UCLouvain

FACULTE DES SCIENCES

Faculté des sciences

Présentation des méthodes DIABLO, MINT et MOFA+ pour l'intégration des données omiques multi-blocs

Auteur : Lionel Panneel
Co-promotrice : Bernadette Govaerts
Co-promoteur : Laurent Gatto
Lecteurs : Rainer von Sachs - Jérôme Ambroise
Année académique 2021-2022

Master en science des données, orientation statistique, à finalité spécialisée

Je remercie Monsieur Laurent Gatto, co-promoteur de ce travail, pour ses lectures et relectures ainsi que pour les nombreux éclaircissements qu'il m'a fournis et ses remarques constructives, qui m'ont permis d'améliorer la qualité de ce travail.

Je remercie également Madame Bernadette Govaerts, co-promotrice de ce mémoire, pour sa patience, ses encouragements, ses conseils judicieux et ses explications toujours pédagogiques. Je la remercie particulièrement de m'avoir donné le goût aux statistiques en 2017, alors que je n'avais aucun bagage scientifique...

Finalement, je m'en voudrais de ne pas remercier ma compagne. La reconversion professionnelle, le retour à l'université et l'éducation d'un enfant sont trois projets passionnants pour lesquels il vaut mieux être bien épaulé lorsqu'on les affronte en même temps...

Table des matières

Chapitre 1

Introduction

La deuxième moitié du XX^{e} siècle a vu l'avènement d'un nouveau domaine d'expertise : l'analyse des données omiques. Cette appellation fait référence à différentes études biologiques portant le suffixe "omique", telles que la génomique, la transcriptomique ou la protéomique, par exemple. En d'autres termes, l'étude du vivant, chaque spécialité s'intéressant à un niveau particulier : les gènes, l'ADN, les protéines, etc.

Ce gain d'intérêt croissant pour les données omiques a été facilité par l'incroyable évolution technologique qui a eu lieu dans ce domaine au cours des 70 dernières années. Des techniques capables de capturer de plus en plus d'information de ce type ont vu le jour et leur coût a parfois diminué, favorisant leur utilisation. Finalement, l'avènement d'internet a fortement contribué au développement et au partage de ce type de données, les datasets étant disponible sur la toile et n'étant plus réservés uniquement aux professionnels de la santé.

Les jeux de données omiques sont caractérisés par un nombre très important de variables et, parallèlement, un nombre souvent limité d'observations, rendant certaines techniques statistiques inefficaces, voire impossible à appliquer. Des techniques omiques spécifiques ont alors été développées pour remédier à ces inconvénients.

L'utilisation des données omiques *multi-blocs* est plus récente et se caractérise par l'utilisation conjointe de plusieurs datasets, appelés *blocs*, dans une seule analyse. L'hypothèse de l'analyse multi-blocs est qu'il y a plus d'information, ou une information plus riche, dans plusieurs jeux de données que dans un seul. Dans un premier temps, un chapitre sera consacré à la démontration de l'intérêt des données omiques et aux outils nécessaires pour comprendre les notions qui seront développées par la suite (Chapitre 2). L'utilisation de ces datasets en une seule analyse engendre cependant d'autres problèmes, qui seront expliqués dans le chapitre 2.3.4.

Ces nouveaux challenges ont nécessité le développement de nombreuses nouvelles méthodes.

Dans ce travail, trois méthodes permettant l'analyse de données omiques multi-blocs ont été sélectionnées et seront approfondies :

• Dans un premier temps, la méthode DIABLO [21], proposée en 2016 par l'équipe MixOmics sera vue en détail (Chapitre 3). Il s'agit d'une méthode adaptée à l'intégration horizontale de datasets, où chaque bloc correspond à une modalité différente et où les données sont communes aux différents jeux de données.

• Ensuite, ce travail présentera la méthode MINT [20], également proposée par l'équipe MixOmics en 2017 (Chapitre 4). Cette méthode est adaptée à l'intégration verticale de jeux de données omiques, où chaque bloc correspond à une étude différente mais où les variables sont identiques entre les blocs.

• Finalement, MOFA+ sera la dernière méthode parcourue dans ce mémoire (Chapitre 5). Il s'agit de la méthode la plus récente des trois, proposée en 2020 par Argelaguet et al [5]. Cette méthode est la plus flexible des trois car elle est adaptée à l'intégration horizontale de blocs de données mais également verticale ou dans les deux sens en même temps.

Dans les chapitres 3.3, 4.3 et 5.3, les méthodes seront expliquées dans leur aspect mathématique et algorithmique. L'interprétation des principales sorties graphiques proposées dans le package de chacune des méthodes sera donnée dans les Chapitres 3.4.3, 4.4.3 et 5.4.3.

Le chapitre 6 sera consacré à la mise en pratique de ces méthodes via deux études de cas et l'interprétation de leurs résultats. Dans ces études de cas, seules les sorties graphiques les plus pertinents seront reprises dans ce travail. L'intégralité des graphiques produits pourra toutefois être retrouvé sur le Git Hub à l'adresse suivante : https ://github.com/LionelPanneel/multi-omics. Les méthodes sélectionnées sont parfois fort différentes, ne répondent pas toujours aux même questions et n'ont pas toujours le même objectif. Toutefois, dans la mesure du possible, une comparaison des méthodes et de leurs sorties graphiques sera proposée.

Finalement, la dernière partie de ce travail (Chapitre 7) tentera de récapituler et de synthétiser les avantages et inconvénients, les ressemblances et les dissimilarités de chaque méthode.

Chapitre 2

Outils et méthodes

2.1 Annotations

Cette section reprend les différentes annotations utilisées dans ce travail.

Par convention, les matrices et vecteurs sont représentés par un caractère gras. Les matrices sont identifiées par une lettre capitale tandis que les vecteurs et les indices sont représentés par une lettre en bas-de-casse.

Abréviation	**Description**
q	Indice d'identification de la modalité des données (bloc horizontal)
m	Indice d'identification de l'étude de données (bloc vertical)
h	Indice d'identification de la composante latente
i, j, k	Indices utilisés selon les besoins, dans certaines méthodes

TABLE 2.1: Abréviation des indices et exposants utilisés

Convention	Description	Remarques
$\mathbf{X}^{(m,q)}$	Désigne le bloc de l'étude (ligne) m et de la modalité (colonne) q	
$\mathbf{X}^{(.,q)}$ ou $\mathbf{X}^{(q)}$	Désigne le bloc de la modalité (colonne) q	En intégration horizontale.
$\mathbf{X}^{(m,.)}$ ou $\mathbf{X}^{(m)}$	Désigne le bloc de l'étude (ligne) m	En intégration verticale
$\mathbf{X}_h$	Désigne la matrice $\mathbf{X}$ déflatée selon la composante latente h	
$\mathbf{v}^{(m,q)}$	Désigne le vecteur (de scores ou de loadings) partiel spécifique au bloc mq	
$\mathbf{v}_h$	Désigne le vecteur (de scores ou de loadings) $\mathbf{v}$ associé à la composante latente h.	
$\mathbf{X}_{[i,j]}$	Désigne l'élément i, j de la matrice $\mathbf{X}$	
$S(x, y)$	Opérateur de soft-thresholding	
$\langle x \rangle$	$\mathbb{E}_q[x]$	Utilisé au Chapitre 5.3.6
$\nabla F(x)$	Gradient de la fonction dérivable $F(x)$	Utilisé au Chapitre 5.3
$\mathbb{I}_0$	Fonction delta de Dirac	Utilisé au Chapitre 5.3

TABLE 2.2: Conventions

Abréviation	Description	Remarques
Q	Nombre de modalités (blocs horizontaux) différentes	
M	Nombre d'études (blocs verticaux) différentes	
N	Taille totale de l'échantillon	
$N^{(m,q)}$	Taille de l'échantillon dans le bloc $\mathbf{X}^{(m,q)}$	$\sum_{m=1}^{M} N^{(m,q)} = N$
P	Nombre total de variables	
$P^{(m,q)}$	Nombre de variables du bloc $\mathbf{X}^{(m,q)}$	$\sum_{q=1}^{Q} P^{(m,q)} = P$
H	Nombre total de composantes latentes	
G	Nombre de classes de la variable d'intérêt $\mathbf{Y}$	
λ_h	Paramètre de pénalisation ℓ_1 (LASSO) associé au nombre de coefficients non nuls dans $\mathbf{a}_h$	
$l_h^{(q)}$	Contrainte ℓ_1 (LASSO) utilisée pour déterminer $\lambda_h^{(q)}$	Utilisé au Chapitre 3.3
$\theta^{(q)}$	Paramètre lié à la matrice "de sparsité" $\mathbf{S_A^{(q)}}$	Utilisé au Chapitre 5.3
$\theta^{(m)}$	Paramètre lié à la matrice "de sparsité" $\mathbf{S_T}^{(m)}$	Utilisé au Chapitre 5.3
$\alpha^{(q)}$	Paramètre lié à la matrice $\mathbf{\hat{A}^{(q)}}$	Utilisé au Chapitre 5.3
$\alpha^{(m)}$	Paramètre lié à la matrice $\mathbf{\hat{T}^{(m)}}$	Utilisé au Chapitre 5.3
$\phi^{(m)}$	Notation générale désignant les paramètres variationnels $\hat{\mathbf{T}}^{(m)}$ et $\mathbf{S}_{\mathbf{T}}^{(m)}$	Utilisé au Chapitre 5.3.5
ψ	Notation générale désignant les paramètres variationnel $\tau^{(m,q)}$, $\hat{\mathbf{A}}^{(q)}$, $\alpha^{(q)}$, $\theta^{(q)}$, $\mathbf{S}_{\mathbf{A}}^{(q)}$, $\alpha^{(m)}$ et $\theta^{(m)}$	Utilisé au Chapitre 5.3.5
ρ	Paramètre de pas lié à l'algorithme de gradient ascent	Utilisé au Chapitre 5.3
B	Taille totale de l'échantillon du minibatch $\mathcal{B}$	Utilisé au Chapitre 5.3.5
τ	Paramètre lié à la précision des résidus ϵ.	

TABLE 2.3: Abréviation des constantes

Abréviation	Description	Dimension
$\mathbf{X}$	Matrice des variables explicatives	$\mathbf{X} \in \mathbb{R}^{N \times P}$
$\mathbf{Y}$	Matrice de la variable d'intérêt (catégorielle)	$\mathbf{Y} \in \mathbb{R}^{N \times 1}$
$\mathbf{T}$	Matrice des vecteurs de scores	$\mathbf{T} \in \mathbb{R}^{N \times H}$
$\mathbf{A}$	Matrice des vecteurs de loadings	$\mathbf{A} \in \mathbb{R}^{P \times H}$
$\mathbf{C}$	Matrice de "design" définissant la corrélation entre les blocs	$\mathbf{C} \in \mathbb{R}^{Q \times Q}$
ϵ	Matrice des résidus	$\epsilon \in \mathbb{R}^{N \times P}$
$\mathbf{I}_k$	Matrice Identité	$\mathbf{I}_n \in \mathbb{R}^{k \times k}$
$\mathbf{S_T}$	Matrice de "sparsité" liée à la matrice $\mathbf{T}$ (utilisée au Chapitre 5.3)	$\mathbf{S_T} \in \mathbb{R}^{N \times H}$
$\mathbf{S_A}$	Matrice de "sparsité" liée à la matrice $\mathbf{A}$ (utilisée au Chapitre 5.3)	$\mathbf{S_A} \in \mathbb{R}^{P \times H}$
$\mathbf{a}_h$	Vecteur global de loadings de $\mathbf{X}$ selon la composante h	$\mathbf{a} \in \mathbb{R}^{P}$
$\mathbf{t}_h$	Vecteur global des scores de $\mathbf{X}$ selon la composante h	$\mathbf{t} \in \mathbb{R}^{N}$
$\mathbf{b}_h$	Vecteur global de loadings de $\mathbf{Y}$ selon la composante h	$\mathbf{b} \in \mathbb{R}^{G}$
$\mathbf{u}_h$	Vecteur global des scores de $\mathbf{Y}$ selon la composante h	$\mathbf{u} \in \mathbb{R}^{N}$
$\mathbf{W}$	Notation générale désignant les matrices des vecteurs latents $\mathbf{T}$ ou $\mathbf{A}$ (utilisée dans le Chapitre 5.3.4)	
$\mathcal{B}$	Matrice avec les observations du minibatch	$\mathcal{B} \in \mathbb{R}^{B \times P}$

TABLE 2.4: Annotation des matrices et vecteurs

Abréviation	Signification
ADN	**A**cide **D**ésoxyribo**N**ucléique
ARN	**A**cide **R**ibo**N**ucléique
ARNm	**A**cide **R**ibo**N**ucléique **m**essager
BER	**B**alanced **E**rror **R**ate
BCC	**B**ayesian **C**onsensus **C**lustering
DIABLO	**D**ata **I**ntegration **A**nalysis for **B**iomarker discovery using **L**atent c**O**mponent
ELBO	**E**vidence **L**ower **BO**und
FIB	Cellules **FIB**roblaste
GC	**G**as **C**hromatography
hESC	**C**ellule **S**ouche **E**mbryonnaire **h**umaine
hiPSC	**C**ellule **S**ouche **P**luripotente **h**umaine **i**nduite
iNMF	**i**ntegrative **N**on-negative **M**atrix **F**actorization
LOGOCV	**L**eave **O**ne **G**roup **O**ut **C**ross **V**alidation
miARN	**mi**cro **A**cide **R**ibo**N**ucléique
MINT	**M**ultivariate **INT**egrative method
MOFA+	**M**ulti-**O**mics **F**actor **A**nalysis v2
MS	**M**ass **S**pectrometry
NIPALS	**N**onlinear estimation by **I**terative **PA**rtial **L**east **S**quares
NMR	**N**uclear **M**agnetic **R**esonance
PCA	**P**rincipal **C**omponent **A**nalysis
PLS	**P**artial **L**east **S**quares
q-PCR	**q**uantitative **P**olymerase **C**hain **R**eaction
sGCCA	**s**parse **G**eneralized **C**anonical **C**orrelation **A**nalysis
SNF	**S**imilarity **N**etwork **F**usion
TCGA	**T**he **C**ancer **G**enome **A**tlas
t-SNE	**t**-distributed **S**tochastic **N**eighbor **E**mbedding
UMAP	**U**niform **M**anifold **A**pproximation and **P**rojection

Table 2.5: Signification des abréviations utilisées

2.2 La diversité des données omiques

2.2.1 L'utilisation des données omiques

L'étude de données dites "omiques" consiste en l'analyse systématique et à haut débit d'une série de données biologiques comme l'ADN, les protéines ou les métabolites par exemple. Les études omiques les plus courantes sont la génomique, l'épigénomique, la transcriptomique, la protéomique et la métabolomique. Chaque étude s'intéresse à un domaine particulier et utilise des plateformes de collecte de données spécifiques, dont les plus fréquentes sont illustrées sur la Figure 2.1.

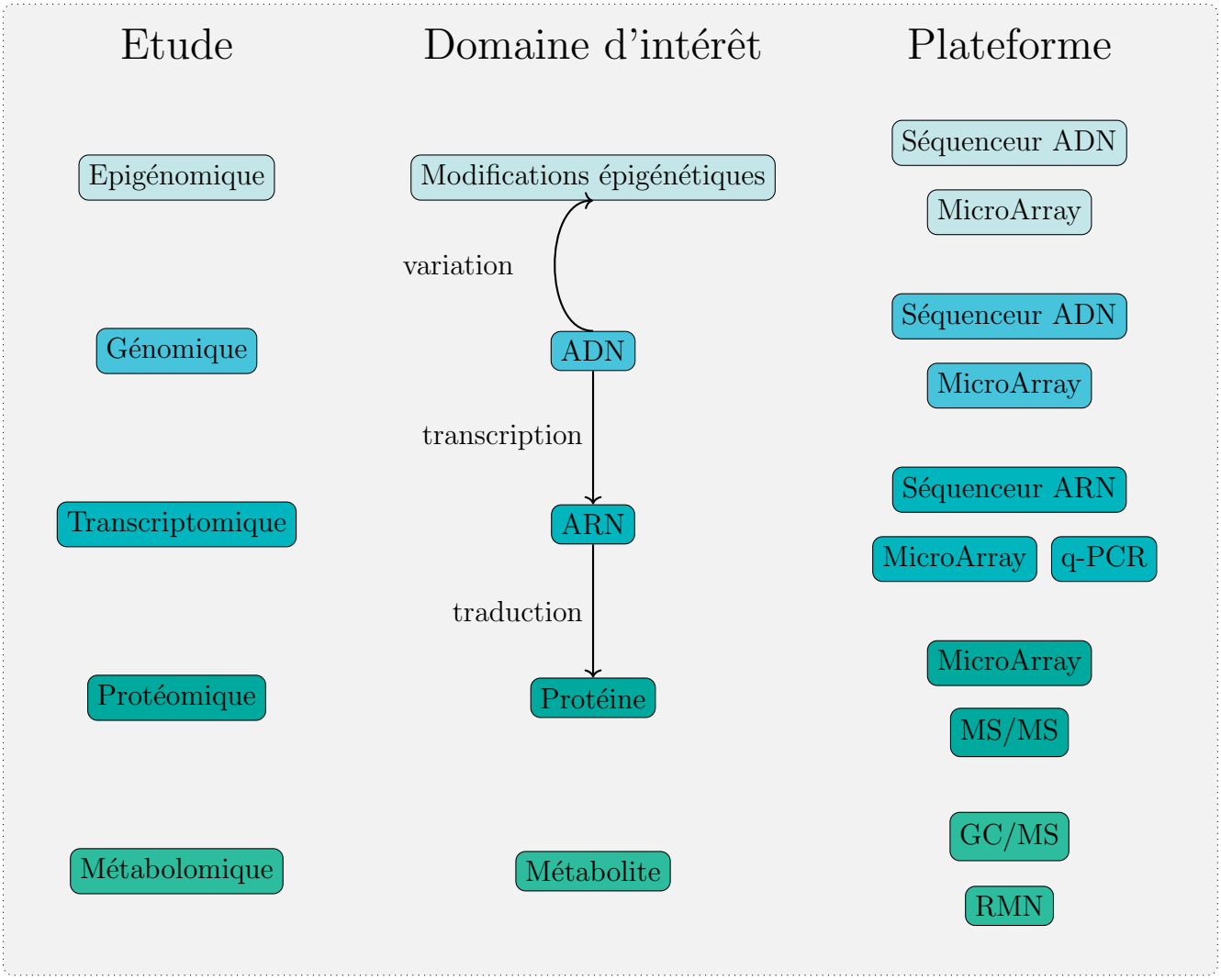


FIGURE 2.1: Principales études omiques, leurs domaines d'intérêt et les plateformes couramment utilisées

La Figure 2.2 illustre l'essor des publications scientifiques sur les principaux thèmes des données omiques durant la seconde partie du XX^{e} et le début du XXI^{e} siècle, en particulier après le lancement du *Human Genome Project*, en 1988 ou du programme *The Cancer Genome Atlas* en 2005. Cet essor peut s'expliquer, entre autre, par l'évolution technolo-

gique dans le domaine de la biologie. En effet, les équipements permettant l'enregistrement des données (spectrométrie de masse, séquençage ADN, microarrays, etc.) se sont diversifiés et perfectionnés. Le coût de certaines de ces technologies a été réduit, tout en permettant parallèlement d'enregistrer davantage d'informations. De plus, l'évolution technologique de l'outil informatique a permis de traiter des bases de données de plus grande ampleur et de les partager.

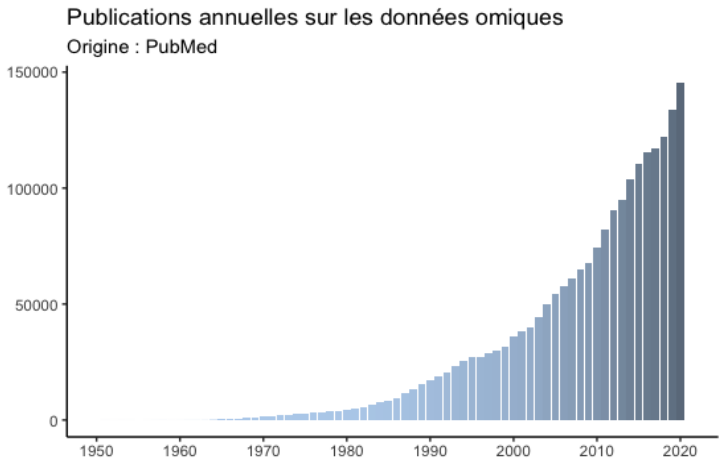


FIGURE 2.2: Publications annuelles sur les thèmes de la génomique, l'épigénomique, la transcriptomique, la protéomique et la métabolomique.

2.2.2 Principe de fonctionnement des plateformes les plus courantes

Comme on l'a vu dans la Figure 2.1, plusieurs plateformes de récolte de données sont utilisées dans le domaine des données omiques. L'idée de cette sous-section est de donner au lecteur la compréhension suffisante et nécessaire du principe de fonctionnement de chaque plateforme, ainsi qu'une idée du type de données obtenu. L'explication du principe de fonctionnement des plateformes est accompagné d'une illustration, lorsque c'est pertinent. Dans le cadre de l'intégration de données multi-blocs, le résultat obtenu par chaque plateforme constitue un bloc ou une matrice quantitative, représenté par $\mathbf{X}^{(q,m)}$, comme illustré plus tard sur la Figure 2.7.

- **q**uantitative **P**olymerase **C**hain **R**eaction (**q-PCR**)
  La q-PCR est utilisée depuis 1985 et permet de quantifier le niveau expression d'un seul gène. L'idée de la qPCR est qu'un gène va être copié artificiellement au cours d'un cycle d'amplification PCR. Le résultat va être lui-même reproduit un certain nombre de fois. On va ensuite calculer le nombre de cycles qu'il aura fallu pour observer l'abondance du transcrit à un certain niveau. Au moins il y a eu de cycles nécessaires, au plus cela signifie que le transcrit était déjà présent (et donc, que le gène était exprimé) dans l'échantillon de départ, et inversément.

- **Micro-array**

  Les micro-array, ou micro-damiers, sont apparus pour la première fois en 1997 et permettent d'analyser l'expression de nombreux gènes d'un même échantillon, en une seule fois. Les révolutions technologiques du micro-array sont la *miniaturisation* de l'appareil (les dimensions d'un micro-array sont généralement de 6 cm x 3 cm) et la *parallélisation* : chaque puce peut être vue comme une matrice de plusieurs lignes et colonnes, dont chaque élément contient l'information relative à l'objet étudié (niveau d'expression d'un gène, variation d'ADN, profil épigénomique, etc.). Il est à noter que l'utilisation du micro-array dans le cadre de la protéomique est moins fréquente.

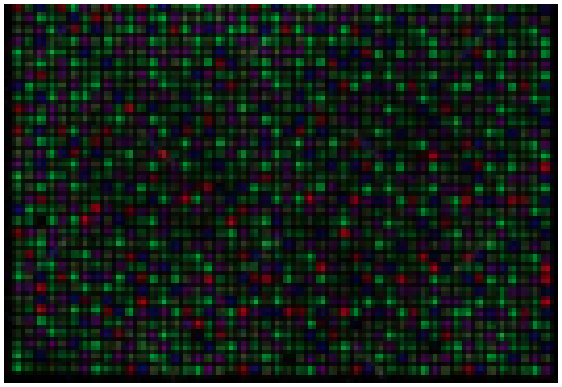


FIGURE 2.3: Illustration d'une plaque de micro-array [4]

- **Séquençage à haut débit (Séquenceur ADN/ARN)**

  La technique du RNA-seq permet de séquencer des brins d'ARNm. L'idée est qu'un ADN/ARNm va être multiplié (à la manière d'une PCR) puis qu'il va être séquencé en une suite de bases nucléiques (A, T/U, G, C) appelée *reads*. L'analyse via séquenceur ARN consiste à étudier l'abondance de ces reads et donc à déterminer le niveau d'expression des gènes. Le séquenceur ADN est davantage utilisé pour l'étude de la variation des gènes.

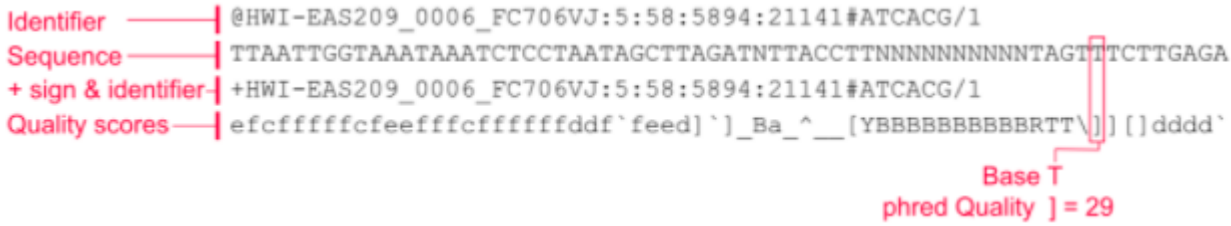


FIGURE 2.4: Exemple de données obtenue via séquenceur ARN (format FASTQ) [2].

- **La spectrométrie de masse**

  La spectrométrie de masse est utilisée pour l'identification de protéines ou de métabolites dans un échantillon. Dans le cas des protéines, l'analyse proprement dite étudie les peptides, qui sont des éléments résultant du clivage de la protéine. Les peptides sont séparés dans une colonne chromatographique et vont ensuite entrer dans le spectromètre de masse à un moment particulier en fonction d'un temps de rétention défini selon le

niveau de pH introduit dans la colonne chromatographique. Les peptides vont être ionisés et séparés en fonction de leur ratio m/z, où m est la masse d'un peptide et z la charge électrique donnée par le spectromètre. Un premier niveau de spectre est produit (MS1) avec le ratio m/z sur l'axe des abscisses et l'intensité sur l'axe des ordonnées. Le spectromètre de masse peut également fournir un second type de spectre (MS2), qui fragmente un pic d'intérêt dans le MS1. L'analyse du spectre MS2 permet d'identifier la "signature" d'un peptide en particulier et de retrouver la protéine à laquelle il appartient. La Figure 2.5 illustre les deux niveaux de spectrométrie obtenus grâce au spectromètre de masse.

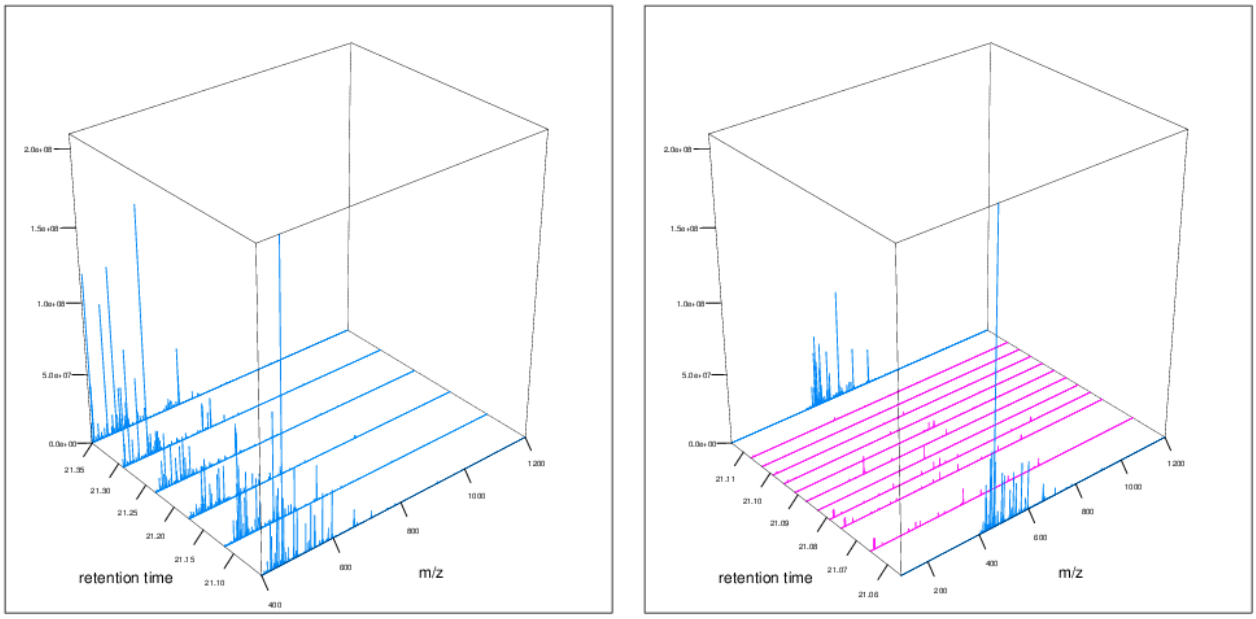


FIGURE 2.5: Schématisation des données de sorties d'un spectromètre de masse. Les spectres MS1 sont illustrés en bleu et les MS2 sont illustrés en fuschia [12]

- **N**uclear **M**agnetic **R**esonance (**RMN**)

  La résonnance magnétique nucléaire est une technique utilisée en métabolomique et dont le principe est que les échantillons (urine, sang, etc...) sont soumis à un champ magnétique intense. Ce champ magnétique externe induit un alignement du champ magnétique des molécules de l'échantillon dans la même direction, ou dans la direction opposée à la sienne. Lorsque l'on arrête le champ magnétique externe, certains protons des molécules de l'échantillon reprennent leur orientation originale, induisant une onde dont la fréquence peut être liée au type de molécule à laquelle ils appartiennent. Les données résultant de cette technique sont donc des spectres, dont certains pics peuvent signifier la présence de molécules typiques. Ce type d'analyse permet de reconnaître la structure et la quantité de molécules présentes dans l'échantillon observé.

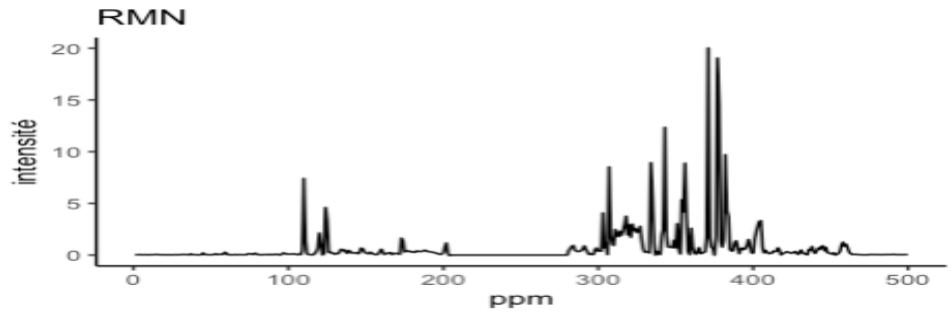


FIGURE 2.6: Illustration d'un spectre RMN

2.3 L'intérêt des données omiques multi-blocs

2.3.1 Considérations générales

Dans le domaine des études omiques, il est fréquent que les variables soient rangées en lignes et les observations en colonnes, contrairement à ce qui se fait habituellement dans le domaine statistique. C'est notamment de cette manière que sont représentés les données de la méthode MOFA+ dans les documents renseignés dans la bibliographie. Toutefois, dans un souci d'harmonisation de ce travail et sauf mention contraire, toutes les lignes des matrices feront référence aux observations de l'échantillon, et les colonnes aux variables.

Il y a également lieu de définir quelques termes qui seront régulièrement utilisés tout au long de ce document :

• Une *modalité* est définie comme une sortie spécifique d'un niveau omique d'analyse. Une analyse de protéomique et une analyse de transcriptomique sont donc deux exemples de modalités différentes. De même, une matrice de données de micro ARN et une autre d'ARN messager (toutes les deux étant l'objet de transcriptomique) sont deux autres modalités différentes. Ces modalités sont parfois appelée *vues* dans certains documents référencés en bibliographie. En général, les différentes *modalités* sont représentées par l'indice q dans ce travail.

• Dans ce mémoire, lorsqu'il est fait mention d'*études différentes*, cela signifie que les études en question s'intéressent aux mêmes modalités et aux mêmes variables, mais qu'elles sont indépendantes : elles ont été réalisées dans deux laboratoires différents, ou avec deux appareils différents, par exemple. Ces études sont parfois appelées *groupes* ou *conditions* dans les documents référencés en bibliographie, mais ces termes ne seront pas retenus pour éviter des confusions. En général, les *études* sont représentées par l'indice m dans ce travail.

• Deux types de vecteurs seront très largement utilisés tout au long de ce mémoire : les vecteurs de loadings et les vecteurs de scores. Les vecteurs de loadings sont des vecteurs où chaque élément représente un poids associé à une variable. Par calcul matriciel, les données d'origine et les vecteurs de loadings construisent des vecteurs de scores, qui peuvent être vus comme la projection des données dans un sous-espace. Les vecteurs de scores sont aussi appelés *facteurs* dans certains documents référencés en bibliographie. Les deux appellation

seront utilisées dans ce travail. Les vecteurs de loadings sont représentés par les lettres **a** ou **b** et les vecteurs de scores sont représentés par **t** ou **u**.

• Un vecteur est défini comme *global* s'il concerne l'ensemble des blocs considérés. A contrario, un vecteur est dit *local* ou *partiel* s'il est associé à un bloc en particulier.

2.3.2 Les types d'intégration possibles

L'intégration de données dites multi-omiques, ou données omiques multi-blocs, consiste en l'utilisation simultanée de plusieurs blocs de données omiques, chacun issus de différentes sources, au sein d'une même analyse. Selon le type d'intégration, les blocs peuvent représenter soit différentes études, soit différentes modalités.

L'intégration des données omiques multi-blocs se fait typiquement de manière horizontale ou verticale mais certaines techniques plus récentes permettent une intégration dans les deux sens, comme illustré sur la Figure 2.7. Le choix du type d'intégration dépend du but de l'analyse ou du jeu de données initial. Les méthodes utilisées sont spécifiquement adaptées à un ou plusieurs types d'intégration. La dénomination des types d'intégrations peut varier en fonction des sources bibliographiques mais les appellations suivantes seront utilisées dans ce travail :

- L'intégration **horizontale** (ou *N-integration*) se pratique sur des données où les blocs représentent des modalités différentes. Dans cette situation, les observations sont communes à tous les blocs mais ce sont les modalités, et donc les variables, qui changent d'un bloc à l'autre. Les méthodes DIABLO et MOFA+ sont adaptées à l'intégration horizontale.

- L'intégration **verticale** (ou *P-integration*) se pratique généralement pour augmenter la taille de l'échantillon d'une analyse. Dans cette situation, les blocs représentent des études différentes. La modalité et les variables sont donc identiques au sein de chaque bloc, mais ce sont les observations qui diffèrent. Les méthodes MINT et MOFA+ supportent ce type d'intégration.

- Certaines méthodes supportent simultanément l'intégration horizontale et verticale. Dans ce mémoire, on parlera alors d'intégration **dans les deux sens**. MOFA+ est la seule méthode qui sera parcourue dans le cadre de ce travail et qui supporte ce type d'intégration.

- Il est à noter que de très récentes méthodes permettent d'intégrer simultanément des blocs dans plus de deux dimensions, en ajoutant une dimension temporelle par exemple. On parlera alors d'intégration **multiple**. Ces méthodes ne seront cependant pas abordées en profondeur dans ce travail.

En fonction du type d'intégration, la variable d'intérêt **Y**, peut également être organisée en un ou plusieurs blocs.

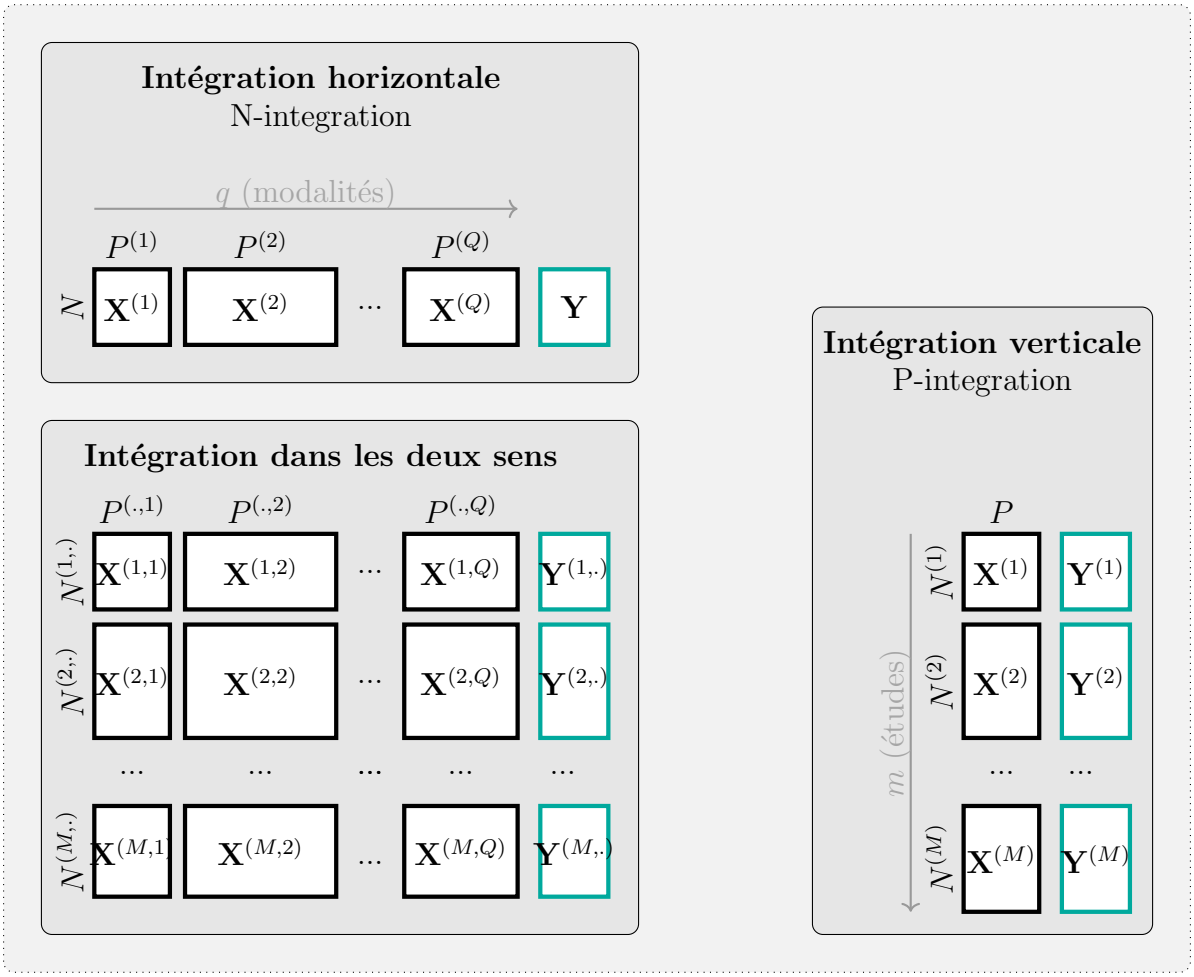


FIGURE 2.7: Les différents types d'intégration de données omiques multi-blocs supportés par les méthodes développées dans ce travail

Comme on l'a vu dans la Figure 2.1, les différentes modalités omiques sont interdépendantes : l'ADN étant transcrite en ARN, l'ARN étant lui-même traduit en protéine par exemple. Les analyses omiques multi-blocs permettent donc d'appréhender les interactions entre les différentes modalités. Ainsi, si le chercheur veut analyser la dépendance entre l'expression d'un certain gène et la production d'une protéine chez des patients atteints d'une maladie, il pourrait recourir à l'intégration de données issues de transcriptomique et de protéomique par exemple.

Il est à noter que deux types d'analyses omiques sont couramment utilisées : les analyses *bulk* et les analyses *single-cell*.

- L'analyse *bulk* est historiquement plus ancienne que l'analyse single-cell. Le principe est qu'une quantité importante de cellules est prélevée en une seule fois et que les données récoltées dans les datasets sont le résultat de la moyenne calculée sur cet amas de cellules. Une ligne dans la matrice $\mathbf{X}^{(m,q)}$ représente donc les moyennes des expressions des gènes calculées sur cet amas de cellules, par exemple. Ceci peut parfois poser problème car il arrive qu'un amas de cellules comprenne des éléments très hétérogènes. C'est particulièrement le cas avec certaines pathologies comme les cancers, où deux cellules

maligne et bénigne peuvent être présentes côte à côte.

Les trois méthodes parcourues dans ce travail (DIABLO, MINT et MOFA+) supportent des données provenant d'analyses bulk.

- Lorsqu'une analyse *single-cell* est réalisée, chaque cellule est isolée et analysée individuellement. Donc, chaque ligne de la matrice $\mathbf{X}^{(m,q)}$ représente les valeurs d'une cellule unique. Ce type d'extraction de données a l'avantage de pouvoir étudier des ensembles de cellules particulièrement hétérogènes. La mise en pratique de cette méthode est cependant plus compliquée que l'analyse bulk (et pas toujours réalisable) : la technique nécessite d'arriver à isoler les cellules, la quantité d'information disponible au sein d'une seule cellule est parfois trop faible pour multiplier les modalités à observer, les protocoles sont plus compliqués, etc. De plus, ce type d'extraction de données se caractérise par une quantité importante de données manquantes. Finalement, on peut noter que les analyses single-cell sont généralement caractérisées par un nombre d'observations très élevé, pouvant se mesurer en millions, contrairement aux analyses bulk.

  La méthode MOFA+ est spécifiquement destinée à l'utilisation de données provenant d'analyse single-cell, car elle permet de faire face à des jeux de données de plusieurs millions d'observations dans un temps raisonnable, tout en pouvant faire face à des données manquantes pour certaines modalités [5].

2.3.3 Les questions d'intérêt des études omiques

L'analyse de données omiques et, par conséquent, des données omiques multi-blocs, tente de répondre à de multiples questions qui peuvent être classifiées en trois catégories selon Subramanian et al. [22] :

1. **Classification : Le sous-typage d'un phénotype**

   Un *phénotype* est une variable d'intérêt biologique ou clinique : la présence d'une maladie, la couleur des yeux, etc. En termes statistiques, le phénotype représente souvent la variable d'intérêt.

   Dans le cas d'études supervisées, le sous-typage du phénotype a pour objectif de construire un modèle capable de retrouver les groupes de la variable d'intérêt $\mathbf{Y}$ d'un nouvel échantillon à partir du vecteur $\mathbf{x}$ des nouvelles observations. Le modèle recherché est donc celui qui minimise le taux d'erreur de classification des observations de ce nouvel échantillon. La première étude de cas de ce travail utilise la méthode DIABLO pour construire un modèle permettant de prédire le type de cancer du sein dont sont atteintes les observations. La seconde étude de cas utilise la méthode MINT pour construire un modèle permettant de prédire le type de cellule souche des observations.

2. **Features selection : La recherche de biomarqueurs**

   Un biomarqueur est un élément biologique mesurable et caractéristique d'un groupe du phénotype. La recherche de biomarqueurs consiste donc à identifier les variables les plus discriminantes, celles qui permettent d'identifier le plus efficacement les différents groupes de la variable d'intérêt $\mathbf{Y}$. L'optimisation du choix des biomarqueurs est d'un grand intérêt pour le scientifique et l'intégration d'études omiques multi-blocs peut apporter un avantage considérable par rapport aux analyses classiques, particulièrement en intégration horizontale. En effet, la discrimination d'un groupe du phénotype (la détection d'une maladie par exemple) pourrait être plus efficace en observant des biomarqueurs issus de blocs différents, plutôt que des biomarqueurs d'un seul et même bloc. Cette recherche de biomarqueurs a également un intérêt dans l'interprétation d'un modèle.

   Lorsque l'on travaille avec certaines plateformes (comme la RMN par exemple), les variables ne sont pas toujours des biomarqueurs car elles ne représentent qu'une valeur exprimée en ppm mais celles-ci permettent parfois de retrouver un biomarqueur qui est exprimé par cet indicateur (dans le cas de la RMN : des métabolites caractérisés par des pics spécifiques).

   On verra dans ce travail que les méthodes DIABLO et MOFA+ identifient toutes les deux le gène ZNF552 (Zinc Finger Protein 552) comme un des biomarqueurs particulièrement important dans la caractérisation du type de cancer du sein par exemple. Dans la deuxième étude de cas, les méthodes MINT et MOFA+ identifient toutes les deux le gène SOX2 (*SRY-box transcription factor 2*) comme biomarqueur utile à la discrimination du type de cellule souche. La recherche de biomarqueurs est l'objectif principal des méthodes DIABLO et MINT.

3. **La recherche de mécanismes biologiques sous-jacents**

   Certaines méthodes répondent à d'autres questions, plus spécifiques. Par exemple, les méthodes parcourues dans ce travail permettent de s'intéresser à la corrélation entre les différentes variables, les vecteurs de scores ou entre les biomarqueurs découverts. Il s'agit d'une question d'intérêt importante pour le biologiste car il peut comprendre la dépendance entre la présence d'une protéine et l'expression d'un gène par exemple.

4. **Le clustering**

   Un quatrième point pourrait être rajouté à ceux proposés par Subramanian et al. : le clustering.

   Dans le cas de méthodes non-supervisées, il n'est pas toujours possible de retrouver le sous-type du phénotype d'un nouvel échantillon puisque ces groupes sont inconnus dans les données d'origine. Par contre, ce type méthode peut se prêter au clustering : le but étant de regrouper les observations selon leurs similitudes. La méthode MOFA+ permet d'effectuer du clustering sur les données, même si ce n'est pas son objectif principal.

2.3.4 Les challenges spécifiques liés aux données omiques multi-blocs

Intrinsèquement, les données omiques se prêtent assez mal aux techniques classiques d'analyse statistique, pour plusieurs raisons :

- Ce type de données est caractérisée par un design où le nombre de variables est particulièrement important : de plusieurs dizaines (q-PCR) à plusieurs dizaines de milliers (micro-arrays). Ce nombre de variables s'additionne et augmente d'autant plus lorsque l'on procède à l'intégration horizontale de plusieurs modalités. Parallèlement, le nombre d'observations est généralement relativement faible (sauf pour les analyses single-cell), rendant inutilisables ou inefficaces la plupart des techniques statistiques classiques, notamment celles basées sur des régressions linéaires ou du clustering [13].

- Par ailleurs, certaines variables sont parfois très fortement corrélées entre elles au sein d'un même bloc, ce qui peut entraîner des problèmes de colinéarité. Dans ces conditions, la résolution d'une régression linéaire peut mener à des coefficients particulièrement extrêmes et instables [11].

En plus de ces problèmes, d'autres challenges spécifiques à l'intégration des données omiques multi-blocs peuvent se présenter :

- Dans le cas d'une intégration verticale, les blocs comparés proviennent généralement d'études ou de laboratoires différents. Des différences dans les protocoles, dans la méthodologie ou dans les appareils utilisés peuvent venir perturber la procédure d' intégration, entrainant une variance non désirable au sein des données.

- Etant donné l'interdépendance entre les différents niveaux omiques, il est fréquent que certaines variables recueillies au sein de différentes modalités d'une même analyse (en intégration horizontale) soient également fortement corrélées entre elles, augmentant le risque de colinéarité et des problèmes déjà mentionnés.

- Notons finalement que le problème de données manquantes est fréquent lors de ce type d'analyse et en particulier lors d'analyses single-cell ou pour certaines modalités, particulièrement en protéomique [26].

Toutes les méthodes présentées dans ce travail parviennent à faire face à ces différents challenges, mais ce n'est pas le cas de toutes les méthodes d'intégration de données omiques multi-blocs. Par exemple, les méthodes BCC (**B**ayesian **C**onsensus **C**lustering), SNF (**S**imilarity **N**etwork **F**usion) ou iNMF (**i**ntegrative **N**on-negative **M**atrix **F**actorization) ne supportent pas les données manquantes.

2.4 Les méthodes présentées dans ce travail

De nombreuses méthodes d'intégration de données omiques multi-blocs ont été développées depuis plusieurs années. En 2016, Bersanelli et al. proposaient de classifier ces méthodes en quatre catégories, selon deux critères combinés : les méthodes basées sur les *réseaux* et les méthodes *bayesiennes* [7].

Trois ans plus tard, en 2019, Subramanian et al. proposaient six classifications de ces méthodes [22]. Les méthodes étaient catégorisées en fonction de leur approche principale : les méthodes de similarité, de corrélation, de réseau, de fusion, les méthodes bayesiennes et multivariées, comme illustré en Figure 2.8.

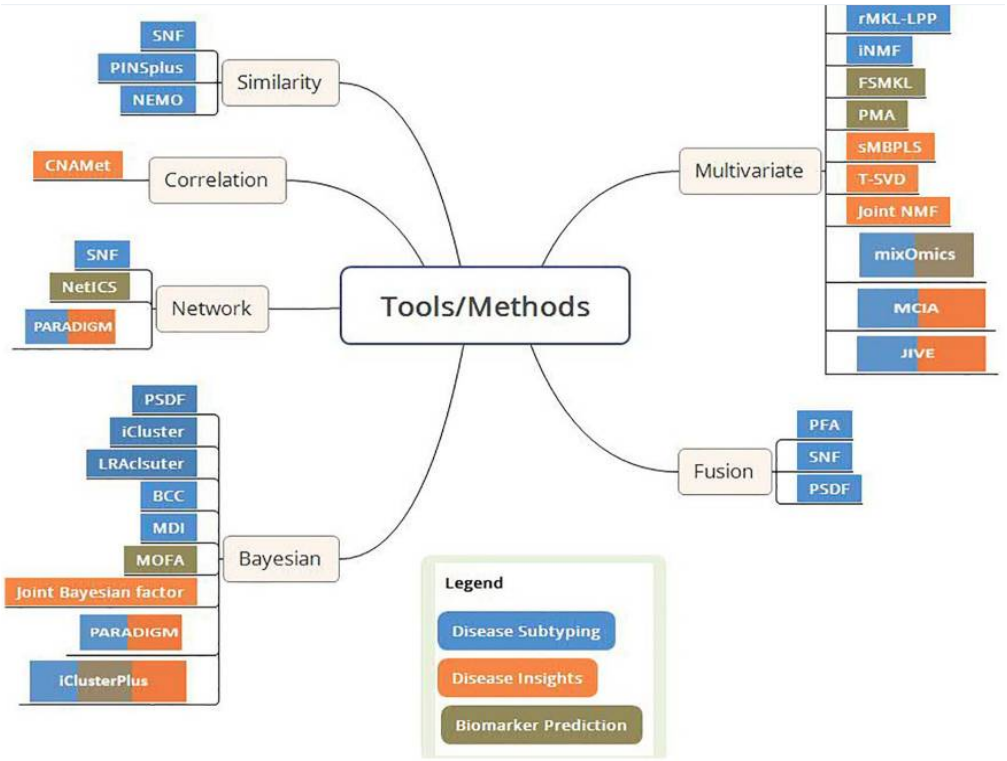


FIGURE 2.8: Classification des méthodes omiques multi-blocs selon Subramanian et al. [22]

Le choix des méthodes étudiées dans le cadre de ce travail s'est fait en respectant plusieurs critères :

• Les méthodes devaient disposer d'un package en *R* afin de pouvoir les tester sur des vrais jeux de données.

• Chaque méthode devait être capable de faire face à un problème de données manquantes.

• Parmi les méthodes sélectionnées, certaines devaient être supervisées et d'autres non-supervisées afin de comparer les résultats entre eux.

Trois méthodes ont donc été sélectionnées et sont explorées dans ce travail : les méthodes DIABLO, MINT et MOFA+. Les deux premières sont reprises sous l'intitulé *mixOmics* dans le schéma de Subramanian et al. Les approches principales de ces trois méthodes sont soit multivariées soit bayesienne. Dans les Chapitres 3.3, 4.3 et 5.3, les méthodes seront expliquées dans leur aspect mathématique et algorithmique. L'interprétation des principales sorties graphiques proposées dans le package de chacune des méthodes sera donnée dans les Chapitres 3.4.3, 4.4.3 et 5.4.3. Finalement, le Chapitre 6 sera consacré à la mise en pratique de ces méthodes via deux études de cas et l'interprétation de leurs résultats. Les méthodes sélectionnées sont parfois fort différentes, ne répondent pas toujours aux même questions et n'ont pas toujours le même objectif. Toutefois, dans la mesure du possible, une comparaison des méthodes et de leurs sorties graphiques sera proposée.

2.5 Jeux de données utilisés

2.5.1 TCGA

Le premier jeu de données provient initialement du TCGA Research Network (**T**he **C**ancer **G**enome **A**tlas) [25]. Ce projet a été lancé en 2005 et avait pour but de créer une carte, un "atlas" des profils génomiques des différents cancers de l'être humain. Les données enregistrées dans le cadre de ce projet ont été rendues publiques, ce qui a amélioré les compétences dans ce domaine et a contribué à l'augmentation des publications sur le sujet. Cette énorme base de données recense plus d'une trentaine de types de cancers différents et plus de 10.000 sous-types de cancers. Les données sont enregistrées via différentes plateformes : micro-arrays, séquenceurs ARN et ADN, etc.

Les données du TCGA qui seront utilisées dans la première étude de cas de ce travail ne sont qu'une toute petite partie des données du TCGA disponibles. Le jeu de données utilisé provient du package *mixOmics* et a notamment été utilisé pour la vignette de la méthode DIABLO. Il s'agit donc de données spécifiquement sélectionnées par l'équipe MixOmics et on peut s'attendre à ce que les résultats présentés pour cette méthode soient probants. Il est à noter qu'en plus de ces données disponibles, le TCGA dispose également de nombreuses données cliniques telles que le genre de l'individu, son âge, son statut de fumeur, etc. mais celles-ci ne seront pas exploitées lors de l'étude de cas.

Les caractéristiques du jeu de données utilisé sont les suivantes et sont illustrées par la Figure 2.9 :

- Les trois blocs utilisés représentent un dataset de protéomique et deux datasets de transcriptomique : micro ARN (miARN) et ARN messager (ARNm). Il s'agit de données provenant de RNA-seq et de micro-arrays.

- Le nombre de variables diffère dans chaque bloc mais les observations sont identiques : [150 x 184] pour le miARN, [150 x 200] pour les données d'ARNm et [150 x 142] pour les données de protéomique dans le training set.

- La variable d'intérêt identifie trois sous-types du cancer du sein : Basal, Her2 et LumA. La proportion de ces trois cancers sont respectivement de 30%, 20% et 50%. Cette variable d'intérêt n'est indispensable que dans le cas de la méthode DIABLO puisque la méthode MOFA+ est une méthode non-supervisée. Dans cette dernière méthode, cette variable sert cependant à la visualisation de certaines sorties graphiques mais pas pour le calcul du modèle.

Etant donné que les observations sont identiques dans les trois blocs, ce jeu est adapté à l'intégration horizontale. Ces données seront analysées au travers des méthodes DIABLO et MOFA+.

Le test set utilisé pour cette analyse est composé de 70 autres observations et des mêmes variables pour les blocs d'ARN messager et de micro ARN. Les données de protéomique sont manquantes. La distribution des groupes de la variable d'intérêt de ce test set correspond à celle du training set.

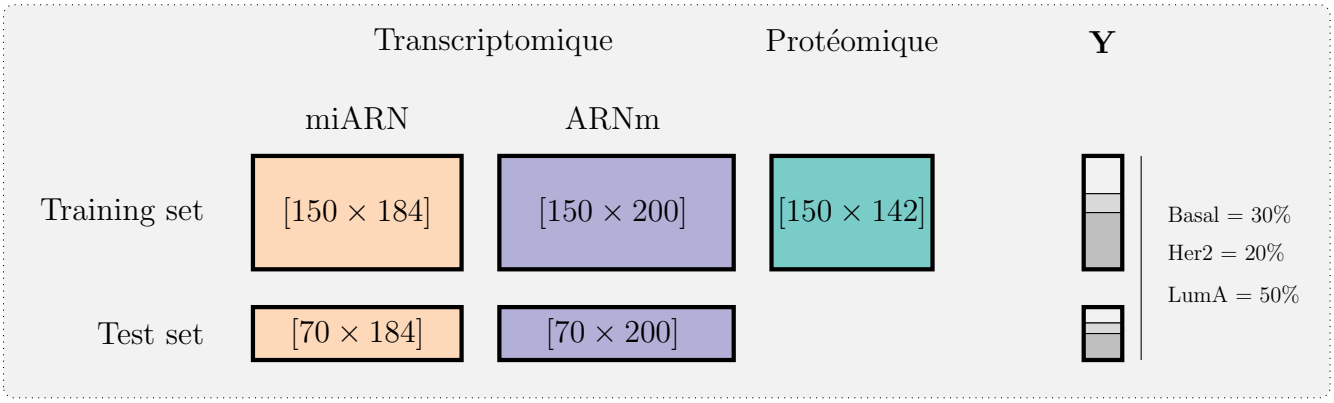


FIGURE 2.9: Illustration du training et du test sets fournis par le package *mixOmics* pour l'étude TCGA

2.5.2 Stemformatics

Les données obtenues proviennent initialement de la plateforme informatique Stemformatics [27] et sont disponibles via le package *mixOmics*. Il s'agit également de données sélectionnées par l'équipe MixOmics en vue de démontrer le bon fonctionnement de la méthode MINT puisqu'elles ont été utilisée pour la vignette de cette méthode. Stemformatics est une plateforme spécialisée dans le stockage et le partage de données de cellules souches provenant d'études différentes. Cette plateforme fournit également quelques outils de visualisation rapide de ce type de données.

Les caractéristiques du jeu de données initial sont les suivantes et sont illustrées par la Figure 2.10 :

- Les blocs des données omiques représentent quatre études indépendantes de transcriptomique, exécutées dans des conditions différentes. Il y a donc, entre chaque étude, une

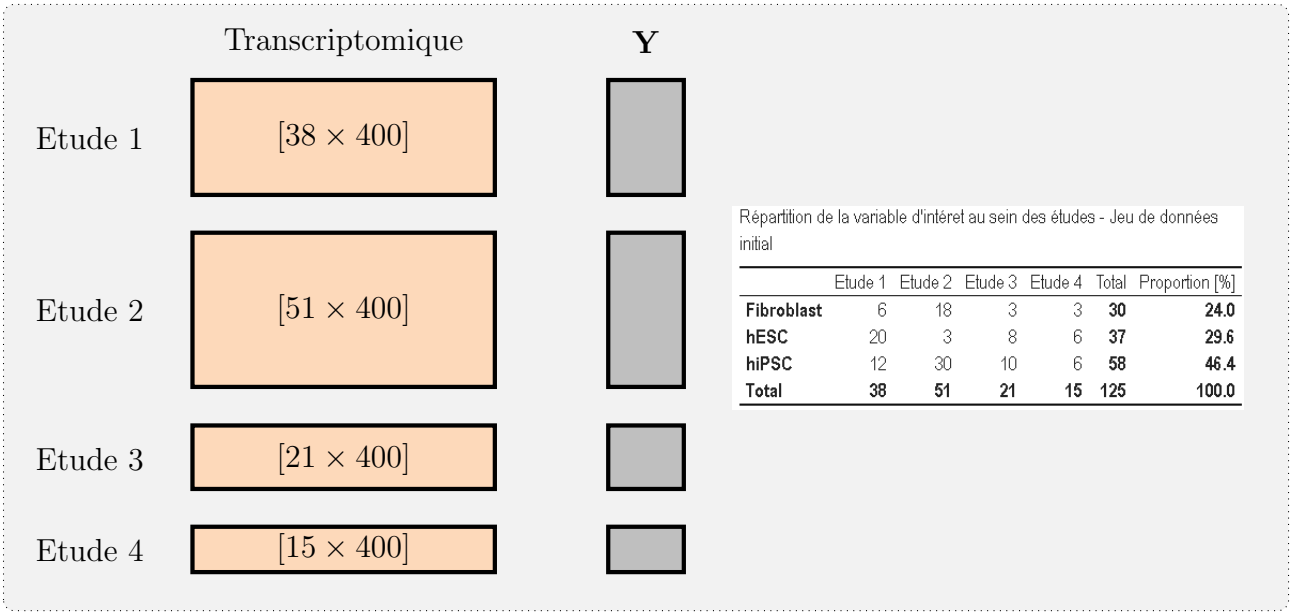
variation indésirable dans les données, due aux différents protocoles ou outils de mesure utilisés, entre autre. Aucune information liée à la plateforme d'origine n'est disponible dans la documentation du package.

- Les quatres blocs ont tous en commun les mêmes 400 variables de transcriptomique mais ils varient par la taille de leurs échantillons : 38, 51, 21 et 15. La taille totale de l'échantillon de l'étude est donc de 125. Chaque étude observe au moins un exemplaire de chaque type de cellule souche.

- La variable d'intérêt comprend 3 types de cellules : FIB (**FIB**roblaste), hiPSC (**C**ellules **S**ouches **P**luripotentes **h**umaines **i**nduites) et hESC (**C**ellules **S**ouches **E**mbryonnaires **h**umaines). Les cellules fibroblastes sont des cellules non-pluripotentes et les deux autres sont des cellules pluripotentes.

  Une cellule *pluripotente* est une cellule qui peut se différencier en plusieurs types de cellules de l'organisme vivant. Il est couramment reconnu que les cellules hiPSC et hESC sont relativement similaires alors qu'elles sont toutes les deux assez différentes des cellules fibroblastes [20]. On peut donc s'attendre à ce que cette différence s'observe également dans les sorties graphiques des méthodes.

Puisque chaque bloc comporte les mêmes variables mais concerne des études différentes, ils sont adaptés à l'intégration verticale. Ce jeu de données sera donc analysé via les méthodes MINT et MOFA+.



Répartition de la variable d'intéret au sein des études - Jeu de données initial

	Etude 1	Etude 2	Etude 3	Etude 4	Total	Proportion [%]
Fibroblast	6	18	3	3	**30**	**24.0**
hESC	20	3	8	6	**37**	**29.6**
hiPSC	12	30	10	6	**58**	**46.4**
Total	**38**	**51**	**21**	**15**	**125**	**100.0**

FIGURE 2.10: Illustration du jeu de données initial fourni par le package *mixOmics* pour l'étude Stemformatics

Le jeu de données initial ne comportant pas de test set, celui-ci a été créé manuellement de manière aléatoire, préalablement aux analyses. La distribution de la variable d'intérêt au

sein du training set et du test set est relativement similaire. Les compositions de ces deux jeux de données sont illustrées par la Figure 2.11.

Répartition de la variable d'intérêt au sein des études - Training set

	Etude 1	Etude 2	Etude 3	Etude 4	Total	Proportion [%]
Fibroblast	5	15	2	2	**24**	**23.8**
hESC	16	2	7	5	**30**	**29.7**
hiPSC	10	24	8	5	**47**	**46.5**
Total	**31**	**41**	**17**	**12**	**101**	**100.0**

Répartition de la variable d'intérêt au sein des études - Test set

	Etude 1	Etude 2	Etude 3	Etude 4	Total	Proportion [%]
Fibroblast	1	3	1	1	**6**	**25.0**
hESC	4	1	1	1	**7**	**29.2**
hiPSC	2	6	2	1	**11**	**45.8**
Total	**7**	**10**	**4**	**3**	**24**	**100.0**

FIGURE 2.11: Illustration de la composition du training set (haut) et du test set (bas) créés pour l'étude de cas Stemformatics

Notons finalement que toutes les variables de ce jeu de données sont identifiées par un nom de 15 caractères. Les neuf premiers caractères sont toujours identiques ("ENSG00000") et désignent l'origine des données (*Ensembl Gene*). Ces caractères ont été enlevés dans certaines sorties graphiques afin d'en faciliter la lecture.

2.6 Outils et notions supplémentaires

Plusieurs notions sont utilisées au sein des méthodes. Cette courte section en explique le principe.

2.6.1 La matrice de confusion

La matrice de confusion est un outil fréquemment utilisé dans des problèmes de classification. Elle permet de compter le nombre d'observations correctement et incorrectement classifiées pour un groupe spécifique de la variable d'intérêt. Elle peut être représentée sous cette forme, si le groupe d'intérêt est la classe A :

	Prédiction : *A*	**Prédiction : *Non A***
Groupe : *A*	VP	FN
Groupe : *Non A*	FP	VN

Où les lettres **V**, **F**, **P** et **N** signifient respectivement "Vrai", "Faux", "Positif" et "Négatif". Sur la base de cette matrice de confusion, de nombreuses mesures peuvent être générées afin d'évaluer la performance d'un modèle.

2.6.2 Le BER

Le BER, ou **B**alanced **E**rror **R**ate est une mesure du taux d'erreur dont l'utilisation est particulièrement recommandée pour l'analyse d'une variable dont les groupes sont mal balancés. Cette mesure se base sur les résultats de la matrice de confusion et peut être utilisée pour l'analyse de la performance d'un modèle. Contrairement au taux d'erreur classique, qui mesure le nombre de mauvaises prédictions par rapport au nombre total d'observations, le BER tient compte du nombre de vraies observations de chaque classe. Ce taux est calculé selon la formule suivante :

$$BER = \frac{\frac{FN}{VP+FN} + \frac{FP}{FP+VN}}{2}$$

Ainsi, par exemple, le BER de la matrice de confusion ci-dessous est de 87.2% alors que le taux d'erreur classique n'est que de 26.9%. Ceci est dû au fait que le BER pénalise fortement la très mauvaise classification du groupe *"Non A"*, qui ne comporte que 4 vraies observations.

	Prédiction : *A*	**Prédiction : *Non A***
Groupe : *A*	56	18
Groupe : *Non A*	3	1

Cet outil sera utilisé dans la phase de paramétrisation des modèles DIABLO et MINT dans les deux études de cas (Chapitres 6.1.2.1 et 6.2.2.1) .

2.6.3 La sensibilité, la spécificité et le score F_1

La sensibilité, la spécificité et le score F_1 sont trois mesures calculées via la matrice de confusion.

- La sensibilité d'un modèle est définie par l'équation suivante :

$$\text{sensibilité} = \frac{VP}{VP + FN}$$

Avec un modèle ayant une sensibilité élevée, la plupart des observations du groupe d'intérêt sont correctement identifiées.

- La spécificité d'un modèle est définie par l'équation suivante :

$$\text{spécificité} = \frac{VN}{VN + FP}$$

Un modèle avec une spécificité élevée identifie correctement la plupart des observations ne faisant pas partie de la classe d'intérêt.

• Le score F_1, ou score F, est une autre mesure de la qualité d'un modèle de classification qui sera utilisé dans ce travail. Il s'agit d'une mesure qui tient compte en même temps de la sensibilité ($\frac{VP}{VP+FN}$) et de la précision ($\frac{VP}{VP+FP}$) d'un modèle. Le score F_1 est défini par la formule suivante :

$$F_1 = \frac{VP}{VP + \frac{FN+FP}{2}}$$

2.6.4 La distance centroïde et de Mahalanobis

Plusieurs mesures de distances sont proposées par les méthodes du package *mixOmics* lors de l'optimisation des paramètres du modèle, dont la distance centroïde et la distance de Mahalanobis. La Figure 2.12 illustre la différence entre ces deux distances. Pour chacune des deux distances, le centroïde de chaque groupe est d'abord calculé (G_g).

La distance centroïde est la distance euclidienne qui sépare un point du centroïde concerné et peut être défini par l'équation suivante :

$$dist_{centroide}(x, G_g) = \sqrt{\sum_{h=1}^{H} \left(x_h - (G_g)_h\right)^2}$$

La distance de Mahalanobis entre un point et un centroïde est définie selon l'équation suivante :

$$dist_{Mahalanobis}(x, G_g) = \sqrt{(x - G_g)' \mathbf{V}^{-1} (x - G_g)}$$

Où $\mathbf{V}$ est la matrice de variance-covariance de $(x - G_g)$. Cela signifie que, dans un espace à plusieurs dimensions, au plus la variance d'une dimension est élevée, au moins la distance calculée dans cette dimension a de l'importance.

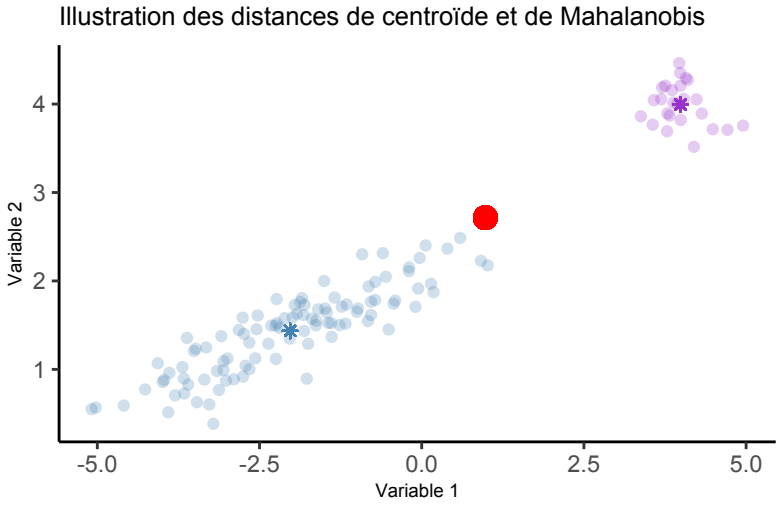


FIGURE 2.12: Bien que la distance euclidienne entre un nouveau point (rouge) et le centroïde bleu ou le centroïde mauve soit identique (distance centroïde), la distance de Mahalanobis avec le centroïde bleu est plus petite.

Chapitre 3

DIABLO

3.1 Principe de fonctionnement

La méthode DIABLO (***D**ata **I**ntegration **A**nalysis for **B**iomarker discovery using **L**atent c**O**mponents*) a été développée par l'équipe de MixOmics en 2016 [21] et fait partie des méthodes qui ont une approche multivariée. Elle s'applique à l'intégration horizontale de données multi-blocs. Les observations sont donc communes à tous les blocs tandis que les modalités varient d'un bloc à l'autre. Il s'agit d'une méthode supervisée dont le but principal est l'identification de biomarqueurs. La découverte de biomarqueurs peut être utile au diagnostic d'une maladie : un gène particulier étant fortement exprimé lorsqu'un patient est atteint d'une maladie par exemple. Une fois qu'un modèle a été entraîné avec la méthode DIABLO, celui-ci peut également être utilisé sur un nouveau jeu de données pour prédire la classe de la variable d'intérêt des nouvelles observations. Il s'agit de l'objectif secondaire de DIABLO.

Cette méthode est basée sur la PLS (***P**artial **L**east **S**quare*) et plus spécifiquement sur la sGCCA (***s**parse **G**eneralized **C**anonical **C**orrelation **A**nalysis*), en étendant son principe. Le but de la sGCCA est d'effectuer une réduction des dimensions de l'espace des données initial en appliquant une combinaison linéaire sur un nombre limité de variables, réparties dans différents blocs $\mathbf{X}^{(.,q)}$. Etant donné son importance dans la compréhension de DIABLO, la sGCCA fera l'objet d'une section dans ce chapitre.

Le principe de fonctionnement de DIABLO consiste à projeter les données dans un sous-espace constitué des vecteurs de scores. D'un point de vue algorithmique, DIABLO calcule des combinaisons linéaires entre les variables des différents blocs d'une part ($\mathbf{X}_h^{(q)}$) et des vecteurs latents sparses (vecteurs de loadings partiels) d'autre part ($\mathbf{a}_h^{(q)}$). Ces combinaisons linéaires permettent de définir des vecteurs de scores ($\mathbf{t}_h^{(q)} = \mathbf{X}_h^{(q)} \cdot \mathbf{a}_h^{(q)}$), qui

peuvent être interprétés comme les projections des observations dans le sous-espace ainsi créé. La recherche des vecteurs de loadings est trouvée à l'aide d'une fonction d'optimisation qui tend à maximiser la somme des covariance des vecteurs de scores des blocs connectés $(\sum_{i,j=1,i\neq j}^{Q} c_{[i,j]} g\big(\, cov(\mathbf{X}_h^{(i)}\mathbf{a}_h^{(i)}, \mathbf{X}_h^{(j)}\mathbf{a}_h^{(j)})\big))$.

Dans cette section, la notation $\mathbf{X}^{(.,q)}$ est simplifiée par $\mathbf{X}^{(q)}$ étant donné que les études (m) ne varient pas.

3.2 Schéma de fonctionnement de DIABLO

La méthode DIABLO peut être représentée selon le schéma illustré en Figure 3.1. L'explication plus précise de son fonctionnement est faite dans les sections qui suivent.

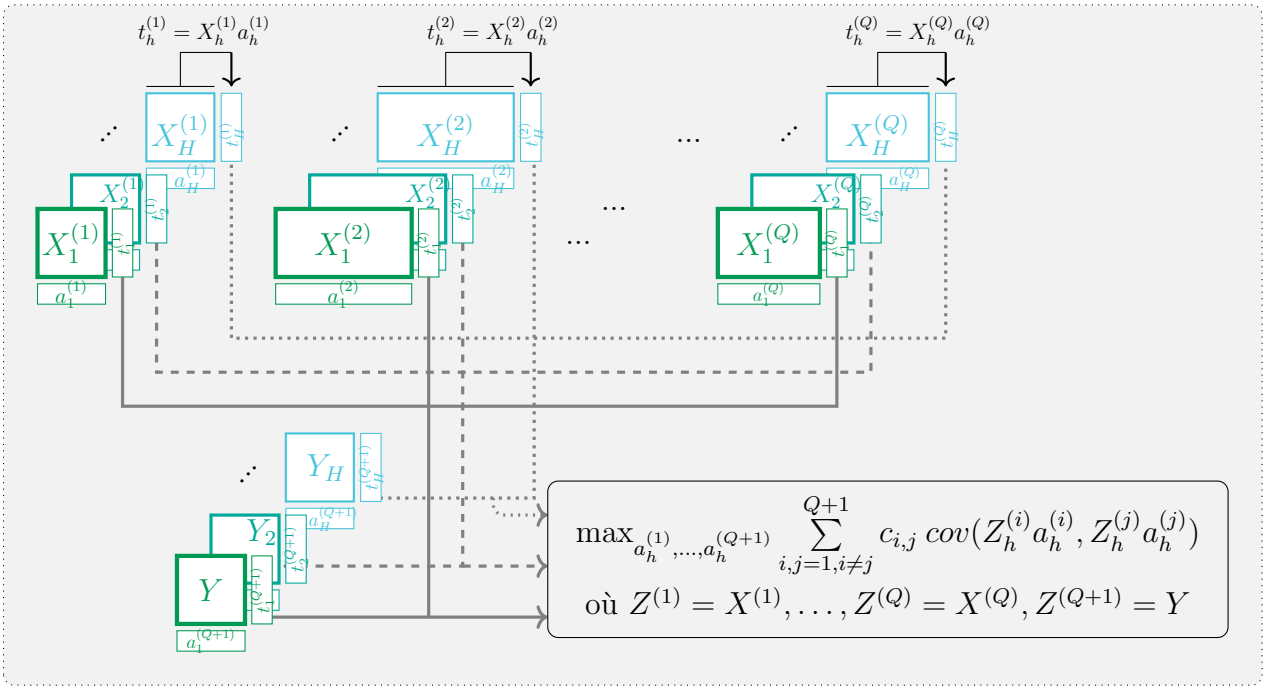


FIGURE 3.1: Fonctionnement schématique de DIABLO. Illustration inspirée et transformée de Dawirs L. [9]

3.3 Aspects mathématiques

3.3.1 sGCCA

La méthode DIABLO étend la sGCCA (**s**parse **G**eneralized **C**anonical **C**orrelation **A**nalysis). La sGCCA est une méthode qui a été proposée pour la première fois en 2013 par Kang et al. [14], puis par Tenenhaus et al. [24]. Il s'agit d'une méthode non-supervisée ; elle n'utilise donc pas la variable d'intérêt $\mathbf{Y}$. Ce chapitre se base sur l'explication de l'article de Tenenhaus. Toutes les variables doivent impérativement être centrées préalablement à l'utilisation

de la sGCCA. La fonction d'optimisation de cette méthode est la suivante :

$$\begin{cases} \max_{\mathbf{a}_h^{(1)},\ldots,\mathbf{a}_h^{(Q)}} \sum_{i,j=1,i\neq j}^{Q} c_{[i,j]} g\big(cov(\mathbf{X}_h^{(i)}\mathbf{a}_h^{(i)}, \mathbf{X}_h^{(j)}\mathbf{a}_h^{(j)})\big) \\ \text{s.t. } \left\|\mathbf{a}_h^{(q)}\right\|_2 = 1 \text{ et } \left\|\mathbf{a}_h^{(q)}\right\|_1 \leq \lambda^{(q)} \qquad \forall\, 1 \leq q \leq Q \end{cases} \tag{3.1}$$

, où :

- $c_{[i,j]}$ est l'élément de la ligne i et de la colonne j de la matrice de design carrée $\mathbf{C}$ ($\mathbf{C} \in \mathbb{R}^{Q\times Q}$). Chaque élément de cette matrice définit si deux blocs sont connectés ($c_{[i,j]} = 1$) ou non ($c_{[i,j]} = 0$).

- $g(x)$ est une fonction qui peut être définie par $g(x) = x$, $g(x) = |x|$ ou $g(x) = x^2$ mais nous estimerons que cette fonction vaut toujours $g(x) = x$ pour l'utilisation de DIABLO.

Notons également que pour h = 1 : $\mathbf{X}_h^{(q)} = \mathbf{X}^{(q)}$,

La sGCCA a donc pour but de maximiser la covariance entre les vecteurs de scores partiels des blocs connectés, étant entendu que $\mathbf{t}_h^{(q)} = \mathbf{X}_h^{(q)}\mathbf{a}_h^{(q)}$. Il n'y a pas de solution analytique à ce problème mais sa résolution peut être obtenue en utilisant les deux algorithmes détaillés ci-dessous.

L'algorithme 1 permet de trouver le vecteur des loadings selon la première composante. Il consiste d'abord à choisir des vecteurs de loadings de manière arbitraire (Etape 2), puis à estimer le premier vecteur de scores partiel du bloc q (Etape 5), à utiliser ce vecteur dans le calcul du poids externe qui y est associé (Etape 7), puis faire de même pour tous les autres blocs (Etape 4), et à répéter toutes ces étapes itérativement jusqu'à la convergence de l'algorithme (Etape 10).

Algorithm 1 : sparse Generalized Canonical Correlation Analysis

Soient Q blocs : $\mathbf{X}^{(1)} \dots \mathbf{X}^{(Q)}$, Q contraintes ℓ_1 (LASSO) : $l^{(1)} \dots l^{(Q)}$ et une matrice de design : $\mathbf{C}$

Choisir arbitrairement Q vecteurs normalisés $\mathbf{a}_0^{(1)} \dots \mathbf{a}_0^{(Q)}$

Répéter pour chaque itérations i $(i = 0, 1, 2...)$

    Pour $q = 1, 2....Q$:

        A : Calculer la composante interne : ▷ Vecteur de scores partiel

$$\mathbf{t}_i^{(q)} \leftarrow \sum_{k=1}^{q-1} c_{[q,k]} g\big(cov(\mathbf{X}^{(q)}\mathbf{a}_i^{(q)}, \mathbf{X}^{(k)}\mathbf{a}_{i+1}^{(k)})\big)\mathbf{X}^{(k)}\mathbf{a}_{i+1}^{(k)} + \sum_{k=q+1}^{Q} c_{[q,k]} g\big(cov(\mathbf{X}^{(q)}\mathbf{a}_i^{(q)}, \mathbf{X}^{(k)}\mathbf{a}_i^{(k)})\big)\mathbf{X}^{(k)}\mathbf{a}_i^{(k)}$$

        B : Calculer le poids externe : ▷ Vecteur des loadings partiel

$$\mathbf{a}_{i+1}^{(q)} \leftarrow \frac{\mathcal{S}\big(\frac{1}{n}(\mathbf{X}^{(q)})'\mathbf{t}_i^{(q)}, \lambda^{(q)}\big)}{\Big\|\mathcal{S}\big(\frac{1}{n}(\mathbf{X}^{(q)})'\mathbf{t}_i^{(q)}, \lambda^{(q)}\big)\Big\|_2}$$

        Où $\mathcal{S}$ correspond à l'opérateur de soft-thresholding (voir ci-dessous) et $\lambda^{(q)} = 0$ si $\left\|\mathbf{a}_{i+1}^{(q)}\right\|_1 \leq l^{(q)}$, ou $\lambda^{(q)}$ est choisi de telle sorte que $\left\|\mathbf{a}_i^{(q)}\right\|_1 = l^{(q)}$ avec $0 \leq l^{(q)} \leq \sqrt{P^{(q)}}$

    Fin

Jusque : $f(\mathbf{a}_{i+1}^{(1)}, \dots, \mathbf{a}_{i+1}^{(Q)}) - f(\mathbf{a}_i^{(1)}, \dots, \mathbf{a}_i^{(Q)}) \leq \epsilon$ ▷ Convergence

, où $f(\mathbf{a}^{(1)}, \dots \mathbf{a}^{(Q)}) = \sum_{q,k=1;q\neq k}^{Q} c_{[q,k]} g\big(cov(\mathbf{X}^{(q)}\mathbf{a}^{(q)}, \mathbf{X}^{(k)}\mathbf{a}^{(k)})\big)$

Résultat : $\mathbf{a}_{i+1}^{(1)}, \dots, \mathbf{a}_{i+1}^{(Q)}$

L'opérateur de soft-thresholding $\mathcal{S}(\mathbf{a}, \lambda)$ permet de définir la valeur d'un élément $\mathbf{a}_{[i]}$ d'un vecteur à zéro s'il est plus petit qu'une valeur λ. Cet opérateur est défini par la relation suivante, appliquée à tous les éléments du vecteur $\mathbf{a}$:

$$\mathcal{S}(\mathbf{a}, \lambda) = \text{sign}(\mathbf{a}) \max(0, |\mathbf{a}| - \lambda)$$

Une fois que les vecteurs de loadings $\mathbf{a}^{(q)}$ ont été trouvés via l'algorithme 1 pour la première composante (h = 1), l'algorithme 2 est utilisé afin de calculer les matrices déflatées selon la composante suivante $\mathbf{X}_{h+1}^{(q)}$. Cette matrice déflatée contient toutes les informations de la matrice précédente $\mathbf{X}_h^{(q)}$, sauf celles qui ont déjà été extraites par le nouveau vecteur de scores $(\mathbf{t}_h^{(q)})$.

Algorithm 2 : Obtention des matrices $\mathbf{X}_{h+1}^{(q)}$ par déflation

1: **Soient** Q blocs : $\mathbf{X}_h^{(1)}, \ldots, \mathbf{X}_h^{(Q)}$ (convention : $\mathbf{X}_1^{(1)} = \mathbf{X}^{(1)}, \ldots, \mathbf{X}_1^{(Q)} = \mathbf{X}^{(Q)}$) et Q vecteurs de scores : $\mathbf{t}_h^{(1)}, \ldots, \mathbf{t}_h^{(Q)}$

2: **Calculer les Q matrices déflatées :**

3: $\mathbf{X}_{h+1}^{(1)} \leftarrow \left(\mathbf{I}_{P^{(1)}} - \frac{\mathbf{t}_h^{(1)}(\mathbf{t}_h^{(1)})'}{(\mathbf{t}_h^{(1)})'\mathbf{t}_h^{(1)}}\right)\mathbf{X}_h^{(1)}, \ldots, \mathbf{X}_{h+1}^{(Q)} \leftarrow \left(\mathbf{I}_{P^{(Q)}} - \frac{\mathbf{t}_h^{(Q)}(\mathbf{t}_h^{(Q)})'}{(\mathbf{t}_h^{(Q)})'\mathbf{t}_h^{(Q)}}\right)\mathbf{X}_h^{(Q)}$

4:

5: **Résultat :** $\mathbf{X}_{h+1}^{(1)}, \ldots, \mathbf{X}_{h+1}^{(Q)}$

Une fois les matrices déflatées obtenues, les vecteurs de scores de la composante suivante ($\mathbf{t}_{h+1}^{(q)}$) seront trouvés en utilisant ces nouvelles matrices déflatées dans l'algorithme 1. On réitère ensuite ces différentes étapes jusqu'à obtenir le nombre de composantes latentes désiré.

On peut remarquer que par construction, les vecteurs de scores partiels, définis par $\mathbf{t}_h^{(q)} = \mathbf{X}_h^{(q)}\mathbf{a}_h^{(q)}$ sont orthogonaux entre eux au sein de chaque bloc. En effet, ils sont calculés sur la base des matrices déflatées, qui sont des sous-espaces orthogonaux au précédentes matrices (Etape 3 de l'algorithme 2).

3.3.2 Adaptation de la sGCCA à DIABLO

DIABLO adapte la sGCCA à une méthode supervisée et à une méthode de classification. Pour ce faire, deux étapes essentielles sont nécessaires :

1. La variable d'intérêt $\mathbf{Y}$ est décomposée en une variable indicatrice avec G colonnes, où G représente le nombre de classes différentes de $\mathbf{Y}$, comme illustré sur la Figure 3.2. Cette nouvelle matrice est ensuite ajoutée comme un nouveau bloc à la matrice $\mathbf{X}$ dans la formule de la sGCCA. Ceci a également un impact direct dans la définition de la matrice de design $\mathbf{C}$, puisque une nouvelle ligne et une nouvelle colonne y sont ajoutées et que $\mathbf{C}$ peut être réécrite selon la Figure 3.3.

Y	A	B	C
A	1	0	0
B	0	1	0
A	1	0	0
C	0	0	1
A	1	0	0
A	1	0	0
B	0	1	0

FIGURE 3.2: Décomposition de la variable d'intérêt $\mathbf{Y}$ en G variables indicatrices

2. Pour forcer la maximisation de la covariance entre les vecteurs de scores de chaque bloc de données $\mathbf{X}^{(q)}$ et celui du bloc d'intérêt $\mathbf{Y}$, les éléments de la dernière colonne et de la dernière ligne de la matrice de design $\mathbf{C}$ (désignant la connection entre les blocs $\mathbf{X}^{(q)}$ et le bloc $\mathbf{Y}$) sont fixés à 1.

   Contrairement à la sGCCA, les autres éléments $c_{[i,j]}$ de la matrice $\mathbf{C}$ ne sont pas limités aux valeurs binaires 0 ou 1 mais peuvent prendre toute valeur comprise entre 0 et 1. Ces valeurs sont encodées par le chercheur, selon la connaissance dont il dispose à priori sur les liens existant entre les différents blocs. Tout comme pour la sGCCA, la valeur 0 représente une absence totale de connection à priori entre les blocs et 1 représente le contraire. Ces valeurs permettent donc d'affiner l'importance accordée à la corrélation souhaitée entre les vecteurs de scores des différents blocs.

   La diagonale de la nouvelle matrice de design est constituée de 0.

$$
\begin{array}{c} \\ \mathbf{X}^{(1)} \\ \mathbf{X}^{(2)} \\ \dots \\ \mathbf{X}^{(Q)} \end{array}
\begin{array}{c}
\begin{array}{cccc} \mathbf{X}^{(1)} & \mathbf{X}^{(2)} & \dots & \mathbf{X}^{(Q)} \end{array} \\
\begin{pmatrix} 0 & c_{[1,2]} & \dots & c_{[1,Q]} \\ c_{[2,1]} & 0 & \dots & c_{[2,Q]} \\ \dots & \dots & \dots & \dots \\ c_{[Q,2]} & c_{[Q,2]} & \dots & 0 \end{pmatrix}
\end{array}
\longrightarrow
\begin{array}{c} \\ \mathbf{X}^{(1)} \\ \mathbf{X}^{(2)} \\ \dots \\ \mathbf{X}^{(Q)} \\ \mathbf{Y} \end{array}
\begin{array}{c}
\begin{array}{ccccc} \mathbf{X}^{(1)} & \mathbf{X}^{(2)} & \dots & \mathbf{X}^{(Q)} & \mathbf{Y} \end{array} \\
\begin{pmatrix} 0 & c_{[1,2]} & \dots & c_{[1,Q]} & 1 \\ c_{[2,1]} & 0 & \dots & c_{[2,Q]} & 1 \\ \dots & \dots & \dots & \dots & 1 \\ c_{[Q,1]} & c_{[Q,2]} & \dots & 0 & 1 \\ 1 & 1 & \dots & 1 & 0 \end{pmatrix}
\end{array}
$$

FIGURE 3.3: Nouvelle définition de la matrice de design $\mathbf{C}$. A gauche : la matrice de design de la sGCCA. A droite : la matrice de design de DIABLO.

3. Le paramètre de pénalisation $\lambda_h^{(q)}$ peut, de manière équivalante, être remplacé par le nombre de variables à conserver dans chaque bloc, en fonction de chaque composante [21].

3.4 Utilisation pratique de DIABLO

3.4.1 Intérêt de la méthode

Identification des biomarqueurs

Les biomarqueurs peuvent être facilement retrouvés grâce aux vecteurs sparse de loadings qui n'attribuent un poids non nul qu'aux variables les plus utiles à la construction des vecteurs de scores, ceux-ci étant sensés être corrélés avec les vecteurs de scores de la variable d'intérêt. Le nombre de biomarqueurs peut être modifié via le paramètre $\lambda_h^{(q)}$.

Prédiction de la classe d'un nouvel échantillon

L'utilisation de DIABLO permet de calculer H vecteurs des loadings $\mathbf{a}_1^{(q)}, \dots, \mathbf{a}_H^{(q)}$ pour chaque bloc de la matrice du training set. Ces vecteurs peuvent ensuite être utilisés sur

les matrices déflatées $\mathbf{X}_1^{(q)}, \dots \mathbf{X}_H^{(q)}$ (avec $\mathbf{X}_1^{(q)} = \mathbf{X}^{(q)}$) pour calculer des vecteurs des scores $\mathbf{t}_1^{(q)}, ..., \mathbf{t}_H^{(q)}$, étant entendu que $\mathbf{t}_h^{(q)} = \mathbf{X}_h^{(q)} \mathbf{a}_h^{(q)}$.

Pour prédire le sous-type d'une nouvelle observation, les données du test set sont d'abord centrée et réduites par bloc. Ensuite, un ensemble de vecteurs de scores $\hat{\mathbf{t}}_{new,1}^{(q)}, \dots \hat{\mathbf{t}}_{new,H}^{(q)}$ est estimé en utilisant les vecteurs des loadings $\mathbf{a}_1^{(q)}, \dots \mathbf{a}_H^{(q)}$, via la relation $\hat{\mathbf{t}}_{new,h}^{(q)} = \mathbf{X}_{new,h}^{(q)} \cdot \mathbf{a}_h^{(q)}$.

La classe prédite par une modalité est celle qui minimise la distance (voir 2.6.4) entre les vecteurs de scores $\hat{\mathbf{t}}_{new}^{(q)}$ et les coordonnées des centroïdes (G_g), calculées sur le training set :

$$arg \min_{1 \leq g \leq G} \left(dist(\hat{\mathbf{t}}_{new}, G_g) \right)$$

En cas de désaccord dans les q prédictions ainsi faites, une méthode de vote peut être utilisée afin d'estimer la classe du phénotype :

• La méthode de vote *maximal* revient à choisir la classe qui a été prédite par le plus de modalités (en cas d'égalité, aucune classe n'est prédite).

• La méthode de vote *pondérée* associe un poids à la prédiction de chaque modalité, en fonction de la corrélation entre les vecteurs de scores de cette modalité $\mathbf{t}_h^{(q)}$ et ceux de la matrice d'intérêt. Cette méthode de vote privilégie donc les datasets qui ont tendance à engendrer des vecteurs de scores qui sont fortement corrélés avec ceux de la variable d'intérêt.

3.4.2 Paramètres de DIABLO

Plusieurs paramètres peuvent être utilisés afin d'affiner le modèle obtenu par la méthode DIABLO :

1. **La matrice de design C**

   Comme on l'a vu, la matrice de design peut prendre différentes valeurs pour ses éléments $c_{[i,j]}$ $(i \neq j, i \neq Q, j \neq Q)$, en fonction de la connaissance à priori dont dispose le chercheur sur les connexions entre les différents blocs analysés. Une matrice avec beaucoup d'éléments $c_{[i,j]}$ prenant la valeur 1 tentera de trouver des combinaisons linéaires qui maximisent la covariance entre toutes les vecteurs de scores de chaque bloc. Au contraire, une matrice $\mathbf{C}$ avec des éléments égaux à 0 partout, sauf sur la dernière ligne et la dernière colonne définira des combinaisons linéaires tentant de maximiser les covariance entre les variables de chaque bloc et la variable d'intérêt uniquement.

2. **Le nombre de composantes** H

   Le nombre de composantes peut également varier. Au plus le nombre de composantes est élevé, au mieux les nouvelles coordonnées représenteront les données initiales. Mais cet avantage se fera au détriment d'un sous-espace de plus grande dimension et donc d'une augmentation du nombre de biomarqueurs, risquant ainsi d'identifier de "faux"

biomarqueurs et d'avoir un modèle overfitté. Lê Cao et al. ont constaté qu'un nombre de composantes égal à $G-1$, où G représente le nombre de groupes différents de $\mathbf{Y}$, était généralement suffisant pour capter l'information nécessaire à caractériser chaque groupe de la variable d'intérêt $\mathbf{Y}$ [15].

3. **Le nombre de variables à garder au sein de chaque bloc de données et selon chaque composante, $\lambda_h^{(q)}$**

   Le paramètre $\lambda_h^{(q)}$ est associé au nombre de variables gardées au sein du bloc q pour construire le vecteur de scores $\mathbf{t}_h^{(q)}$. L'analyste peut introduire cette donnée manuellement s'il a une connaissance à priori du problème mais, de manière plus générale, la fonction d'optimisation `tune.block.splsda()` du package *mixOmics* peut être employée. Il s'agit d'une fonction qui va effectuer une K-fold cross validation combinée à une grille de recherche introduite par l'utilisateur. Cette recherche d'optimisation du nombre de variables va se faire en tentant de minimiser un critère, qui peut soit être le taux global de mauvaise classification, soit le BER (**B**alanced **E**rror **R**ate). Cette notion est expliquée en Section 2.6.2. Ce dernier critère a l'avantage de tenir compte du nombre d'observations dans chaque classe de la variable d'intérêt et est recommandé lorsque celles-ci sont mal balancées.

3.4.3 Sorties graphiques

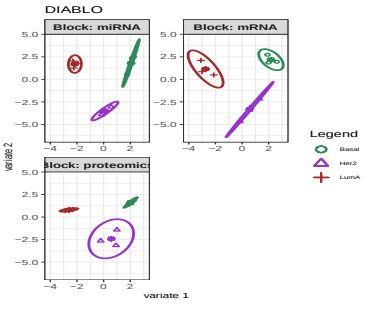
De nombreuses sorties graphiques sont proposées par le package, dont les plus pertinentes sont expliquées ci-dessous. Le nom des graphiques correspond à la fonction utilisée dans le package. Une illustration est fournie afin que le lecteur puisse avoir un repère visuel de la sortie décrite. Mais le but de ces illustrations n'est pas d'en distinguer toutes les subtilités : des graphiques plus grands et plus lisibles seront utilisés plus tard dans ce travail, lors de l'étude de cas.

Graphiques d'échantillon

Avec ce type de graphique, chaque point représente une observation. La couleur des points indique le groupe de la variable d'intérêt auquel l'observation appartient.

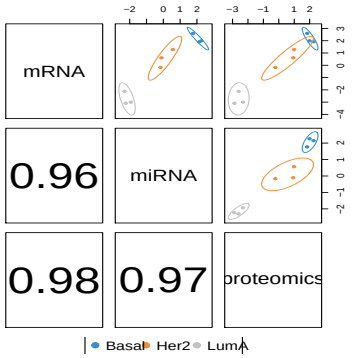
1. `plotIndiv()`

   Ce graphique projette les données dans le sous-espace défini par les vecteurs des scores partiels ($\mathbf{t}_h^{(q)}$) de chaque bloc, selon les deux composantes h sélectionnées. En général, les deux premières composantes sont observées car ce sont celles qui sont le plus corrélées avec les vecteurs de scores de la variable d'intérêt. Pour visualiser les groupes, les centroïdes et des ellipses peuvent également être ajoutées par l'utilisateur. Les ellipses représentent un intervalle de confiance de 95% dans lequel les observations sont projetées.



2. `plotDiablo()`

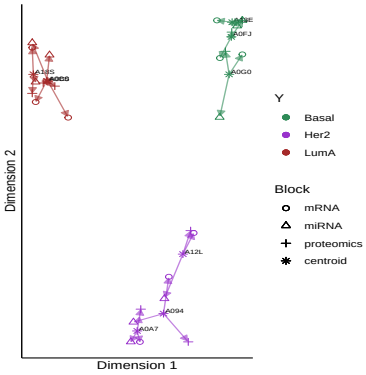
Dans sa partie triangulaire inférieure, ce graphique renseigne le coefficient de corrélation de Pearson des vecteurs de scores partiels ($\mathbf{t}_h^{(q)}$) de deux blocs, en fonction d'une seule composante choisie.



Dans la partie triangulaire supérieure, le graphique permet de visualiser cette corrélation dans le sous-espace définit par les vecteurs de scores partiels des deux blocs, selon la composante latente observée. Ceci permet de constater le pouvoir de classification des deux vecteurs de scores partiels, selon la composante choisie.

3. `plotArrow()`

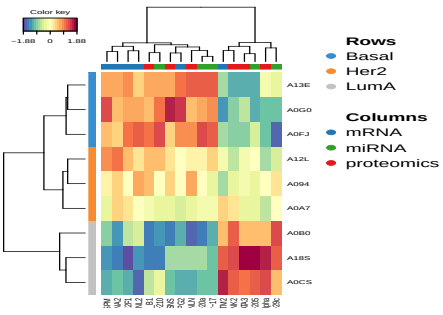
Ce graphique peut être vu comme l'empilement des trois graphiques obtenus via `plotIndiv()`. Il s'agit donc d'une représentation des observations, projetées dans le sous-espace défini par les vecteurs de scores partiels ($\mathbf{t}_h^{(q)}$) pour deux dimensions choisies. Toutefois, afin de permettre cet empilement des graphiques, ces vecteurs ont subi une transformation : le minimum et le maximum de chaque vecteur est identique entre les blocs et par dimension.



Chaque observation est représentée par autant de flèches qu'il y a des blocs (Q). L'origine des flèches représente le centroïde de chaque observation et chaque pointe de flèche représente sa projection dans les différentes modalités analysées. Au plus les flèches sont proches pour un même individu, au plus cela signifie que les différentes modalités de l'observation donnent des résultats similaires dans les vecteurs de scores.

4. `cimDiablo()`

Le dernier graphique d'échantillon proposé par la méthode DIABLO du package *mixOmics* est une heatmap, où les lignes représentent les observations et où les colonnes représentent les biomarqueurs détectés dans une ou plusieurs composantes sélectionnées. Ce graphique permet donc d'identifier des patterns et des comportement particuliers des observations ou des biomarqueurs estimés par le modèle.



Deux dendrogrammes sont également générés une fois que les biomarqueurs ont été sélectionnés en fonction des paramètres introduits dans le modèle (H, $\mathbf{C}$ et $\lambda_h^{(q)}$). Le dendrogramme représenté à gauche du graphique permet de visualiser la similarité des observations par rapport aux biomarqueurs sélectionnés. Il est à remarquer qu'aucune option ne permet de choisir la mesure des distances ou la méthode à utiliser pour le

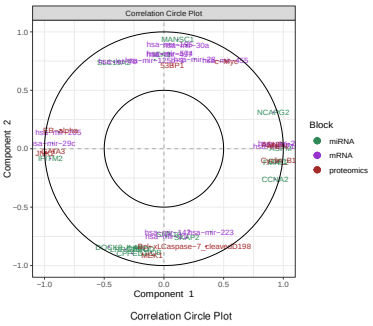
calcul du dendrogramme.

Graphiques de variables

Ce type de graphique a pour but de montrer chaque variable, projeté dans un sous-espace. Il permet de s'intéresser à la similarité entre les variables et à leur influence dans la construction des vecteurs de scores. La couleur utilisée dans les graphes représente la modalité à laquelle appartiennent chaque variable.

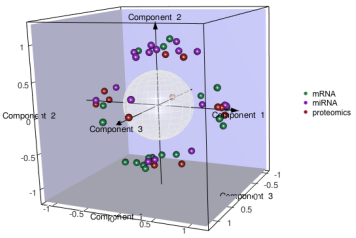
5. `plotVar()`

Il s'agit d'un graphique représentant le cercle des corrélations des variables, similaire au cercle des corrélations d'une PCA. Chaque point, représentant une variable, permet donc de constater si celle-ci est bien représentée par les deux vecteurs de scores partiels ($\mathbf{t}_h^{(q)}$) associés à sa modalité, selon les composantes choisies. Au plus le point se trouve proche de l'extrémité du cercle, au plus la variable est correctement représentée par les deux vecteurs de scores partiels.



Correlation Circle Plot

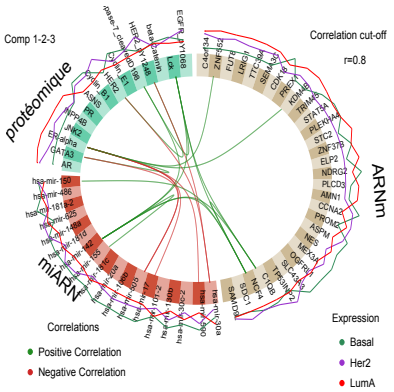
Un seuil peut être spécifié afin d'alléger la lisibilité des données. De plus, il est possible de demander l'affichage du nom des variables afin de les identifier, mais cela devient rapidement illisible dès que plus de 20 variables sont représentées sur le graphique.



Une version de ce graphique en 3 dimensions est aussi disponible. Cette dernière visualisation peut être intéressante lorsqu'il faut au moins 3 composantes latentes pour représenter correctement les variables. L'observation et l'analyse de ce graphique en 3 dimensions sont facilitées par son interactivité de la plateforme informatique.

6. `circosPlot()`

Ce graphique a plusieurs fonctionnalités. Il sert notamment à l'identification des biomarqueurs, mais donne également une information relative à la corrélation entre eux ainsi qu'au niveau d'expression des groupes de la variable d'intérêt, selon chaque biomarqueur sélectionné. Il est possible de générer un circos plot en fonction d'une, de plusieurs ou de toutes les composantes du modèle.



Trois types d'information sont visibles sur ce graphique, de l'intérieur vers l'extérieur :

→ La détection des biomarqueurs corrélés

Le centre du cercle comprend des courbes de couleur. Chaque courbe relie deux biomarqueurs dont le coefficient de corrélation de Pearson est supérieur à un seuil minimal qui peut être défini par l'utilisateur. Si la valeur précise du coefficient de corrélation n'est pas visible sur ce graphique, il y a cependant un code couleur qui permet d'indiquer s'il s'agit d'une corrélation positive (ici en vert) ou négative (rouge). Eventuellement, la largeur de ces courbes peut varier en fonction de la valeur de la corrélation entre les deux biomarqueurs.

→ La détection des biomarqueurs

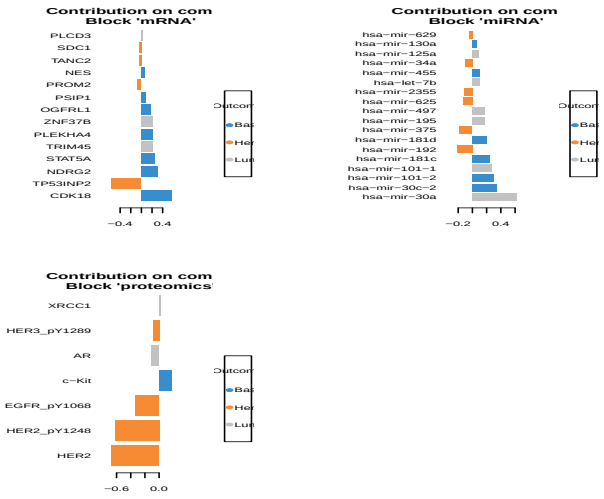
Les variables ayant un coefficient non nul dans les vecteurs des loadings ($\mathbf{a}_h^{(q)}$) sont identifiées comme les plus influentes et constituent les biomarqueurs estimés par le modèle. Les noms de ces variables sont affichés en cercle et sont groupés en fonction de la modalité à laquelle elles appartiennent.

→ Le niveau d'expression de la variable d'intérêt selon chaque biomarqueur

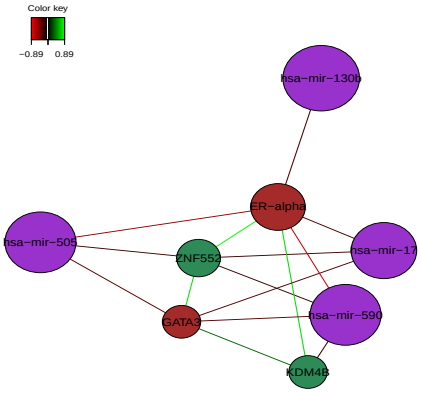
Finalement, en bordure du cercle, on peut voir G lignes de couleurs différentes. Chacune de ces lignes représente le niveau d'expression moyen d'une classe de la variable d'intérêt en fonction des biomarqueurs. Au plus la ligne s'éloigne du centre du cercle, au plus la valeur moyenne du biomarqueur sera élevée pour les observations appartenant à la classe concernée (par rapport aux autres classes). Ce graphique permet donc d'identifier les biomarqueurs qui ont un comportement fort différent selon les groupes de la variable d'intérêt. On peut d'ailleurs constater qu'en général, les biomarqueurs corrélés positivement ont un profil d'expression de la variable $\mathbf{Y}$ qui est similaire, et inversément.

7. `plotLoadings()`

Ce graphique illustre les coefficients non nuls associés aux biomarqueurs dans le vecteur des loadings partiel ($\mathbf{a}_h^{(q)}$) pour une certaine composante. Ceci permet donc d'analyser le rôle (et l'importance) de chaque biomarqueur dans la construction de chaque vecteur de scores partiel. Le poids des variables peut être négatif (barre à gauche) ou positif (à droite). Finalement, la couleur de chaque barre indique la classe de la variable d'intérêt dans laquelle la valeur moyenne du biomarqueur concerné est la plus élevée.



8. `network()`



Ce graphique permet une autre visualisation de la corrélation entre les différents biomarqueurs, mais limite leur nombre grâce à un seuil de corrélation minimal entre eux. Le but du seuil est donc légèrement différent de celui utilisé via la fonction `circosPlot()` puisque ce dernier était relatif aux courbes reliant les biomarqueurs, mais ne limitait pas leur nombre. Chaque noeud représente un biomarqueur et sa couleur représente la modalité à laquelle il appartient. La taille du noeud n'a pas de signification précise. Les liens représentent les corrélations entre les biomarqueurs dont la valeur absolue est supérieure au seuil minimal défini par l'utilisateur. La couleur des liens représente la valeur de cette corrélation.

Ce graphique ne permet toutefois pas toujours une visualisation évidente dès que le nombre de biomarqueurs devient trop important, à cause du chevauchement des noeuds. Il est par ailleurs dommage que la taille du noeud ne puisse pas être réduite et qu'elle dépende surtout de la longueur du nom du biomarqueur. Une option de transparence des noeuds est disponible mais ne fonctionne pas correctement. Dans tous les cas, même avec cette option, ce type de graphique deviendrait probablement illisible si le nombre de biomarqueurs corrélés était très important (> 50).

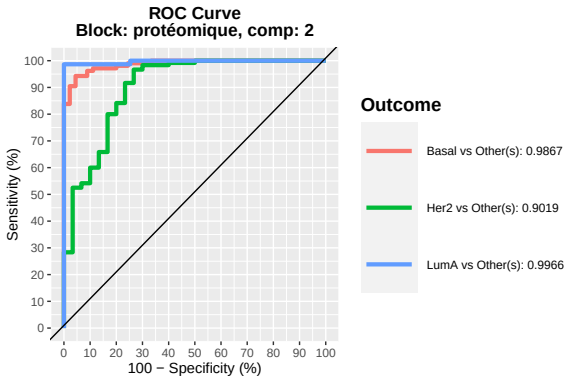
Une option permet de faire varier le seuil de manière interactive, ce qui peut être utile, mais nous ne sommes jamais parvenus à la faire fonctionner. Finalement, nous avons également rencontré des problèmes d'affichage avec ce graphique. La possibilité de l'apparition de ces problèmes est signalée dans la vignette de la méthode DIABLO et peut être résolue en agrandissant la fenêtre d'affichage du graphique au maximum dans l'environnement du programme utilisé.

- Autres graphiques

Le package propose un autre graphique utile lors de l'évaluation du modèle.

9. `auroc()`

Cette fonction permet de générer un graphique avec une courbe ROC pour chaque groupe de la variable **Y**. Au plus la courbe se rapproche du coin gauche du graphique, au plus la sensibilité et la spécificité du modèle sont importants. L'aire sous la courbe (**A**rea **U**nder the **C**urve) est également renseignée en légende du graphique. Cette valeur est comprise entre 0 et 1. Un modèle ayant une AUC inférieure à 0.5 n'est pas utile car il prédit moins bien les données que si la prédiction était faite aléatoirement.



Chapitre 4

MINT

4.1 Principe de fonctionnement

La méthode MINT (**M**ultivariate **INT**egration method) a été proposée par l'équipe de MixOmics et présentée dans un article pour la première fois en 2017 [20]. Il s'agit d'une méthode multivariée adaptée à l'intégration verticale des données omiques multi-blocs. Chaque bloc représente une étude différente et le nombre d'observations dans chaque étude peut donc varier. Par contre, les variables (et la modalité) sont toutes identiques d'un bloc à l'autre. Il s'agit d'une méthode supervisée dont le but principal est la détection de biomarqueurs. Une fois qu'un modèle a été entraîné avec la méthode MINT, celui-ci peut être utilisé sur un nouveau jeu de données pour prédire la classe de la variable d'intérêt des nouvelles observations. Il s'agit de l'objectif secondaire de MINT.

MINT se base sur la PLS-DA (**P**artial **L**east **S**quares-**D**iscriminant **A**nalysis) et plus particulièrement sur la mgPLS (**m**ulti-**g**roup PLS) en étendant son principe en incluant une pénalisation ℓ_1 sur le vecteur des loadings. Cette pénalisation aide à l'identification des biomarqueurs et également à l'interprétation de l'analyse. Etant donné son importance dans la méthode MINT, la mgPLS sera expliquée dans la section 4.3.1.

Le principe de fonctionnement de MINT est schématisé à la section 4.2 et consiste à projetter les données dans un sous-espace constitué des vecteurs de scores. D'un point de vue algorithmique, MINT calcule des combinaisons linéaires entre les variables des différents blocs d'un part ($\mathbf{X}_h^{(m)}$) et des vecteurs latents sparses (vecteurs de loadings globaux) d'autre part ($\mathbf{a}_h$). Ces combinaisons linéaires permettent de définir des vecteurs de scores ($\mathbf{t}_h^{(m)} = \mathbf{X}_h^{(m)} \cdot \mathbf{a}_h$) qui peuvent être interprétés comme les projections des observations dans le sous-espace ainsi créé. L'optimalisation des vecteurs de loadings est faite en maximisant la somme des covariances des vecteurs de scores de $\mathbf{X}$ et de $\mathbf{Y}$. Chacune des covariances est pondérée par un

poids lié au nombre d'observations de chaque bloc concerné : $\sum_{m=1}^{M} N^{(m)} cov\big(\mathbf{X}_h^{(m)}\mathbf{a}_h, \mathbf{Y}_h^{(m)}\mathbf{b}_h\big) + \lambda_h||\mathbf{a}_h||$. Ainsi, au plus un bloc a un nombre important d'observations, au plus la covariance calculée a de l'importance.

Dans ce chapitre, la notation $\mathbf{X}^{(m,.)}$ est remplacée par $\mathbf{X}^{(m)}$ étant donné que la modalité (q) ne varie pas.

4.2 Schéma de fonctionnement de MINT

Le principe de fonctionnement de la méthode MINT est illustré par la Figure 4.1.

On peut noter que, contrairement à la méthode DIABLO, il n'y a qu'un seul vecteur de loadings par composante $(\mathbf{a}_h)$, commun à tous les blocs $\mathbf{X}^{(m)}$. Il s'agit donc d'un vecteur de loadings global. Il en va de même pour le vecteur de loadings de $\mathbf{Y}$ $(\mathbf{b}_h)$.

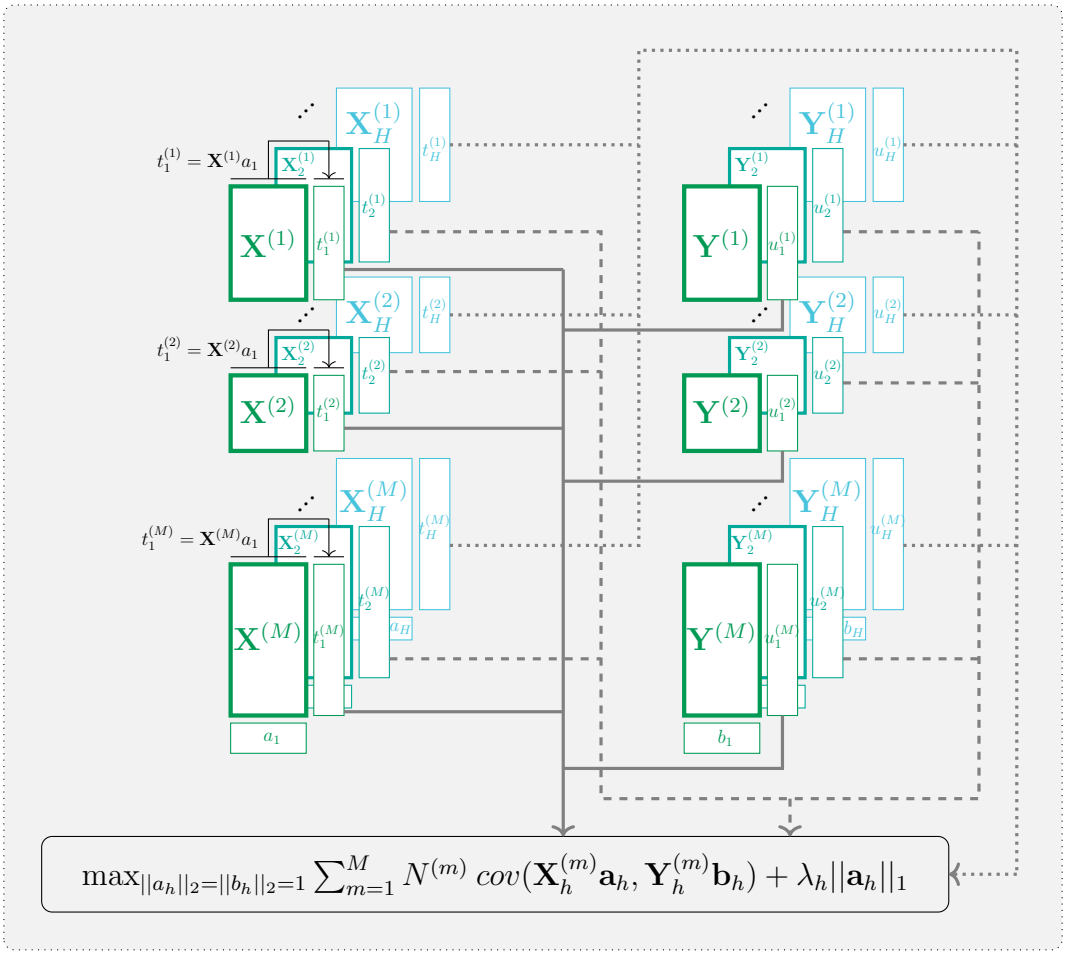


FIGURE 4.1: Fonctionnement schématique de MINT

4.3 Aspects mathématiques

4.3.1 mgPLS

Lorsque l'on dispose d'un échantillon trop petit, il peut être utile de s'intéresser à plusieurs études ayant observé le même phénomène sur différents individus afin d'augmenter le nombre d'observations. Dans ce cas, une solution naïve consisterait à concaténer toutes les études ensemble. Cependant, ce genre de pratique risquerait fortement d'introduire une variabilité non désirée dans les jeux de données. Cette variabilité pourrait s'expliquer par des appareils de mesure ou des protocoles différents au sein de chaque étude, etc. Il serait également possible d'effectuer une analyse par étude, mais la multiplication de tests peut engendrer d'autres problèmes. La mgPLS permet de tenir compte de cette structure en blocs et d'effectuer une analyse PLS sur l'ensemble de ceux-ci en une seule fois.

La méthode mgPLS, qui se base sur la PLS, a été proposée pour la première fois en 2013 par Abdi et al. [1]. Il s'agit d'une méthode de régression qui ne s'applique pas uniquement aux données omiques mais à toutes les données ayant une structure en blocs verticaux. Chaque bloc de données est défini comme $\mathbf{X}^{(m,.)} = \mathbf{X}^{(m)}$ et a une variable d'intérêt $\mathbf{Y}^{(m,.)} = \mathbf{Y}^{(m)}$ qui lui est associé. La mgPLS part du principe que le poids associé à chaque bloc $\mathbf{X}^{(m)}$ dépend du nombre d'observations qui le composent. Par ailleurs, chaque variable doit être centrée par bloc, préalablement à l'application de la méthode.

Le critère de maximisation de la méthode permettant de retrouver la première composante latente est le suivant :

$$\begin{aligned} &\max_{||\mathbf{a}_1||_2=||\mathbf{b}_1||_2=1} \sum_{m=1}^{M} N^{(m)}\, cov\big(\mathbf{t}_1^{(m)}, \mathbf{u}_1^{(m)}\big) \\ &= \max_{||\mathbf{a}_1||_2=||\mathbf{b}_1||_2=1} \sum_{m=1}^{M} N^{(m)}\, cov\big(\mathbf{X}^{(m)}\mathbf{a}_1, \mathbf{Y}^{(m)}\mathbf{b}_1\big) \\ &= \max_{||a_1||_2=||b_1||_2=1} \mathbf{a}_1' \left[\sum_{m=1}^{M} (\mathbf{X}^{(m)})'\mathbf{Y}^{(m)}\right] \mathbf{b}_1 \end{aligned} \tag{4.1}$$

, où $\mathbf{u}_1^{(m)}$ est le vecteur de scores partiel de la variable d'intérêt associée au bloc m dans la première composante et $\mathbf{b}_1$ est le vecteur de loadings global permettant de construire ce vecteur $\mathbf{u}_1^{(m)}$.

Une contrainte de cette méthode est donc que les vecteurs des loadings $\mathbf{a}_1$ et $\mathbf{b}_1$ soient communs à tous les blocs. Abdi et al. ont proposé une solution à ce problème. Pour un vecteur $\mathbf{a}_1$ fixe, le maximum est atteint lorsque :

$$\mathbf{b}_1 = \frac{\sum_{m=1}^{M} (\mathbf{Y}^{(m)})'\mathbf{X}^{(m)}\mathbf{a}_1}{||\sum_{m=1}^{M} (\mathbf{Y}^{(m)})'\mathbf{X}^{(m)}\mathbf{a}_1||_2} \tag{4.2}$$

En remplaçant la valeur de $\mathbf{b}_1$ dans l'équation (4.1) par celle trouvée dans l'équation (4.2), le critère à maximiser peut être réécrit de la manière suivante :

$$\max_{||\mathbf{a}_1||_2=||\mathbf{b}_1||_2=1} \mathbf{a}_1' \left(\frac{\sum_{m=1}^{M} (\mathbf{X}^{(m)})'\mathbf{Y}^{(m)} \sum_{m=1}^{M} (\mathbf{Y}^{(m)})'\mathbf{X}^{(m)}}{|| \sum_{m=1}^{M} (\mathbf{Y}^{(m)})'\mathbf{X}^{(m)}\mathbf{a}_1||_2} \right) \mathbf{a}_1 \tag{4.3}$$

Ce qui revient à maximiser :

$$\max_{||\mathbf{a}_1||_2=||\mathbf{b}_1||_2=1} \sqrt{\mathbf{a}_1' \left(\sum_{m=1}^{M} (\mathbf{X}^{(m)})'\mathbf{Y}^{(m)} \sum_{m=1}^{M} (\mathbf{Y}^{(m)})'\mathbf{X}^{(m)} \right) \mathbf{a}_1} \tag{4.4}$$

Une solution peut finalement être trouvée en deux étapes :

1 : Assigner au vecteur de loadings $\mathbf{a}_1$ le vecteur propre associé à la plus grande valeur propre de $\sum_{m=1}^{M} (\mathbf{X}^{(m)})'\mathbf{Y}^{(m)} \sum_{m=1}^{M} (\mathbf{Y}^{(m)})'\mathbf{X}^{(m)}$.

2 : Trouver le vecteur de loadings $\mathbf{b}_1$ de $\mathbf{Y}$ en utilisant cette valeur de $\mathbf{a}_1$ dans l'équation (4.2).

Connaissant $\mathbf{a}_1$ et $\mathbf{b}_1$, il est aisé de retrouver $\mathbf{t}_1^{(m)}$ et $\mathbf{u}_1^{(m)}$ via les relations suivantes :

$$\begin{cases} \mathbf{t}_1^{(m)} = \mathbf{X}^{(m)}\mathbf{a}_1 \\ \mathbf{u}_1^{(m)} = \mathbf{Y}^{(m)}\mathbf{b}_1 \end{cases} \tag{4.5}$$

Une autre solution itérative peut être trouvée en utilisant l'algorithme NIPALS. Cet algorithme a l'avantage de ne pas nécessiter de calcul de vecteurs propres et de valeurs propres. L'algorithme NIPALS est donné en Annexe A mais n'est pas retranscrit ici car il est très similaire à l'algorithme MINT, qui sera développé dans le chapitre suivant.

4.3.2 Adaptation de la mgPLS à MINT

MINT se base sur la mgPLS et la PLS-DA. L'adaptation de la PLS (méthode de régression), à la PLS-DA (méthode de classification) se fait en transformant la variable $\mathbf{Y}$ en autant de variables binaires qu'il y a de classes différentes dans $\mathbf{Y}$ (voir Figure 3.2).

MINT étend la mgPLS en ajoutant un paramètre de pénalisation non-négatif λ_h au critère de maximisation. Ce paramètre définit le niveau de shrinkage et a donc un impact direct sur le nombre de variables avec un coefficient non nul dans le vecteur des loadings $\mathbf{a}_h$. Contrairement à la méthode DIABLO, ce vecteur est un vecteur global, identique pour tous les blocs.

Le critère de maximisation de la méthode MINT est donc le suivant :

$$\max_{||a_h||_2=||b_h||_2=1} \sum_{m=1}^{M} N^{(m)} \, cov\left(X_h^{(m)} a_h, Y_h^{(m)} b_h\right) + \lambda_h ||a_h||_1 \tag{4.6}$$

La méthode a donc pour objectif de maximiser la covariance entre les vecteurs de scores des blocs et ceux de la variable d'intérêt de la modalité correspondante, tout en restreignant le nombre de biomarqueurs sélectionnés via le paramètre λ_h. Il est à noter que le paramètre $N^{(m)}$ pondère la covariance qui y est associée selon le nombre d'observations de chaque bloc.

L'algorithme 3 permet de trouver une solution au critère de maximisation de la méthode MINT.

Algorithm 3 MINT

1: Notons $\forall 1 \leq m \leq M, \mathbf{X}_1^{(m)} = \mathbf{X}^{(m)}, \mathbf{Y}_1^{(m)} = \mathbf{Y}^{(m)}, \mathbf{X}_h = \left[\left(\mathbf{X}_h^{(1)}\right)', ..., \left(\mathbf{X}_h^{(M)}\right)'\right]'$ = et $\mathbf{Y}_h = \left[\left(\mathbf{Y}_h^{(1)}\right)', ..., \left(\mathbf{Y}_h^{(M)}\right)'\right]'$, où les $\mathbf{X}^{(m)}$ et $\mathbf{Y}^{(m)}$ sont centrés et réduits par bloc.
2: Pour chaque $h \leq H$, on choisit λ_h et une valeur d'initialisation pour $\mathbf{a}_h$ avec $||\mathbf{a}_h||_2 = 1$ et on applique les étapes 4 à 13,
3: **Calcul de $\mathbf{a}_h$ et $\mathbf{b}_h$:**
4: **Répéter :**
5: $\mathbf{t}_h^{(m)} \leftarrow \mathbf{X}_h^{(m)}\mathbf{a}_h$ ▷ Vecteur des scores ($\mathbf{X}$) partiel
6: $\mathbf{t}_h \leftarrow \mathbf{X}_h\mathbf{a}_h$ ▷ Vecteur des scores ($\mathbf{X}$) global
7: $\mathbf{b}_h^{(m)} \leftarrow (\mathbf{Y}_h^{(m)})'\mathbf{t}_h^{(m)}$ ▷ Vecteur des loadings ($\mathbf{Y}$) partiel
8: $\mathbf{b}_h \leftarrow \frac{\sum_{m=1}^{M} \mathbf{b}_h^{(m)}}{||\sum_{m=1}^{M} \mathbf{b}_h^{(m)}||_2}$ ▷ Vecteur des loadings ($\mathbf{Y}$) global
9: $\mathbf{u}_h^{(m)} \leftarrow \mathbf{Y}_h^{(m)}\mathbf{b}_h$ ▷ Vecteur des scores ($\mathbf{Y}$) partiel
10: $\mathbf{a}_h^{(m)} \leftarrow (\mathbf{X}_h^{(m)})'\mathbf{u}_h^{(m)}$ ▷ Vecteur des loadings ($\mathbf{X}$) partiels
11: $\mathbf{a}_h \leftarrow \frac{\sum_{m=1}^{M} \mathbf{a}_h^{(m)}}{||\sum_{m=1}^{M} \mathbf{a}_h^{(m)}||_2}$ ▷ Vecteur des loadings ($\mathbf{X}$) global
12: $\mathbf{a}_h \leftarrow S(\mathbf{a}_h, \lambda_h)$ ▷ Opérateur de soft-thresholding
13: **Jusqu'à** convergence de $\mathbf{a}_h$ et de $\mathbf{b}_h$
14: **Etapes de déflation :**
15: $\mathbf{P} \leftarrow \mathbf{I}_N - t_h(\mathbf{t}_h'\mathbf{t}_h)^{-1}\mathbf{t}_h'$ ▷ , où $\mathbf{I}_N$ est la matrice identité de dimention $N \times N$
16: $\mathbf{X}_{h+1} \leftarrow \mathbf{P}\mathbf{X}_h$ ▷ Déflation de la matrice $\mathbf{X}$
17: $\mathbf{Y}_{h+1} \leftarrow \mathbf{P}\mathbf{Y}_h$ ▷ Déflation de la matrice $\mathbf{Y}$

Le principe de fonctionnement de l'algorithme 3 est représenté par la Figure 4.2 et provient de l'annexe *Supplementary Materials* de [20].

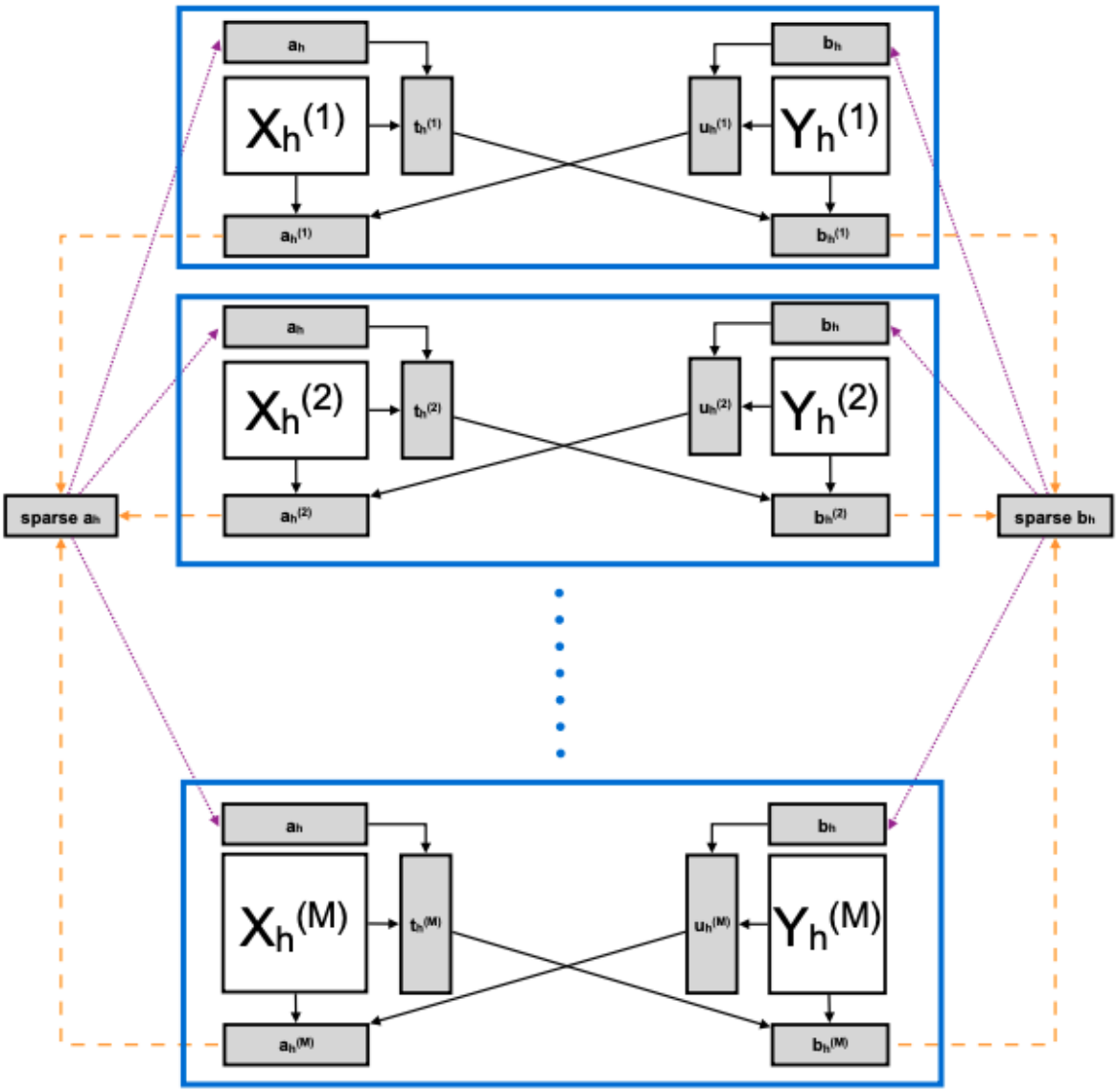


FIGURE 4.2: Principe de fonctionnement de l'algorithme de MINT. Les flèches noires représentent les multiplications matricielles ; les lignes orange représentent les additions et les lignes rose ne représentent pas d'opération.

4.4 Utilisation pratique de MINT

4.4.1 Intérêt de la méthode

Identification des biomarqueurs

Les biomarqueurs sont trouvés grâce au vecteur des loadings $\mathbf{a}_h$ car il s'agit des variables dont le coefficient dans ce vecteur est non nul. Au plus le nombre de composantes H augmente, au plus le nombre total de biomarqueurs augmente également. Il est possible que des biomarqueurs définis dans une composante le soient également dans une autre composante. Les biomarqueurs les plus discriminants sont généralement ceux qui ont un coefficient non nul dans les premiers vecteurs de loadings.

Prédiction de la classe d'un nouvel échantillon

La première chose à faire avec un nouvel échantillon est de centrer et réduire le jeu de données par bloc, puisqu'il s'agit d'une des contraintes du modèle. Ensuite, une fois que le modèle MINT a été obtenu avec les données d'origine, le vecteur des scores d'un nouvel échantillon ($\mathbf{t}^{(m)}_{new,h}$) peut facilement être trouvé en utilisant le vecteur des loadings préalablement trouvé : $\mathbf{t}^{(m)}_{new,h} = \mathbf{X}^{(m)}_{new,h}\mathbf{a}_h$. Des mesures de distances peuvent être calculées entre ces vecteurs de scores et les centroïdes de chaque groupe d'intérêt, et une prédiction peut être faite pour les G groupes de la variable d'intérêt. Le groupe finalement prédit pour une composante h correspond à la colonne de $\mathbf{Y}_{new,h}$ qui a la plus grande valeur positive.

4.4.2 Paramètres de MINT

1. **Le nombre de composantes latentes** H

   Comme pour DIABLO, le nombre de composantes latentes peut être défini par le chercheur. Un nombre trop faible de composantes pourrait ne pas capter suffisamment d'information mais un nombre trop important pourrait engendrer la détection de faux biomarqueurs et un sous-espace de trop grande dimension. Il est conseillé de fixer $H = G - 1$, où G est le nombre de classes différentes de la variable d'intérêt $\mathbf{Y}$ [21].

2. **Le paramètre de pénalisation** λ_h

   Comme mentionné précédemment, ce paramètre définit le degré de shrinkage du modèle et est donc directement associé au nombre de variables qui seront utilisées pour calculer les vecteurs des scores. Au plus ce paramètre est élevé, au plus il y aura de variables avec un coefficient non nul qui leur sera associé dans le vecteur de loadings. Il s'agit donc de trouver un compromis sur ce paramètre car le principal but de cette méthode consiste bien à retrouver les variables qui sont des biomarqueurs réels de la variable d'intérêt $\mathbf{Y}$.

La recherche de l'optimisation de ces deux paramètres peut être effectué via une cross-validation. Le package *mixOmics* propose une fonction `tune()`, qui accepte une option `test.keepX`. Cette fonction exécute une LOGOCV (**L**eave **O**ne **G**roup **O**ut **C**ross **V**alidation) qui comme son nom l'indique, consiste en une cross-validation où un bloc complet (une étude) est retiré du modèle à la fois. L'option `test.keepX` permet d'encoder un vecteur dont les éléments représentent les différentes valeurs de λ_h à tester lors de cette cross-validation. La fonction renvoie le taux d'erreur calculé sur le jeu de données, selon les différentes valeurs pour λ_h, celui-ci devant être le plus faible possible. Ceci sera illustré plus tard par la Figure 6.26

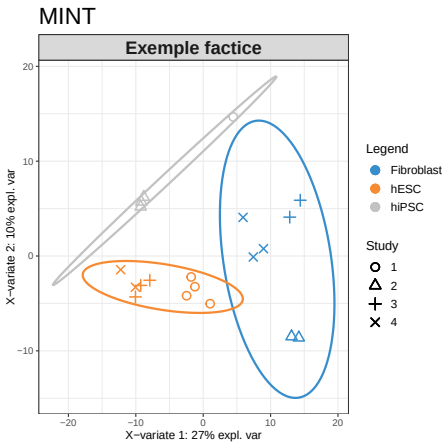
4.4.3 Sorties graphiques

La méthode MINT ayant aussi été développée par l'équipe de *MixOmics*, une grande partie des graphiques sont similaires à ceux de DIABLO et les noms des fonctions qui les appellent sont identiques également.

Graphiques d'échantillons

1. `plotIndiv()`

   Ce graphique a la même fonctionnalité que celui pour la méthode DIABLO. Appliqués à cette méthode, les options `study = "global"` ou `study = "all.partial"` définissent si la projection des données est faite sur les vecteurs de scores globaux ou partiels. Si la projection est faite sur les vecteurs de scores globaux, le pourcentage de la variance expliquée par chaque composante est ajouté. L'analyse des projections sur les vecteurs des scores globaux permet de vérifier que les données sont groupées selon les différentes classes de la variable d'intérêt $\mathbf{Y}$ et pas selon les différentes études.

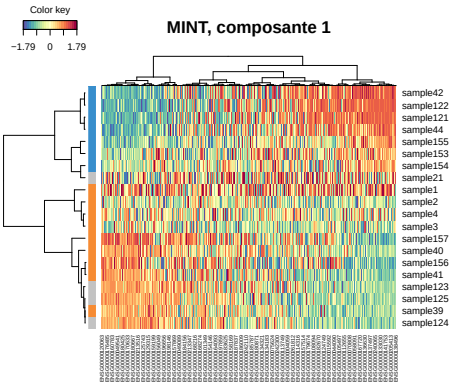


2. `plotArrow()`

   Ce graphique a exactement la même fonctionnalité et la même interprétation que celui de la méthode DIABLO, sauf que les flèches représentent les différentes études. L'allure du graphique ressemble très fort à celui obtenu via la fonction `plotIndiv()`.

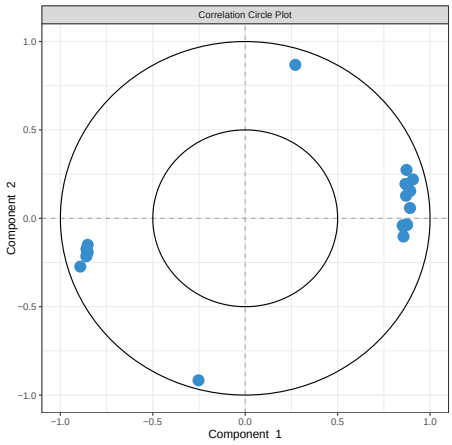
3. `cim()`

   Ce graphique représente une heatmap. Son interprétation est similaire à celle de l'output obtenu via la fonction `cimDiablo()` mais on remarque quelques petites différences : il est possible de choisir la mesure de distance (`euclidienne`, `coefficient de corrélation de Pearson`) et la méthode utilisée pour la réalisation du dendrogramme (`Ward`, `complète`). Il est dommage que, contrairement à l'output obtenu via `cimDiablo()`, ce graphique ne fournisse pas automatiquement la légende des groupes de la variable d'intérêt. Il est toutefois possible de l'encoder manuellement.



Graphiques de variables

4. `plotVar()`

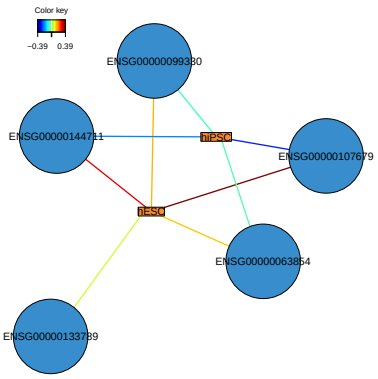


Ce graphique a la même fonctionnalité que pour la méthode DIABLO. Toutefois, étant donné qu'il n'y a qu'une seule modalité au sein des différents blocs (les différentes études), tous les points sont de la même couleur.

5. `plotLoadings()`

Ce graphique a exactement la même fonctionnalité et la même interprétation que pour la méthode DIABLO.

6. `network()`



Ce graphique a une interprétation différente de celui obtenu avec la même fonction sur un objet de type DIABLO. Il y a deux types de noeuds dans ce graphique : les biomarqueurs (cercles) et les différentes classes de la variable d'intérêt (rectangles). La taille des noeuds n'a aucune importance.
L'utilisateur peut définir un cutoff. Les biomarqueurs affichés dans ce graphique sont ceux dont la corrélation avec un ou plusieurs vecteurs de scores des groupes de la variable d'intérêt ($\mathbf{u}_h$) est supérieure au cutoff. La valeur de cette corrélation est représentée par la couleur des liens entre les noeuds.

Autres graphiques

7. `auroc()`

Ce graphique a exactement la même fonctionnalité et la même interprétation que pour la méthode DIABLO.

Chapitre 5

MOFA+

5.1 Principe de fonctionnement

Le framework MOFA+ ou MOFA.v2 (**M**ulti-**O**mics **F**actor **A**nalysis version 2) a été proposé par Argelaguet et al. en 2020. Cette méthode non-supervisée se base sur MOFA, qui avait été proposé par la même équipe deux ans plus tôt, en étendant son principe à l'intégration verticale. Ce framework est particulièrement adapté, d'un point de vue algorithmique, aux datasets de très grande taille et donc aux analyses single-cell.

MOFA+ utilise des combinaisons linéaires des variables originales pour projeter les données dans un sous-espace de plus petite dimension, tout comme les deux méthodes précédentes. Toutefois, ce framework a le challenge supplémentaire d'arriver à faire ceci en intégration horizontale, verticale, ou dans les deux sens en même temps. Les vecteurs de scores de MOFA+ sont appelés "Facteurs" dans les documents référencés en bibliographie et ce terme a parfois été retenu dans ce travail.

A la différence de DIABLO ou MINT, qui utilisent la variable d'intérêt pour la construction du modèle, MOFA+ est une méthode non-supervisée. Similairement à une PCA, MOFA+ vise à capturer une variance maximale des données dans le sous-espace engendré, tout en éliminant la variance non désirée.

Les objectifs principaux de cette méthode sont la détection de biomarqueurs et l'analyse de la variance des données au sein des composantes latentes. MOFA+ est considérée par Subramanian comme un framework ayant une approche bayesienne [22]. La raison en sera expliquée dans le Chapitre 5.3 : *Aspects mathématiques*. Cependant, comme cette technique utilise aussi des combinaisons linéaires des variables originales, on pourrait l'inclure dans les méthodes multivariées

Dans cette section, un maximum a été fait pour harmoniser les notations avec le reste du mémoire. Pour aider le lecteur qui le souhaiterait à faire le lien entre cette partie et le document de Argelaguet et al [5], une table a été ajoutée en Annexe B. En outre, la présentation des matrices dans les documents référencés en bibliographie se fait de manière transposée à ce qui est présenté dans ce travail : les lignes représentent les variables et les colonnes les observations.

5.2 Schéma de fonctionnement de MOFA+

La Figure 5.1 illustre le principe de fonctionnement de la méthode. Il est à noter que la matrice ϵ est une matrice des résidus. Cette matrice contient les informations non capturées dans le sous-espace calculé.

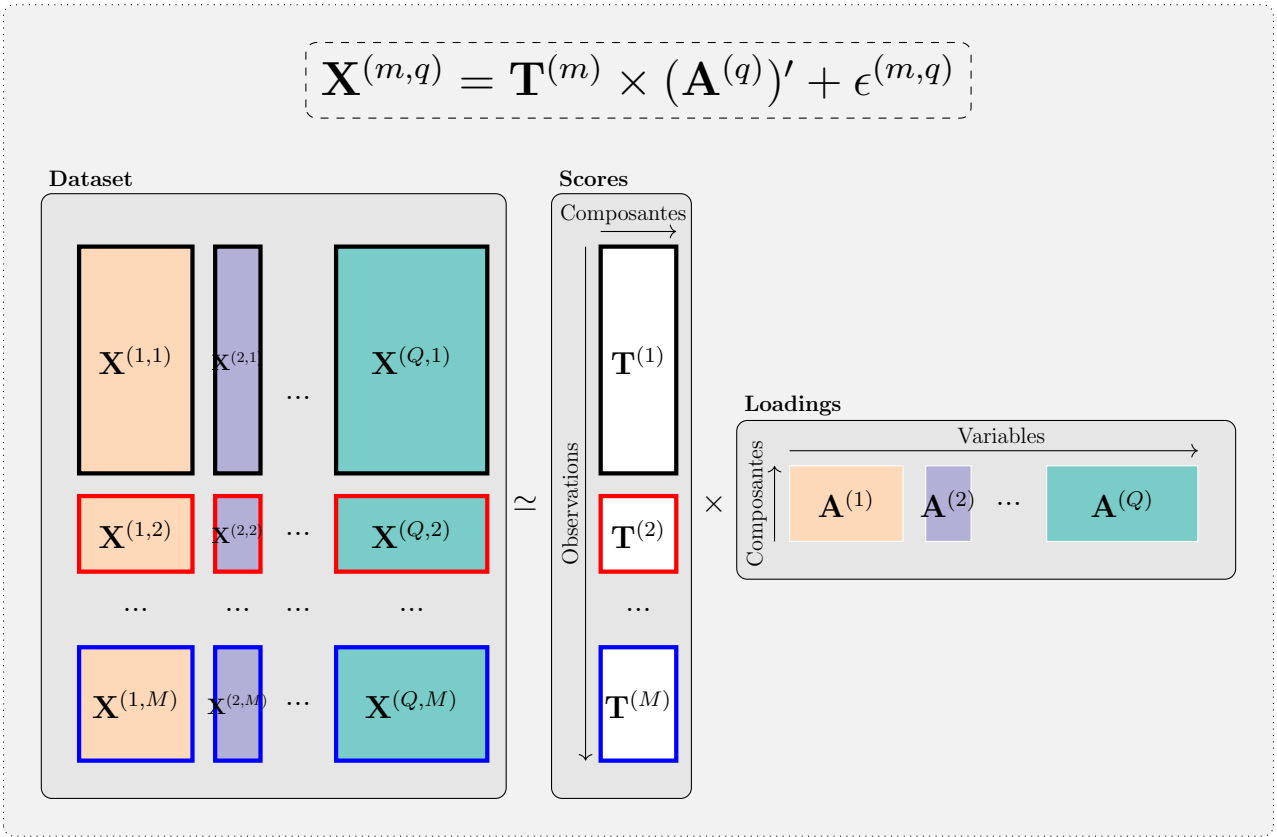


FIGURE 5.1: Schéma de principe de MOFA+. Image inspirée et modifiée de [5]

5.3 Aspects mathématiques

MOFA+ se base sur le framework MOFA, qui avait été proposé deux ans auparavant par la même équipe. MOFA+ étant une méthode relativement complexe, la section 5.3.1 expliquera dans un premier temps le principe de MOFA. La section 5.3.3 expliquera ensuite comment la méthode de MOFA a été adaptée à MOFA+ dans son principe général. Ensuite,

la section 5.3.4 détaillera plusieurs méthodes mathématiques complémentaires, permettant d'obtenir une solution à MOFA+. Finalement, la dernière section 5.3.5 expliquera l'algorithme de MOFA+ dans ses détails. Ce chapitre est directement inspirée de l'article de R. Argelaguet et al. ainsi que de ses annexes [6].

Dans cette section, un vecteur ou un scalaire provenant d'une matrice sera régulièrement représenté par la lettre capitale de la matrice dont il provient (en gras), ainsi que par deux indices. L'indice de gauche représente la ligne de la matrice et l'indice de droite la colonne de la matrice utilisés. Par exemple, la notation $\mathbf{Z}_{[i,.]}$ représente le vecteur provenant de la ligne i de la matrice $\mathbf{Z}$ et la notation $\mathbf{Z}_{[i,j]}$ définit le scalaire provenant de la ligne i et de la colonne j de la matrice $\mathbf{Z}$. De même, un scalaire provenant d'un vecteur sera régulièrement représenté par la lettre minuscule (en gras) du vecteur, accompagné d'un indice, désignant la place de l'élément dans ce vecteur. Par exemple, la notation $\mathbf{v}_{[i]}$ représente l'élément i du vecteur $\mathbf{v}$.

5.3.1 MOFA

MOFA est une méthode adaptée à l'intégration horizontale uniquement, et tente de capturer un maximum de variabilité des données dans le sous-espace engendré. Son schéma de fonctionnement est celui présenté en Figure 5.2 :

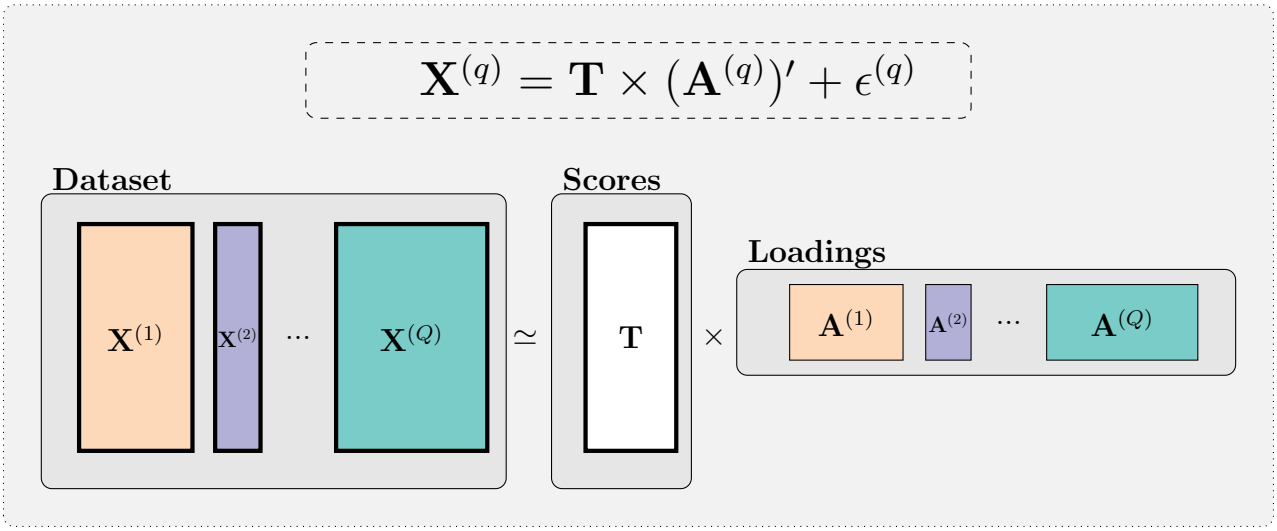


FIGURE 5.2: Schéma de principe de MOFA. Image inspirée et modifiée de [6]

Comme on peut le voir, le modèle proposé par MOFA est assez similaire à celui de MOFA+ mais les dimensions de certains éléments diffèrent. Les blocs peuvent être représentés par les vecteurs de scores, de loadings et les résidus suivants :

$$\mathbf{X}^{(q)} = \mathbf{T} \times (\mathbf{A}^{(q)})' + \epsilon^{(q)} \tag{5.1}$$

5.3.1.1 Dimensions des vecteurs et matrices utilisées

La Figure 5.3, inspirée de [6] (*Appendix : Supplementary Methods*) récapitule les dimensions des divers paramètres, vecteurs et matrices utilisés dans ce chapitre et illustre

la construction des éléments de chaque matrice. La couleur des assiettes fait référence aux dimensions des matrices et vecteurs et les flèches illustrent ce dont dépendent chaque éléments [1].

Voici un exemple de lecture de ce graphique : La matrice $\mathbf{A}^{(q)}$, contenant l'élément $\mathbf{A}^{(q)}_{[p,h]}$, est de dimensions $P^{(q)} \times H$ et il y a Q matrices $\mathbf{A}^{(q)}$. Elle se retrouve donc dans les trois assiettes représentant ces dimensions. Par ailleurs, on peut voir que l'élément $\mathbf{A}^{(q)}_{[p,h]}$ dépend des éléments des matrices $\mathbf{S}^{(q)}_{\mathbf{A}}$ et $\hat{\mathbf{A}}^{(q)}$ puisqu'il est le résultat du produit des éléments de ces deux matrices, pour la même ligne p et colonne h.

Davantage d'explications seront fournies sur les paramètres τ, $\mathbf{S}$, $\hat{\mathbf{A}}$, θ et α dans les sous-sections suivantes.

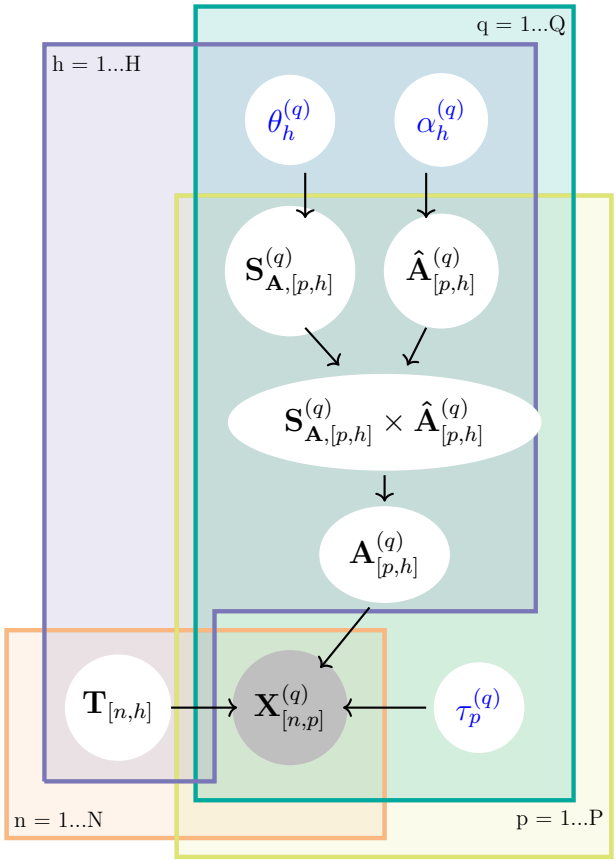


FIGURE 5.3: Illustration des dimensions des vecteurs et matrices utilisés dans MOFA et construction des éléments. Les bulles blanches représentent les éléments non observés et la bulle grise représente la matrice des données originale. Les vecteurs et matrices sont écrits en noir et les paramètres dont ils dépendent sont écrits en bleu. Les dépendances d'un élément par rapport à un autre sont représentées par les flèches. Image inspirée et transformée de [6].

1. Ce schéma n'est pas très rigoureux car, par exemple, la matrice $\mathbf{X}^{(q)}$ est de dimension $N \times P^{(q)}$ mais elle est également dans l'assiette Q car il y a Q matrices $\mathbf{X}^{(q)}$. Nous avons cependant laissé ce schéma car il nous a semblé utile pour la compréhension de la méthode.

5.3.2 Les hypothèses du modèle

1. **Hypothèse sur la distribution des résidus**

L'élément $\epsilon_{[p]}^{(q)}$ fait référence aux résidus de la colonne p du bloc $\mathbf{X}^{(q)}$. La distribution de ces résidus peut être spécifiée par l'utilisateur en fonction du type de données : une distribution suivant une loi Normale pour des données continues, une loi Benoulli pour des données binaires ou une loi de Poisson pour des données de comptage. Dans ce sous-chapitre, les vecteurs des résidus sont supposés suivre une loi Normale centrée à zéro et d'écart-type $1/\tau_p^{(q)}$:

$$p\big(\epsilon_p^{(q)}\big) \sim \mathcal{N}\big(0, 1/\tau_p^{(q)}\big)$$

Les résidus sont donc supposés être homoscédastiques au sein de la même variable mais cette homoscédasticité n'est pas requise entre les variables. Une valeur élevée du paramètre τ définit donc un modèle qui décrit précisément les données et qui a donc une valeur prédictive élevée. Au contraire une valeur basse de τ définit un écart-type plus important et des résidus qui varient davantage autour de leur moyenne et donc une moins bonne précision du modèle. La distribution de probabilité d'une observation $\mathbf{X}_{[n,p]}^{(q)}$ peut donc être définie par l'Equation suivante :

$$p\big(\mathbf{X}_{[n,p]}^{(q)}\big) \sim \mathcal{N}\big(\mathbf{T}_{[n,.]} \cdot (\mathbf{A}_{[p,.]}^{(q)})', 1/\tau_p^{(q)}\big)$$

2. **Les distributions à priori du modèle**

• **La précision des résidus**

Le paramètre $\tau_p^{(q)}$ est supposé suivre une distribution Gamma non informative :

$$p\big(\tau_p^{(q)}\big) \sim \mathcal{G}\big(a_0, b_0\big)$$

, avec une valeur de a_0 et b_0 proche de zéro. L'allure de la distribution de cette fonction est illustrée plus tard sur la Figure 5.5.

• **Les vecteurs de scores**

La distribution des scores peut également être définie à priori. Nous assumerons dans ce chapitre qu'elle suit une distribution de loi Normale :

$$p\big(\mathbf{T}_{[n,h]}\big) \sim \mathcal{N}\big(0, 1\big)$$

• **Les vecteurs de loadings**

Finalement, il y a lieu de définir la distribution à priori des loadings. Cette définition est plus complexe car deux niveaux de sparsité sont définis dans le modèle MOFA, comme illustré dans la Figure 5.4.

(a) Le premier niveau définit la sparsité en fonction des modalités et des composantes. Ceci permet d'identifier les modalités les plus influentes au sein de chaque composante (en attribuant un poids non nul à leurs variables), et d'attribuer un poids nul à l'ensemble des variables des modalités qui n'ont pas d'effet dans la composante.

(b) Le second niveau définit la sparsité au niveau des variables d'un même bloc en ne donnant de l'importance qu'à celles qui expliquent le plus de variabilité dans les données et en attribuant un poids moindre aux autres variabes.

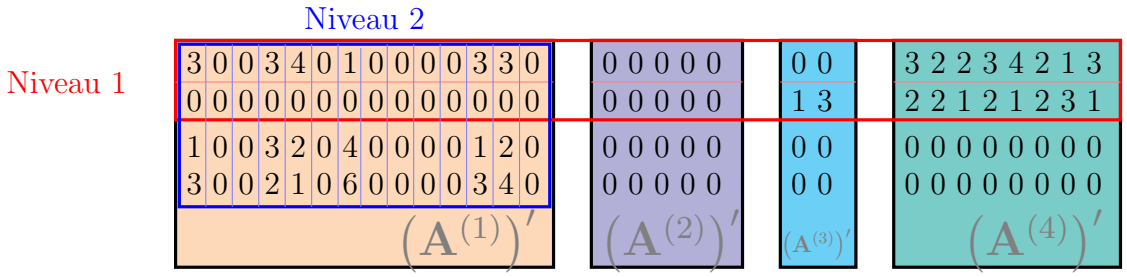


FIGURE 5.4: Illustration des deux niveaux de sparsité de la matrice des loadings $\mathbf{A}$. Le premier niveau de sparsité pourra être observé dans les études de cas sur les Figures 6.16 et 6.39.

Les deux niveaux de sparsité sont combinés de la manière suivante :

$$p\big(\mathbf{A}^{(q)}_{[p,h]}\big) \sim (1-\theta^{(q)}_h)\mathbb{I}_0(\mathbf{A}^{(q)}_{[p,h]}) + \theta^{(q)}_h \mathcal{N}\big(0, 1/\alpha^{(q)}_h\big) \tag{5.2}$$

, où $\mathbb{I}_0$ est une fonction delta de Dirac centrée à 0. Cette fonction est définie par une valeur égale à l'infini lorsque x vaut zéro et à zéro partout ailleurs. Cette fonction est caractérisée par une intégrale valant 1. L'utilisation de la fonction delta de Dirac pose cependant des problèmes algorithmiques. Pour palier à ce problème, chaque élément de la matrice des loadings $(\mathbf{A}_{[p,h]})$ peut être réécrit comme le produit des éléments de la matrice "de sparsité" $\mathbf{S}_{\mathbf{A},[p,h]}$ et de $\hat{\mathbf{A}}_{[p,h]}$:

$$\mathbf{A}_{[p,h]} = \mathbf{S}_{\mathbf{A},[p,h]} \cdot \hat{\mathbf{A}}_{[p,h]} \tag{5.3}$$

, où $\quad \mathbf{S}^{(q)}_{\mathbf{A},[p,h]} \sim \mathcal{B}er\big(\theta^{(q)}_h\big)$

et $\quad \hat{\mathbf{A}}^{(q)}_{[p,h]} \sim \mathcal{N}\big(0, 1/\alpha^{(q)}_h\big).$

La distribution de probabilité à priori des éléments des vecteurs de loadings peut donc s'écrire :

$$p\big(\mathbf{A}_{[p,h]}\big) \sim \mathcal{B}er\big(\theta_h^{(q)}\big) \cdot \mathcal{N}\big(0, 1/\alpha_h^{(q)}\big)$$

→ Le paramètre $\theta_h^{(q)}$ prend des valeurs entre 0 et 1 et détermine le niveau de sparsité de la composante h dans la modalité q et a une distribution non-informative :

$$p\big(\theta_h^{(q)}\big) \sim \mathcal{B}eta\big(1, 1\big)$$

Lorsque $\theta_h^{(q)}$ prend une valeur proche de zéro, cela signifie que la plupart des coefficients du vecteur de loadings valent zéro dans cette modalité. Dans ce cas, le modèle limite donc fortement le nombre de variables de cette modalité contribuant à la construction du vecteur de scores correspondant. Ce paramètre est donc lié à la régulation du niveau 2 de sparsité.

→ Le paramètre $\alpha_h^{(q)}$ détermine l'intensité du vecteur de loadings $\mathbf{a}_h$ dans la modalité q et suit une distribution non-informative :

$$p\big(\alpha_h^{(q)}\big) \sim \mathcal{G}\big(0.001, 0.001\big)$$

Ce paramètre est donc lié à la régulation du niveau 1 de sparsité.

La Figure ci-dessous illustre les distribution non-informatives des deux paramètres θ et α :

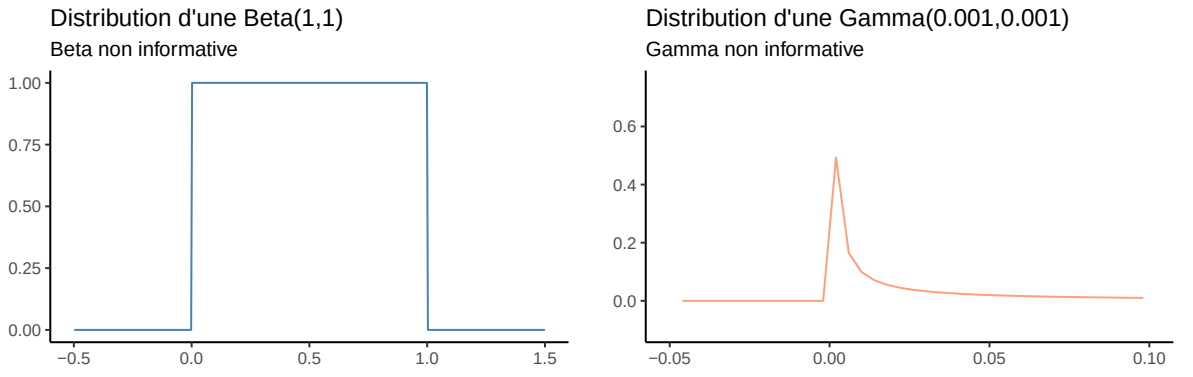


FIGURE 5.5: Fonctions non informatives utilisées pour les paramètres θ et α

Etant donné ces distribution à priori, la densité de probabilité jointe du modèle est définie

comme suit :

$$
\begin{aligned}
p\big(\mathbf{X}, \hat{\mathbf{A}}, \mathbf{S_A}, \mathbf{T}, \theta, \alpha, \tau\big) = & \prod_{q=1}^{Q}\prod_{n=1}^{N}\prod_{p=1}^{P^{(q)}} \mathcal{N}\Big(\sum_{h=1}^{H} \mathbf{S}^{(q)}_{\mathbf{A},[p,h]}\hat{\mathbf{A}}^{(q)}_{[p,h]}\mathbf{T}_{[n,h]}, \tfrac{1}{\tau_p}\Big) \\
& \times \prod_{q=1}^{Q}\prod_{p=1}^{P^{(q)}}\prod_{h=1}^{H} \mathcal{N}\Big(0, \tfrac{1}{\alpha_h^{(q)}}\Big)\mathcal{B}er\Big(\theta_h^{(q)}\Big) \\
& \times \prod_{n=1}^{N}\prod_{h=1}^{H} \mathcal{N}\big(0,1\big) \\
& \times \prod_{p=1}^{P^{(q)}}\prod_{h=1}^{H} \mathcal{B}eta\big(1,1\big) \\
& \times \prod_{p=1}^{P^{(q)}}\prod_{h=1}^{H} \mathcal{G}\big(0.001, 0.001\big) \\
& \times \prod_{q=1}^{Q}\prod_{p=1}^{P^{(q)}} \mathcal{G}\big(a_0, b_0\big)
\end{aligned}
\tag{5.4}
$$

5.3.3 Adaptation de MOFA à MOFA+

Deux améliorations importantes ont été apportées à MOFA pour aboutir à MOFA+ : l'adaptation à l'intégration verticale des données et la vitesse de calcul de l'algorithme. La possibilité d'intégrer les groupes verticalement s'accompagne d'un nouveau niveau de sparsité.

Illustrons ce besoin d'un troisième niveau de sparsité et le besoin de la reformulation de la distribution à priori des vecteurs de scores via un exemple. Imaginons une intégration multi-blocs dans les deux sens, avec deux études (blocs verticaux) de données omiques de trois modalités (blocs horizontaux). Il est possible qu'un effet soit présent dans la première étude, mais totalement absent dans la seconde. Il serait intéressant de pouvoir définir quelles études sont à prendre en compte dans la construction du vecteur des scores. Ceci définit le troisième niveau de sparsité, tel qu'illustré sur la Figure 5.6.

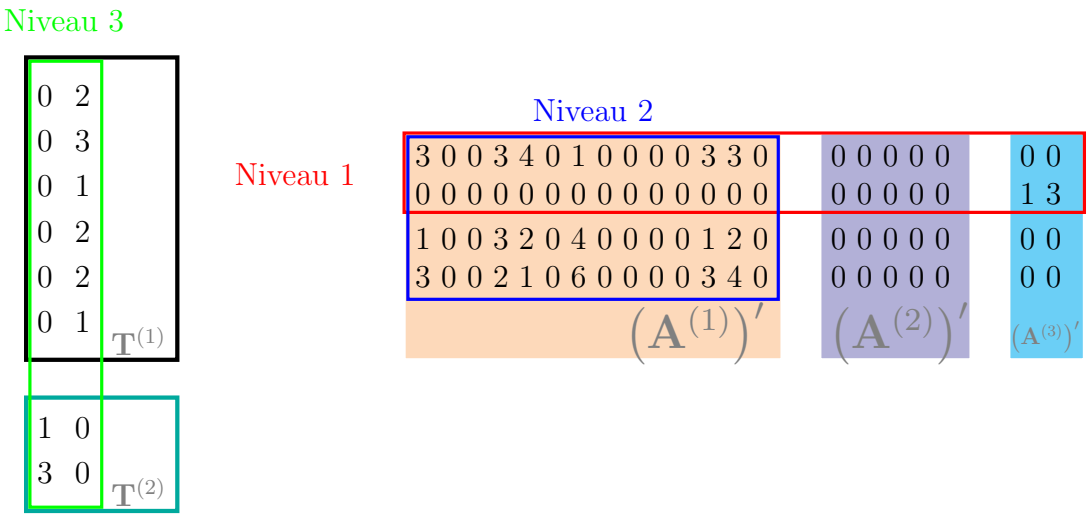


FIGURE 5.6: Illustration des trois niveaux de sparsité de la matrice des loadings $\mathbf{A}$ et de la matrice des scores $\mathbf{T}$

Ce nouveau niveau de sparsité, appliqué aux vecteurs des scores $\mathbf{t}$ nécessite une reformulation de la distribution à priori des éléments. De manière similaire aux éléments de la matrice des loadings de MOFA, chaque élement de la matrice de scores ($\mathbf{T}_{[n,h]}$) de MOFA+ peut être redéfini comme le produit des élements de la matrice "de sparsité" $\mathbf{S}_{\mathbf{T},[n,h]}$ et de $\hat{\mathbf{T}}_{[n,h]}$:

$$\mathbf{T}_{[n,h]} = \mathbf{S}_{\mathbf{T},[n,h]} \cdot \hat{\mathbf{T}}_{[n,h]}$$

, où $\mathbf{S}_{\mathbf{T},[n,h]} \sim \mathcal{B}er\big(\theta_h^{(m)}\big)$
et $\hat{\mathbf{T}}_{[n,h]} \sim \mathcal{N}\big(0, 1/\alpha_h^{(m)}\big)$

La distribution de probabilité à priori des éléments de la matrice de scores est la suivante :

$$p\big(\mathbf{T}_{[n,h]}\big) \sim \mathcal{B}er\big(\theta_h^{(m)}\big) \cdot \mathcal{N}\big(0, 1/\alpha_h^{(m)}\big) \tag{5.5}$$

Et la distribution de probabilité à priori des nouveaux paramètres $\theta_h^{(m)}$ et $\alpha_h^{(m)}$:

$$p\big(\theta_h^{(m)}\big) = \mathcal{B}eta\big(a_0, b_0\big) \tag{5.6}$$

$$p\big(\alpha_h^{(m)}\big) = \mathcal{G}\big(a_0, b_0\big) \tag{5.7}$$

, Avec des valeurs pour a_0 et b_0 proches de zéro. Par ailleurs, étant donné que les observations peuvent fortement varier d'une étude à l'autre, la distribution des résidus n'est plus forcément identique entre les différents blocs. L'hypothèse sur ces résidus reste qu'ils sont distribués selon une loi Normale autour de leur moyenne centrée à zéro, mais l'écart-type peut varier d'un bloc à l'autre :

$$p\big(\epsilon_p^{(m,q)}\big) \sim \mathcal{N}\big(0, 1/\tau_p^{(m,q)}\big) \tag{5.8}$$

$$p\big(\tau_p^{(m,q)}\big) \sim \mathcal{G}\big(a_0, b_0\big) \tag{5.9}$$

L'interprétation des différents paramètres reste cependant identique à celle donnée dans MOFA.

Les dimensions des paramètres, des vecteurs et des matrices, ainsi que la construction des éléments de la matrice des scores $\mathbf{T}$ ont changé et sont illustrées sur la Figure 5.7.

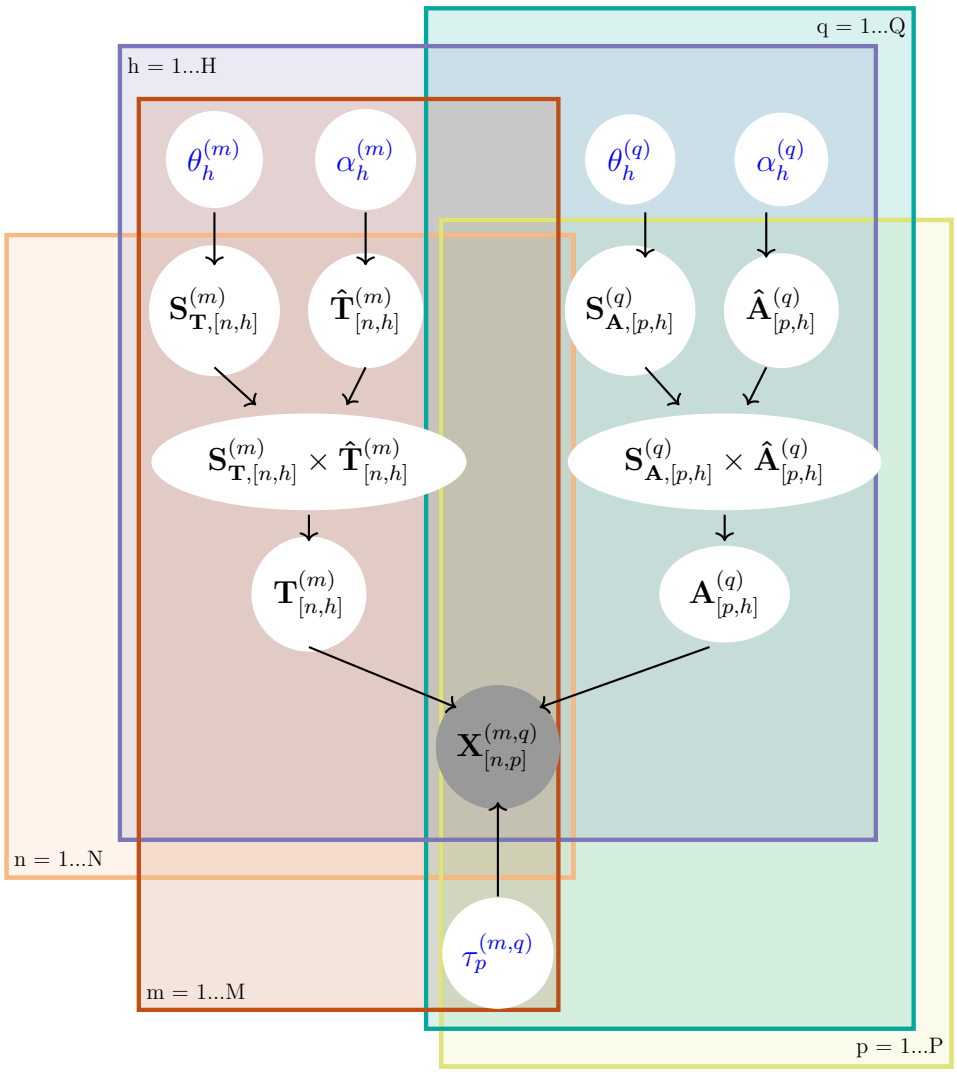


FIGURE 5.7: Illustration des dimensions des vecteurs et matrices utilisés dans MOFA+ et construction des élements. Les bulles blanches représentent les éléments non observés et la bulle grise représente la matrice de données originale. Les éléments des matrices sont écrits en noir et les paramètres dont ils dépendent sont écrits en bleu. Les dépendances d'un élément par rapport à un autre sont représentées par les flèches. Image inspirée et transformée de [5].

Pour conclure cette section, notons que le nombre de facteurs peut être déterminé soit via un seuil minimal de variance expliquée par vecteur de scores, soit fixé manuellement par l'utilisateur.

5.3.4 Méthodes complémentaires

Cette partie s'inspire de l'annexe *Supplementary Methods* de l'article *MOFA+ : a statistical framework for comprehensive integration of multi-modal single-cell data* [5]. Cette section étant plus complexe, nous en expliquons ici l'idée générale. Dans un premier temps, le concept d'**I**nférence **V**ariationnelle (VI) et de distribution variationnelle ($q(\mathbf{W})$) seront brièvement expliqués (5.3.4.1). Pour ce faire, les notions de divergence de **K**ullback-**L**eibler (KL) et d'**E**vidence **L**ower **BO**und (ELBO) y seront également abordées. Ces nouveaux

concepts sont utiles pour expliquer le principe suivant :

Soit un modèle génératif défini par des observations $\mathbf{X}$ et des variables latentes $\mathbf{W}$. Dans le contexte de MOFA+, les variables latentes $\mathbf{W}$ représentent les éléments $\hat{\mathbf{T}}^{(m)}$, $\alpha^{(m)}$, $\mathbf{S}_{\mathbf{T}}^{(m)}$, $\theta^{(m)}$, $\hat{\mathbf{A}}^{(q)}$, $\alpha^{(q)}$, $\mathbf{S}_{\mathbf{A}}^{(q)}$, $\theta^{(q)}$ et $\tau^{(m,q)}$. La vraie - mais inconnue - distribution de probabilité à posteriori $p(\mathbf{W}|\mathbf{X})$ est estimée par une distribution variationnelle $q(\mathbf{W})$. Cette distribution est celle qui se rapproche le plus de la distribution $p(\mathbf{W}|\mathbf{X})$. Cette similarité entre les deux distributions est mesurée par la divergence de Kullback-Leibler et est également celle qui maximise la valeur de l'ELBO. Ce principe sera représenté par la Figure 5.9.

Ensuite, l'algorithme de Stochastic gradient ascent naturel sera brièvement expliqué (5.3.4.2). Ce dernier algorithme est appliqué afin de retrouver les variables latentes $\mathbf{W}$ qui maximisent la valeur de l'ELBO $\mathcal{L}(\mathbf{W})$ et donc de trouver la distribution variationnelle $q(\mathbf{W})$ qui se rapproche le plus de la vraie distribution à posteriori $p(\mathbf{W}|\mathbf{X})$.

Dans le cadre de ce travail, l'objectif de cette section est de comprendre l'intuition de ces différents concepts, mais pas de les expliquer dans leurs détails mathématiques.

5.3.4.1 l'Inférence Variationnelle et la distribution variationnelle : $q(\mathbf{W})$

Imaginons un modèle génératif défini par des observations $\mathbf{X}$ et des variables latentes $\mathbf{W}$. Le théorème de Bayes nous permet d'écrire :

$$p\big(\mathbf{W}|\mathbf{X}\big) \quad = \frac{p\big(\mathbf{W},\mathbf{X}\big)}{\int_w p\big(\mathbf{W},\mathbf{X}\big)} \tag{5.10}$$

Toutefois, cette distribution peut être très compliquée à calculer dans la pratique et peut être approximée par une autre distribution $q\big(\mathbf{W}\big)$, appelée aussi distribution variationnelle. La distribution variationnelle est celle qui approxime le mieux la vraie distribution à posteriori $p\big(\mathbf{W}|\mathbf{X}\big)$ parmi une série de famille de distributions prédéfinies.

La divergence entre ces deux distributions peut être calculée en utilisant la divergence de Kullback-Leibler qui est définie de la sorte [18] :

$$KL\big(q(\mathbf{W})||p(\mathbf{W}|\mathbf{X})\big) = KL\big(q||p\big) = \int q\big(x\big) \log \frac{q\big(x\big)}{p\big(x\big)} dx \tag{5.11}$$

L'utilité de la divergence de Kullback-Leibler est illustrée sur la Figure 5.8.

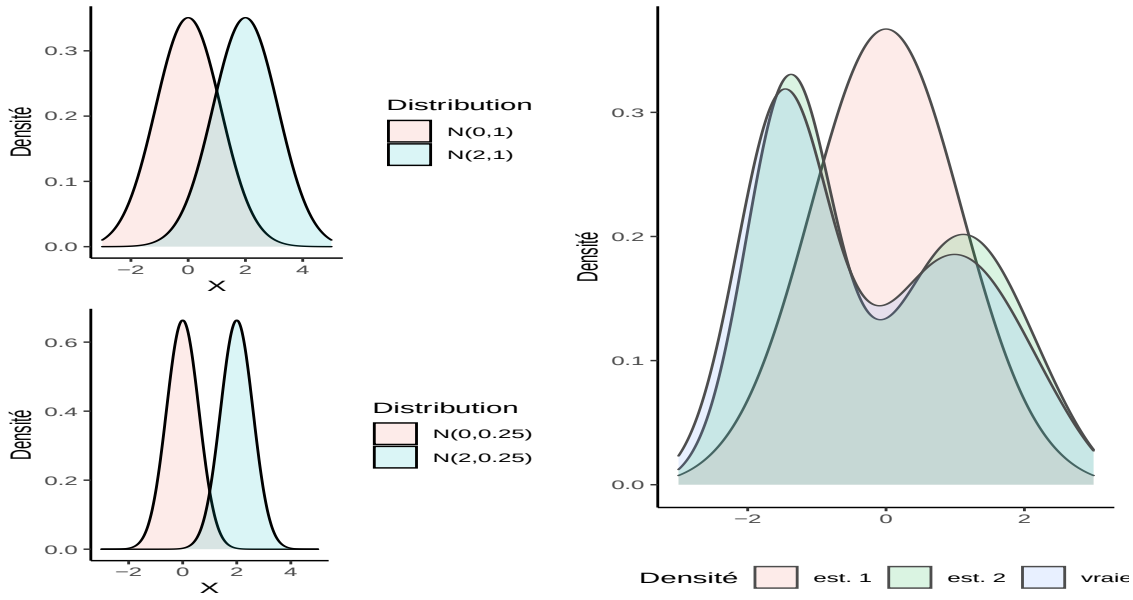


FIGURE 5.8: Illustration de la divergence de Kullback-Leibler. A gauche : la distance euclidienne entre les moyennes des distributions rouge et bleue est identique sur les graphiques du haut et du bas. Toutefois, la divergence de Kullback-Leibler est plus importante dans le graphique du bas, où l'écart-type des Normales est plus petit. A droite : Trois distributions sont représentées : la vraie distribution (inconnue) en bleu et deux distributions qui l'approximent, en vert et en rouge. La divergence de Kullback-Leibler sera plus petite entre la distribution bleue et la distribution verte, qu'entre la distribution bleue et la distribution rouge puisque la distribution verte approxime bien mieux la bleue.

M. Svensén et C. Bishop ont démontré [23] que l'équation (5.11) était équivalante à :

$$KL\big(q||p\big) = \overbrace{\log\big(p(\mathbf{X})\big)}^{Cste} - \overbrace{\mathcal{L}\big(\mathbf{W}\big)}^{ELBO} \tag{5.12}$$

où le premier terme est une constante et le second terme correspond à l'**E**vidence **L**ower **BO**und (ELBO). L'ELBO peut donc être représenté selon la Figure 5.9.

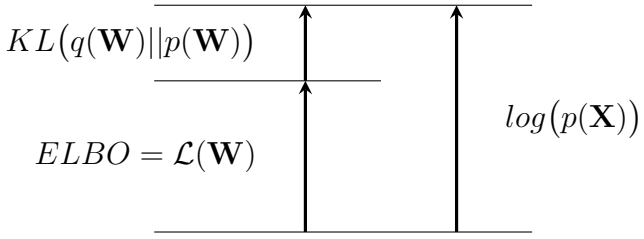


FIGURE 5.9: Illustration de la relation entre la divergence de Kullback-Leibler et l'ELBO. Image inspirée et modifiée de l'annexe *Supplementary Methods* de [5]

Minimiser la divergence de Kullback-Leibler revient donc à maximiser l'ELBO. Ce terme est défini comme la différence entre (a) l'espérance du logarithme de la probabilité jointe de

$\mathbf{W}$ et $\mathbf{X}$ et (b) l'entropie de la distribution variationnelle :

$$ELBO = \mathcal{L}(\mathbf{W}) = \underbrace{\mathbb{E}_q\Big[\log p\big(\mathbf{W}, \mathbf{X}\big)\Big]}_{(a)} - \underbrace{\mathbb{E}_q\Big[\log q\big(\mathbf{W}\big)\Big]}_{(b)} \tag{5.13}$$

5.3.4.2 Stochastic gradient ascent naturel

La recherche d'un maximum d'une fonction peut se faire via un algorithme de gradient ascent. En pratique, MOFA+ utilise une variante stochastique de cet algorithme et l'adapte au gradient naturel afin de trouver les valeurs optimales des variables latentes $\mathbf{W}$ qui maximisent l'ELBO $\mathcal{L}(\mathbf{W})$.

Le gradient ascent est un algorithme itératif qui a pour but de trouver le maximum d'une fonction. L'idée de l'algorithme est similaire à celle du *gradient descent*, ou *la descente de plus forte pente* (qui ne sera pas vu dans ce travail) mais la recherche d'optimisation se fait en sens inverse par rapport au gradient de la fonction. L'idée est que les coordonnées d'un point d'une fonction dérivable $F(w)$ sont itérées jusqu'à ce qu'elles convergent, selon la formule suivante :

$$w_{i+1} = w_i + \rho_i \nabla F(w_i) \tag{5.14}$$

, où i représente l'index de l'itération, $\nabla F(w_i)$ est le gradient de la fonction calculé aux coordonnées w_i, définissant la direction de la plus forte pente et ρ_i représente la valeur du pas fait dans cette direction à l'itération i. La valeur de ρ peut être réévaluée au cours des itérations de l'algorithme. Le concept de l'algorithme est illustré sur la Figure 5.10.

Au fur et à mesure des itérations et sous les conditions de Robbins-Monro [2], l'algorithme assure que les coordonnées du point se déplacent dans la direction d'un maximum local de la fonction.

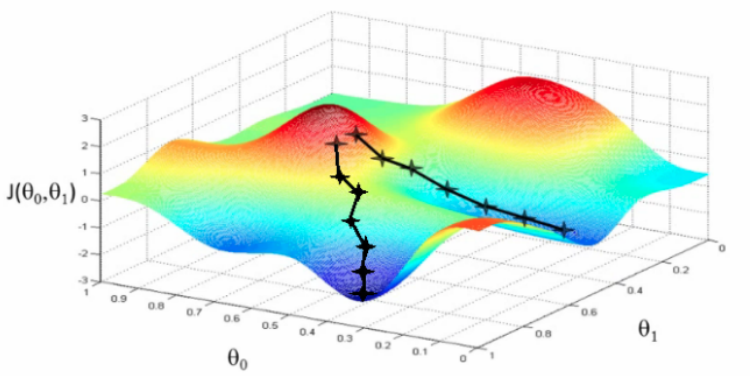


FIGURE 5.10: Illustration du fonctionnement du Gradient Ascent, selon [8]. Les coordonnées de deux points, pris en différents endroits, se dirigent vers un maximum local de la fonction, en suivant la plus forte pente positive, dont la direction est définie par le gradient de cette fonction $\nabla F(w_i)$ et à une vitesse définie par ρ_i.

2. $\sum_i \rho_i = \infty$ et $\sum_i (\rho_i)^2 < \infty$

Les jeux de données des analyses single-cell, pour lesquelles MOFA+ est particulièrement adaptée, sont généralement de très grande dimension. Calculer le gradient de la fonction $\mathcal{L}(\mathbf{W})$ à chaque itération peut donc prendre un temps considérable, et ce temps de calcul peut être encore plus long si plusieurs analyses single-cell sont intégrées verticalement. Partant de l'hypothèse qu'il y a des redondances dans les observations du jeu de données, la méthode utilise la version stochastique du gradient ascent, qui consiste à utiliser l'estimation du gradient de la fonction $(\nabla \hat{F}(w))$ à la place du gradient exact, en calculant cette estimation sur un échantillon des données, un *minibatch* :

$$w_{i+1} = w_i + \rho_i \nabla \hat{F}(w_i) \tag{5.15}$$

Notons finalement que MOFA+ utilise le gradient *naturel* dans son algorithme de stochastic gradient ascent. L'objectif de l'utilisation de ce gradient naturel est que cela permet à l'algorithme de se déplacer dans la direction de la plus grande pente définie par la distance de Kullback-Leibler au lieu de tenir compte de la distance euclidienne, comme le fait l'algorithme de gradient ascent classique. Le gradient naturel de la fonction ELBO pour un paramètre θ est donc défini comme :

$$\nabla_{KL}\mathcal{L}(\theta) = \lim_{h\to 0} \frac{1}{h} \arg \max_{d\, s.t. KL\Big((p(\theta)||p(\theta+d)\Big)=h} \mathcal{L}(\theta+d) - \mathcal{L}(\theta) \tag{5.16}$$

Où la distance en termes de divergence de Kullback-Leibler est définie comme suit :

$$\hat{d}_{KL} \propto \mathbf{F}^{-1}(\theta)\nabla\mathcal{L}(\theta) \tag{5.17}$$

Et où $\mathbf{F}$ est la matrice d'information de Fisher pour $q(w|\theta)$ et est définie comme :

$$\mathbf{F}(\theta) = \mathbb{E}_{q(w|\theta)}[\big(\nabla \log q(w|\theta)\big)\big(\nabla \log q(w|\theta)\big)'] \tag{5.18}$$

5.3.5 Algorithme

Argelaguet et al. ont démontré que l'algorithme 4 pouvait être vu comme un algorithme de stochastic gradient ascent naturel [5].

Dans cette partie, lorsqu'il est fait mention des *paramètres locaux*, cela fait référence à $\phi^{(m)}_{[n,h]} = \{\hat{\mathbf{T}}^{(m)}_{[n,h]}, \mathbf{S}^{(m)}_{\mathbf{T},[n,h]}\}$ t.q. $n \in \mathcal{B}$, où $\mathcal{B}$ est le minibatch.

Lorsqu'il est fait mention des *paramètres globaux*, cela fait référence à $\psi = \{\tau^{(m,q)}, \hat{\mathbf{A}}^{(q)}, \alpha^{(q)}, \theta^{(q)}, \mathbf{S}^{(q)}_{\mathbf{A}}, \alpha^{(m)}, \theta^{(m)}\}$.

Algorithm 4 : MOFA+

Initialisation aléatoire des paramètres des variables globales ψ
Initialisation du pas $\rho_{i=0}$
Répéter :
    Création d'un minibatch $\mathcal{B}$ avec un nombre d'observations $B << N$
    A. Update des paramètres variationnels locaux $\phi^{(m)}$:
        Pour chaque $\phi^{(m)}_{[n,h]}$:
            $\phi^{(m)}_{[n,h],i+1}$ est le paramètre $\phi^{(m)}_{[n,h]}$ actualisé selon les équations définies en 5.3.6.1
    B. Update des paramètres variationnels globaux ψ :
        Pour chaque ψ :
            $\psi_{i+1} = (1-\rho_i)\cdot\psi_i + \rho_i \cdot \psi^{\mathcal{B}}_{i+1}$
            , où $\psi^{\mathcal{B}}_{i+1}$ est le paramètre ψ actualisé selon les équations définies en 5.3.6.2 sur le minibatch $\mathcal{B}$ et répété N/B fois
Jusqu'à la convergence de l'ELBO de $q(\mathbf{TA}' + \epsilon)$

Notons finalement que MOFA+ parvient à diminuer encore davantage son temps de compilation en utilisant les capacités du processeur graphique de l'ordinateur (GPU-accelerated Stochastic Variational Inference), mais cette dernière variante ne sera pas abordée dans ce travail.

5.3.6 Equations nécessaires à l'algorithme

Il convient avant tout de définir la distribution variationnelle du modèle :

$$
\begin{aligned}
q\big(\mathbf{TA}' + \epsilon\big) = \; & q\big(\{\hat{\mathbf{T}}^{(m)}, \mathbf{S}^{(m)}_{\mathbf{T}}, \alpha^{(m)}, \theta^{(m)}\}, \{\hat{\mathbf{A}}^{(q)}, \mathbf{S}^{(q)}_{\mathbf{A}}, \alpha^{(q)}, \theta^{(q)}\}, \{\epsilon^{(m,q)}\}\big) \\
& = \prod_{m=1}^{M}\prod_{n=1}^{N}\prod_{h=1}^{H} q\big(\hat{\mathbf{T}}^{(m)}_{[n,h]}, \mathbf{S}^{(m)}_{\mathbf{T},[n,h]}\big) \prod_{m=1}^{M}\prod_{h=1}^{H} q\big(\alpha^{(m)}_h\big) \prod_{m=1}^{(m)}\prod_{h=1} q\big(\theta^{(m)}_h\big) \\
& \times \prod_{q=1}^{Q}\prod_{p=1}^{P^{(q)}}\prod_{h=1}^{H} q\big(\hat{\mathbf{A}}^{(q)}_{[p,h]}, \mathbf{S}^{(q)}_{\mathbf{A},[p,h]}\big) \prod_{q=1}^{Q}\prod_{h=1}^{H} q\big(\alpha^{(q)}_h\big) \prod_{q=1}^{Q}\prod_{h=1}^{H} q\big(\theta^{(q)}_h\big) \\
& \times \prod_{m=1}^{M}\prod_{q=1}^{Q}\prod_{p=1}^{P^{(q)}} q\big(\epsilon^{(m,q)}_p\big)
\end{aligned}
\tag{5.19}
$$

Les équations qui suivent définissent la distribution variationnelle de chaque terme de l'équation précédente.

Afin de tenter d'alléger les notations mathématiques de cette partie, l'expression $\mathbb{E}_q[x]$, définissant l'espérance de x sous la distribution q, a été remplacée par $\langle x \rangle$.

5.3.6.1 Paramètres variationnels locaux

- **Construction des facteurs sparse $\mathbf{T}^{(m)}_{[n,h]}$: $\mathbf{S}^{(m)}_{\mathbf{T},[n,h]}$ et $\hat{\mathbf{T}}^{(m)}_{[n,h]}$**

$$
\boxed{q(\mathbf{S}^{(m)}_{\mathbf{T},[n,h]}) \sim \mathcal{B}er(\gamma^{(m)}_{nh})}
\tag{5.20}
$$

Avec

$$\begin{aligned}
\gamma_{nh}^{(m)} &= \frac{1}{1+\exp(-\delta_{nh}^{(m)})} \\
\delta_{nh}^{(m)} &= \langle \ln \frac{\theta_h^{(m)}}{1-\theta_h^{(m)}} \rangle + 0.5 \ln \frac{\langle \alpha_h^{(m)} \rangle}{\langle \tau_p^{(m,q)} \rangle} - 0.5 \ln \Big(\sum_{q=1}^{Q} \sum_{p=1}^{P^{(q)}} \langle (\mathbf{A}_{[p,h]}^{(q)})^2 \rangle + \frac{\langle \alpha_h^{(m)} \rangle}{\langle \tau_p^{(m,q)} \rangle} \Big) \\
&\quad + \frac{\langle \tau_p^{(m,q)} \rangle}{2} \frac{\Big(\sum_{q=1}^{Q} \sum_{p=1}^{P^{(q)}} \mathbf{X}_{[n,p]}^{(m,q)} \langle \mathbf{A}_{[p,h]}^{(q)} \rangle - \sum_{j \neq h} \{ \langle \mathbf{S}_{[n,j]}^{(m)} \hat{\mathbf{T}}_{[n,j]}^{(m)} \rangle \sum_{q=1}^{Q} \sum_{p=1}^{P^{(q)}} \langle \mathbf{A}_{[p,h]}^{(q)} \rangle \langle \mathbf{A}_{[p,j]}^{(q)} \rangle \} \Big)^2}{\sum_{q=1}^{Q} \sum_{p=1}^{P^{(q)}} \langle (\mathbf{A}_{[p,h]}^{(q)})^2 \rangle + \frac{\langle \alpha_h^{(m)} \rangle}{\langle \tau_p^{(m,q)} \rangle}}
\end{aligned}$$

$$\boxed{q(\hat{\mathbf{T}}_{[n,h]}^{(m)} | \mathbf{S}_{\mathbf{T},[n,h]}^{(m)} = 0) \sim \mathcal{N}(0, \frac{1}{\alpha_h^{(q)}})} \tag{5.21}$$

$$\boxed{q(\hat{\mathbf{T}}_{[n,h]}^{(m)} | \mathbf{S}_{\mathbf{T},[n,h]}^{(m)} = 1) \sim \mathcal{N}(\mu_{\mathbf{T}_{[n,h]}^{(m)}}, \sigma^2_{\mathbf{T}_{[n,h]}^{(m)}})} \tag{5.22}$$

Avec

$$\begin{aligned}
\mu_{\mathbf{T}_{[n,h]}^{(m)}} &= \frac{\sum_{q=1}^{Q} \sum_{p=1}^{P^{(q)}} \mathbf{X}_{[n,p]}^{(m,q)} \langle \mathbf{A}_{[p,h]}^{(q)} \rangle - \sum_{j \neq h} \{ \langle \mathbf{S}_{\mathbf{T},[n,j]}^{(m)} \hat{\mathbf{T}}_{[n,j]}^{(m)} \rangle \sum_{q=1}^{Q} \sum_{p=1}^{P^q} \langle \mathbf{A}_{[p,h]}^{(q)} \rangle \langle \mathbf{A}_{[p,j]}^{(q)} \rangle \}}{\sum_{q=1}^{Q} \sum_{p=1}^{P^{(q)}} \langle (\mathbf{A}_{[p,h]}^{(q)})^2 \rangle + \frac{\langle \alpha_h^{(m)} \rangle}{\langle \tau_p^{(m,q)} \rangle}} \\
\sigma^2_{\mathbf{T}_{[n,h]}^{(m)}} &= \frac{\langle \tau_p^{(m,q)} \rangle^{-1}}{\sum_{q=1}^{Q} \sum_{p=1}^{P^{(q)}} \langle (\mathbf{A}_{[p,h]}^{(q)})^2 \rangle + \frac{\alpha_h^{(m)}}{\tau_p^{(m,q)}}}
\end{aligned}$$

5.3.6.2 Paramètres variationnels globaux

• **Paramètre de précision des facteurs :** $\alpha_h^{(m)}$

$$\boxed{q(\alpha_h^{(m)}) \sim \mathcal{G}(\hat{a}_h^{(m)}, \hat{b}_h^{(m)})} \tag{5.23}$$

Avec

$$\begin{aligned}
\hat{a}_h^{(m)} &= a_0 + \frac{N^{(m)}}{2} \\
\hat{b}_h^{(m)} &= b_0 + \frac{\sum_{n=1}^{N^{(m)}} \langle (\hat{\mathbf{T}}_{[n,h]}^{(m)})^2 \rangle}{2}
\end{aligned}$$

• **Paramètre de sparsité des facteurs :** $\theta_h^{(m)}$

$$\boxed{q(\theta_h^{(m)}) = \mathcal{B}eta(\hat{a}_h^{(m)}, \hat{b}_h^{(m)})} \tag{5.24}$$

Avec

$$\begin{aligned}
\hat{a}_h^{(m)} &= \sum_{n=1}^{N^{(m)}} \langle \mathbf{S}_{\mathbf{T},[n,h]}^{(m)} \rangle + a_0 \\
\hat{b}_h^{(m)} &= b_0 - \sum_{n=1}^{N^{(m)}} \langle \mathbf{S}_{\mathbf{T},[n,h]}^{(m)} \rangle + N^{(m)}
\end{aligned}$$

• **Construction des paramètres de loadings $\mathbf{A}_{[p,h]}^{(q)}$: $\mathbf{S}_{\mathbf{A},[p,h]}^{(q)}$ et $\hat{\mathbf{A}}_{[p,h]}^{(q)}$**

$$\boxed{q(\mathbf{S}_{\mathbf{A},[p,h]}^{(q)}) = \mathcal{B}er(\gamma_{ph}^{(q)})} \tag{5.25}$$

Avec

$$\begin{aligned}
\gamma_{ph}^{(q)} &= \tfrac{1}{1+\exp(-\delta_{ph}^{(q)})} \\
\delta_{hp}^{(q)} &= \langle \ln \tfrac{\theta_h^{(q)}}{1-\theta_h^{(q)}} \rangle + 0.5 \ln \tfrac{\langle \alpha_h^{(q)} \rangle}{\langle \tau_p^{(m,q)} \rangle} - 0.5 \ln \Big(\sum_{m=1}^{M} \sum_{n=1}^{N_m} \langle (\mathbf{T}_{[n,h]}^{(m)})^2 \rangle + \tfrac{\langle \alpha_h^{(q)} \rangle}{\langle \tau_p^{(m,q)} \rangle} \Big) \\
&\quad + \tfrac{\langle \tau_p^{(m,q)} \rangle}{2} \frac{\Big(\sum_{m=1}^{M} \sum_{n=1}^{N^{(m)}} \mathbf{X}_{[n,p]}^{(m,q)} \langle \mathbf{T}_{[n,h]}^{(m)} \rangle - \sum_{j \neq h} \{ \langle \mathbf{S}_{[j,p]}^{(q)} \hat{\mathbf{A}}_{[j,p]}^{(q)} \rangle \sum_{m=1}^{M} \sum_{n=1}^{N^{(m)}} \langle \mathbf{T}_{[n,h]}^{(m)} \rangle \langle \mathbf{T}_{[n,j]}^{(m)} \rangle \} \Big)^2}{\sum_{m=1}^{M} \sum_{n=1}^{N^{(m)}} \langle (\mathbf{T}_{[n,h]}^{(m)})^2 \rangle + \frac{\langle \alpha_h^{(q)} \rangle}{\langle \tau_p^{(m,q)} \rangle}}
\end{aligned}$$

$$\boxed{q(\hat{\mathbf{A}}_{[p,h]}^{(q)} | \mathbf{S}_{\mathbf{A},[p,h]}^{(q)} = 0) = \mathcal{N}(0, \tfrac{1}{\alpha_h^{(q)}})} \tag{5.26}$$

$$\boxed{q(\hat{\mathbf{A}}_{[p,h]}^{(q)} | \mathbf{S}_{\mathbf{A},[p,h]}^{(q)} = 1) = \mathcal{N}(\mu_{\mathbf{A}_{[p,h]}^{(q)}}, \sigma^2_{\mathbf{A}_{[p,h]}^{(q)}})} \tag{5.27}$$

Avec

$$\begin{aligned}
\mu_{\mathbf{A}_{[p,h]}^{(q)}} &= \frac{\sum_{m=1}^{M} \sum_{n=1}^{N^{(m)}} \mathbf{X}_{[n,p]}^{(m,q)} \langle \mathbf{T}_{[n,h]}^{(m)} \rangle - \sum_{j \neq h} \{ \langle \mathbf{S}_{[jp,j}^{(q)} \hat{\mathbf{A}}_{[p,j]}^{(q)} \rangle \sum_{m=1}^{M} \sum_{n=1}^{N^{(m)}} \langle \mathbf{T}_{[n,h]}^{(m)} \rangle \langle \mathbf{T}_{[n,j]}^{(m)} \rangle \}}{\sum_{m=1}^{M} \sum_{n=1}^{N^{(m)}} \langle (\mathbf{T}_{[n,h]}^{(m)})^2 \rangle + \frac{\langle \alpha_h^{(q)} \rangle}{\langle \tau_p^{(m,q)} \rangle}} \\
\sigma^2_{\mathbf{A}_{[p,h]}^{(q)}} &= \frac{\langle \tau_p^{(m,q)} \rangle^{-1}}{\sum_{m=1}^{M} \sum_{n=1}^{N^{(m)}} \langle (\mathbf{T}_{[n,h]}^{(m)})^2 \rangle + \frac{\langle \alpha_h^{(q)} \rangle}{\langle \tau_p^{(m,q)} \rangle}}
\end{aligned}$$

• **Paramètre de précision des vecteurs de loadings $\alpha_h^{(q)}$**

$$\boxed{q(\alpha_h^{(q)}) = \mathcal{G}(\hat{a}_h^{(q)}, \hat{b}_h^{(q)})} \tag{5.28}$$

Avec

$$\begin{aligned}
\hat{a}_h^{(q)} &= a_0 + \tfrac{P^{(q)}}{2} \\
\hat{b}_h^{(q)} &= b_0 + \frac{\sum_{p=1}^{P^{(q)}} \langle (\hat{\mathbf{A}}_{[p,h]}^{(q)})^2 \rangle}{2}
\end{aligned}$$

• **Paramètre de sparsité des vecteurs de loadings $\theta_h^{(q)}$**

$$\boxed{q(\theta_h^{(q)}) = \mathcal{B}eta(\hat{a}_h^{(q)}, \hat{b}_h^{(q)})} \tag{5.29}$$

Avec

$$\begin{aligned}
\hat{a}_h^{(q)} &= \sum_{p=1}^{P^{(q)}} \langle \mathbf{S}_{\mathbf{A},[p,h]}^{(q)} \rangle + a_0 \\
\hat{b}_h^{(q)} &= b_0 - \sum_{p=1}^{P^{(q)}} \langle \mathbf{S}_{\mathbf{A},[p,h]}^{(q)} \rangle + P^{(q)}
\end{aligned}$$

• **Résidus** $\tau_p^{(m,q)}$

$$\boxed{q(\tau_p^{(m,q)}) = \mathcal{G}(\hat{a}_p^{(m,q)}, \hat{b}_p^{(m,q)})} \tag{5.30}$$

Avec

$$\begin{aligned} \hat{a}_p^{(m,q)} &= a_0 + \frac{N^{(m)}}{2} \\ \hat{b}_p^{(m,q)} &= b_0 + \frac{1}{2} \sum_{n=1}^{N^{(m)}} \left\langle \left(\mathbf{X}_{[n,p]}^{(m,q)} - \sum_{h=1}^{H} \mathbf{A}_{[p,h]}^{(q)} \mathbf{T}_{[n,h]}^{(m)} \right)^2 \right\rangle \end{aligned}$$

5.4 Utilisation pratique de MOFA+

5.4.1 Intérêt de la méthode

Identification des biomarqueurs

Comme pour DIABLO et MINT, les biomarqueurs peuvent être repérés grâce au vecteur de loadings. Les variables les plus influentes, pouvant donc être considérées comme des biomarqueurs, sont celles dont la valeur absolue du poids qui leur est associé dans les vecteurs de loadings est la plus élevée. Contrairement à DIABLO et MINT qui sont des méthodes supervisées, les premiers vecteurs de scores ne sont pas toujours ceux qui discriminent le mieux les différents groupes de la matrice d'intérêt $\mathbf{Y}$. Les premiers vecteurs de loadings sont souvent caractérisés par une sparsité moins intense que les suivants. Ceci est dû au fait que les premiers vecteurs capturent des effets qui sont souvent présents dans la plupart des variables.

Lorsque le vecteur des loadings n'est pas sparse, la détermination des biomarqueurs n'est pas aussi objective que dans le cas de DIABLO ou MINT. C'est l'utilisateur qui estime à partir de quel seuil une variable est considérée comme un biomarqueur. Il peut également choisir de ne retenir qu'un certain nombre de premiers biomarqueurs (ceux avec le poids le plus élevé).

Clustering des observations

Le clustering des observations peut être réalisé directement en observant les vecteurs de scores deux à deux ou en effectuant une réduction des dimensions complémentaire, sur la base de l'ensemble des vecteurs de scores calculés par la méthode. Cette deuxième option peut être faite soit via une Analyse en Composantes Principales, soit via une méthode de réduction des dimensions non-linéaires, comme une t-SNE ou UMAP. Ces deux derniers outils sont destinés à des jeux de données où le nombre d'observations et le nombre de vecteurs de scores est élevé, ce qui se prête donc bien aux analyses single-cell en général.

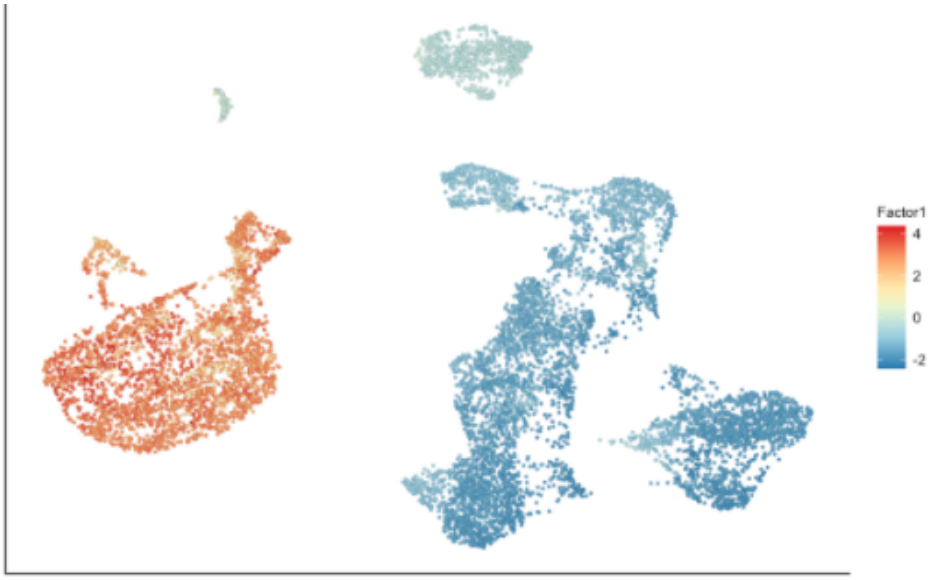


FIGURE 5.11: Illustration d'une réduction des dimensions non-linéaire appliquée sur les vecteurs de scores d'un objet de type MOFA+ [19].

5.4.2 Paramètres du modèle

— **Le nombre d'observations B du minibatch $\mathcal{B}$**

La réduction de la taille totale du jeu de données influence fortement le temps de compilation de l'algorithme de MOFA+ puisque la version stochastique de l'algorithme est basé sur le minibatch. Argelaguet et al. recommandent de garder un nombre d'observation B égal à 10 à 50 % de la taille totale de l'échantillon.

— **Les hyperparamètres τ et κ, liés au paramètre ρ**

Le paramètre ρ utilisé dans la construction des paramètres variationnels globaux ψ de l'algorithme 4 définit l'importance accordée aux informations déjà récoltées dans les itérations précédentes :

$$\psi_{i+1} = \underbrace{(1-\rho_i)\cdot\psi_i}_{\textit{Souvenir}} + \underbrace{\rho_i\cdot\psi_{i+1}^{\mathcal{B}}}_{\textit{Update}}$$

Accorder une valeur proche de 1 signifierait que presque aucune information récoltée dans l'itération précédente (Souvenir) n'est gardée pour construire ψ_{i+1} . Au contraire, allouer une valeur proche de zéro pour ρ_i ne modifierait presque pas la valeur de ψ_{i+1} avec les informations récoltées sur le minibatch $\mathcal{B}$ (Update).

Dans MOFA+, cet hyperparamètre varie au cours du temps. Il est défini de la manière suivante :

$$\rho_i = \frac{\tau}{(1+\kappa\cdot i)^{3/4}}$$

- τ influence positivement la valeur initiale de ρ.

- κ influence la vitesse à laquelle ρ va diminuer au cours des itérations : au plus cette valeur est élevée, au plus la valeur du paramètre diminue rapidement.

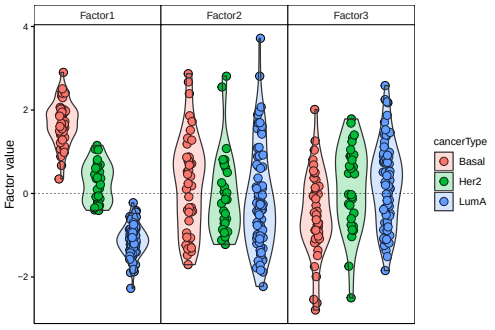
5.4.3 Sorties graphiques

- **Graphiques d'échantillon**

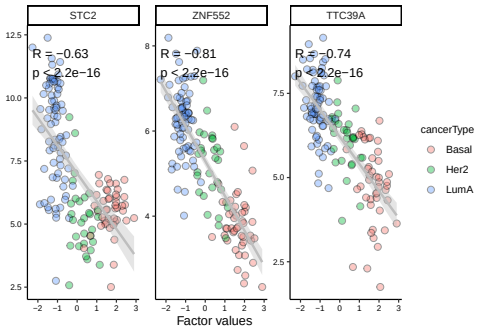
Contrairement à DIABLO et à MINT, les graphiques d'échantillons de MOFA+ ne permettent pas de générer chaque sortie par bloc spécifique. Il est toutefois parfois possible d'encoder cette information comme une métadonnée et de l'utiliser dans le graphique.

1. `plot_factor()`

   Ce graphique projette les observations dans le sous-espace engendré par le vecteur de score calculé. Il permet d'analyser le comportement des observations au sein d'un ou plusieurs vecteurs de scores, selon une variable (continue ou discrète) choisie. Cette variable est typiquement présente dans les métadonnées : dans l'exemple ci-contre, le type de cellule cancéreuse. Cela peut permettre d'analyser le caractère discriminant de MOFA+ selon cette variable. La largeur des violons est proportionnelle à la densité des observations.

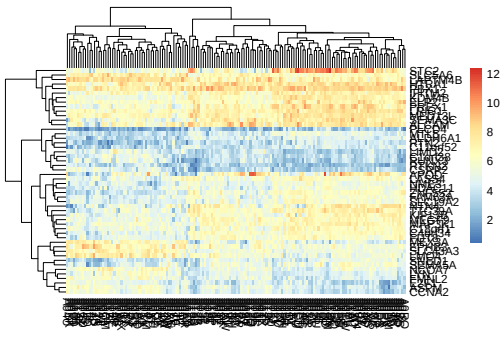


2. `plot_data_scatter()`
   Ce graphique projette les observations dans un sous-espace engendré par la valeur d'un biomarqueur sur l'axe des ordonnées (ci-contre : STC2, ZNF552 et TTC39A) et d'un vecteur des scores sur l'axe des abscisses. Ceci permet d'observer la corrélation entre ces deux variables. Eventuellement, les observations peuvent être colorées selon une troisième variable (continue ou discrète) supplémentaire. En plus de cette information visuelle, ce graphique renseigne le coefficient de corrélation de Pearson ainsi que sa p-valeur associée.

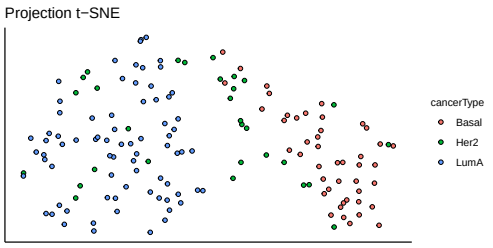


3. `plot_data_heatmap()`

   Lors de la création de cette heatmap, l'utilisateur peut encoder manuellement le nombre de biomarqueurs à analyser, parmi ceux dont le poids (en valeur absolue) est le plus élevé. Les lignes correspondent aux biomarqueurs et les colonnes aux observations. Comme pour les heatmaps des autres méthodes, cela permet de reconnaître des patterns dans les données.

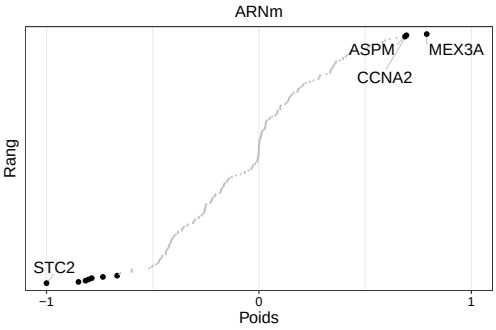


4. `plot_dimred()`
   Cette fonction permet d'exécuter une projection t-SNE (**t**-distributed **S**tochastic **N**eighbor **E**mbedding) ou UMAP (**U**niform **M**anifold **A**pproximation and **P**rojection) en se basant sur les vecteurs des scores. Ces deux méthodes ne sont pas expliquées dans ce travail mais il s'agit d'une réduction non-linéaire de dimensions. Le graphique permet donc de visualiser l'ensemble du sous-espace dans un espace à deux dimensions. Le principe est qu'au plus deux observations sont similaires, au plus les points qui les représentent seront proches dans l'espace réduit.

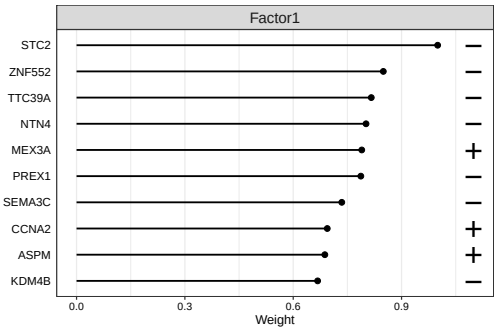


- **Graphiques des variables**

5. `plot_weights()`
   Ce graphique donne une information sur l'allure des coefficients associés aux variables dans le vecteur partiel des loadings d'une modalité spécifique, selon la composante observée. Au plus une variable a une valeur extrême sur l'axe des abscisses, au plus elle joue un rôle important dans la construction du vecteur des scores de la même composante. On peut décider du nombre de variables à mettre en évidence mais certains noms de variables ne sont pas toujours visibles en raison de leur chevauchement.



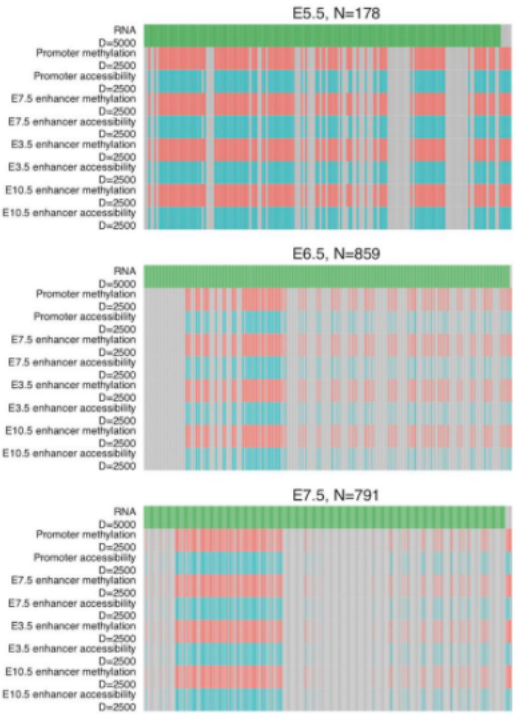
6. `plot_top_weights()`
   Ce graphique permet une autre visualisation du poids des variables dans la construction des vecteurs partiels des scores, selon une modalité et une composante choisie. Il a l'avantage d'identifier plus facilement le nom des variables qui ont un poids important (en valeur absolue) sur le vecteur de loadings. Le nombre de variables à montrer peut être défini par l'utilisateur. Le signe du coefficient associé à cette variable est renseigné sur la marge droite du graphique



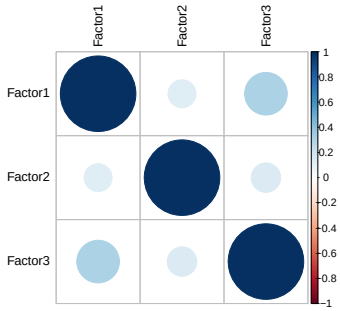
- **Autres graphiques**

7. `plot_data_overview()`

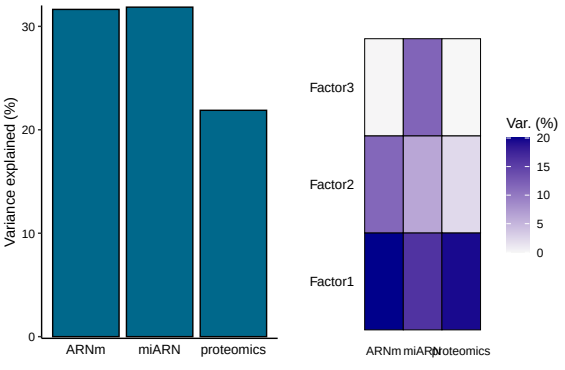
Ce graphique permet de visualiser la présence de données manquantes ainsi que l'allure générale des données, des variables, des groupes et des modalités originales. Il s'agit d'un graphique qui est utilisé en début d'analyse exploratoire et qui est particulièrement utile lors d'une intégration dans les deux sens. L'image illustrée sur la droite est tirée de [5] et n'est pas conforme à la représentation faite tout au long de ce travail : les observations sont en colonne, les variables en lignes, les blocs horizontaux sont empilés verticalement dans chaque sous-graphique, etc. Sur cette illustration, chaque sous-graphique correspond à une étude différente ($M = 3$), chacune avec un nombre d'observations différent (178, 859 et 791). Au sein de chaque étude, 9 blocs horizontaux ($Q = 9$) ont été utilisés, avec un total de 25.000 variables. Les parties grisées du graphique représentent les données manquantes. On peut voir que certaines observations ont un profil très complet dans les 9 modalités et d'autres n'ont qu'une seule modalité encodée.



8. `plot_factor_cor()`
   Ce graphique permet d'analyser la corrélation entre les vecteurs des scores globaux produits. Bien qu'il ne s'agisse pas d'une obligation, on peut cependant s'attendre à ce que la corrélation entre les facteurs ne soit pas très élevée. La fonction `plot_factor_cor()` produit donc un graphique qui est utilisé pour vérifier qu'un modèle est bien construit, mais son interprétation en tant que telle n'a pas beaucoup d'intérêt. Chaque élément de la matrice représente le coefficient de corrélation (Pearson, Spearman ou Kendall) entre les facteurs, représentés en lignes et en colonnes.



9. `plot_variance_explained()`
   Cette fonction produit une liste avec deux graphiques différents, chacun donnant une information sur la variance expliquée par tous les vecteurs de scores. Le graphique de gauche donne la valeur de la variance expliquée par l'ensemble des vecteurs de scores pour un bloc entier. Le graphique de droite détaille davantage cette information : il explique comment cette variance est répartie au sein de chaque bloc et de chaque vecteur de score.



Chapitre 6

Etudes de cas

6.1 Données du TCGA

Les données ont été fournies par le package *mixOmics* et sont décrites dans le chapitre 2.5.1 de ce travail. L'analyse complète, reprenant l'intégralité des graphiques, est disponible à l'adresse Git Hub https ://github.com/LionelPanneel/multi-omics. Seuls les graphiques le plus pertinents sont repris dans ce travail.

6.1.1 Visualisation des données via une ACP

La première partie de ce chapitre modélise une Analyse en Composantes Principales sur les données centrées et réduites par bloc puis concaténées, sans plus tenir compte de leur structure en blocs. La projection des données selon les deux premières composantes de l'ACP peut être observée sur la Figure 6.1. On peut constater que les cancers de type LumA et Basal sont très différents l'un de l'autre selon ces deux composantes. Par contre, la classification est loin d'être parfaite car les observations atteintes du cancer de type Her2 se confondent fortement avec ceux des deux autres types de cancer. Le pourcentage cumulé de variance expliquée par les deux premières composantes de cette ACP est de 24%.

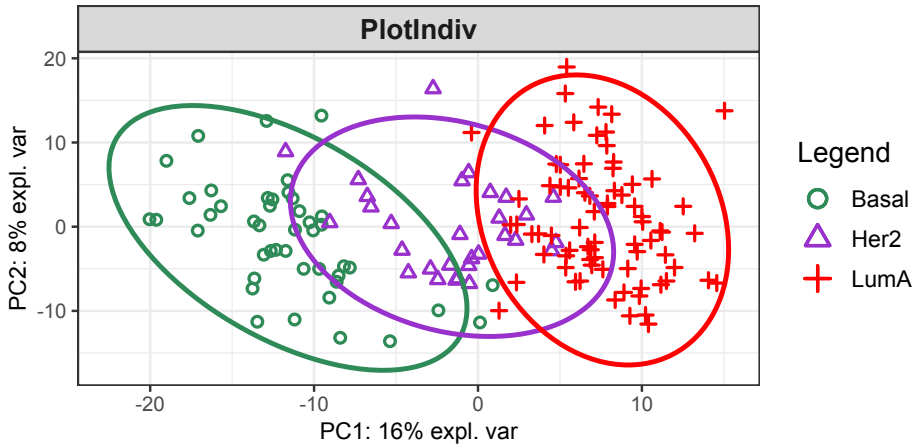


FIGURE 6.1: Projection des données dans les deux premières composantes d'une Analyse en Composante Principales

6.1.2 Analyse des données du TCGA avec DIABLO

6.1.2.1 Paramètres du modèle

Les paramètres de la méthode sont les suivants :

• Sans connaissance biologique particulière sur les liens entre les blocs étudiés, nous avons fixé des valeurs de 0.1 pour les éléments interblocs $c_{[i,j]}$ de la matrice de design $\mathbf{C}$ $(i \neq j, i \neq Q, j \neq Q)$, comme cela est proposé dans la vignette de DIABLO, qui utilise les mêmes données [17]. La matrice de design retenue est donc la suivante :

$$\begin{array}{c} \\ ARNm \\ miARN \\ prot. \\ Y \end{array} \begin{array}{c} \begin{array}{cccc} ARNm & miARN & prot. & Y \end{array} \\ \begin{pmatrix} 0 & 0.1 & 0.1 & 1 \\ 0.1 & 0 & 0.1 & 1 \\ 0.1 & 0.1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

La matrice de design $\mathbf{C}$ est donc définie de sorte que la méthode DIABLO tende à maximiser les corrélations entre les vecteurs de scores des blocs $\mathbf{X}^{(q)}$ et de la matrice de la variable d'intérêt $\mathbf{Y}$ ainsi que, dans une moindre mesure, les corrélations entre les vecteurs des scores des blocs $\mathbf{X}^{(q)}$ entre eux.

• Lê Cao et al. proposent de garder un nombre de composantes latentes égales à $G-1$ [15], où G représente le nombre de groupe de phénotypes différents à découvrir. En effet, ce nombre semble pertinent lorsque nous observons la Figure 6.2, qui illustre le taux d'erreur en fonction du nombre de composantes du modèle. On voit sur cette figure une baisse importante du taux d'erreur lorsqu'une deuxième composante est ajoutée au modèle. Toutefois, nous décidons de rajouter une troisième composante au modèle ($H = 3$). Ce choix est motivé par le fait que le BER (voir 2.6.2) diminue encore sensiblement lorsqu'on s'intéresse à la distance de Mahalanobis (voir 2.6.4). Il est à noter que nous n'avons pas trouvé d'information sur le

calcul des intervalles de confiance dessinés sur ce graphique, mais on peut supposer qu'il s'agit d'un écart-type calculé au cours de la K-fold cross-validation.

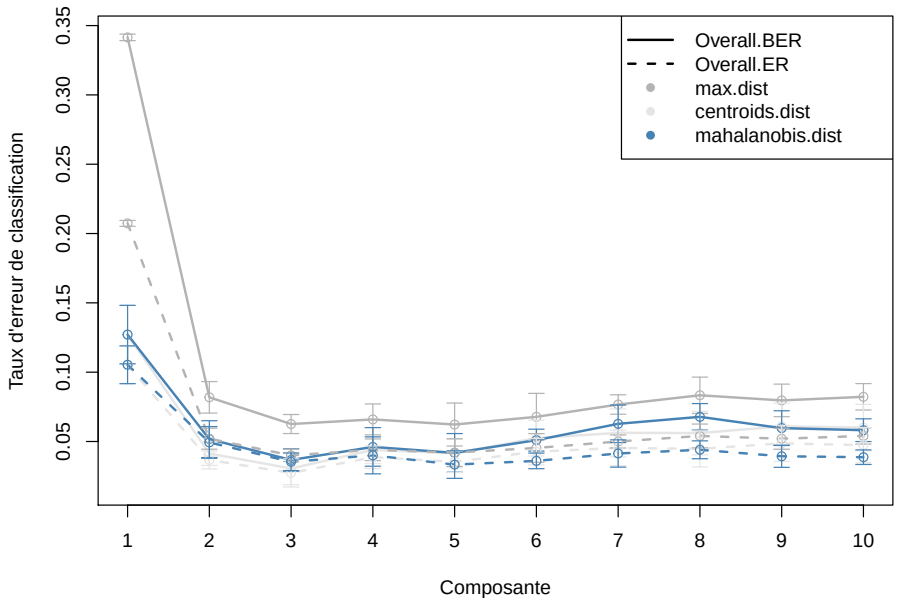


FIGURE 6.2: Analyse du taux d'erreur de classification. Graphique illustrant les résultats obtenus via la fonction `perf()` appliquée sur un objet de type DIABLO

• Le nombre optimal de variables avec un poids non nul associé au vecteur de loadings au sein de chaque bloc est recherché en utilisant la fonction `tune.block.splsda()` du package, qui exécute un K-fold cross validation (K = 10) et sélectionne le modèle qui minimise la distance choisie (ici : Mahalanobis). Suite à cette analyse, nous gardons le nombre suivant de variables dans chaque bloc, en fonction de la composante latente (la colonne $P^{(q)}$ rappelle le nombre de variables initialement présentes dans chaque bloc) :

	$\mathbf{a}_1^{(q)}$	$\mathbf{a}_2^{(q)}$	$\mathbf{a}_3^{(q)}$	Tot.	$P^{(q)}$	%
ARNm	12	15	3	**30**	200	15%
miARN	6	3	9	**18**	184	9.8%
protéomique	9	3	3	**15**	142	10.6%
TOTAL :	**27**	**21**	**15**	**63**	**526**	

Le pourcentage de la variance expliquée par les vecteurs de scores partiels $\mathbf{t}_h^{(q)}$ dans chaque bloc est donnée dans le tableau ci-dessous :

	$\mathbf{t}_1^{(q)}$	$\mathbf{t}_2^{(q)}$	$\mathbf{t}_3^{(q)}$
ARNm	17.39 %	4.01 %	8.89 %
miARN	19.78 %	2.64 %	4.29 %
protéomique	10.56 %	3.50 %	3.91 %
Y	**55.86 %**	**44.48 %**	**35.33 %**

Comme on peut s'en apercevoir, le pourcentage de la variance expliquée dans chaque bloc de la matrice $\mathbf{X}^{(q)}$ par chaque vecteur partiel de scores ne diminue pas toujours. Ceci est dû au fait que la matrice de design impose une corrélation maximale entre les vecteurs partiels de scores de chaque bloc et ceux de la variable d'intérêt $\mathbf{Y}$ ($C_{[i,Q]}$ et $C_{[Q,j]} = 1$, pour $i, j \neq Q$) mais que la maximisation des vecteurs partiels de scores des blocs entre eux est secondaire ($C_{[i,j]} = 0.1$ pour $i \neq j$ et $i, j \neq Q$).

6.1.2.2 Analyse des observations

Une analyse des vecteurs de scores partiels pour les composantes 1 et 2 est illustrée sur les deux Figures suivantes. On peut voir que les trois premiers vecteurs de scores partiels $\mathbf{t}_1^{(1)}$, $\mathbf{t}_1^{(2)}$et $\mathbf{t}_1^{(3)}$ (Figure 6.3) utilisés seuls semblent classifier les données, mais de manière imparfaite. Par contre, les trois deuxièmes vecteurs de scores partiels (Figure 6.4) semblent particulièrement incapables de différencier les observations du type Basal et LumA mais permettent de différencier ces deux-ci du type Her2, dans les trois blocs.

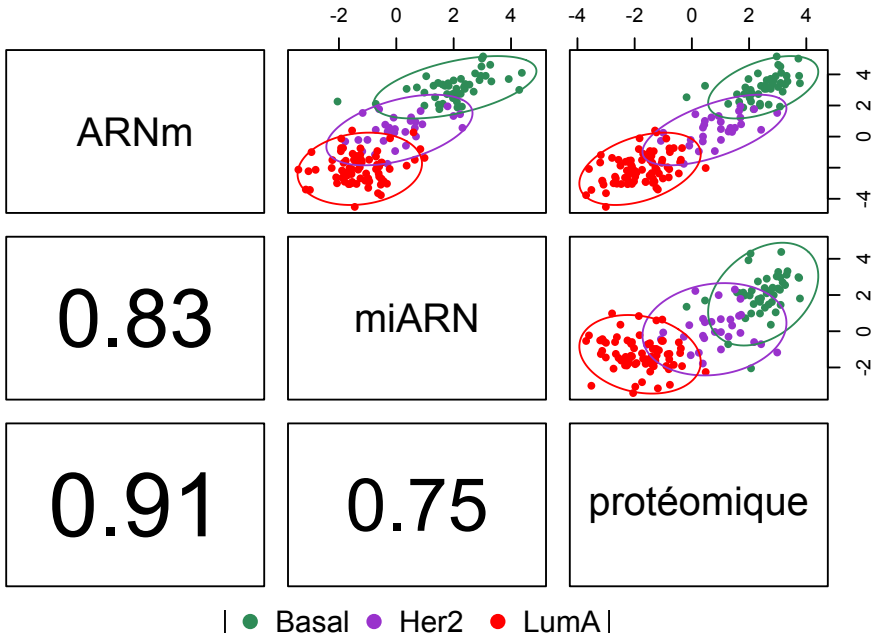


FIGURE 6.3: Analyse des premiers vecteurs de scores partiels. Graphique obtenu via la fonction `plotDiablo()` appliquée sur les données du TCGA

Le graphique des trois troisièmes vecteurs de scores partiels est visible sur la Figure 6.5 mais ne semble pas apporter de pouvoir de discrimination supplémentaire. Par contre, les corrélations entre les vecteurs de scores des différents blocs de cette troisième composante (matrice triangulaire inférieure) est plus important que dans la deuxième composante. On peut émettre l'hypothèse qu'une fois que l'algorithme ne sait plus expliquer la variable $\mathbf{Y}$, il maximise la corrélation entre les vecteurs des scores des blocs, comme cela est spécifié dans la matrice de design $\mathbf{C}$.

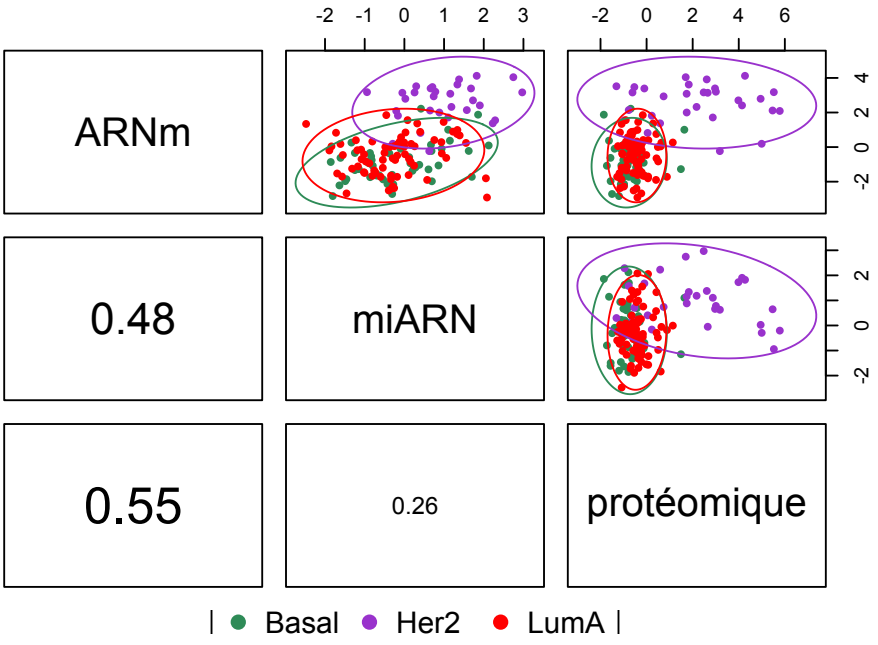


FIGURE 6.4: Analyse des seconds vecteurs de scores partiels.

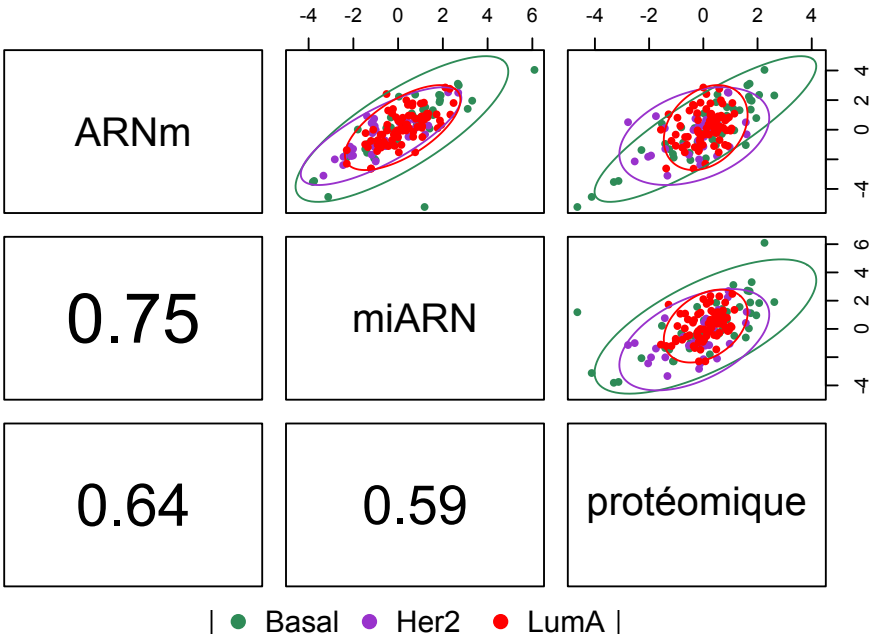


FIGURE 6.5: Analyse des troisièmes vecteurs de scores partiels.

Sur la Figure 6.6, on peut voir que les vecteurs de scores partiels de la première et de la deuxième composante, lorsqu'ils sont utilisés ensemble, permettent de discriminer presque parfaitement les données dans le bloc de l'ARN messager et assez bien les données de protéomique. On peut à nouveau remarquer que la deuxième composante semble surtout utile à discriminer le cancer de type Her2 des deux autres. Les données de micro ARN ne

semblent par contre pas très bien discriminées dans ces deux premières composantes.

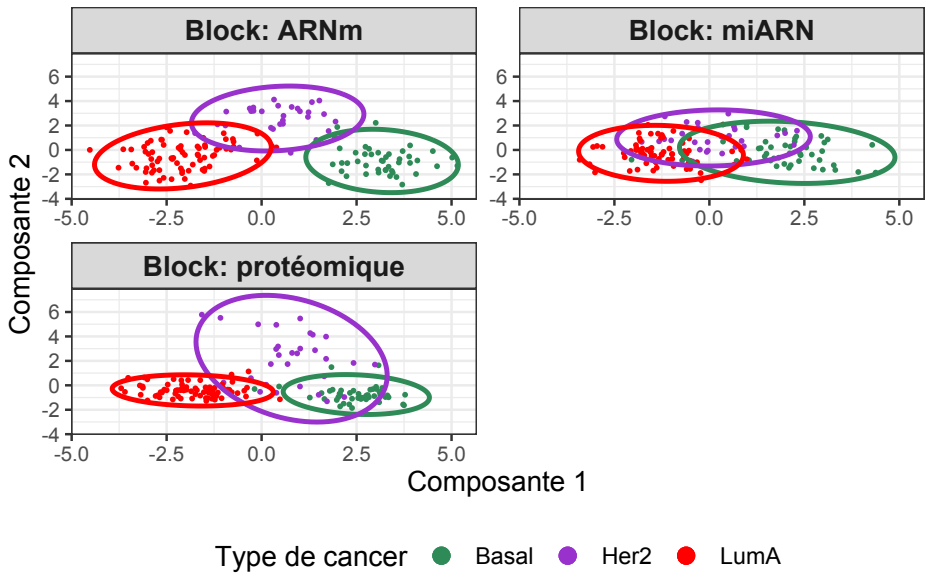


FIGURE 6.6: Analyse des vecteurs de scores partiels dans la première et la seconde composante. Graphique obtenu avec la fonction `plotIndiv()` appliquée sur les données du TCGA

Lorsque l'on intègre les différentes modalités et que l'on observe les vecteurs de scores partiels via la fonction `plotArrow()`, on peut voir que les observations semblent bien classifiées selon le groupe de cancer auquel elles appartiennent, comme illustré sur la Figure 6.7.

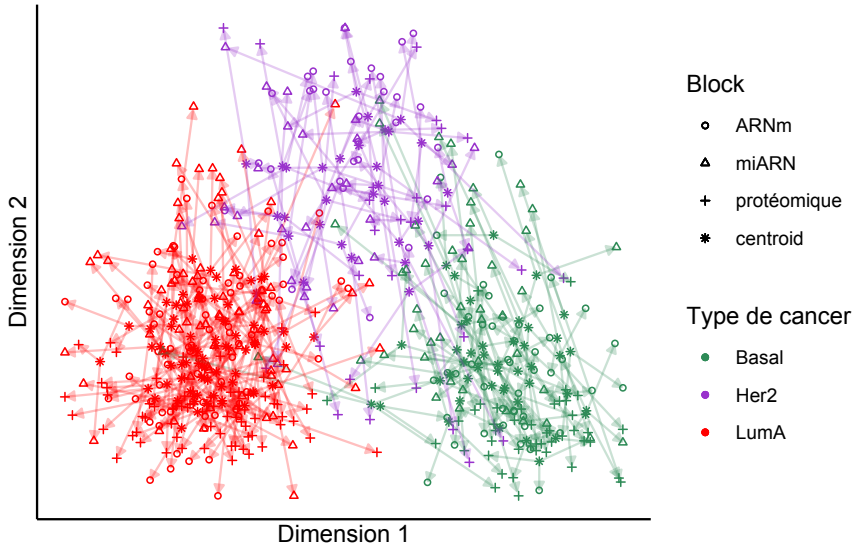


FIGURE 6.7: Analyse des vecteurs de scores partiels des composantes 1 et 2, toutes modalités intégrées. Graphique obtenu avec la fonction `plotArrow()` appliquée sur les données du TCGA

Sur la Figure 6.8, on peut constater que le dendrogramme des observations (sur la gauche du graphique) les classifie parfaitement, à l'exception de sept (deux observations de type

Basal et cinq observations du type LumA regroupées avec les observations de type Her2).

On remarque aussi que les biomarqueurs peuvent être classifiés en trois groupes, qui représentent d'ailleurs les types de cancer :

• Les sept premiers biomarqueurs en partant de la gauche semblent avoir un comportement similaire et prennent une valeur élevée pour les observations du type de cancer Her2.

• Les 18 biomarqueurs suivants semblent attribuer une valeur importante aux observations du type LumA.

• Les biomarqueurs restants semblent finalement attribuer une valeur plus importante aux observations du type Basal.

L'utilisation de ce graphique ne nous a cependant pas semblé évidente : l'appréciation des patterns et des "groupes" de biomarqueurs nous semble plutôt subjective. Par ailleurs, malgré le nombre de biomarqueurs limités (63), leur identification, sur le bas du graphique, n'est pas évidente. De même, à moins d'avoir un graphique de très grande dimension, il n'est pas possible d'afficher le nom des observations en face de chaque ligne qui lui correspond. Finalement, on peut mentionner que la classification des biomarqueurs que nous venons de proposer ne correspond pas aux trois premiers groupes désignés par le dendrogramme.

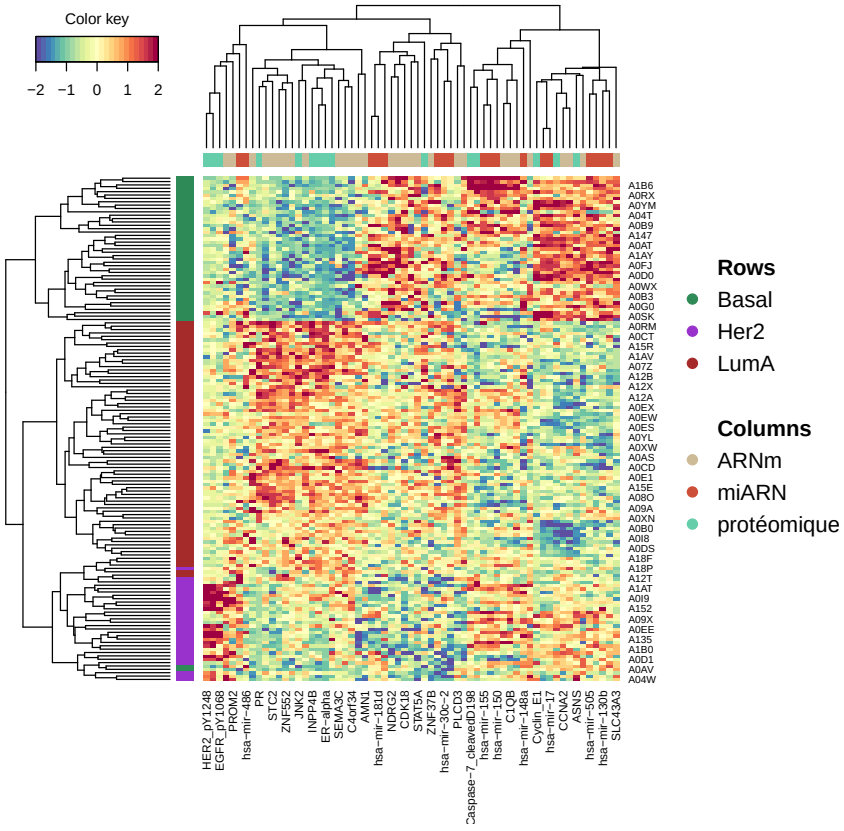


FIGURE 6.8: Analyse des patterns des biomarqueurs et des observations. Graphique obtenu via la fonction `cimDiablo()` appliquée sur les données du TCGA

6.1.2.3 Analyse des biomarqueurs estimés

L'analyse des biomarqueurs est réalisée ci-dessous via les fonctions `plotVar()`, `circosPlot()`, `network()` et `plotLoadings()` du package *mixOmics* et mène aux deux hypothèses principales suivantes :

1. Le premier vecteur des scores semble servir principalement à la discrimination des données atteintes du cancer du sein de type Basal.

2. Le second vecteur des scores semble discriminer les patients atteints du cancer du sein de type Her2.

Sur la Figure 6.9, seuls les biomarqueurs ayant un coefficient de corrélation suppérieur à 0.7 avec leur vecteur de scores partiel correspondant ont été représentés. On peut constater, sur ce graphique :

• qu'un groupe de six biomarqueurs de micro ARN semble particulièrement bien représenté sur la première composante.

• que 13 des 19 variables de protéomique et d'ARN messager les mieux représentées sur la première composante ont un coefficient de corrélation négatif.

• qu'il n'y a que 7 biomarqueurs fortement corrélées (> 0.7) avec leur second vecteur de scores partiel correspondant (contre 25 biomarqueurs pour la première composante).

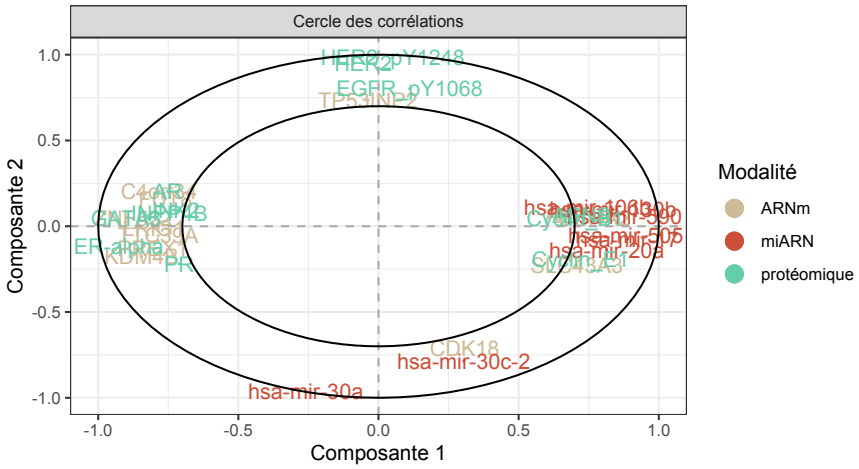


FIGURE 6.9: Analyse du cercle des corrélations selon les composantes 1 et 2. Graphique obtenu via la fonction `plotVar()` appliquée aux données du TCGA

Sur la Figure 6.10, on peut voir que les biomarqueurs de micro ARN semblent exprimer davantage le cancer du sein de type Basal. Or, nous avions vu dans la Figure 6.9 que ces données étaient particulièrement bien représentées sur le premier vecteur des scores avec un coefficient de corrélation positif.

On peut également s'apercevoir que les biomarqueurs HER2 et HER2_pY1248 d'une part et hsa-mir-30a (miARN) d'autre part sont corrélés négativement. Ces biomarqueurs ont un niveau d'expression du cancer de type Her2 très opposé. Il s'agit également des biomarqueurs qui sont représentés de manière opposée sur les extrémités de l'axe des ordonnées de la Figure 6.9.

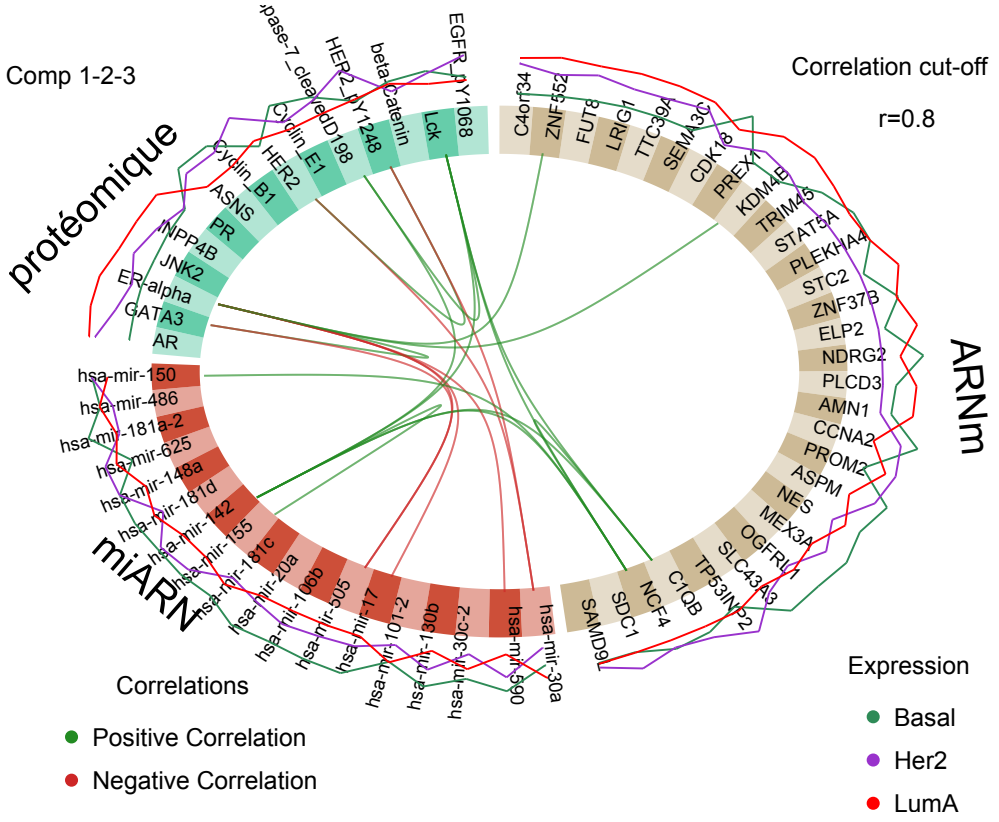


FIGURE 6.10: Analyse des biomarqueurs, des relations entre eux et du niveau d'expression de la variable d'intérêt en fonction des biomarqueurs, selon les trois composantes, avec un cutoff de 0.8. Graphique obtenu via la fonction `circosPlot()` appliquée sur les données du TCGA

Sur la Figure 6.11, on constate trois clusters de biomarqueurs, nous permettant d'émettre les hypothèses suivantes :

• Le cluster en haut à droite comprend des biomarqueurs où l'expression du cancer du sein de type Her2 est élevée.

• Le cluster en bas est celui qui comprend des biomarqueurs où l'expression du cancer du sein de type LumA est plus importante.

• Le cluster en-haut à gauche réunit les biomarqueurs qui expriment fortement à la fois les observations atteintes du cancer du sein de type Her2 et celles atteintes du cancer du sein de type Basal.

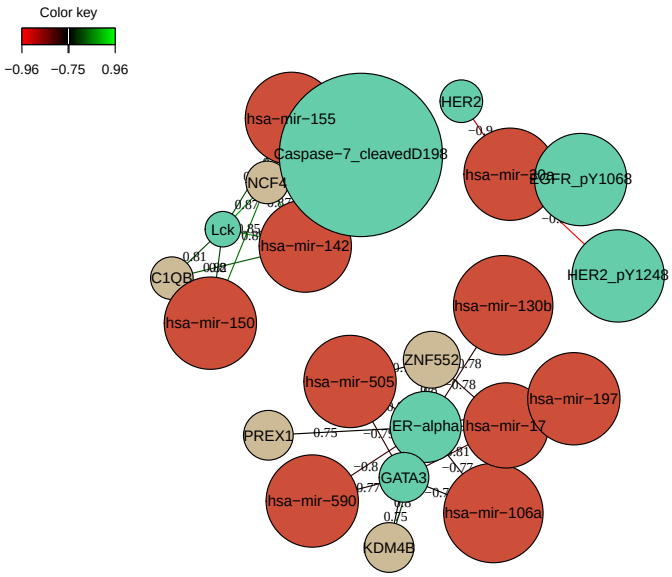


FIGURE 6.11: Analyse des clusters des biomarqueurs ayant un coefficient de corrélation de plus de 0.75 (en valeur absolue) avec des biomarqueurs d'autres blocs. Graphique obtenu via la fonction `network()` appliquée sur les données du TCGA.

En comparant les Figures 6.12 et 6.9, on peut interpréter plus précisément les vecteurs de scores partiels des deux premières composantes :

• Si on observe l'expression majoritaire (Figure 6.12) des biomarqueurs qui ont une valeur positive élevée sur l'axe des abscisses du cercle de corrélation (Figure 6.9) , elle est du type Basal.

• Si on observe l'expression majoritaire (Figure 6.12) des biomarqueurs qui ont une valeur négative élevée sur l'axe des abscisses du cercle de corrélation (Figure 6.9) , elle est du type LumA.

• Finalement, les biomarqueurs fortement corrélés positivement avec le deuxième vecteur des scores (Figure 6.9) semblent avoir une expression du cancer Her2 plus élevée (Figure 6.12).

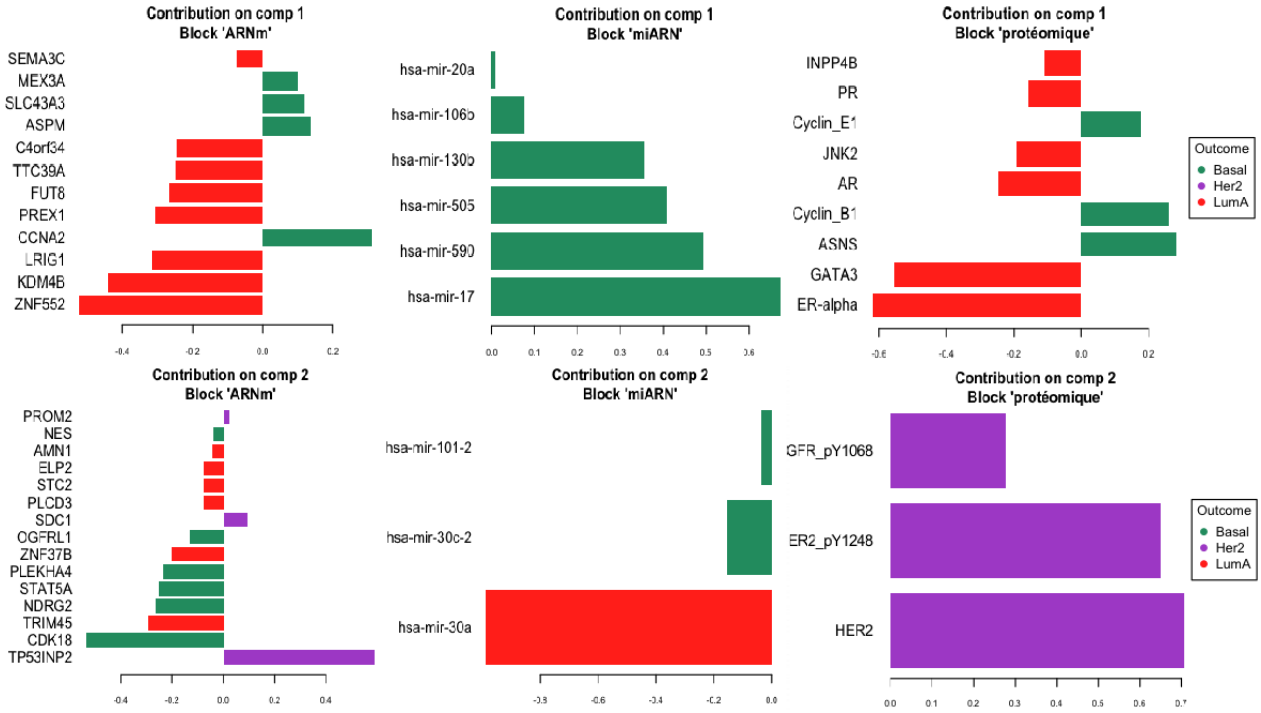


FIGURE 6.12: Analyse des poids associés aux variables utiles dans la construction des vecteurs de scores partiels de la première (haut) et de la deuxième (bas) composante. Graphiques obtenus via la fonction `plotLoadings()` appliquée sur les données du TCGA

6.1.2.4 Analyse de la performance du modèle

L'analyse de la performance du modèle peut être faite de plusieurs manières. La Figure 6.13 permet d'analyser la performance de classification du modèle avec une, deux ou trois composantes sur le test set, selon une modalité choisie. Il faut rappeler que ce jeu de données ne comporte que des modalités d'ARNm et de miARN.

Comme cela a déjà été mentionné plus haut, on peut voir que la première composante, utilisée seule, est particulièrement incapable de classifier correctement les observations de type Her2, et cela dans le bloc de ARNm (AUC = 0.64) ou de miARN (AUX = 0.55) . Par contre, elle permet une très bonne classification des observations de type Basal et LumA (AUC $>$ 0.91 dans les deux blocs).

On peut constater que la seconde composante permet d'améliorer très nettement la classification des observations de type Her2, et cela dans les trois modalités. En effet, l'évolution de l'aire sous la courbe dans les trois blocs pour les observations du cancer du sein de type Her2 est repris dans le tableau ci-dessous. On ne remarque pas d'amélioration de l'AUC avec l'ajout d'une troisième composante dans le test set.

	ARNm	miARN
Modèle avec une composante	0.64	0.55
Modèle avec deux composantes	0.96	0.85
Modèle avec trois composantes	0.96	0.84

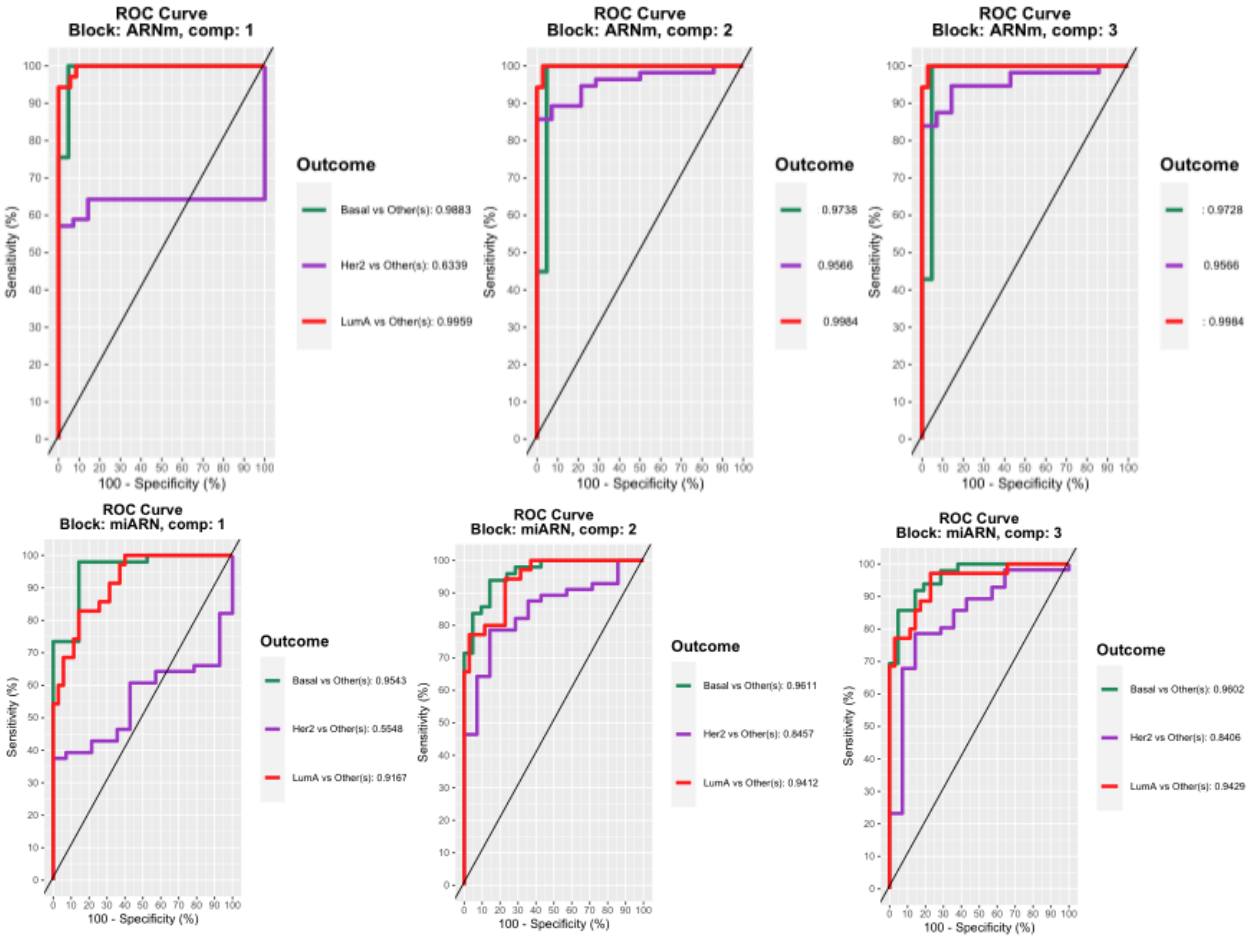


FIGURE 6.13: Analyse de la performance du modèle avec une (gauche), deux (milieu) ou trois composantes (droite), selon chaque modalité (haut : ARNm. Bas : miARN). Graphique obtenu via la fonction `auroc()` appliquée sur les données du TCGA

Une autre manière de constater la performance du modèle est de regarder la matrice de confusion, donnée ci-dessous. La prédiction des variables a été faite en tenant compte des distances de Mahalanobis et selon un vote pondéré (voir 3.4.1).

Le test set comporte 70 nouvelles données et est uniquement composé des blocs miARN et ARNm, le bloc de protéomique étant manquant (comme illustré dans la Figure 2.9). On peut voir que le modèle classifie presque parfaitement les données avec trois composantes. Seules 2 observations du cancer du sein *Her2* (2.9% des données) ont été mal classifiées.

		Classe prédite		
		Basal	Her2	LumA
Classe	Basal	20	1	0
	Her2	0	14	0
	LumA	0	1	34

6.1.3 Analyse des données du TCGA avec MOFA+

6.1.3.1 Paramètres du modèle

Les hyperparamètres du modèles sont les suivants :

• Après plusieurs essais, le paramètre de pas de l'algorithme de stochastic gradient ascent, ρ, est défini selon les paramètres $\tau = 1$ et $\kappa = 0.5$ dans l'équation :

$$\rho_i = \frac{\tau}{(1 + \kappa \cdot i)^{3/4}}$$

En effet, ces paramètres permettent de définir une valeur de ρ qui est un bon compromis entre la vitesse de convergence de l'algorithme et la stabilité de l'ELBO. La Figure 6.14 illustre la vitesse de convergence de l'ELBO pour diverses valeurs de ces deux paramètres.

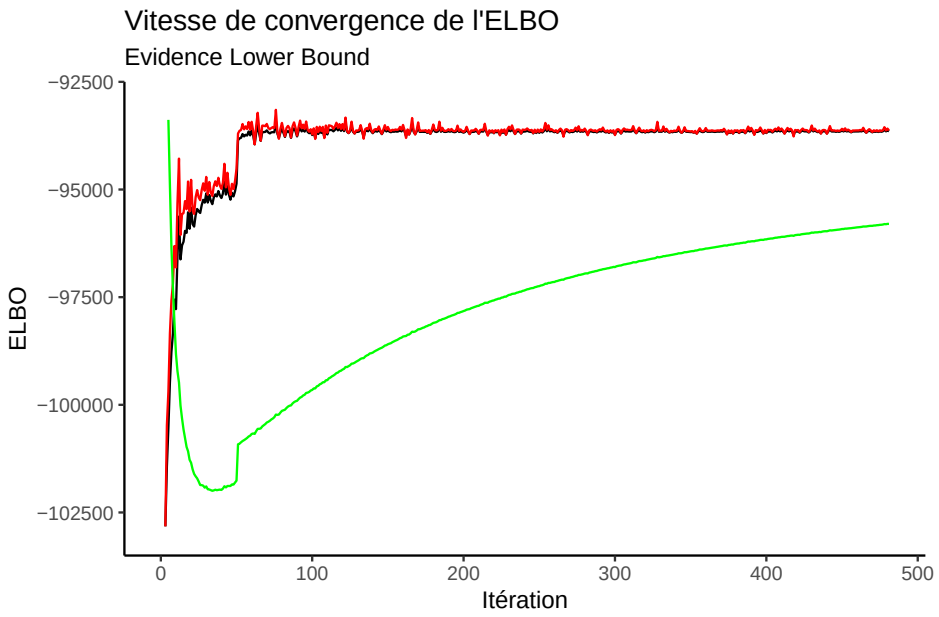


FIGURE 6.14: Convergence de l'ELBO selon divers paramètres pour τ et κ. En noir : $\tau = 1$ et $\kappa = 0.5$. En rouge : $\tau = 1$ et $\kappa = 0.25$. En vert : $\tau = 0.25$ et $\kappa = 1$

• Comme proposé dans [5], la taille du minibatch, sur lequel l'algorithme stochastique est calculé, est fixée à la moitié de la taille de l'échantillon total.

6.1.3.2 Vérification du modèle

Le modèle proposé par MOFA+ trouve des vecteurs de scores qui maximisent la variance des données, similairement à une PCA, mais en tenant compte de la structure des blocs. Les vecteurs de scores calculés sont supposés être peu corrélés entre eux. La Figure 6.15 illustre la corrélation entre les cinq premiers vecteurs de scores. On peut voir que les facteurs sont généralement peu corrélés entre eux ($r_{max} = r_{[1,3]} \simeq 0.21$), ce qui laisse penser qu'il n'y a pas

de gros problème dans le modèle. La raison pour laquelle seuls les cinq premiers vecteurs de scores ont été gardés dans le modèle final sera expliquée au point suivant.

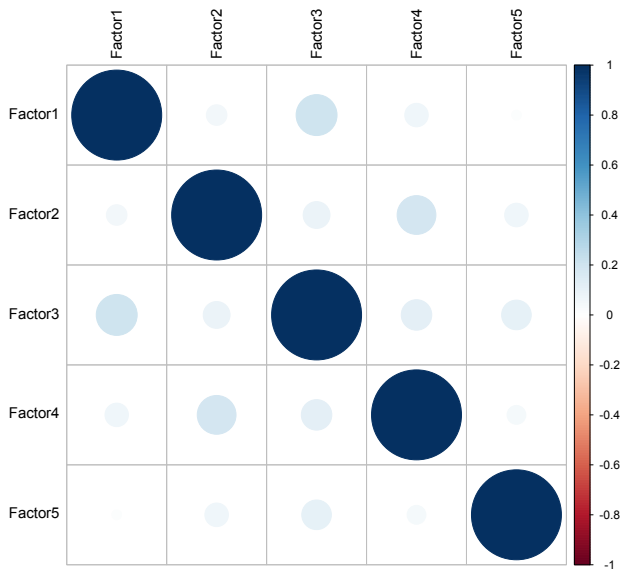


FIGURE 6.15: Analyse de la corrélation des cinq facteurs calculés par MOFA+. Graphique obtenu via la fonction `plot_factor()` appliquée sur les données du TCGA

6.1.3.3 Analyse de la variance

Sur la Figure 6.16, on peut s'apercevoir que le premier vecteur des scores est celui qui capture le plus de variabilité des données, et cela au sein de chaque groupe. A lui seul, il capture près de 20% de la variabilité des données des blocs d'ARNm et de protéomique. On constate également que la variance capturée par les vecteurs de scores supplémentaires tend à diminuer au fur et à mesure et que le cinquième facteur capture moins de 4% de variabilité des données dans chaque bloc (ARNm : 3.67% - miARN : 2.69% - protéomique : 1.15%). Etant donné le peu de variabilité des données capturées au-delà du cinquième facteur, le modèle final ne comportera que les cinq premiers vecteurs de scores.

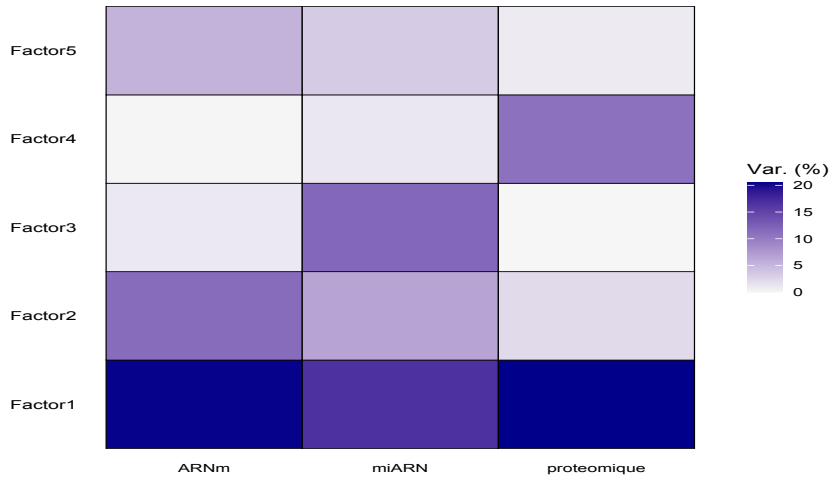


FIGURE 6.16: Analyse de la variance expliquée par chaque vecteur de scores au sein de chaque modalité. Graphique obtenu via la fonction `plot_variance_explained()` sur les données du TCGA

La variance totale expliquée par le modèle dans chaque bloc est représentée par la Figure 6.17. On peut constater que la réduction d'un espace de 526 variables en un sous-espace à cinq facteurs permet de capturer plus d'un tiers de la variation des données au sein de chaque bloc.

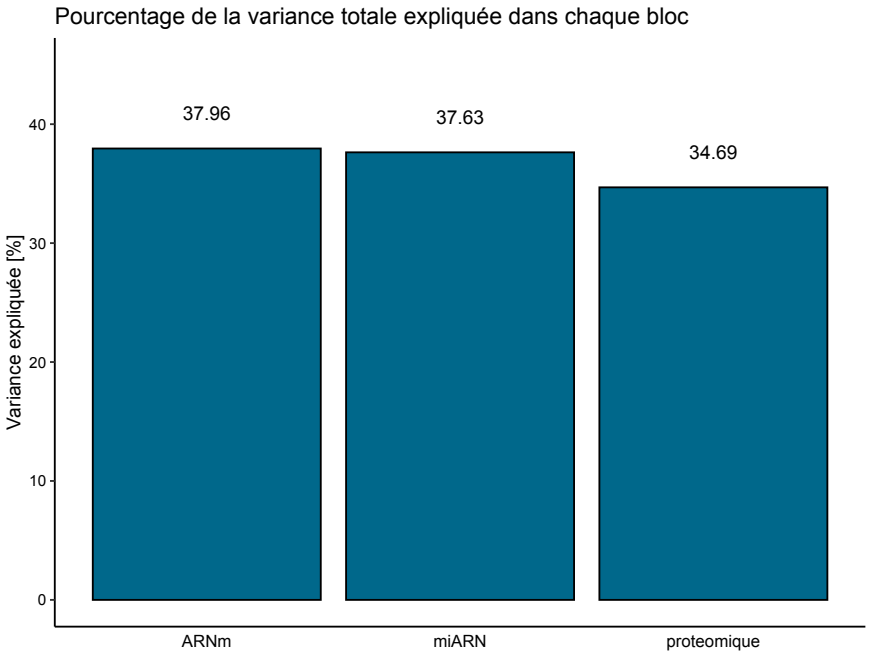


FIGURE 6.17: Analyse de la variance des données expliquée par le modèle final (5 composantes), par modalité. Graphique obtenu via la fonction `plot_variance_explained()` sur les données du TCGA

6.1.3.4 Analyse des observations

Sur la Figure 6.18 on peut constater que, si les facteurs permettent tous de capturer de la variabilité des données, c'est bien le premier facteur qui semble discriminer le mieux les différents types de cancer du sein, particulièrement le cancer de type LumA. Il est important de rappeler à ce stade que MOFA+ est une méthode non-supervisée ; la couleur des points n'a été ajoutée sur le graphique que pour analyser la répartition des groupes de la variable d'intérêt au sein de chaque facteur. Etant donné ce constat, les Figures suivantes se concentreront principalement sur l'analyse du premier vecteur de scores.

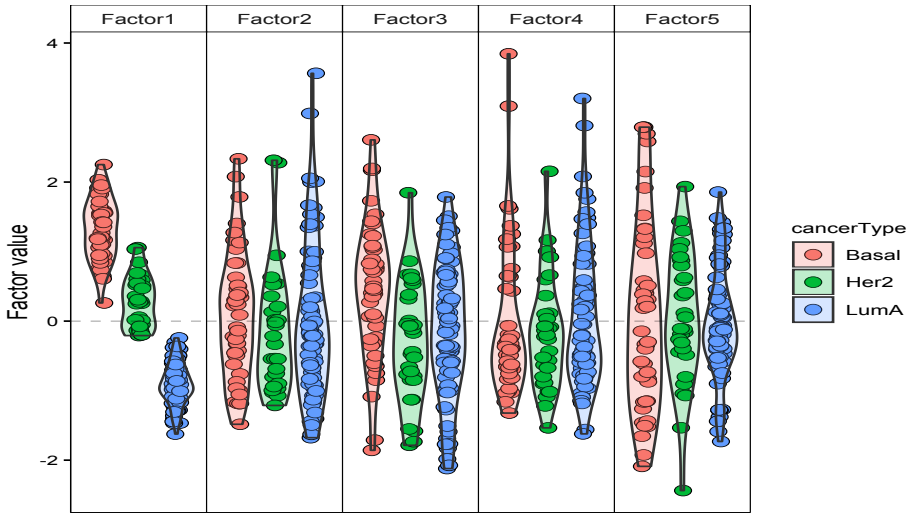


FIGURE 6.18: Projection des vecteurs de scores selon chaque composante. Graphique obtenu via la fonction `plot_factors()` appliquée sur les données du TCGA

Une autre manière de voir ceci est illustré sur la Figure 6.19. Les facteurs 2 à 5 ne semblent pas jouer un rôle quelconque dans la discrimination du type de cancer du sein.

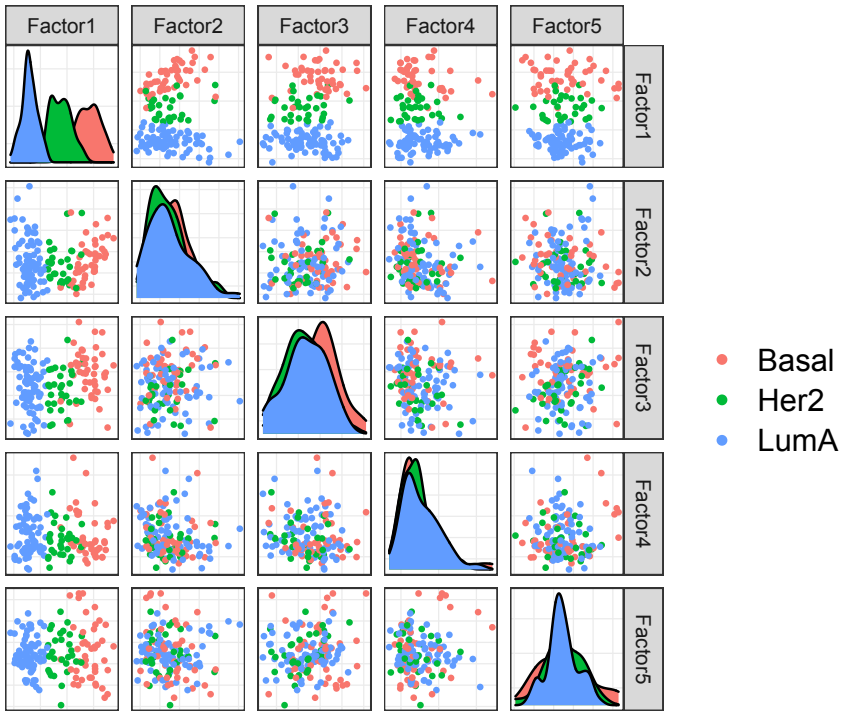


FIGURE 6.19: Autre visualisation de la projection des vecteurs de scores selon chaque composante. Graphique obtenu via la fonction `plot_factors()` appliquée sur les données du TCGA

La Figure 6.20 représente trois heatmaps, chacune montrant les dix biomarqueurs (en ligne) avec les poids les plus élevés dans la première composante au sein de chaque bloc :

• La heatmap en haut à gauche représente le bloc de ARN messager. On peut constater un pattern qui semble séparer un groupe d'observations (en colonne) mais qui semble plus confus pour la deuxième moitié des observations à droite. En comparant le nom des biomarqueurs de cette heatmap avec ceux de la Figure 6.10, on peut remarquer que certains d'entre eux étaient déjà détectés via la méthode DIABLO.

• La heatmap en haut à droite est celle relative aux micro ARN. Le pattern représenté sur cette heatmap est très confus, ne permettant pas une interprétation sûre pour ce graphique.

• Finalement, la heatmap du bas représente les données de protéomique. Le pattern est assez clair et semble distinguer deux groupes d'individus et deux groupes de biomarqueurs.

On peut une fois de plus s'interroger sur l'utilité de ce type de graphique car la détection de patterns reste subjective. Par ailleurs, étant donné que MOFA+ est une méthode non-supervisée, aucune indication relative au type de cancer n'est disponible en marge des observations. Finalement, le graphique affiche les noms des observations, mais ceux-ci sont illisibles, à moins d'avoir un graphique de très grande taille. Qui plus est, MOFA+ est spécifiquement destinée aux analyses single cell, lesquelles comptent parfois plusieurs millions d'observations.

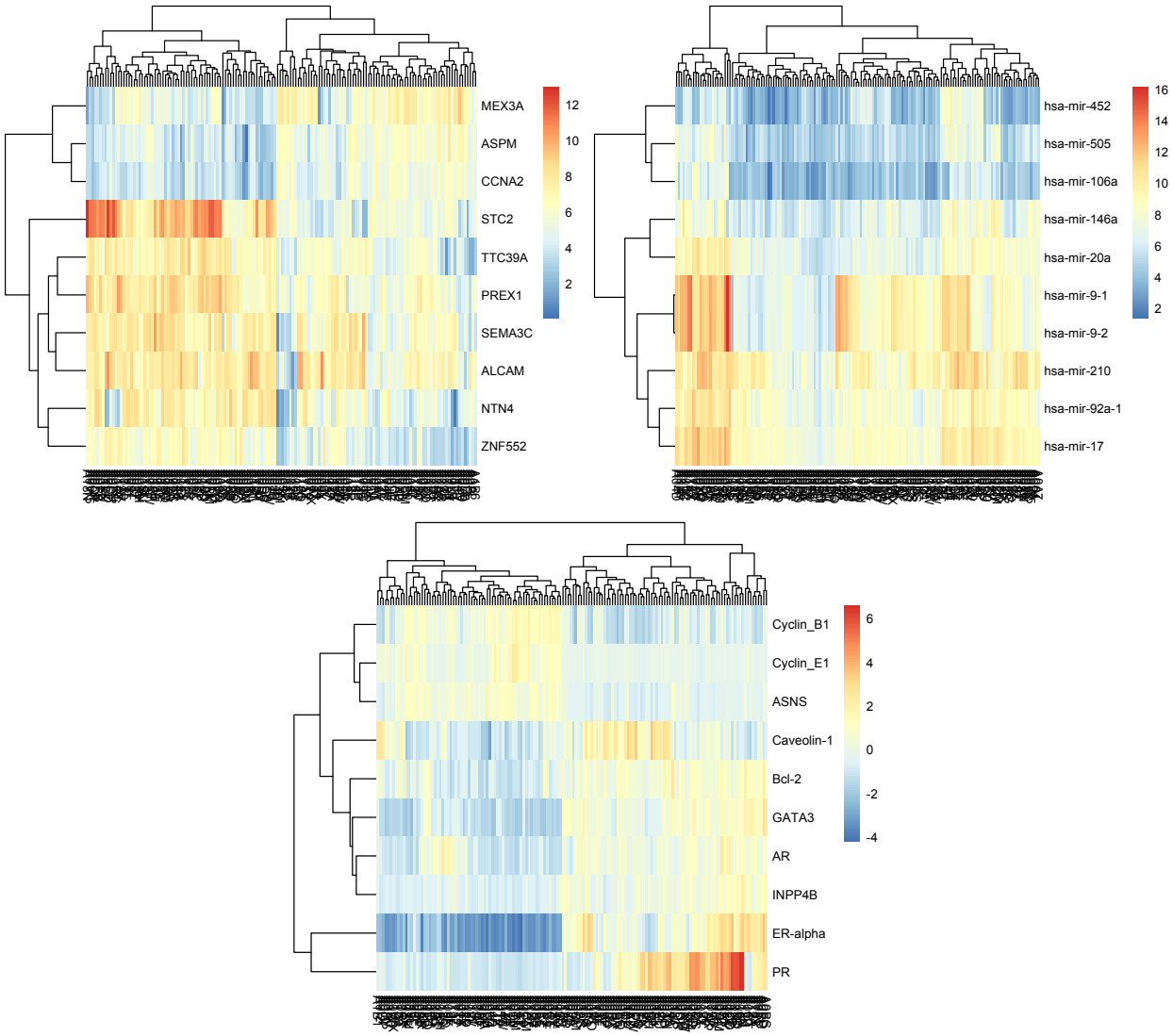


FIGURE 6.20: Analyse du pattern des observations selon les biomarqueurs retenus dans la première composante. Les modalités représentées sont les données d'ARNm (haut, gauche), de microARN (haut, droite) et de protéomique (bas). Graphique obtenu via la fonction `heatmap()` appliquée sur les données du TCGA

Finalement, la Figure 6.21 permet d'observer les trois variables les plus corrélés (positivement ou négativement) avec le premier vecteur de scores, dans les trois modalités. Etant donné que le premier vecteur de scores est celui qui discrimine le mieux le type de cancer du sein, les graphiques pour les autres facteurs n'ont pas été générés.

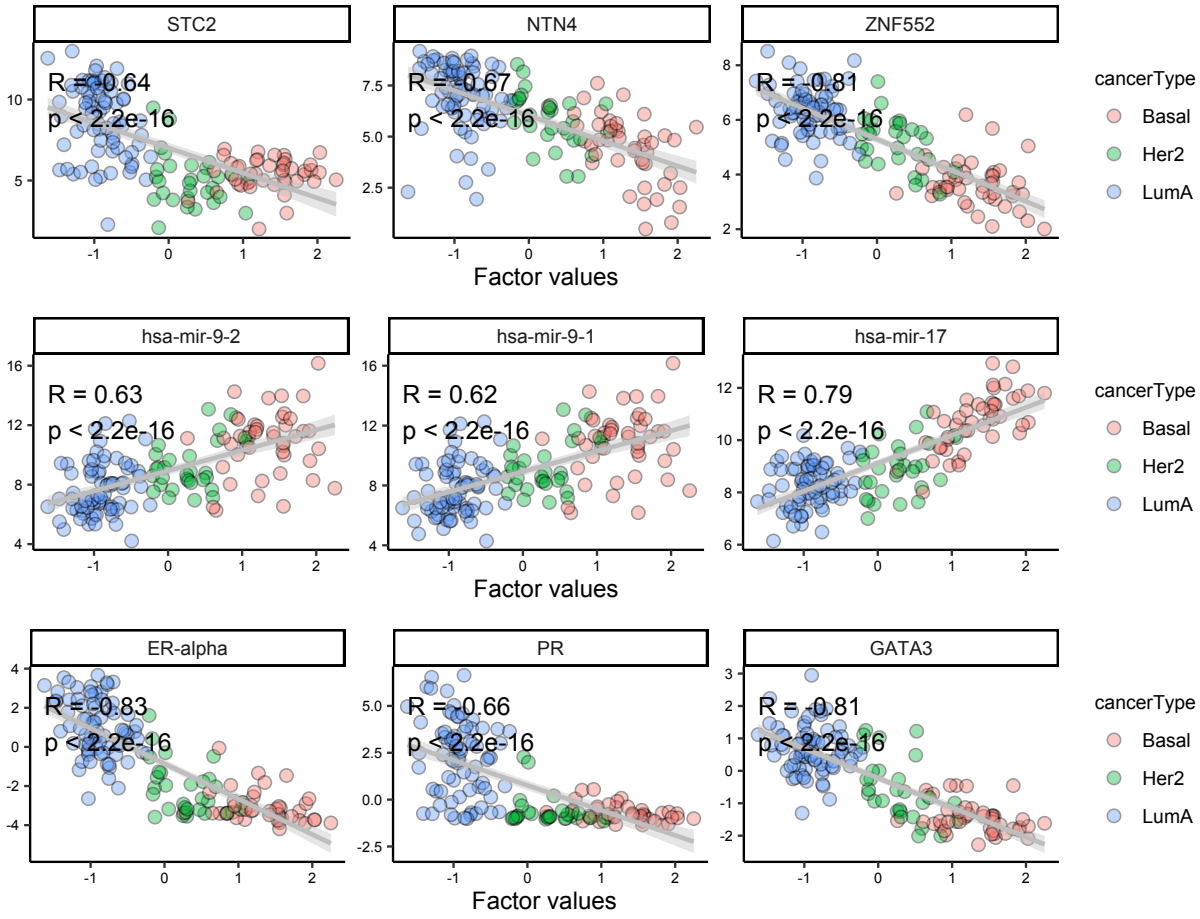


FIGURE 6.21: Analyse de la corrélation entre les biomarqueurs les plus influents au sein de chaque modalité et le premier vecteur de scores. Graphique obtenu via la fonction `plot_data_scatter()` appliqué sur les données du TCGA selon les modalités ARNm (haut), miARN (milieu) et protéomique (bas)

6.1.3.5 Analyse des biomarqueurs estimés

Les biomarqueurs les plus influents dans la construction du premier vecteur de scores calculé par la méthode MOFA+ (ceux ayant un poids associé important dans le vecteur de loadings) peuvent être trouvés via la Figure 6.22. Assez logiquement, on retrouve les mêmes biomarqueurs que ceux identifiés dans les patterns de la Figure 6.20. Il faut noter que le vecteur de loadings a été standardisé sur ce graphique. Ceci n'a pas d'incidence dans l'allure du graphique et cette transformation peut donc être utilisée si l'intention est de comparer l'allure générale des poids des variables.

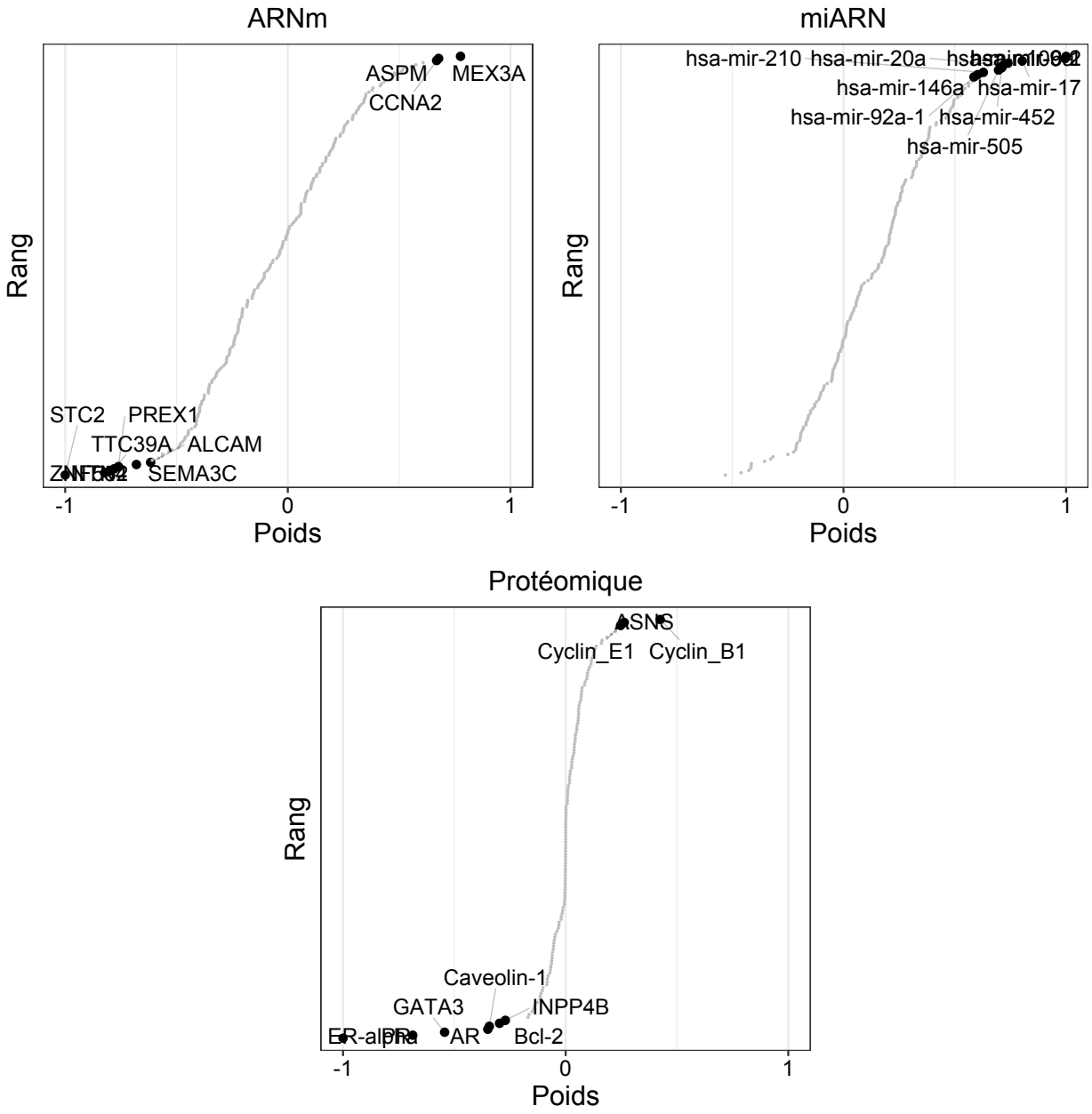


FIGURE 6.22: Analyse des vecteurs de loadings et des poids associés aux variables initiales pour la construction du premier vecteur de scores. Graphique obtenu via la fonction `plot_weights()` appliquée sur les données du TCGA

Finalement, la Figure 6.23 permet de trouver précisément le nom des variables identifiées comme biomarqueurs pour la construction du premier facteur ainsi que le poids (standardisé) qui leur est associé. Ceci sera nécessaire pour comparer les deux méthodes dans la section suivante.

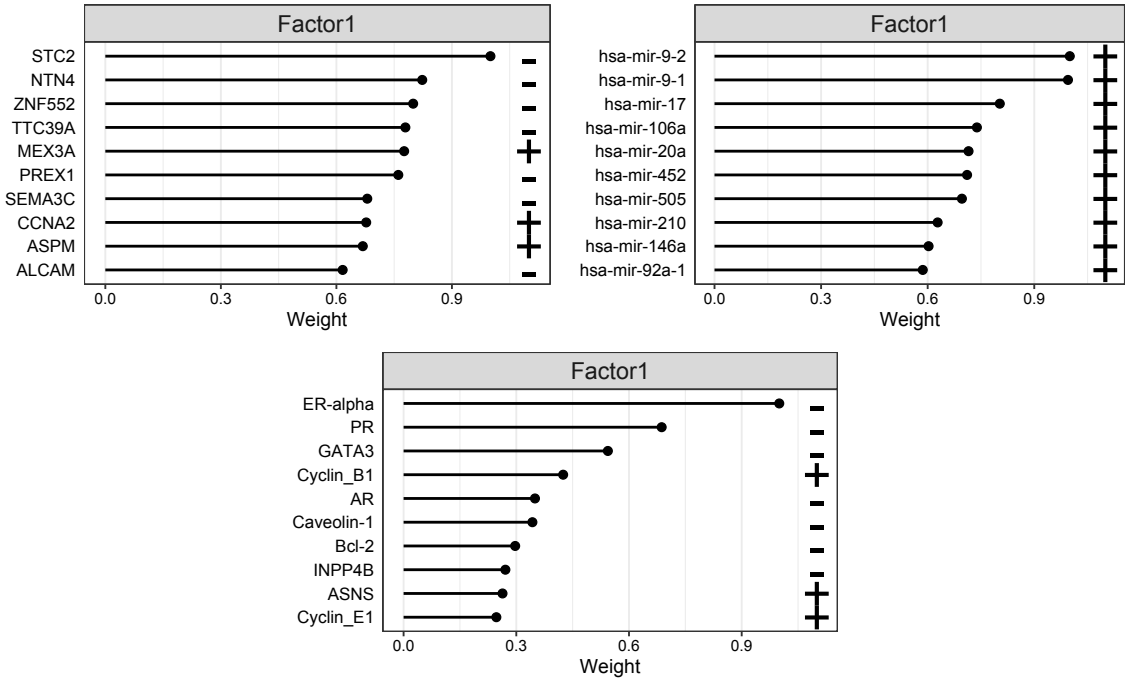


FIGURE 6.23: Identification des biomarqueurs les plus influents d'ARNm (haut, gauche), miARN (haut, droite) et protéomique (bas). Graphique obtenu via la fonction `plot_top_weights()` appliquée sur les données du TCGA

6.1.4 Conclusions de l'analyse des données du TCGA

Les deux méthodes sont construites de manière différentes : l'une cherche des vecteurs de scores qui sont corrélés un maximum avec ceux de la variable d'intérêt et, dans une moindre mesure, avec les vecteurs de scores des autres blocs. L'autre cherche à capturer un maximum de variabilité des données en tenant compte de la structure des données en blocs.

On a vu que, pour la méthode MOFA+, le premier vecteur de scores est celui qui discrimine le mieux les différents types de cancer (Figure 6.18). Il est à noter cependant que cette information n'est pas utilisée par le modèle puisqu'il s'agit d'une méthode non-supervisée. C'est également le premier vecteur de scores qui capture un maximum de variabilité des données (Figure 6.16). Les variables avec un poids maximum dans ce vecteur de scores peuvent donc être considérées comme des biomarqueurs potentiels des trois maladies (Figures 6.22 ou 6.23).

En tout, le jeu de données complet comporte 526 variables, séparées en trois blocs d'ARN messager, de micro ARN et de protéomique. Parmi ces 526 variables, les deux méthodes s'accordent sur quelques variables les plus influentes, désignées comme biomarqueurs.

On a vu que la méthode DIABLO gardait, au total, 30 variables pour construire les trois vecteurs de scores partiels du bloc d'ARN messager, 18 pour ceux des micro ARN et 15 pour ceux de protéomique. Nous avons donc comparé ces biomarqueurs aux 30, 18 et 15 biomarqueurs avec les coefficients les plus élevés dans le vecteur des loadings de MOFA+ (dans les blocs respectifs) comme illustré sur le diagramme de Venn de la Figure 6.24.

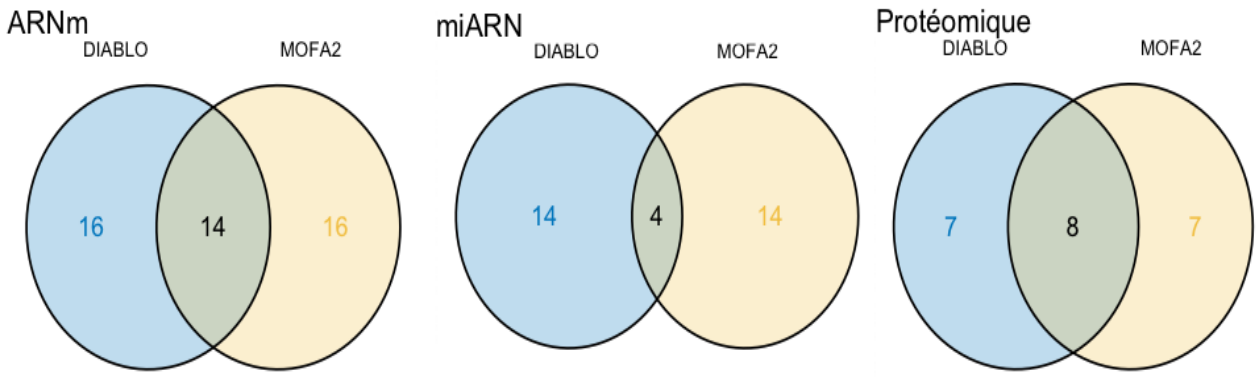


FIGURE 6.24: Analyse des biomarqueurs trouvés par les deux méthodes. Graphique obtenu avec la fonction `ggven()` du package *ggven*

Le tableau ci-dessous s'intéresse aux biomarqueurs qui ont été identifiés de cette manière par les deux méthodes et qui peuvent donc être considérés comme des biomarqueurs potentiellement sérieux. Le tableau renseigne, pour chacun des biomarqueurs, le poids qui lui est associé dans un des vecteurs de loadings ainsi que son rang. Le rang 1 est attribué au biomarqueur avec le poids qui lui est associé le plus important (en valeur absolue) dans sa modalité. Les biomarqueurs apparaissent dans l'ordre croissant des rang de la méthode DIABLO. Les biomarqueurs les plus influents sont donc ceux ayant un rang peu élevé dans chaque méthode.

Modalité	Identifiant	DIABLO		MOFA+	
		Poids	**Rang**	**Poids**	**Rang**
ARNm	C1QB	-0.592	2	0.789	21
	ZNF552	-0.522	5	-1.114	3
	KDM4B	-0.442	6	-0.848	15
	SAMD9L	- 0.350	7	0.839	17
	LRIG1	-0.314	8	-0.786	22
	CCNA2	0.311	9	0.944	12
	PREX1	-0.308	10	-1.060	6
	FUT8	-0.266	13	-0.744	27
	TTC39A	-0.249	15	-1.085	4
	C4orf34	-0.245	16	-0.809	18
	ASPM	0.138	19	0.934	13
	MEX3A	0.102	22	1.081	5
	STC2	-0.075	25	-1.393	1
	SEMA3C	0.075	26	-0.948	11
miARN	hsa-mir-17	0.675	2	0571	5
	hsa-mir-505	0.410	7	0.487	7
	hsa-mir-130b	0.357	8	0.464	11
	hsa-mir-20a	0.007	18	-0.435	13
Protéomique	ER-alpha	-0.618	5	-1.663	1
	GATA3	-0.555	6	-0.905	3
	ASNS	0.281	7	0.438	13
	Cyclin_B1	0.259	9	0.707	4
	AR	-0.245	10	-0.582	8
	Cyclin_E1	0.176	12	0.411	14
	PR	-0.155	13	-1.143	2
	INPP4B	- 0.111	14	-0.451	12

La proportion de biomarqueurs retrouvés par les deux méthodes sur les 30 (ARNm), 18 (miARN) et 15 biomarqueurs (protéomique) ayant un poids le plus important dans le vecteur des loadings des différents blocs est reprise dans le tableau suivant :

Modalité	Nbr de variables initiales	Nbr de biomarqueurs observés	Nbr de biomarqueurs communs	% de biomarqueurs de DIABLO retrouvés par MOFA+
ARNm	200	30	14	46.7%
miARN	184	18	4	22.2%
Protéomique	142	15	8	53.3%

Plusieurs articles semblent confirmer le choix des biomarqueurs. En effet, l'expression du gène *Stanniocalcin 2* (STC2) [10], le micro ARN17(hsa-mir-17) [16] ou encore la présence de protéine *estrogen receptor* α (ER-α) [3] sont couramment utilisées comme biomarqueurs dans le cas des études du cancer du sein.

Il est par contre curieux de constater que certaines variables n'ont pas été considérées comme biomarqueurs par une des deux méthodes. Par exemple, la protéine *human epidermal growth factor receptor 2* (HER2) n'a pas été retrouvée par MOFA+, alors que c'est précisément l'abondance de cette protéine qui définit le cancer du sein de type Her.

Finalement, la Table 6.1 indique le coefficient de corrélation des vecteurs de scores des deux méthodes, selon chaque modalité. On peut voir que les premiers vecteurs de scores des deux méthodes sont très fortement corrélés positivement, pour chacune des modalités (> 0.85). Le troisième vecteur de scores de DIABLO semble également fortement corrélé positivement avec le troisième vecteur de scores de MOFA+, dans chaque modalité (> 0.7). Par contre, le second vecteur de scores de DIABLO, qui permettait d'augmenter la discrimination du cancer de type *Her2*, ne semble pas particulièrement fortement corrélé avec un des vecteurs de scores de MOFA+.

	ARNm	**MOFA+**				
		VS1	VS2	VS3	VS4	VS5
DIABLO	VS1	0.97	0.23	-0.19	0.04	0.17
	VS2	0.01	-0.22	0.01	-0.01	-0.48
	VS3	-0.01	0.31	0.91	-0.03	-0.22
	miARN	**MOFA+**				
		VS1	VS2	VS3	VS4	VS5
DIABLO	VS1	0.87	0.42	-0.13	-0.12	0.15
	VS2	0.12	-0.27	0.04	0.12	-0.28
	VS3	-0.01	0.19	0.80	-0.07	-0.08
	Protéomique	**MOFA+**				
		VS1	VS2	VS3	VS4	VS5
DIABLO	VS1	0.94	0.14	-0.18	0.12	0.06
	VS2	-0.06	-0.11	-0.03	-0.10	-0.36
	VS3	0.01	0.16	0.71	-0.31	-0.13

TABLE 6.1: Analyse de la corrélation entre les vecteurs de scores des deux méthodes

6.2 Stemformatics

Les données utilisées proviennent de la plateforme informatique Stemformatics. Elles ont été fournies par le package *mixOmics* et sont décrites dans le chapitre 2.5.2 de ce travail. L'analyse complète, reprenant davantage de graphiques, est disponible à l'adresse Git Hub https ://github.com/LionelPanneel/multi-omics. Seuls les graphiques le plus pertinents sont repris dans ce travail.

6.2.1 Visualisation des données via une ACP

Cette première section modélise une Analyse en Composantes Principales sur les données de Stemformatics. La Figure 6.25 montre la projection dans les deux premières composantes des données brutes (gauche) et après avoir centré et réduit les variables par bloc (droite). Chaque couleur correspond à un type de cellule tandis que la forme du point représente l'étude à laquelle chaque observation appartient. On peut se rendre compte que, sur les données brutes, les observations sont regroupées en fonction des différentes études. La standardisation des données a évité les clusters selon les études mais les données ne sont pas regroupées selon le type de cellule à laquelle elles appartiennent.

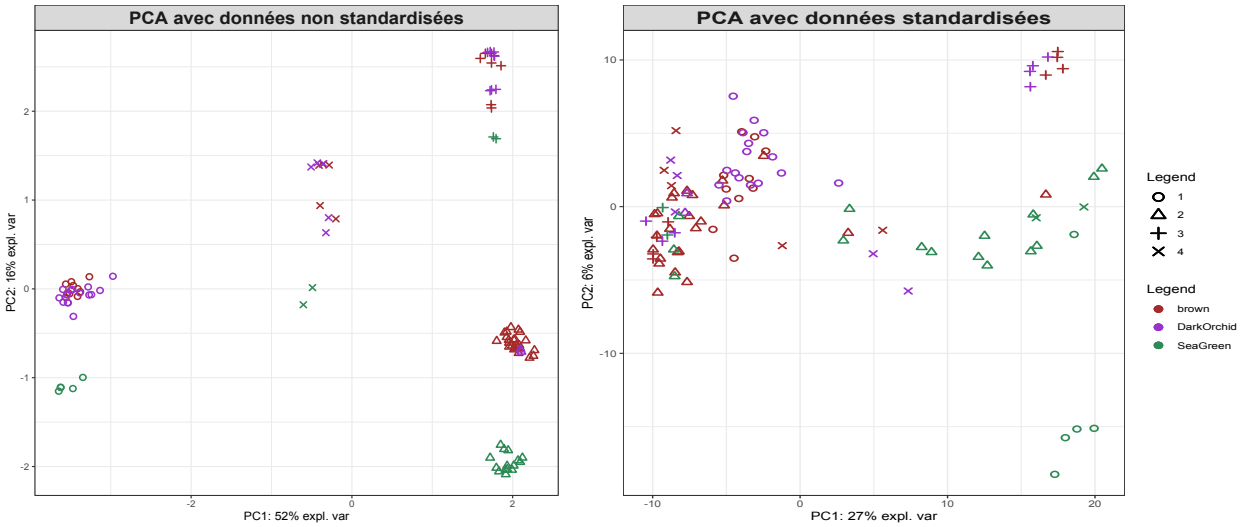


FIGURE 6.25: Projection des données Stemformatics dans les deux premières composantes d'une ACP sur données brutes (gauche) ou centrées et réduites (droite)

6.2.2 Analyse de Stemformatics avec MINT

6.2.2.1 Paramètres du modèle

Les paramètres du modèle sont les suivants :

• MixOmics propose de garder $G - 1 = 2$ composantes, où G est le nombre de groupes de la variable d'intérêt. Toutefois, en observant la Figure 6.26, on peut voir que le BER (voir Chapitre 2.6.2) est minimal lorsque 1 seule composante est utilisée en tenant compte de la distance de Mahalanobis (voir Chapitre 2.6.4) : $BER = 0.2$. Cependant, lors de cette étude

de cas, nous garderons **trois composantes**. Ce nombre de composantes nous permettra d'utiliser les outils graphiques proposés par le package et d'analyser plus spécifiquement les données. En outre, le BER est plus petit avec trois composantes ($BER = 0.3$) qu'avec $G - 1 = 2$ composantes ($BER = 0.38$).

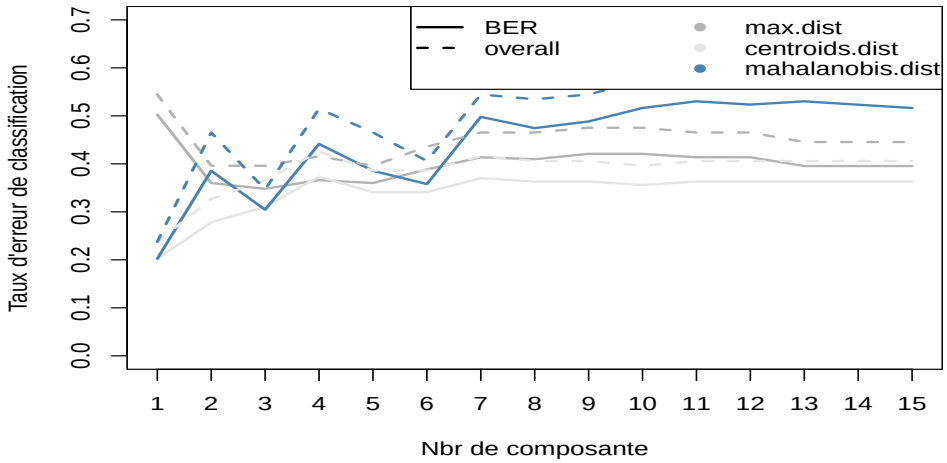


FIGURE 6.26: Analyse du taux d'erreur de classification en fonction du nombre de composantes choisies par MINT sur les données Stemformatics. Graphique obtenu via la fonction `perf()` appliquée sur un objet de type MINT

• Le nombre optimal de variables avec un coefficient non nul dans le vecteur des loadings global est obtenu via la fonction `tune()` du package de *mixOmics* et est représentée dans le tableau ci-dessous. Il s'agit d'un nombre de variables identique pour les trois blocs. On peut s'apercevoir que le nombre de variables à garder dans le premier vecteur de loadings est plus important que dans l'analyse des données du TCGA, ce qui est moins efficace en termes d'interprétation du modèle.

$\mathbf{a}_1$	$\mathbf{a}_2$	$\mathbf{a}_3$
72	6	14

L'évolution du taux d'erreur en fonction du nombre de variables ayant un poids non nul dans le vecteur des loadings est visible sur la Figure 6.27. On voit clairement que, pour chaque composante, le nombre de variables choisi est le plus petit nombre de variables qui minimise le taux d'erreur.

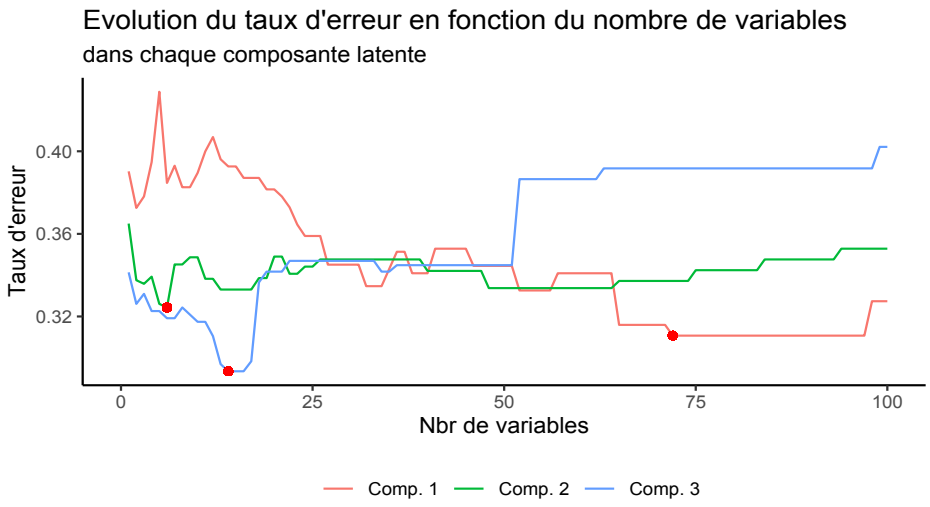


FIGURE 6.27: Evolution du taux d'erreur en fonction du nombre de variables dans chaque composante de MINT. Graphique illustrant les résultats obtenus avec la fonction `tune()` appliquée sur les données de Stemformatics

Avec les composantes ainsi créées, le taux de mauvaise classification de chaque type de cellules (sur le training set) est donné dans le tableau ci-dessous. Le modèle ainsi obtenu semble très efficace pour distinguer les cellules fibroblastes des deux autres.

	$H = 1$	$H = 2$	$H = 3$
Fibroblaste	0	0	0
hESC	0.267	0.467	0.567
hiPSC	0.298	0.596	0.340

FIGURE 6.28: Taux de mauvaise classification des cellules en fonction de la composante utilisée

Finalement, la Figure 6.29 illustre la proportion de la variance expliquée par chaque composante du modèle. Dans ce cas-ci, la première composante est celle qui explique le plus la variabilité des données (26.06%). On peut voir que la variabilité expliquée par les composantes diminue au fur et à mesure des composantes, ce qui n'est pas obligatoirement le cas avec MINT puisque l'objectif de cette méthode n'est pas de maximiser la variance des données, contrairement à une ACP ou à MOFA+ par exemple.

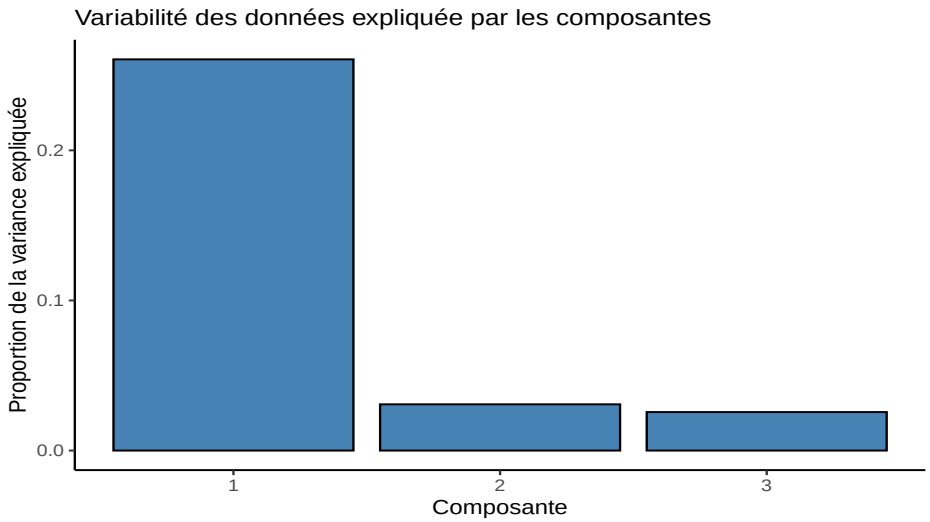


FIGURE 6.29: Analyse de la variance expliquée par les composantes de MINT sur les données de Stemformatics

6.2.2.2 Analyse des observations

La projection des données dans le sous-espace défini par les vecteurs de scores 1 et 2 est illustrée sur la Figure 6.30. On peut constater que la première composante semble permettre une distinction entre les cellules fibroblastes d'une part et les deux autres types de cellules d'autre part, les points représentant les cellules fibroblastes formant un groupe dans lequel on ne retrouve aucune observation d'un des deux autres groupes. La deuxième composante semble améliorer, de manière imparfaite, la distinction entre les cellules hESC et hiPSC. La troisième composante ne semblant pas apporter de pouvoir de classification supplémentaire, son graphique n'a pas été intégré dans ce travail mais est disponible dans l'analyse complète sur le Git Hub (https ://github.com/LionelPanneel/multi-omics).

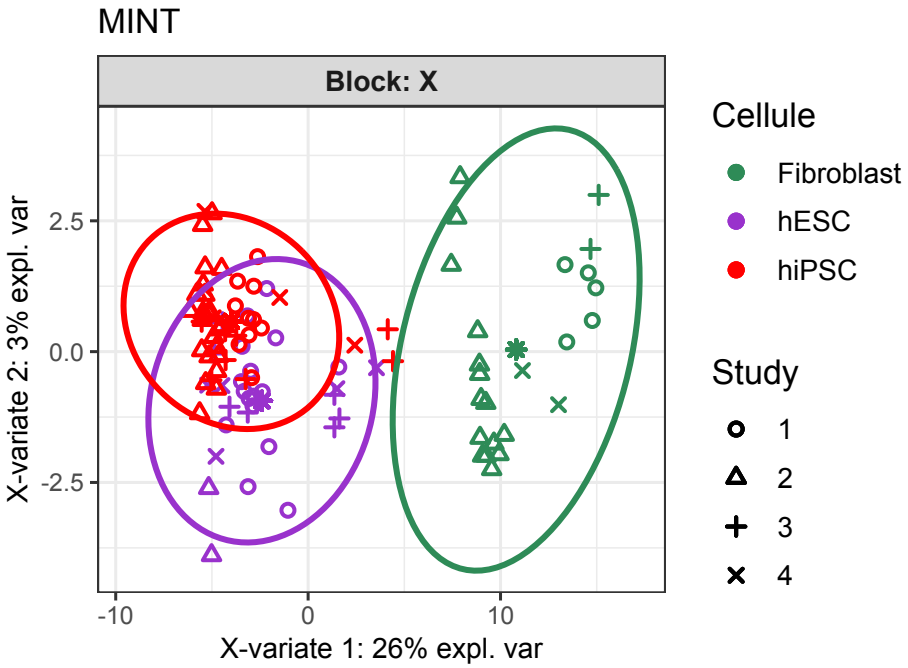


FIGURE 6.30: Analyse des vecteurs de scores globaux selon la première et la deuxième composante. Graphique obtenu via la fonction `plotIndiv()` appliquée sur les données de Stemformatics avec l'option `study = 'global'`

La Figure 6.31 permet de constater le pouvoir de discrimination des deux premiers vecteurs des scores partiels au sein de chaque étude. Une fois de plus, on peut constater que le premier vecteur de scores semble permettre une différentiation nette entre les cellules fibroblastes et les deux autres types de cellules. Par contre, ce vecteur ne permet pas de discriminer les cellules hESC des cellules hiPSC.

Le deuxième vecteur des scores partiel permet d'améliorer la distinction entre les cellules hESC et hiPSC particulièrement dans les modalités 1, 2 et 4. Par contre, il ne discrimine pas ces deux types de cellules des cellules fibroblastes, sauf dans l'étude 3.

Ensemble, les deux premiers vecteurs de scores partiels semblent permettre une discrimination correcte des trois types de cellules, particulièrement dans les études 1, 2 et 4.

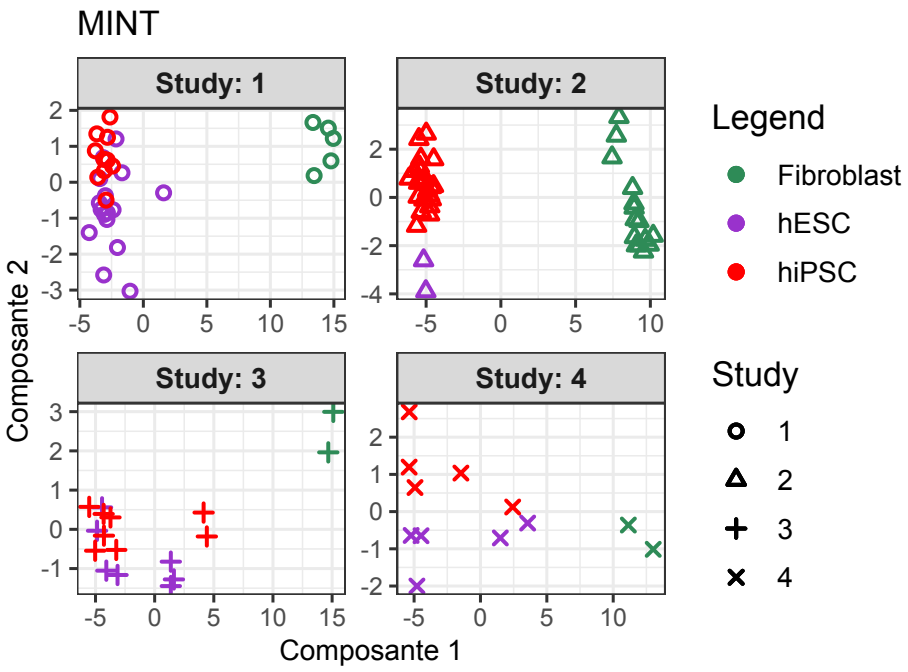


FIGURE 6.31: Analyse des vecteurs de scores partiels selon la première et la deuxième composante. Graphique obtenu via la fonction `plotIndiv()` appliquée sur les données de Stemformatics

La heatmap avec les biomarqueurs ayant un poids non nul dans la première composante est montrée en Figure 6.32. Son pattern met en évidence deux types de cellules (en ligne). Les cellules fibroblastes sont parfaitement classifiées avec le dendrogramme généré. Par contre, les cellules hESC et hiPSC sont moins bien classifiées : de nombreuses observations des deux groupes sont mélangées par le dendrogramme, ce qui correspond à ce qui a été révélé en Figure 6.30.

On peut observer que les biomarqueurs sont divisés en deux parties très nettes. Les biomarqueurs de la partie de gauche prennent des valeurs plutôt négatives lorsque les cellules sont de type fibroblastes et une valeur plutôt positive avec les deux autres types de cellules. Les autres biomarqueurs ont la tendance inverse.

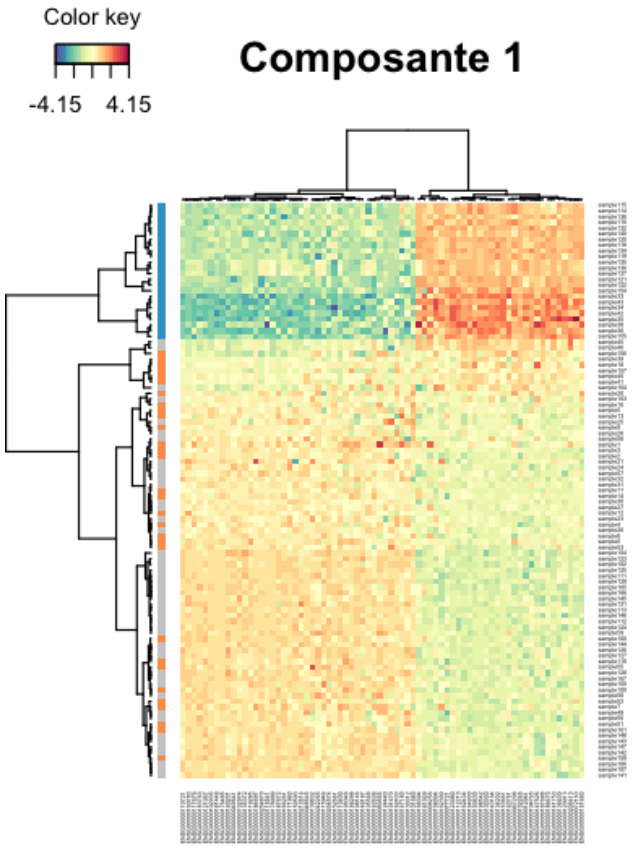


FIGURE 6.32: Analyse des patterns des observations et des biomarqueurs. Graphique obtenu via la fonction `cim()` appliqué sur les données de Stemformatics

6.2.2.3 Analyse des biomarqueurs estimés

On peut voir sur la Figure 6.33 que de nombreux biomarqueurs sont bien représentées sur la première composante du modèle ($r > 0.90$ ou $r < -0.9$) mais il n'a pas été possible d'identifier correctement ces variables à cause de leur chevauchement. Une analyse plus approfondie de ces biomarqueurs sera toutefois disponible en Figure 6.35. Par contre, peu de variables sont fortement corrélées avec les composantes suivantes.

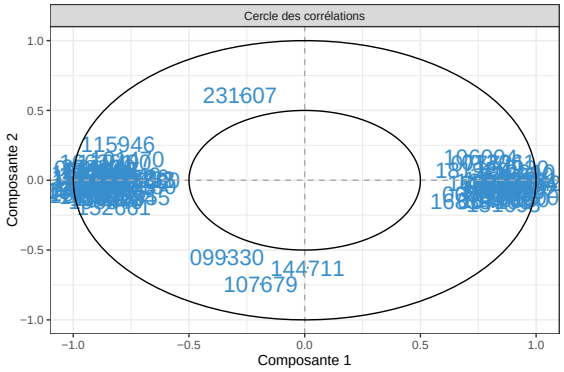


FIGURE 6.33: Graphique obtenu via la fonction `plotVar()` appliquée sur les données de Stemformatics

La Figure 6.34 permet de comprendre comment sont composés les vecteurs de loadings et d'identifier les gènes les plus influents dans la construction des vecteurs de scores du modèle.

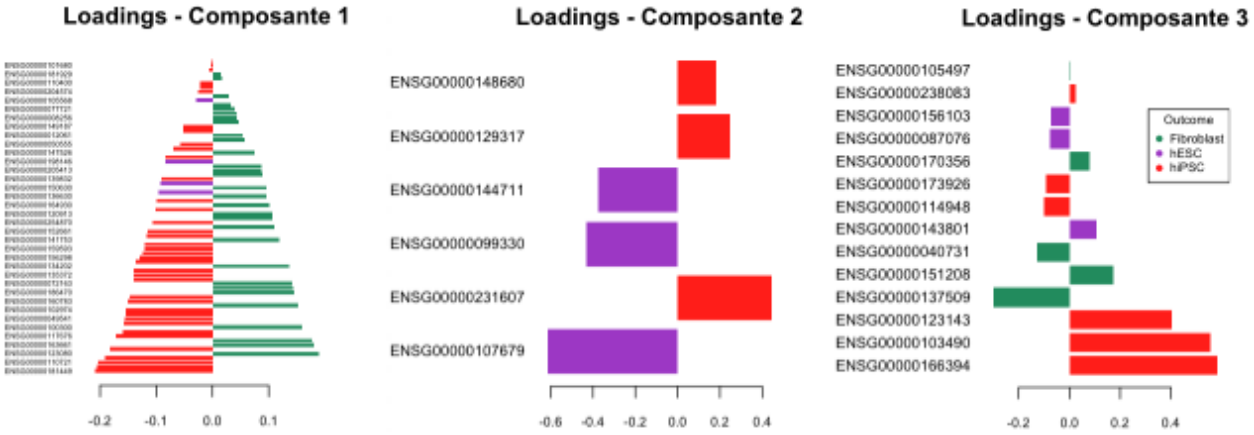


FIGURE 6.34: Analyse des vecteurs de loadings selon la première (gauche), deuxième (Milieu) ou troisième composante (droite). Graphique obtenu via la fonction `plotLoadings()` sur les données de Stemformatics

La Figure 6.35 permet d'analyser plus précisément les variables corrélées avec un des vecteurs de scores de la variable d'intérêt ($\mathbf{u}_h$). Afin de ne pas surcharger le graphique, le seuil de cette Figure a été fixé de telle sorte que seules les cinq variables les plus corrélées soient visibles pour chaque composante.

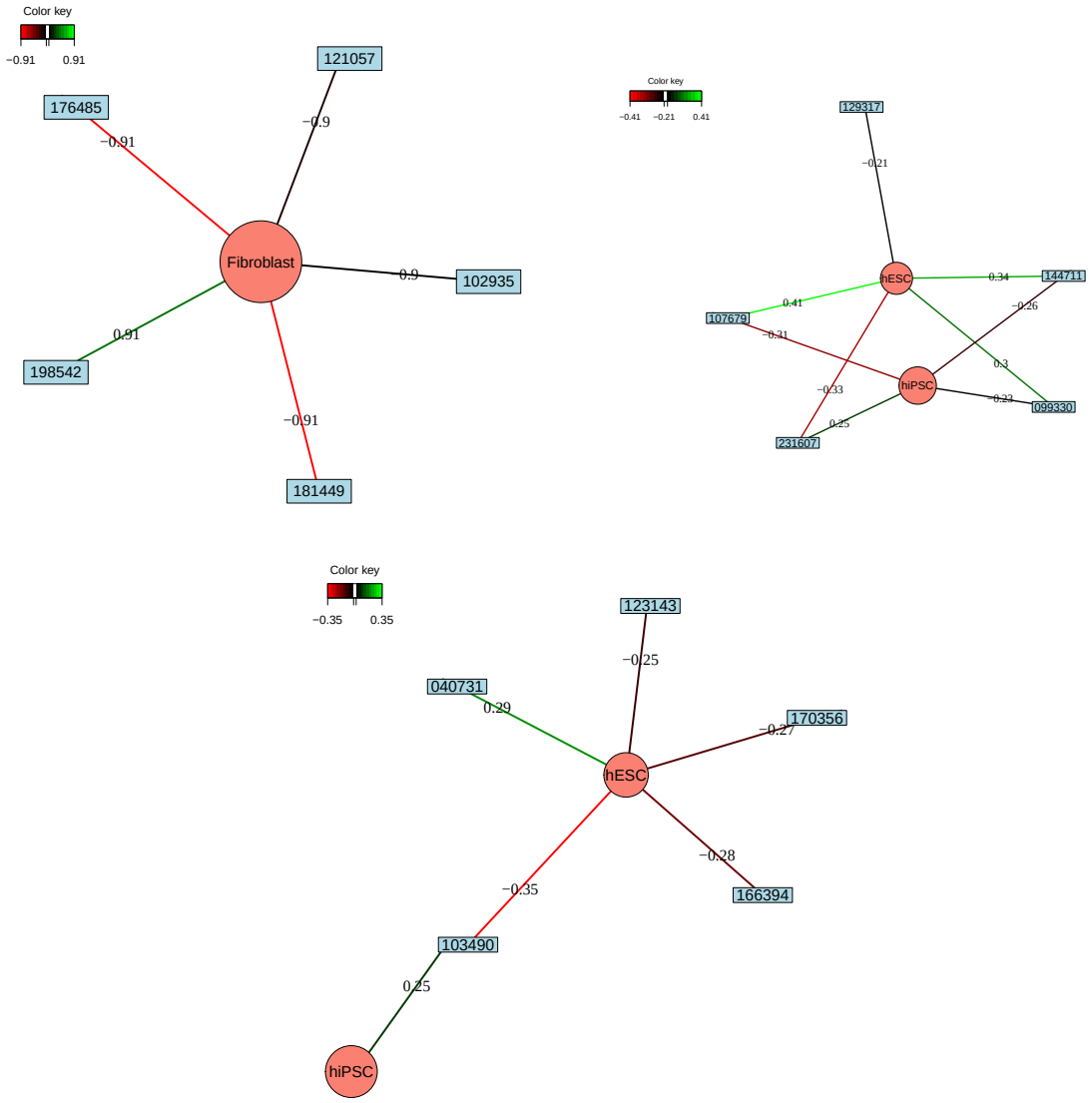


FIGURE 6.35: Graphique obtenu via la fonction `network()` appliquée sur les données de Stemformatics selon la première (haut, gauche), la seconde (haut, droite) ou la troisième composante (bas)

Il est à noter que la lecture de ce type de graphique devient difficile à lire dès lors que plus d'une quinzaine de biomarqueurs sont affichés. En effet, dans ce cas, les liens entre les noeuds ne sont parfois plus visibles, leur couleur ou la valeur du coefficient est parfois cachée par d'autres informations.

L'analyse des graphiques générés avec la fonction `network()` permet de mieux comprendre la construction de chaque composante. La Table 6.2 reprend les informations de la Figure 6.35 et est divisée en trois parties, chaque partie liste les cinq variables les plus influentes dans la construction du vecteur de loadings de la composante concernée ($\mathbf{a}_h$). Les valeurs renseignées dans cette Table représentent le coefficient de corrélation entre ces variables et le vecteur de scores de chaque groupe de la variable d'intérêt ($\mathbf{u}_h^{(y)}$) qui lui est associé.

On peut s'apercevoir que chaque composante semble être construite avec des variables

plus ou moins corrélées avec un certain type de cellule :

• Ainsi, il semblerait que la première composante soit construite avec des variables très fortement corrélées avec le vecteur de scores des cellules fibroblastes.

• La deuxième et la troisième composante semblent être construites avec des variables principalement corrélées avec le vecteur de scores des cellules hESC.

		Coef. de corr. Fibroblaste	Coef de corr. hESC	Coef de corr. hiPSC
Construit a_1	176485	**-0.914**	0.249	0.558
	198542	**0.907**	-0.247	-0.553
	181449	**-0.915**	0.249	0.558
	121057	**-0.902**	0.246	0.551
	102935	**-0.900**	0.245	0.550
Construit a_2	144711	-0.044	**0.344**	-0.261
	107679	-0.052	**0.408**	-0.309
	231607	-0.043	**0.335**	0.253
	099330	-0.038	**0.301**	-0.228
	129317	0.027	**-0.211**	-0.160
Construit a_3	103490	0.054	**-0.352**	0.252
	170356	0.041	**-0.267**	0.191
	040731	-0.045	**0.294**	-0.210
	123143	0.038	**-0.249**	0.178
	166394	0.043	**-0.281**	0.202

TABLE 6.2: Synthèse des graphiques *network* de MINT sur les données Stemformatics, dans les trois composantes

6.2.2.4 Performance du modèle

La performance du modèle a été testée sur le test set défini au Chapitre 2.5.2.

Si on analyse les courbes ROC du modèle, sur la Figure 6.36, on peut voir que le modèle est très efficace dans la classification des cellules fibroblastes. On constate également que la deuxième composante semble améliorer la classification des cellules hESC et hiPSC car les courbes correspondantes se rapprochent du coin supérieur gauche du graphique et que l'aire sous la courbe a tendance à se rapprocher de 1 pour les trois types de cellules.

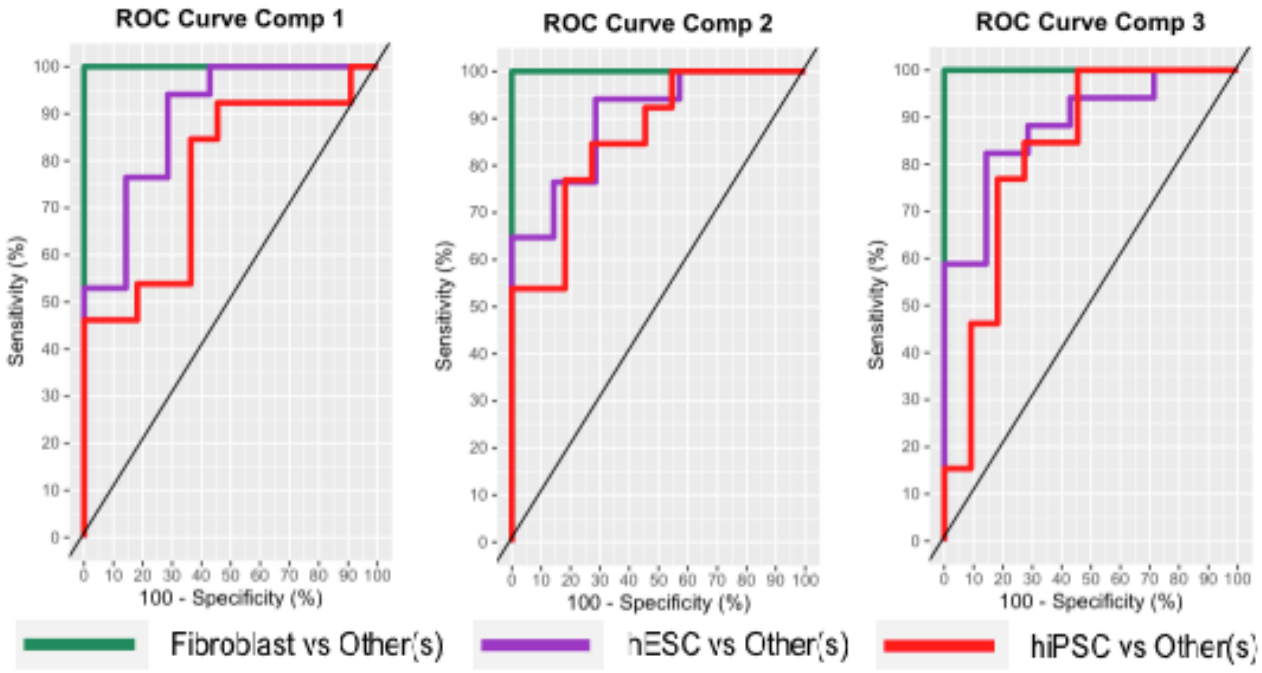


FIGURE 6.36: ROC-curve de MINT pour un modèle avec une seule (gauche), deux (milieu) ou trois composantes (droite), sur les données de Stemformatics. Graphiques obtenus via la fonction `auroc()` appliquée sur les données de Stemformatics

Pour plus de lisibilité, les valeurs d'AUC ont été retranscrites sur la Table 6.3. On peut remarquer que l'ajout d'une troisième composante au modèle semble avoir dégradé légèrement les valeurs de l'aire sous la courbe relatives aux cellules hiPSC et hESC. Il est probable que cela soit dû à un modèle légèrement overfitté.

AUC	Fibroblaste	hESC	hiPSC
Comp 1	1	0.8908	0.7692
Comp 1 - 2	1	0.8992	0.8601
Comp 1 - 3	1	0.8824	0.8252

TABLE 6.3: Analyse de l'Aire Sous la Courbe pour un modèle à une, deux ou trois composantes.

La matrice de confusion générée sur la base du test set (Chapitre 2.5.2) est donnée dans la Table ci-dessous. Toutes les cellules fibroblastes ont été reconnues correctement. Par contre, la détection des cellules hESC est plus hasardeuse sur ce jeu de données. La détection des cellules hiPSC n'est pas parfaite non plus.

		Classe prédite		
		Fibroblaste	hESC	hiPSC
Classe	Fibroblaste	6	0	0
	hESC	0	4	3
	hiPSC	0	4	7

TABLE 6.4: Matrice de confusion générée sur la base du test set des données de Stemformatics.

Cette difficulté à distinguer les cellules hESC et hiPSC se traduit également par les faibles valeurs du score F_1 (voir Chapitre 2.6.3) données dans la Table ci-dessous. Ces valeurs permettent de constater une fois de plus que le modèle est très efficace pour distinguer les cellules fibroblastes et plutôt médiocre pour identifier les deux autres types de cellules.

	Fibroblaste	hESC	hiPSC
F1	1	0.533	0.667

6.2.3 Analyse de Stemformatics avec MOFA+

Les hyperparamètres du modèle de MOFA+ sont les suivants :

• Afin d'assurer une bonne convergence de l'algorithme de stochastic gradient ascent, le paramètre ρ est défini avec $\tau = 1$ et $\kappa = 0.5$:

$$\rho_i = \frac{\tau}{(1 + \kappa \cdot i)^{3/4}}$$

• La taille du minibatch $\mathcal{B}$ est fixée à la moitié de l'échantillon total, selon les recommandations de Argelaguet et al [5].

6.2.3.1 Vérification du modèle

Le modèle final a été créé avec 5 composantes. Nous avons choisi de garder 5 composantes pour les raisons expliquées au point suivant. Ce nombre de composantes permet de capturer entre 38% et 69% de la variance totale des données dans chaque étude. La corrélation entre les vecteurs de scores est visible sur la Figure 6.37. On peut observer que les facteurs sont globalement peu corrélés entre eux et donc qu'aucune erreur n'est visible de ce point de vue.

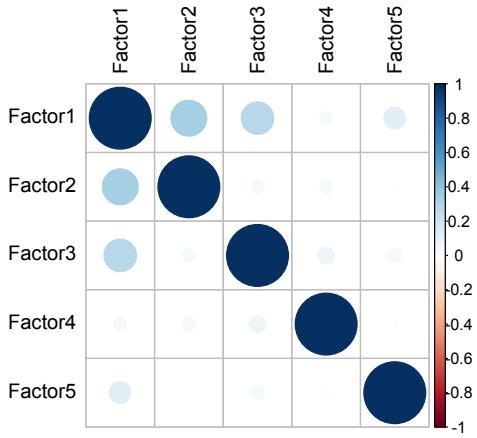


FIGURE 6.37: Corrélation des cinq facteurs de MOFA+ sur les données de Stemformatics

6.2.3.2 Analyse de la variance

Le pourcentage de la variance expliquée dans chaque bloc est représenté sur la Figure 6.38. On peut voir que, pour chaque étude, le modèle avec cinq facteurs capture une grande partie de la variance. La variance expliquée de l'étude 4 semble toutefois légèrement moins importante (38.44%). Une analyse plus approfondie peut être faite en observant la Figure 6.39.

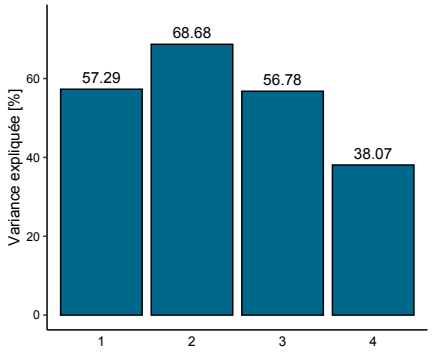


FIGURE 6.38: Variance expliquée par le modèle à cinq facteurs, par étude, sur les données de Stemformatics

Le graphique de la Figure 6.39 est séparé en quatre parties, une pour chaque étude. Sur cette figure, on peut voir que le premier vecteur des scores capture une grande partie de la variabilité des données au sein des différents blocs. On peut constater également que la variance des données de l'étude 2 est particulièrement bien capturée par ce vecteur de scores (et très peu par les facteurs suivants). Le deuxième facteur semble principalement capturer de la variance au sein de la première étude et le troisième facteur capture surtout la variance des données de la troisième étude.

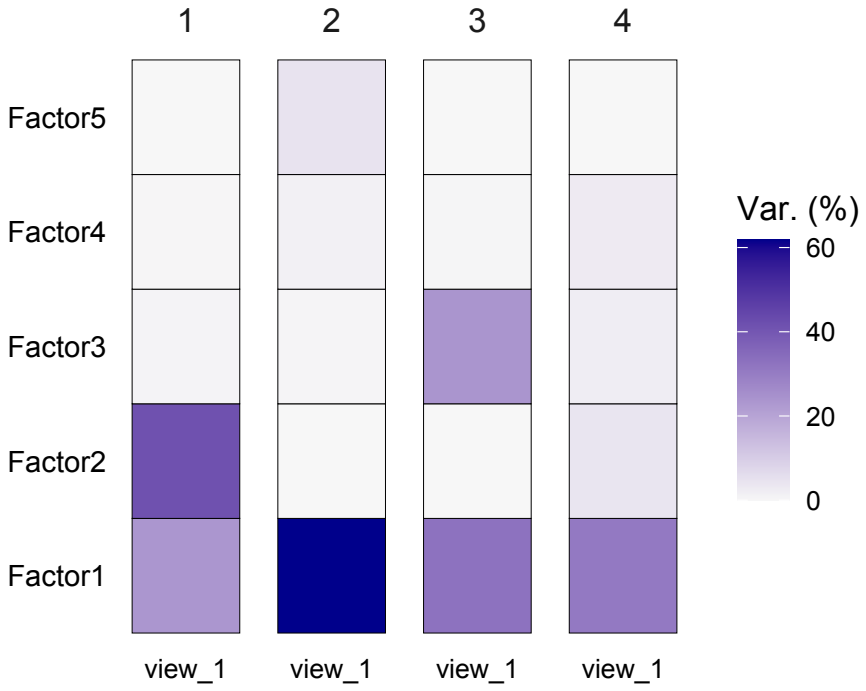


FIGURE 6.39: Variance expliquée par les cinq facteurs, par étude et par facteur. Graphique obtenu via la fonction `plot_factor_cor()` appliquée sur les données de Stemformatics

On remarque également que les facteurs 4 et 5 ne capturent presque pas de variabilité des données. On aurait donc pu réduire le modèle à trois facteurs. Toutefois, il a n'a pas été techniquement possible de procéder à cette modification [1]. Néanmoins, le reste de l'analyse s'intéressera particulièrement aux trois premiers vecteurs de scores.

6.2.3.3 Analyse des observations

On peut voir sur la Figure 6.40 (droite) que le premier vecteur de scores permet de différencier les cellules fibroblastes des deux autres groupes, particulièrement pour les études 1, 2 et 4. A gauche, on constate que le deuxième facteur différencie très fortement les cellules fibroblastes des autres dans la première étude et que le troisième facteur semble surtout permettre une distinction des cellules fibroblastes et hiPSC de la deuxième étude.

1. Ce genre de problème technique peut se présenter lorsque l'utilisateur introduit un seuil minimal de variance expliquée par un des facteurs qui est trop élevé (défini dans l'analyse par `train_opts$drop_factor_threshold`). Le facteur calculé est alors retiré du modèle final (*pruning*). Toutefois, dans notre cas, nous avons tenté de résoudre ce problème en fixant une valeur de 0.0000001 pour ce seuil, mais un modèle à exactement trois facteurs n'a pas pu être généré

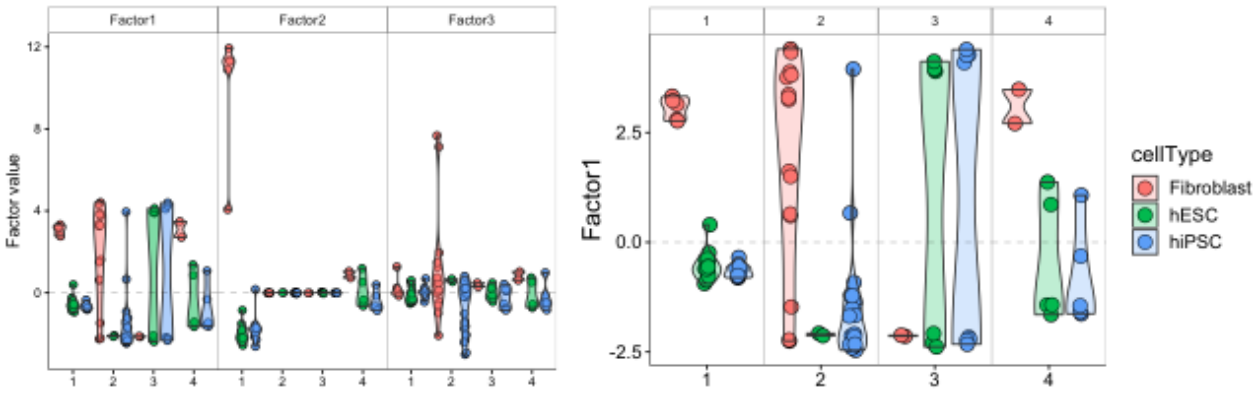


FIGURE 6.40: Analyse des vecteurs de scores. A gauche : analyse par facteur et par étude. A droite : zoom sur le premier facteur. Graphique obtenu via la fonction `plot_factor()` appliquée sur les données de Stemformatics.

La Figure 6.41 représente plusieurs heatmaps. A nouveau, il ne nous a pas semblé évident d'émettre des constats sur ces graphiques pour les les mêmes raisons que celles déjà expliquées au Chapitre 6.1.3.4. Toutefois, on peut constater que :

• Le deuxième graphique (haut, droite) permet de voir qu'il y a un groupe d'observation (en colonne) qui a un comportement très différent des autres. On peut rappeler que le deuxième facteur était celui qui différenciait principalement les données fibroblastes de l'étude 1 (voir Figure 6.40). En zoomant sur cette Figure et en identifiant manuellement chaque observation, on peut s'apercevoir que ces colonnes correspondent bien aux cellules fibroblastes de l'étude 1. On peut remarquer également qu'il y a une variable en particulier (en ligne) qui a un comportement très différent des autres. Cette variable est labellisée ENSG00000181449 et celle-ci avait déjà été renseignée comme un biomarqueur potentiel pour les cellules fibroblastes avec MINT.

• Le troisième graphique (bas) permet d'identifier environ un quart des observations qui semble différent des autres (à droite du graphique). Rappelons que le troisième facteur a principalement pour but de discriminer les cellules fibroblastes des cellules hiPSC de l'étude 2 (voir Figure 6.40). En zoomant sur ces colonnes, on constate qu'il s'agit des cellules fibroblastes de l'étude 2. Les cellules hiPSC de l'étude 2 sont représentées dans le troisième quartile des colonnes de ce graphique. On peut voir qu'elles ont un comportement inverse (valeurs basses pour les variables représentées en-haut du graphique et plus élevé pour celles représentées en-bas).

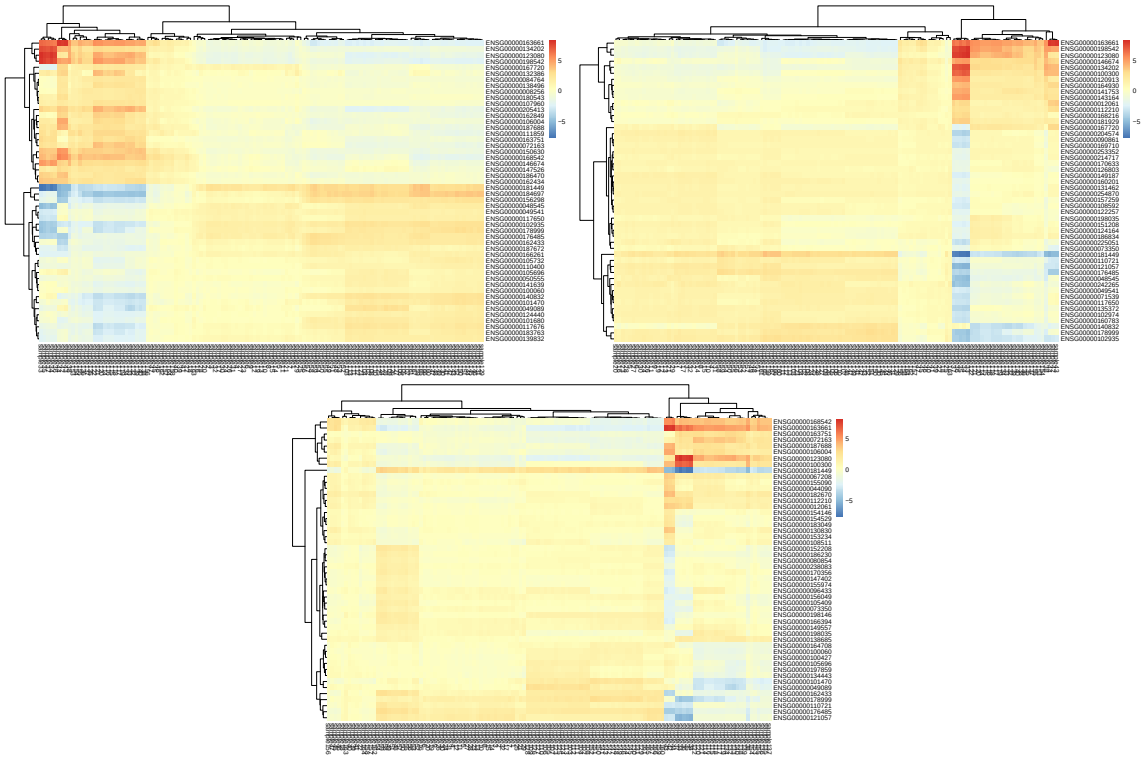


FIGURE 6.41: Heatmap de MOFA+ sur les données de Stemformatics avec le premier facteur (haut, gauche) le second facteur (haut, droite) et le troisième facteur (bas). Graphique obtenu via la fonction `plot_data_heatmap()` appliquée sur les données de Stemformatics

Sur la Figure 6.42, on peut voir la corrélation entre les biomarqueurs trouvés par la méthode et le premier vecteur de scores. On constate qu'en général les cellules fibroblastes sont différentes des autres mais que les cellules hiPSC et hESC sont plutôt confondues, ce qui laisse penser que l'utilisation de ce seul facteur n'est pas suffisante pour séparer les trois types de cellules correctement. Ces graphiques pour les facteurs 2 et 3 n'ont rien révélé d'intéressant en termes de classification du type de cellule et c'est la raison pour laquelle ils n'ont pas été représentés.

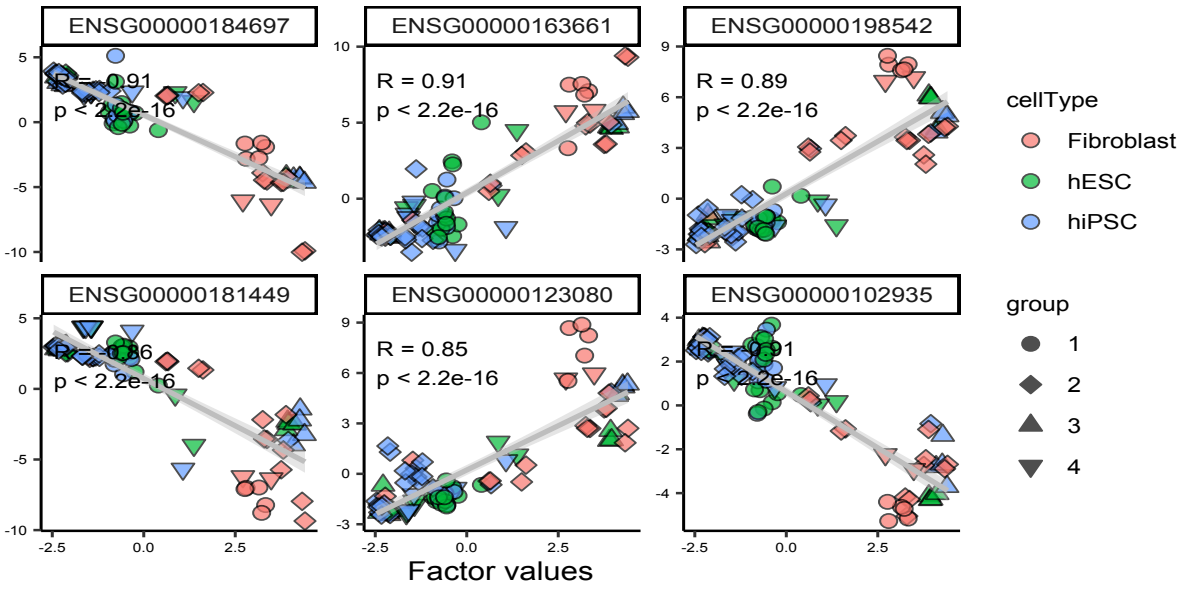


FIGURE 6.42: Corrélation entre les six premiers biomarqueurs et le premier vecteur de scores de MOFA+, sur les données de Stemformatics

6.2.3.4 Analyse des biomarqueurs estimés

Dans la Figure 6.43, on peut voir la structure en "S" représentant les coefficients du vecteur des loadings.

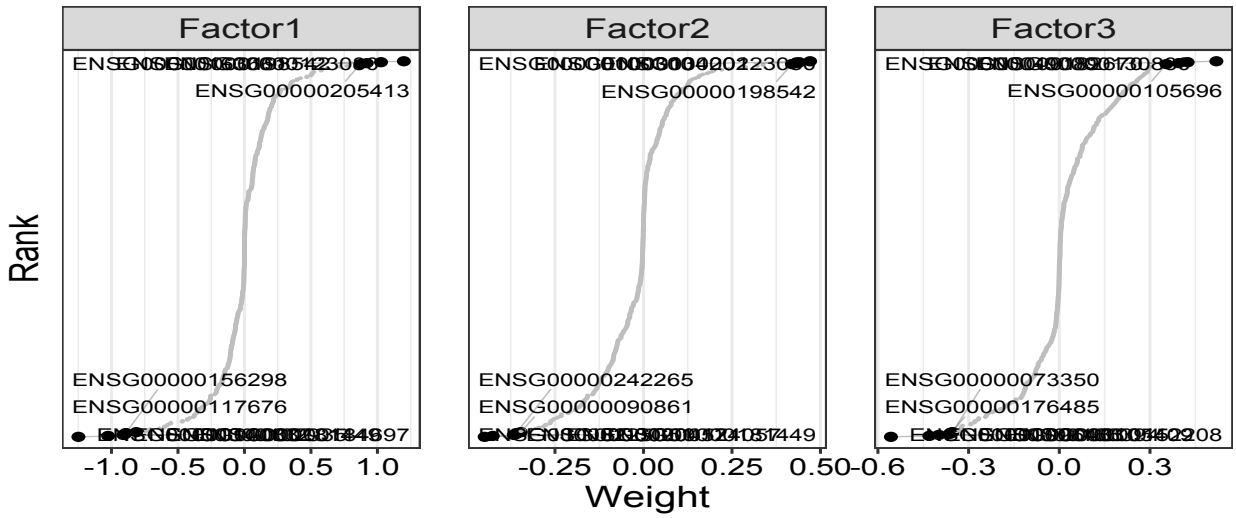


FIGURE 6.43: Analyse des vecteurs de loadings de MOFA+ sur les données de Stemformatics. Graphique obtenu via la fonction `plot_weights()` appliquée sur les données de Stemformatics

L'analyse de la Figure 6.44 permet d'identifier plus précisément les variables les plus influentes dans la construction des facteurs. Les biomarqueurs qui construisent le premier vecteur de scores sont probablement ceux qui ont le plus d'intérêt puisqu'il s'agit d'un facteur qui capte beaucoup de variabilité des données au sein des quatre études, comme on l'a

vu précédemment. Parmi ces biomarqueurs, on peut notamment retrouver le biomarqueur *ENSG00000181449*, qui avait déjà été identifié à plusieurs reprises dans cette analyse.

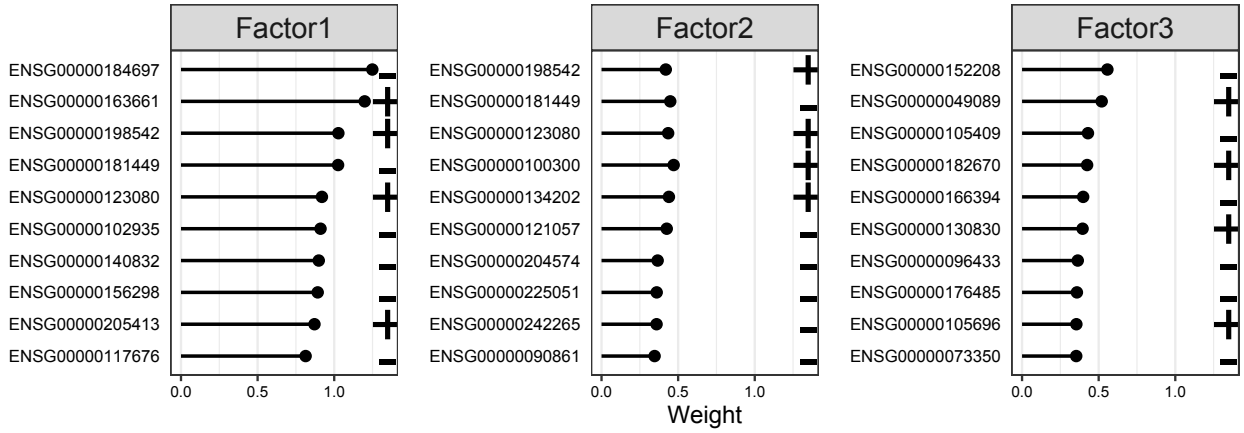


FIGURE 6.44: Analyse des variables les plus influentes dans la construction des vecteurs de scores de MOFA+ sur les données de Stemformatics

6.2.4 Conclusions de l'analyse de Stemformatics

Il est bon de rappeler à ce stade que les deux méthodes sont assez différentes, notamment parce que MINT est une méthode supervisée, utilisant donc l'information de la variable d'intérêt, et que MOFA+ est une méthode non-supervisée.

On a vu que l'analyse en ACP pour les données de Stemformatics ne permettait pas du tout de classifier les données selon le type de cellules auquel elles appartiennent, et n'était donc pas appropriée pour ce type d'analyse. En outre, l'ACP ne permet pas d'identifier des biomarqueurs.

L'analyse de ces données avec MINT nous a permis de trouver trois vecteurs de scores, dont le premier était particulièrement intéressant puisqu'il s'agissait de celui qui classifie le plus efficacement les différents types de cellules (Figure 6.30 et 6.31). Toutefois, ce vecteur, utilisé seul, n'était pas suffisamment efficace pour différencier les cellules hiPSC des cellules hESC. L'utilisation conjointe des trois premiers vecteurs de scores permettait cependant d'obtenir de meilleurs résultats même si le modèle généré restait imparfait pour la différenciation des cellules hESC et hiPSC (Figure 6.36 et Table 6.3).

L'analyse des données avec MOFA+ a permis de générer un modèle à cinq composantes, dont les trois premières étaient celles qui capturaient le plus de variance au sein des données. Le premier vecteur de scores était celui qui permettait de différencier le mieux les cellules fibroblastes des deux autres, particulièrement au sein des études 1, 2 et 4 (Figure 6.40, droite). Il s'agit également du facteur qui capturait un maximum de variabilité des données au sein de chaque étude (Figure 6.39). Les autres facteurs semblaient permettre une différenciation des cellules au sein d'une étude en particulier (Figure 6.40, gauche).

La Table 6.5 donne les coefficients de corrélation entre les différents vecteurs de scores des deux méthodes étudiées. Le premier vecteur de scores de MINT est légèrement corrélé

($|r| > 0.30$) avec les vecteurs de scores 1, 2, 3 et 5 produits par MOFA+. Par contre, aucun coefficient de corrélation entre les vecteur de scores de MOFA+ et le deuxième vecteur de scores de MINT, qui permettait d'augmenter le pouvoir de discrimination entre les cellules hiPSC et hESC, n'excède 0.20. Finalement, le troisième vecteur de scores de MINT qui permet, comme le second vecteur de scores de MINT, de distinguer les cellules hiPSC des cellules hESC, est légèrement corrélé négativement ($r_{(DIABLO_{VS1}, MOFA+_{VS3})} = -0.27$) avec le troisième vecteur de scores de MOFA+.

		MOFA+				
		VS1	VS2	VS3	VS4	VS5
DIABLO	VS1	0.43	0.54	0.36	-0.09	-0.30
	VS2	0.03	0.20	0.08	0.05	0.06
	VS3	-0.01	0.18	-0.27	0.06	0.04

TABLE 6.5: Analyse de la corrélation des vecteurs de scores des méthodes MINT et MOFA+ sur les données Stemformatics

Plusieurs biomarqueurs sont communs aux différentes méthodes . La Table 6.45, s'intéresse aux dix biomarqueurs les plus influents dans la construction du premier vecteur de scores de MINT. Pour chaque biomarqueur, on peut voir le poids associé à cette variable dans la construction de chacun des cinq facteurs de MOFA+, ainsi que son rang, le rang 1 étant attribué à la variable la plus influente au sein de ce vecteur de scores. Pour faciliter la lecture du tableau, les variables qui font à la fois partie des dix biomarqueurs les plus influents dans la construction du vecteur de scores de MINT (VS1) et de MOFA+ (VS1-VS5) sont mis en exergue. On peut s'apercevoir qu'il y a un nombre non négligeable de variables qui sont communes aux dix biomarqueurs les plus influentes du premier vecteur des scores de MINT et de MOFA+ et peuvent être considérées comme des biomarqueurs potentiellement sérieux. Par contre, ceci n'est pas le cas pour les vecteurs de scores de MOFA+ plus élevés (VS4 et VS5).

	MINT		**MOFA+**									
	VS1		**VS1**		**VS2**		**VS3**		**VS4**		**VS5**	
Biomarqueur	**Poids**	**Rang**	**Poids.1**	**Rang.1**	**Poids.2**	**Rang.2**	**Poids.3**	**Rang.3**	**Poids.4**	**Rang.4**	**Poids.5**	**Rang.5**
ENSG00000181449	-0.2096	1	1.0252	4	0.4486	2	0.3198	19	0.0051	354	0.2911	20
ENSG00000176485	-0.2060	2	0.5822	22	0.2628	24	0.3579	8	0.0032	368	0.1014	156
ENSG00000110721	-0.2031	3	0.3772	44	0.1735	49	0.2922	26	0.2403	26	0.2095	46
ENSG00000184697	-0.1905	4	1.2487	1	0.0948	113	0.0181	260	0.2023	45	0.1240	124
ENSG00000123080	0.1899	5	0.9197	5	0.4348	4	0.2818	30	0.2055	40	0.0439	255
ENSG00000102935	-0.1822	6	0.9107	6	0.2402	31	0.0379	222	0.2143	39	0.0604	216
ENSG00000163661	0.1805	7	1.1987	2	0.1855	44	0.3000	20	0.0016	380	0.0040	376
ENSG00000198542	0.1765	8	1.0271	3	0.4193	6	0.1073	135	0.0201	298	0.2388	33
ENSG00000117676	-0.1725	9	0.8133	10	0.0846	128	0.2096	64	0.1666	74	0.0571	225
ENSG00000121057	-0.1597	10	0.3539	51	0.4256	5	0.3261	15	0.0567	211	0.0152	329

FIGURE 6.45: Comparaison des biomarqueurs détectés dans les deux méthodes MINT et MOFA+

Chapitre 7

Conclusions

Aspects généraux

Ce travail a d'abord abordé la question des données omiques et de leurs challenges, ainsi que l'intérêt et l'utilité de l'intégration des données omiques multi-blocs. Cette intégration se fait généralement de manière horizontale ou verticale. Les outils permettant une intégration des données omiques multi-blocs dans les deux sens étant apparus bien plus récemment, cette technique paraît moins utilisée à l'heure actuelle : nous avons trouvé peu de publications sur ce sujet spécifique et aucun jeu de données qui se serait prêté à l'analyse d'une telle intégration de données.

Dans ce travail, trois méthodes ont été expliquées au niveau de leur développement mathématique, algorithmique et de leurs sorties graphiques : DIABLO, MINT et MOFA+. Les deux premières ont été développées par l'équipe de MixOmics. Ces trois méthodes disposent d'un package en R qui leur est dédié (*mixOmics* pour DIABLO et MINT et *MOFA2* pour MOFA+).

Intrinsèquement, les trois méthodes sont assez différentes sur plusieurs aspects :

• DIABLO et MINT sont des méthodes supervisées et MOFA+ est une méthode non-supervisée.

• DIABLO est adaptée à l'intégration verticale et MINT à l'intégration horizontale. MOFA+ supporte les deux types d'intégration, une seule à la fois ou les deux en même temps.

• DIABLO et MINT cherchent à maximiser la corrélation entre des vecteurs de scores. Les corrélations à maximiser sont définies selon une matrice de design et, dans tous les cas, une maximisation de la corrélation entre les vecteurs de scores des blocs et de la variable d'intérêt est attendue. MOFA+, par contre, tend à capturer un maximum de variabilité des

données, en tenant compte de leur structure en blocs. De cette manière, MOFA+ peut être vue comme une généralisation d'une ACP [5].

• L'approche principale de MOFA+ est bayesienne alors que l'approche principale de DIABLO et MINT est plutôt multivariée.

Détection des biomarqueurs

Au niveau de la détection des biomarqueurs, les trois méthodes ont montré des aptitudes intéressantes. Souvent, les biomarqueurs trouvés avec une méthode étaient également retrouvés par l'autre méthode. Ceci tend à montrer une certaine consistance dans les résultats produits. Certains articles scientifiques sur le sujet ont conforté le choix de plusieurs biomarqueurs ainsi trouvés par les méthodes.

Il est toutefois à noter que les jeux de données analysés étaient particulièrement bien sélectionnés pour les méthodes DIABLO et MINT puisqu'ils étaient utilisés dans les vignettes de présentation des deux méthodes par l'équipe MixOmics. Lors d'un workshop organisé en novembre 2021 entre l'équipe de MOFA+ et de MixOmics, le constat a été fait que les outils de MixOmics pouvaient être plus performants pour la découverte des biomarqueurs.

Sorties graphiques

Les sorties graphiques présentées dans ce travail sont celles qui nous ont semblé être les plus utiles pour une analyse basée sur l'intégration de données omiques multi-blocs. La Table 7.1 résume les différentes sorties graphiques ainsi que leur intérêt. Nous avons eu plus de facilité à interpréter les graphiques du package *mixOmics* et ceux-ci nous ont également parus plus utiles que ceux de MOFA+. Ceci aurait probablement été plus véridique encore si les sorties graphiques de MOFA+ n'avaient pas été accompagnée de l'information relative à la variable d'intérêt, comme c'est souvent le cas lors d'analyses non-supervisées.

Dans les trois méthodes, certaines sorties graphiques nous ont semblé redondantes et ne nous ont pas semblé apporter beaucoup d'information complémentaire intéressante ou indispensable : par exemple, il nous semble que la plupart des informations obtenues via la fonction `network()`, utilisée avec un objet de type DIABLO, étaient déjà visibles via le `circosPlot`.

Certaines sorties graphiques nous ont également laissé perplexe sur leur utilité. En particulier les graphiques obtenus via les fonctions `cimDiablo()`, `network()`, `plot_data_scatter()`, `plot_data_heatmap()` et `plot_weights()`. Au contraire, certains graphiques nous ont semblé particulièrement pertinents. Il s'agit notamment des graphiques obtenus avec les fonctions `plotIndiv()`, `plotVar()`, `circosPlot()`, `auroc()` utilisée avec un objet de type MINT, `plot_factor()`, `plot_top_weights()` et `plot_variance_explained()`. Notons que ces différents constats sur les sorties graphiques sont probablement dus à un manque d'expérience professionnelle dans le domaine spécifique de l'intégration des données omiques.

Méthode	Nom de la fonction du graphique	Analyse des individus	Analyse de toutes les variables	Analyse des biomarqueurs	Analyse des blocs	Analyse de **Y**	Analyse des vecteurs de poids	Analyse des vecteurs de scores	Analyse de la variance	Analyse de la performance du modèle
DIABLO	`plotIndiv()`	x			x	x		x		
	`plotDiablo()`	x			x	x		x		
	`plotArrow()`	x			x	x		x		
	`cimDiablo()`	x		x	x	x				
	`plotVar()`			x	x			x		
	`circosPlot()`			x	x	x				
	`plotLoadings()`			x	x	x	x			
	`network()`			x	x					
	`auroc()`					x				x
MINT	`plotIndiv()`	x			x	x		x		
	`plotArrow()` *	x				x		x		
	`cim()`	x		x		x				
	`plotVar()`			x			x			
	`plotLoadings()`			x		x	x			
	`network()`			x		x			x	x
	`auroc()`									x
MOFA+	`plot_factor()`	x						x		
	`plot_data_scatter()`	x	x	x	x				x	
	`plot_data_heatmap()`	x	x	x				x		
	`plot_dimred()`	x								
	`plot_weights()`		x	x	x		x			
	`plot_top_weights()`		x	x	x		x			
	`plot_data_overview()`	x	x		x					
	`plot_factor_cor()`							x		x
	`plot_variance_explained()` [[1]]				x				x	x
	`plot_variance_explained()` [[2]]				x			x	x	x

TABLE 7.1: Résumé des fonctionnalités des sorties graphiques. * Lors de la rédaction de ce travail, la fonction `plotArrow()` était sujette à un bug informatique pour l'utilisation de MINT et nous n'avons pas pu la tester. Ce bug est en cours de traitement par l'équipe MixOmics.

Autres considérations

Nous avons pu constater que, lors de l'utilisation de jeux de données de grande taille, DIABLO et MINT pouvaient mettre plusieurs heures à trouver une solution, alors que MOFA+, grâce à l'utilisation stochastique de son algorithme, a toujours fourni des résultats dans des délais raisonnables (quelques secondes). Il s'agit d'après nous d'un des plus gros avantages de MOFA+ face à une analyse single-cell par exemple. Par contre, pour des jeux de données de taille similaire à ceux utilisés dans ce travail, le temps de calcul des trois méthodes est comparable, même si MOFA+ reste le plus rapide.

Notons que, d'un point de vue de leur facilité d'utilisation, DIABLO et MINT nous ont semblé plus convainquants. Le fait que la plupart des fonctions ont un nom identique pour les deux méthodes facilite l'utilisation de l'analyste. L'importation des données nous a aussi semblé plus aisée que celles de MOFA+. En outre, la documentation disponible sur les sorties graphiques nous a également semblé mieux renseignée avec le package *mixOmics*. Il faut toutefois reconnaître que MOFA+ est plus flexible que DIABLO ou MINT dans le type de structure qu'il peut accepter en input : une liste de matrices, un seul dataframe, un objet de type MultiAssayExperiment, un objet de type Seurat, etc., faisant probablement de cet outil une méthode plus adaptée aux professionnels du domaine de l'intégration des données omiques multi-blocs.

Au cours de la rédaction de ce travail, nous avons eu l'occasion d'interpeller les deux équipes des trois méthodes. Dans les deux cas, nous avons pu constater un support technique rapide et relativement efficace.

Dernières nouvelles de MixOmics et MOFA+...

Nous concluons ce travail en notant que l'équipe de MOFA+ a, depuis lors, décliné un nouvel outil, MEFISTO, permettant encore plus de flexibilité d'un point de vue de l'intégration des données omiques multiples. L'intérêt de ce nouvel outil se situe dans la possibilité d'intégrer des données selon les modalités (blocs verticaux), les études (blocs horizontaux) ou la dimension temporelle (blocs dans une troisième dimension).

Quant à l'équipe de MixOmics, le 9 novembre 2021, elle a publié un livre synthétisant les différentes méthodes qu'elle a développées : *Multivariate Data Integration Using R : Methods and Applications with the mixOmics Package*.

Bibliographie

[1] Herve Abdi, Wynne W Chin, Vincenzo Esposito Vinzi, Giorgio Russolillo, and Laura Trinchera. *New perspectives in partial least squares and related methods*. Springer, 2013.

[2] Altuna Akalin. Computational genomics with r. https ://compgenomr.github.io/book/, September 2020. consulté le : 19/12/2021.

[3] D Craig Allred. Issues and updates : evaluating estrogen receptor-α, progesterone receptor, and her2 in breast cancer. *Modern Pathology*, 23(2) :S52–S59, 2010.

[4] Jérome Ambroise. Rapport d'analyse de données affimetrix. Technical report, UCLouvain, 2019.

[5] Ricard Argelaguet, Damien Arnol, Danila Bredikhin, Yonatan Deloro, Britta Velten, John C Marioni, and Oliver Stegle. Mofa+ : a statistical framework for comprehensive integration of multi-modal single-cell data. *Genome biology*, 21 :1–17, 2020.

[6] Ricard Argelaguet, Britta Velten, Damien Arnol, Sascha Dietrich, Thorsten Zenz, John C Marioni, Florian Buettner, Wolfgang Huber, and Oliver Stegle. Multi-omics factor analysis—a framework for unsupervised integration of multi-omics data sets. *Molecular systems biology*, 14(6) :e8124, 2018.

[7] Matteo Bersanelli, Ettore Mosca, Daniel Remondini, Enrico Giampieri, Claudia Sala, Gastone Castellani, and Luciano Milanesi. Methods for the integration of multi-omics data : mathematical aspects. *BMC bioinformatics*, 17(2) :167–177, 2016.

[8] Brendan Coady. Gradient ascent. https ://medium.com/common-notes/gradient-ascent-e23738464a19, November 2017. consulté le : 12/11/2021.

[9] Lorraine Dawirs. Etude de méthodes d'analyse multibloc pour les données multi-omiques. Master's thesis, UCL, 2017.

[10] Selma Esseghir, Alan Kennedy, Pooja Seedhar, Ashutosh Nerurkar, Richard Poulsom, Jorge S Reis-Filho, and Clare M Isacke. Identification of ntn4, tra1, and stc2 as prognostic markers in breast cancer in a screen for signal sequence encoding proteins. *Clinical cancer research*, 13(11) :3164–3173, 2007.

[11] Thierry Foucart. Colinéarité et régression linéaire. *Mathématiques et sciences humaines. Mathematics and social sciences*, (173), 2006.

[12] Laurent Gatto. Mass spectrometry-based proteomics. Technical report, UCLouvain, 2021.

[13] Alexander Hinneburg and Daniel A Keim. Optimal grid-clustering : Towards breaking the curse of dimensionality in high-dimensional clustering. 1999.

[14] Mingon Kang, Baoju Zhang, Xiaoyong Wu, Chunyu Liu, and Jean Gao. Sparse generalized canonical correlation analysis for biological model integration : A genetic study

of psychiatric disorders. In *2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 1490–1493, 2013.

[15] Kim-Anh Lê Cao, Simon Boitard, and Philippe Besse. Sparse pls discriminant analysis : biologically relevant feature selection and graphical displays for multiclass problems. *BMC bioinformatics*, 12(1) :1–17, 2011.

[16] Hongling Li, Chunjing Bian, Lianming Liao, Jing Li, and Robert Chunhua Zhao. mir-17-5p promotes human breast cancer cell migration and invasion through suppression of hbp1. *Breast cancer research and treatment*, 126(3) :565–575, 2011.

[17] MixOmics. N-integration discriminant analysis with diablo. http ://mixomics.org/mixdiablo/case-study-tcga/, Octobre 2016. consulté le : 13/10/2021.

[18] Kevin P Murphy. *Machine learning : a probabilistic perspective*. MIT press, 2012.

[19] Oliver Stegle Ricard Argelaguet, Danila Bredikhin and John Marioni. Mofa analysis of the chromium single cell multiome atac + gene expression assay. https ://raw.githack.com/bioFAM/MOFA2_tutorials/master/R_tutorials/10x_scRNA_scATAC.html, January 2020. consulté le : 23/12/2021.

[20] Florian Rohart, Aida Eslami, Nicholas Matigian, Stéphanie Bougeard, and Kim-Anh Le Cao. Mint : a multivariate integrative method to identify reproducible molecular signatures across independent experiments and platforms. *BMC bioinformatics*, 18(1) :1–13, 2017.

[21] Amrit Singh, Benoit Gautier, Casey P Shannon, Michaël Vacher, Florian Rohart, Scott J Tebbutt, and Kim-Anh Le Cao. Diablo–an integrative, multi-omics, multivariate method for multi-group classification. *BioRxiv*, page 067611, 2016.

[22] Indhupriya Subramanian, Srikant Verma, Shiva Kumar, Abhay Jere, and Krishanpal Anamika. Multi-omics data integration, interpretation, and its application. *Bioinformatics and biology insights*, 14 :1177932219899051, 2020.

[23] Markus Svensén and Christopher M Bishop. Pattern recognition and machine learning, 2007.

[24] 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.

[25] Katarzyna Tomczak, Patrycja Czerwińska, and Maciej Wiznerowicz. The cancer genome atlas (tcga) : an immeasurable source of knowledge. *Contemporary oncology*, 19(1A) :A68, 2015.

[26] Christophe Vanderaa and Laurent Gatto. Replication of single-cell proteomics data reveals important computational challenges. *Expert Review of Proteomics*, pages 1–9, 2021.

[27] Christine A Wells, Rowland Mosbergen, Othmar Korn, Jarny Choi, Nick Seidenman, Nicholas A Matigian, Alejandra M Vitale, and Jill Shepherd. Stemformatics : visualisation and sharing of stem cell gene expression. *Stem Cell Research*, 10(3) :387–395, 2013.

Annexes

Annexe A

Algorithme NIPALS adapté à la mgPLS

Algorithm 5 NIPALS adapté à la mgPLS

1: Notons $\forall 1 \leq m \leq M, \mathbf{X}_1^{(m)} = \mathbf{X}^{(m)}, \mathbf{Y}_1^{(m)} = \mathbf{Y}^{(m)}$, où les variables de $\mathbf{X}$ et $\mathbf{Y}$ sont centrées et réduites.
2: On choisit une valeur d'initialisation pour $\mathbf{a}_h$ avec $||\mathbf{a}_h||_2 = 1$,
3: **Répéter :**
4: $\quad \mathbf{t}_h^{(m)} \leftarrow \mathbf{X}_h^{(m)} \mathbf{a}_h$ ▷ Vecteur des scores ($\mathbf{X}$) partiel
5: $\quad \mathbf{b}_h^{(m)} \leftarrow (\mathbf{Y}_h^{(m)})' \mathbf{t}_h^{(m)}$ ▷ Vecteur des loadings ($\mathbf{Y}$) partiel
6: $\quad \mathbf{b}_h \leftarrow \frac{\sum_{m=1}^{M} \mathbf{b}_h^{(m)}}{||\sum_{m=1}^{M} \mathbf{b}_h^{(m)}||_2}$ ▷ Vecteur des loadings ($\mathbf{Y}$) global
7: $\quad \mathbf{u}_h^{(m)} \leftarrow \mathbf{Y}_h^{(m)} \mathbf{b}_h$ ▷ Vecteur des scores ($\mathbf{Y}$) partiel
8: $\quad \mathbf{a}_h^{(m)} \leftarrow (\mathbf{X}_h^{(m)})' \mathbf{u}_h^{(m)}$ ▷ Vecteur des loadings ($\mathbf{X}$) partiels
9: $\quad \mathbf{a}_h \leftarrow \frac{\sum_{m=1}^{M} \mathbf{a}_h^{(m)}}{||\sum_{m=1}^{M} \mathbf{a}_h^{(m)}||_2}$ ▷ Vecteur des loadings ($\mathbf{X}$) global
10: **Itérer** depuis l'étape 4 jusqu'à convergence

Annexe B

Equivalances pour MOFA+

Le chapitre *MOFA+* de ce travail est basé sur l'article [5] et de son annexe *Supplementary Methods*. Toutefois, pour faire correspondre les annotations à celles de ce travail, plusieurs abréviations ont dû être renommées. La table ci-dessous donne les équivalances entre les annotations utilisées dans les articles et dans ce travail.

Annotation dans l'article	Annotation dans ce travail	Signification	Remarques
Matrices			
$\mathbf{Y}$	$\mathbf{X}$	Matrice des données originale	
$\mathbf{W}$	$\mathbf{A}$	Matrice des vecteurs de loadings	
$\mathbf{Z}$	$\mathbf{T}$	Matrice des vecteurs de scores	
$\mathbf{X}$	$\mathbf{W}$	Variables latentes	
Vecteurs			
$\mathbf{w}$	$\mathbf{a}$	Vecteur de loadings	
$\mathbf{z}$	$\mathbf{t}$	Vecteur de scores	
Indices et valeurs			
g	m	Indice de l'étude (blocs verticaux)	
m	q	Indice de la modalité (blocs horitontaux)	
k	h	Indice de la composante latente / facteur	
D	P	Nombre de colonnes	
Paramètres			
λ	ψ	Désigne $\tau^{(m,q)}$, $\hat{\mathbf{a}}^{(q)}$, $\alpha^{(q)}$, $\theta^{(q)}$, $\mathbf{s}^{(q)}$, $\alpha^{(m)}$ ou $\theta^{(m)}$	Dans 2.2.4) de *Supplementary Methods*
λ	δ		Dans 2.2.3.a) de *Supplementary Methods*
Annotations			
$\mathbf{X}^q$	$\mathbf{X}^{(q)}$	Désigne le bloc q de la matrice $\mathbf{X}$	
$\mathbf{X}_{ij}$	$\mathbf{X}_{[i,j]}$	Désigne l'élément {i,j} de la matrice $\mathbf{X}$	Le document mentionne que leur annotation devrait correspondre à celle utilisée dans ce travail, mais ce n'est pas le cas.
$\mathbf{v}^{(t)}$	$\mathbf{v}_i$	Désigne l'itération i du vecteur $\mathbf{v}$	