulations per set of parameters for $p_1 = 10, p_2 = 100, q_1 = q_2 = 10$.

and q_2 are chosen such as $q_1 = q_2 = 10$. Figure 11 gathers results of those simulations.
Variances shrink for large n's. For large n's also, the correct dimensional descriptions, $R \approx 10$, tend to present multiblock models as better descriptors of the data-set structure. But for small n, which is the desired field of application of the method, multiblock models seem to perform worse than its associated monoblock models.

Conclusion and further researches

The multiblock approach permits interpretable results, due to weights/components and super-weights/super-components, but its interest, in terms of explained variance, versus its associated monoblock model must be explored more deeply.
For low n, balanced block effects on the response and different block sizes (see section 2.2.2), especially for low R, monoblock models seem to outperform multiblock models. But multiblock models have the advantage when $R \approx 10$, which is the dimension of the intersect of each covariate block and the response block ($q_1 = q_2 = 10$). Actually, both of the blocks are supposed to be uncorrelated and this is a limitation of those analyses and it might be interesting to perform the same type of analyses with a correlation structure linking both of the blocks.
Moreover it has been chosen to keep only 2-blocks models, the effect of more groups should be investigated.

CHAPITRE 3

LE PACKAGE R DDSPLS

> *La maturité de l'homme est d'avoir retrouvé le sérieux qu'on avait au jeu quand on était enfant.*
>
> Alain Damasio, La Horde du Contrevent

Sommaire

La méthode discutée précédemment a été implémentée dans un package R nommé ddsPLS. Une version Python de ce package existe aussi mais est beaucoup moins aboutie, en termes de sorties graphiques et d'aide à l'interprétation. En effet, il a été choisi de valoriser en priorité l'implémentation R dans notre cas.

Nous reprenons ici le corps d'un article soumis au journal "The R Journal". Ce dernier est rédigé en langue anglaise.

3.1 ddsPLS : A Package to Deal with Multiblock Supervised Problems with Missing Samples in High Dimension

ddsPLS: A Package to Deal with Multiblock Supervised Problems with Missing Samples in High Dimension

by Hadrien Lorenzo, Jérôme Saracco and Rodolphe Thiébaut

Abstract The ddsPLS method considers regression and classification problems in the context of multiblock structured covariate data sets taking into account missing samples. The response outcome can be multidimensional and constitutes the response block. Multiblock covariate data sets are constituted by heterogeneous types of data (e.g. generated by various bioassays) and/or temporal realisations of the same covariates. The ddsPLS approach only deals with a specific case of missing values, denoted as missing samples, which means that some individuals have missing values for all the covariates in a given block. The ddsPLS method performs imputation of missing data, covariate selection and prediction of the response. The method is based on singular value decompositions of soft-thresholded covariance blocks via a three-steps procedure. The tuning parameters of the ddsPLS method can be optimized through an available cross-validation procedure. The **ddsPLS** package is illustrated on a real data set.

Introduction

Analyzing high dimensional data where the number of predictors is much higher than the number of statistical units (e.g. individuals) is challenging in mathematical and practical point of views. Regularization solutions, such as the regularization model introduced by Tikhonov and Arsenin and translated by Willoughby (1979), show theoretical and also practical solutions to the high dimension problem. To introduce sparsity in the models, Tibshirani (1996) has developed a regularization method named Lasso, close in form to the Tikhonov solution. Putting to 0 the coefficient associated with unimportant covariates allows the selection of the most relevant covariates and thus improve the interpretability of the model.

The Lasso method has been adapted to high correlation structured data sets thanks to the Elastic-Net model by Tibshirani (1996). It has also been adapted to grouped data sets, i.e. when the covariates can be divided in *a priori* meaningful groups leading to the so-called group-Lasso and sparse group-Lasso methods (see Bakin, 1999; Simon et al., 2013). However, those methods only consider an unidimensional response variable.

Multiple response predictions can be performed with PLS (Partial Least Squares) which is introduced by Wold et al. (2001). PLS methods work with covariance structures using singular value decompositions, denoted as SVD in the following. Their flexibility allows to develop multiblock covariate typed methods such as the multiblock PLS algorithm, denoted as MBPLS, and developped by Wangen and Kowalski (1989). Then, using classical regularization approaches, a sparse version of the PLS has been proposed by Lê Cao et al. (2008) using Lasso and by Liquet et al. (2015) using group Lasso. Multiblock methods with regularization and covariate selection have been generalized through RGCCA and SGCCA (see Tenenhaus and Tenenhaus, 2011; Tenenhaus et al., 2014) and gather canonical correlation analyses and PLS methods in a regularized general framework. The package **RGCCA** implements different solutions associated with that context.

Missing values occur for numerous reasons. Here, we consider the simplest case referred as *MCAR* (Missing Completely At Random), as detailed by Rubin (1976). In that context, no association exists between the missing probability of a realization and any measured or unmeasured covariate. An usual approach to deal with missing data in such context is to remove units/individuals with missing information. Even if not biased, this complete case analysis shrinks the statistical power and must not be used such as shown in Josse et al. (2009). Obviously, the issue is even more pronounced when the number of samples is low. Another approach is to complete missing data by imputation. The model for imputation could be very simple such as the imputation of the same fixed value for all missing observations based on the mean or the median of observed values. This leads to hard modifications of the covariance structures (Schafer and Graham, 2002) and should therefore be avoided. The alternative is to use the information carried by the observed covariates to drive the imputation of missing data. Since the imputation itself modifies the covariance structure, it is reasonable to consider iterative methods, close to the *EM* (*Estimation-Maximization*) algorithm described by Dempster et al. (1977). Those *EM-typed* methods give valid results in the context of imputation as detailed in Mazumder et al. (2010) for the SVD of very large matrices with sparse structure and a large proportion of missing values. The R package **softImpute** (Hastie and Mazumder, 2015) allows to use that method. Stekhoven and

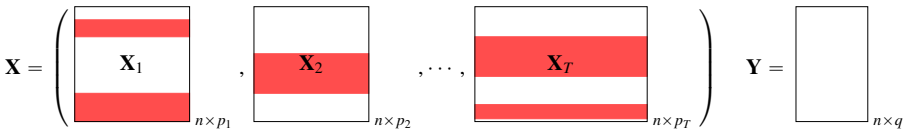


Figure 1: A multiblock data set, of T blocks, where missing values are represented by red areas. n is the number of samples, q is the number of variables in the response matrix and $(p_t)_{t\in[\![1,T]\!]}$ are the number of covariates in each covariate block.

Bühlmann (2011) have developed an *EM-typed* algorithm for random forest objects implemented in the **missForest** package. For unsupervised analyses, Josse and Husson (2012) have proposed a regularized PCA (Principal Component Analysis) which imputes missing values through an *EM* approach. The **missMDA** (see Josse and Husson, 2016) package allows to use that method to multiblock data sets, through the `imputeMFA()` function. The task view *MissingData* gives more examples of packages doing imputation.

The *CMTF* (*Collective Matrix Tensor Factorization*) problem (see for example Acar et al., 2011, 2013) has been designed to deal with heterogeneous problems where a collection of matrices sharing the same structure (which is called a tensor) and an objective matrix ($\mathbf{Y}$ in our case) share a common description of the same individuals. The *CMTF-WOPT* algorithm (*CMTF-Weighted Optimal Collective Matrix Tensor Factorization*) is a modification of *CMTF* which allows to deal with missing values in the collection of covariate matrices by ignoring missing values in the data structure estimation.

The **ddsPLS** package (*data-driven sparse Partial Least Squares*) proposes an *EM-typed* algorithm dedicated to multiblock supervised for classification or multivariate regression taking into account missing samples in the covariate part and also providing covariate selection in order to predict the response part. Figure 1 shows the context of application of the method for a T covariate blocks measured on the same n samples where each block $t \in [\![1, T]\!]$ describes each sample with $p_t \in \mathbb{N}^\star$ covariates. The response block $\mathbf{Y}$ contains $q \in \mathbb{N}^\star$ response variables measured on the same n samples. The `ddsPLS` approach only deals with a specific case of missing values (which often occurs in health data problems), denoted as missing samples, which means that some individuals have missing values for all the covariates in a given block. In Figure 1, red areas correspond to missing samples.

That package has been used through three applications presented in Lorenzo et al. (2019a,b); Ellies-Oury et al. (2019).

The `ddsPLS` method needs to fix two regularization parameters (the number R of components to retain, and the maximum number L_0 of covariates to consider in prediction). Cross-validation can be used to determine those parameters from the available data and a function has been implemented to perform such optimization.

The **ddsPLS** package is based on three S3 classes which are

- the class `"mddsPLS"` provides regression or classification pieces of information of a prediction model (including imputation),
- the class `"perf_mddsPLS"` gathers cross-validation results,
- the third class `"MddsPLS_core"` is never directly built by the user but in each object of the class `"mddsPLS"`, an attribute is an object of the class `"MddsPLS_core"`.

Their associated S3 methods are detailed in Table 1.

In the following, the `ddsPLS` method is briefly presented. Then, the main functions associated with `"mddsPLS"` (estimation/imputation, prediction) and `"perf_mddsPLS"` (cross-validation to tune both of the parameters of the model) classes are described and illustrated on a real data set in a regression context. No example of classification analysis is provided but the last section provides key indications to perform this type of analyses.

Brief description of the ddsPLS method

The `ddsPLS` method is able to impute missing samples (in covariate blocks) in a supervised context (regression or classification), to estimate the underlying model and to make predictions. A full description of the `ddsPLS` method is available in Lorenzo et al. (2019b).

The `ddsPLS` method only needs two parameters to be tuned by the user:

Object	Type	Accessible	Description
`"MddsPLS_core"`	S3 class	No	The core function of the `mddsPLS` class.
`"mddsPLS"`	S3 class	Yes	Main class.
`plot.mddsPLS()`	S3 method	Yes	The plot method
`predict.mddsPLS()`	S3 method	Yes	The predict method.
`summary.mddsPLS()`	S3 method	Yes	The summary method.
`"perf_mddsPLS"`	S3 class	Yes	Function to estimate cross -validation performances.
`plot.perf_mddsPLS()`	S3 method	Yes	The plot method
`summary.perf_mddsPLS()`	S3 method	Yes	The summary method.

Table 1: Main functions of the **ddsPLS** R package

- the number of components $R \in [\![1, min(n, p, q)]\!]$, equivalent to the number of components in PLS or PCA,
- the regularization parameter which can be controlled through one of the following two parameters, depending on user's choice:
  - $\lambda \in [0, 1]$ – `lambda` in the software – which corresponds to the minimum value of the correlation between a covariate and a response variable to take into account in the model,
  - $L_0 \in [1, p]$ – `L0` in the package – which represents the maximum number of covariates to be include in the model.

Those parameters (R, λ) or (R, L_0) can be optimally determined by cross-validation through the `"perf_mddsPLS"` class.

For given values of these two parameters, the class `"mddsPLS"` builds the corresponding model (including imputation of missing samples), and this model can then be used for predictions.

Notations

Let us consider a multiblock data set built on T blocks $(\mathbf{X}_1, \cdots, \mathbf{X}_T)$ with the additional response block, $\mathbf{Y}$, that supervises the analysis. Given any block $t \in [\![1, T]\!]$ and a number R of built components, let us define

- $\mathbf{M}_t = S_\lambda\left(\frac{\mathbf{Y}^T\mathbf{X}}{n-1}\right) \in \mathbb{R}^{q \times p_t}$, the soft-thresholded covariance matrix between the response block $\mathbf{Y}$ and the covariate block $\mathbf{X}_t$, where $S_\lambda : \quad x \to \text{sign}(x)\left(|x| - \lambda\right)_+$ is the soft-thresholding function.
- $\lambda \in [0, 1]$, the regularization coefficient, which can also be parameterized through the L_0 parameter. L_0 corresponds to the number of columns of
$$S_\lambda\left(\frac{\mathbf{Y}^T\left[\mathbf{X}_1, \cdots, \mathbf{X}_T\right]}{n-1}\right)$$
that must be put to 0 with the lowest value of λ (the smallest degree of regularization).
- $\mathbf{U}_t = \mathbf{SVD}_R(\mathbf{M}_t) \in \mathbb{R}^{p_t \times R}$ the *weight* matrix where ". $\to \mathbf{SVD}_R(.)$" gives the R right singular vectors associated with the R largest singular values.
- $\mathbf{Z} = [\mathbf{M}_1\mathbf{U}_1, \cdots, \mathbf{M}_T\mathbf{U}_T] \in \mathbb{R}^{q \times R}$, the concatenation of the projected soft-thresholded covariance matrices.
- $\underline{\boldsymbol{\beta}} = \mathbf{SVD}_R(\mathbf{Z}) \in \mathbb{R}^{RT \times R}$, the *super-weights* per block per axis concatenated per row. Defining also $\underline{\boldsymbol{\beta}} = \left[\boldsymbol{\beta}_1^T, \cdots, \boldsymbol{\beta}_T^T\right]^T$, where $\boldsymbol{\beta}_t \in \mathbb{R}^{R \times R}$.
- $\mathbf{U}_t^\star = \mathbf{U}_t\boldsymbol{\beta}_t \in \mathbb{R}^{p_t \times R}$, the *scaled super-weights* per block.
- $\mathbf{T}_t = \mathbf{X}_t\mathbf{U}_t \in \mathbb{R}^{n \times R}$, the *scores* per block per axis.
- $\mathbf{T}^\star = \sum_{t=1}^T \mathbf{X}_t\mathbf{U}_t^\star \in \mathbb{R}^{n \times R}$, the *super-scores*.
- $\mathbf{V}^\star = \text{norm}_2(\mathbf{Z}\underline{\boldsymbol{\beta}}) \in \mathbb{R}^{q \times R}$, the *weights* of the response, where the function ". $\to \text{norm}_2(.)$" returns the columns normalized to a $\mathcal{L}_2$-norm equal to 1 if the corresponding column is non null and returns the null-column otherwise.
- $\mathbf{S}^\star = \mathbf{Y}\mathbf{V}^\star \in \mathbb{R}^{n \times R}$, the *super-scores* of the response.
- $\mathcal{B} = (\mathbf{B}_t \in \mathbb{R}^{p_t \times q})_{t \in [\![1,T]\!]}$ the list of regression matrices per block.

The imputation algorithm

For a given couple of tuning parameters (R, λ), the algorithm denoted as *Koh-Lanta* is an *EM-typed* algorithm applying the following three-steps procedure to the complete data set

- STEP 1: solve independently the T mono-block `ddsPLS` problems based on $\mathbf{X}_t$ and $\mathbf{Y}$. This allows to calculate, $(\mathbf{U}_t)_{t\in[\![1,T]\!]}$ and $\mathbf{T}$.
- STEP 2: combine information from the T blocks. This allows to calculate $\underline{\boldsymbol{\beta}}$, $\forall t \in [\![1, T]\!]$ $\mathbf{U}_t^\star$, $\mathbf{V}^\star$, $\mathbf{T}^\star$ and $\mathbf{S}^\star$.
- STEP 3: predict $\mathbf{Y}$ using a regression model based on the T blocks. This allows to calculate, $\forall t \in [\![1, T]\!]$, $\mathbf{B}_t = \mathbf{U}_t^\star \mathbf{B}_0 \mathbf{V}^{\star T}$, where $\mathbf{B}_0 = \left(\mathbf{T}^{\star T}\mathbf{T}^\star\right)^+ \mathbf{T}^{\star T}\mathbf{S}^\star$ is the linear regression matrix of $\mathbf{S}^\star$ on $\mathbf{T}^\star$ estimated through minimization of *OLS* (*Ordinary Least Squares*, see Kenney and Keeping, 1962, for example) and "$\to (.)^+$" denotes the Moore-Penrose pseudo-inverse operator (see Planitz, 1979).
  The objective regression model is
  $$\mathbf{Y} \approx \sum_{t=1}^{T} \mathbf{X}_t \mathbf{B}_t.$$

For a given couple of tuning parameters (R, λ), the *Koh-Lanta* algorithm allows to impute the missing samples using the above algorithm in an iterative way, alternating model building and missing data prediction. Its specificity is to deal with missing values in the train, in a first part, and in the test data sets, in a second part. More precisely :

- *The Tribe Stage*, the train data set imputation:
  1. For each block $\mathbf{X}_t$, with missing values, predict the missing values using a mono-block `ddsPLS` model taking $\mathbf{Y}$ as covariates and $\mathbf{X}_t$ as response. Note that non missing samples are used for training the underlying model.
  2. Build the multiblock `ddsPLS` model with covariate blocks $\{\mathbf{X}_1, \cdots, \mathbf{X}_T\}$ and response block $\mathbf{Y}$. Keep in memory the variables selected in each block.
  3. Re-do step 1 with only the selected variables of step 2 as response.
  4. Iterate steps 1 and 2 until stabilization of the super-scores.

  The model built in step 2 during the last iteration is then used in the following test data set imputation.
- *The Reunification Stage*, the test data set imputation :
  For each individual i, if there is at least one missing sample, split the train data set in 2 multiblock data sets. The first one, denoted as $B_i^{(1)}$, corresponds to the blocks where i has missing values and the second one, denoted as $B_i^{(2)}$, to the non missing blocks. The algorithm builds a prediction model using scores of $B_i^{(2)}$ as covariates and the concatenation of the blocks $B_i^{(1)}$ as response matrix. The missing values are then calculated according to that model and $B_i^{(2)}$ data can be projected onto the scores thanks to the train model.

In the following, the **ddsPLS** package formalism is illustrated on a toy example (based on a real data).

Build a model of prediction

The `"mddsPLS"` class allows to build prediction models (including imputation and variable selection). The `mode` argument allows to choose between regression or classification. The arguments of that class are detailed in Table 2.

Illustration on a toy example

To illustrate the use of the package, the `liverToxicity` data set (see Bushel et al., 2007) is used ($n = 64$, $p = 3116$ covariates, $q = 10$ response variables). The covariate part has been arbitrarly split in two blocks denoted as `Block_1` and `Block_2`. Moreover, missing samples have been generated as follows: the first five samples of `Block_1` have been deleted and the next five samples of `Block_2` have also been deleted. This has been done with the following code:

Argument	Description
Xs	A matrix (covariate block), if there is only one block. A list of matrices, if there is more than one block, of n rows each, the number of individuals. Some rows can be missing. The different matrices (blocks) can have different numbers of columns. The length of Xs is denoted by T and corresponds to the number of blocks.
Y	A matrix of n rows or a vector of length n detailing the response matrix. No missing values are allowed in that matrix.
lambda	A real $\in [0,1]$. This is a regularization parameter to select variables. The closer to 1, the less variables are selected. Fix this parameter or L0.
L0	A non null integer which is the largest number of covariates that can be selected in all the T covariate blocks. Fix this parameter or lambda.
R	A strictly positive integer corresponding to the number of components to build in the model.
mode	A character chain. Possibilities are 'reg', 'lda' or 'logit', which correspond respectively to regression problem, linear discriminant analysis and logistic regression (through functions glm() and lda(), respectively, from the package **MASS**). Default is 'reg'. Note that, for classification, 'logit' does not allow to deal with more than two classes.
keep_imp_mod	Logical. Whether or not to save imputation "mddsPLS" models which are created in the *Koh-Lanta tribe stage* imputation step. Default to FALSE. Be careful because each of the model savings can be memory demanding.
reg_imp_model	Logical. Whether or not to regularize the imputation models. TRUE by default.
errMin_imput	Positive real. Convergence threshold in the *Tribe Stage* of the *Koh-Lanta* algorithm. Default is $1e-9$.
maxIter_imput	Positive integer. Maximum number of iterations in the *Tribe Stage* of the *Koh-Lanta* algorithm. Default is 5.
verbose	Logical. If TRUE, the convergence progress of the *Koh-Lanta* algorithm is reported. Default is FALSE.
NZV	Float. The floating value above which the weights are set to 0.
getVariances	Logical. Whether or not to compute explained variances and RV coefficients of the model. Useful for interpretation but time consuming. Default is TRUE.

Table 2: Arguments of the class "mddsPLS"

```
library(ddsPLS)
data("liverToxicity")
X <- scale(liverToxicity$gene)
Xs <- list(Block_1 = X[, 1:1910], Block_2 = X[, -(1:1910)])
Xs[[1]][1:5, ] = Xs[[2]][6:10, ] <- NA
Y <- scale(liverToxicity$clinic)
```

Let us first illustrate the main function mddsPLS(). For easy readability of the outputs, let us first consider $R = 3$ and $L_0 = 10$. These two parameters will be tuned in the last part via cross-validation with the specific function perf_mddsPLS().

```
model <- mddsPLS(Xs = Xs, Y = Y, L0 = 10, R = 3, mode = 'reg')
```

Some useful outputs

The most important outputs, from the user point of view, are :

- var_selected: the list of the weights and super-weights per block and per component. This object is useful for technical reasons. The plot method allows helpful visualizations.
- Xs: the list of the covariate blocks imputed with the *Koh-Lanta* algorithm.
- mod: an object of the class "MddsPLS_core", whose class is described at the end of the paper. It gathers all the quantities calculated in the algorithm.

The summary method

The `mddsPLS.summary()` S3 method provides useful information divided into several parts:

- the main information about the data and the model, just above the title `ddsPLS object description`.
- the description of the quality of association between the scores and the super-scores with the response variables through two different descriptors (explained variances and RV coefficients which are defined below).
- the missing value information,
- and an overview of the selection results of the algorithm, for the `X` and the `Y` parts, per block and per component. The complete list of the selected covariates is one of the three most useful outputs (`var_selected`) of the model.

The two descriptors (explained variances and RV coefficients) are defined as follows.

- The `Explained Variance`, in percent, is based on

$$f : (\mathbf{A}, \mathbf{B}) \longrightarrow \begin{cases} \frac{\left|\left|\mathbf{A}\left(\mathbf{A}^T\mathbf{A}\right)^+\mathbf{A}^T\mathbf{B}\right|\right|_F}{||\mathbf{B}||_F} \times 100 & \text{if} \quad \mathbf{B} \neq 0 \\ 0 & \text{else} \end{cases},$$

  where "$. \to ||.||_F$" is the Frobenius norm.
- The `RV coefficient` generalizes the correlation notion to matrices, see Robert and Escoufier (1976), can be computed using

$$g : (\mathbf{A}, \mathbf{B}) \longrightarrow \begin{cases} \frac{\mathrm{Tr}\left(\mathbf{A}\mathbf{A}^T\mathbf{B}\mathbf{B}^T\right)}{\mathrm{Tr}(\mathbf{A}\mathbf{A}^T)\mathrm{Tr}(\mathbf{B}\mathbf{B}^T)} \times 100 & \text{if} \quad \mathbf{A} \neq 0 \text{ and } \mathbf{B} \neq 0 \\ 0 & \text{else.} \end{cases}.$$

Those two functions, f and g, are used in the `summary` method in 4 different ways corresponding to 4 different association aspects of the built scores with the response variables. In the following list, the calculations of each of those 4 association descriptors are described.

- *Total Y with all the Super Components*: this is one single value corresponding to the overall association of the final super components and the response matrix.

$$\begin{array}{ll} \texttt{Explained Variance :} & f\left(norm_2\left(\mathbf{T}^\star\right), norm_2\left(\mathbf{Y}\right)\right) \\ \texttt{RV coefficient :} & g\left(norm_2\left(\mathbf{T}^\star\right), norm_2\left(\mathbf{Y}\right)\right) \end{array}$$

- *Total Y variance explained by each Super Component*: this is a vector whose length is equal to the number of components in the model. It allows to evaluate the association of each super component separately on the response matrix.

$$\forall r \in [\![1, R]\!], \quad \begin{array}{ll} \texttt{Explained Variance :} & f\left(norm_2\left(\mathbf{T}^{\star(r)}\right), norm_2\left(\mathbf{Y}\right)\right) \\ \texttt{RV coefficient :} & g\left(norm_2\left(\mathbf{T}^{\star(r)}\right), norm_2\left(\mathbf{Y}\right)\right) \end{array}$$

  where $\mathbf{A}^{(r)}$ denotes the rth column of the matrix $\mathbf{A}$.
- *Marginal Y variable variance explained by each Super Component*: this is a matrix with q rows and R columns that shows the association between each super component and each response variable.

$$\forall (r, j) \in [\![1, R]\!] \times [\![1, q]\!], \quad \begin{array}{ll} \texttt{Explained Variance :} & f\left(norm_2\left(\mathbf{T}^{\star(r)}\right), norm_2\left(\mathbf{Y}^{(j)}\right)\right) \\ \texttt{RV coefficient :} & g\left(norm_2\left(\mathbf{T}^{\star(r)}\right), norm_2\left(\mathbf{Y}^{(j)}\right)\right) \end{array}$$

- *Total Y variance explained by each component of each block*: this is a matrix with T rows and R columns corresponding to the association between each covariate block and each of its corresponding components.

$$\forall (r, t) \in [\![1, R]\!] \times [\![1, T]\!], \quad \begin{array}{ll} \texttt{Explained Variance :} & f\left(norm_2\left(\mathbf{T}_t^{(r)}\right), norm_2\left(\mathbf{Y}\right)\right) \\ \texttt{RV coefficient :} & g\left(norm_2\left(\mathbf{T}_t^{(r)}\right), norm_2\left(\mathbf{Y}\right)\right) \end{array}$$

The corresponding output of the `summary` is provided below.

```
summary(model)
============================================================
           ddsPLS object description
============================================================

Number of blocks: 2
Number of dimensions: 3
Regularization coefficient: 10
Number of individuals: 64
Number of variables in Y part: 10
Model built in mode regression
Maximum number of iterations in the imputation process: 50
Algorithm of imputation has converged


     Variance Explained (%)
-------------------------------------------
Total Y variance explained by all the Super Components
[1] 67
Total Y variance explained by each Super Component
[1] 60 57 54

Marginal Y variable variance explained by each Super Component
          Super Comp. 1 Super Comp. 2 Super Comp. 3
BUN.mg.dL.         69.0     66.0     54.0
Creat.mg.dL.        15.0     18.0      9.2
TP.g.dL.           2.4      1.1     15.0
ALB.g.dL.          16.0     11.0      7.5
ALT.IU.L.          96.0     88.0     84.0
SDH.IU.L.          11.0     21.0     24.0
AST.IU.L.          95.0     85.0     83.0
ALP.IU.L.          38.0     42.0     33.0
TBA.umol.L.         84.0     83.0     90.0
Cholesterol.mg.dL.     66.0     60.0     46.0

Total Y variance explained by each component of each block
    Comp. 1 Comp. 2 Comp. 3
Block_1   59   58    0
Block_2   61   54   21


          RV coefficient
-----------------------------------------------------------
Total Y with all the Super Components
[1] 0.33

Total Y with each Super Component
[1] 0.37 0.32 0.29

Each Y variable with each Super Component
          Super Comp. 1 Super Comp. 2 Super Comp. 3
BUN.mg.dL.        0.47000    0.44000    0.3000
Creat.mg.dL.       0.02100    0.03300    0.0084
TP.g.dL.          0.00058    0.00013    0.0220
ALB.g.dL.         0.02400    0.01100    0.0056
ALT.IU.L.         0.93000    0.77000    0.7100
SDH.IU.L.         0.01200    0.04300    0.0590
AST.IU.L.         0.91000    0.73000    0.6900
ALP.IU.L.         0.14000    0.17000    0.1100
TBA.umol.L.        0.71000    0.68000    0.8100
Cholesterol.mg.dL.    0.44000    0.36000    0.2100

Total Y with each component of each block
    Comp. 1 Comp. 2 Comp. 3
```

```
Block_1  0.35  0.34  0.000
Block_2  0.37  0.29  0.042


     Missing value information
-----------------------------------------

                Block_1 Block_2  Total
Number of covariates        1910.00 1206.00 3116.00
Number of missing samples      5.00  5.00  10.00
Proportion of missing samples (%)  7.81  7.81  7.81


       mddsPLS results
-----------------------------------------

At most 10 covariate(s) can be selected in the X part


 ---- For each block of X, are selected
    Super Comp. 1 Super Comp. 2 Super Comp. 3
Block_1       4       4       0
Block_2       5       5       1
 ---- For the Y block, are selected
    @ (2,2,1) variable(s)


         Thank s for using me
------------------------------------------------------------
                    Hadrien Lorenzo
               hadrien.lorenzo.2015@gmail.com
============================================================
```

In the following, the different visualizations available via the `plot` function are described.

Visualization of the weights

The `weights`, $\mathbf{U}_t, t = 1 \ldots, T$, computed in the first step of the algorithm, denote the importance of the marginal interaction between the selected covariates of each covariate block and the response block.

```
plot(model, vizu = 'weights', mar_left = 3, variance = 'RV')
```

Figure 2 shows these weights. One can observe that:

- the third component of `Block_1` is empty, i.e. no covariate has been selected on this component;
- the other components of *Block_1* or *Block_2* are all non null;
- the second component of *Block_2* is built with only one covariate and has a *RV coefficient* equal to 0.29 with the response matrix, which is important for a single covariate;
- all the other components are built with four or five covariates.

Note that only the study of the *scaled super-weights*, $\boldsymbol{U}_t^\star, t = 1, \ldots, T$, would allow to compare the importance of the different covariates from one block to another.

Visualization of the scaled super-weights

The *scaled super-weights*, $\boldsymbol{U}_t^\star, t = 1 \ldots, T$, computed in the second step of the algorithm, denote the importance of each covariate predicting the response block in the first column as shown in Figures 3, 4 and 5. These three figures are respectively obtained with the following code lines:

```
plot(model, vizu = 'weights', super = T, mar_left = 3, variance = 'RV',
     addY = T, pos_legend = NULL, mar_bottom = 3.5, comp = 1)
plot(model, vizu = 'weights', super = T, mar_left = 3, variance = 'RV',
     addY = T, pos_legend = NULL, mar_bottom = 3.5, comp = 2)
plot(model, vizu = 'weights', super = T, mar_left = 3, variance = 'RV',
     addY = T, pos_legend = NULL, mar_bottom = 3.5, comp = 3)
```

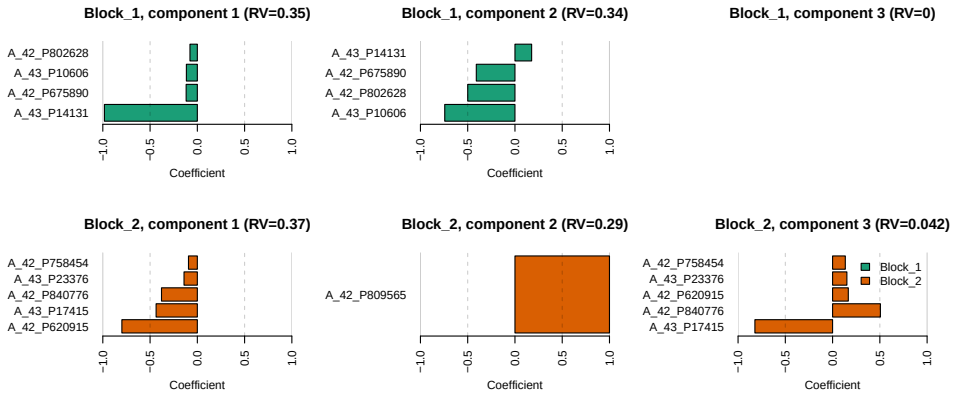


Figure 2: Visualization of the *weights*

One can note that

- the second column denotes the variance shared (according to the *RV coefficient* in that case) between the built component and each of the variables of the response block, the variables which are not selected are plotted in light blue while the selected variables are in dark blue;
- the first two components – Figures 3 and 4 – mainly explain the response variables *ALT.IU.L* and *AST.IU.L* and the third component – Figures 5 – explain the response variable *TBA.umol.L* preferentially, using the single covariate selected on component 2 of *Block_2* already described in Figure 2;
- these three response variables are the most described by the current model and the associated covariates are ranked by increasing importance in the first column of those Figures 3, 4 and 5, by component.

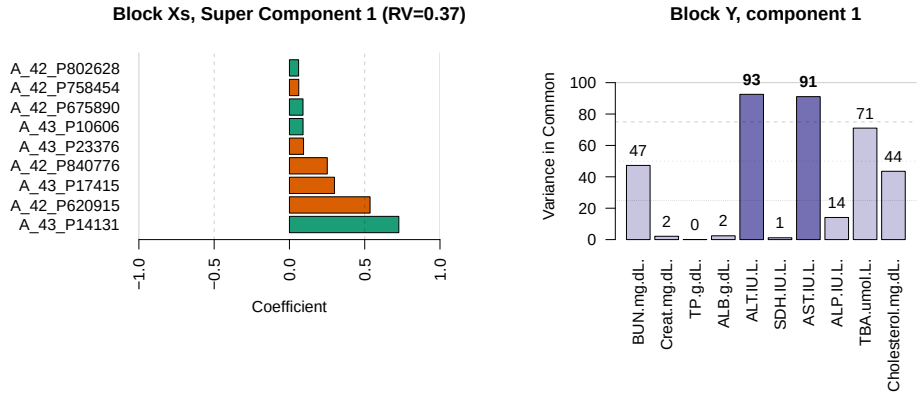


Figure 3: Visualization of the *scaled super-weights* of the 1^{st} component

Heatmaps to visualize individual behaviors along single super-component

The package also allows to draw heatmaps mixing covariates and response variables selected per components. For example, Figure 6 shows the results for the first and the third super components generated by the following code lines:

```
plot(model, vizu = 'heatmap', comp = 1, pos_legend = 'topright',
     margins_heatmap = c(1.5, 11))
plot(model, vizu = 'heatmap', comp = 1, pos_legend = 'topright',
     margins_heatmap = c(1.5, 11))
```

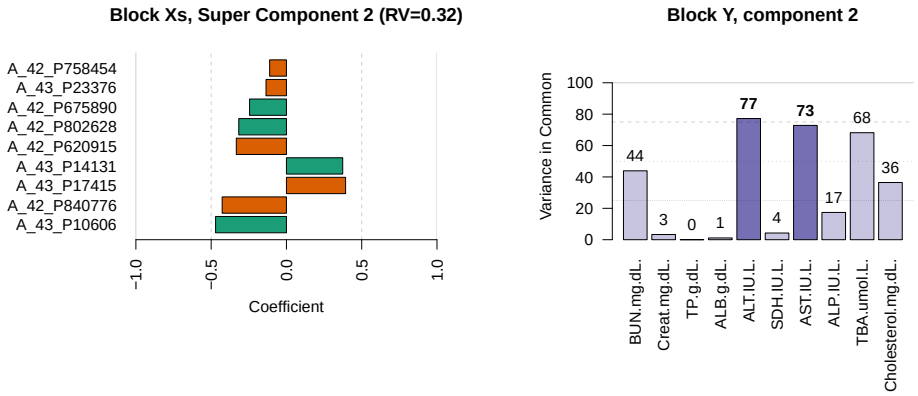


Figure 4: Visualization of the *scaled super-weights* of the 2^{nd} component

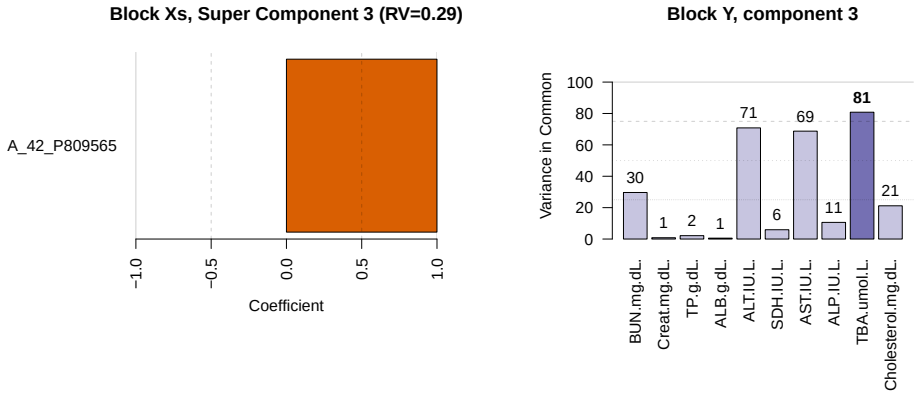


Figure 5: Visualization of the *scaled super-weights* of the 3^{rd} component

In these plots, dendrograms have been performed using hierarchical classification based on Ward distance (see Ward Jr, 1963, for example). In Figure 6, one can observe that

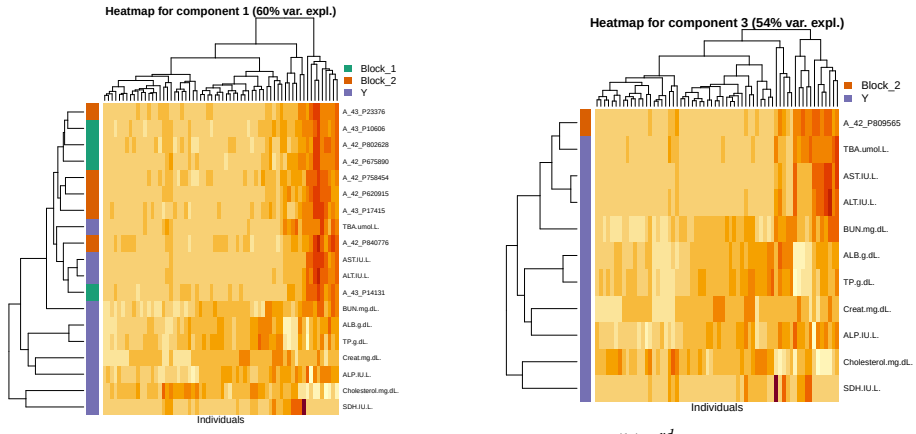
- each heatmap exhibits that individuals are strongly divided in two groups (dendrogram on the top of the graph);
- for each component, the two groups of individuals are unbalanced. For example, for the first component, the group of 9 individuals on the right of the heatmap are represented with dark colors, which is associated with high expression. On Figure 6b, 13 individuals seem to show high expression, at the far right of the figure, and correlated values.

Correlograms per super-component

Figure 7 represents the correlation matrices grouping covariates selected on components 1 and 3 respectively obtained with the code lines

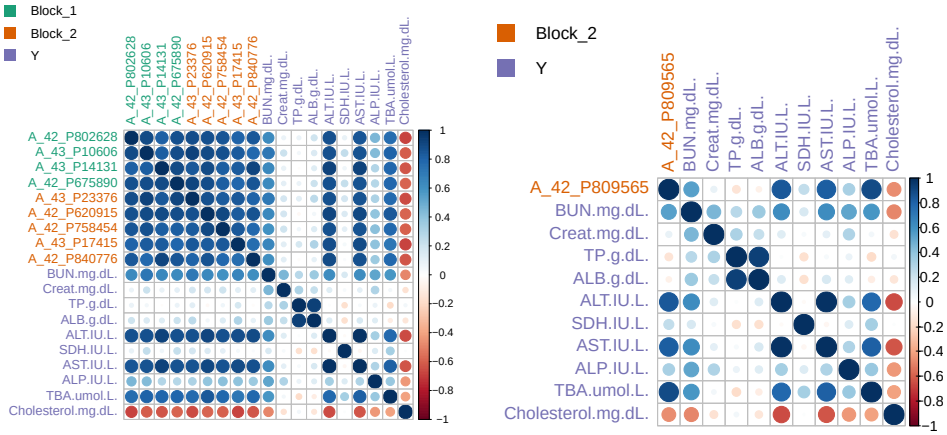
```
plot(model, vizu = 'correlogram', comp = 1)
plot(model, vizu = 'correlogram', comp = 3)
```

Contrary to heatmaps previously presented, the covariates are not ordered thanks to the Ward distance. This may be an interesting way to present the already discussed points. The value of the correlation coefficients can be plotted by setting the *values_corr* parameter to `TRUE` in the command showed just above. One can observe in Figure 7 that the variable *Cholesterol.mg.dl.* is negatively correlated with other response variables but also with all the covariates.



(a) 1^{st} component **(b)** 3^{rd} component

Figure 6: Heatmaps of the 1^{st} and 3^{rd} components mixing response and covariate selected per components and ordered with associated dendrogram. High expressions are represented with dark colors.



(a) 1^{st} Component **(b)** 3^{rd} Component

Figure 7: Correlograms of the 1^{st} and 3^{rd} components mixing response and covariates.

Predict values from a new sample

Based on an existing `mddsPLS` model, the `mddsPLS.predict()` S3 method allows to predict, from a new sample `X`,

- the response values associated with the new `X` sample,
- the imputed new `X` sample (if there are missing values).

Parameter `type` allows to choose between one of those two prediction types: 'y' or 'x' respectively allows the user to predict the response values and the imputed values of the concerned data set. The case 'both' allows to get both of the results in a single list object. In practice this can be done with the code lines:

```
y_predict <- predict(model, newdata = Xs, type = 'y')
x_predict <- predict(model, newdata = Xs, type = 'x')
both_predict <- predict(model, newdata = Xs, type = 'both')
```

Perform cross-validation

The aim of the "perf_mddsPLS" class is to determine optimal values for the tuning parameters, (R, λ) or (R, L_0), via cross-validation. The arguments of that class are detailed in the Table 3. For sake of clarity, arguments for which description are filled with "_" have been already detailed in Table 2.

Argument	Description
Xs	_
Y	_
lambda_min	A real in [0,1]. The minimum value of λ considered. Default is 0.
lambda_max	A real in [0,1]. The maximum value of λ considered. Default is NULL, interpreted as the largest correlation between X and Y.
n_lambda	A strictly positive integer corresponding to the number of λ values to consider in cross-validation, the λ values are equidistributed between lambda_min and lambda_max. Default is 1.
lambdas	A vector of λ values (in [0,1]) to consider in cross-validation. Default is NULL, when that parameter is not taken into account.
R	_
reg_imp_model	Logical. Whether or not to regularize the imputation models. Initialized to TRUE.
L0s	A vector of non null positive integers corresponding to the L_0 values to consider in cross-validation. Default is NULL and is then not taken into account.
kfolds	Character or integer. If equals to 'loo' then a leave-one-out cross-validation is started. No other characters are understood. Any strictly positive integer gives the number of folds to make in the cross-validation process.
mode	_
fold_fixed	Vector of length n. Each element corresponds to the number of the corresponding fold (see details below). If NULL then that argument is not considered. Default to NULL.
maxIter_imput	_
errMin_imput	_
NCORES	Integer. The number of cores to use (see details below). Default is 1.
NZV	_
plot_result	Logical. Whether or not to plot the result. Initialized to TRUE. The reg_error argument of the plot.perf_mddsPLS() function is left to its default value.
legend_label	Logical. Whether or not to add the legend names to the plot. Initialized to TRUE.

Table 3: Argumets of the class "perf_mddsPLS"

The NCORES allows to select the number of *cpu* that should be used by the software. This is bounded by the maximum number of *cpu* that are present on the machine and protected in that way.

The user must choose between λ and L_0 to make the regularization/selection. The L0s parameter is a vector of L_0 values to consider in the cross-validation process. Otherwise, for the λ parameters, the user can whether use

- n_lambda values of λ to consider in the cross-validation process. These values are taken equidistributed in the range corresponding to no covariate to remove on the left and no covariate to add on the right. lambda_min and/or lambda_max can be used to move those two previous automatic bounds.
- or lambdas, a vector of λ values to consider in the cross-validation process.

The number of folds in the cross-validation process is fixed thanks to the parameter kfolds, when that parameter is equal to the number of individuals or if it is equal to 'loo', then leave-one-out cross-validation is performed, otherwise k-fold cross-validation is performed.

Sometimes a more complex case can be encountered when dependant samples are considered for instance. In that specific context it might be interesting or even necessary to keep some samples together. The vector parameter fold_fixed of size n must then be used. If the considered data-set is build on nine samples and the first three must be kept together, as the next three and the last three ones, then fold_fixed should be equal to:

```
fold_fixed <- c(1, 1, 1, 2, 2, 2, 3, 3, 3)
```

Start cross-validation

The following example allows to select, for $R = 2$ components (value fixed by the user), the best value for L_0 using leave-one-out cross-validation. Only six different values of L_0 have been tested, deploying the calculations on three cores.

```
cv <- perf_mddsPLS(Xs = Xs, Y = Y, L0s = c(1, 3, 5, 10, 25, 50),
                   R = 2, mode = 'reg', kfolds = 'loo', NCORES = 3,
                   plot_result = FALSE)
```

The summary method

The part of the output denoted as `Cross-Validation results`allows to access to the proportion of models which have converged against the total number of models built (third column) and also to the main time statistics in terms of computation time (fourth column) for given R and L_0 parameters (first and second columns respectively).

```
summary(cv, plot_res_cv = FALSE)
=======================================================
   Cross-Validation ddsPLS object description
=======================================================

Number of blocks: 2
Number of individuals: 64
Number of variables in Y part: 10
Model built in mode regression


  Missing value information
---------------------------------

                Block_1 Block_2  Total
Number of variables        1910.00 1206.00 3116.00
Number of missing samples      5.00  5.00  10.00
Proportion of missing samples (%)  7.81  7.81  7.81


   Cross-Validation results
---------------------------------

 R L0 Nb.of.convergences.VS.nb.of.fold Mean.AND.sd.time.of.computation
1 2 1         64/64      0.19(0.025)
2 2 3         64/64      0.21(0.023)
3 2 5         64/64      0.22(0.021)
4 2 10         64/64      0.21(0.02)
5 2 25         64/64      0.22(0.022)
6 2 50         64/64      0.22(0.019)


         Thank s for using me
-------------------------------------------------------
                   Hadrien Lorenzo
           hadrien.lorenzo.2015@gmail.com
=======================================================
```

In that example, all the models have converged and each model is build in around $0.2s$.

The plot method

The plot method allows to plot, for a given value for R, the error curves in the regression case (the lower the better) and the accuracy curves in the classification case (so the higher the better). In the case of multivariate regression, it is important to perform the same normalization on response variables as

to get comparable *MSEP* (*Mean Squared Error in Prediction*) curves. As a reminder, *MSEP* is computed on the estimation $\hat{\mathbf{Y}}$ of $\mathbf{Y}$ such as

$$MSEP\left(\mathbf{Y},\hat{\mathbf{Y}}\right) = \text{diag}\left(\left(\mathbf{Y}-\hat{\mathbf{Y}}\right)^T\left(\mathbf{Y}-\hat{\mathbf{Y}}\right)\right)/n,$$

where ". $\rightarrow$ diag(.)" gives the diagonal of the square matrix argument. And so, $MSEP(\mathbf{Y},\hat{\mathbf{Y}})$ is filled with q elements corresponding to each of the q *MSEP*s of each predicted variable of $\mathbf{Y}$.

Remark. It is also possible to check the *MPE* (*Mean Prediction Error*), the sum of the absolute errors, using the argument `reg_error` and putting it to '`MPE`'. The default value is '`MSEP`' and allows to appreciate the *MSEP* error.

As mentionned before, each of the response variables has been standardized (zero mean and unit variance) such as all error curves would be comparable. Hence, the error would be comparable to 1 above which the model does not perform better than mean prediction, at the limit. The following line code

```
plot(cv, legend_names = colnames(Y), pos_legend = 'right', plot_mean = T)
```

provides Figure 8 which shows the corresponding results. The three well predicted response variables are the ones already discussed in the model building part. The other variables have a *MSEP* close to 1 which means that the different leave-one-out model would not perform better than mean prediction. Two vertical dotted lines represent the value for which the mean of the *MSEP* curves is minimum and the minimum value of the q *MSEP* curves. Both of those minima can be good candidates but it belongs to the analyst to choose the most suitable parameter, taking into account that also the number of components must be chosen (and this procedure must be repeated for various values of R). In the next paragraph, a way to determine simultaneously, thanks to cross-validation,the parameters L_0 and R are provided.

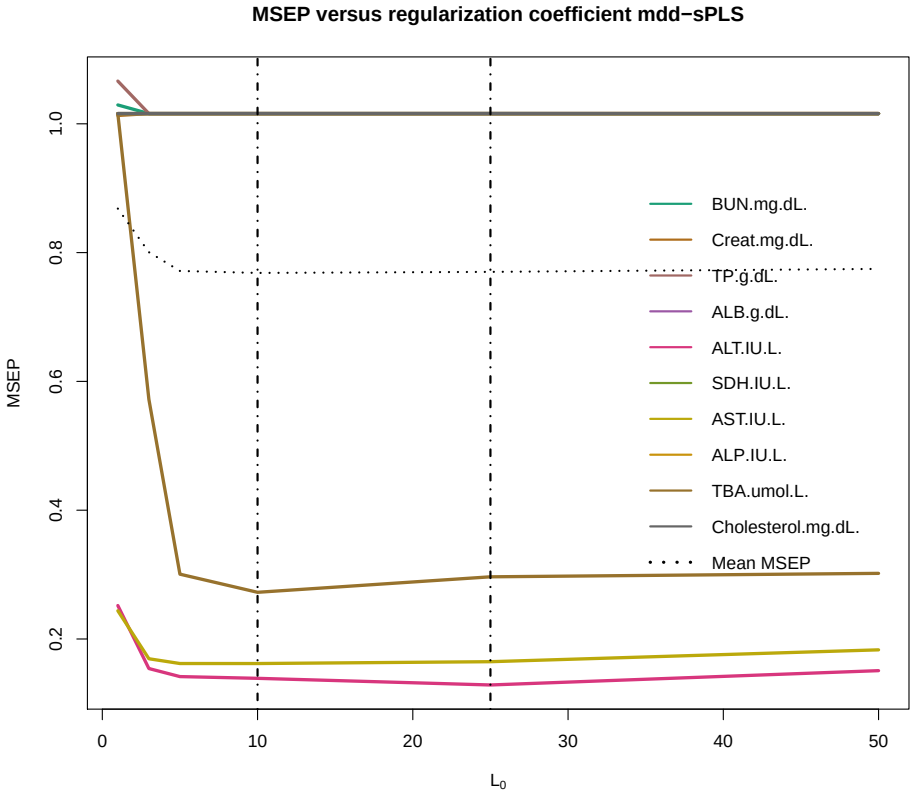


Figure 8: Cross-validation leave-one-out curve errors for the ten to be predicted variables

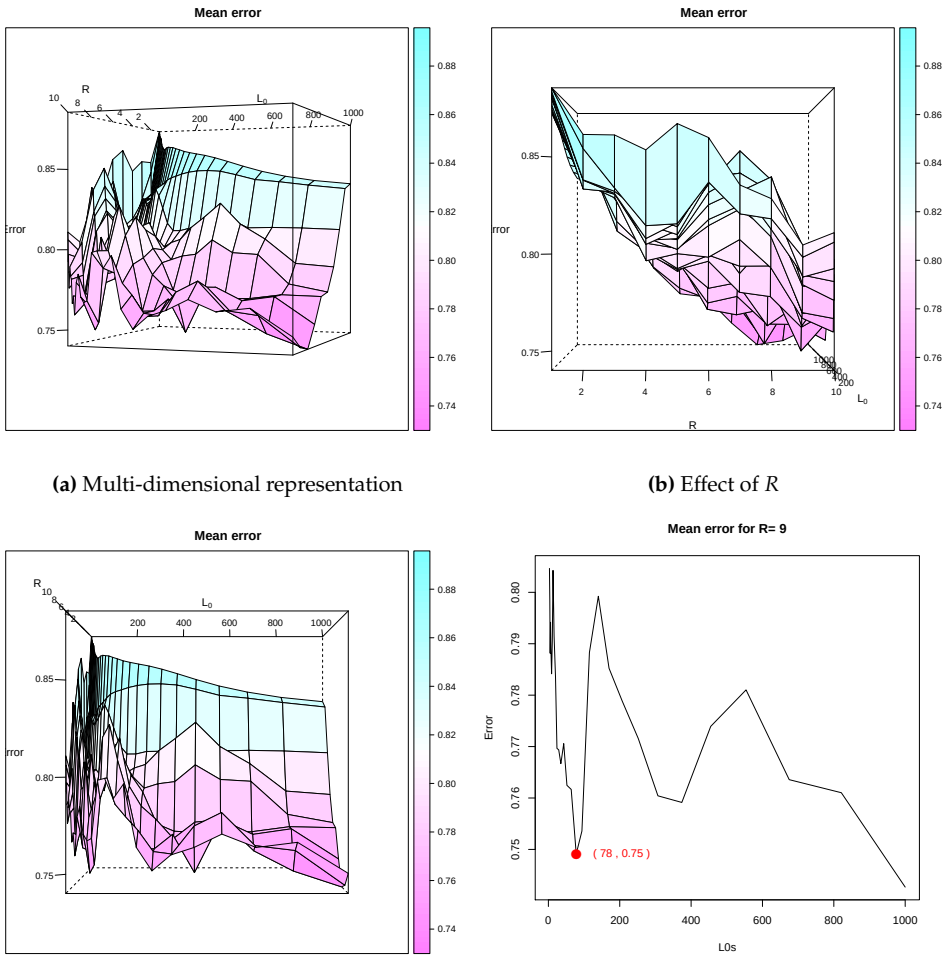
Cross-validation to fix the all parameters

Both of the tuning parameters (R, L_0) or (R, λ) of the method can be fixed thanks to cross-validation. The following code shows how to perform this analysis for (R, L_0) using the toy example. Since this is multivariate regression, it has been chosen to consider the mean of the 10 *MSEP* of leave-one-out cross-validation error as a common criterion. No function has yet been implemented to visualize that error but the following script gives an idea of a feasible solution.

```
L0s <- unique(round(seq(1, 1000, length.out = 30)))
Rs <- 1:10
mat_means <- matrix(NA, length(L0s), length(Rs))
CVs <- list()
for(r in Rs){
  CVs[[r]] <- perf_mddsPLS(Xs = Xs, Y = Y, L0s = L0s,
                           R = r, mode = 'reg', kfolds = 'loo',
                           NCORES = 7, plot_result = FALSE)
  mat_means[, r] <- rowMeans(CVs[[r]]$RMSEP[, -c(1:2)])
}
data <- expand.grid(Rs, L0s)
data <- cbind(data, as.vector(t(mat_means)))
data.f <- data.frame(data)
names(data.f) <- c('L0', 'R', 'Error')
lattice::wireframe(Error ~ L0*R, data = data.f,
                   ylab = expression(L[0]), xlab = 'R',
                   main = 'Mean error',
                   drape = TRUE,
                   colorkey = TRUE,
                   screen = list(x = -90, y = -110),
                   scales = list( arrows = FALSE)
)
lattice::wireframe(Error ~ L0*R, data = data.f,
                   ylab = expression(L[0]), xlab = 'R',
                   main = 'Mean error',
                   drape = TRUE,
                   colorkey = TRUE,
                   screen = list(x = -90),
                   scales = list( arrows = FALSE)
)
lattice::wireframe(Error ~ L0*R, data = data.f,
                   ylab = expression(L[0]), xlab = 'R',
                   main = 'Mean error',
                   drape = TRUE,
                   colorkey = TRUE,
                   screen = list(y = -90, z = -90),
                   scales = list( arrows = FALSE)
)
r <- 9
Err <- rowMeans(CVs[[r]]$RMSEP[, -c(1:2)])
pos_min <- which.min(Err[which(L0s<200)])
L0_min <- L0s[pos_min]
Err_min <- Err[pos_min]
plot(L0s, Err, ylab = 'Error', main = paste('Mean error for R = ', r),
     type = 'l')
points(L0_min, Err_min, pch = 16, col = 'red', cex = 2)
text(L0_min, Err_min,
     labels = paste('(', round(L0_min, 2), ', ',
round(Err_min, 2), ')'), col = 'red', adj = -0.3)
```

Figures 9a, 9b and 9c show the results through 3-dimensional plots of the mean errors along the 10 response variables. It seems that $R = 9$ and $L_0 \approx 78$ (red point in Figure 9d) allow to get an interesting point in terms of prediction error and sparsity, discussions should be conducted with domain specialists in front of those curves to select the best model.

Since the tuning parameters are now determined, the methods detailed before in the manuscript can be used to build the model and should help the user to learn more about the data.



(a) Multi-dimensional representation

(b) Effect of R

(c) Effect of L_0

(d) Mean error for $R = 9$, red point corresponds to the chosen minimum

Figure 9: Plot of the means of the 10 response variables MSEP's for cross-validation leave-one-out analysis.

How to implement the classification case?

Until now, the paper mainly focuses on the regression case. Classification case can also be taken into account, following Barker and Rayens (2003). It is quite transparent for the user, changing the argument `mode` in functions `mddsPLS()` and `perf_mddsPLS()` to one of

- '`lda`', for *linear discriminant analysis*,
- '`logit`', for *logistic regression* (it only works in two-classes classification problems).

Recall also that the cross-validation is monitored through the *good classification rate* (and not the *error rate* as in regression case) which implies to look for maxima on the curves (and not minima as in regression case).

The hidden class of the package

The user cannot build directly an object of this class because it is protected according to different aspects corresponding to whether there are missing values and whether L_0 or λ is used as a parameter. This class is then used internally in the *Koh-Lanta* process. Any object of this class would be an attribute of the "`mddsPLS`" class. The attributes of that class are given in Table 4.

Attribute	Symbol	Description
`u`	$(\mathbf{U}_t)_{t\in[\![1,T]\!]}$	A list of length `T`. Each element is a $p_t \times R$ matrix.
`u_t_super`	$(\mathbf{U}_t^\star)_{t\in[\![1,T]\!]}$	A list of length `T`. Each element is a $p_t \times R$ matrix.
`v`	$\mathbf{V}^\star$	A $q \times R$ matrix : the weights for the `Y` part.
`ts`		A list of length R. Each element is a $n \times T$ matrix : the scores per component.
(t, s)	$(\mathbf{T}^\star, \mathbf{S}^\star)$	Two $n \times R$ matrices, super-scores of the `X` and `Y` parts.
(t_ort,s_ort)		Two $n \times R$ matrices, final scores of the `X` and `Y` part. They correspond to `PLS` scores of `(t,s)` scores and so $\texttt{t\_ort}^T$ `s_ort` is diagonal, `t_ort`, respectively `s_ort`, carries the same information as `t`, respectively `s`.
`B`	$\mathcal{B}$	A list of length `T`, for the number of blocks. Each element is a $p_t \times q$ matrix.
(mu_x_s,sd_x_s)		Two lists of length `T`, for the number of blocks. Each element is a p_t vector : the mean and standard deviation covariates per block.
(mu_y,sd_y)		Two vectors of length q : the mean and the standard deviation variables for `Y` part.
`R`	R	Given as an input.
`q`	q	A non negative integer.
`Ms`	$(\mathbf{M}_t)_{t\in[\![1,T]\!]}$	A list of length `T`, for the number of blocks.
`lambda`	λ	The regularization coefficient.

Table 4: Attributes of the class `mddsPLS_core`

Let be supposed that a model `Model` from the "`mddsPLS`" class has been created, then one can get to the corresponding object of the "`MddsPLS_core`" class such as

```
Model <- mddsPLS(Xs = Xs, Y = Y, L0 = 10, R = 3, mode = 'reg')
Model_MddsPLS_core_object <- Model$mod
Out <- lapply(Model_MddsPLS_core_object, function(x){
  out <- dim(x)
  if(is.null(out)){
    out <- length(x)
  }else{
    out <- paste(out, sep = '', collapse = 'x')
  }
  out
 })
```

The printed command shows the dimension(s) of the attributes of the considered object of the "`MddsPLS_core`" class.

```
print(do.call(rbind, Out))
     [,1]
u     "2"
u_t_super "2"
V_super  "10x2"
ts    "2"
beta_comb "4x2"
T_super  "64x2"
S_super  "64x2"
t_ort   "64x2"
s_ort   "64x2"
B     "2"
mu_x_s  "2"
sd_x_s  "2"
mu_y   "10"
sd_y   "10"
R     "1"
q     "1"
Ms    "2"
lambda  "1"
```

Conclusion

The package **ddsPLS** allows to solve linear multivariate regression or classification problems in the context of heterogeneous multiblock data sets for high dimensional data.

The **ddsPLS** package is simple to use and provides dimension reduction and variable selection through a computational method based on SVD of correlation matrix soft-thresholding. Computation time has been reduced thanks to C++ functions using the **Rcpp** package. A cross-validation function using parallel computations has also been included in that package thanks to the **doParallel**, **parallel** and **foreach** packages. Different S3 methods allow to visualize model specificities (through barplots, heatmaps or correlogram) and cross-validation performances.

Classification case analyses are performed thanks to `lda()` or `logit()` functions (from the **MASS** package), those functions are based on the `ddsPLS` objects built on the regression multivariate models predicting the dummy matrix generated from the class response.

Acknowledgments

Hadrien Lorenzo is supported by a 2016 Inria-Inserm thesis grant *Médecine Numérique* (for *Digital Medicine*).

Included data sets

Two data sets are included in the package and correspond to

- a regression problem, through the `liverToxicity` data set. This data set contains the expression measure of 3116 genes and 10 clinical measurements for $n = 64$ subjects (rats) that were exposed to non-toxic, moderately toxic or severely toxic doses of acetaminophen in a controlled experiment (see Bushel et al., 2007).
- a classification problem. The data set `penicilliumYES` has 36 rows and 3754 columns. The covariates are 1st order statistics from multi-spectral images of three species of Penicillium fungi: Melanoconidium, Polonicum, and Venetum (see Clemmensen et al., 2011).

Bibliography

E. Acar, T. G. Kolda, and D. M. Dunlavy. All-at-once Optimization for Coupled Matrix and Tensor Factorizations. *arXiv preprint arXiv:1105.3422*, 2011. [p2]

E. Acar, M. A. Rasmussen, F. Savorani, T. Næs, and R. Bro. Understanding data fusion within the framework of coupled matrix and tensor factorizations. *Chemometrics and Intelligent Laboratory Systems*, 129:53–63, 2013. URL https://doi.org/10.1016/j.chemolab.2013.06.006. [p2]

S. Bakin. *Adaptive regression and model selection in data mining problems*. PhD thesis, School of Mathematical Sciences, Australian National University, 1999. URL https://doi.org/10.25911/5d78db4c25dbb. [p1]

M. Barker and W. Rayens. Partial least squares for discrimination. *Journal of Chemometrics: A Journal of the Chemometrics Society*, 17(3):166–173, 2003. URL https://doi.org/10.1002/cem.785. [p17]

P. R. Bushel, R. D. Wolfinger, and G. Gibson. Simultaneous clustering of gene expression data with clinical chemistry and pathological evaluations reveals phenotypic prototypes. *BMC Systems Biology*, 1(1):15, 2007. URL https://doi.org/10.1186/1752-0509-1-15. [p4, 18]

L. Clemmensen, T. Hastie, D. Witten, and B. Ersbøll. Sparse discriminant analysis. *Technometrics*, 53(4): 406–413, 2011. URL https://doi.org/10.1198/TECH.2011.08118. [p18]

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977. [p1]

M.-P. Ellies-Oury, H. Lorenzo, C. Denoyelle, J. Saracco, and B. Picard. An original methodology for the selection of biomarkers of tenderness in five different muscles. *Foods*, 8(6):206, 2019. URL https://doi.org/10.3390/foods8060206. [p2]

T. Hastie and R. Mazumder. softImpute: Matrix Completion via Iterative Soft-Thresholded SVD. *R package version*, 1, 2015. [p1]

J. Josse and F. Husson. Handling missing values in exploratory multivariate data analysis methods. *Journal de la Société Française de Statistique*, 153(2):79–99, 2012. [p2]

J. Josse and F. Husson. missMDA: A Package for Handling Missing Values in Multivariate Data Analysis. *Journal of Statistical Software*, 70(1):1–31, 2016. URL https://doi.org/10.18637/jss.v070.i01. [p2]

J. Josse, F. Husson, and J. Pagès. Gestion des données manquantes en analyse en composantes principales. *Journal de la Société Française de Statistique*, 150(2):28–51, 2009. [p1]

J. F. Kenney and E. Keeping. Linear regression and correlation. *Mathematics of statistics*, 1:252–285, 1962. [p4]

K.-A. Lê Cao, D. Rossouw, C. Robert-Granié, and P. Besse. A Sparse PLS for Variable Selection when Integrating Omics data. *Statistical applications in genetics and molecular biology*, 7(1), 2008. URL https://doi.org/10.2202/1544-6115.1390. [p1]

B. Liquet, P. L. de Micheaux, B. P. Hejblum, and R. Thiébaut. Group and sparse group partial least square approaches applied in genomics context. *Bioinformatics*, 32(1):35–42, 2015. URL https://doi.org/10.1093/bioinformatics/btv535. [p1]

H. Lorenzo, R. Misbah, J. Odeber, P.-E. Morange, J. Saracco, D.-A. Trégouët, and R. Thiébaut. High-dimensional multi-block analysis of factors associated with thrombin generation potential. In *2019 IEEE 32nd International Symposium on Computer-Based Medical Systems (CBMS)*, pages 453–458. IEEE, 2019a. URL https://doi.org/10.1109/CBMS.2019.00094. [p2]

H. Lorenzo, J. Saracco, and R. Thiébaut. Supervised learning for multi-block incomplete data. *arXiv preprint arXiv:1901.04380*, 2019b. [p2]

R. Mazumder, T. Hastie, and R. Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. *Journal of machine learning research*, 11(Aug):2287–2322, 2010. [p1]

M. Planitz. 3. Inconsistent systems of linear equations. *The Mathematical Gazette*, 63(425):181–185, 1979. URL https://doi.org/10.2307/3617890. [p4]

P. Robert and Y. Escoufier. A Unifying Tool for Linear Multivariate Statistical Methods: The RV-Coefficient. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 25(3):257–265, 1976. URL https://doi.org/10.2307/2347233. [p6]

D. B. Rubin. Inference and missing data. *Biometrika*, 63(3):581–592, 1976. URL https://doi.org/10.2307/2335739. [p1]

J. L. Schafer and J. W. Graham. Missing data: our view of the state of the art. *Psychological methods*, 7 (2):147, 2002. URL http://doi.org/10.1037/1082-989X.7.2.147. [p1]

N. Simon, J. Friedman, T. Hastie, and R. Tibshirani. A Sparse-Group Lasso. *Journal of Computational and Graphical Statistics*, 22(2):231–245, 2013. URL https://doi.org/10.1080/10618600.2012.681250. [p1]

D. J. Stekhoven and P. Bühlmann. MissForest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1):112–118, 2011. URL https://doi.org/10.1093/bioinformatics/btr597. [p1]

A. Tenenhaus and M. Tenenhaus. Regularized generalized canonical correlation analysis. *Psychometrika*, 76(2):257, 2011. URL https://doi.org/10.1007/s11336-011-9206-8. [p1]

A. Tenenhaus, C. Philippe, V. Guillemot, K.-A. Le Cao, J. Grill, and V. Frouin. Variable selection for generalized canonical correlation analysis. *Biostatistics*, 15(3):569–583, 2014. URL https://doi.org/10.1093/biostatistics/kxu001. [p1]

R. Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996. URL https://doi.org/10.1111/j.2517-6161.1996.tb02080.x. [p1]

L. Wangen and B. Kowalski. A multiblock partial least squares algorithm for investigating complex chemical systems. *Journal of chemometrics*, 3(1):3–20, 1989. URL https://doi.org/10.1002/cem.1180030104. [p1]

J. H. Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244, 1963. URL https://doi.org/10.1080/01621459.1963.10500845. [p10]

R. A. Willoughby. Solutions of ill-posed problems (an tikhonov and vy arsenin). *SIAM Review*, 21(2): 266, 1979. URL https://doi.org/10.1137/1021044. [p1]

S. Wold, M. Sjöström, and L. Eriksson. PLS-regression: a basic tool of chemometrics. *Chemometrics and intelligent laboratory systems*, 58(2):109–130, 2001. URL https://doi.org/10.1016/S0169-7439(01)00155-1. [p1]

Hadrien Lorenzo
ISTM INRIA BSO
Inserm-U1219 BPH
ISPED, Bordeaux University
146 rue Léo Saignat
33000, Bordeaux, France
hadrien.lorenzo@u-bordeaux.fr

Jérême Saracco
CQFD INRIA BSO
IMB, UMR 5251 CNRS
ENSC - Bordeaux INP
109 avenue Roul
33400, Talence, France
jerome.saracco@ensc.fr

Rodolphe Thiébaut
SISTM INRIA BSO
INSERM-U1219 BPH
ISPED, Bordeaux University
146 rue Léo Saignat
33000, Bordeaux, France
rodolphe.thiebaut@u-bordeaux.fr

CHAPITRE 4

AUTRES APPLICATIONS SUR DONNÉES RÉELLES

Sommaire

La méthode ddsPLS, détaillée dans les sections précédentes, a été appliquée à deux jeux de données qui ont chacun abouti à une publication.

Le premier jeu de données concerne l'analyse d'un ensemble de variables (protéines, marques de méthylation de l'ADN, microARN et phénotypes biologiques) mesurée dans un échantillon de patients atteints de maladie thromboembolique veineuse dont les manifestations les plus connues sont la phlébite et l'embolie pulmonaire.

Le second jeu de données détaille les abondances de 20 biomarqueurs musculaires protéiques au sein de 5 muscles de la carcasse, pour 10 individus différents ainsi que les niveaux de tendreté associés pour chacun de ces 5 x 10 muscles. Voir section 4.2.

Au delà des résultats scientifiques obtenus, ces applications ont permis l'amélioration de la méthode, de son implémentation ainsi que de proposer différents types de visualisation des résultats afin d'en faciliter l'interprétation.

4.1 Thrombose veineuse et prédiction du niveau de thrombine

La thrombose veineuse est la troisième maladie cardiovasculaire la plus fréquente chez l'homme après l'infarctus du myocarde et les accidents vasculaires cérébraux et est associée à une forte mortalité sous sa forme embolie pulmonaire. La thrombose veineuse se caractérise par la formation d'un caillot de sang (également appelé thrombus) dans une veine, elle est alors vulgairement appelée phlébite, caillot qui lorsqu'il se décroche peut migrer vers l'artère pulmonaire et la boucher, provoquant alors une embolie.

Les données à disposition pour cette application concernaient un échantillon 705 patients atteints de thrombose veineuse pour lesquels plusieurs variables avaient été mesurées mais pas toutes chez tous les individus. 234 protéines avaient été mesurées pour la totalité de ces patients, 3174 marques de méthylation de l'ADN sanguin avaient été mesurées et conservées chez 350 patients, et 6 variables cliniques chez ces patients. En revanche, tous ces patients avaient été phénotypés pour 3 variables biologiques résumant la capacité d'un individu à produire de la thrombine, molécule clé de la cascade de la coagulation. Ces trois variables sont le potentiel de génération de thrombine (*Endogeneous Thrombin Potential, ETP*, en anglais), le temps de latence (*Lagtime*) et la hauteur du pic (*peak*), ces trois variables étant dérivées d'une courbe de cinétique. Un profil de cette cinétique est visible en figure 1 de l'article qui suit.

L'objectif de cette application était de déterminer à partir des 3414 déterminants moléculaires disponibles une signature associée aux 3 variables de génération de thrombine, permettant alors l'identification de nouvelles molécules participant à la cascade de coagulation.

Par rapport à la méthode, il nous est apparu nécessaire, au vu de la proportion de données manquantes, de modifier les estimations des données manquantes qui sont faites à l'initialisation de l'algorithme. En effet, en initialisant à la moyenne comme réalisé précédemment on déforme de façon importante les structures de corrélations des données. La solution choisie est de diviser chaque bloc en une partie servant à l'entraînement, la partie sans donnée manquantes, et l'autre qui doit être estimée puisque totalement manquante, avec la partie $\mathbf{Y}$ comme variables à prédire. Les coefficients de régularisation sont pris égaux à ceux renseignés par l'utilisateur. De plus, la paramétrisation a été légèrement modifiée de façon à rendre l'algorithme plus interprétable par les utilisateurs. Ce nouveau paramètre remplace le paramètre λ qui représentait le seuil de corrélation en dessous duquel un lien $< \mathbf{X}_i, \mathbf{Y}_j >$ n'est plus à considérer en introduisant le nouveau paramètre L_0 fixant le nombre maximum de variables de la partie des covariables que l'utilisateur désire sélectionner. Plus précisément, il est directement associé à la plus petite valeur de λ permettant d'annuler exactement L_0 colonnes de la matrice

$$S_\lambda\left(\frac{\mathbf{Y}^T\left[\mathbf{X}_1,\cdots,\mathbf{X}_T\right]}{n-1}\right).$$

Il a été aussi remarqué à la suite de ce travail que la méthode peut avoir un comportement contre-intuitif en validation croisée. En effet, nous avons observé que la première composante qui est identifiée pour expliquer le plus de la variabilité des données utilisées dans l'apprentissage des modèles pour *ETP* et *Peak* alors que ces phénotypes sont moins bien prédits dans la phase de test que ne l'est le *Lagtime* qui est associé à la deuxième composante, moins bien caractérisée dans la partie apprentissage. Ceci est sûrement associé à la forte proportion de données manquantes dans les données de méthylation qui n'assure pas un bon comportement lorsque les variables dans le modèle ne sont pas toutes associées à ces données manquantes. Ceci semble être un cas de suraprentissage de la méthode et gagnerait à être étudié plus spécifiquement au travers de

futurs travaux.

Ces travaux ont donné à une présentation lors d'une conférence internationale intitulée *32nd IEEE CBMS International Symposium on Computer-Based Medical Systems* à Cordoue (Espagne) du 5 au 7 juin 2019. Ceci en plus de la publication de l'article qui suit.

2019 IEEE 32nd International Symposium on Computer-Based Medical Systems (CBMS)

High-dimensional multi-block analysis of factors associated with thrombin generation potential

Hadrien Lorenzo*, Misbah Razzaq†, Jacob Odeberg‡, Pierre-Emmanuel Morange§, Jérôme Saracco¶, David-Alexandre Trégouët† and Rodolphe Thiébaut*

* Univ. Bordeaux, Inria BSO, Inserm U1219 Bordeaux Population Health Research Center, SISTM team, France,
Emails: hadrien.lorenzo@u-bordeaux.fr, rodolphe.thiebaut@u-bordeaux.fr
† Univ. Bordeaux, Inserm U1219 Bordeaux Population Health Research Center, VINTAGE team, France,
Emails: misbah.razzaq@inserm.fr, david-alexandre.tregouet@inserm.fr
‡Department of Proteomics, School of Biotechnology, Science for Life Laboratory,
KTH Royal Institute of Technology, Stockholm, Sweden,
Email: jacob.odeberg@scilifelab.se
§Laboratory of Haematology, La Timone Hospital, Marseille, France,
Email: pierre.morange@ap-hm.fr
¶ Inria BSO, CQFD team, CNRS UMR5251 Institut Mathmatique de Bordeaux, ENSC Bordeaux INP, Talence, France,
Email: jerome.saracco@inria.fr

***Abstract*—The identification of novel biological factors associated with thrombin generation, a key biomarker of the coagulation process, remains a relevant strategy to disentangle pathophysiological mechanisms underlying the risk of venous thrombosis (VT). As part of the MARseille THrombosis Association Study (MARTHA), we measured whole blood DNA methylation levels, plasma levels of 300 proteins, 3 thrombin generation biomarkers (endogeneous thrombin potential, peak and lagtime), clinical and genetic data in 700 patients with VT. The application of a novel high-dimensional multi-levels statistical methodology we recently developed, the data driven sparse Partial Least Square method (`ddsPLS`), on the MARTHA datasets enabled us 1/ to confirm the role of a known mutation of the variability of endogenous thrombin potential and peak, 2/ to identify a new signature of 7 proteins strongly associated with lagtime.**

***Index Terms*—Multi-Omics, High Dimensional Data, Missing Data, SVD, Partial Least Square, Variable Selection, Multi-Block Analysis, Machine Learning, Thrombine Generation**

I. Introduction

Venous thrombosis (VT) is a complex disease characterized by the formation of a blood clot in a deep vein that can later break free and travel to the lung to provoke pulmonary embolism.

In this process, thrombin is a key molecule and individuals that have a strong capacity to produce thrombin are at higher risk for VT. The thrombin generation potential (TGP) of an individual can be measured by thrombin generation assays that capture the complete dynamics of the coagulation process following clot formation. TGP is generally summarized by the use of three associated parameters, the $LagTime$, time after the lag-phase that follows trigger of coagulation until the initiation of thrombin generation, the $Peak$, the maximum amount of thrombin that can be produced, and the ETP that corresponds to the area under the thrombogram curve (i.e. amount of thrombin generated), see Figure 1. While ETP and $Peak$ are highly correlated, the $LagTime$ variable generally shows moderate correlation with the two other markers [1].

We had previously demonstrated that the $F2$ G20210A mutation was the main genetic factor contributing to ETP and $Peak$ plasma variability without impacting $LagTime$, see [1].

In addition, using a methylation-wide association strategy, we reported that DNA methylation marks in whole blood did not strongly associate with TGP biomarkers [2]. Motivated by the search for novel molecular determinants of TGP biomarkers, plasma samples of MARTHA participants were profiled for 200 proteins using a recent high-throughput technology [3].

In the current work, these proteomics data were studied for association with ETP, $Peak$ and $LagTime$. Using a recently developed high-dimensional multi-$omics$ algorithm, we jointly studied the association between ETP, $Peak$ and $LagTime$ variables through a multivariate analysis with known TGP determinants, the proteomics data and DNA methylation that was available only in a subsample of participants. The proposed methodology uses the data driven sparse Partial Least Square method (`ddsPLS`) to predict each of the three TGP biomarkers in a multivariate fashion taking into account possible missing values in the covariates using information in the response matrix while keeping sparsity in the context of high-dimensional data where the number of predictors p is in the same order or superior to the number of individuals n. `ddsPLS` has been implemented in **R** and **Python**, and is available on **CRAN** [1] and on **PyPi** [2], respectively.

The present work is divided in four parts. The first part describes the data set. The second part describes the method

[1]See https://cran.r-project.org/package=ddsPLS.
[2]See https://pypi.org/project/py_ddspls/.

2372-9198/19/$31.00 ©2019 IEEE
DOI 10.1109/CBMS.2019.00094

and the third one describes the results applying the method to the data set. The fourth part concludes and gives perspectives.

II. Notation

Lets denote the matrices in bold upper cases and vectors in bold lower cases. In the case of greek letter objects, matrices are represented with underlined greek letters and no difference in the notation in those case between matrices and column vector matrices. $\mathbf{X}$ denoted matrices account for predictor associated matrices and $\mathbf{Y}$ for response matrices. Indices are used when necessary. Without further indication, all matrices are standardized, zero mean and unit variance for each column. n is used to represent the number of rows of a matrix, and also the number of individuals. The number of columns of a $\mathbf{X}$ matrix, resp. $\mathbf{Y}$ matrix, is represented by the letter p, resp. by the letter q. The r^{th} column vector of a given matrix $\mathbf{U}$ is denoted $\mathbf{u}^{(r)}$. Matrix sets are denoted as $\mathbb{R}^{n\times p}$ and correspond to n rows and p columns matrices. $\mathbb{I}_n$ denotes identity matrix with n columns. $||.||_2$ denotes the $\mathcal{L}_2$-norm of a given vector and $(\mathbf{x},\mathbf{y}) \in \mathbb{R}^{n\times 1^2} \rightarrow \mathbf{x}^T\mathbf{y}$ the associated $\mathcal{L}_2$-cross-product where the transpose operator is symbolized by ".^{T}". Let the proportion of variance of a matrix $\mathbf{Y}$ explained by a matrix $\mathbf{X}$ be expressed as

$$\frac{||\mathbf{XB}(\mathbf{Y},\mathbf{X})||_F^2}{||\mathbf{Y}||_F^2} \times 100,$$

where $||.||_F$ is the Frobenius norm and $\mathbf{B}(\mathbf{Y},\mathbf{X})$ is the multivariate coefficient regression matrix which solves the Ordinary Least Square (OLS) problem $\max_{\mathbf{B}} ||\mathbf{Y}-\mathbf{XB}||_F^2$.
Let be denoted by R the number of dimensions built in the following.

III. Materials

This work was based on the MARseille THrombosis Association Study (MARTHA) cohort including patients with VT recruited at the Thrombophilia center of La Timone hospital (Marseille, France) between January 1994 and October 2005. This study has been extensively described in previous works [1], [4].

A. Biological measurements

Thrombin generation potential (TGP) was measured in platelet-poor plasma (PPP) of 705 individuals using the CAT method as described in [1]. Plasma levels of these individuals were profiled for 384 antibodies (referred thereafter to as HPAs as they were selected from the Human Protein Atlas) targeting 234 proteins using high-affinity bead array technology. These proteins have been selected because of their potential role in the coagulation and fibrinolysis cascades or because they have been reported to be associated with cardiovascular traits. From this sample of participants, 350 have been epityped for whole blood DNA methylation (referred to as mDNA) using the Illumina H450K array as described in [4]. For the current application, we only used the 3,174 most variable and relevant

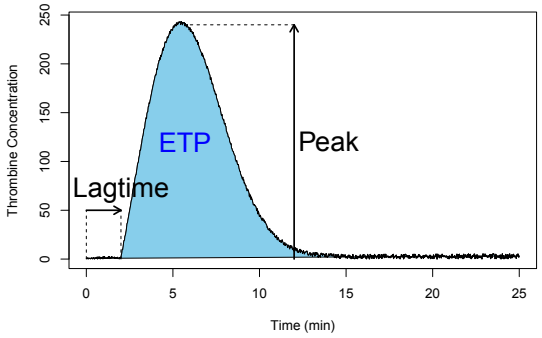


Fig. 1. Thrombin generation test curve and it main features.

CpG sites [3] among the ≈380,000 measured CpG sites.
All patients were genotyped for the F2 G20210A mutation and measured for Protein S (PS), Protein C (PC) and Antithrombin (AT), the three main natural coagulation inhibitors, as previously described [4]. From the initial set of 705 participants with TGP measurements, 9 were excluded from the final analyses because they exhibited extreme outlier values for $LagTime$.

In the studied population, the correlations between ETP and $LagTime$, ETP and $Peak$ and $LagTime$ and $Peak$ were 0.17, 0.77 and 0.013, respectively.

B. Missing sample structure

All studied variables were divided in seven matrices, ie. blocks, according to the nature of the data. Blocks were of unequal sizes because of missing values. Table I shows the division of missing samples in each data set. Only five blocks showed samples with missing data. The number of samples missing is provided according to each couple of blocks ($Block_{row}, Block_{column}$). Colors of rows and columns symbolize the same blocks respectively and so the diagonal represents the number of missing samples for the single corresponding block. The 696 participants selected for the present analysis had no missing information on the $\mathbf{Y}$ block defined by the three TGP biomarkers (ETP, $Peak$, $LagTime$). As a consequence, working on available data only would lead to the exclusion of 470 individuals, see Table I.

IV. Methods

Many methods are available for high-dimensional data analysis. For example, the sparse PLS, see [5], is popular but deal with missing values in a two-steps approach and not in a supervised framework. Non multi-block and non supervised methods such as softImpute [6] can also deal with missing values but not in the context of supervised analysis. imputeMFA, see [7], deals with missing values but is not

[3] We removed the CpG sites for which InterQuartiles Range was lower than 0.05 and for which maximum absolute correlation with any of the three TGP biomarkers was below 0.25.

TABLE I
MISSING INFORMATION ACCORDING TO THE TYPE OF DATA IN THE 696 PARTICIPANTS INCLUDED IN THE STUDY. ONLY TWO SETS OF INDIVIDUALS, CORRESPONDING TO [a] AND [b] IN THE FOLLOWING TABLE, ARE MISSING IN THREE BLOCKS.

BLOCK	HPAs	PS	F2_G20210A	PC	AGE	BMI	AT	mDNA
HPAs	4	0	0	0	0	0	0	4
PS	0	3	0	2	0	0	0	1
F2_G20210A	0	0	0	0	0	0	0	2
PC	0	2	0	4	0	0	0	4
AGE	0	0	0	0	0	0	0	0
BMI	0	0	0	0	0	2	0	18
AT	0	0	0	0	0	0	0	0
mDNA	4	1	2	4	0	18	0	433
Total	8	6 +6[a]	2	10 +8[a,b]	0	20 +2[b]	0	462 +8[a,b]
	8	12	2	18	0	22	0	470

[a] **6** individuals are missing for {**PS**,**PC**,**mDNA**} consequently.
[b] **2** individuals are missing for {**BMI**,**PC**,**mDNA**} consequently.

supervised and shows poor results in high dimensional setting. Support Vector Machine (SVM), introduced by [8] and neural networks do not allow variable selection and generally require a very large number of samples to achieve efficiency and accuracy. Aggregative methods such as random forests [9] are also very attractive methodologies but are computationally time demanding. We here propose to apply a data driven sparse Partial Least Square (`ddsPLS`, see [10]) that has the multiple advantages of addressing high-dimensional multi-block supervised problems, multivariate regression or classification, in the presence of missing data with regularization and variable selection, and in a time effective manner.
A variance-covariance matrix soft-thresholding algorithm inspired from `ddsPLS` tools allows regularization and variable selection while missing data imputation is performed thanks to the `Koh-Lanta` algorithm that the authors developed. Both of those aspects are described below.

A. A `PLS` (Partial Least Square) inspired method

`PLS` looks for common structure, through singular value decomposition (`SVD`) decomposition, to $\mathbf{X}$ and $\mathbf{Y}$ maximizing $(\mathbf{YX})^T(\mathbf{YX})$. Only the first principal vector is built through the **weight** vector $\mathbf{u}$, resp. $\mathbf{v}$, for the $\mathbf{X}$ part, resp. the $\mathbf{Y}$ part, corresponding to **component** vector $\mathbf{t} = \mathbf{Xu}$, resp. $\mathbf{s} = \mathbf{Yv}$. A technical step, denoted as **deflation** allows to remove the information carried by that **component** to both of the matrices. Classically, once that deflation is performed, the same procedure is done on the residual matrices, building a second then a third up to build R components, as ordered by the user. Deflation is not performed in our method for reasons exposed in [10].

B. `ddsPLS`: a three steps algorithm

The first step permits to extract the marginal R-dimensional common structure of each block t and the block $\mathbf{Y}$. The second step finds a R-dimensional common structure to the T different R-dimensional components and the block $\mathbf{Y}$. The last step builds the linear regression matrix predicting block $\mathbf{Y}$.
The soft-thresholding operation, $\forall \lambda \in [0,1]$, denoted as S_λ, applied to any matrix to each of its coefficient such as $S_\lambda : x \to sign(x)(|x| - \lambda)_+$, where $sign$ gives the sign of a real, $|.|$ denotes the absolute value and $(.)_+$ the max between its argument and 0.
That operator is applied to the different variance-covariance matrices, let say the multi-block data set is built on T blocks and so, $\forall t \in \{1, \cdots, T\}$, the following `SVD` decompositions are performed

$$\max_{\mathbf{U}_t \in \mathbb{R}^{p_t \times R}} \sum_{r=1}^{R} ||S_\lambda\Big(\frac{\mathbf{Y}^T\mathbf{X}_t}{n-1}\Big)\mathbf{u}_t^{(r)}||_2^2$$
$$\text{s.t.} \quad \mathbf{U}_t^T\mathbf{U}_t = \mathbb{I}_R,$$

where each $\mathbf{u}^{(r)}$ is the r^{th} **weight** associated with the block $\mathbf{X}_t$ and the corresponding r^{th} **component** is denoted as $\mathbf{t}_t^{(r)} = \mathbf{X}_t\mathbf{u}_t^{(r)} \in \mathbb{R}^{n\times 1}$. The soft-thresholding operation implies that some coefficients are naturally put to 0 in the resulting matrix permitting efficient and sparse **weight** extraction. In the following example $\mathbf{X}$, resp. $\mathbf{Y}$, is defined through $p = 3$, resp. $q = 2$, variables and the **weight** matrix is indeed sparse

$$\underbrace{\begin{bmatrix} 0.15 & 0.9 \\ 0.5 & 0.2 \\ 0.6 & 0.1 \end{bmatrix}}_{\frac{(\mathbf{Y}^T\mathbf{X})^T}{n-1}} \underset{\lambda=0.2}{\Longrightarrow} \underbrace{\begin{bmatrix} 0 & 0.7 \\ 0.3 & 0 \\ 0.4 & 0 \end{bmatrix}}_{S_\lambda\left(\frac{(\mathbf{Y}^T\mathbf{X})^T}{n-1}\right)} \underset{\text{SVD}}{\Longrightarrow} \mathbf{U} = \left[\underbrace{\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}}_{\mathbf{u}^{(1)}}, \underbrace{\begin{bmatrix} 0 \\ 0.6 \\ 0.8 \end{bmatrix}}_{\mathbf{u}^{(2)}}\right].$$

Once each t structure is defined, a common structure to the T blocks is build thanks to another R-dimensional `SVD` decomposition applied to the concatenation of the T different R-dimensional descriptions $S_\lambda\Big(\frac{\mathbf{Y}^T\mathbf{X}_t}{n-1}\Big)\mathbf{U}_t$. The part of the built weights corresponding to block t is denoted as **super-weight** and is symbolized by $\underline{\boldsymbol{\beta}}_t \in \mathbb{R}^{R\times R}$. It corresponds to the impact of the previously selected variables of every blocks on the **super** description of the block $\mathbf{Y}$ for each **super-component** $\mathbf{X}_t\mathbf{U}_t\underline{\boldsymbol{\beta}}_t \in \mathbb{R}^{n\times R}$. The **scaled super-weights** $\mathbf{U}_t\underline{\boldsymbol{\beta}}_t \in \mathbb{R}^{n\times R}$ permit to interpret the effect of one variable of a given block on the considered **super-component** since its absolute value is inferior or equal to 1 and, for a given **super-component**, variables are ordered in term of importance by their **scaled super-weights**.
A last step builds a regression model such as

$$\mathbf{Y} \approx \sum_{t=1}^{T} \mathbf{X}_t\mathbf{B}_t \in \mathbb{R}^{n\times q},$$

using Moore-Penrose pseudo-inverse in the case of regression and a linear discriminant analysis model (`LDA`) is built on the basis of the R **super-components** to predict new individual classes.

C. The Koh-Lanta algorithm

Missing values might appear in a $train$ or in a $test$ data set but the model is built on the $train$ part and tested on the $test$ part. The way of dealing with the missing values must therefore be different in both cases, which is the objective of the *Koh-Lanta* algorithm developped in [10]. The *Tribe Stage* permits imputation in the train data set and must enrich the being built model while the *Reunification Stage* estimates missing values in the $test$ data set with no modification of the model.
The *Tribe Stage* is an iterative procedure which uses, considering a given block for which samples are missing, pieces of information present for the other samples in the block $\mathbf{Y}$ [4] are used to build a `ddsPLS` model on non missing valued data sets and then estimates potential positions of missing samples. At each iteration of the algorithm, only previously selected variables are taken into account and others are removed from the analysis, this is the *Tribe Stage* of *Koh Lanta*. Convergence is controlled with a maximum number of iterations and a minimum variation of the Moore-Penrose description of the $\mathbf{X}$ part, according to the $\mathcal{L}_2$-cross-product.
The *Reunification Stage* uses the final model of the *Tribe Stage* to predict the missing values of the $test$ data set in a single loop.

D. A new parametrization of the `ddsPLS`

So far `ddsPLS` models depend on two user tunable parameters which are

- $R \in \mathbb{N}^\star$: The number of dimensions to be built.
- $\lambda \in [0, 1]$: The correlation threshold above which an interaction between a variable of a $\mathbf{X}$ block variable and a $\mathbf{Y}$ variable is not taken into account.

The `ddsPLS` has been modified and parameter λ has been replaced with parameter $L_0 \in \mathbf{N}^\star$ which represents the maximum number of $\mathbf{X}$ variables to be selected in the model. But for a given L_0, λ is no unique. The chosen rule was to consider the model corresponding to the smallest λ for a given L_0 because it would eventually give the same degree of sparsity but gathers more information since soft-thresholding operation removes less information in that case.
Also this solution would certainly be efficient in the cases of data sets with variance-covariance matrices particularly sensible to down-sampling but this has not been explored yet.

E. A new initialization of the missing values in the `ddsPLS`

The mean imputation for initializing missing values was so far considered. This drives the algorithm to bias the correlations between the $\mathbf{Y}$ variables and the predictors. In the context of many missing samples, which is the case in the $Methylation$ block ($\approx 67.5\%$ of missing samples) that bias implies sub-optimal choice over the soft-thresholding operation. The most correlated $Methylation$ CpG site is $cg08719422$ and its highest absolute correlation with one of the three TGP biomarkers is equal to:

- ≈ 0.404 on the present individuals,
- ≈ 0.232 if missing samples have been imputed to mean.

In that context it has been decided to slightly modify the missing sample imputations at the initialization step. And so, $\forall t \in \{1, \cdots, T\}$ such as some rows of that block are missing, the others are not missing. The algorithm builds a `ddsPLS` with $Block_t$-non missing samples as a response matrix and the $\mathbf{Y}$ matrix (which is the "official" response matrix) for the $Block_t$-non missing samples as the predictor matrix. Then it computes the predicted values on the $\mathbf{Y}$ matrix for the $Block_t$-missing samples. Those are the initialization missing samples. The regularization and the number of components are taken accordingly to the choice of the user.

V. Results

A. Cross validation

The two parameters of the `ddsPLS` model were tuned with 40-folds cross-validation using a unitary step on L_0. The error criterion chosen is the Mean Square Error in Prediction (MSEP), R is upper bounded by the number of columns in the $\mathbf{Y}$ block, which is equal to 3. While the identified first and second components (referred thereafter to as super-components) substantially contributed to the model by explaining around 24% and 11% of the total variance of $\mathbf{Y}$, respectively, the third component added little information with less than 2% of the variance explained. The MSEP curve displayed on Figure 2 clearly showed that $Peak$ was badly predicted in cross-validation since the error was always above 1, the upper-bound limit for prediction when the outcome variable is standardized. The ETP showed a minimum MSEP for $L_0 = 2$ which then dramatically increased for higher L_0. By contrast, $LagTime$ reached a clear minimal MSEP for $L_0 = 12$. For the following results, we will focus on the model identified with $L_0 = 12$ that provided the best predictions for $LagTime$ and denoted as $\mathcal{M}_{TGP}$ in the following.

TABLE II
SELECTED VARIABLES AND MSEP ERRORS OF $\mathcal{M}_{TGP}$

	Variables selected			MSEP			
L_0	***HPA***	***Bio.***	***mDNA***	***Lagtime***	***Peak***	***ETP***	***Mean***
12	7	2[c]	3	0.796	1.14	1.05	0.994

[c] $F2\_G20210A$ and AGE are selected.

B. Description of the selected model

Model $\mathcal{M}_{TGP}$, see Section V-A, selects a total of 12 variables. In Figure 5 are shown the values, per **super-component**, of the **scaled super-weights**, as described in IV-B, as well as the variance they explain for each of the three TGP biomarkers. Each row of Figure 5 describes the variables that compose one of the two identified **super-components**

[4] Only the $\mathbf{Y}$ is used as a covariate matrix, through its **super-scores**, to use only dimensions linked to the current model.

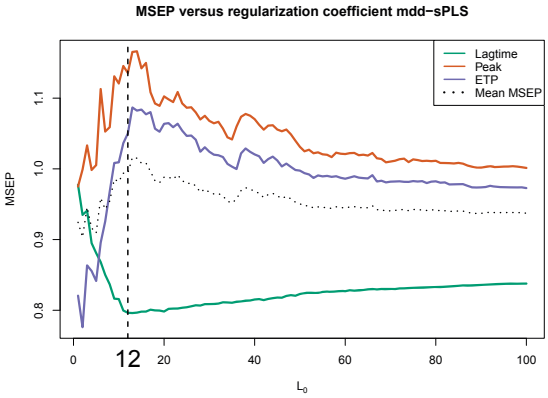


Fig. 2. Mean Square Error in Prediction (MSEP) for $R = 2$ with mean error curves and the mean of both of those curves. The dashed line represents the chosen mode, for $L_0 = 12$. The **Y** variables were standardized before application of the cross validation procedure.

with their corresponding **scaled super-weights**. The second column shows the percentage of variance explained by the **super-component** for each of the three TGP biomarkers. It is important to note that those variances are computed on the imputed data sets, once the model is built.

The first **super-component** explains 24% of the total **Y** joint variance of the three TGP biomarkers, with a decreasing contribution of $Peak$, ETP and $Lagtime$ (40% of explained variance, 29% and 1.9%, respectively). Interestingly, the first **super-component** was mainly driven by one CpG site, `cg08719422`, with some minor contribution of the $F2$ G20210 mutation and two other CpG, `cg18876487` and `cg11015505`. The pattern of correlation between selected features is given on Figure 3.

The three selected CpG sites were correlated with each other and the correlation between `cg08719422` and $Peak$ was 0.64. While this first **super-component** demonstrated good descriptive characteristics with high percentages of variance explained for $Peak$ and ETP, its predictive properties were relatively poor as illustrated by the high values of the corresponding MSEP that were slightly greater than 1.

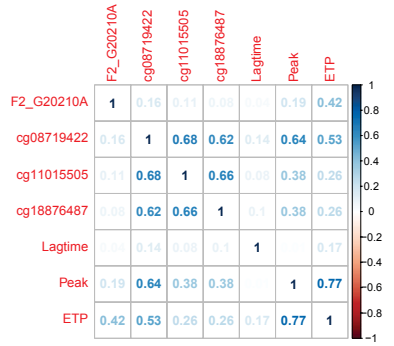


Fig. 3. Correlation structures of super-component 1 of the model

The second **super-component** was mainly driven by Age and seven antibodies, and was mainly associated with $LagTime$. It explained about 23% of the $LagTime$ variance but only 5.8% and 4.3% for ETP and $Peak$ variability, respectively. Figure 4 shows strong correlations between proteins but also with Age, which helps imputation in $test$ data sets. Four of these seven antibodies were targeting proteins of the complement cascade (C5, C9) and two were antibodies targeting proteins (C4BPA, PROS1) associated with the Protein S pathway [11].

The same model including Age and these seven antibodies was identified when the `ddsPLS` algorithm was applied on the $LagTime$ variable only (data not shown).

In order to replicate the observed association of the second super-component with $LagTime$, we investigated it in an independent sample of 133 MARTHA patients measured for $LagTime$ and for the seven antibodies using the same technique but without $Methylation$ data available. In this independent population, we observed a trend for a positive correlation between the seven antibodies signature (plus age) and $LagTime$ (r = 0.16, p = 0.069).

VI. Clinical Interests and Future Works

Using `ddsPLS`, a recently developed method for multi-$omics$ data analysis, we identified a biomarker signature composed of antibodies targeting seven distinct proteins that explain about 20% of the plasma variability of $LagTime$. Interestingly, this signature is enriched in proteins belonging to the complement cascade adding support to the emerging link between the complement and coagulation pathways. Further works are needed to assess the relevance of this signature with respect to the risk of VT.

`ddsPLS` has also permitted to extract information from $Methylation$ data set filed with $\approx 70\%$ of missing values, sadly `ddsPLS` does not yet permit to correctly predict information on $test$ data sets with such a proportion of missing values but this will drive further researches.

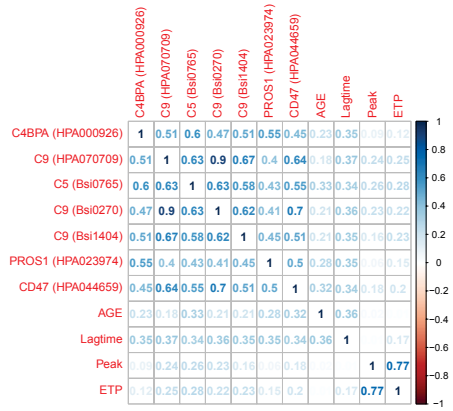


Fig. 4. Correlation structures of super-component 2 of the model

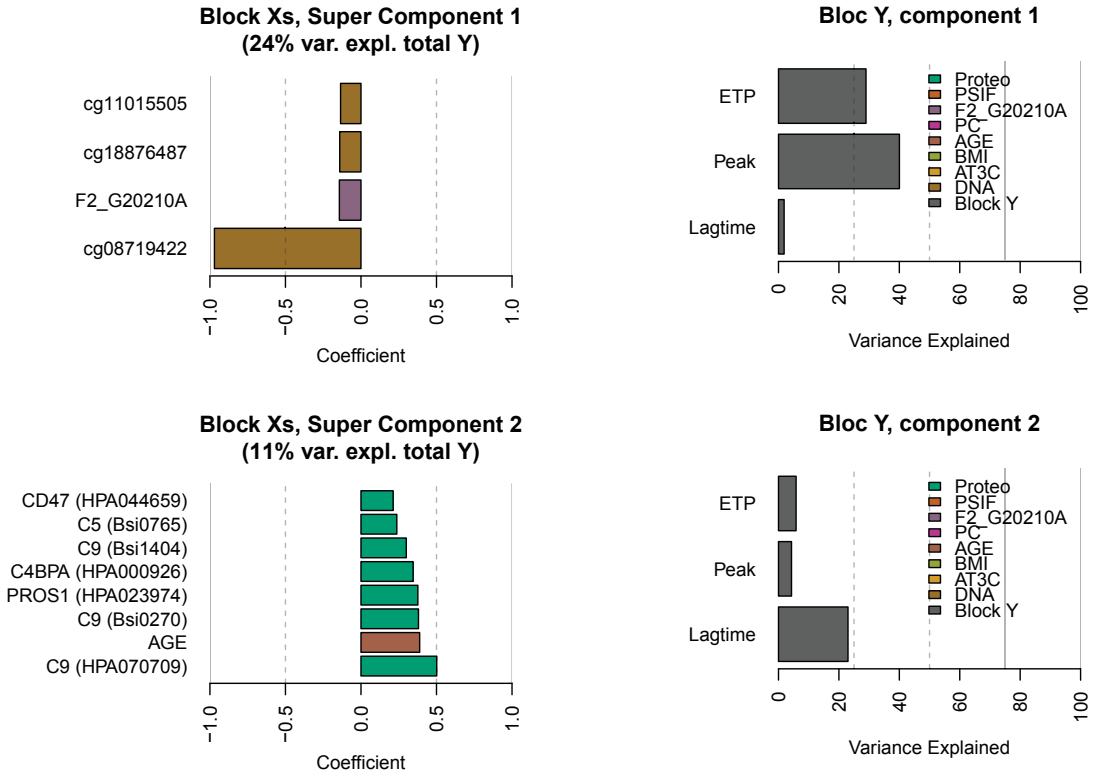


Fig. 5. Scaled super-weights per super-component and variance explained per response variable per component.

Further developments are also needed for widening the use of the `ddsPLS` method in the context of mutli-*omics* epidemiological cohorts. These include the handling of missing data in the response block, denoted as **Y**, and the possibility of integrating millions of genetic variables produced by high-throughput genotyping/sequencing instruments, the latter raising some computational challenges.

Acknowledgment

Hadrien Lorenzo is supported by a 2016 Inria-Inserm thesis grant *Médecine Numérique* (for *Digital Medicine*). Misbah Razzaq is supported by a grant from the GENMED Laboratory of Excellence on Medical Genomics [ANR-10-LABX-0013]. David-Alexandre Trégouët is supported by the EPIDEMIOM-VTE Senior Chair from the Initiative of Excellence of the University of Bordeaux.

References

[1] A. Rocanin-Arjo, W. Cohen, L. Carcaillon, C. Frère, N. Saut, L. Letenneur, M. Alhenc-Gelas, A.-M. Dupuy, M. Bertrand, M.-C. Alessi *et al.*, "A meta-analysis of genome-wide association studies identifies orm1 as a novel gene controlling thrombin generation potential," *Blood*, vol. 123, no. 5, pp. 777–785, 2014.

[2] A. Rocañín-Arjó, J. Dennis, P. Suchon, D. Aïssi, V. Truong, D.-A. Trégouët, F. Gagnon, and P.-E. Morange, "Thrombin generation potential and whole-blood dna methylation," *Thrombosis research*, vol. 135, no. 3, pp. 561–564, 2015.

[3] K. Drobin, P. Nilsson, and J. M. Schwenk, "Highly multiplexed antibody suspension bead arrays for plasma protein profiling," in *The Low Molecular Weight Proteome*. Springer, 2013, pp. 137–145.

[4] T. Oudot-Mellakh, W. Cohen, M. Germain, N. Saut, C. Kallel, D. Zelenika, M. Lathrop, D.-A. Trégouët, and P.-E. Morange, "Genome wide association study for plasma levels of natural anticoagulant inhibitors and protein c anticoagulant pathway: the martha project," *British journal of haematology*, vol. 157, no. 2, pp. 230–239, 2012.

[5] K.-A. Lê Cao, D. Rossouw, C. Robert-Granié, and P. Besse, "A sparse pls for variable selection when integrating omics data," *Statistical applications in genetics and molecular biology*, vol. 7, no. 1, 2008.

[6] T. Hastie, R. Mazumder, J. D. Lee, and R. Zadeh, "Matrix completion and low-rank svd via fast alternating least squares," *The Journal of Machine Learning Research*, vol. 16, no. 1, pp. 3367–3402, 2015.

[7] J. Josse and F. Husson, "missmda: a package for handling missing values in multivariate data analysis," *Journal of Statistical Software*, vol. 70, no. 1, pp. 1–31, 2016.

[8] V. Vapnik, *The nature of statistical learning theory*. Springer science & business media, 2013.

[9] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[10] H. Lorenzo, J. Saracco, and R. Thiébaut, "Supervised learning for multiblock incomplete data," *arXiv preprint arXiv:1901.04380*, 2019.

[11] A. Buil, D.-A. Trégouët, J. C. Souto, N. Saut, M. Germain, M. Rotival, L. Tiret, F. Cambien, M. Lathrop, T. Zeller *et al.*, "C4bpb/c4bpa is a new susceptibility locus for venous thrombosis with unknown protein s independent mechanism: results from genome-wide association and gene expression analyses followed by case-control studies," *Blood*, pp. blood–2010, 2010.

4.2 Prédiction de la tendreté de la viande

Ce travail correspond à une collaboration fructueuse avec Marie-Pierre Ellies-Oury, maître de conférence à Bordeaux Sciences Agro et membre de l'équipe Biomarqueurs, UMR Herbivores 1213 pour l'INRA, Centre Auvergne Rhône-Alpes et a permis la rédaction de Lorenzo et collab. (2019a). Le jeu de données sur lequel la méthode **ddsPLS** a été appliquée détaille 5 morceaux de viande correspondant à 5 muscles différents pour 10 vaches d'une même race, Holstein. Pour chacun de ces morceaux nous avons à disposition 20 marqueurs protéiques ainsi qu'une variable quantitative décrivant sa tendreté.L'objectif est de pouvoir prédire la tendreté de chacun des muscles (et potentiellement de la tendreté de la carcasse) par une combinaison limitée de biomarqueurs protéiques, afin de s'affranchir des méthodes directes (sensorielles ou instrumentales) coûteuses et invasives.
L'identification d'un nombre restreint de protéines marqueurs de tendreté pourrait permettre une orientation précoce des carcasses vers le marché de distribution le plus adapté à son potentiel de tendreté (boucherie, linéaire de supermarché, plats cuisinés, préparation pour cuisson lente...).
Ce travail nous a permis de peaufiner les visualisations et les interprétations de ces dernières, en plus de montrer ses capacités dans le cas d'un échantillon de très faible taille ($n = 10$).

foods

MDPI

Article

An Original Methodology for the Selection of Biomarkers of Tenderness in Five Different Muscles

Marie-Pierre Ellies-Oury [1,†], Hadrien Lorenzo [2,†], Christophe Denoyelle [3], Jérôme Saracco [4] and Brigitte Picard [1,*]

1 Université Clermont Auvergne, INRA, VetAgro Sup, UMR Herbivores, F-63122 Saint-Genès-Champanelle, France; marie-pierre.ellies@inra.fr
2 Univiversité de Bordeaux, Inria BSO, Inserm U1219 Bordeaux Population Health Research Center, SISTM Team, 33800 Bordeaux, France; Hadrien.Lorenzo@u-bordeaux.fr
3 Service Qualite des Carcasses et des Viandes, Institut de l'Elevage, 69007 Lyon, France; Christophe.Denoyelle@idele.fr
4 Inria BSO, CQFD Team, CNRS UMR5251 Institut Mathématiques de Bordeaux, ENSC Bordeaux INP, F-33400 Talence, France; jerome.saracco@math.u-bordeaux1.fr
* Correspondence: brigitte.picard@inra.fr; Tel.: +33-5-57-35-38-70
† The authors contributed equally.

Received: 24 April 2019; Accepted: 8 June 2019; Published: 11 June 2019

Abstract: For several years, studies conducted for discovering tenderness biomarkers have proposed a list of 20 candidates. The aim of the present work was to develop an innovative methodology to select the most predictive among this list. The relative abundance of the proteins was evaluated on five muscles of 10 Holstein cows: *gluteobiceps*, *semimembranosus*, *semitendinosus*, *Triceps brachii* and *Vastus lateralis*. To select the most predictive biomarkers, a multi-block model was used: The Data-Driven Sparse Partial Least Square. *Semimembranosus* and *Vastus lateralis* muscles tenderness could be well predicted (R^2 = 0.95 and 0.94 respectively) with a total of 7 out of the 5 times 20 biomarkers analyzed. An original result is that the predictive proteins were the same for these two muscles: µ-calpain, m-calpain, h2afx and Hsp40 measured in m. *gluteobiceps* and µ-calpain, m-calpain and Hsp70-8 measured in m. *Triceps brachii*. Thus, this method is well adapted to this set of data, making it possible to propose robust candidate biomarkers of tenderness that need to be validated on a larger population.

Keywords: predictive model; tenderness; meat; biomarker; calpain; h2afx

1. Introduction

As emphasized by many authors, tenderness is considered the most important qualitative characteristic of meat. Tenderness is also a highly variable characteristic, and this wide variability appears to be a significant reason for consumer dissatisfaction and reduction in beef consumption. Therefore, tenderness inconsistency is a priority issue for the meat industry [1].

Tenderness can be evaluated by direct methods, such as instrumental or sensorial assessments, or by indirect methods, using muscular characteristics as tenderness predictors. Sensory methods are expensive, difficult to organize and time consuming [2]. In addition, these methods are invasive, and cannot be performed early enough to allow carcasses to be adapted to markets according to their level of tenderness. Thus, there is a need for method that is available early post mortem to estimate the tenderness potential of each carcass, and among the possible methods, the abundance of certain proteins is of particular interest for predicting tenderness [3,4]. Previous works reviewed by Picard et al. [5] made it possible to identify candidate biomarkers of tenderness. They belong to numerous biological pathways: heat shock proteins, oxidative and glycolytic metabolism, oxidative stress, muscle structure and contraction and proteolysis.

Analyse supervisée multibloc en grande dimension

Predicting the overall tenderness of the whole carcass with a reduced number of indicators could be of interest for the beef meat chain. Indeed, with such information, retailers would be able to guide meat samples either to the traditional butchery circuits (if they are of good quality) or to the boning circuit (for those with a poor level of quality), thereby meeting consumer expectations. Therefore, it is interesting to investigate whether biomarkers of one muscle could predict the tenderness of another muscle of the carcass, and thus to evaluate the possibility of predicting the tenderness of a whole carcass by sampling only a reduced number of muscles.

Thus, the aim of the present work was to identify, from among a pool of 20 biomarkers of interest measured in five different muscles, whether a combination of biomarkers could predict the tenderness of each of the five muscles. Therefore, a specific statistical methodology adapted to a low number of samples with a large number of observations [6,7] was tested.

To investigate this question, we looked at five muscles that have been described in the literature as being muscles with different levels of tenderness. These were classified, according the Warner-Bratzler measurements, as tender (3.2 < Warner-Bratzler Shear Force < 3.9 kg; *Triceps brachii* (TB) or intermediate (3.9 < Warner-Bratzler Shear Force < 4.6 kg; *gluteobiceps* (GB), *Vastus lateralis* (VL), *semimembranosus* (SM), *semitendinosus* (ST)) according to [8]. Indeed, data in the literature evaluating the correlations between the sensory tenderness of each of these five muscles and the "carcass sensory tenderness value" showed significant correlation coefficients ($p < 0.0001$) going from 0.75 for the ST muscle, to 0.81 to the VL muscle [9]. According to the literature, meat tenderness variation among muscles of the same carcass might be explained by animal genetics, feeding, handling or slaughter process [10], but also by muscle characteristics. Indeed, muscle background toughness is mainly determined by its organization and amount of connective tissue. This property is also influenced by the level of intramuscular fat, which is known to be highly variable across the muscles [11]. Then, the toughening and tenderization phases occur during postmortem storage of meat, as a result of sarcomere shortening during rigor development. The degree of contraction at which a muscle enters the state of rigor mortis is highly variable among different muscles within the carcass [12]. Thus, it might be supposed that the five muscles studied represent a diversity of physicochemical and muscular characteristics, which could be considered to be representative of the whole carcass.

2. Material and Methods

2.1. Animals

This study was conducted on 10 Holstein cows slaughtered between 29 and 90 months of age (56.5 ± 18.3 months on average) in a commercial slaughterhouse. The animals were slaughtered at an average carcass weight of 319 kg (±22 kg) and their carcasses were classified as 3 for fat score (European grade with a scale going from 1 to 5), with a conformation score going from P= to O-.

The slaughter was made in compliance with current ethical guidelines for animal welfare.

2.2. Muscle Sampling

In this experiment, we selected five muscles, which were excised for each animal and sampled for further analysis. A standardization of the sampling procedures was adopted for each muscle with a clearly identified sample location. The location of each muscle on the carcass can be seen in Figure 1.

Samples for biochemical (biomarkers and myosin heavy chains proportions) analysis were collected 15 min after slaughter, cut into small fragments, frozen in liquid nitrogen and packaged at −80 °C before grinding. Samples intended for sensory analysis were collected 24 h post mortem, and then they were vacuum-packaged and chilled for 7 days at +4 °C for ageing.

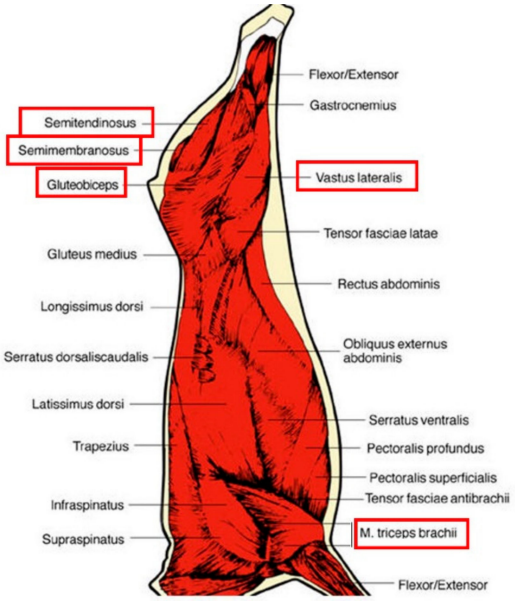


Figure 1. Location of the 5 muscles studied, the names of which are framed in red.

2.3. Biomarker Analysis

Total protein extractions were performed according to Bouley et al. [13]. The protein concentration was determined by spectrophotometry with the Bradford assay [14]. The evaluation of the relative abundance of protein biomarkers was realized by dot blot using antibodies previously validated as described in Guillemin et al. [15] by western-blotting (Table 1; [16]).

Table 1. Gene names and protein names.

Biological Functions	Protein Names	Gene Names
Heat Shock proteins	αB-crystalin	CRYAB
	Hsp20	HspB6
	Hsp27	HspB1
	Hsp40	DNAJA1
	Hsp70-8	HspA8
	Hsp70-1b	HspA1B
	GRP75	HAPS9
Oxidative resistance	Dj1	PARK7
	Cis-Peroxiredoxin	PRDX6
	Super-oxide dismutase Cu/Z	SOD1
Structure	Myosin binding protein H	MyBPH
	Myosin light chain 1F (Mylc-1f)	MYL1
	Myosin heavy chain IIx	MyH1
	α-actin	ACT1
Metabolism	β-Enolase	ENO3
	Phosphoglucomutase 1	PGM1
Apoptosis and signaling	Tumor protein p53	TP53
	H2A Histone Family Member X	H2AFX
Proteolysis	m-calpain	CAPN2
	μ-calpain	CAPN1

For dot-blot analysis, 15 µg of protein samples were spotted on a nitrocellulose membrane using Minifold I Dot-Blot apparatus. Dot-Blots were saturated during 1 h at room temperature with Analyse supervisée multidimensionnelle

a 10% milk blocking buffer at 37 °C for 20 min. Then a primary antibody specific to the protein studied was incubated at 37 °C for 90 min. The anti-mouse fluorochrome-conjugated LICOR-antibody IRDye 800CW (1 mg/mL) diluted at 1/20,000 in 1% milk blocking buffer was hybridized for 30 min at 37 °C. Membranes were scanned by the Odyssey scanner (LI-COR Biosciences, Lincoln, NE, USA) and analyzed with GenePix (Axon Laboratory, Union City, CA, USA).

2.4. Myosin Heavy Chain Isoforms Proportions

The different types of myosin heavy chain (MyHC) isoforms were determined on the basis of previously determined migration patterns [17] using sodium Dodecyl Sulfate polyacrylamide gel electrophoresis (SDS-PAGE). This protocol was adapted from that of Talmadge and Roy [18]. Myofibrillar proteins were extracted from 200 mg of muscles with a buffer containing 0.5 M NaCl, 20 mM NaPPi, 50 mM Tris, 1 mM EDTA and 1mM DTT according to the protocol described in [19]. Then 5 µg of proteins were loaded in each well on a Mini-Protein II Dual Slab Cell electrophoretic system (Bio-Rad, Hercules, CA, USA). The separating gel consisted of 35% (*w*/*v*) glycerol, 9% (*w*/*v*) acrylamide-N,N'-methylenebisacrylamide (Bis) (50:1), 230 mM Tris (pH 8.8), 115 mM glycine, and 0.4% *w*/*v* SDS and the stacking gel of 47% (*w*/*v*) glycerol, 6% (*w*/*v*) acrylamide-Bis (50:1), 110mM Tris (pH 6.8), 6 mM EDTA, and 0.4% (*w*/*v*) SDS. The electrophoresis was performed at a constant voltage of 70 V for 30 h at 4 °C. At the end of migration, the gels were stained with Coomassie Blue [19] (Figure 2), and relative amounts of the different MyHC isoforms were quantified using ImageQuant TL v2003 software (Amersham Biosciences Corp., Piscataway, NJ, USA).

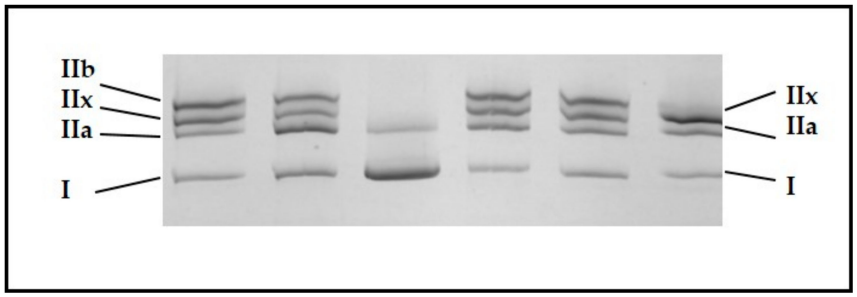


Figure 2. Illustration of the separation of the different myosin heavy chains isoforms depending on their molecular weight in different cattle skeletal muscles according to the protocol of Picard et al. [17].

2.5. Tenderness Evaluation

After ageing, the 5 muscles were trimmed and cut into 1.5 cm thick homogeneous steaks, then vacuum-packaged a second time and frozen at −20 °C until the sensory analysis. Meat in the form of 15 mm steaks was thawed and then cooked for 1 min 45 s in an Infra grill Duo Sofraca set at a temperature of 300 °C. After cooking, the steaks were cut into 20 mm cubes that were served on a plastic plate at an internal temperature of 55 °C.

Sensory assessment was conducted by a 12-member panel selected, trained and controlled according to the XP V09-503 standard. The products were evaluated according to the conventional profile method (QDA) based on NF ISO 13300 [20]. Samples were presented in a monadic mode to the panelists.

The panelists evaluated the cooked samples for tenderness. Each attribute was rated on a 100-point non-graduated scale with a score of zero, on an ascending scale of quality for each attribute, being equivalent to tough, and a score of 100 being equivalent to tender. The sessions were carried out in a sensory analysis room equipped with individual boxes, under artificial non-colored lighting (Institut de l'Elevage, Villers-Bocage, France).

Warner-Bratzler shear force on cooked samples (internal temperature of 55 °C) was measured according to the method of Schakelford et al. [21] with an INSTRON 3343 material testing machine.

For each sample (each muscle of each animal), 10 measurements of shear force were made at different thicknesses of meat cores between 0.8 and 1.2 cm (Institut de l'Elevage, Villers-Bocage, France).

2.6. Statistical Analysis

Firstly, basic statistics, such as variance analysis and mean multiple comparisons with one factor and Pearson correlations were carried out using R-software [22].

Then, a specific statistical methodology adapted to a low number of samples with a large number of observations was tested. This method is a purely geometrical dimension reduction approach, which is suitable for managing a limited sample size with numerous variables stored in blocks. This makes it possible to provide interpretable information from the available data. However, it is important to mention that one should be wary of generalizing the results too quickly. Indeed, the generalization of the results would require a validation on a larger sample. Nevertheless, the proposed method will make it possible to clearly highlight some links that exist between the biomarkers and the force/or the tenderness on the available dataset.

2.6.1. Methodology for Biomarkers Selection

Among the different methodologies that can deal with multi-block structured data sets, the Data-Driven Sparse Partial Least Square (ddsPLS) shows interest for supervised problems whether in the case of regression (i.e., when the response variables are numerical) or classification (i.e., when the response variables are categorical). The ddsPLS approach allows variable selection in the covariate and in the output parts using two parameters, which are: R, the number of components in the model, and L_0, the maximum number of variables to be selected in the final model in the covariate part. These two parameters are fixed by cross-validation according to general supervised learning problem optimization solutions. The block structure can be designed according to many known factors, which are the time or the type of variable chosen for example.

Please note that, even if there are no missing values in the data set analyzed in the present work, the ddsPLS approach was initially developed to deal with missing samples, such as entire rows of missing values in given blocks.

2.6.2. Partial Least Square (PLS)

Let $X \in \mathbb{R}^{n \times p}$ and $Y \in \mathbb{R}^{n \times p}$ be, respectively, the covariate matrix and the response matrix, describing n individuals through p resp. q variables. The *PLS* problem has been introduced by [18]. Its objective is to maximize the norm of the projected variance-covariance matrix $Y^T X \in \mathbb{R}^{q \times p}$ along the principal direction defined through $(u, v) \in \mathbb{R}^p \times \mathbb{R}^q$ with $(Yv)^T Xu \in \mathbb{R}$ being the chosen projection of individuals that must be maximized, under the unity constraints $u^T u = v^T v = 1$. Finally, the optimization problem can be written as $(u, v) = argmax_{u^T u = v^T v = 1} (Yv)^T Xu$.

This makes it possible to define u and v as the weights of the first component of the model for the X part and for the Y part respectively. The nipals algorithm [18] makes it possible to find u and v in the same time.

Those two weights make it possible to define $t = Xu \in \mathbb{R}^n$ and $s = Yv \in \mathbb{R}^n$, with the respective X and Y parts being first component scores of the model. In other words, they represent the individual positions in the dimension found.

Once that first component is defined, other dimensions are successively obtained thanks to the same kind of optimization. The deflation step makes it possible to remove the information from the current dimension to the residual data set. This step ensures that each component is orthogonal to the others.

The ddsPLS algorithm does not use the deflation step, since it has raised many questions for years [19]. It performs directly the R-dimensional Singular Value Decomposition (*SVD*) of the soft-thresholded variance covariance matrix. This solution does not permit the creation of more components than the rank of the soft-thresholded variance covariance matrix, which is itself majorized

by the rank of the variance covariance matrix. In the high-dimensional context, this represents the number of variables in the Y part for multivariate regression, or the number of classes minus one in the classification cases.

2.6.3. Data-Driven Sparse Partial Square (ddsPLS)

In the multi-block framework, the covariate part is now described by T blocks (different covariates matrices) $X_t \in \mathbb{R}^{n \times p_t}, t = 1, \ldots, T$ describing the same n individuals. The ddsPLS algorithm, for a couple (R, L_0) chosen by the user, begins with the T SVD of each of the variance-covariance soft-thresholded matrices $S_\lambda\left(\frac{Y^T X_t}{n-1}\right)_+$. $S_\lambda : x \to sign(x)(|x| - \lambda)_+$, where $sign$ denotes the sign of a real, $|.|$ denotes the absolute value, and $(.)_+$ the max between its argument and 0. The parameter $\lambda > 0$ is chosen such as the cumulative number of non-null columns in the matrices $S_\lambda\left(\frac{Y^T X_t}{n-1}\right), t \in \{1, \ldots, T\}$ is exactly equal to L_0 and $\forall \partial\lambda \in \mathbb{R}^*_+$ the cumulative number of non-null columns in the matrices $S_{\lambda - \partial\lambda}\left(\frac{Y^T X_t}{n-1}\right)_+, t \in \{1, \ldots, T\}$ is strictly higher than $L_0 + 1$. Let $U_t \in \mathbb{R}^{p_t \times R}, t \in \{1, \ldots, T\}$, denote the respective R-dimensional weights obtained through the corresponding R-dimensional SVD.

The next step makes it possible to gather that information through the super-weights, which are symbolized by $\beta_t \in \mathbb{R}^{R \times R}$. The scaled super-weights $U_t\beta_t \in \mathbb{R}^{n \times R}$ show the effect of one variable of block t on each of the super-components.

Finally, a third step makes it possible to build a regression model based on T regression matrices B_t, such as

$$Y \approx \sum_{t=1}^{T} X_t B_t \in \mathbb{R}^{n \times q}$$

Using the Moore-Penrose pseudo-inverse, which does not imply matrix inversion, is particularly tricky in high dimensions or when $p_t > n$ in the context of regression. Let us mention that a linear discriminant analysis model is built using the *super-components* to predict new individual classes in the context of classification.

2.6.4. Model Selection

As previously mentioned, the ddsPLS approach requires two parameters to be determined by the user: L_0, the maximum number of covariates (in the X part described by the T blocks) to be included in the model, and R, the number of components (calculated in the T underlying SVD), which is the analogous of the number of components in Principal Component Analysis.

Those parameters are determined according to the minimization of a chosen error risk. For the regression framework considered in this paper, the Mean Square Error in Prediction ($MSEP$) is naturally minimized over the response variables, i.e., the five tenderness variables. The error is computed on the Leave-One-Out (LOO) cross-validation predictions, denoted $\hat{y}_{m,i}$, for muscle $m \in \{VL, ST, TB, SM, GB\}$ and for carcass $i \in [1, 10]$:

$$\forall m \in \{VL, ST, TB, SM, GB\}, MSEP(m) = \frac{1}{n} \sum_{i=1}^{10} (y_{m,i} - \hat{y}_{m,i})^2$$

Since it has been chosen to standardize the Y matrix (zero-mean and unit-variance), when MSEP values are lower than 1, this reveals that the cross-validated models are better (on average) than just predicting the average of the selected sample at each iteration of the LOO cross-validation process, and thus the ddsPLS approach succeeds in retrieving links between the blocks of covariates and the corresponding response variables. On the other hand, when MSEP values are larger than 1, this indicates that predicting to the average of the selected sample at each iteration of the LOO cross-validation process provides more accurate predictions than the underlying cross-validated

models, and thus the ddsPLS approach fails to retrieve information from the blocks of covariates and the corresponding response variables.

3. Results

3.1. Data Description

The five muscles were characterized by various slow oxidative MyHC I ($p = 0.007$) and fast oxido-glycolytic MyHC IIA ($p < 0.001$) proportions, whereas MyHC IIX (fast glycolytic) proportions were not significantly different among muscles ($p = 0.129$; Table 2).

The m. *semimembranosus* (SM) appeared to be the least slow oxidative muscle, with 10.5% of MyHC I, whereas the m. *gluteobiceps* (GB) and m. *Triceps Brachii* (TB), with proportions of MyHC I twice as large (20.4 and 22.5%, respectively), were the slowest oxidative muscles. The m. *semitendinosus* (ST) might be considered to be the most glycolytic muscle; even if the proportions of MyHC IIX were not significantly different among the five muscles, this muscle had low proportions of both MyHC I and MyHC IIA.

Table 2. Muscular characteristics of the 5 muscles.

Variables	GB	SM	ST	TB	VL	SEM	*p*
MyHC I (%)	20.4 ^{b}	10.5 ^{a}	16.8 ^{ab}	22.5 ^{b}	15.2 ^{ab}	1.15	$p = 0.007$
MyHC IIA (%)	36.4 ^{bc}	42.9 ^{c}	25.8 ^{a}	28.8 ^{ab}	38.2 ^{bc}	1.50	$p < 0.001$
MyHC IIX (%)	43.2	46.6	57.4	48.7	46.6	1.80	$p = 0.129$
Tenderness (on 100)	33.7 ^{a}	42.3 ^{a}	43.3 ^{a}	57.9 ^{b}	41.9 ^{a}	1.85	$p < 0.001$
Shear force (daN)	11.0 ^{b}	6.4 ^{a}	7.4 ^{a}	5.2 ^{a}	6.8 ^{a}	0.39	$p < 0.001$

Different letters indicate a significant difference ($p \leq 0.05$) between the muscles. GB, *gluteobiceps*; SM, *semimembranosus*; ST, *semitendinosus*; TB, *Triceps Brachii*; VL, *Vastus lateralis*; SEM, standard error of the mean.

The TB muscle was the tenderest muscle, according the panelists. In accordance with this result, this muscle also had the lowest Warner-Bratzler shear force (Table 2). At the same time, the GB, which was the toughest muscle according to the mechanical analysis, had the lowest scores for sensory tenderness. These data are related to the significant correlation between tenderness scores and shear force values observed in this study (correlation: −0.53 when considering the five muscles together; $p < 0.001$). However, the correlation between tenderness and shear force is very muscle-dependent: for VL and ST muscles, the correlation is negative and significant (respectively −0.68 and 0.79), whereas for muscles GB, SM and TB, the correlation is non-significant.

Among the 20 analyzed biomarkers, three did not show a difference in relative abundances between the five muscles: αB-crystalin ($p = 0.179$), prdx6 ($p = 0.293$) and α-actin ($p = 0.323$) (Table 3).

The m. *Vastus lateralis* (VL) showed relative abundances significantly different from other muscles for many biomarkers. Indeed, it contained significantly less eno3, mylc1f, and Hsp40, and more m-calpain, pgm1, h2afx, and MyhcIIX than the other four muscles.

However, the VL muscle was close to the ST muscle for many biomarkers, especially Hsp70-1a, grp75 and mybph.

The SM muscle might be distinguished by a relatively high abundance of Hsp27, Hsp40, Hsp70-8, Dj1, sod1, mylc1f and μ-calpain.

The relative abundance of GB and TB muscle biomarkers was close, except for the Hsp20, which was lower in the TB muscle than in the GB muscle.

Table 3. Relative abundance of each biomarker in the five muscles.

Biomarkers	ST	SM	GB	TB	VL	SEM	*p*
Hsp20	95 ^{a}	150 ^{b}	152 ^{b}	99 ^{a}	68 ^{a}	7.7	<0.001
Hsp27	101 ^{ab}	129 ^{c}	118 ^{bc}	111 ^{bc}	86 ^{a}	3.5	<0.001
Hsp40	118 ^{b}	162 ^{c}	112 ^{ab}	117 ^{b}	99 ^{a}	4.0	<0.001
Hsp70-8	105 ^{a}	141 ^{bc}	123 ^{ab}	129 ^{abc}	156 ^{c}	4.9	=0.012
Hsp70-1b	85 ^{a}	188 ^{c}	155 ^{bc}	146 ^{b}	81 ^{a}	8.5	<0.001
GRP75	119 ^{a}	153 ^{b}	147 ^{b}	155 ^{b}	122 ^{a}	4.0	=0.002
αB-crystalin	147 ^{a}	195 ^{a}	211 ^{a}	153 ^{a}	156 ^{a}	10.3	=0.179
Dj1	95 ^{a}	125 ^{c}	110 ^{abc}	114 ^{bc}	101 ^{ab}	2.8	=0.004
PRDX6	97 ^{a}	110 ^{a}	110 ^{a}	110 ^{a}	115 ^{a}	2.7	=0.293
SOD1	79 ^{ab}	95 ^{a}	68 ^{b}	62 ^{b}	58 ^{b}	3.7	=0.008
MyBP-H	102 ^{a}	128 ^{b}	140 ^{b}	144 ^{b}	101 ^{a}	3.8	<0.001
Mylc-1f	78 ^{ab}	109 ^{c}	74 ^{a}	89 ^{b}	47 ^{d}	3.6	<0.001
MyHCIIx	27 ^{a}	33 ^{a}	28 ^{a}	29 ^{a}	45 ^{b}	1.3	<0.001
ENO3	103 ^{a}	134 ^{b}	121 ^{b}	133 ^{b}	83 ^{c}	3.5	<0.001
h2afx	137 ^{a}	173 ^{b}	127 ^{a}	139 ^{a}	226 ^{c}	6.4	<0.001
PGM1	121 ^{a}	165 ^{b}	132 ^{a}	132 ^{a}	306 ^{c}	10.8	<0.001
TP53	106 ^{a}	142 ^{b}	94 ^{a}	104 ^{a}	153 ^{b}	4.2	<0.001
α-actin	82 ^{a}	89 ^{a}	92 ^{a}	96 ^{a}	99 ^{a}	2.7	=0.323
m-calpain	106 ^{a}	139 ^{b}	120 ^{ab}	126 ^{ab}	214 ^{c}	6.8	=0.003
μ-calpain	123 ^{a}	175 ^{b}	136 ^{a}	118 ^{a}	134 ^{a}	5.3	<0.001

3.2. Links between Biomarkers and Tenderness Intra Muscles

When considering the tenderness of 1 muscle with associated biomarkers in the same muscle, Pearson correlations make it possible to highlight the following significant correlations (Figure 3):

- in the ST muscle, sensorial tenderness was negatively linked to Hsp40, Hsp27, pgm1
- in the SM muscle, sensorial tenderness was negatively linked only to mylc1f
- in the VL, GB and TB muscles, the sensorial tenderness was not significantly correlated with the abundance of any biomarkers (Figure 3).

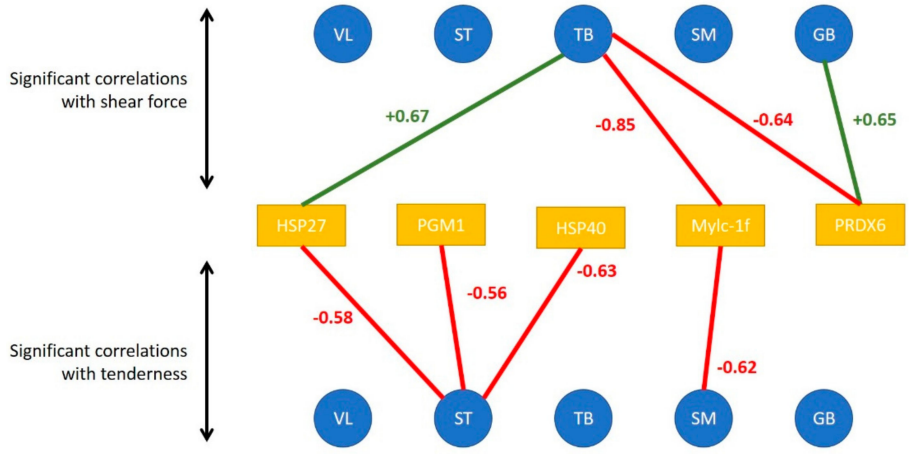


Figure 3. Significant correlations between tenderness/shear force of each muscle (represented by the blue disks) and the abundance of muscular biomarkers of the concerned muscle (represented by the orange rectangles). Red values correspond to negative correlations and green ones to negative correlations.

When considering the correlations between shear force and protein biomarkers in the same muscle, Pearson correlations make it possible to highlight the following significant correlations:

- in the GB muscle, shear force was positively linked to prdx6
- in the TB muscle, shear force was negatively linked to prdx6 and mylc1f, and positively to Hsp27;
- in the other three muscles, the shear force was not significantly correlated with the abundance of an analyzed biomarker.

3.3. Links between Biomarkers and Tenderness among Muscles

When considering the correlation that appeared between the tenderness of one muscle and the biomarker abundance of the 4 other muscles, the highest number of significant correlations were observed for VL and SM tenderness with biomarkers measured in GB and TB muscles (Figure 4).

For biomarkers measured in GB muscle, the coefficient of correlation with VL tenderness were high particularly for h2afx (+0.85), m-calpain (+0.83), μ-calpain (+0.80), Hsp40 (+0.76), Hsp70-8 (+0.65) and for m-calpain (+0.77) and μ-calpain (+0.86), Hsp70-8 (+0.71) with SM tenderness.

For the biomarkers measured in TB muscle, high correlations were observed with VL tenderness for Hsp70-8 (+0.91), m-and μ- calpains (+0.77), as observed with biomarkers measured in GB muscle. For tenderness of SM muscle, high correlations were found with m- and μ-calpains (+0.89 and +0.71, respectively), Hsp70-8 (+0.63) as observed with VL tenderness, and with α-B crystalin (+0.63).

The tenderness of these two muscles VL and SM was especially correlated with biomarkers of GB and TB muscles.

Among these correlations, it might be noted that among biomarkers, μ-calpain, m-calpain, Hsp70-1a, Hsp70-8, tp53, pgm1 and mybph are always positively linked with tenderness, as opposed to MyhcIIx and Hsp40, which can be either positively or negatively linked to meat tenderness. Some biomarkers appeared to be more frequently correlated with tenderness: m-calpain, μ-calpain and Hsp70-8. These three biomarkers, evaluated in the TB and GB muscles, might be considered interesting biomarkers to predict meat tenderness.

Some coefficients of correlations were observed between biomarkers from TB muscle and tenderness of ST, SM and VL muscles (positively). However, biomarkers from TB muscles are mostly correlated with ST and TB shear force (negatively). It should be noted that many correlations can be observed with the tenderness of ST: dj1 (the highest; +0.76), MyhcIIx (in accordance with previous results in this muscle; +0.64), pgm1 (+0.58) and a-actin (+0.59) (Figure 4).

Many correlations were noted between the proteins measured in SM muscle and GB tenderness, whereas for the other muscles, only a few correlations were significant. The highest number of correlations measured in ST muscle was observed for the tenderness of GB muscle (Figure 4).

With regard to shear force, the muscle that is the most correlated with biomarkers is the ST muscle, the shear force of this muscle being especially correlated with biomarkers from VL, TB and GB muscles (Figure 5). The biomarkers that are significantly correlated with shear force are quite different from those found for muscle tenderness even if shear force and tenderness are significantly correlated in ST and VL muscles.

The details of the correlations obtained between the either tenderness of shear force of each of the five muscles and the 20 biomarkers of each muscle have been annexed in Figure 5.

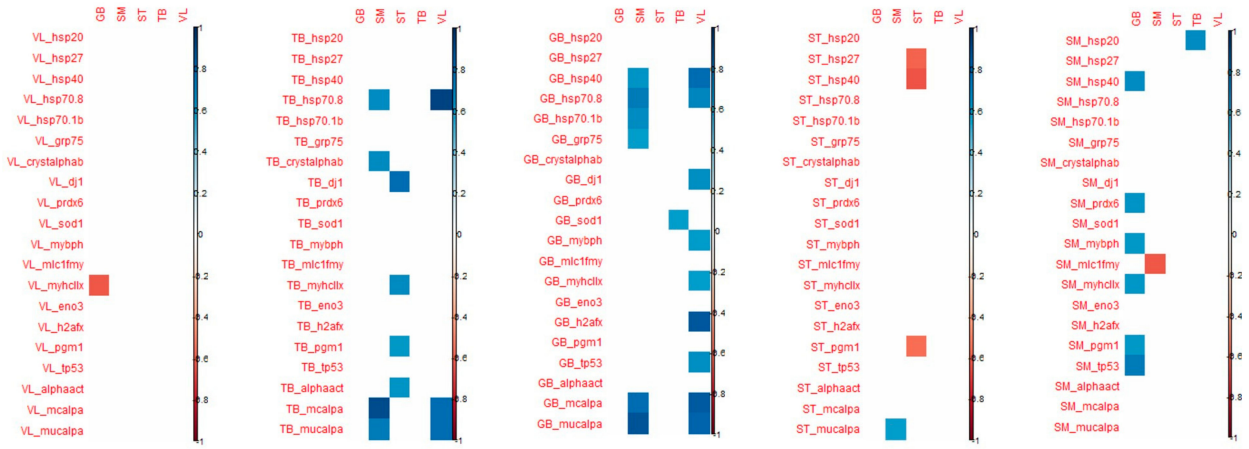


Figure 4. Correlations demonstrated between the tenderness of each of the 5 muscles (in column) and the 20 biomarkers of each muscle (in-line). Only the coefficients that were significant (corresponding to those that were superior or equal to 0.55/inferior or equal to −0.55) are indicated (positive correlations in blue and negative ones in red).

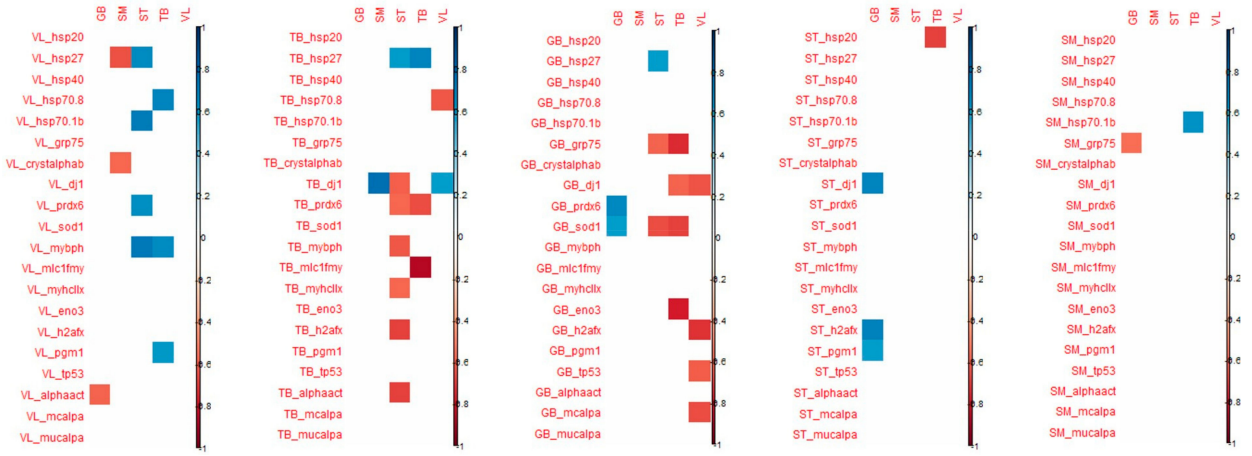


Figure 5. Correlations demonstrated between the shear force of each of the 5 muscles (in columns) and the 20 biomarkers of each muscle (in lines). Only the coefficients that were significant (corresponding to those that were superior or equal to 0.55/inferior or equal to −0.55) are indicated (positive correlations in blue and in negative ones red).

3.4. Selection of Biomarkers the Most Predictive of Tenderness

To predict tenderness, the ddsPLS multi-block approach was used for five blocks of covariates X_t , $t = 1, \ldots, 5$ (each one corresponding to the pool of 20 biomarkers for each of the 5 muscles) and for one block of response variables Y (corresponding to the five tenderness scores, one score for each muscle). Note that in the model, we entered a 6th block, X_6, consisting of performance variables (age/weight). The covariates of X_6 were never selected in tenderness prediction, indicating that the protocol is well balanced, and that the variability of the tenderness is explained neither by weight or by age, nor by the combination of those factors.

Among all the proposed models, we selected the one that minimizes the *MSEP* (see Figure 6). Thus, the model with $L_0 = 8$ and $R = 2$ was selected.

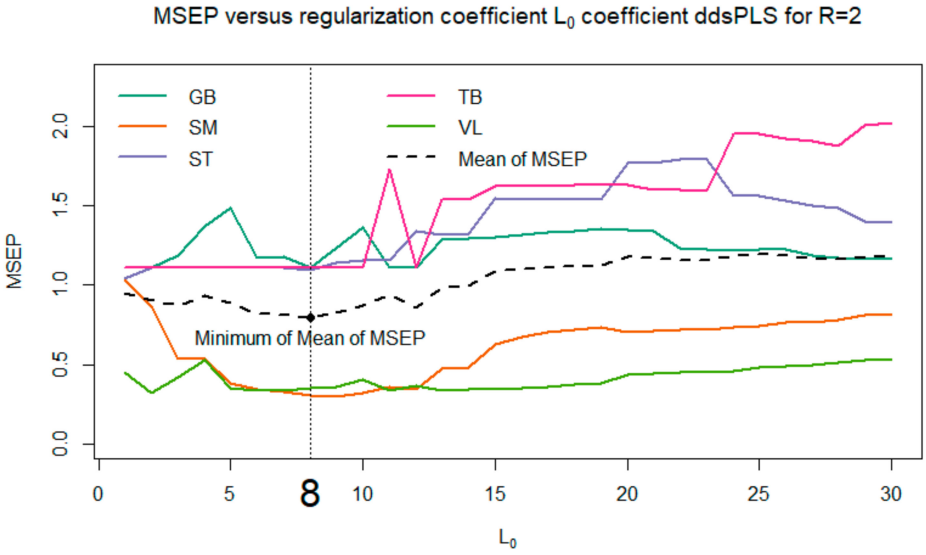


Figure 6. Evolution of the mean square error in prediction for each tenderness muscle and for the average tenderness depending on the maximum number of biomarkers selected in the model. In the present figure, the maximum is fixed at 30 biomarkers out of 100, i.e., 5 time 20 biomarkers.

For the three muscles GB, ST and TB, the model indicated that the best way to predict tenderness is to predict the average value of the corresponding tenderness (i.e., the predicted values are equal to a constant, since there is no link with biomarkers). Hence, for these three muscles, we observe a horizontal alignment of the predicted values of tenderness, as shown in Figure 7.

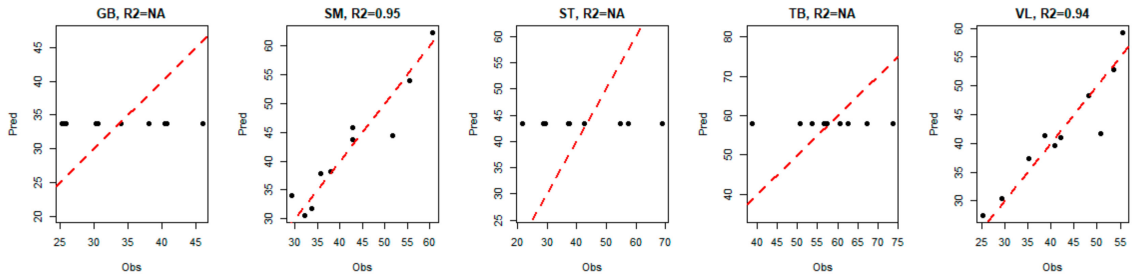


Figure 7. Evolution of predicted tenderness according to observed tenderness in each muscle.

For VL and SM muscles, the seven biomarkers that were selected by the model to predict global tenderness were derived from two different muscles: GB and TB. However, none of the selected biomarkers came from the muscle for which tenderness was predicted (Table 4). Among these seven biomarkers, two are derived from both GB muscle and TB muscle, leading to the number of distinct biomarkers solicited in the model being five.

Table 4. Biomarkers selected of tenderness prediction.

Muscle in Which Biomarkers Were Measured	Biomarkers Selected for the Prediction of Tenderness	
	SM	**VL**
GB	μ-calpain (+) m-calpain (+) h2afx (−) Hsp40 (−)	μ-calpain (+) m-calpain (+) h2afx (+) Hsp40 (+)
SM	-	-
ST	-	-
TB	μ-calpain (−) m-calpain (+) Hsp70-8 (−)	μ-calpain (+) m-calpain (+) Hsp70-8 (+)
VL	-	-
R^2	0.95	0.94

The seven selected biomarkers measured in GB or TB muscles were the same for VL tenderness prediction and for SM tenderness prediction. These proteins belong to the heat shock protein family (Hsp40 and Hsp70-8) or were associated with the proteolysis (m-calpain and μ-calpain) and to the cellular response to DNA damage during oxidative stress (h2afx). Among the biomarkers selected in the model, two were selected in both the GB and the TB muscles: μ-calpain and m-calpain. The three other biomarkers are not the same; the model selected the h2afx and Hsp40 from the GB muscle and the Hsp70-8 from the TB muscle.

It is interesting to note that three biomarkers (see Figure 8) enter with a positive impact on SM and VL tenderness prediction: μ-calpain and m-calpain of the GB muscle and m-calpain of the TB muscle. Whereas the other biomarkers are associated with tenderness prediction either positively (in the VL muscle) or negatively (in the SM muscle), depending on the muscle in which it was measured: h2afx and Hsp40 of the GB muscle, Hsp70-8 of the TB muscle.

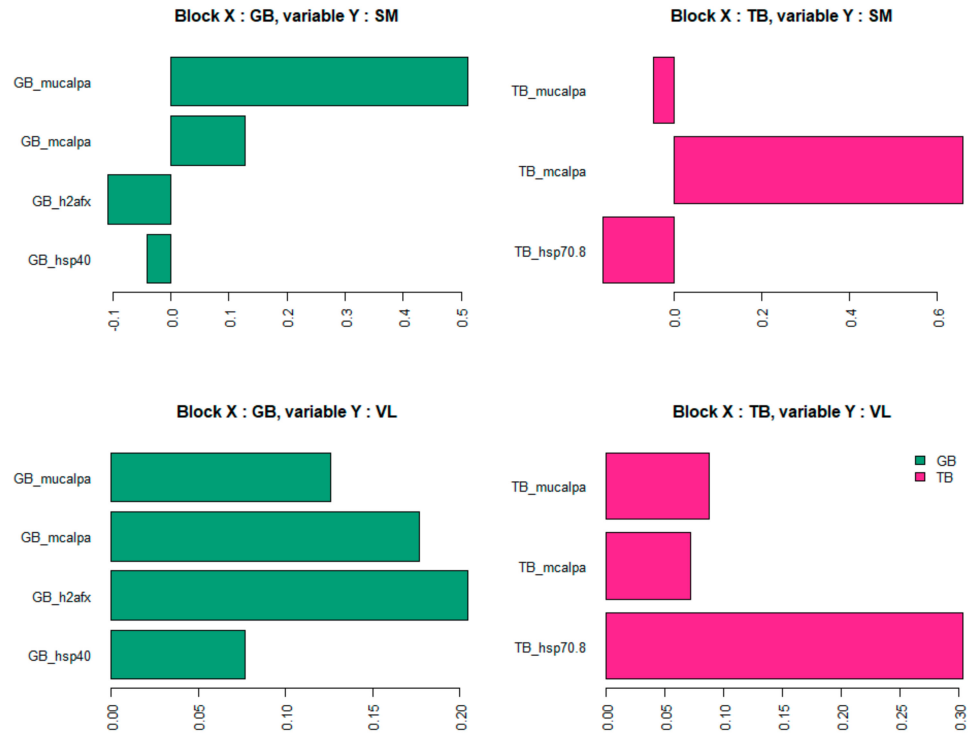


Figure 8. Coefficients associated with each biomarker in tenderness prediction models (representing the relative importance of each selected biomarker).

Thus, the contribution of biomarkers in the prediction of tenderness seems to be modulated according to the muscle.

The same analysis was performed by replacing the tenderness score with the shear force value. It appears that none of the shear force values were correctly predicted by the muscular biomarkers, regardless of the origin of the biomarkers. The model explained no more than 20% of the shear force, except for the TB muscle, where 75% of the shear force could be explained by 2 proteins: the eno3 coming from the GB muscle, and the mylc1f coming from the TB muscle. However, due to the poor quality of the model, it was difficult to draw conclusions on the relevance of these two biomarkers in a definitive way.

4. Discussion

To complete the approach presented here, several models were evaluated for various values of the parameters L_0 and R; the corresponding results are not provided here. One can observe that these models show good stability. More precisely, if we focused in the present paper on the case $L_0 = 8$ and $R = 2$, which makes it possible to minimize the mean of MSEP, we also studied cases where $L_0 \in \{4, 5, 6, 7, 8, 9\}$ (corresponding to the flat area of the MSEP curves in Figure 6). It appears that the biomarkers selected for $L_0 = 4$ are successively supplemented with 1, 2, 3 (and so on) biomarkers when selecting a maximum of 5, 6, 7 (and so on) biomarkers. Moreover, we also tried to explore the models based on a multi-block model with a block of response variables $\boldsymbol{Y}$ built only with the VL and SM tenderness, instead of the tenderness scores of the five muscles. The corresponding models selected the same biomarkers as previously indicated, and returned R^2 values close to the values indicated in Figure 7. These different elements confirm the interest and relevance of the method, accurately predicting meat tenderness. The use of a multi-block model here (based on a linear relationship) allows us to highlight significant links between the tenderness and the abundance of certain biomarkers, even if they were not significantly correlated with tenderness. It is not inconsistent that the combination of several markers (individually little correlated with tenderness) can provide a relevant prediction of tenderness. It can also be noted that several biomarkers, significantly correlated with muscle tenderness, were not selected in the predictive models; this was certainly due to there being redundant information among different biomarkers. The two approaches, correlations and predictions, are thus fully complementary for resulting in relevant conclusions.

The highest number of significant correlations was observed for tenderness of VL and SM muscles with biomarkers measured in GB and TB muscles, mainly for VL tenderness with 12 proteins significantly correlated with tenderness, and 13 in SM muscle. Some coefficients of correlation were high both with SM and VL tenderness: for Hsp70-8, m-calpain and μ-calpain measured in GB and TB muscles. High correlations were found with VL tenderness for h2afx, and Hsp40 measured in GB and TB muscles. These proteins are the same that retained in the equations of prediction of the tenderness of these two muscles with the protein measured in GB muscle. Globally, the highest coefficients were found for VL and SM tenderness for most of the correlations: Hsp20, Hsp40, Hsp70-8, prdx6, dj1, MyBP-H, MyhcIIX, eno3, h2afx, pgm1, m- and μ-calpain. This suggests an important contribution of these proteins in the tenderness of these muscles.

Two models made it possible to predict the tenderness of a muscle (SM and VL) by the biomarkers of two other muscles of the carcasses (GB and TB). The selected proteins are the same in both predicted muscles, which testifies to the importance of these proteins in tenderness construction. Nevertheless, the coefficients associated with certain biomarkers highlight a different hierarchy in the selected biomarkers. Moreover, it might be noted that some coefficients might be reversed from one muscle to another. The fact that tenderness can be predicted only in the two muscles VL and SM can be explained by the fact that these two muscles are very different from the other three in terms of abundances of the analyzed proteins. The two muscles VL and SM are opposite for some proteins such as mylc1f and Hsp40, with the highest abundances being found in SM and the lowest in VL muscle. The abundances

of m-calpain and μ-calpain are particular to these two muscles, in comparison to the three others, as m-calpain abundance was the highest in VL and μ-calpain abundance was the highest in SM muscle.

Both correlation and prediction analysis highlighted five proteins among the 20 putatively analyzed as interesting candidate biomarkers of tenderness: m-calpain, μ-calpain, Hsp40, Hsp70-8 and h2afx.

Among the selected biomarkers, the abundance of the m-calpain measured in GB and TB muscles enters the model with a positive coefficient in the prediction of the tenderness of the two muscles (VL and SM) and with a high effect. That highlights a major role of this protein in tenderness independent of the type of muscle. Meanwhile, the role of μ-calpain seems to be more muscle type-dependent.

Calpains (CAPN) are a large family of intracellular Ca^{2+}-dependent cysteine-neutral proteases. The CAPN system is one of the endogenous proteolysis systems, and it plays a major role in meat tenderization [23]. Calpain's actions are mainly due to having two major isoforms: μ-calpain and m-calpain. These two isoforms require different amounts of calcium in vitro, and their names suggest that they help promote the microscopic and milli-molar concentrations of intracellular Ca^{2+} [24]. Among CAPN enzymes responsible for meat tenderization, μ-calpain, which is encoded by the CAPN1 gene, plays a predominant role [25]. As far as we know, calpains can cleave limited myofibrillar proteins such as titin, desmin and vinculin, and contribute to the improvement of tenderness, whereas, high levels of calpastatin are related to decreased proteolysis and increased meat toughness [26,27]. Furthermore, both proteases were found to interact with several proteins belonging to different pathways: homeostasis, structure, glucose metabolism, heat stress, mitochondria, apoptosis. The different implication of calpains in meat tenderness observed in the present study could be explained by a different role of each calpain in tenderization. In accordance with this hypothesis, it was shown in lamb m. *Biceps femoris* that the autolysis of μ-calpain and loss of most of the activity occur within 7 days post-mortem whereas m-calpain activity did not decrease up to 56 days post-mortem. Moreover, very little post-mortem proteolysis is observed in the muscles of μ-calpain knockout mice suggesting its predominant effect on catalysis of proteins during postmortem aging.

Thus, we confirm that these proteases might be considered good biomarkers of beef tenderness, as already shown in the literature [16,28,29]. Moreover, we also confirmed, as previously shown by Guillemin et al. [16], that the contribution of proteolysis to tenderness might be different from one muscle to another. Indeed, these authors indicated that, in the LT muscle, m-calpain and μ-calpain are positively correlated with tenderness, whereas, in the ST muscle, tenderness appears to be positively correlated with μ-calpain, but negatively with m-calpain. These authors indicated that the inverse relation between tenderness and μ-calpain from one muscle to another is coherent with the inverse correlation between contractile and metabolic proteins and tenderness in ST and LT, and thus that muscular particularities might determine calpain contribution to tenderness. Thus, our results seem to confirm this hypothesis to the extent that the contribution of μ-calpain of TB in the construction of tenderness is positive in the VL and negative in the SM. Some studies analyzing a suppression of μ-calpain expression showed an increase in cell death through an activation of apoptotic caspases and an increase in Hsp27, Hsp70, and Hsp90 expressions. These data illustrate a link between the different biomarkers involved in tenderness prediction.

Heat Shock Proteins preserve cellular proteins against denaturation and possible loss of function [30]. A large set of Hsp have been associated with meat tenderness. For example, Guillemin et al. [28] proposed the ratio of Hsp70/small Hsps as a good predictor of muscle tenderness in charolais young bulls. The results of Bernard et al. [31], on the same cattle, found Hsp40 (DNAJA1 gene) to be a positive marker of beef toughness in ST muscle. Hsp40s represent a large protein family that functions as a co-chaperone protein of Hsp70. These two families, Hsp70 and Hsp40, are often co-localized in the same subcellular compartment. Hsp40s function as ATP-independent chaperones that bind non-native polypeptides and protect cells from stress by preventing protein aggregation. The major function of Hsp40 proteins is to regulate ATP-dependent polypeptide binding by Hsp70

protein. Among the Hsp70 family, the Hsp70-8 is known to slow down the process of cellular death and to protect tissues against oxidative stress.

According to Picard et al. [29], there is an inverse relationship between tenderness and proteins from the small Hsp family according to muscle type and breed. It is thus not astonishing that the influence of these proteins on tenderness might be reversed when considering two different muscles such as SM and VL. In accordance with these authors, the results of Guillemin et al. [32] showed different relationships between proteins of the interactome of tenderness and tenderness in two different muscles such as ST and LT. The results of the present study confirm these observations.

The last protein involved in the prediction of VL and SM tenderness in the present study was h2afx. H2afx belongs to the histone h2a family involved in the cellular response to DNA damage, notably during oxidative stress [33]. It has recently been shown that h2ax specifically controls the recruitment of DNA repair proteins to the sites of DNA damage [34]. In eukaryotic cells, one of the first cellular response is the phosphorylation of h2afx within 1 to 3 min after damage. The number of H2AX phosphorylated molecules, γ-h2ax, increases linearly with the severity of the damage and is necessary for the recruitment of other factors to the sites of DNA damage [35]. During stresses, h2afx recruits metabolism enzymes to activate energy generation, enhance protein synthesis, and recruit anti-stress proteins as hspa1a to protect cells from degradation. H2afx was proposed as a potential biomarker of beef tenderness by Guillemin et al. [16], who showed that three proteins, h2afx, sumo4 and tp53, were situated at the crossroads between groups of proteins involved in tenderness, as metabolism, structure, cellular stress, oxidative stress, apoptosis, and calcium signaling proteins. As h2afx modulates different pathways and ensure genome integrity [36], it could have a crucial role in meat tenderness, which is confirmed by the results of the present study. However, why its abundance only as measured in GB muscle is explicative of tenderness of SM and VL muscles, and why the relationships with tenderness are inverted in these two muscles, remain to be understood.

In conclusion, this study made it possible to propose an original statistical methodology well adapted to this type of multi-block data set with several covariates (biomarkers) larger than the number n of individuals (carcasses). Moreover, seven proteins could be proposed as candidate biomarkers of tenderness, but the understanding of the biological mechanisms involved according to the muscle considered, remain to be deeply analyzed.

Author Contributions: Conceptualization: M.-P.E.-O., H.L., J.S., B.P.; Methodology: M.-P.E.-O., H.L., J.S.; Software: H.L. and J.S.; Validation: M.-P.E.-O., H.L., J.S., B.P.; Writing—Original draft preparation: M.-P.E.-O., H.L., J.S., B.P.; Writing—Review and editing: M.-P.E.-O., H.L., C.D., J.S., B.P.; Funding acquisition: B.P.

Funding: This research was funded by APIS-GENE, as a part of a French national project called Phenotend (AG-CC-V1).

Acknowledgments: The authors thank all the technicians of Inra and Idele who participated in muscle sampling, protein extraction, dot-blot analysis, sensory evaluation and measure of shear force in particular Nicole Dunoyer from INRA and Valérie Hardit from Idele.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Destefanis, G.; Brugiapaglia, A.; Barge, M.T.; Dal Molin, E. Relationship between beef consumer tenderness perception and Warner–Bratzler shear force. *Meat Sci.* **2008**, *78*, 153–156. [CrossRef]
2. Platter, W.J.; Tatum, J.D.; Belk, K.E.; Chapman, P.L.; Scanga, J.A.; Smith, G.C. Relationships of consumer sensory ratings, marbling score, and shear force value to consumer acceptance of beef strip loin steaks. *J. Anim. Sci.* **2003**, *81*, 2741–2750. [CrossRef] [PubMed]
3. Cassar-Malek, I.; Picard, B. Expression marker-based strategy to improve beef quality. *Sci. World J.* **2016**, *2016*, 2185323. [CrossRef] [PubMed]
4. Picard, B.; Lebret, B.; Cassar-Malek, I.; Liaubet, L.; Berri, C.; Le Bihan-Duval, E.; Hocquette, J.F.; Renand, G. Recent advances in omic technologies for meat quality management. *Meat sci.* **2015**, *109*, 18–26. [CrossRef] [PubMed]

5. Picard, B.; Gagaoua, M. Proteomic Investigations of Beef Tenderness. In *Proteomics in Food Science*; Elsevier: Amsterdam, The Netherlands, 2017; pp. 177–197.
6. Lorenzo, H.; Saracco, J.; Thiébaut, R. Supervised Learning for Multi-Block Incomplete Data. Available online: https://arxiv.org/abs/1901.04380 (accessed on 14 January 2019).
7. Lorenzo, H.; Razzaq, M.; Morange, P.-E.; Saracco, J.; Trégouët, D.-X.; Thiébaut, R. High-dimensional multi-block analysis of factors associated with thrombin generation potential. In Proceedings of the 32th International Symposium on Computer-Based Medical Systems, Córdoba, Spain, 5–7 June 2019.
8. Belew, J.B.; Brooks, J.C.; McKenna, D.R.; Savell, J.W. Warner–Bratzler shear evaluations of 40 bovine muscles. *Meat Sci.* **2003**, *64*, 507–512. [CrossRef]
9. Crosley, R.I.; Heinz, P.H.; De Bruyn, J.F. The relationship between beef tenderness and age classification of beef carcasses in South Africa. In Proceedings of the 8th Meat Symposium: Meat, Le Needs of the South Africaner Consumer, Pretoria, South Africa, 25 April 1995; pp. 57–66.
10. Warner, R.D.; Greenwood, P.L.; Pethick, D.W.; Ferguson, D.M. Genetic and environmental effects on meat quality. *Meat Sci.* **2010**, *86*, 171–183. [CrossRef]
11. Veiseth-Kent, E.; Pedersen, M.E.; Rønning, S.B.; Rødbotten, R. Can postmortem proteolysis explain tenderness differences in various bovine muscles? *Meat Sci.* **2018**, *137*, 114–122. [CrossRef]
12. Rhee, M.S.; Wheeler, T.L.; Shackelford, S.D.; Koohmaraie, M. Variation in palatability and biochemical traits within and among eleven beef muscles. *J. Anim. Sci.* **2004**, *82*, 534–550. [CrossRef]
13. Bouley, J.; Chambon, C.; Picard, B. Mapping of bovine skeletal muscle proteins using two-dimensional gel electrophoresis and mass spectrometry. *Proteomics* **2004**, *4*, 1811–1824. [CrossRef]
14. Bradford, M.M. A rapid and sensitive method for the quantitation of microgram quantities of protein utilizing the principle of protein-dye binding. *Anal. Biochem.* **1976**, *72*, 248–254. [CrossRef]
15. Guillemin, N.; Meunier, B.; Jurie, C.; Cassar-Malek, I.; Hocquette, J.F.; Leveziel, H.; Picard, B. Validation of a dot-blot quantitative technique for large scale analysis of beef tenderness biomarkers. *J. Physiol. Pharmacol.* **2009**, *60*, 91–97.
16. Guillemin, N.; Bonnet, M.; Jurie, C.; Picard, B. Functional analysis of beef tenderness. *J. Proteom.* **2011**, *75*, 352–365. [CrossRef]
17. Picard, B.; Barboiron, C.; Chadeyron, D.; Jurie, C. Protocol for high-resolution electrophoresis separation of myosin heavy chain isoforms in bovine skeletal muscle. *Electrophoresis* **2011**, *32*, 1804–1806. [CrossRef] [PubMed]
18. Talmadge, R.J.; Roy, R.R. Electrophoretic separation of rat skeletal muscle myosin heavy-chain isoforms. *J. Appl. Physiol.* **1993**, *75*, 2337–2340. [CrossRef]
19. Picard, B.; Barboiron, C.; Duris, M.P.; Gagniere, H.; Jurie, C.; Geay, Y. Electrophoretic separation of bovine muscle myosin heavy chain isoforms. *Meat Sci.* **1999**, *53*, 1–7. [CrossRef]
20. *NF ISO 13300. Sensory Analysis—General Guidance for the Staff of a Sensory Evaluation Laboratory*; International Organization for Standardization: Geneva, Switzerland, 2006.
21. Shackelford, S.D.; Wheeler, T.L.; Koohmaraie, M. Technical Note: Use of belt grill cookery and slice shear force for assessment of pork longissimus tenderness1. *J. Anim. Sci.* **2004**, *82*, 238–241. [CrossRef] [PubMed]
22. R: The R Project for Statistical Computing. Available online: https://www.r-project.org/ (accessed on 5 April 2019).
23. Lian, T.; Wang, L.; Liu, Y. A new insight into the role of calpains in post-mortem meat tenderization in domestic animals: A review. *Asian Aus. J. Anim. Sci.* **2013**, *26*, 443. [CrossRef]
24. Glass, J.D.; Culver, D.G.; Levey, A.I.; Nash, N.R. Very early activation of m-calpain in peripheral nerve during Wallerian degeneration. *J. Neurol. Sci.* **2002**, *196*, 9–20. [CrossRef]
25. Koohmaraie, M. Biochemical factors regulating the toughening and tenderization processes of meat. *Meat Sci.* **1996**, *43*, 193–201. [CrossRef]
26. Kemp, C.M.; Sensky, P.L.; Bardsley, R.G.; Buttery, P.J.; Parr, T. Tenderness—An enzymatic view. *Meat Sci.* **2010**, *84*, 248–256. [CrossRef]
27. Kent, M.P.; Spencer, M.J.; Koohmaraie, M. Postmortem proteolysis is reduced in transgenic mice overexpressing calpastatin. *J. Anim. Sci.* **2004**, *82*, 794–801. [CrossRef] [PubMed]
28. Guillemin, N.; Jurie, C.; Renand, G.; Hocquette, J.-F.; Micol, D.; Lepetit, J.; Picard, B. Different phenotypic and proteomic markers explain variability of beef tenderness across muscles. *Int. J. Biol.* **2012**, *4*, 26.

29. Picard, B.; Gagaoua, M.; Micol, D.; Cassar-Malek, I.; Hocquette, J.-F.; Terlouw, C.E. Inverse relationships between biomarkers and beef tenderness according to contractile and metabolic properties of the muscle. *J. Agric. Food Chem.* **2014**, *62*, 9808–9818. [CrossRef]
30. Goldberg, A.L. Protein degradation and protection against misfolded or damaged proteins. *Nature* **2003**, *426*, 895. [CrossRef] [PubMed]
31. Bernard, C.; Cassar-Malek, I.; Le Cunff, M.; Dubroeucq, H.; Renand, G.; Hocquette, J.-F. New indicators of beef sensory quality revealed by expression of specific genes. *J. Agric. Food Chem.* **2007**, *55*, 5229–5237. [CrossRef] [PubMed]
32. Guillemin, N.; Jurie, C.; Cassar-Malek, I.; Hocquette, J.-F.; Renand, G.; Picard, B. Variations in the abundance of 24 protein biomarkers of beef tenderness according to muscle and animal type. *Animal* **2011**, *5*, 885–894. [CrossRef] [PubMed]
33. Paull, T.T.; Rogakou, E.P.; Yamazaki, V.; Kirchgessner, C.U.; Gellert, M.; Bonner, W.M. A critical role for histone H2AX in recruitment of repair factors to nuclear foci after DNA damage. *Curr. Biol.* **2000**, *10*, 886–895. [CrossRef]
34. Stewart, G.S.; Wang, B.; Bignell, C.R..; Taylor, A.M.; Elledge, S.J. MDC1 is a mediator of the mammalian DNA damage checkpoint. *Nature* **2003**, *421*, 961. [CrossRef]
35. Kao, J.; Milano, M.T.; Javaheri, A.; Garofalo, M.C.; Chmura, S.J.; Weichselbaum, R.R.; Kron, S.J. γ-H2AX as a therapeutic target for improving the efficacy of radiation therapy. *Curr. Cancer Drug Targets* **2006**, *6*, 197–205. [CrossRef]
36. Dickey, J.S.; Redon, C.E.; Nakamura, A.J.; Baird, B.J.; Sedelnikova, O.A.; Bonner, W.M. H2AX: Functional roles and potential applications. *Chromosoma* **2009**, *118*, 683–692. [CrossRef]

CHAPITRE 5

CONCLUSION ET PERSPECTIVES

Dans le cas des données hétérogènes et/ou longitudinales, que l'on représente ici par des données multiblocs, présentant des données manquantes par blocs, une solution a été proposée pour répondre aux problèmes supervisés de grande dimension avec sélection de variables. L'approche se concentre sur une solution par réduction de dimension assimilable à une recherche de sous-espaces propres dans les "espaces de covariance". La philosophie de cette méthode est très proche de celle des méthodes de type PLS bien que l'approche proposée ne nécessite pas d'étape de déflation. C'est donc naturellement qu'elle y soit associée par son nom : ddsPLS.
La gestion des données manquantes a été pensée afin d'être intégrée au coeur de la méthode d'analyse supervisée. Comprenons ici qu'il a fallu créer un algorithme capable de gérer la présence de données manquantes dans les échantillons d'entraînement et de test. En effet, ces deux cas sont à dissocier puisque, dans le cas de l'échantillon d'entraînement, la donnée $\mathbf{Y}$ est accessible et guide l'information à inclure dans l'imputation alors qu'elle ne l'est pas dans un échantillon de test. L'algorithme ainsi défini porte le nom d'algorithme Koh-Lanta et se trouve être transposable à d'autres modèles de prédiction que le modèle ddsPLS seul, tant que les données sont manquantes par blocs.

Cette méthode a été comparée à des méthodes classiques d'imputation, à savoir l'imputation par la moyenne, NIPALS, soft-impute (Mazumder et collab., 2010) et missMDA (voir Josse et Husson, 2016). Ces méthodes ne permettent pas de prendre en compte l'aspect supervisé du problème mais leurs auteurs ont mis en ligne des packages qui permettent une prise en main rapide et efficace de leurs méthodes. La méthode ddsPLS montre une meilleure efficacité que ces dernières, efficacité qui est quantifiée au moyen du temps de calcul et de l'erreur en prédiction. Différents *scenarii*, associés à des modèles de simulation simples, permettent d'apprécier les effets de la proportion de données manquantes, du nombre d'individus, de la corrélation inter-blocs et de la corrélation intra-blocs.

D'autres méthodes, comme par exemple CMTF-WOPT, permettent de gérer l'aspect multibloc dans un contexte supervisé avec données manquantes mais la durée de cette thèse n'a pas permis de créer un code capable de générer des solutions CMTF-WOPT afin de les comparer aux autres méthodes. De plus cette dernière méthode masque les données manquantes au moyen d'une pondération dans l'optimisation du modèle.
Nous souhaiterions tout de même comparer ddsPLS à des méthodes telles que CMTF-WOPT. Il serait notamment intéressant de faire cette comparaison en estimant la qualité d'imputation de chacune des méthodes, bien que ddsPLS ne soit adaptée qu'à l'imputation marginale. Intuitive-

ment, CMTF-WOPT devrait être plus efficace que ddsPLS dans le contexte de la grande dimension, là où l'information marginale associée à la réponse est partagée avec un grand nombre de dimensions.

Le travail amorcé dans la section 2.2, permettant de comparer l'approche multibloc et son équivalent monobloc en absence de données manquantes semble être la continuité de ce travail. Il serait notamment intéressant de comprendre quelles structures de données valorisent plutôt l'approche multibloc ou l'approche monobloc associée.

Le changement de paramétrisation de λ vers L_0 semble impacter les choix de modèles. Lorsque les corrélations ne sont pas stables par sous-échantillonnage, la paramétrisation λ ne semble plus devoir être valorisée, bien que chacun des modèles sous-échantillonnés soient généralisables, on pense ici au comportement en validation croisée. L'hypothèse de travail porte sur le comportement de la méthode ddsPLS pour n faible, lorsque les individus extrêmes sont plus probablement présents, ce qui induit une instabilité de la sélection des variables en cas de sous-échantillonnage. Sur des échantillons à n faible, la paramétrisation λ devrait alors être plus instable en sélection que la paramétrisation L_0, ce que nous aimerions étudier dans le futur.

Ce travail a été valorisé au travers d'un package disponible sur le CRAN à l'adresse `https://CRAN.R-project.org/package=ddsPLS` et nommé 'ddsPLS', comme la méthode. Ce package a été codé de façon à permettre des calculs rapides, au moyen de fonctions codées en **C++** et interprétées grâce au package 'Rcpp' (Eddelbuettel et François, 2011). De plus, puisque les calculs de validation croisée sont au coeur de la méthode, il a été choisi de rendre possible la parallélisation de ces calculs grâce aux packages 'doParallel' et 'foreach' de **R**. Des fonctions de visualisations ont été proposées qui permettent d'observer les différents poids par composantes mais aussi les données sous forme de heatmaps ou de matrices de corrélations. Les packages 'corrplot' et 'RColorBrewer' ont permis ces résultats. Les fonctions `lda` et `logit` du package 'MASS' (Venables et Ripley, 2002) ont permis de gérer les cas de problèmes de classification.

Différentes applications ont permis d'améliorer les fonctionnalités du package mais aussi de tester son bon fonctionnement. Les données associées à ces applications admettent des structures variées, certaines avec peu d'individus (10 ou 20), d'autres avec un grand nombre d'individus ($\approx$ 700). La proportion de données manquantes va de 0% à presque 70%. Le nombre de variables par bloc varie de un à plusieurs milliers. Parmi les améliorations de la méthode initiées par l'analyse de ces données, on peut citer deux points importants. Tout d'abord l'initialisation des données manquantes, qui se faisait alors à la moyenne, s'est révélée être non judicieuse, une initialisation marginale par bloc des données manquantes est maintenant utilisée. Le coefficient de régularisation a lui-même été modifié pour permettre une paramétrisation plus intuitive. Initialement, il correspondait au niveau minimum absolu de corrélation entre les variables de prédiction et à prédire via l'opération de seuillage doux, soit un réel entre 0 et 1. Il correspond maintenant au nombre maximum de variables à inclure dans le modèle, c'est donc un entier compris entre 0 et le nombre total de covariables dans le modèle. L'intuition pratique derrière ce dernier aspect est de limiter l'impact d'un individu aberrant dans le cas des échantillons avec n petit. En effet, la corrélation est très sensible à ce genre d'individus et dans le cas de la validation croisée, une variable peut très bien être sortie du modèle du fait de ce genre de fluctuations. Cette nouvelle paramétrisation semble corriger ce défaut. Si l'on suppose les covariables toujours ordonnées de la même manière, en terme de corrélations aux variables à prédire,

un individu aberrant ne peut alors pas avoir pour effet de supprimer une variable dans certains des modèles de validation croisée. Aucun travail théorique ou de simulation n'a été réalisé sur le sujet mais ce serait être un futur sujet de recherche.

Perspectives

Les qualités de sélection de ddsPLS n'ont été évaluées que sur des critères de prédiction. L'application de ddsPLS à un jeu de données réelles, lui-même étudié par Clemmensen et collab. (2011), et présent en section 2.1, montre les premières qualités de l'approche en terme de sélection. Pourtant cette analyse s'est faite en absence de données manquantes et dans un contexte monobloc. Une future étude, qu'elle soit théorique ou par simulations, pourrait nous amener à estimer les qualités de la méthode en terme de sélection. Pour l'instant le seuillage doux a été évalué mais il pourrait être intéressant de comparer différents seuillages dans cet objectif, le seuillage doux ayant été choisi initialement pour les qualités démontrées en terme de reconstruction, notamment par Deshpande et Montanari (2016).

Parmi les challenges posés par de nouvelles données, la présence de données manquantes dans la partie réponse nous fait imaginer une adaptation semi-supervisée de l'algorithme ddsPLS. Ce cas de figure s'est par exemple présenté dans le jeu de données issu de la cohorte MARTHA et présenté en section 4.1. La présence de données manquantes dans la réponse n'est pas un problème isolé et oblige bien souvent l'analyste à mettre un grand nombre de données de côté, entraînant des problèmes éventuels sur un biais des estimateurs et assurément sur un défaut de puissance dans le cas de données issus d'échantillons de taille faible. Nous imaginons réutiliser un modèle typé-EM afin d'alterner entre estimations des modèles et des données manquantes dans la réponse, ajoutant ainsi une étape à l'algorithme Koh-Lanta. Ce genre de choix risquant d'introduire des risques de non convergence ce qui a été observé dans d'autres algorithmes "Typés-EM".

Le modèle de régression construit par ddsPLS ne permet pas de rendre compte de liens existants entre les blocs de variables puisque l'approche est purement multibloc. Il serait intéressant d'améliorer la méthode afin de pouvoir introduire directement les notions de blocs de données répétées ou de données multivoies. Une forme de hiérarchie dans les relations entre les blocs doit être imaginée, comme cela a été décrit par les approches structurelles telles que la méthodologie PLS-PM introduite par (Lohmöller, 2013) et adapté au contexte régularisé par Tenenhaus et Tenenhaus (2011); Tenenhaus et collab. (2014).

De façon plus générale, bien que les méthodes basées sur des grands n comme l'approche par renforcement soient très attractives, elles ne sont pas et ne sauraient être transposables à des problématiques de grande dimension, à n réduit.
Il est souvent dit que l'apprentissage statistique a pris un tournant au début des années 2010 avec l'avènement des données en ligne, il semblerait que ce genre de problématique assure un avenir radieux aux méthodes linéaires de fouille de données moins gourmandes en terme de nombre de paramètres. Je dirais donc que les années 2010 ont vu se scinder en deux les perspectives en terme de recherche et d'applications de l'apprentissage statistique. Les méthodes à composantes comme l'ACP où les méthodes linéaires pénalisées (autres que le Lasso qui souffre toujours d'un grand p pour un petit n) ainsi appréciées pour leurs simplicités algorithmique, d'implémentation

et d'interprétation sont assurées de répondre pour encore un bon moment aux problématiques de données sensibles.

ACRONYMES

kernelPCA ACP à noyaux – ou kernelPCA (Scholkopf et Smola, 2001). 8, 168

shrink Signifie rétrécissement en anglais. Correspond à la réduction de la norme de l'estimateur. Réalisé dans le spectre par les estimateurs Ridge et PCR et en norme $\mathcal{L}_1$ pour le Lasso, par exemple.. 29, 33, 34, 168

ACP Analyse en Composantes Principales – ou PCA pour Principal Component Analysis (Hotelling, 1933). 2, 8, 27, 29, 34, 36, 50, 55–57, 60, 167, 168, 180, 181

AFM Analyse Factorielle Multiple (Escofier et collab., 1984). 49, 56, 57, 60, 61, 168

AIC Critère d'Information d'Akaike – ou Akaike Information Criterion (Akaike, 1973). 39, 168

aLasso Lasso adaptatif (Zou, 2006). 35, 168

ALS-SI Imputation Simple par Alternance – ou Alternate Simple-Imputation (Tomasi et Bro, 2006). 57, 168

BIC Critère d'Information Bayésienne – ou Bayesian Information Criterion (Schwarz et collab., 1978). 39, 168

BLUE Meilleur Estimateur Linéaire sans Biais – ou Best Linear Unbiased Estimator (Pearson, 1901). 18, 168

CBMS Computer-Based Medical Systems. 168

CCA Analyse Canonique des Corrélations – ou Canonical Correlation Analysis (Hotelling, 1936). 48, 168

CMTF Factorisation Tensorielle pour une Collection de Matrices – ou Collective Matrix Tensor Factorization (Acar et collab., 2011b, 2013). 51, 57, 58, 60, 168

CMTF-ALS Factorisation ALternée Tensorielle pour une Collection de Matrices – ou Alternate Collective Matrix Tensor Factorization (Acar et collab., 2013). 168

CMTF-OPT Factorisation Tensorielle Optimale pour une Collection de Matrices – ou Optimal Collective Matrix Tensor Factorization (Acar et collab., 2013). 168

CMTF-WOPT Factorisation Tensorielle Optimale Pondérée pour une Collection de Matrices – ou Weighted Optimal Collective Matrix Tensor Factorization (Acar et collab., 2013). 57, 60, 61, 165, 166, 168

CP-WOPT PARAFAC Optimale Pondérée – ou Weighted Optimal PARAFAC (Acar et collab., 2011b). 57, 60, 168

CRAN Comprehensive R Archival Network Miscellaneous (R Core Team, 2019). 3, 33, 55, 168

CT Seuillage de Covariance – ou Covariance Thresholding. 36, 168

DCCA Analyse Canonique Profonde des Corrélations – ou Deep Canonical Correlation Analysis. 168

ddsPLS *data-driven sparse PLS* (Lorenzo et collab., 2019b). vii, 3, 61–63, 109–111, 113, 115, 137, 165–168, 196

dGN Gauss-Newton amorti – ou damped Gauss-Newton . 168

DT Seuillage de Diagonale – ou Diagonal Thresholding (Johnstone et Lu, 2009). 36, 168

Elastic-net Réseau Élastique – ou Elastic Network, méthode de régularisation (Zou et Hastie, 2005). 26, 60, 61, 168

EM Espérance-Maximisation – ou Expectation-Maximization (Dempster et collab., 1977). 3, 54–58, 61, 167, 168, 194, 196

GCV Validation-Croisée Généralisée (Golub et collab., 1979). 43, 168

gLasso group-Lasso (Simon et collab., 2013). 26, 27, 48, 168

gPLS PLS groupée – ou group PLS (Liquet et collab., 2015). 33, 48, 60, 168

INDAFAC paraFAC pour Données INcomplètes – ou INcomplete DAta paraFAC (Tomasi et Bro, 2005). 57, 60, 168

iterative PCA itérative PCA (Josse et Husson, 2012). 56, 168

LARS Régression à Corrélations Minimales Hiérarchique – ou Least Angle Regression Stagewise (Jolliffe et collab., 2003). 22, 23, 168

Lasso Least Absolute Shrinkage and Selection Operator (Tibshirani, 1996). 2, 8, 16, 18, 21–27, 33–36, 48, 60, 61, 167–169, 173, 186

LBO Leave-Bar-Out (Louwerse et collab., 1999). 58, 168

LOO Leave-One-Out (Stone, 1974; Allen, 1974; Geisser, 1975). 40, 168

MBPLS Moindres Carrés Partiels Multi Blocs – ou Multiblocks pls (Wangen et Kowalski, 1989). 47, 48, 60, 168

MCO Critère des Moindres Carrés Ordinaires – ou OLS pour Ordinary Least Squares criterion. 1, 16–19, 21–23, 27–29, 33, 34, 36, 38, 46, 60, 168, 183, 185, 186

MLE Estimateur du Maximum de Vraisemblance MLE. 168, 191

MMMF Factorisation de Matrice à Marge Maximale – ou Maximum-Margin Matrix Factorization (Hastie et collab., 2015). 55, 168

MSE Erreur Quadratique Moyenne – ou Mean Squared Error. 168

multibloc Données formées par plusieurs groupes de variables ne partagean pas forcemment la même structure ni la même nature. 2, 3, 8, 33, 47–49, 52, 57, 61, 62, 165–168, 173

multivoie Données formées par plusieurs modalités d'un même ensemble de variables. 2, 4, 47, 49, 50, 52, 57, 58, 61, 167, 168

NIPALS PLS Itérative Non linéaire – ou Nonlinear Iterative Partial Least Squares (Wold et collab., 1989). 32, 47–51, 55, 56, 60, 165, 168, 180

NPLS PLS à N voies (Bro, 1996). 50, 51, 60, 168

PARAFAC/CP Analyse Factorielle PARAallèle – ou PARAllel FACtor analysis, aussi appelée CandecomP (CP) (Harshman, 1970). 50–52, 57, 60, 168

PCR Régression sur Composantes Principales (Hotelling, 1957). 27–29, 33, 34, 60, 168, 169

PLASM Probing Least Absolute Squares Modelling (Bakin, 1999). 26, 48, 60, 168

PLS Moindres Carrés Partiels – ou Partial Least Squares (Wold et collab., 1983). 2, 29–33, 47, 48, 56, 60, 63, 165, 168

PLS-DA Analyse Discriminante par PLS – ou PLS-DA (Barker et Rayens, 2003). 63, 168

PLS-PM PLS Path Modeling (Lohmöller, 2013). 60, 167, 168

PPCA Probabilist ACP (PPCA) (Caussinus, 1986). 168

PRESS Somme des Carrés Résiduels de l'Erreur en Prédiction – ou Predictive Residual Error Sum of Squares. 58, 168

principe ERM Principe de Minimisation du Risque Empirique – ou Empirical Risk Minimization inductive principle (Vapnik et Chervonenkis, 2015). 9, 39, 168

R Langage de programmation (R Core Team, 2019). 33, 168

RDA Analyse des Redondances – ou Redundancy Analysis (van den Wollenberg, 1977). 48, 60, 168

RGCCA Analyse Canonique des Corrélations Régularisée Généralisée – ou Regularized Generalized Canonical Correlation Analysis (Tenenhaus et Tenenhaus, 2011). 48, 60, 168

Ridge Principe de régularisation limitant à un espace atteignable par les paramètres du problème à une hyperboule de norme 2 et de rayon le paramètre λ (voir Hoerl et Kennard, 1970). 1, 2, 8, 18, 20–22, 24–26, 29, 33, 34, 43, 46, 60, 168, 169, 173, 184, 185

RMSE Erreur Quadratique Moyenne sous Racine– ou Root Mean Squared Error. 168

SCAD Smoothly Clipped Absolute Deviation Penalty (Rothman et collab., 2009). 35, 168

SCE Somme des Carrés Expliqués. 37, 168

SCoTLASS Simplification de Méthode à Composantes par Lasso – ou Simplified Component Technique-Lasso (Jolliffe et collab., 2003). 33, 168

SCR Somme des Carrés des Résidus. 37, 38, 168

SCT Somme des Carrés Totale. 37, 168

SDP Programmation Positive – ou Semi-Definite Programming (d'Aspremont et collab., 2005). 36, 168

SE Erreur Quadratique – ou Squared Error. 168

SGCCA Analyse Canonique des Corrélations Généralisée Parcimonieuse – ou Sparse Generalized Canonical Correlation Analysis (Tenenhaus et collab., 2014). 48, 60, 168

sgLasso sparse group-Lasso (Simon et collab., 2013). 27, 48, 60, 168

sgPLS PLS parcimonieuse groupée – ou sparse group PLS (Liquet et collab., 2015). 33, 48, 60, 168

sparseACP Analyse en Composantes Principales Parcimonieuse – ou sparse PCA (Johnstone et Lu, 2004). 34, 36, 60, 168

SPCA Sensible Principal Component Analysis (SPCA). 168

sPLS PLS parcimonieuse – ou sparse PLS (Lê Cao et collab., 2008). 30, 33, 168

SRM Minimisation du Risque Structurel – ou Structural Risk Minimization (Vapnik et Chervonenkis, 2015). 10, 168

STATIS Structuration de Tableaux À Trois Indices de la Statistique (L'Hermier des Plantes, 1976). 49, 61, 168

SVD Décomposition en Valeurs Singulières – ou Singular Value Decomposition (Pearson, 1901). 17, 36, 109, 168, 178

SVM Machine à Vecteur de Support – ou Support Vector Machine (Cortes et Vapnik, 1995). 168

Tucker3 Three-mode principal component model (Tucker3) (Kroonenberg, 1983). 50, 58, 60, 168

VC Dimension de Vapnik-Chervonenkis. 168

LISTE DES FIGURES

Annexes

ANNEXE A

NOTIONS MATHÉMATIQUES

A.1 Multiplicateur de Lagrange

Théorème A.1 (Multiplicateurs de de $\mathcal{L}$agrange, cas de contraintes d'égalité)**.** *Soient f et g_1,..,g_p des fonctions de classe C^1 sur un ouvert $\mathcal{U}$ de $\mathbb{R}^n$, à valeurs dans $\mathbb{R}$ et $\mathcal{X}$ l'ensemble défini par*

$$\mathcal{X} \quad = \quad \left\{x \in \mathcal{U} | g_1(x) = \ldots = g_p(x) = 0\right\}.$$

Si la restriction de f à $\mathcal{X}$ admet un extremum local en a et si les différentielles $dg_1(a), \ldots, dg_p(a)$ sont des formes linéaires indépendantes, alors il existe des réels uniques $\lambda_1, \ldots, \lambda_p$ tels que

$$df(a) \quad = \quad \sum_{i=1}^{p} \lambda_i dg_i(a).$$

Et alors a est un optimum de f dans $\mathcal{X}$.

Remarque A.1. *Les coefficients λ_i sont appelés les multiplicateurs de $\mathcal{L}$agrange et on appelle "lagrangien" la fonction*

$$\mathcal{L}(x, \lambda_1, \ldots, \lambda_p) \quad = \quad f(x) \quad + \quad \sum_{i=1}^{p} \lambda_i g_i(x).$$

Remarque A.2. *Dans le Théorème précédent on dit que les contraintes sont actives lorsque le point critique considéré est bien dans $\mathcal{X}$.*

A.2 Pseudo-inverses

On remarque que pour toute matrice inversible $\mathbf{A}$, la matrice $\mathbf{A}^{-1}$ est aussi pseudo-inverse de $\mathbf{A}$. Il est possible de montrer que l'ensemble des pseudo-inverses de $\mathbf{A}$ est infini.

Définition A.1 (Pseudo-inverse)**.** *Soit une matrice réelle $\mathbf{A} \in \mathbb{R}^{n \times p}$ quelconque, on appelle matrice pseudo-inverse de $\mathbf{A}$ toute matrice $\mathbf{B} \in \mathbb{R}^{n \times n}$ qui vérifie*

$$\mathbf{ABA} = \mathbf{A}.$$

Cette définition permet aussi d'introduire la pseudo-inverse de Moore-Penrose, qui est un cas particulier de pseudo-inverse

Définition A.2. ***Pseudo-inverse de Moore-Penrose (Penrose, 1956).*** *Soit une matrice* $\mathbf{A} \in \mathbb{R}^{n\times p}$ *de rang* R_A *quelconque. Alors la pseudo inverse de Moore-Penrose de* $\mathbf{A}$*, notée* $\mathbf{A}^+$ *vérifie les quatre propriétés*

1. $\mathbf{A}\mathbf{A}^+\mathbf{A} = \mathbf{A}$,
2. $\mathbf{A}^+\mathbf{A}\mathbf{A}^+ = \mathbf{A}^+$,
3. $\mathbf{A}^+\mathbf{A}$ *est symétrique,*
4. $\mathbf{A}\mathbf{A}^+$ *est symétrique.*

Il est possible de démontrer que pour toute matrice réelle, sa pseudo-inverse de Moore-Penrose existe et est unique (voir par exemple Rao et collab., 1972).

A.3 Décomposition SVD

Afin de déterminer la forme de la pseudo-inverse de Moore-Penrose, il convient d'introduire le principe de *valeur singulière*.

Définition A.3. ***Valeurs singulières*** *Soit* $\mathbf{A} \in \mathbb{R}^{n\times p}$ *quelconque de rang* R_A*, on appelle valeurs singulières de* $\mathbf{A}$ *les* R_A *réels* $\sigma_1 > \sigma_2 > \ldots > \sigma_{R_A} > 0$ *qui sont les racines carrés des valeurs propres de la matrice* $\mathbf{A}^T\mathbf{A}$.

La matrice $\mathbf{A}^T\mathbf{A}$ est symétrique réelle, elle est donc diagonalisable à valeurs propres positives ou nulles, par application du Théorème spectral. On peut démontrer facilement que le spectre de $\mathbf{A}^T\mathbf{A}$ est égal au spectre de $\mathbf{A}\mathbf{A}^T$. Le Théorème spectral permet la formalisation de la décomposition en valeurs singulières de toute matrice via le Théorème suivant.

Théorème A.2. ***Décomposition en valeurs singulières (Singular Value Decomposition – SVD)*** *Soit une matrice réelle* $\mathbf{A} \in \mathbb{R}^{n\times p}$ *quelconque de rang* R_A*, on appelle décomposition en éléments propres de* $\mathbf{A}$ *le triplet* $(\mathbf{U}, \mathbf{D}, \mathbf{V}) \in \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times p} \times \mathbb{R}^{p\times p}$ *telles que* $\mathbf{U}$ *et* $\mathbf{V}$ *soient à colonnes orthogonales,* $\mathbf{D}$ *soit une matrice dont les* R_A *premiers coefficients diagonaux sont strictement positifs et les autres sont tous nuls. Et ce triplet vérifie*

$$\mathbf{X} \quad = \quad \mathbf{U}\mathbf{D}\mathbf{V}^T. \tag{A.1}$$

Corollary A.1. ***SVD et pseudo-inverse de Moore-Penrose*** *Soit une matrice* $\mathbf{A} \in \mathbb{R}^{n\times p}$ *quelconque de rang* R_A *et sa décomposition SVD* $(\mathbf{U}, \mathbf{D}, \mathbf{V}) \in \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times p} \times \mathbb{R}^{p\times p}$ *dont les caractéristiques sont guidées par le Théorème A.2, alors la pseudo-inverse de Moore-Penrose de* $\mathbf{A}$ *est la matrice* $\mathbf{X}^+$ *qui s'écrit*

$$\mathbf{X}^+ \quad = \quad \mathbf{V}\mathbf{D}^{\sim 1}\mathbf{U}^T, \tag{A.2}$$

où $\mathbf{D}^{\sim 1}$ *est la matrice de dimension* $p \times n$ *dont les* R_A *premiers coefficients diagonaux sont les inverses respectifs des* R_A *premiers termes de la diagonale de* $\mathbf{D}$ *et les autres coefficients sont tous nuls.*

Démonstration. Par vérification immédiate des 4 propriétés de la Définition A.2. □

Théorème A.3 (Solution généralisée du problème linéaire)**.** *Soit le problème inverse de régression linéaire ayant pour inconnue* $\mathbf{X}$ *qui s'écrit*

$$\arg\min_{\mathbf{X}} \|\mathbf{B} - \mathbf{A}\mathbf{X}\|_F^2, \tag{A.3}$$

où $\mathbf{A} \in \mathbb{R}^{n\times p}$, $\mathbf{B} \in \mathbb{R}^{n\times q}$. *Alors la solution de ce problème de norme de frobenius minimale est*

$$\mathbf{X} = \mathbf{A}^{+}\mathbf{B}$$

où $\mathbf{A}^{+}$ *est la pseudo-inverse de Moore-Penrose de* $\mathbf{A}$.

Démonstration. Soit le problème d'optimisation sous contrainte

$$\begin{array}{ll} \arg\min_{\mathbf{X}} & \|\mathbf{X}\|_F^2 \\ \text{sous contrainte} & \mathbf{A}\mathbf{X}_{\bullet,j} - \mathbf{B}_{\bullet,j} = 0,\ \forall j \in [\![1,q]\!] \end{array} \tag{A.4}$$

où $\mathbf{X}_{\bullet,j}$ est la $j^{\text{ème}}$ colonne de $\mathbf{X}$ tout comme $\mathbf{B}_{\bullet,j}$ pour $\mathbf{B}$. C'est un problème d'optimisation sous contraintes d'égalité où le critère et les contraintes sont des fonctions de classe C^1. On définit alors le $\mathcal{L}$agrangien de ce problème

$$\begin{array}{rcl} \mathcal{L}\left(\mathbf{X}, \boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_q\right) & = & \|\mathbf{X}\|_F^2 + 2\sum_{j=1}^{q} \boldsymbol{\lambda}_j^T \left(\mathbf{A}\mathbf{X}_{\bullet,j} - \mathbf{B}_{\bullet,j}\right), \\ & = & \sum_{j=1}^{q} \|\mathbf{X}_{\bullet,j}\|^2 + 2\sum_{j=1}^{q} \boldsymbol{\lambda}_j^T \left(\mathbf{A}\mathbf{X}_{\bullet,j} - \mathbf{B}_{\bullet,j}\right), \\ & = & \sum_{j=1}^{q} \left[\|\mathbf{X}_{\bullet,j}\|^2 + 2\boldsymbol{\lambda}_j^T \left(\mathbf{A}\mathbf{X}_{\bullet,j} - \mathbf{B}_{\bullet,j}\right)\right], \end{array}$$

qui est en fait la somme de q problèmes indépendemment résolus en $\mathbf{X}_{\bullet,j}$. Nous allons donc résoudre le problème suivant

$$\begin{array}{ll} \arg\min_{\mathbf{x}} & \|\mathbf{x}\|^2 \\ \text{sous contrainte} & \mathbf{a}_i\mathbf{x} - b_i = 0,\ \forall i \in [\![1,n]\!] \end{array} \tag{A.5}$$

où $\mathbf{a}_{i,\bullet}$ est la $i^{\text{ème}}$ ligne de $\mathbf{A}$ et b_i est le $i^{\text{ème}}$ élément de $\mathbf{b}$, une des colonnes de $\mathbf{B}$ correspondant avec la colonne de $\mathbf{X}$ courante. Le $\mathcal{L}$agrangien de ce problème s'écrit

$$\mathcal{L}_0\left(\mathbf{x}, \boldsymbol{\lambda}\right) = \|\mathbf{x}\|^2 + 2\sum_{i=1}^{n} \lambda_i \left(\mathbf{a}_i\mathbf{x} - b_i\right),$$

avec $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_n)^T$ et qui, après différentiation et annulation des différentielles, conduit au système

$$\left\{ \begin{array}{rcl} \mathbf{x} & = & -\mathbf{A}^T\lambda/2, \\ \mathbf{A}\mathbf{x} & = & \mathbf{b}. \end{array} \right.$$

On applique dès lors le Théorème A.2 et son Corollaire A.1 à la matrice $\mathbf{A}$ pour déterminer $\left(\mathbf{U}, \mathbf{D}, \mathbf{D}^{\sim 1}, \mathbf{V}\right)$ vérifiant les relations (A.1) et (A.2). On note alors

$$\left(\mathbf{x}_0, \boldsymbol{\lambda}_0\right) = \left(\mathbf{V}\mathbf{D}^{\sim 1}\mathbf{U}^T\mathbf{b}, -2\mathbf{U}\mathbf{D}^{\sim 1^T}\mathbf{D}^{\sim 1}\mathbf{U}^T\mathbf{b}\right).$$

On peut remarquer que $(\mathbf{x}_0, \boldsymbol{\lambda}_0)$ vérifie le système précédent. Les contraintes sont donc actives en $(\mathbf{x}_0, \boldsymbol{\lambda}_0)$.
Puisque les contraintes $\forall i \in [\![1,n]\!]$, $g_i(\mathbf{x}) = \mathbf{a}_i\mathbf{x} - b_i$ ont pour différentielles des formes linéaires indépendantes, car ce sont alors des constantes. Nous nous retrouvons dans le cadre d'application du Théorème A.1, dit Théorème du multiplicateur de $\mathcal{L}$agrange, puisque de plus le carré du produit scalaire est de classe C^1 et les contraintes, comme formes linéaires, sont aussi de classe

C^1. Ainsi par application du dit Théorème, le problème (A.5) est résolu en $(\mathbf{x}_0, \boldsymbol{\lambda}_0)$ de façon unique.

En utilisant la remarque sur l'indépendance des problèmes, il vient que

$$\mathbf{X} = \mathbf{V}\mathbf{D}^{\sim 1}\mathbf{U}^T \left(\mathbf{B}_{\bullet,1}, \ldots, \mathbf{B}_{\bullet,q}\right) = \mathbf{V}\mathbf{D}^{\sim 1}\mathbf{U}^T\mathbf{B} = \mathbf{A}^+\mathbf{B}$$

est l'unique solution du problème (A.3), ce qui conclut la démonstration. □

A.4 Décomposition en ACP

Définition A.4 (ACP)**.** *Soit une matrice réelle* $\mathbf{X}$ *de dimension* $n \times p$ *et à colonnes centrées (la somme de tous les éléments d'une colonne vaut 0) alors on définit la décomposition en composantes principales d'ordre* R *le duet* $(\mathbf{U}, \mathbf{d})$ *où* $\mathbf{U}$ *est une matrice de dimension* $p \times R$ *à colonnes orthogonales et* $\mathbf{d}$ *est un vecteur de dimension* R*.* $\forall r \in [\![1, R]\!]$*, on note* $\mathbf{u}_r$*, la* $r^{ième}$ *colonne de* $\mathbf{U}$ *et* d_r *chaque élément de* $\mathbf{d}$ *avec*

$$d_1 \geqslant d_2 \geqslant \ldots \geqslant d_R \geqslant 0.$$

De plus, chaque poids $\mathbf{u}_r$ *est*

- *orthogonal à tous les autres poids,*
- *vérifie*

$$\mathbf{u}_r = \begin{array}{cc} \arg\max_{\mathbf{u}} & \|\mathbf{X}\mathbf{u}\|_2^2 \\ \textit{sous contrainte} & \mathbf{u}^T\mathbf{u} = 1 \end{array}, \tag{A.6}$$

- $d_r = \|\mathbf{X}\mathbf{u}_r\|_2^2$

Théorème A.4 (ACP et décomposition SVD)**.** *Soit une matrice réelle* $\mathbf{X}$ *de dimension* $n \times p$ *et à colonnes centrées et* $(\mathbf{U}, \mathbf{d})$ *l'ACP de* $\mathbf{X}$*. Alors chaque axe* $r \in [\![1, R]\!]$ *est le* $r^{ième}$ *vecteur singulier droit de* $\mathbf{X}$ *et* $d_r = \sigma_r^2$ *est le* $r^{ième}$ *élément diagonal de* $\mathbf{D}$ *où* σ_r *est la* $r^{ième}$ *valeur singulière de* $\mathbf{X}$*.*

Remarque A.3 (Choix de démonstration)**.** *La démonstration par récurrence qui est présentée ici est aussi l'algorithme de résolution de l'ACP originel et s'appelle NIPALS. La recherche des sous-espaces propres est détaillée dans l'annexe C.*

Démonstration. Dans cette démonstration et pour tout $r \in [\![1, R]\!]$ nous noterons

$$\pi_r = \left(\mathbb{I} - \mathbf{u}_r\mathbf{u}_r^T\right),$$

qui est une matrice de projection orthogonale et qui a pour noyau le sous-espace porté par $\mathbf{u}_r$.
Nous allons procéder par récurrence.
Hypothèse. $\forall r \in [\![1, R]\!]$, $\pi_{1,r} = \prod_{s=1}^{r} \pi_s$ est une matrice de projection orthogonale avec pour noyau $(\mathbf{u}_1, \ldots, \mathbf{u}_r)$. De plus, $\mathbf{u}_r$ qui résout l'équation (A.6) en posant $\mathbf{X} \leftarrow \mathbf{X}\pi_{1,r}$ est le vecteur singulier droit de rang r associé à la valeur singulière σ_r^2 de $\mathbf{X}$.
Initialisation. Par résolution du problème (A.6), pour $r = 1$, qui vérifie les conditions d'applications du Théorème A.1 du multiplicateur de $\mathcal{L}$agrange pour une contrainte unique dont la différentielle est une forme linéaire et où le critère et la contrainte sont bien des fonctions de classe C^1. Le lagrangien s'écrit alors

$$\mathcal{L}(\mathbf{u}, \lambda) = \|\mathbf{X}\mathbf{u}\|_2^2 - \lambda\left(\|\mathbf{u}\|_2^2 - 1\right)$$

Par double différentiation, le problème se met sous la forme

$$\begin{cases} \mathbf{X}^T\mathbf{X}\mathbf{u} &= \lambda\mathbf{u} \\ \mathbf{u}^T\mathbf{u} &= 1 \end{cases},$$

et en notant le point critique $\left(\mathbf{u}_1, \sigma_1^2\right)$, le premier vecteur singulier droit de $\mathbf{X}$ et le carré de la valeur singulière associée et en injectant dans l'équation, on se retrouve avec une contrainte active et un système résolu. $\left(\mathbf{u}_1, \sigma_1^2\right)$ est donc la première composante principale de l'ACP et vérifie l'hypothèse. On définit alors une matrice $\mathbf{X}_1$ telle que

$$\mathbf{X}_1 = \mathbf{X}\pi_1,$$

c'est l'opération de *déflation*. La projection de cette matrice sur $\mathbf{u}_1$ est nulle car le noyau de π_1 est porté par $\mathbf{u}_1$. L'hypothèse est donc bien vérifiée au rang 1.
Propagation. Supposons que le résultat soit vrai jusqu'au rang $R-1$ où R est inférieur ou égal au rang de X, et que la suite de matrices $(\mathbf{X}_1, \ldots, \mathbf{X}_{R-1})$ ait été construite, alors il est possible de résoudre le problème (A.6) en considérant $\mathbf{X} \leftarrow \mathbf{X}_{R-1}$, soit le problème

$$\begin{array}{cc} \max_{\mathbf{u}} & \|\mathbf{X}_{R-1}\mathbf{u}\|_2^2 \\ \text{sous contrainte} & \mathbf{u}^T\mathbf{u} = 1 \end{array}, \tag{A.7}$$

il vient donc à résoudre le système

$$\begin{cases} \mathbf{X}_{R-1}^T\mathbf{X}_{R-1}\mathbf{u} &= \lambda\mathbf{u} \\ \mathbf{u}^T\mathbf{u} &= 1 \end{cases}. \tag{A.8}$$

En remarquant que $\mathbf{X}_{R-1} = \mathbf{X}\pi_{1,R-1}$, on teste le duet $\left(\mathbf{u}_R, \sigma_R^2\right)$ formé par le vecteur singulier gauche de rang r de la matrice $\mathbf{X}$ et le carré de la valeur singulière associée. Ainsi

$$\begin{aligned} \pi_{1,R-1}\mathbf{X}^T\mathbf{X}\pi_{1,R-1}\mathbf{u}_R &= \pi_{1,R-1}\mathbf{X}^T\mathbf{X}\mathbf{u}_R \\ &= \pi_{1,R-1}\left(\sigma_R^2\mathbf{u}_R\right) \\ &= \sigma_R^2\pi_{1,R-1}\mathbf{u}_R \\ &= \sigma_R^2\mathbf{u}_R \end{aligned}, \tag{A.9}$$

où nous avons utilisé que $\mathbf{u}_R$ est dans l'image de l'endomorphisme associé à la matrice de projection orthogonale $\pi_{1,R-1}$. Comme par définition $\mathbf{u}_R^T\mathbf{u}_R = 1$, il vient que le duet candidat active les contraintes et par application du Théorème A.1, la solution testée est la bonne et est unique.
Conclusion. La décomposition ACP de rang R est formée par, $r \in [\![1, R]\!]$, le $r^{\text{ième}}$ vecteur singulier droit de $\mathbf{X}$ où $d_r = \sigma_r^2$ est la $r^{\text{ième}}$ valeur singulière au carré de $\mathbf{X}$.
Ce qui permet de clore cette démonstration. □

ANNEXE B

CALCULS DE RISQUES QUADRATIQUES

On suppose travailler dans le cas d'une réponse univariée ($q = 1$) par simplification de l'écriture d'un raisonnement qui se généralise simplement. Dans le cas du modèle linéaire à bruit additif centré on note

$$Y = X\mathbf{b} + \epsilon,$$

avec $\mathbf{b}$ la matrice colonne à estimer et ϵ la variable aléatoire associée au bruit d'observation supposé centré et de variance σ_ϵ^2. Un jeu de données $\mathcal{D}_n$ représenté par les deux matrices $\mathbf{X}$ et $\mathbf{y}$ est utilisé pour l'entraînement avec l'erreur d'observation $\boldsymbol{\epsilon}$. On suppose de plus un nouvel échantillon d'un seul individu $(\mathbf{x}_1, y_1)$, où le premier terme est une matrice ligne et le second terme est un réel avec l'erreur d'observation ϵ_1. On peut donc noter

$$\begin{aligned} \mathbf{y} &= \mathbf{X}\mathbf{b} + \boldsymbol{\epsilon}, \\ y_1 &= \boldsymbol{x}_1\mathbf{b} + \epsilon_1. \end{aligned}$$

Les notations utilisées se rapprochent de celles utilisées dans la partie 1.2.

B.1 Minimisation des moindres carrés ordinaires

On suppose $\hat{\mathbf{b}}_n^{(MCO)}$ l'estimateur des MCO solution du problème $\arg\min_{\mathbf{b}} \|\mathbf{y} - \mathbf{X}\mathbf{b}\|^2$ qui, par application du Théorème A.3 s'écrit

$$\hat{\mathbf{b}}_n^{(MCO)} = \mathbf{X}^+\mathbf{y},$$

où $\mathbf{X}^+$ désigne la pseudo-inverse de Moore-Penrose de $\mathbf{X}$. On peut donc écrire, pour la nouvelle observation

$$\begin{aligned} f(\mathbf{x}_1) &= y_1 = \mathbf{x}_1\mathbf{b} + \epsilon_1 &&, & f^\star(\mathbf{x}_1) &= \mathbf{x}_1\mathbf{b}, \\ g^\star(\mathbf{x}_1) &= \mathbb{E}_n\left[\mathbf{x}_1\hat{\mathbf{b}}_n^{(MCO)}\right] &&, & \hat{f}_n(\mathbf{x}_1) &= \mathbf{x}_1\hat{\mathbf{b}}_n^{(MCO)}, \\ &= \mathbf{x}_1\mathbf{X}^+\mathbb{E}_n\left[\mathbf{X}\mathbf{b} + \boldsymbol{\epsilon}\right] &&, & &= \mathbf{x}_1\mathbf{X}^+\left(\mathbf{X}\mathbf{b} + \boldsymbol{\epsilon}\right), \\ &= \mathbf{x}_1\mathbf{X}^+\mathbf{X}\mathbf{b} &&, & &= \mathbf{x}_1\mathbf{b} + \mathbf{x}_1\mathbf{X}^+\boldsymbol{\epsilon} \\ &= \mathbf{x}_1\mathbf{b} &&. \end{aligned} \tag{B.1}$$

Le terme de variance associé au bruit pour cette individu et son espérance pour toutes les valeurs possibles pour $\mathbf{X}_1$ deviennent

$$\begin{array}{rcl} \left(f(\mathbf{x}_1) - f^\star(\mathbf{x}_1)\right)^2 & = & \epsilon_1^2 \\ \text{bruit}\left(\hat{\mathbf{b}}_n^{(MCO)}\right) & = & \mathbb{E}_{\mathbf{x}_1}\left[\mathbb{E}_y\left[\left(f(\mathbf{x}_1) - f^\star(\mathbf{x}_1)\right)^2\right]\right] \\ & = & \mathbb{E}_{\mathbf{x}_1}\left[\epsilon_1^2\right] = \sigma_\epsilon^2 \end{array}$$

Le terme de biais pour ce nouvel échantillon s'écrit alors, par lecture immédiate des équations (B.1)

$$\begin{array}{rcl} \left(f^\star(\mathbf{x}_1) - g^\star(\mathbf{x}_1)\right)^2 & = & 0 \\ \text{biais}^2\left(\hat{\mathbf{b}}_n^{(MCO)}\right) & = & \mathbb{E}_{\mathbf{x}_1}\left[\left(f^\star(\mathbf{x}_1) - g^\star(\mathbf{x}_1)\right)^2\right] \\ & = & 0 \end{array} \tag{B.2}$$

et on démontre ainsi que le biais du modèle linéaire est nul. Le terme de variance du modèle pour ce nouvel échantillon s'écrit alors

$$\begin{array}{rcl} \mathbb{E}_n\left[\left(\hat{f}_n(\mathbf{x}_1) - g^\star(\mathbf{x}_1)\right)^2\right] & = & \mathbb{E}_n\left[\left(\mathbf{x}_1\hat{\mathbf{b}}_n^{(MCO)} - \mathbb{E}\left[\mathbf{x}_1\hat{\mathbf{b}}_n^{(MCO)}\right]\right)^2\right] \\ & = & \mathbf{x}_1\mathbf{X}^+\mathbb{E}_n\left[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^T\right]\left(\mathbf{x}_1\mathbf{X}^+\right)^T \end{array} \quad = \quad \mathbb{E}_n\left[\left(\mathbf{x}_1\mathbf{X}^+\boldsymbol{\epsilon}\right)^2\right] \quad = \quad \sigma_\epsilon^2\mathbf{x}_1\mathbf{X}^+\left(\mathbf{X}^+\right)^T\mathbf{x}_1^T$$

et en sommant à l'ensemble des possibilités accessibles à $\mathbf{x}_1$, il vient

$$\begin{array}{rcl} \text{var}\left(\hat{\mathbf{b}}_n^{(MCO)}\right) & = & \mathbb{E}_{\mathbf{x}_1}\left[\mathbb{E}_n\left[\left(\hat{f}_n(\mathbf{x}_1) - g^\star(\mathbf{x}_1)\right)^2\right]\right] \\ & = & \sigma_\epsilon^2\text{Trace}\left(\mathbb{E}_{\mathbf{x}_1}\left[\mathbf{x}_1^T\mathbf{x}_1\right]\mathbf{X}^+\mathbf{X}^{+T}\right) \\ & = & \sigma_\epsilon^2\text{Trace}\left(\mathbf{X}^+\mathbf{X}^{+T}\right)\text{Trace}\left(\mathbb{E}_{\mathbf{x}_1}\left[\mathbf{x}_1^T\mathbf{x}_1\right]\right) \\ & = & \sigma_\epsilon^2\text{Trace}\left(\mathbf{X}^+\mathbf{X}^{+T}\right)p \end{array} \tag{B.3}$$

où nous avons utilisé l'inégalité de Cauchy-Schwarz afin de passer de la seconde à la troisième ligne. Or la structure de covariance de X est présente dans $\mathcal{D}_n$ et donc les matrices "$\mathbf{X}^+\mathbf{X}^{+T}$" et "$\mathbf{x}_1^T\mathbf{x}_1$" partagent les mêmes sous espaces propres. C'est donc un cas d'égalité de l'inégalité de Cauchy-Schwarz. Pour passer à la dernière ligne nous avons arbitrairement considéré que toutes les variables avec la même variance, égale à l'unité, ce qui n'est pas une hypothèse forte.

L'espérance du risque quadratique de l'estimateur des moindres carrés dans le cas d'un bruit additif centré de variance σ_ϵ^2 entraîné sur un jeu de données de n individus pour p covariables et $q = 1$ variable réponse s'écrit donc

$$\begin{array}{rcl} \mathbb{E}\left[\mathcal{R}\left(\hat{\mathbf{b}}_n^{(MCO)}\right)\right] & = & \sigma_\epsilon^2 + p\sigma_\epsilon^2\text{Trace}\left(\mathbf{X}^+\mathbf{X}^{+T}\right) \\ & = & \sigma_\epsilon^2 + p\sigma_\epsilon^2\text{Trace}\left(\mathbf{X}^{+T}\mathbf{X}^+\right) \end{array} \tag{B.4}$$

B.2 Modèle pénalisé Ridge

On suppose $\hat{\mathbf{b}}_n^{(Ridge)}$ estimé par la méthode Ridge, $\forall\lambda > 0$, il vient

$$\hat{\mathbf{b}}_n^{(Ridge)} = \left(\mathbf{X}^T\mathbf{X}/n + \lambda\mathbb{I}\right)^{-1}\mathbf{X}^T\mathbf{y}/n = \mathbf{X}^{(+,\lambda)}\mathbf{y},$$

où $\mathbf{X}^{(+,\lambda)} = \left(\mathbf{X}^T\mathbf{X}/n + \lambda\mathbb{I}\right)^{-1}\mathbf{X}^T/n$. On peut donc écrire

$$\begin{array}{ll} f(\mathbf{x}_1) = y_1 = \mathbf{x}_1\mathbf{b} + \epsilon_1, & f^\star(\mathbf{x}_1) = \mathbf{x}_1\mathbf{b}, \\ g^\star(\mathbf{x}_1) = \mathbb{E}_n\left[\mathbf{x}_1\hat{\mathbf{b}}_n^{(Ridge)}\right], & \hat{f}_n(\mathbf{x}_1) = \mathbf{x}_1\hat{\mathbf{b}}_n^{(Ridge)}, \end{array}$$

Avec, plus précisément

$$\begin{array}{rcl} \hat{f}_n(\mathbf{x}_1) & = & \mathbf{x}_1\hat{\mathbf{b}}_n^{(Ridge)} \\ & = & \mathbf{x}_1\mathbf{X}^{(+,\lambda)}\left(\mathbf{X}\mathbf{b} + \epsilon\right) \\ & = & \mathbf{x}_1\mathbf{X}^{(+,\lambda)}\mathbf{X}\mathbf{b} + \mathbf{x}_1\mathbf{X}^{(+,\lambda)}\epsilon, \\ g^\star(\mathbf{x}_1) & = & \mathbb{E}_n\left[\hat{f}_n(\mathbf{x}_1)\right] \\ & = & \mathbf{x}_1\mathbf{X}^{(+,\lambda)}\mathbf{X}\mathbf{b} + 0. \end{array}$$

L'erreur de bruit n'est pas modifiée par le modèle, en se référant au cas des MCO, il vient

$$\mathbb{E}\left[\left(f(\mathbf{x}_1) - f^\star(\mathbf{x}_1)\right)^2\right] = \sigma_\epsilon^2.$$

Nous avons décidé d'estimer le biais de façon plus classique dans ce cas, soit en considérant le biais de l'estimateur de $\mathbf{b}$, il vient

$$\begin{array}{rcl} \hat{\mathbf{b}}_n^{(Ridge)} - \mathbf{b} & = & \left(\mathbf{X}^T\mathbf{X}/n + \lambda\mathbb{I}\right)^{-1}\mathbf{X}^T\mathbf{X}\mathbf{b}/n - \mathbf{b} \\ & = & \left(\mathbf{X}^T\mathbf{X}/n + \lambda\mathbb{I}\right)^{-1}\mathbf{X}^T\mathbf{X}\mathbf{b}/n - \mathbf{b} \\ & = & -\lambda\left(\mathbf{X}^T\mathbf{X}/n + \lambda\mathbb{I}\right)^{-1}\mathbf{b}, \end{array}$$

et ainsi

$$\begin{array}{rcl} g^\star(\mathbf{x}_1) - f^\star(\mathbf{x}_1) & = & \mathbf{x}_1\left(\hat{\mathbf{b}}_n^{(Ridge)} - \mathbf{b}\right) \\ & = & -\lambda\mathbf{x}_1\left(\mathbf{X}^T\mathbf{X}/n + \lambda\mathbb{I}\right)^{-1}\mathbf{b}. \end{array}$$

Il vient donc que l'estimateur Ridge est biaisé. L'erreur associée à ce biais de modèle s'écrit

$$\mathbb{E}_{\mathbf{x}_1}\left[\left(f^\star(\mathbf{x}_1) - g^\star(\mathbf{x}_1)\right)^2\right] = \lambda^2\mathbf{b}^T\left(\mathbf{X}^T\mathbf{X}/n + \lambda\mathbb{I}\right)^{-2}\mathbf{b}.$$

Par la suite on peut calculer la variance de cet estimateur selon

$$\begin{array}{rcl} \mathbb{E}_n\left[\left(\hat{f}_n(\mathbf{x}_1) - g^\star(\mathbf{x}_1)\right)^2\right] & = & \mathbb{E}_n\left[\left(\mathbf{x}_1\mathbf{X}^{(+,\lambda)}\epsilon\right)^2\right], \\ & = & \mathbf{x}_1\mathbf{X}^{(+,\lambda)}\mathbb{E}_n\left[\epsilon\epsilon^T\right]\mathbf{X}^{(+,\lambda)^T}\mathbf{x}_1^T, \\ & = & {\sigma_\epsilon}^2\mathbf{x}_1\mathbf{X}^{(+,\lambda)}\mathbf{X}^{(+,\lambda)^T}\mathbf{x}_1^T. \end{array}$$

Ainsi par intégrale sur $\mathbf{x}_1$

$$\begin{array}{rcl} \mathbb{E}_{\mathbf{x}_1}\left[\mathbb{E}_n\left[\left(\hat{f}_n(\mathbf{x}_1) - g^\star(\mathbf{x}_1)\right)^2\right]\right] & = & {\sigma_\epsilon}^2\text{Trace}\left(\mathbb{E}_{\mathbf{x}_1}\left[\mathbf{x}_1^T\mathbf{x}_1\right]\left(\mathbf{X}^T\mathbf{X}/n + \lambda\mathbb{I}\right)^{-2}\mathbf{X}^T\mathbf{X}/n\right), \\ & = & {\sigma_\epsilon}^2\text{Trace}\left(\left(\mathbf{X}^T\mathbf{X}/n + \lambda\mathbb{I}\right)^{-2}\mathbf{X}^T\mathbf{X}/n\right). \end{array}$$

L'espérance du risque quadratique de l'estimateur Ridge dans le cas d'un bruit additif centré de variance σ_ϵ^2 entraîné sur un jeu de données de n individus pour p covariables et $q = 1$ variable réponse s'écrit donc

$$\begin{array}{rcl} \mathbb{E}\left[\mathcal{R}\left(\hat{\mathbf{b}}_n^{(Ridge)}\right)\right] & = & {\sigma_\epsilon}^2 + \lambda^2\mathbf{b}^T\left(\mathbf{X}^T\mathbf{X}/n + \lambda\mathbb{I}\right)^{-2}\mathbf{b} \\ & & + {\sigma_\epsilon}^2\text{Trace}\left(\left(\mathbf{X}^T\mathbf{X}/n + \lambda\mathbb{I}\right)^{-2}\mathbf{X}^T\mathbf{X}/n\right). \end{array} \quad \text{(B.5)}$$

B.3 Méthode pénalisée Lasso

Cette méthode nécessite que les variables soient centrées

$$\sum_{i=1}^{n} y_i = 0, \qquad \forall j \in [\![1,p]\!],\ \sum_{i=1}^{n} x_{i,j} = 0,$$

où $(\mathbf{X})_j$ est la j$^{\text{ème}}$ colonne de $\mathbf{X}$ assimilable à sa j$^{\text{ème}}$ variable et $x_{i,j}$ la valeur que prend l'échantillon i pour cette variable. De plus, les covariables doivent être réduites, soit

$$\forall j \in [\![1,p]\!],\ \sum_{i=1}^{n} x_{i,j}^2 = n.$$

On note aussi

$$L_l(\mathbf{b}) = \|\mathbf{y} - \mathbf{X}\mathbf{b}\|_2^2 / n + 2\lambda \|\mathbf{b}\|_1, \tag{B.6}$$

le critère que la méthode Lasso cherche à minimiser pour un certain choix de $\lambda > 0$. L_l est différentiable partout sauf aux points de changements de signe de $\mathbf{b}$. On recherche un minimum de L_l qui correspond à une annulation du gradient de L_l par rapport à $\mathbf{b}$ en supposant être assez loin des points d'annulation de chacun des éléments de $\mathbf{b}$.

$$\begin{aligned} d_{\mathbf{b}} L_l\left(\hat{\mathbf{b}}\right) &= -\mathbf{X}^T\left(\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}\right)/n + \lambda \text{sign}\left(\hat{\mathbf{b}}\right), \\ &= 0, \end{aligned}$$

et qui conduit à l'équation normale,

$$\mathbf{X}^T\mathbf{X}\hat{\mathbf{b}} = \mathbf{X}^T\mathbf{y} - n\lambda \text{sign}\left(\hat{\mathbf{b}}\right), \tag{B.7}$$

dont on va étudier les solutions dans la suite.

B.3.1 Solution dans le cas d'un "design orthogonal"

Dans cette partie on suppose $\mathbf{X}^T\mathbf{X} = n\mathbb{I}$, on peut alors remarquer que la solution des MCO, voir équation (1.14), s'écrit $\hat{\mathbf{b}}_n^{(MCO)} = \mathbf{X}^T\mathbf{y}/n$, il est possible de réécrire l'équation (B.6) qui devient

$$\begin{aligned} L_l\left(\hat{\mathbf{b}}\right) &= \mathbf{y}^T\mathbf{y}/n + \hat{\mathbf{b}}^T\hat{\mathbf{b}} - 2\hat{\mathbf{b}}^T\hat{\mathbf{b}}_n^{(MCO)} + 2\lambda\|\hat{\mathbf{b}}\|_1, \\ &= \text{cste} + \sum_{j=1}^{p}\hat{b}_j^2 - 2\hat{b}_j\hat{b}_j^{(MCO)} + 2\lambda|\hat{b}_j|, \end{aligned} \tag{B.8}$$

qui devient solvable en chaque $\hat{b}_j$ séparément en notant $\left(\hat{\mathbf{b}}_n^{(MCO)}\right)_j = \hat{b}_j^{(MCO)}$. Il est donc nécessaire de minimiser les p fonctions $l_j(b) = b^2 - 2b\hat{b}_j^{(MCO)} + 2\lambda|b_j|$. En supposant tout d'abord que :

— $\hat{b}_j^{(MCO)} > 0$. On peut supposer que $\hat{b}_j < 0$ il vient que $|\hat{b}_j| = -\hat{b}_j$ et alors la condition d'optimalité s'écrit

$$d_b l_j(\hat{b}_j) = 2\hat{b}_j - 2\hat{b}_j^{(MCO)} - 2\lambda\hat{b}_j = 0,$$

soit en réordonnant les termes

$$\hat{b}_j = \hat{b}_j^{(MCO)} + \lambda,$$

ce qui est une contradiction puisque $\hat{b}_j^{(MCO)} > 0$ et $\lambda > 0$ et donc $\hat{b}_j \geqslant 0$.

— Par une analyse symétrique on montre que si $\hat{b}_j^{(MCO)} < 0$ alors $\hat{b}_j \leqslant 0$.

En supposant maintenant que :

— $\hat{b}_j > 0$, il vient que $|\hat{b}_j| = \hat{b}_j$ et alors la condition d'optimalité s'écrit

$$d_b l_j(\hat{b}_j) \quad = \quad 2\hat{b}_j - 2\hat{b}_j^{(MCO)} + \lambda\hat{b}_j \quad = \quad 0,$$

soit en réordonnant les termes

$$\hat{b}_j = \hat{b}_j^{(MCO)} - \lambda,$$

réalisable si et seulement si $\hat{b}_j^{(MCO)} \geqslant \lambda$.

— $\hat{b}_j < 0$, par une analyse symétrique on montre que $\hat{b}_j^{(MCO)} \leqslant -\lambda$ et alors $\hat{b}_j = \hat{b}_j^{(MCO)} + \lambda$

Enfin en supposant que :

— $0 < \hat{b}_j^{(MCO)} < \lambda$. On sait que $\hat{b}_j \geqslant 0$ et on suppose $\hat{b}_j > 0$, il vient par application de la condition d'optimalité et en réogranisant les termes que

$$\hat{b}_j = \hat{b}_j^{(MCO)} - \lambda,$$

ce qui est impossible puisque $0 < \hat{b}_j^{(MCO)} < \lambda$. C'est donc que $\hat{b}_j = 0$

— Par une raisonnement semblable on montre que si $0 < -\hat{b}_j^{(MCO)} < \lambda$, alors $\hat{b}_j = 0$.

Finalement on raccorde le point $\hat{b}_j^{(MCO)}$ par un argument de continuité, il vient alors que $\hat{b}_j$ peut s'exprimer comme une fonction croissante de $\hat{b}_j^{(MCO)}$, à savoir

$$\hat{b}_j^{(Lasso)} = \begin{cases} \hat{b}_j^{(MCO)} - \text{sign}\left(\hat{b}_j^{(MCO)}\right)\lambda & si \quad \left|\hat{b}_j^{(MCO)}\right| > \lambda \\ 0 & sinon \end{cases}$$

ANNEXE C

MÉTHODE DES PUISSANCES ITÉRÉES

Cette méthode permet de diagonaliser les matrices semi-définies positives, non nulles, dont les éléments propres non nuls sont tous distincts.

Cet algorithme fonctionne de façon séquentielle en commençant par la recherche du premier sous-espace (voir section C.1), le sous-espace porté par la plus grande valeur propre, puis l'information de la matrice à diagonaliser et portée par ce sous-espace est retirée de la matrice puis la première étape est de nouveau appliquée à la matrice résiduelle, c'est ce que l'on appelle la déflation (voir section C.2).

C.1 Recherche du premier sous-espace propre

On suppose dans cette partie une matrice réelle semi-définie positive $\mathbf{A}$ ($\mathbf{A} \in \mathcal{S}_n^+(\mathbb{R})$) dont nous recherchons le premier sous-espace propre.

Puisque $\mathbf{A} \in \mathcal{S}_n^+(\mathbb{R})$, $\mathbf{A}$ est aussi diagonalisable de valeurs propres $(\lambda_k)_{k\in[\![1,n]\!]}$ ordonnées telles que

$$\lambda_1 \geqslant \lambda_2 \geqslant .. \geqslant \lambda_r \geqslant .. \geqslant \lambda_n \geqslant 0,$$

en notant de plus r le rang de la matrice $\mathbf{A}$, $r = rang(A)$, nous avons $\forall k \in]\!]r,n]\!], \lambda_k = 0$, et donc :

$$\lambda_1 \geqslant \lambda_2 \geqslant .. \geqslant \lambda_r \geqslant 0.$$

Dans le cas de données réelles et donc bruitées, il est très peu probables d'obtenir des sous-espaces propres dont la multiplicité soit strictement supérieure à 1, sans considérer le noyau de $\mathbf{A}$, et donc

$$\lambda_1 > \lambda_2 > .. > \lambda_r > 0.$$

On note $\mathcal{V}$ la base orthonormée de $\mathbb{R}^n$ associée de diagonalisation de $\mathbf{A}$. Soit $(\mathbf{v}_k)_{k\in[\![1,n]\!]}$ les n vecteurs normés associés à chacune des valeurs propres.

$\forall \mathbf{x} \in \mathbb{R}^n, \exists!(a_k)_{k\in[\![1,n]\!]} \in \mathbb{R}^n | \mathbf{x} = \sum_{k=1}^n a_k \mathbf{v}_k$. Dont le produit par $\mathbf{A}$ s'écrit

$$\mathbf{A}\mathbf{x} = \sum_{k=1}^{n} \mathbf{A}\mathbf{x}_k\mathbf{v}_k = \sum_{k=1}^{n} a_k(\mathbf{A}\mathbf{v}_k) = \sum_{k=1}^{n} \lambda_k a_k \mathbf{v}_k = \sum_{k=1}^{r} \lambda_k a_k \mathbf{v}_k.$$

Par itérations successives il vient $\forall N \in \mathbb{N}^\star$:

$$\mathbf{A}^N\mathbf{x} = \sum_{k=1}^{n} \mathbf{A}^N a_k \mathbf{v}_k = \sum_{k=1}^{n} a_k(\mathbf{A}^N\mathbf{v}_k) = \sum_{k=1}^{n} a_k\mathbf{A}^{N-1}(\mathbf{A}\mathbf{v}_k) = \sum_{k=1}^{r} \lambda_k^N a_k \mathbf{v}_k.$$

En divisant par λ_1^N, il vient

$$\frac{\mathbf{A}^N\mathbf{x}}{\lambda_1^N} = \sum_{k=1}^{r} (\frac{\lambda_k}{\lambda_1})^N a_k \mathbf{v}_k \tag{C.1}$$

où dans le cas limite $N \to +\infty$

$$\left(\frac{\lambda_k}{\lambda_1}\right)^N \xrightarrow[N\to+\infty]{} \begin{cases} 1 & \text{if } k = 1 \\ 0 & \text{otherwise} \end{cases}.$$

Et donc pour l'équation (C.1) il vient $\frac{\mathbf{A}^N\mathbf{x}}{\lambda_1^N}$, when $N \to +\infty$

$$\frac{\mathbf{A}^N\mathbf{x}}{\lambda_1^N} \xrightarrow[N\to+\infty]{} a_1\mathbf{v}_1.$$

En utilisant la norme euclidienne, qui est continue sur les espaces vectoriels de dimension finie, il vient $||\frac{\mathbf{A}^N\mathbf{x}}{\lambda_1^N}||_2 \xrightarrow[N\to+\infty]{} ||x_1 v_1||_2 = |x_1| \cdot ||v_1||_2 = |x_1| \cdot 1 = |x_1|$. Ce qui permet d'écrire

$$\frac{\mathbf{A}^N\mathbf{x}}{||\mathbf{A}^N\mathbf{x}||_2} = \frac{\frac{\mathbf{A}^N\mathbf{x}}{\lambda_1^N}}{||\frac{\mathbf{A}^N\mathbf{x}}{\lambda_1^N}||_2} \xrightarrow[N\to+\infty]{} \text{sign}(x_1)\mathbf{v}_1,$$

où le terme de signe est souvent mis de côté en considérant que le plus grand coefficient de $\mathbf{v}_1$ doit être positif.

On vient donc de mettre en lumière un algorithme capable d'évaluer le sous-espace propre associé à la plus grande valeur propre d'une matrice semi-définie positive

$$\mathbf{A}$$

. De façon plus concise on peut écrire l'algorithme comme suit

- **Étape 1** : Choisir $\mathbf{x} \in \mathbb{R}^n$
- **Étape 2.1** : Calculer $\mathbf{b}_1 = \frac{\mathbf{A}\mathbf{x}}{||\mathbf{A}\mathbf{x}||_2}$
- **Étape 2.2** : Pour $N = 1..$, faire

  $$\mathbf{b}_{N+1} = \frac{\mathbf{A}b_N}{||\mathbf{A}\mathbf{b}_N||_2}$$

  Si $||\mathbf{b}_{N+1} - \mathbf{b}_N||_2 < \epsilon$, la convergence est atteinte, arrêter la boucle.
- **Étape 3** : Finalement $\mathbf{v}_1 = \mathbf{b}_{N+1}$ et $\lambda_1 = \mathbf{b}_{N+1}^T\mathbf{A}\mathbf{b}_{N+1}$

C.2 Déflation

Soit $\mathbf{v}_1$, le vecteur propre associé au premier sous-espace de la matrice $\mathbf{A}$, on appelle déflation de $\mathbf{A}$ sur $\mathbf{v}_1$ l'opération

$$\mathbf{A} \leftarrow \mathbf{A} - \frac{\mathbf{v}_1(\mathbf{A}\mathbf{v}_1)^T}{||\mathbf{A}\mathbf{v}_1||_2},$$

la matrice rsiduelle est dite la déflatée de la matrice initiale.

ANNEXE D

L'ALGORITHME EM POUR L'ESTIMATION DE DONNÉES MANQUANTES

Dans cette partie on suppose que $\mathbf{X}$ est un échantillon indépendant de n observations du vecteur aléatoire réel $\mathbf{x}$ de paramètre Θ à valeurs réelles.

D.1 Vraisemblance et maximisation de vraisemblance

On définit la vraisemblance de l'échantillon $\mathbf{X}$ observé pour un paramètre Θ par

$$\mathbb{P}(\mathbf{X}|\Theta) = \prod_{i=1}^{n} \mathbb{P}(\mathbf{X}_i|\Theta),$$

puisque les n observations sont indépendantes. Il est souvent plus intéressant de décrire la *log-vraisemblance* de l'échantillon, qui s'écrit

$$L(\Theta;\mathbf{X}) = \ln \mathbb{P}(\mathbf{X}|\Theta) = \sum_{i=1}^{n} \ln \mathbb{P}(\mathbf{X}_i|\Theta), \tag{D.1}$$

où la plus vraisemblable des valeurs de Θ est trouvée en maximisant ce dernier. La méthode du maximum de vraisemblance recherche le paramètre Θ qui maximise la vraisemblance, on l'appelle Estimateur du Maximum de Vraisemblance (*MLE*). Ce maximum, en $\Theta = \Theta_{MLE}$ est solution de

$$\frac{\partial L(\Theta;\mathbf{X})}{\partial \Theta}_{\Theta=\Theta_{MLE}} = \mathbf{0},$$

à supposer que cette écriture a un sens du point de vue de la dérivabilité de la fonction de *log-vraisemblance*, ce que nous supposerons dans la suite de cet énoncé.

Θ_{MLE} peut avoir une écriture simple, littérale nous entendons, et alors la méthode du maximum de vraisemblance est rapide. Dans le cas général il est nécessaire d'avoir recours à des méthodes approchées, telles que les méthodes de type Newton, c'est d'ailleurs la solution utilisée pour le calcul approché des solutions au problème de la régression logistique dont la genèse semblerait remonter au XIX$^{\text{ème}}$ siècle et aurait permis de décrire la croissance d'une population

d'une réaction chimique dont le taux de croissance ne doit pas seulement tenir compte d'une croissance exponentielle classique mais aussi de l'espace encore accessible à la population dans la chambre de croissance. Ces résultats sont dus à Verhulst (1838), première communication d'un sujet qui sera suivie par la littérature que nous connaissons.

D.2 Vraisemblances complexes et idée de l'algorithme EM

Dans le cas où le jeu de données $\mathbf{X}$ contient des données manquantes, il n'est plus possible d'estimer directement Θ_{MLE}, pourtant la simplicité de cette méthode et l'interprétabilité des paramètres estimés ont poussé à rechercher une solution pour estimer les données manquantes en fonction des données présentes et d'une estimation courante des paramètres du modèle statistique.

L'algorithme EM (pour *Estimation-Maximisation*) est une approche générale pour le calcul itératif de vraisemblances optimales en présence de données manquantes. Des vraisemblances sont calculées de façon itératives en mettant à jour l'estimation des données manquantes. Cet algorithme se divise en deux étapes qui sont

— **Étape E**, l'étape d'estimation, les données manquantes sont estimées au moyen d'une espérance conditionnelle sur les données présentes et les paramètres estimés à l'itération précédente.

— **Étape M**, l'étape de maximisation (sous-entendu "maximisation de la vraisemblance"), les paramètres sont calculés par maximisation de la vraisemblance en utilisant les données estimés à l'**étape E** en plus des données présentes.

Les travaux de Dempster et collab. (1977) ont permis de formaliser cet algorithme, démontrant que ce dernier, quoiqu'à convergence lente, est assuré de converger théoriquement. Les conditions d'utilisation de ce dernier prévoient que le mécanisme de données manquantes doit être au moins MAR, or nous avons pris l'hypothèse d'un mécanisme MCAR, la méthode est donc bien applicable. Ceci implique notamment que les distributions de probabilité des échantillons manquants et des échantillons présents sont identiques.

Cet algorithme est particulièrement efficace, tout comme l'est l'algorithme du maximum de vraisemblance, si la distribution des données n'implique pas une fonction de vraisemblance trop complexe. Si donc la vraisemblance conditionnelle admet une écriture simple. Dans le cas de l'hypothèse d'un mélange de gaussiennes, ce qui est l'exemple classique, la donnée manquante est l'appartenance de chaque observation à l'une des classes, cet inconnu étant fixé, le reste de la vraisemblance est beaucoup plus simple.

D.3 Présence de données manquantes

Soit $\mathbf{X}^{(m)}$ les données manquantes et $\mathbf{X}^{(p)}$ les données présentes, alors la fonction de densité jointe des variables $\mathbf{X}^{(m)}$ et $\mathbf{X}^{(p)}$ peut s'écrire

$$\mathbb{P}(\mathbf{X}|\Theta) = \mathbb{P}\left(\mathbf{X}^{(m)}, \mathbf{X}^{(p)}|\Theta\right) = \mathbb{P}\left(\mathbf{X}^{(m)}|\mathbf{X}^{(p)}, \Theta\right)\mathbb{P}\left(\mathbf{X}^{(p)}|\Theta\right).$$

On définit alors $L(\Theta;\mathbf{X}) = \ln \mathbb{P}\left(\mathbf{X}|\Theta\right)$. L'objectif est de maximiser $L(\Theta;\mathbf{X})$, on note

$$\Theta^{\star} = \arg\max_{\Theta} L(\Theta;\mathbf{X})$$

D.4 Esprit de l'algorithme EM

En notant $\mathcal{Y}$ l'ensemble formé par les évènements possibles pour les données manquantes et $\forall\ \mathbf{Y} \in \mathcal{Y}$, on réécrit $L(\Theta;\mathbf{X})$ en sommant sur $\mathcal{Y}$ au moyen de la formule de Bayes tel que

$$L(\Theta;\mathbf{X}) \quad = \quad \ln \int_{\mathcal{Y}} \mathbb{P}\left(\mathbf{X}^{(p)}|\ \mathbf{Y},\Theta\right)\mathbb{P}(\ \mathbf{Y}|\Theta) d\ \mathbf{Y},$$

si l'on note Θ_0 un paramètre initial, cohérent avec la distribution des données, il vient

$$\begin{aligned}
L\left(\Theta;\mathbf{X}\right) &= \ln \int_{\mathcal{Y}} \mathbb{P}\left(\mathbf{X}^{(p)}|\ \mathbf{Y},\Theta\right)\mathbb{P}\left(\ \mathbf{Y}|\Theta\right)\frac{\mathbb{P}\left(\ \mathbf{Y}|\mathbf{X}^{(p)},\Theta_0\right)}{\mathbb{P}\left(\ \mathbf{Y}|\mathbf{X}^{(p)},\Theta_0\right)} d\ \mathbf{Y}, \\
&= \ln \int_{\mathcal{Y}} \mathbb{P}\left(\ \mathbf{Y}|\mathbf{X}^{(p)},\Theta_0\right)\frac{\mathbb{P}\left(\mathbf{X}^{(p)}|\ \mathbf{Y},\Theta\right)\mathbb{P}(\ \mathbf{Y}|\Theta)}{\mathbb{P}\left(\ \mathbf{Y}|\mathbf{X}^{(p)},\Theta_0\right)} d\ \mathbf{Y}, \\
&\geqslant \int_{\mathcal{Y}} \mathbb{P}\left(\ \mathbf{Y}|\mathbf{X}^{(p)},\Theta_0\right)\ln\frac{\mathbb{P}\left(\mathbf{X}^{(p)}|\ \mathbf{Y},\Theta\right)\mathbb{P}(\ \mathbf{Y}|\Theta)}{\mathbb{P}\left(\ \mathbf{Y}|\mathbf{X}^{(p)},\Theta_0\right)} d\ \mathbf{Y}, \\
&= \int_{\mathcal{Y}} \mathbb{P}\left(\ \mathbf{Y}|\mathbf{X}^{(p)},\Theta_0\right)\ln\frac{\mathbb{P}\left(\mathbf{X}^{(p)}|\ \mathbf{Y},\Theta\right)\mathbb{P}(\ \mathbf{Y}|\Theta)}{\mathbb{P}\left(\ \mathbf{Y}|\mathbf{X}^{(p)},\Theta_0\right)\mathbb{P}\left(\mathbf{X}^{(p)}|\Theta_0\right)} d\ \mathbf{Y} + \ln \mathbb{P}\left(\mathbf{X}^{(p)}|\Theta_0\right),
\end{aligned} \tag{D.2}$$

qui est réalisé en appliquant l'inégalité de Jensen (voir Jensen et collab., 1906) à la fonction convexe ln puisque $\int_{\mathcal{Y}} \mathbb{P}\left(\ \mathbf{Y}|\mathbf{X}^{(p)},\Theta_0\right) d\ \mathbf{Y} = 1$ et en remarquant que $\ln \mathbb{P}\left(\mathbf{X}^{(p)}|\Theta_0\right)$ est une constante vis à vis de $\mathbf{z}$. Le dernier terme de l'équation (D.2) peut être reconnu comme étant égal à $L(\Theta_0;\mathbf{X})$, alors que le premier terme peut être renommé $\Delta(\Theta|\Theta_0)$. On remarque que

$$\begin{aligned}
\Delta\left(\Theta_0|\Theta_0\right) &= \int_{\mathcal{Y}} \mathbb{P}\left(\ \mathbf{Y}|\mathbf{X}^{(p)},\Theta_0\right)\ln\frac{\mathbb{P}\left(\mathbf{X}^{(p)}|\ \mathbf{Y},\Theta_0\right)\mathbb{P}(\ \mathbf{Y}|\Theta_0)}{\mathbb{P}\left(\ \mathbf{Y}|\mathbf{X}^{(p)},\Theta_0\right)\mathbb{P}\left(\mathbf{X}^{(p)}|\Theta_0\right)} d\ \mathbf{Y} \\
&= \int_{\mathcal{Y}} \mathbb{P}\left(\ \mathbf{Y}|\mathbf{X}^{(p)},\Theta_0\right)\ln\frac{\mathbb{P}\left(\mathbf{X}^{(p)},\ \mathbf{Y}|\Theta_0\right)}{\mathbb{P}\left(\ \mathbf{Y},\mathbf{X}^{(p)}|\Theta_0\right)} d\ \mathbf{Y} \\
&= \int_{\mathcal{Y}} \mathbb{P}\left(\ \mathbf{Y}|\mathbf{X}^{(p)},\Theta_0\right)\ln 1 d\ \mathbf{Y} \\
&= 0,
\end{aligned}$$

par application du Théorème de Bayes. On définit alors une fonction $l(\Theta|\Theta_0) = L(\Theta_0;\mathbf{X}) + \Delta(\Theta|\Theta_0)$ qui vérifie

$$l(\Theta|\Theta_0) \leqslant L(\Theta|\mathbf{X}),$$

$l(\Theta|\Theta_0)$ est donc majorée par la vraisemblance conditionnelle que l'on recherche mais lui est aussi égale en $\Theta = \Theta_0$. Donc tout élément Θ_1 qui accroit la valeur de $l(\Theta|\Theta_0)$ est une meilleure estimation de $\Theta^\star$ puisqu'alors $L(\Theta_1|\mathbf{X}) \geqslant L(\Theta_0|\mathbf{X})$.

Nous sommes en mesure de construire une suite de paramètres croissants et majorés. Cette suite doit donc converger vers un maximum local qui est l'estimation de $\Theta^\star$.

D.5 L'algorithme EM

L'algorithme tel que présenté précédemment peut être allégé en évitant de calculer les termes constants dans le logarithme, comprendre les termes indépendants de Θ. On obtient alors

$$\begin{aligned}
\Delta\left(\Theta|\Theta_0\right) &= \int_{\mathcal{Y}} \mathbb{P}\left(\ \mathbf{Y}|\mathbf{X}^{(p)},\Theta_0\right)\ln\left(\mathbb{P}\left(\mathbf{X}^{(p)}|\ \mathbf{Y},\Theta\right)\mathbb{P}\left(\ \mathbf{Y}|\Theta\right)\right) d\ \mathbf{Y} + C \\
&= \mathbf{E}_{\mathbf{Y}|\mathbf{X}^{(p)},\Theta_0}\left(\ln\left(\mathbb{P}\left(\mathbf{X}^{(p)}|\ \mathbf{Y},\Theta\right)\mathbb{P}\left(\ \mathbf{Y}|\Theta\right)\right)\ |\ \mathbf{X}^{(p)},\Theta_0\right) + C \\
&= \mathbf{E}_{\mathbf{Y}|\mathbf{X}^{(p)},\Theta_0}\left(\ln\left(\mathbb{P}\left(\mathbf{X}^{(p)},\ \mathbf{Y}|\Theta\right)\right)\ |\ \mathbf{X}^{(p)},\Theta_0\right) + C \\
&= Q\left(\Theta|\Theta_0\right) + C
\end{aligned}$$

On définit alors précisément l'algorithme EM. Pour toute initialisation raisonnable Θ_0 de Θ, et pour une itération quelconque $i+1>0$ de l'algorithme EM, on réalise les deux étapes :

— **Étape E** : Estimer $Q(\Theta|\Theta_i)$.

— **Étape M** : Maximiser, par rapport à Θ, la fonctionnelle précédente pour obtenir Θ_{i+1}.

Remarquons que même si l'écriture recherchée dans l'**Étape E** peut être simple, l'optimisation de l'autre étape peut ne pas l'être et la recherche d'un optimum strict n'est pas requise. L'algorithme EM généralisé permet de résoudre les problèmes où cette seconde étape est particulièrement lourde en recherchant des accroissements dans la suite des Θ_i sans que ce soit la suite optimale. Il convient de signaler que cet algorithme ne recherche pas un maximum global mais un maximum local et qu'il est, dans les faits, conditionné par l'initialisation Θ_0. Des adaptations de cet algorithme (notamment par Celeux, 1985) permettent de répondre, dans certains cas, aux problèmes des *minima* locaux non informatifs via un échantillonnage des estimations entre les deux étapes précédemment décrites.

D.6 Cas d'école d'utilisation de l'algorithme EM pour l'imputation

L'algorithme EM nécessite qu'un modèle statistique soit associé à l'observation qui est faite des données courantes. Nous nous placerons ici dans le cas simple où deux variables aléatoires $X^{(1)}$ et $X^{(2)}$ sont observées au travers de $n=100$ observations indépendantes et identiquement distribuées selon une loi normale centrée. La variable $X^{(1)}$ admet une proportion de donnée manquante p_{NA} et la variable $X^{(2)}$ est totalement observée. Nous supposerons que les deux variables sont associés par un modèle linéaire tel que

$$\forall i \in [\![1,n]\!],\ X_i^{(2)} = a + bX_i^{(1)} + \epsilon_i, \tag{D.3}$$

où a et b représentent respectivement l'ordonnée à l'origine et la pente associées au modèle linéaire et ϵ_i est l'erreur d'observation de l'échantillon i avec $\epsilon_i \sim \mathcal{N}(0,\sigma^2)$. Dans le cadre de ces simulations nous choisirons les paramètres suivants

$$a=3,\ b=2,\ \sigma^2=1,$$

L'objectif est d'estimer les données manquantes grâce à ce modèle statistique en utilisant l'algorithme EM qui demande une initialisation des données manquantes et des paramètres du modèle et qui résout, de façon itérative jusqu'à convergence, les deux étapes suivantes, où k est l'indice de l'itération courante :

— **Étape E** : Estimer $Q(\Theta|\Theta_{k-1})$.

— **Étape M** : Maximiser, par rapport à Θ, la fonctionnelle précédente pour obtenir Θ_k.

Avec $\Theta=[a,b,\sigma^2]^T$ le vecteur de paramètre associé au modèle et Q la fonctionnelle dérivée de la log-vraisemblance associée au modèle statistique. Dans le cas présent nous avons

$$Q(\Theta|\Theta_{k-1}) = -\frac{1}{\sigma^2}\sum_{i=1}^{n}\left(\mathbf{x}_i{}^{(2)} - a - b\mathbf{x}_i{}^{(1)}\right)^2 - n\ln\sigma^2,$$

où $\mathbf{x}_i{}^{(1)}$ et $\mathbf{x}_i{}^{(2)}$ représentent les observations des variables $X^{(1)}$ et $X^{(2)}$ pour l'observation i. Il vient que certaines valeurs $\mathbf{x}_i{}^{(1)}$, celles qui sont initialement manquantes, doivent être indicées

par k pour correspondre à leur estimation courante mais nous prenons la liberté de ne pas surcharger les équations avec cette notation.

L'**étape E** nécessite de connaître un modèle pour prédire $X^{(1)}$ à partir de $X^{(2)}$, par inversion directe du modèle (D.3), il vient le modèle

$$\forall i \in [\![1,n]\!],\ x_i^{(1)} = -\frac{a}{b} + \frac{1}{b}x_i^{(2)} + \nu_i,$$

où $\nu_i \sim \mathcal{N}(0, \frac{\sigma^2}{b^2})$ et ainsi le calcul des $\mathbf{x}_i{}^{(1)}$ manquants peut se faire au moyen de l'équation d'estimation

$$x^{(1)} \approx -\frac{a}{b} + \frac{1}{b}x^{(2)}. \tag{D.4}$$

L'**étape M** estime les paramètres du modèle par maximisation de $Q(\Theta|\Theta_{k-1})$ par rapport à chacune des variables, soit en résolvant les trois équations

$$\begin{array}{rcccrcl}
\partial_a Q(\Theta|\Theta_{k-1}) & = & 0 & \iff & b & = & \frac{\mathrm{cov}\left(\mathbf{X}^{(2)},\mathbf{X}^{(1)}\right)}{\mathrm{var}\left(\mathbf{X}^{(1)}\right)} \\
\partial_b Q(\Theta|\Theta_{k-1}) & = & 0 & \iff & a & = & \frac{1}{n}\sum_{i=1}^n x_i^{(2)} - a - b x_i^{(1)} \\
\partial_{\sigma^2} Q(\Theta|\Theta_{k-1}) & = & 0 & \iff & \sigma^2 & = & \frac{1}{n}\sum_{i=1}^n \left(x_i^{(2)} - a - b x_i^{(1)}\right)^2
\end{array},$$

où $\mathbf{X}^{(1)}$ et $\mathbf{X}^{(2)}$ sont les vecteurs d'observation associés aux variables $X^{(1)}$ et $X^{(2)}$ respectivement.

La figure 12 présente les résultats de trois simulations réalisées sur ce modèle pour un nombre $n = 100$ d'observations. Dans cette figure sont représentées trois séries de graphes correspondant à des proportions $p_{NA} \in \{20\%, 50\%, 80\%\}$ de données manquantes pour la variable $X^{(1)}$. Sur chacun de ces graphes trois séries de courbe sont représentées, en bleu les corrélations entre $\mathbf{X}^{(1)}$ et $\mathbf{X}^{(2)}$ pour les données imputées (symbole $\circ$) et pour l'ensemble du jeu de données (symbole $\bullet$). De la même manière sont aussi représentées les variances, en rouge, de la variable $X^{(1)}$. La courbe verte représente les valeurs prises par la fonction $Q(\Theta|\Theta_k)$. La première itération correspond à l'initialisation, qui est ici réalisée par une imputation à la moyenne. Un échantillon de $n = 100$ observations est considéré.

Notons que cor $\left(\mathbf{X}^{(1)}, \mathbf{X}^{(2)}\right)$ vaut initialement 0 pour les données imputées, ce qui est naturel puisque la variance de ces réalisations est alors nulle (imputation à la moyenne). De plus, cette même fonction vaut 1 pour toutes les autres itérations puisque l'imputation est faite au moyen du modèle (D.4), qui ne présente pas de bruit d'observation.

On remarque que la courbe d'évolution de $Q(\Theta|\Theta_k)$ est bien croissante, comme le prévaut la théorie. De plus, chacune des deux courbes de variance et de corrélation (correspondant aux estimations sur données complètes et sur données imputées) semblent se rapprocher. Ceci s'interprète en s'appuyant sur le proportion de données manquantes. Plus cette dernière est importante et plus les données vont être guidées par le comportement des données imputées, à la limite les données imputées guident complètement l'information et les deux courbes sont, pour la variance et pour la corrélation, superposées.

On peut aussi noter que plus la proportion de données manquantes est importante et plus les variances et corrélations estimées (sur données totales et sur données imputées) s'éloignent de la valeur sur données observées. Le modèle prévaut une corrélation parfaite entre les deux variables pour les estimations des données manquantes, ceci a pour effet de surestimer la force du lien entre les observations et devient préjudiciable lorsque la proportion de données manquantes devient importante. Le nombre d'itérations nécessaires à la stabilisation de l'algorithme

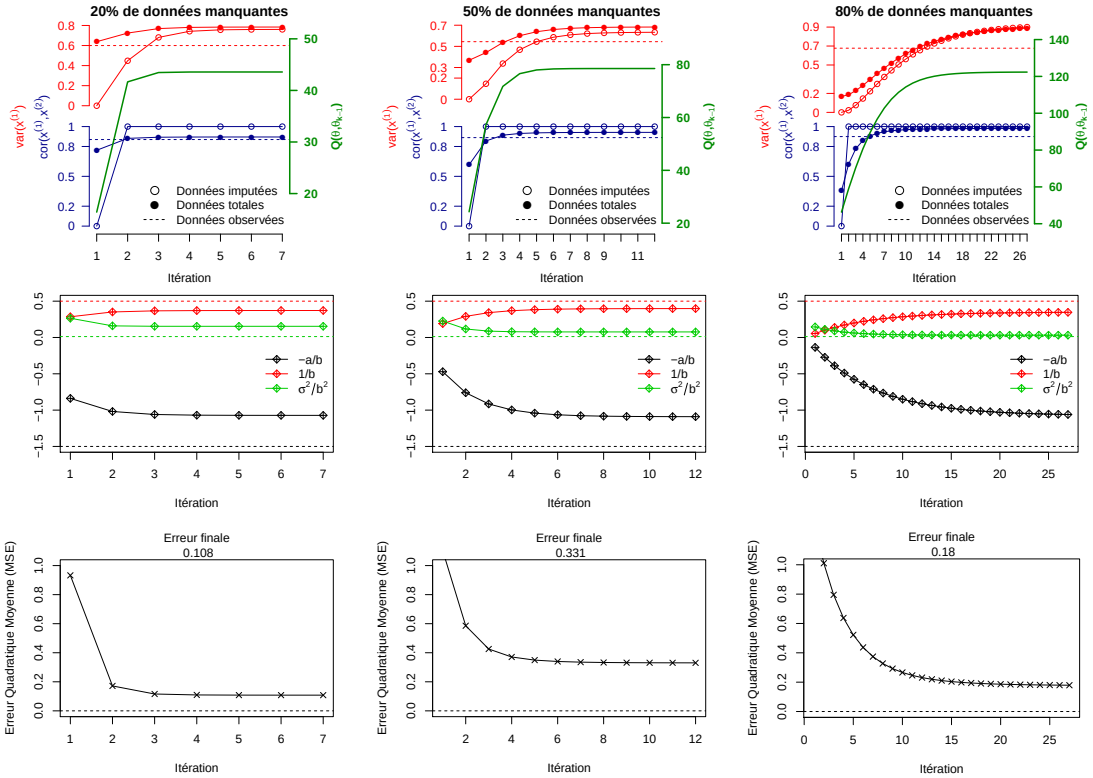


FIGURE 12 – Simulations de 3 jeux de données, pour 3 proportions différentes de données manquantes $(20\%, 50\%, 80\%)$ de 2 variables $\left(x^{(1)}, x^{(2)}\right)$, où la variable $x^{(1)}$ présente seule des données manquantes. L'imputation est effectuée au moyen d'un algorithme EM en supposant un modèle linéaire liant ces deux variables. En première ligne : les asymptotes horizontales présentent les valeurs de corrélation et de variance pour les données observées. En seconde ligne : les asymptotes reprennent les valeurs théoriques des paramètres. En troisième ligne : les erreurs sur les données imputées lors de la dernière itération. $n = 100$ échantillons sont utilisés.

est lui aussi croissant selon la proportion de données manquantes. L'analyse de la seconde ligne montre que le modèle permet d'avoir une meilleure estimation des paramètres du modèle linéaire lorsque l'algorithme EM a convergé plutôt qu'au début. La troisième ligne montre que l'algorithme EM permet de diminuer l'erreur. On remarque aussi que les erreurs pour le modèle avec 50% de données manquantes sont plus importantes que dans le cas à 20%. De même il n'est pas assuré que cette erreur soit minimale à la convergence de l'algorithme.

Les résultats qui viennent d'être présentés montrent que l'algorithme EM permet de résoudre le problème des données manquantes dans un contexte unidimensionnel simple. Plus précisément, il permet 1) d'estimer les paramètres du modèle statistique appliqué aux données et 2) d'imputer les données manquantes. La vraisemblance associée au problème étudié est souvent complexe à écrire ce qui peut être symptomatique d'un algorithme qui ressemble à un EM sans pour autant en être tout à fait un. La littérature emploie souvent le terme de "typé-EM". L'algorithme ddsPLS, que nous avons développé pendant cette thèse et qui est détaillé dans la partie 2 présente un profil d'EM sans que cela ne soit démontré formellement.

BIBLIOGRAPHIE

Acar, E., D. M. Dunlavy et T. G. Kolda. 2011a, «A scalable optimization approach for fitting canonical tensor decompositions», Journal of Chemometrics, vol. 25, n° 2, p. 67–86.

Acar, E., T. G. Kolda et D. M. Dunlavy. 2011b, «All-at-once optimization for coupled matrix and tensor factorizations», arXiv preprint arXiv :1105.3422.

Acar, E., M. A. Rasmussen, F. Savorani, T. Næs et R. Bro. 2013, «Understanding data fusion within the framework of coupled matrix and tensor factorizations», Chemometrics and Intelligent Laboratory Systems, vol. 129, p. 53–63.

Akaike, H. 1973, «Information theory and an extension of the maximum likelihood principle», dans 2nd International Symposium on Information Theory, Akademiai Kiado, p. 267–281.

Allen, D. M. 1974, «The relationship between variable selection and data agumentation and a method for prediction», Technometrics, vol. 16, n° 1, p. 125–127.

Arlot, S. 2018, «Validation croisée», .

Bakin, S. 1999, Adaptive regression and model selection in data mining problems, thèse de doctorat, School of Mathematical Sciences, Australian National University.

Barker, M. et W. Rayens. 2003, «Partial least squares for discrimination», Journal of Chemometrics : A Journal of the Chemometrics Society, vol. 17, n° 3, p. 166–173. URL https://doi.org/10.1002/cem.785.

Becht, E., L. McInnes, J. Healy, C.-A. Dutertre, I. W. Kwok, L. G. Ng, F. Ginhoux et E. W. Newell. 2019, «Dimensionality reduction for visualizing single-cell data using umap», Nature biotechnology, vol. 37, n° 1, p. 38.

Bickel, P. J., E. Levina et collab.. 2008, «Regularized estimation of large covariance matrices», The Annals of Statistics, vol. 36, n° 1, p. 199–227.

Bougeard, S., E. M. Qannari, C. Lupo et M. Hanafi. 2011, «From multiblock partial least squares to multiblock redundancy analysis. a continuum approach», Informatica, vol. 22, n° 1, p. 11–26.

Breiman, L. 1996, «Bagging predictors», Machine learning, vol. 24, n° 2, p. 123–140.

Breiman, L. 2001, «Random forests», Machine learning, vol. 45, n° 1, p. 5–32.

Breiman, L. et P. Spector. 1992, «Submodel selection and evaluation in regression. the x-random case», International statistical review/revue internationale de Statistique, p. 291–319.

Bro, R. 1996, «Multiway calibration. multilinear pls», Journal of chemometrics, vol. 10, n° 1, p. 47–61.

Carroll, J. D. et J.-J. Chang. 1970, «Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition», Psychometrika, vol. 35, n° 3, p. 283–319.

Caussinus, H. 1986, «Models and uses of principal component analysis», Multidimensional data analysis, vol. 86, p. 149–170.

Celeux, G. 1985, «The sem algorithm : a probabilistic teacher algorithm derived from the em algorithm for the mixture problem», Computational statistics quarterly, vol. 2, p. 73–82.

Chun, H. et S. Keleş. 2010, «Sparse partial least squares regression for simultaneous dimension reduction and variable selection», Journal of the Royal Statistical Society : Series B (Statistical Methodology), vol. 72, n° 1, p. 3–25.

Clemmensen, L., T. Hastie, D. Witten et B. Ersbøll. 2011, «Sparse discriminant analysis», Technometrics, vol. 53, n° 4, p. 406–413.

Cornuéjols, A. et L. Miclet. 2010, «Apprentissage artificiel : Concepts et algorithmes», .

Cortes, C. et V. Vapnik. 1995, «Support-vector networks», Machine learning, vol. 20, n° 3, p. 273–297.

d'Aspremont, A., L. E. Ghaoui, M. I. Jordan et G. R. Lanckriet. 2005, «A direct formulation for sparse pca using semidefinite programming», dans Advances in neural information processing systems, p. 41–48.

De Jong, S. 1995, «Pls shrinks», Journal of chemometrics, vol. 9, n° 4, p. 323–326.

Dempster, A. P., N. M. Laird et D. B. Rubin. 1977, «Maximum likelihood from incomplete data via the em algorithm», Journal of the Royal Statistical Society : Series B (Methodological), vol. 39, n° 1, p. 1–22.

Deshpande, Y. et A. Montanari. 2016, «Sparse pca via covariance thresholding», Journal of Machine Learning Research, vol. 17, n° 141, p. 1–41. URL http://jmlr.org/papers/v17/15-160.html.

Domingos, P. 2000, «A unified bias-variance decomposition», dans Proceedings of 17th International Conference on Machine Learning, p. 231–238.

Eddelbuettel, D. et R. François. 2011, «Rcpp : Seamless R and C++ integration», Journal of Statistical Software, vol. 40, n° 8, doi :10.18637/jss.v040.i08, p. 1–18. URL http://www.jstatsoft.org/v40/i08/.

Efron, B. 1983, «Estimating the error rate of a prediction rule : improvement on cross-validation», Journal of the American statistical association, vol. 78, n° 382, p. 316–331.

Efron, B., T. Hastie, I. Johnstone, R. Tibshirani et collab.. 2004, «Least angle regression», The Annals of statistics, vol. 32, n° 2, p. 407–499.

Escofier, B. et collab.. 1984, «L'analyse factorielle multiple», Cahiers du Bureau universitaire de recherche opérationnelle Série Recherche, vol. 42, p. 3–68.

Fan, J. 1997, «Comments on «wavelets in statistics : A review» by a. antoniadis», Journal of the Italian Statistical Society, vol. 6, n° 2, p. 131.

Geisser, S. 1975, «The predictive sample reuse method with applications», Journal of the American statistical Association, vol. 70, n° 350, p. 320–328.

Geman, S., E. Bienenstock et R. Doursat. 1992, «Neural networks and the bias/variance dilemma», Neural computation, vol. 4, n° 1, p. 1–58.

Genuer, R., J.-M. Poggi et C. Tuleau-Malot. 2010, «Variable selection using random forests», Pattern Recognition Letters, vol. 31, n° 14, p. 2225–2236.

Genuer, R., J.-M. Poggi et C. Tuleau-Malot. 2015, «Vsurf : an r package for variable selection using random forests», The R Journal.

van Ginkel, J. R. et P. M. Kroonenberg. 2017, «Evaluation of multiple-imputation procedures for three-mode component models», Journal of Statistical Computation and Simulation, vol. 87, n° 16, p. 3059–3081.

Golub, G. H., M. Heath et G. Wahba. 1979, «Generalized cross-validation as a method for choosing a good ridge parameter», Technometrics, vol. 21, n° 2, p. 215–223.

Hanafi, M., S. S. Ouertani, J. Boccard, G. Mazerolles et S. Rudaz. 2015, «Multi-way pls regression : Monotony convergence of tri-linear pls2 and optimality of parameters», Computational Statistics & Data Analysis, vol. 83, p. 129–139.

Harshman, R. 1970, «Foundations of the parafac procedure : Models and conditions for an "explanatory" multi-modal factor analysis», UCLA Working Papers in Phonetics, vol. 16.

Hastie, T., R. Mazumder, J. D. Lee et R. Zadeh. 2015, «Matrix completion and low-rank svd via fast alternating least squares», Journal of Machine Learning Research, vol. 16, n° 1, p. 3367–3402. URL http://jmlr.org/papers/volume16/hastie15a/hastie15a.pdf.

Hoerl, A. E. et R. W. Kennard. 1970, «Ridge regression : Biased estimation for nonorthogonal problems», Technometrics, vol. 12, n° 1, p. 55–67.

Hotelling, H. 1933, «Analysis of a complex of statistical variables into principal components.», Journal of educational psychology, vol. 24, n° 6, p. 417.

Hotelling, H. 1936, «Relations between two sets of variates», Biometrika, vol. 28, n° 3/4, p. 321–377.

Hotelling, H. 1957, «The relations of the newer multivariate statistical methods to factor analysis», British Journal of Statistical Psychology, vol. 10, n° 2, p. 69–79.

Hubert, M., J. Van Kerckhoven et T. Verdonck. 2012, «Robust parafac for incomplete data», Journal of Chemometrics, vol. 26, n° 6, p. 290–298.

Husson, F. et J. Josse. 2013, «Handling missing values in multiple factor analysis», Food quality and preference, vol. 30, n° 2, p. 77–85.

Jensen, J. L. W. V. et collab.. 1906, «Sur les fonctions convexes et les inégalités entre les valeurs moyennes», Acta mathematica, vol. 30, p. 175–193.

Johnstone, I. M. et A. Y. Lu. 2004, «Sparse principal components analysis», Unpublished manuscript, vol. 7.

Johnstone, I. M. et A. Y. Lu. 2009, «On consistency and sparsity for principal components analysis in high dimensions», Journal of the American Statistical Association, vol. 104, n° 486, p. 682–693.

Jolliffe, I. T., N. T. Trendafilov et M. Uddin. 2003, «A modified principal component technique based on the lasso», Journal of computational and Graphical Statistics, vol. 12, n° 3, p. 531–547.

Josse, J. et F. Husson. 2012, «Handling missing values in exploratory multivariate data analysis methods», Journal de la Société Française de Statistique, vol. 153, n° 2, p. 79–99.

Josse, J. et F. Husson. 2016, «missmda : a package for handling missing values in multivariate data analysis», Journal of Statistical Software, vol. 70, n° 1, p. 1–31.

Josse, J., F. Husson et J. Pagès. 2009, «Gestion des données manquantes en analyse en composantes principales», Journal de la Société Française de Statistique, vol. 150, n° 2, p. 28–51.

Kiers, H. A. 1997, «Weighted least squares fitting using ordinary least squares algorithms», Psychometrika, vol. 62, n° 2, p. 251–266.

Koltchinskii, V. 2009, «Sparsity in penalized empirical risk minimization», dans Annales de l'IHP Probabilités et statistiques, vol. 45, p. 7–57.

Krauthgamer, R., B. Nadler, D. Vilenchik et collab.. 2015, «Do semidefinite relaxations solve sparse pca up to the information limit ?», The Annals of Statistics, vol. 43, n° 3, p. 1300–1322.

Kroonenberg, P. M. 1983, Three-mode principal component analysis : Theory and applications, vol. 2, DSWO press.

Kruskal, J. B. 1977, «Three-way arrays : rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics», Linear algebra and its applications, vol. 18, n° 2, p. 95–138.

Lê Cao, K.-A., I. González et S. Déjean. 2009, «integromics : an r package to unravel relationships between two omics data sets», Bioinformatics, vol. 25, n° 21, p. 2855–2856.

Lê Cao, K.-A., D. Rossouw, C. Robert-Granié et P. Besse. 2008, «A sparse pls for variable selection when integrating omics data», Statistical applications in genetics and molecular biology, vol. 7, n° 1.

LeCun, Y., L. Bottou, Y. Bengio, P. Haffner et collab.. 1998, «Gradient-based learning applied to document recognition», Proceedings of the IEEE, vol. 86, n° 11, p. 2278–2324.

Liquet, B., P. L. de Micheaux, B. P. Hejblum et R. Thiébaut. 2015, «Group and sparse group partial least square approaches applied in genomics context», Bioinformatics, vol. 32, n° 1, p. 35–42.

Lohmöller, J.-B. 2013, Latent variable path modeling with partial least squares, Springer Science & Business Media.

Lorenzo, H., M.-P. Ellies-Oury, C. Denoyelle, J. Saracco et B. Picard. 2019a, «An original methodology for the selection of biomarkers of tenderness in five different muscles», Foods, vol. 8, n° 6, p. 206.

Lorenzo, H., J. Saracco et R. Thiébaut. 2019b, «Supervised learning for multi-block incomplete data», arXiv preprint arXiv :1901.04380.

Louwerse, D., A. K. Smilde et H. A. Kiers. 1999, «Cross-validation of multiway component models», Journal of Chemometrics : A Journal of the Chemometrics Society, vol. 13, n° 5, p. 491–510.

van der Maaten, L. et G. Hinton. 2008, «Visualizing data using t-sne», Journal of machine learning research, vol. 9, n° Nov, p. 2579–2605.

Mallows, C. L. 1973, «Some comments on c p», Technometrics, vol. 15, n° 4, p. 661–675.

Marquardt, D. W. 1963, «An algorithm for least-squares estimation of nonlinear parameters», Journal of the society for Industrial and Applied Mathematics, vol. 11, n° 2, p. 431–441.

Mazumder, R., T. Hastie et R. Tibshirani. 2010, «Spectral regularization algorithms for learning large incomplete matrices», Journal of machine learning research, vol. 11, n° Aug, p. 2287–2322.

Nesterov, Y. 2013, «Gradient methods for minimizing composite functions», Mathematical Programming, vol. 140, n° 1, p. 125–161.

Osten, D. W. 1988, «Selection of optimal regression models via cross-validation», Journal of Chemometrics, vol. 2, n° 1, p. 39–48.

Pearson, K. 1901, «On lines and planes of closest fit to systems of point in space», Philosophical Magazine, vol. 2, p. 559–572.

Penrose, R. 1956, «On best approximate solutions of linear matrix equations», dans Mathematical Proceedings of the Cambridge Philosophical Society, vol. 52, Cambridge University Press, p. 17–19.

L'Hermier des Plantes, H. 1976, «Structuration des tableaux à trois indices de la statistique», Université de Montpellier II, Thesis.

Qin, S. J., S. Valle et M. J. Piovoso. 2001, «On unifying multiblock analysis with application to decentralized process monitoring», Journal of chemometrics, vol. 15, n° 9, p. 715–742.

R Core Team. 2019, R : A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org.

Rao, C. R., S. K. Mitra et collab.. 1972, «Generalized inverse of a matrix and its applications», dans Proceedings of the Sixth Berkeley Symposium on Mathematical Statistics and Probability, Theory of Statistics, vol. 1, The Regents of the University of California, p. 601–620.

Rennie, J. D. et N. Srebro. 2005, «Fast maximum margin matrix factorization for collaborative prediction», dans Proceedings of the 22nd international conference on Machine learning, ACM, p. 713–719.

Rosset, S., J. Zhu et T. Hastie. 2004, «Boosting as a regularized path to a maximum margin classifier», Journal of Machine Learning Research, vol. 5, n° Aug, p. 941–973.

Rothman, A. J., E. Levina et J. Zhu. 2009, «Generalized thresholding of large covariance matrices», Journal of the American Statistical Association, vol. 104, n° 485, p. 177–186.

Roweis, S. T. 1998, «Em algorithms for pca and spca», dans Advances in neural information processing systems, p. 626–632.

Rubin, D. B. 1976, «Inference and missing data», Biometrika, vol. 63, n° 3, p. 581–592.

Schafer, J. L. et J. W. Graham. 2002, «Missing data : our view of the state of the art.», Psychological methods, vol. 7, n° 2, p. 147.

Scholkopf, B. et A. J. Smola. 2001, Learning with kernels : support vector machines, regularization, optimization, and beyond, MIT press.

Schwarz, G. et collab.. 1978, «Estimating the dimension of a model», The annals of statistics, vol. 6, n° 2, p. 461–464.

Simon, N., J. Friedman, T. Hastie et R. Tibshirani. 2013, «A sparse-group lasso», Journal of Computational and Graphical Statistics, vol. 22, n° 2, p. 231–245.

Smilde, A. K., J. A. Westerhuis et R. Boque. 2000, «Multiway multiblock component and covariates regression models», Journal of Chemometrics : A Journal of the Chemometrics Society, vol. 14, n° 3, p. 301–331.

Stekhoven, D. J. et P. Bühlmann. 2011, «Missforest—non-parametric missing value imputation for mixed-type data», Bioinformatics, vol. 28, n° 1, p. 112–118.

Stone, M. 1974, «Cross-validatory choice and assessment of statistical predictions», Journal of the Royal Statistical Society : Series B (Methodological), vol. 36, n° 2, p. 111–133.

Tenenhaus, A., C. Philippe, V. Guillemot, K.-A. Le Cao, J. Grill et V. Frouin. 2014, «Variable selection for generalized canonical correlation analysis», Biostatistics, vol. 15, n° 3, p. 569–583.

Tenenhaus, A. et M. Tenenhaus. 2011, «Regularized generalized canonical correlation analysis», Psychometrika, vol. 76, n° 2, p. 257.

Tibshirani, R. 1996, «Regression shrinkage and selection via the lasso», Journal of the Royal Statistical Society. Series B (Methodological), p. 267–288.

Tikhonov, A. N. 1943, «On the stability of inverse problems», dans Dokl. Akad. Nauk SSSR, vol. 39, p. 195–198.

Tipping, M. E. et C. M. Bishop. 1999, «Probabilistic principal component analysis», Journal of the Royal Statistical Society : Series B (Statistical Methodology), vol. 61, n° 3, p. 611–622.

Tomasi, G. et R. Bro. 2005, «Parafac and missing values», Chemometrics and Intelligent Laboratory Systems, vol. 75, n° 2, p. 163–180.

Tomasi, G. et R. Bro. 2006, «A comparison of algorithms for fitting the parafac model», Computational Statistics & Data Analysis, vol. 50, n° 7, p. 1700–1734.

Vapnik, V. 1992, «Principles of risk minimization for learning theory», Advances in neural information processing systems, p. 831–838.

Vapnik, V. 1998, «Statistical learning theory. john wiley&sons», Inc., New York.

Vapnik, V. 1999a, The Nature of Statistical Learning Theory, Springer Science & Business Media.

Vapnik, V. N. 1999b, «An overview of statistical learning theory», IEEE transactions on neural networks, vol. 10, n° 5, p. 988–999.

Vapnik, V. N. et A. Y. Chervonenkis. 2015, «On the uniform convergence of relative frequencies of events to their probabilities», dans Measures of complexity, Springer, p. 11–30.

Venables, W. N. et B. D. Ripley. 2002, Modern Applied Statistics with S, 4e éd., Springer, New York. URL http://www.stats.ox.ac.uk/pub/MASS4, iSBN 0-387-95457-0.

Verhulst, P.-F. 1838, «Notice sur la loi que la population suit dans son accroissement», Corresp. Math. Phys., vol. 10, p. 113–126.

Wangen, L. et B. Kowalski. 1989, «A multiblock partial least squares algorithm for investigating complex chemical systems», Journal of chemometrics, vol. 3, n° 1, p. 3–20.

Westerhuis, J. A., P. M. Coenegracht et C. F. Lerk. 1997, «Multivariate modelling of the tablet manufacturing process with wet granulation for tablet optimization and in-process control», International journal of Pharmaceutics, vol. 156, n° 1, p. 109–117.

Westerhuis, J. A. et A. K. Smilde. 2001, «Deflation in multiblock pls», Journal of chemometrics, vol. 15, n° 5, p. 485–493.

Wold, S. 1984, «Three pls algorithms according to sw», dans Proc. : Symposium MULDAST (multivariate analysis in science and technology), p. 26–30.

Wold, S., N. Kettaneh-Wold et B. Skagerberg. 1989, «Nonlinear pls modeling», Chemometrics and intelligent laboratory systems, vol. 7, n° 1-2, p. 53–65.

Wold, S., H. Martens et H. Wold. 1983, «The multivariate calibration problem in chemistry solved by the pls method», dans Matrix pencils, Springer, p. 286–293.

Wold, S., M. Sjöström et L. Eriksson. 2001, «Pls-regression : a basic tool of chemometrics», Chemometrics and intelligent laboratory systems, vol. 58, n° 2, p. 109–130.

van den Wollenberg, A. L. 1977, «Redundancy analysis an alternative for canonical correlation analysis», Psychometrika, vol. 42, n° 2, p. 207–219.

Yuan, M. et Y. Lin. 2006, «Model selection and estimation in regression with grouped variables», Journal of the Royal Statistical Society : Series B (Statistical Methodology), vol. 68, n° 1, p. 49–67.

Zhao, P. et B. Yu. 2006, «On model selection consistency of lasso», Journal of Machine learning research, vol. 7, n° Nov, p. 2541–2563.

Zou, H. 2006, «The adaptive lasso and its oracle properties», Journal of the American statistical association, vol. 101, n° 476, p. 1418–1429.

Zou, H. et T. Hastie. 2005, «Regularization and variable selection via the elastic net», Journal of the royal statistical society : series B (statistical methodology), vol. 67, n° 2, p. 301–320.

Zou, H., T. Hastie et R. Tibshirani. 2006, «Sparse principal component analysis», Journal of computational and graphical statistics, vol. 15, n° 2, p. 265–286.

Résumé

Analyse supervisée multibloc en grande dimension

L'apprentissage statistique consiste à apprendre à partir de données mesurées dans un échantillon d'individus et cherche à prédire la grandeur d'intérêt chez un nouvel individu. Dans le cas de la vaccination, ou dans d'autres cas dont certains présentés dans ce manuscrit, le nombre de variables mesurées dépasse le nombre d'individus observés, c'est un cas dégénéré d'analyse statistique qui nécessite l'utilisation de méthodes spécifiques. Les propriétés des algorithmes de régularisation permettent de gérer ces cas. Il en existe plusieurs types en fonction de la structure des données considérées et du problème qui sont étudiés.

Dans le cas de ce travail, l'objectif principal a été d'utiliser l'information disponible à l'issue de décompositions en éléments propres des matrices de covariances transformées via un opérateur de seuillage doux. Cette solution est particulièrement peu coûteuse en termes de temps de calcul et permet la sélection des variables d'intérêt.

Nous nous sommes centrés sur les données qualifiées d'hétérogènes, c'est à dire issues de jeux de données qui sont provenant de sources ou de technologies distinctes. On parle aussi de données multiblocs. Les coûts d'utilisation de certaines technologies pouvant être prohibitifs, il est souvent choisi de ne pas acquérir certaines données sur l'ensemble d'un échantillon, mais seulement sur un sous-échantillon d'étude. Dans ce cas, le jeu de données se retrouve amputé d'une partie non négligeable de l'information. La structure des données associée à ces défauts d'acquisition induit une répartition elle-même multibloc de ces données manquantes, on parle alors de données manquantes par blocs. Le second objectif de notre méthode est de gérer ces données manquantes par blocs en s'appuyant sur l'information à prédire, ceci dans le but de créer un modèle prédictif qui puisse gérer les données manquantes aussi bien pour les données d'entraînement que pour celles de test. Cette méthode emprunte au seuillage doux afin de sélectionner les variables d'intérêt et ne nécessite que deux paramètres à régler qui sont le nombre de composantes et le nombre de variables à sélectionner parmi les covariables. Ce paramétrage est classiquement réalisé par validation croisée.

La méthode développée a fait l'objet de simulations la comparant aux principales méthodes existantes. Elle montre d'excellents résultats en prédiction et en termes de temps de calcul. Elle a aussi été appliquée à plusieurs jeux de données.

Mots clés : apprentissage statistique, grande dimension, multibloc, données manquantes, sélection de variable, science des données

Abstract

Supervised analysis of high dimensional multiblock data

Statistical learning objective is to learn from observed data in order to predict the response for a new sample. In the context of vaccination, the number of features is higher than the number of individuals. This is a degenerate case of statistical analysis which needs specific tools. The regularization algorithms can deal with those drawbacks. Different types of regularization methods can be used which depends on the data set structure but also upon the question.

In this work, the main objective was to use the available information with soft-thresholded empirical covariance matrix estimations through SVD decompositions. This solution is particularly efficient in terms of variable selection and computation time.

Heterogeneous typed data sets (coming from different sources and also called multiblock data) were at the core of our methodology. Since some data set generations are expensive, it is common to down sample the population acquiring some types of data. This leads to multi-block missing data patterns. The second objective of our methodology is to deal with those missing values using the response values. But the response values are not present in the test data sets and so we have designed a methodology which permits to consider both the cases of missing values in the train or in the test data sets.

Thanks to soft-thresholding, our methodology can regularize and select features. This estimator needs only two parameters to be fixed which are the number of components and the maximum number of features to be selected. The corresponding tuning is performed by cross-validation.

According to simulations, the proposed method shows very good results comparing to benchmark methods, especially in terms of prediction and computation time. This method has also been applied to several real data sets associated with vaccine, thomboembolic and food researches.

Keywords: statistical learning, high dimension, multiblock, missing values, variable selection, data science

Discipline : Santé publique – option Biostatistique
Laboratoire : Unité INSERM U1219, Bordeaux Population Health center - INRIA - Université de Bordeaux
146 rue Léo Saignat 33076 Bordeaux, FRANCE