Université de Lille 1 SCIENCES ET TECHNOLOGIES

Master 1 MAS
Année 2017-2018

Travaux Encadrés de Recherche

Master 1 ISN

Modélisation des relations quantitatives structure à activité

Étudiants :
WAEYTENS Emilie
HEYSE Wilfried

Encadrants :
DERMOUNE Azzouz
TRAN Viet Chi

Remerciements

Nous adressons nos remerciements aux personnes qui nous ont aidés dans la réalisation de ce travail encadré de recherche portant sur les données de l'hépatite C.

En premier lieu, nous tenons à remercier M. Viet Chi TRAN et M. Azzouz DERMOUNE, nos tuteurs sur ce projet, pour nous avoir guidé dans notre travail, et pour nous avoir permis de trouver des solutions rapidement à nos difficultés.

Nous remercions également M. CHERQUAOUI, professeur de chimie à l'Université Caddi Ayyad qui nous a fourni les données sur lesquelles nous avons travaillé, ainsi que M. Ismail HDOUFANE, thésard en Chimie, à l'Université Caddi Ayyad, qui a su nous conseiller dans ce projet.

Table des matières

Remerciements **1**

Introduction **5**

1 Méthodes de sélection de variables **10**
1.1 Méthodes de sélection de variables pas à pas 10
1.1.1 Backward 10
1.1.2 Forward 10
1.1.3 Stepwise 11
1.2 LASSO 12
1.2.1 Présentation de la méthode 12
1.2.2 Mise en œuvre 17
1.3 Analyse en composantes principales 19
1.3.1 Idée générale 19
1.3.2 Concept 19
1.4 Méthodes de classification des données 22
1.4.1 Classification hiérarchique ascendante 22
1.4.2 Méthode K-means 23
1.5 Indicateurs 24

2 Nos fonctions **25**
2.1 Création des familles 25
2.1.1 Création des familles en se basant sur les noms 25
2.1.2 Création des familles à partir de la classification 29
2.1.3 Familles du spécialiste 31
2.2 Autres Fonctions 33

3 Application et Résultats **35**
3.1 Table de données initiales 36
3.1.1 ACP sur la table brute 36
3.1.2 Résulats de la sélection par LASSO et par Stepwise sur toute la table . . 38
3.2 Chefs de familles 40
3.2.1 Nos familles 40
3.2.2 Familles du spécialiste 41
3.2.3 Familles issues de classification 41
3.3 ACP sur les familles : Composantes principales 44
3.3.1 Nos familles 44
3.3.2 Familles du spécialiste 45
3.3.3 Familles issues de classification 45

3.4 ACP sur les familles : Variables à forte contribution 47
3.4.1 Nos familles 47
3.4.2 Familles du spécialiste 48
3.4.3 Familles issues de classification 48
3.5 Récapitulatif 51

Conclusion **52**

Bibliographie **55**

Annexe **56**

Introduction

Une 'relation quantitative structure à activité' (QSAR) décrit la façon dont une structure chimique est corrélée à un effet bien déterminé comme l'activité ou la réactivité chimique. La QSAR la plus commune exprime l'activité comme une fonction des propriétés physico-chimiques et/ou structurales. Nous disposons de données réelles concernant les bases moléculaires des inhibiteurs du VHC (Virus hépatite C). Le but de ce TER est de trouver un modèle permettant d'expliquer au mieux l'activité chimique du virus, en sélectionnant les variables apportant le plus d'information sur cette activité. Nous cherchons à créer un modèle, qui permettrait de prédire correctement l'activité chimique.

Ces données sont structurées dans un tableau Excel. La table de données du VHC comporte 115 lignes, représentant les individus sur lesquels ont été prélevés les données, et 5255 colonnes représentant les différentes variables. La première colonne (COMPOUND) est l'identifiant de l'individu, la deuxième représente l'activité chimique du VHC. Les autres colonnes sont des variables quantitatives, et représentent des données chimiques, par exemple MW mesure la masse moléculaire, et nN mesure le nombre d'atomes Nitrogen (Azote). On dispose également d'un fichier Excel récapitulant les spécificités des prélèvements : les données viennent de trois études effectuées dans différents laboratoires.

Le but de ce Travail Encadré de Recherche est de trouver un modèle permettant d'expliquer au mieux l'activité chimique de ce virus à l'aide des statistiques. Pour cela, nous avons d'abord effectué une analyse descriptive des données, c'est-à-dire étudié le comportement de la variable que l'on cherche à expliquer (l'activité chimique), que l'on va appeler Y, ainsi que des mesures concernant les variables explicatives, comme la moyenne des variables, leurs corrélations ...

Nettoyage de la table

En regardant comment la table est organisée, nous avons repéré des données manquantes. A l'aide de la fonction `RechercheNA` (voir Annexe), nous avons localisé puis supprimé ces données manquantes car elles sont centrées sur deux colonnes et quatre lignes. Elles sont situées plus précisément sur les colonnes "Psi_e_1d" , "Psi_e_1s", puis sur les lignes "13_3b", "13_3f", "14_3i" et "14_k" (dans les colonnes "AMR", "ALOGP", "ALOGP2" et "CMC-80" à "Infective-50").

De plus, 2061 variables de la table apparaissent totalement constantes, c'est-à-dire prennent une seule valeur, quelque soit l'individu. Par exemple, la variable "nB" est constante à 0, et "Rbrid" constante à 1. Ces variables n'apportent donc pas d'information supplémentaire sur l'activité chimique. De ce fait, la matrice X^TX, nécessaire pour effectuer un régression linéaire multiple, n'est pas inversible car les variables sont linéairement liées entre elles. On ne peut donc pas analyser ces données. On élimine ces variables de la table grâce à la fonction `VarianceNulle`

(voir Annexe), celle-ci comporte alors 111 lignes et 3194 variables après nettoyage.

Statistique descriptive sur Y

L'étape suivante de notre travail fut d'analyser le comportement de Y. On trouvera en Figure 1 un histogramme récapitulant la répartition des valeurs prises par la variable, et en Figure 2 les statistiques de bases de l'activité chimique, comme le moyenne, l'écart-type, la variance, et la médiane sont donnés.

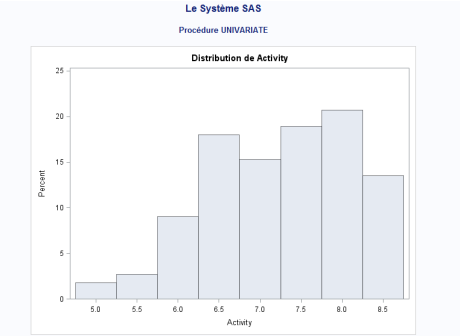

FIGURE 1 – Histogramme de l'activité chimique

Mesures statistiques de base			
Emplacement		Variabilité	
Moyenne	7.258780	Ecart-type	0.88401
Médiane	7.397940	Variance	0.78147
Mode	8.397940	Intervalle	3.69897
		Ecart interquartile	1.30103

FIGURE 2 – Statistiques de l'activité chimique

Ensuite, nous avons sélectionné quelques variables à étudier, pour avoir un aperçu de la table, c'est-à-dire voir quelles valeurs elles pouvaient prendre, ainsi que la corrélation entre les variables. On souhaiterait que certaines variables soient corrélées entre elles car cela nous permettrait de les regrouper, et d'en éliminer car l'apport d'information de plusieurs variables fortement corrélées entre elles pourrait être plus faible que des variables peu corrélées les unes des autres. Sur la Figure 3, on donne certaines statistiques de bases de quelques variables qui nous sont apparues régulièrement par la suite dans les résultats de régression, ou ensemble dans nos regroupements de variables. Enfin, la Figure 4 représente une matrice des corrélations pour ces variables.

On peut déduire de ces deux tableaux que les variables n'ont pas forcement les mêmes amplitudes et ordres de grandeurs ce qui sera certainement problématique dans la suite de notre travail. Ensuite, sur la Figure 4, il y a également les résultats des tests de corrélation. Ce test consiste à tester l'hypothèse nulle, c'est-à-dire que la corrélation entre deux variables est nulle. Une p-valeur petite (inférieure à 0.05) signifie que l'hypothèse nulle est rejeté, et donc que les variables sont significativement corrélées. On remarque donc que plusieurs variables ne sont pas indépendantes les unes des autres. Par exemple, les variables "AMW" et "MW" sont fortement corrélées entre elles, contrairement aux variables "MATS5p" et "Se". Par la suite, nous essaierons de trouver une méthode permettant de supprimer la corrélation, ainsi que les différences de variance et d'échelle des variables.

Procédure CORR

11 Variables :	AMW MW Sv Se Sp Si ZM1MulPer ATS8p MATS2i MATS5p CATS3D_17_LL

Statistiques simples

Variable	N	Moyenne	Ecart-type	Somme	Minimum	Maximum	Libellé
AMW	111	9.19799	0.63653	1021	7.47700	10.39900	AMW
MW	111	491.89216	29.55795	54600	412.53000	547.55000	MW
Sv	111	36.22248	1.67062	4021	30.87100	39.65500	Sv
Se	111	55.62226	2.87924	6174	46.67600	62.63200	Se
Sp	111	36.67605	1.77885	4071	31.17600	41.36400	Sp
Si	111	61.33814	3.36679	6809	51.12700	69.22200	Si
ZM1MulPer	111	699.78879	89.14165	77677	465.87100	860.11700	ZM1MulPer
ATS8p	111	3.90174	0.12647	433.09300	3.61100	4.18200	ATS8p
MATS2i	111	0.15641	0.07909	17.36100	-0.11600	0.31900	MATS2i
MATS5p	111	0.04523	0.06737	5.02100	-0.06300	0.17100	MATS5p
CATS3D_17_LL	111	0.02703	0.16290	3.00000	0	1.00000	CATS3D_17_LL

FIGURE 3 – Statistiques de certaines variables explicatives

Coefficients de corrélation de Pearson, N = 111
Proba > |r| sous H0: Rho=0

	AMW	MW	Sv	Se	Sp	Si	ZM1MulPer	ATS8p	MATS2i	MATS5p	CATS3D_17_LL
AMW	1.00000	0.63933	-0.07597	-0.35897	-0.42953	-0.47711	0.85478	-0.27854	0.52418	0.51628	-0.32457
AMW		<.0001	0.4281	0.0001	<.0001	<.0001	<.0001	0.0031	<.0001	<.0001	0.0005
MW	0.63933	1.00000	0.69839	0.48515	0.36862	0.36738	0.90981	0.41713	0.48916	0.49731	-0.09581
MW	<.0001		<.0001	<.0001	<.0001	<.0001	<.0001	<.0001	<.0001	<.0001	0.3172
Sv	-0.07597	0.69839	1.00000	0.91962	0.91150	0.87512	0.36671	0.87742	0.05481	0.31403	0.19878
Sv	0.4281	<.0001		<.0001	<.0001	<.0001	<.0001	<.0001	0.5678	0.0008	0.0365
Se	-0.35897	0.48515	0.91962	1.00000	0.91335	0.99086	0.14226	0.79703	0.02796	-0.02437	0.26200
Se	0.0001	<.0001	<.0001		<.0001	<.0001	0.1364	<.0001	0.7708	0.7996	0.0055
Sp	-0.42953	0.36862	0.91150	0.91335	1.00000	0.92113	-0.03610	0.90113	-0.23611	0.14394	0.35388
Sp	<.0001	<.0001	<.0001	<.0001		<.0001	0.7068	<.0001	0.0126	0.1318	0.0001
Si	-0.47711	0.36738	0.87512	0.99086	0.92113	1.00000	0.01296	0.78698	-0.05182	-0.09226	0.30101
Si	<.0001	<.0001	<.0001	<.0001	<.0001		0.8926	<.0001	0.5891	0.3355	0.0013
ZM1MulPer	0.85478	0.90981	0.36671	0.14226	-0.03610	0.01296	1.00000	0.07322	0.64401	0.47713	-0.27463
ZM1MulPer	<.0001	<.0001	<.0001	0.1364	0.7068	0.8926		0.4450	<.0001	<.0001	0.0035
ATS8p	-0.27854	0.41713	0.87742	0.79703	0.90113	0.78698	0.07322	1.00000	-0.15588	0.30599	0.27571
ATS8p	0.0031	<.0001	<.0001	<.0001	<.0001	<.0001	0.4450		0.1023	0.0011	0.0034
MATS2i	0.52418	0.48916	0.05481	0.02796	-0.23611	-0.05182	0.64401	-0.15588	1.00000	0.09417	-0.31556
MATS2i	<.0001	<.0001	0.5678	0.7708	0.0126	0.5891	<.0001	0.1023		0.3255	0.0007
MATS5p	0.51628	0.49731	0.31403	-0.02437	0.14394	-0.09226	0.47713	0.30599	0.09417	1.00000	-0.10993
MATS5p	<.0001	<.0001	0.0008	0.7996	0.1318	0.3355	<.0001	0.0011	0.3255		0.2507
CATS3D_17_LL	-0.32457	-0.09581	0.19878	0.26200	0.35388	0.30101	-0.27463	0.27571	-0.31556	-0.10993	1.00000
CATS3D_17_LL	0.0005	0.3172	0.0365	0.0055	0.0001	0.0013	0.0035	0.0034	0.0007	0.2507	

FIGURE 4 – Corrélations de certaines variables explicatives

La Figure 5 indique la corrélation des variables sélectionnées au hasard avec Y. On aimerait que les variables soient fortement corrélées avec Y et donc que les p-valeurs du test de corrélation soient petites, et ainsi pouvoir expliquer au mieux la variable Y. La variable "MATS5p" est fortement corrélées avec Y, alors que la variable "Sp" l'est très peu. En représentant graphiquement Y en fonction de ces deux variables, respectivement Figure 6 et Figure 7, on voit bien une tendance pour "MATS5p" alors que pour "Sp", le nuage est diffus et ne semble pas admettre de tendance particulière.

Coefficients de corrélation de Pearson, N = 111 Proba > |r| sous H0: Rho=0												
	Activity	AMW	MW	Sv	Se	Sp	Si	ZM1MulPer	ATS8p	MATS2i	MATS5p	CATS3D_17_LL
Activity	1.00000	0.39450	0.35068	0.17192	-0.04844	0.03088	-0.09995	0.37512	0.14456	0.04863	0.60636	-0.25040
Activity		<.0001	0.0002	0.0712	0.6137	0.7477	0.2966	<.0001	0.1301	0.6122	<.0001	0.0080

FIGURE 5 – Corrélations entre les variables et Y

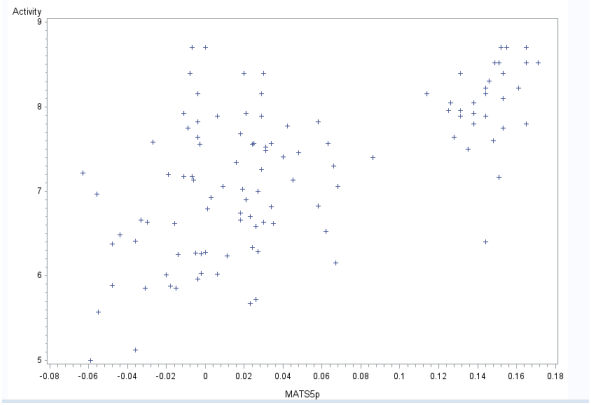

FIGURE 6 – Graphique de Y en fonction de MATS5p

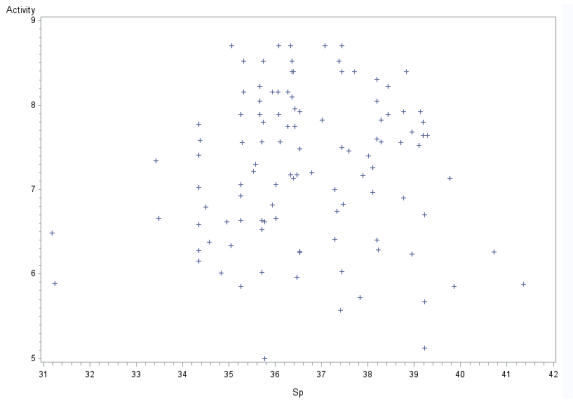

FIGURE 7 – Graphique de Y en fonction de Sp

Contexte

Le but de notre TER est donc de trouver un modèle expliquant l'activité chimique du VHC en fonction des variables explicatives. Nous cherchons à construire un modèle de régression linéaire qui peut être écrit de la manière suivante :

$$Y_i = \sum_{j=1}^{p} X_{ij}\beta_j + \epsilon_i$$

On définit l'écriture matricielle de ce problème comme suit :

$$Y = X\beta + \epsilon$$

où $Y = (Y_1, ..., Y_n)^T \in \mathbb{R}^n$ est le vecteur des n observations de la variable expliquée, $\beta = (\beta_1, ..., \beta_p)^T \in \mathbb{R}^p$ est le vecteur des coefficients que l'on cherche à estimer, $\epsilon = (\epsilon_1, ..., \epsilon_n)^T \in \mathbb{R}^n$ est le terme d'erreur et $X = (X_{.1}, ..., X_{.p}) \in \mathcal{M}_{n\times p}$ est la matrice dans laquelle on a rangé en colonne les n observations de chaque variable explicative $X_{.j}$.

Dans un cas de régression standard, c'est-à-dire où on dispose d'une base de données avec un nombre p de variables explicatives inférieur au nombre n d'individus, pour estimer β on utilise la méthode des moindres carrés ordinaires qui consiste à trouver $\widehat{\beta}$ tel que :

$$\widehat{\beta}_{MCO} = \underset{\beta\in\mathbb{R}^p}{\arg\min}\left(\|Y - X\beta\|_2^2\right) \tag{1}$$

Alors, sous l'hypothèse que ϵ est de moyenne nulle et de variance $\sigma^2 I_n$, on a que :

— $\widehat{\beta}_{MCO} = (X^T X)^{-1} X^T Y$

— $\widehat{\beta}_{MCO}$ est un estimateur sans biais de variance $\sigma^2(X^T X)^{-1}$ où $\sigma^2 = \dfrac{\epsilon^T \epsilon}{n - p - 1}$

— $\widehat{\beta}_{MCO} \sim \mathcal{N}(\beta, \sigma^2(X^T X)^{-1})$

Cependant, les tables que l'on souhaite étudier sont dites de grande dimension. On définit par problèmes statistiques en grande dimension le fait que l'on ait à disposition une base de données avec un nombre p de variables explicatives qui soit beaucoup plus grand que la taille de l'échantillon n : on note $p \gg n$. Le problème qui se pose lors de l'étude de tels problèmes est que la méthode usuelle des moindres carrés ordinaire ne peut plus être mise en place. Effectivement, la matrice $X^T X$ utilisée pour calculer l'estimateur $\widehat{\beta}$ de β devient non inversible puisque les colonnes sont forcément algébriquement liées, et donc le rang de la matrice $X^T X$ est inférieur à p, ce qui entraîne que l'estimateur devient non unique. On pourrait vouloir tester tous les sous-groupes de variables explicatives possibles et prendre celui qui donne le meilleur modèle cependant cela représente 2^p combinaisons possible. Il n'est donc pas envisageable de tester un à un tous les sous-groupes. Il faut donc trouver un moyen de sélectionner un nombre q de variables tel que l'on ait $q < n$ voir même $q \ll n$ et que l'on puisse construire à partir des variables sélectionnées un modèle valide et qui soit suffisamment convaincant. L'idée qu'un tel modèle puisse être construit repose principalement sur l'hypothèse que β est creux, c'est-à-dire qu'il existe un grand nombre de j tels que $\beta_j = 0$.

Modélisation

Après ces premières analyses, notre objectif est donc de trouver une méthode permettant d'aboutir à un modèle qui expliquerait l'activité chimique le mieux possible, avec le moins de variables possibles. La question principale qui se pose est : Est-ce que les statistiques apportent une aide concernant la compréhension de l'activité chimique ?

Dans une première partie, nous avons commencé par étudier les différentes méthodes de sélection de variables : les méthodes fonctionnant pas à pas, c'est-à-dire les méthodes forward, backward et stepwise, puis la méthode LASSO bien adaptée à notre problème de grande dimension, ensuite nous avons résumé l'Analyse en Composantes Principales qui nous permettra de créer de nouvelles variables non corrélées entre elles, pour enfin expliquer le fonctionnement des méthodes de classification. Nous discuterons également des différents indicateurs de comparaison des modèles obtenus.

Dans une deuxième partie, nous avons appliqué ces différentes méthodes à nos données. Pour cela, nous avons créé plusieurs fonctions dont vous trouverez les explications au Chapitre 2. Nous avons donc d'abord commencé par les données brutes, c'est-à-dire la table de 111 individus et 3194 variables. Ensuite nous avons créé des familles de variables, basées sur les noms des variables, et grâce à la classification. Ces familles nous ont permis de sélectionner au sein de ces groupes des représentants via diverses méthodes. Enfin, nous avons utilisé les familles du spécialiste qui travaille également sur ces données, M. Ismail HDOUFANE, thésard en Chimie à l'Université Caddi Ayyad, que nous avons obtenu après avoir constitué nos familles, dans le but de comparer nos résultats avec les siens.

Le but final serait donc d'obtenir des résultats meilleurs que ceux correspondant aux données brutes, et aussi bons que ceux obtenus par le spécialiste, et également d'obtenir une amélioration des résultats grâce à l'application de méthodes statistiques.

Chapitre 1

Méthodes de sélection de variables

1.1 Méthodes de sélection de variables pas à pas

Nous cherchons donc une méthode permettant d'obtenir un modèle ne contenant que des variables significatives, c'est-à-dire que chaque variable retenue a un coefficient significatif sur la variable Y, ici l'activité. Le but de la sélection de variables est de choisir les variables les plus pertinentes pour expliquer la variable Y, et ainsi d'avoir des bonnes prédiction de Y. Le modèle alors obtenu nous permet de comprendre la corrélation entre les variables explicatives et Y.

On peut d'abord penser à retirer toutes les variables non significatives en une fois, mais une variable non significative peut toutefois avoir un effet sur la significativité des autres variables. Cette solution ne semble donc pas appropriée, et ne donnerait donc pas obligatoirement le meilleur modèle possible, étant donné que certaines variables pourraient être éliminées du modèle à tort. Il faut donc procéder pas à pas.

1.1.1 Backward

La méthode Backward consiste tout d'abord à réaliser une régression linéaire multiple sur l'ensemble des variables disponibles. On fixe un seuil de sortie α_S, à partir duquel on considère qu'une variable est non significative, par exemple 5%. Si le modèle est globalement significatif, c'est-à-dire que le test de Fisher sur les coefficients est rejeté, on teste la significativité de chaque variable indépendamment les unes des autres via un test de Fischer. Si chacune des variables apparaît avoir une action sur Y, c'est-à-dire que toutes les p-valeurs sont inférieures au alpha fixé, alors on sélectionne le modèle complet. Sinon, on retire de la régression la variable la moins significative (celle avec la p-valeur la plus forte), et on recommence le mécanisme jusqu'à l'obtention d'un modèle ne contenant que des variables significatives. Nous avons essayé la méthode Backward sur nos données brutes, et celle-ci ne fonctionne pas, car l'hypothèse d'inversibilité de la matrice X^TX n'est pas vérifiée.

1.1.2 Forward

La méthode Forward procède à l'inverse de celle Backward, elle consiste à faire une sélection successives des variables significatives. On fixe également un seuil d'entrée α_E dans le modèle, à partir duquel on considère que la variable est significative. On commence par choisir la variable ayant la plus petite p-valeur ou la plus grande statistique de test de Fisher, c'est-à-dire celle qui pourrait le mieux expliquer Y. Ensuite, on choisit parmi les variables restantes celle ayant la meilleure p-valeur au test de Fisher, et on l'ajoute au modèle, et on recommence le mécanisme jusqu'à ce qu'aucune p-valeur ne soit en dessous du seuil d'entrée.

1.1.3 Stepwise

La procédure de régression Stepwise est une procédure de sélection de variables pas à pas pour les modèles de régression linéaire multiple. Le principe général de cette procédure est de construire un modèle de régression en ajoutant, à chaque étape, la variable considérée comme la plus significative, puis en regardant l'impact de l'ajout de cette variable au modèle sur la significativité des autres variables. Alors, si la significativité d'une variable a subitement chuté, cette variable est retirée du modèle. On procède ainsi de suite jusqu'à ce qu'on ne puisse plus ni ajouter ni retirer de variables.

La procédure se déroule donc comme suit : il faut d'abord définir deux niveaux de significativité : alpha-pour-entrer α_E et alpha-pour-sortir α_S. La première étape consiste à chercher le modèle de régression simple le plus pertinent au sens où il minimise la p-valeur du test de Fisher (qui teste la nullité du coefficient de la variable). Si cette p-valeur est supérieure à α_E alors aucune variable ne peut entrer dans le modèle et la procédure ne démarrera pas. Dans le cas contraire on fait entrer cette variable dans le modèle. Supposons que nous sommes à l'étape k et que nous disposons maintenant d'un modèle avec ℓ variables $(X_{k_1}, ..., X_{k_\ell})$. Si on dispose de p variables au total, on effectue alors $p - \ell$ régressions linéaire multiples à $\ell + 1$ variables sur les variables qui ne sont pas dans le modèle : $(X_{k_1}, ..., X_{k_\ell}, X_{k_{\ell+1}})$ avec $k_{\ell+1} \notin \{k_1, ..., k_\ell\}$ et on regarde la variable parmi $\{X_{k_{\ell+1}},\ k_{\ell+1} \notin \{k_1, ..., k_\ell\}\}$ pour laquelle la p-valeur du test de Fisher est minimale. Si cette p-valeur est supérieure à α_E alors aucune nouvelle variable n'entre dans le modèle. Dans le cas contraire on fait entrer cette variable dans le modèle. Que l'on ait ajouté une nouvelle variable ou pas, on regarde alors quelle est la variable pour laquelle la p-valeur du test de Fisher est la plus forte, si cette p-valeur est supérieure à α_S alors on sort cette variable du modèle et on recommence jusqu'à ce qu'il n'y ait plus de variables à retirer puis on passe à l'étape suivante. La procédure s'arrête quand, à une étape, on ne retire ni n'ajoute aucune variable.

L'avantage de cette procédure est qu'elle prend en compte la corrélation des variables et regarde à chaque étape l'évolution de la significativité des variables, et donc n'avance pas aveuglement comme les procédures Backward et Forward. De plus le choix des valeurs de α_E et de α_S permet de rendre la procédure plus ou moins souple à l'entrée et à la sortie de variables. En effet, en prenant un α_E petit, les variables auront « plus de difficultés » à entrer dans le modèle qu'avec un α_E grand, de même, en prenant un α_S petit, les variables auront « plus de facilités » à sortir du modèle qu'avec un grand α_S.

Ces trois procédures peuvent être adaptés à d'autres critères que les p-valeurs des tests de Fisher. En effet, on peut aussi vouloir, à chaque étape, maximiser la valeur de R^2 ou minimiser la valeur des coefficients AIC ou BIC (respectivement Critère d'Information d'Akaike et Critère d'Information Bayésienne). Le AIC permet de pénaliser les modèles en fonction du nombre de paramètres, et le critère BIC en fonction de la taille de l'échantillon et du nombre de paramètres.

Si les différentes méthodes convergent vers le même modèle final, on peut alors penser que le modèle obtenu est le meilleur possible, c'est à dire celui qui explique le mieux Y, et dont toutes les variables ont une action réelle sur l'activité.

1.2 LASSO

1.2.1 Présentation de la méthode

Le LASSO, pour Least Absolute Shrinkage and Selection Operator, introduit par Tibshirani [8], est une méthode de régression pénalisée visant à réduire le nombre de variables explicatives quand celui-ci est trop important. L'idée du LASSO se base sur l'hypothèse de parcimonie de β, en effet, le LASSO va "forcer" le $\widehat{\beta}$ qu'il va construire à être parcimonieux. Pour quantifier cette parcimonie on peut imaginer utiliser la norme ℓ_0 définie telle que $\|\beta\|_0 = \sum_{i=1}^{p} |\beta_i|^0 = \#\{j : \beta_j \neq 0\}$ cependant pour des problèmes de programmabilité et de propriétés de continuité, on préfère utiliser la norme ℓ_1. Le LASSO va pénaliser la régression par la somme des coefficients de β obtenu en utilisant la norme ℓ_1 comme suit :

$$\widehat{\beta}(\lambda) = \underset{\beta \in \mathbb{R}^p}{\arg\min} \Big(\frac{1}{n} \|Y - X\beta\|_2^2 + \lambda \|\beta\|_1 \Big) \tag{1.1}$$

avec $\|\beta\|_1 = \sum_{i=1}^{p} |\beta_i|$, $\|Y - X\beta\|_2^2 = \sum_{i=1}^{p} \big(Y_i - (X\beta)_i\big)^2$ et $\lambda \geq 0$.

Le paramètre λ peut-être vu comme un paramètre de contrôle de la puissance de régularisation. En effet, quand $\lambda = 0$ alors on retrouve la méthode des moindres carrés ordinaires et quand $\lambda = +\infty$ alors $\beta = 0_{\mathbb{R}^p}$.

Proposition 1. *Il y a équivalence entre (1.1) et :*

$$\widehat{\beta}_{primal}(R) = \underset{\beta \in \mathbb{R}^p ; \|\beta\|_1 \leq R}{\arg\min} \Big(\frac{1}{n} \|Y - X\beta\|_2^2 \Big) \tag{1.2}$$

Et on peut établir une relation entre λ dans (1.1) et R dans (1.2) qui dépendra des données X et Y.

Idée de la démonstration
Nous allons montrer que les problèmes de minimisation sont équivalents. On sait grâce à l'optimisation convexe que la résolution d'un problème de maximisation/minimisation est équivalente à la résolution de son dual. Nous allons montrer que ces deux problèmes sont duals l'un de l'autre. Soit R fixé, considérons

$$(\mathcal{P}) \quad \min_{\beta \in P} f(\beta) \quad \text{avec } f(\beta) = \|Y - X\beta\|_2^2 \quad \text{et} \quad P = \{\beta \in \mathbb{R}^p : \|\beta\|_1 - R \leq 0\}$$

Vérifions que ce problème est un problème d'optimisation convexe. On a bien $f(\beta)$ convexe car $\nabla^2 f(\beta) = X^T X$ est bien symétrique positif et P est bien convexe car c'est la boule de rayon R de la norme ℓ_1.

Ce problème est donc bien un problème d'optimisation convexe, son dual s'écrit :

$$(\mathcal{D}) \quad \max_{\lambda \geq 0} \omega(\lambda) \quad \text{avec } \omega(\lambda) = \min_{\beta} \big(\|Y - X\beta\|_2^2 + \lambda(\|\beta\|_1 - R) \big)$$

On sait que si la condition de Slater est verifiée : $\exists\ \beta\ :\ \|\beta\|_1 - R < 0$ alors le saut de dualité est nul. C'est-à-dire que $f(\beta^*) = \omega(\lambda^*)$ pour β^* et λ^* vérifiant respectivement $(\mathcal{P})$ et $(\mathcal{D})$. Ici, la condition de Slater est vérifiée pour $\beta = 0$ donc on sait que les problèmes $(\mathcal{P})$ et $(\mathcal{D})$ sont équivalents.

On a donc que :

$$(\mathcal{P}) \quad \min_{\beta \in P} \|Y - X\beta\|_2^2 = \max_{\lambda \geq 0} \Big(\min_{\beta} \big(\|Y - X\beta\|_2^2 + \lambda(\|\beta\|_1 - R) \big) \Big) \quad (\mathcal{D})$$

Notons :

$$(\mathcal{D}_C) \quad \max_{0 \leq \lambda \leq C} \Big(\min_{\beta ; \|\beta\|_1 \leq C} \big(\|Y - X\beta\|_2^2 + \lambda(\|\beta\|_1 - R) \big) \Big)$$

Ce problème est équivalent à $\mathcal{D}$ lorsque $C \to +\infty$. Les contraintes étant maintenant convexes compactes, nous allons vérifier que l'on peut appliquer le théorème du min-max. Posons $g(\lambda, \beta) = \|Y - X\beta\|_2^2 + \lambda(\|\beta\|_1 - R)$ et vérifions que $g(\lambda, \beta)$ est bien convexe pour λ fixé et concave pour β fixé.
Pour λ fixé, le fonction $g(\lambda, \cdot)$ est convexe car c'est une addition de deux fonction convexes. Pour β fixé, le fonction $g(\cdot, \beta)$ est concave car c'est un fonction de $\mathbb{R} \to \mathbb{R}$ qui est affine donc elle est à la fois convexe et concave.
Le théorème de min-max nous indique donc que :

$$\begin{aligned}(\mathcal{D}_C) \quad & \max_{0 \leq \lambda \leq C} \Big(\min_{\beta ; \|\beta\|_1 \leq C} \big(\|Y - X\beta\|_2^2 + \lambda(\|\beta\|_1 - R) \big) \Big) \\ & = \min_{\beta ; \|\beta\|_1 \leq C} \Big(\max_{0 \leq \lambda \leq C} \big(\|Y - X\beta\|_2^2 + \lambda(\|\beta\|_1 - R) \big) \Big)\end{aligned}$$

On a donc que, pour un λ_R dépendant de R,

$$(\mathcal{D}_C) \min_{\beta ; \|\beta\|_1 \leq C} \big(\|Y - X\beta\|_2^2 + \lambda_R(\|\beta\|_1 - R) \big)$$

qui est équivalent, comme R est fixé, à :

$$(\mathcal{D}_C) \min_{\beta ; \|\beta\|_1 \leq C} \big(\|Y - X\beta\|_2^2 + \lambda_R \|\beta\|_1 \big)$$

On a donc bien une équivalence entre les deux problèmes (1.1) et (1.2) avec une relation deux à deux entre les λ dans (1.1) et les R dans (1.2).

□

Cette forme, plus intuitive du problème permet de comprendre que le LASSO va empêcher les coefficients de $\widehat{\beta}$ d'être grands. De plus, à cause de la géométrie ℓ_1, le LASSO va sélectionner des variables en ce sens qu'il va attribuer à certaines variables le coefficient 0. Cela est dû au fait qu'en géométrie ℓ_1 les boules sont en réalité des "carrés" (du moins dans $\mathbb{R}^2$), on peut observer ce phénomène sur la Figure 1.1. Sur cette figure, les courbes de niveau (ovales) de la fonction $f(\widehat{\beta})$ avec $\widehat{\beta}$ la solution de 1.1 et le carré représente la boule de centre 1 et de rayon R en norme ℓ_1. On constate que les courbes de niveau 'touchent' la boule ℓ_1 au niveau d'un angle c'est-à-dire là où le coefficient β_1 vaut 0. Cet exemple simple permet de comprendre pourquoi le LASSO assigne certains coefficients exactement à 0.

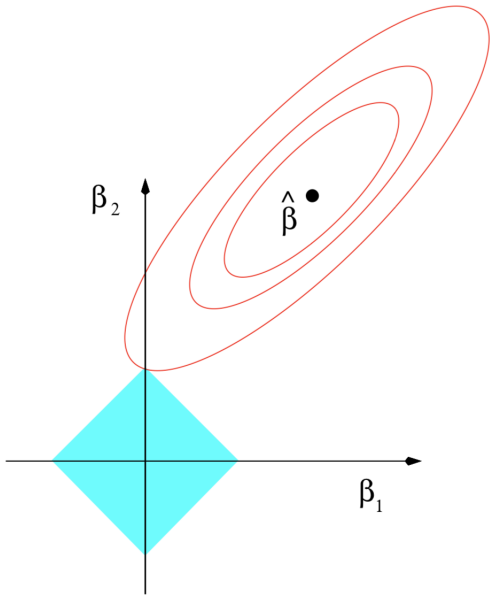

FIGURE 1.1 – Exemple de LASSO en dimension 2

Le problème qui se pose maintenant est de choisir un λ pertinent, qui ne soit ni trop grand, ni trop petit pour que $\widehat{\beta}$ soit proche de β en norme ℓ_1.

D'après le corrolaire 6.1 de Bülhmann et Van de Geer [1] on a que en raisonnant asymptotiquement sur la valeur de n, la théorie nous donne que

$$\|\beta\|_1 = o\left(\sqrt{\frac{n}{log(p)}}\right)$$

Alors, en prenant $\lambda \approx \sqrt{\frac{log(p)}{n}}$ on obtient que $\|\widehat{\beta}(\lambda) - \beta\|_1 \underset{n\to+\infty}{\longrightarrow} 0$.

Une caractérisation importante de $\widehat{\beta}$ est donnée par un résultat d'optimisation convexe, le théorème KKT (Karush-Kuhn-Tucker) :

Proposition 2. *Par la proposition 1 on à équivalence entre 1.1 et 1.2, on va donc considérer la fonction* $f(\beta) = \frac{1}{n}\|Y - X\beta\|_2^2$. *Son gradient est donné par* $\nabla f(\beta) = \frac{-2}{n}X^T(Y - X\beta)$. *Alors un condition nécessaire et suffisante pour que* $\widehat{\beta}$ *soit solution de (1.2) et donc de (1.1) est :*

$$\nabla f(\beta)_j = -sign(\beta_j) \times \lambda, \ \textit{si } \beta_j \neq 0,$$
$$|\nabla f(\beta)_j| \leq \lambda, \ \textit{si } \beta_j = 0.$$

Idée de la démonstration

Pour donner une idée de la démonstration de la proposition, nous allons nous placer dans le cas à 1 dimension qui est généralisable au cas multidimensionnel. Considérons le problème $\min_{\beta}(y - x\beta)$ alors selon le théorème de KKT une condition nécessaire et suffisante pour que β soit solution de ce problème est que le gradient du Lagrangien doit être nul. Pour notre problème cela se traduit en :

$$\nabla L(\beta, \lambda) = 0 \quad \text{avec} \quad L(\beta, \lambda) = \|y - x\beta\|_2^2 + \lambda\|\beta\|_1^2 = (y - x\beta)^2 + \lambda|\beta|$$

Dans le cas où $\beta \neq 0$ alors on a :

$$\begin{aligned} &\nabla L(\beta, \lambda) = 2(y - x\beta) + \lambda \times sign(\beta) = 0 \\ \Leftrightarrow\ &2(y - x\beta) = -\lambda \times sign(\beta) \end{aligned}$$

Ce qui est la première condition de la proposition.

Dans le cas contraire, si $\beta = 0$ alors $L(\beta, \lambda)$ n'est pas différentiable en $\beta = 0$, on s'intéresse donc au sous-différentiel. Le sous-différentiel d'une fonction f en β_0 est noté et défini de la façon suivante :

$$\partial f(\beta_0) = \{\eta \in \mathbb{R},\ f(\beta) \geq f(\beta_0) + < \eta, \beta - \beta_0 >\}$$

Dans notre cas, on a que :

$$\begin{aligned} \partial L(0, \lambda) &= \{\eta \in \mathbb{R},\ L(\beta, \lambda) \geq L(0, \lambda) + < \eta, \beta - 0 >\} \\ &= \{\eta \in \mathbb{R},\ L(\beta, \lambda) \geq L(0, \lambda) + \eta\beta\} \\ &= \{\eta \in \mathbb{R},\ \frac{(y - x\beta)^2}{n} + \lambda|\beta| \geq \frac{(y - x \times 0)^2}{n} + \lambda \times 0 + \eta\beta\} \\ &= \{\eta \in \mathbb{R},\ \frac{(y - x\beta)^2}{n} - \frac{y^2}{n} \geq -\lambda|\beta| + \eta\beta\} \end{aligned}$$

En faisant $\beta \to 0$ et $\beta \geq 0$ alors on a :

$$\begin{aligned} \partial L(0, \lambda) \cap \{\beta \geq 0\} &= \{\eta \in \mathbb{R},\ \frac{(y - x\beta)^2}{n} - \frac{y^2}{n} \geq -\lambda|\beta| + \eta\beta\} \\ &= \{\eta \in \mathbb{R},\ \frac{(y - x\beta)^2 - y^2}{n\beta} \geq -\lambda + \eta\} \\ &= \{\eta \in \mathbb{R},\ \frac{-2xy}{n} \geq -\lambda + \eta\} \\ &= \{\eta \in \mathbb{R},\ \lambda + \frac{-2xy}{n} \geq \eta\} \end{aligned}$$

En faisant $\beta \to 0$ et $\beta \leq 0$ alors on a :

$$\begin{aligned} \partial L(0, \lambda) \cap \{\beta \leq 0\} &= \{\eta \in \mathbb{R},\ (y - x\beta)^2 - y^2 \geq -\lambda|\beta| + \eta\beta\} \\ &= \{\eta \in \mathbb{R},\ \frac{(y - x\beta)^2 - y^2}{\beta} \geq \lambda + \eta\} \\ &= \{\eta \in \mathbb{R},\ \frac{-2xy}{n} \geq \lambda + \eta\} \\ &= \{\eta \in \mathbb{R},\ -\lambda + \frac{-2xy}{n} \geq \eta\} \end{aligned}$$

On a donc que $\partial L(0, \lambda) = [\partial L(0, \lambda) \cap \{\beta \leq 0\}] \cup [\partial L(0, \lambda) \cap \{\beta \geq 0\}] = \left[-\lambda - \frac{2xy}{n}; \lambda - \frac{2xy}{n}\right]$.

De plus, par le Théorème 3.6 du polycopié de Philippe Mahey [6] on sait que : une condition nécessaire et suffisante pour qu'un point β_0 soit un minimum global d'une fonction $L(\beta, \lambda)$

convexe en β est $\beta_0 \in \partial L(\beta_0, \lambda)$.
On a donc, ici, que $0 \in \partial L(0, \lambda)$ puisque l'on est dans le cas où $\beta = 0$ est le minimum global de $L(., \lambda)$, on a donc que

$$-\lambda - \frac{2xy}{n} \leq 0 \Leftrightarrow -\frac{2xy}{n} \leq \lambda$$
$$\lambda - \frac{2xy}{n} \geq 0 \Leftrightarrow -\frac{2xy}{n} \geq -\lambda$$

ce qui revient à

$$-\lambda \leq -\frac{2xy}{n} \leq \lambda \Leftrightarrow |-\frac{2xy}{n}| \leq \lambda$$

Ce qui est la seconde condition de la proposition.

□

Au delà de sélectionner des variables, le but du LASSO va aussi être de ne sélectionner que des variables qui soient explicatives. Avec l'hypothèse de parcimonie de β, le LASSO va devoir construire un $\widehat{\beta}$ tel que l'ensemble des $\widehat{\beta}_j$ non nuls corresponde avec l'ensemble des β_j non nuls. On va donc considérer l'ensemble des variables actives :

$$S = \{j : \beta_j \neq 0\}$$

Le but du LASSO est donc de construire un estimateur $\widehat{\beta}$ tel que le jeu de variables actives sélectionné soit proche du vrai jeu de variables actives S. On note :

$$\widehat{S}(\lambda) = \{j : \widehat{\beta}_j(\lambda) \neq 0\}$$

Une solution idéale serait alors de pouvoir déduire S des données. Une solution plus réaliste va donc être de se concentrer sur les variables dites significatives, c'est-à-dire les variables pour lesquelles le coefficient β_j associé est suffisamment important pour ne pas être négligé. On va donc nommer un ensemble de la sorte :

$$S^C = \{j : |\beta_j| \geq C\} \text{ avec } C > 0$$

Proposition 3. *On a que :*

$$\mathbb{P}\big(S^C \subset \widehat{S}(\lambda)\big) \underset{n\to+\infty}{\longrightarrow} 1$$

Démonstration. On a, d'après le Théorème 6.1 Bülhmann et Van de Geer [1], que

$$\mathbb{P}\big(\|\widehat{\beta} - \beta_0\|_1 \geq K\lambda\big) \leq 2e^{-\frac{t^2}{2}} \quad \text{avec } K > 0 \text{ une constante et } t > 0$$

et $\lambda \geq 4\sigma\sqrt{\dfrac{t^2 + \log(p_n)}{n}}$ avec σ le paramètre de variance des erreurs.

On obtient donc que $\mathbb{P}\left(\|\widehat{\beta} - \beta_0\|_1 \geq K\sqrt{\dfrac{t^2 + \log(p_n)}{n}}\right) \leq 2e^{-\frac{t^2}{2}}$.

En notant $\sqrt{\dfrac{\log(p_n)}{n}} \simeq \lambda_n \underset{n\to+\infty}{\longrightarrow} 0$, alors si on choisit :

$$t^2 = O\Big(\frac{\log(p_n)}{n}\Big) \times \frac{1}{\lambda_n^{1+\alpha}} = \frac{1}{\lambda_n^{\alpha}} \underset{n\to+\infty}{\longrightarrow} +\infty \quad \text{avec } \alpha > 0,$$

$$\text{on a que} \quad e^{-\frac{t^2}{2}} \underset{n\to+\infty}{\longrightarrow} 0.$$

De plus $\dfrac{t^2}{n} \to 0 \;\Leftrightarrow\; \dfrac{1}{n\lambda_n^{\alpha}} \to 0 \;\Leftrightarrow\; n\lambda_n^{\alpha} \to +\infty$

Finalement, obtient que $\mathbb{P}\big(\|\widehat{\beta}(\lambda) - \beta\|_1 > K\lambda_n\big) \underset{n\to+\infty}{\longrightarrow} 0$.

De cette façon on a que $\mathbb{P}\big(S^C \subset \widehat{S}(\lambda)\big) \to 1$ car dans le cas contraire, on a que il existe un ensemble Ω_0 de probabilité non nulle tel que tel que $S^C \not\subset \widehat{S}(\lambda)$. C'est-à-dire que pour chaque $\omega \in \Omega_0$, $\exists j(\omega)$ tel que $|\widehat{\beta}_{j(\omega)} - \beta_{j(\omega)}| \geq C$. Ce qui entraînerait que $\mathbb{P}\big(\|\widehat{\beta}(\lambda) - \beta\|_1 > C\big) \geq \mathbb{P}(\Omega_0) > 0$.

□

Ainsi, avec l'ensemble des résultats pré-cités, on conclut que le modèle sélectionné par la LASSO va contenir avec une grande probabilité les variables significatives. De plus le LASSO va sélectionner un modèle avec au plus $\min(n, p)$ variables. Ainsi, pour des problèmes où $p \gg n$ le LASSO va considérablement réduire le nombre de variables à prendre en compte.

En pratique, le choix du λ se fait par validation croisée, cependant on conserve les résultats énoncés ci-dessus. En général, on observe aussi que la valeur de λ sélectionnée doit être plus grande que $\sqrt{\dfrac{log(p)}{n}}$.

1.2.2 Mise en œuvre

Pour mettre en œuvre la procédure du LASSO, nous avons utilisé le package `R` nommé `glmnet` [3], les détails concernant l'utilisation de ce package sont donnés dans la partie 2.2.

Un autre algorithme de résolution du LASSO est l'algorithme FISTA développé dans la thèse de M. DAOUD [7] qui est l'un des algorithmes les plus rapides pour la résolution du LASSO. On donne un aperçu de cet algorithme ci-après.

On pose :

$$F(\beta) = \frac{1}{2}\|Y - X\beta\|_2^2 + \lambda\|\beta\|_1$$

avec les mêmes notations que précédemment.

Le but de l'algorithme FISTA est de construire des minimisateurs de F en utilisant la fonction de seuillage $S_\lambda : \mathbb{R} \to \mathbb{R}$, définie par :

$$S_\lambda(x) = x + \lambda \text{ si } x \leq -\lambda,$$
$$S_\lambda(x) = 0 \text{ si } x \leq \lambda,$$
$$S_\lambda(x) = x - \lambda \text{ si } x \geq \lambda.$$

On prolonge cette fonction à une fonction $\mathbb{R}^p \to \mathbb{R}^p$ en posant :

$$S_\lambda(\beta) = \big(S_\lambda(\beta_i), i = 1, \ldots, p\big)$$

On aura besoin de deux choses : $\theta_k = \frac{2}{k+1}$ et une suite (λ_k) choisie convenablement. L'algorithme FISTA est le suivant :

Choisir $\beta_0 = v_0$, pour $k \geq 1$ répéter jusqu'à la convergence :

$$w_{k-1} = (1 - \theta_k)\beta_{k-1} + \theta_k v_{k-1}$$
$$\beta_k = S_{\lambda_k}\big(w_{k-1} - \lambda_{k-1} X^T (X w_{k-1} - Y)\big)$$
$$v_k = \beta_{k-1} + \frac{\beta_k - \beta_{k-1}}{\theta_k}$$

Le pas de cet algorithme t_k peut être choisit de différentes façons (pas fixe ou non) qui vont modifier le comportement de l'algorithme.

1.3 Analyse en composantes principales

1.3.1 Idée générale

L'analyse en composantes principales est une méthode d'analyse factorielle des données qui consiste à transformer des variables liées entre elles, c'est-à-dire ayant une corrélation différente de zéro, en nouvelles variables décorrélées les unes des autres. Ces nouvelles variables sont nommées composantes principales. Cette méthode permet de réduire le nombre de variables, de rendre l'information apportée par les variables moins redondante, et de réduire la dimension du nuage pour faciliter l'exploration statistique de données quantitatives complexes, c'est-à-dire de grandes bases de données.

Il s'agit donc d'une approche statistique, mais également géométrique, qui vise à synthétiser l'information. En effet, l'Analyse en Composantes Principales peut être considérée comme une méthode de projection qui permet de projeter les observations depuis l'espace à n dimensions des p variables vers un espace à k dimensions (k < n) dans le but qu'un maximum d'information soit conservée sur un minimum dimensions. On mesure l'information grâce à la variance totale du nuage de points. Si l'information associée aux 2 ou 3 premiers axes représente un pourcentage suffisant de la variabilité totale du nuage de points, on pourra représenter les observations sur un graphique à 2 ou 3 dimensions, facilitant ainsi grandement l'interprétation.

L'ACP permet donc d'explorer les liaisons entre variables, de créer des groupes de variables ayant les mêmes caractéristiques, et ainsi d'obtenir des nouvelles variables pouvant résumer ces groupes de variables. Concernant les individus, elle permet de mettre en évidence les ressemblances entre individus, de visualiser les individus via une notion de distance entre ceux-ci. Comme distance, une méthode est de prendre la distance euclidienne.

Si les variables étaient qualitatives, nous pourrions effectuer une variante de l'ACP, comme l'ACF (Analyse Factorielle des correspondances) dans le cas de deux variables, ou l'ACM (Analyse des correspondances multiples) dans le cas de plusieurs.

1.3.2 Concept

On affecte un poids p_i à chaque individu. La matrice des poids est donc : $D = \begin{pmatrix} p_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & p_n \end{pmatrix}$.

Si les individus ont un rôle symétrique, tous ont un poids de $\frac{1}{n}$, ce qui est notre cas, alors la matrice des poids des individus devient $\frac{1}{n} I_d$.

Pour ne pas prendre en compte les variances des variables, et donc donner plus de poids aux variables ayant une grande variance, on choisit de normaliser les variables. On effectue donc l'opération suivante : $x_{i,j} = \frac{x_{i,j} - \overline{x_j}}{\sigma_j}$, où $\sigma_j^2 = \frac{1}{n} \sum_{i=1}^{n} (x_{i,j} - \overline{x_j})^2$ et $\overline{x_j} = \frac{1}{n} \sum_{i=1}^{n} x_{i,j}$.

On définit les nuages des individus et des variables comme suit : $N = (x_{i,.}, \frac{1}{n}),\ 1 \leq i \leq n$ pour les individus, et $N_v = (x_{.,j}, \frac{1}{p}),\ 1 \leq j \leq p$ pour les variables.

Le principe général de l'ACP est de trouver une représentation incluant les n individus, dans un sous-espace F_k de $\mathbb{R}^p$ de dimension k, avec k petit. Autrement dit, on cherche à définir k nouvelles variables combinaisons linéaires des p variables initiales qui permettraient de garder le plus possible d'information par rapport aux variables initiales. Ces axes sont appelés composantes principales.

Pour trouver le sous-espace optimal, nous cherchons à minimiser la somme des carrés des distances euclidiennes des points représentant les individus à F_k. Ceci nous amène à choisir pour F_k l'espace sur lequel le nuage projeté a une inertie maximale, l'inertie de la projection du nuage de dimension k étant définie comme suit : $I(F_k) = \sum_{i=1}^{n} p_i \|(x_i - \hat{x_i})\|^2$, où $\hat{x_i}$ est la projection de x_i sur F_k. On écrit $g = \sum_{i=1}^{n} p_i x_{i.}$ est le centre de gravité du nuage. On appelle matrice d'inertie pour les individus la matrice suivante :$V = \frac{1}{n} X^t X \in \mathcal{M}_{p,p}$.

La solution est $F_k = Vect(u_1, \cdots, u_k) \forall k \geq 1$ où les $u_1 \cdots u_k$ sont les vecteurs propres de la matrice d'inertie V. On a alors le résultat suivant : $I(\Delta u_l) = \lambda_l$, et donc $I(F_k) = \sum_{l=1}^{n} \lambda_l$, où λ_l est la $l^{\text{ième}}$ plus grande valeur propre. La part d'inertie expliquée par Δu_k est alors : $\tau_k = \frac{\lambda_k}{\sum_{l=1}^{p} \lambda_l}$, et celle expliquée par F_k est $\sum_{l=1}^{k} \tau_k = \frac{\sum_{l=1}^{k} \lambda_l}{\sum_{l=1}^{p} \lambda_l}$.

On en déduit que l'inertie des k premiers vecteurs propres est égale à la somme des k plus grandes valeurs propres associés à ces vecteurs propres.

Les composantes principales sont associées à des axes principaux ayant pour direction les vecteurs propres de la matrice de variance-covariance. La $k^{\text{ième}}$ composante principale représente les coordonnées des individus sur l'axe k.

Comment choisir k ? Il existe plusieurs solutions :

- On peut fixer un seuil α d'inertie expliquée, par exemple 80%, et choisir le nombre d'axes qui permettent de dépasser ce seuil.
- La règle du point d'inflexion utilise le diagramme des valeurs propres, et permet de choisir le nombre d'axes à retenir en regardant la monotonie des valeurs propres successives. Dès que celle-ci change, on ne retient plus les axes suivants.
- On peut également choisir de retenir un axe si $\tau_k \geq \frac{1}{p}$.
- La règle du coude permet de choisir le nombre d'axes en fonction de la différence entre les valeurs propres successives. En effet, dès que la différence entre valeurs propres successives devient petite, on ne retient plus les axes suivants.
- Enfin, on peut choisir de ne retenir que les axes que l'on pourra par la suite expliquer.

Les composantes principales du nuage des individus sont définies par : $C_k = Xu_k$ où u_k est le $k^{\text{ième}}$ vecteur propre associé à la valeur propre λ_k de la matrice d'inertie défini précédemment.

On représente les points associés aux individus sur une sphère. Il existe des instruments de mesures pour pouvoir vérifier si un individu est bien représenté sur le nuage, et par quel axe il est le mieux représenté. En effet, on peut choisir de regarder la contribution de l'individu j à l'axe k, c'est-à-dire le pourcentage de l'inertie de l'axe k dû à l'individu j : $CTR_k(j) = \frac{p_i c_{ik}^2}{\lambda_k} \in [0,1]$, avec $\sum_{i=1}^{n} CTR_k(i) = 1$. Un autre indicateur est la qualité de la représentation de la variable j sur l'axe k : $CO2_k(i) = cos^2\theta_{ik} = \frac{c_{ik}^2}{\|x_i\|^2}$, où c_{ik}^2 est le $i^{\text{ième}}$ élément de la $k^{\text{ième}}$ composante principale. Plus $CO2_k(i)$ est proche de 1, plus x_i est proche de δu_k.

1.4 Méthodes de classification des données

La classification est une autre méthode d'analyse de données (clustering=partition en anglais). Elle consiste à regrouper en classes des objets similaires selon certains critères, tel que tout individu soit dans une classe, et que les classes forment une partition. Ici, nous souhaitons travailler sur les variables explicatives, donc notre objectif est de regrouper les p variables en fonction de leur homogénéité, en un certain nombre de groupes, où tous les groupes sont bien différenciés les uns des autres, pour ensuite sélectionner une ou plusieurs variables dans chaque groupe pour créer notre modèle. Il existe diverses techniques de classification, nous allons aborder la méthode de classification hiérarchique ascendante (CHA), et la méthode K-means (méthode de partitionnement). Ces deux méthodes sont complémentaires.

1.4.1 Classification hiérarchique ascendante

La classification hiérarchique ascendante consiste à créer une hiérarchie, c'est-à-dire à obtenir une collection de groupes de variables. C'est une méthode de classification itérative et automatique. L'algorithme démarre avec n singletons, chaque variable constitue un groupe, puis rassemble les points variables qui sont considérés les plus proches, selon un critère choisi, et ce jusqu'à l'obtention d'un groupe contenant toutes les variables. Nous avons donc besoin de définir une notion de distance entre les points variables, ainsi que d'un critère de regroupement des groupes, aussi appelé stratégie d'agrégation ou stratégie de classification.

Il existe plusieurs possibilités pour le choix de la notion de distance entre les points :

— La distance euclidienne : $d_2(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$.

— La distance L_1 : $d_1(x,y) = \sum_{i=1}^{n}|x_i - y_i|$.

— La distance L_∞ : $d_\infty(x,y) = max|x_i - y_i| : i = 1, \cdots, n$.

Après avoir défini une distance entre individus, on cherche à définir des distances entre ensembles qui correspondent à différentes stratégies d'agrégation :

— Single Linkage Clustering (Saut minimum ou lien simple) : On regroupe les 2 éléments présentant la plus petite distance entre éléments des deux classes.
$D(C_1, C_2) = \min d(a,b) : a \in C_1, b \in C_2$.

— Complete Linkage Clustering (Lien complet) : On regroupe les 2 éléments présentant la plus grande distance entre éléments des deux classes.
$D(C_1, C_2) = \max d(a,b) : a \in C_1, b \in C_2$.

— Average Linkage Clustering : On choisit comme référence de distance la moyenne des distances entre chaque élément des deux groupes. $D(C_1, C_2) = \frac{\sum_{a \in C_1}\sum_{b \in C_2} d(a,b)}{cardC_1 cardC_2}$.

— Critère d'inertie : On choisit le critère de WARD, visant à maximiser l'inertie inter-classe, et donc minimiser l'inertie intra-classe. On a : $D(C_1, C_2) = \sqrt{\frac{p_1 p_2}{p_1 + p_2}}\|g_{C_1} - g_{C_2}\|^2$, où p_i correspond au poids du groupe C_i. Si on choisit de regrouper les groupes C_1 et C_2, l'inertie inter-classe diminue de $D(A,B)$.

A chaque étape de la CHA, on regroupe les deux ensembles dont la distance D est minimale.

Le critère d'inertie est celui qui donne les meilleurs résultats dans la pratique.

On dit qu'une classification est bonne si la variabilité intra-classe est petite, et la variabilité inter-classe est grande. On sait notamment que l'inertie totale d'un groupe (la dispersion) est équivalent à la somme de l'inertie inter-classe et l'inertie intra-classe. Le but est donc de regrouper les variables qui minimisent la perte d'inertie inter-classe.

La classification hiérarchique implique que les résultats dépendent du choix de distance, et du critère de regroupement. De plus, dans le cas de données importantes, le nombre de calculs devient très important. Cependant, cette méthode permet d'obtenir un arbre optimal, grâce à une meilleure lecture de l'arbre de classification, ou dendogramme. La Figure 1.2 est un exemple de dendogramme associé à un nuage de points. En effet, elle permet de mettre en évidence une hiérarchie entre variables, et groupes de variables. La racine de l'arbre, située au sommet, correspond au groupe contenant tous les points. On peut choisir une partition en tronquant l'arbre, selon différents critères.

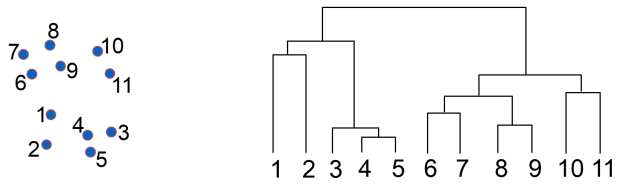

FIGURE 1.2 – Exemple de Dendogramme

1.4.2 Méthode K-means

La classification K-means a été introduite par MacQueen en 1967, elle consiste à créer k groupes, k étant fini. La première étape est de tirer au hasard les centres des k classes. L'algorithme va alors rapprocher les points selon ces centres et créer les k groupes en regroupant les variables les plus proches d'un centre donné. Ensuite, on calcule les centres de gravité de chaque groupe puis on créé de nouveaux groupes en rapprochant les variables du centre de gravité le plus proche. On réitère ce processus, jusqu'à ce que les points ne changent plus de groupes, et que la répartition devienne donc stable. L'algorithme converge toujours vers une solution. En effet, au cours de l'algorithme, l'inertie inter-classe est croissante, alors il y a convergence à condition qu'il n'y ait pas de partitions à k classes de même inertie inter-classe. En appliquant l'algorithme plusieurs fois, nous n'aboutissons pas forcement à la même solution, ce qui est dû au choix aléatoire des premiers centres.

Cette méthode de partitionnement a pour avantage de pouvoir classifier des ensembles volumineux, d'être facile de compréhension, et rapide. De plus, un point variable peut changer de groupe au cours de l'algorithme. Cependant, on impose au départ de l'algorithme le nombre de groupes, et donc le résultat dépend de ce choix, ainsi que du tirage initial des centres de classes. Au contraire de la classification hiérarchique ascendante, l'algorithme K-means n'aboutit pas sur un arbre optimal, puisqu'il dépend des données initiales.

1.5 Indicateurs

Pour qualifier la qualité d'un modèle nous avons plusieurs estimateurs à disposition.

Le MSE est la moyenne arithmétique des carrés des écarts entre les prévisions et les observations, autrement dit la moyenne des erreurs au carré.
Un autre indicateur est le R^2, il est définit ci-après et est lié au MSE :

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y}_i)^2 \quad \text{avec} \quad \widehat{y}_i \quad \text{la prédiction de} \quad y_i$$

$$R^2 = 1 - \frac{\frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y}_i)^2}{\frac{1}{n}\sum_{i=1}^{n}(y_i - \overline{y})^2} = 1 - \frac{MSE}{\frac{1}{n}\sum_{i=1}^{n}(y_i - \overline{y})^2} \quad \text{avec} \quad \overline{y} \quad \text{la moyenne des} \quad y_i$$

On remarque que plus le MSE sera petit, plus le R^2 sera proche de 1. Un coefficient de détermination (R^2) proche de 1 signifie que le modèle explique bien l'activité chimique, et donc que les variables retenues apportent de l'information sur la variable à expliquer. Autrement dit un MSE proche de 0 signifie que le modèle est très satisfaisant.

Un autre indicateur est le $R^2_{ajusté}$. Le $R^2_{ajusté}$ est une version modifiée du R^2, il est ajusté pour tenir compte du nombre de variables dans le modèle. En effet, le R^2 augmente mécaniquement à chaque ajout d'une variable dans le modèle, tandis que le $R^2_{ajusté}$ n'augmente que si le nouveau terme améliore significativement la qualité du modèle. Le $R^2_{ajusté}$ est toujours inférieur au R^2 et est défini comme suit :

$$R^2_{ajusté} = 1 - \frac{(1 - R^2)(n-1)}{n-k-1} \quad \text{avec} \quad k \quad \text{le nombre de variables dans le modèle.}$$

Un autre indicateur que l'on pourra utiliser pour comparer les résultats des différents LASSO entre eux est un indicateur que l'on appellera $MSELASSO$ et qui est le résultat de l'optimisation exécutée par le LASSO. On le définit comme suit :

$$MSELASSO = \frac{1}{2n}\|Y - X\hat{\beta}\|_2^2 + \lambda\|\hat{\beta}\|_1$$

Comme nous le verrons dans la suite de l'étude, la valeur qui va nous servir le plus est le MSE. En effet, celui-ci est facilement calculable et comparable d'un modèle à l'autre. Le $MSELASSO$ nous à donné des résultats auxquels nous ne nous attendions pas, qui nous ont semblé peu comparable les uns aux autres.

Chapitre 2

Nos fonctions

Dans ce chapitre, nous allons détailler la façon dont nous allons traiter les données et la façon dont nous avons raisonné dans la création des nos fonctions situées en Annexe.
Tout d'abord, nous aborderons la création de groupes de variables, que nous appellerons familles. Ceci représente une grosse partie de notre travail. Puis ensuite, nous expliquerons l'utilité de nos autres fonctions, concernant le LASSO, ou la classification.

2.1 Création des familles

En observant les variables de la table HCV, nous nous sommes rendus compte que les noms des variables voisines dans la table étaient souvent proches les uns des autres. Notre intuition nous à conduit à penser que ces variables pouvaient avoir des significations chimiques proches les unes des autres. Dans ce cas, sélectionner plusieurs variables de la même famille dans un modèle n'aurait pas beaucoup de sens d'un point de vue chimique puisque l'information serait considérée comme redondante. De plus cela nous permettrait de réduire la dimension des données à traiter. Nous avons donc pris contact avec le spécialiste à l'origine des données pour lui demander confirmation sur nos intuitions et effectivement, il s'est avéré que les variables forment effectivement des groupes de variables de sens chimique proches. Nous avons donc décidé de créer des "familles" de variables et nous nous y prendrons de trois façons différentes : un regroupement en fonction des noms, un regroupement selon les méthodes de clustering et enfin les familles constituées par le spécialiste.

2.1.1 Création des familles en se basant sur les noms

La premier élément qui nous a mis sur la piste des familles est le nom des variables. Nous avons donc observé les noms des variables pour essayer de trouver un sens à familles de variables qui puisse être traduisible algorithmiquement. Nous nous sommes rendus compte que beaucoup de variables possédaient une "racine" commune, c'est-à-dire un groupe de lettres communes auxquelles à été ajouté un terme d'indexation, soit devant soit derrière la racine pour différencier les différents individus d'une même famille. Il existe alors deux types de familles : $X(i)$ et $(i)X$ avec X la racine et (i) le terme d'indexation. On peut illustrer ces deux différents types de familles par les variables suivantes : ($PW2$, $PW3$, $PW4$, $PW5$) est une famille du type $X(i)$ avec $X = PW$ et $(i) \in \{2, 3, 4, 5\}$; et ($X0sol$, $X1sol$, $X2sol$, $X3sol$, $X4sol$, $X5sol$) est une famille du type $(i)X$ avec $X = sol$ et $(i) \in \{X0, X1, X2, X3, X4, X5\}$. Nous avons donc choisi de construire une fonction sous `R` permettant de créer des familles en ce sens : deux variables sont dites appartenant à la même famille si elles sont voisines dans la table et si leurs noms

commencent ou finissent par les mêmes lettres.
La fonction `CreationFamille` réalisant ce travail fonctionne selon les étapes suivantes :

1. On regroupe les variables voisines pour lesquelles les trois dernières lettres sont communes.
2. On regroupe les variables voisines non encore regroupées pour lesquelles les trois premières lettres sont communes
3. On regroupe les variables voisines non encore regroupées pour lesquelles les deux premières lettres sont communes
4. On regroupe les variables voisines non encore regroupées pour lesquelles les deux dernières lettres sont communes
5. On regroupe les variables voisines non encore regroupées pour lesquelles la première lettre est commune
6. On regroupe les variables voisines non encore regroupées pour lesquelles la dernière lettre est commune

A chaque étape, une famille créée n'est plus modifiée.

Nous sommes conscients que cet algorithme n'est pas parfait, en effet, certaines variables ne sont pas regroupées alors que selon nous, elles devraient l'être. On peut penser qu'échanger les étapes de notre algorithme et préférer commencer par regrouper les variables ayant les trois premières lettres communes, pourrait aboutir à un meilleur résultat. Cependant chaque choix implique la création de familles différentes et bien entendu des erreurs de regroupement différentes. Après de nombreux essais, et comparaisons entre les différentes façons de créer ces familles, nous nous sommes arrêtés sur l'algorithme décrit précédemment. Nous avons choisit cet ordre de regroupement car certaines variables peuvent être regroupées par les premières, et certaines par les dernières lettres de leurs noms. La Figure 2.1 montre bien un cas de figure où regrouper d'abord par les trois dernières est préférable. En effet, si nous avions commencé par les trois premières lettres, toutes ces familles se seraient retrouvées ensemble, on aurait donc eu une famille de 96 éléments alors que nous les avons séparés en 10 familles. Cela nous semblait un peu réducteur de rassembler tout les "CATS3D" ensemble.

```
> famille[c(1:10), c(428:433)]
      [,1]          [,2]          [,3]          [,4]          [,5]          [,6]
 [1,] "CATS3D_00_AA" "CATS3D_02_AP" "CATS3D_02_AL" "CATS3D_00_PP" "CATS3D_03_PL" "CATS3D_00_LL"
 [2,] "CATS3D_01_AA" "CATS3D_04_AP" "CATS3D_03_AL" NA            "CATS3D_04_PL" "CATS3D_01_LL"
 [3,] "CATS3D_02_AA" "CATS3D_07_AP" "CATS3D_04_AL" NA            "CATS3D_06_PL" "CATS3D_02_LL"
 [4,] "CATS3D_03_AA" "CATS3D_08_AP" "CATS3D_05_AL" NA            "CATS3D_07_PL" "CATS3D_03_LL"
 [5,] "CATS3D_04_AA" "CATS3D_12_AP" "CATS3D_06_AL" NA            "CATS3D_08_PL" "CATS3D_04_LL"
 [6,] "CATS3D_05_AA" NA            "CATS3D_07_AL" NA            "CATS3D_09_PL" "CATS3D_05_LL"
 [7,] "CATS3D_06_AA" NA            "CATS3D_08_AL" NA            "CATS3D_10_PL" "CATS3D_06_LL"
 [8,] "CATS3D_07_AA" NA            "CATS3D_09_AL" NA            "CATS3D_11_PL" "CATS3D_07_LL"
 [9,] "CATS3D_08_AA" NA            "CATS3D_10_AL" NA            NA            "CATS3D_08_LL"
[10,] "CATS3D_09_AA" NA            "CATS3D_11_AL" NA            NA            "CATS3D_09_LL"
>
```

FIGURE 2.1 – Echantillon des familles créées

Cet algorithme est celui qui offre, selon nous, les familles les plus représentatives, et qui présentent le moins d'erreurs de regroupement. Voici un échantillon de nos familles, Figure 2.2.

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]	[,11]
[1,]	"MWC01"	"piPC01"	"Uindex"	"BIC0"	"ATS1m"	"JGI1"	"SpMax1_Bh(m)"	"SpMax1_Bh(s)"	"SM02_EA(ed)"	"Eig01_EA(bo)"	"CATS3D_00_LL"
[2,]	"MWC02"	"piPC02"	"Vindex"	"BIC1"	"ATS2m"	"JGI2"	"SpMax2_Bh(m)"	"SpMax2_Bh(s)"	"SM03_EA(ed)"	"Eig02_EA(bo)"	"CATS3D_01_LL"
[3,]	"MWC03"	"piPC03"	"Xindex"	"BIC2"	"ATS3m"	"JGI3"	"SpMax3_Bh(m)"	"SpMax3_Bh(s)"	"SM04_EA(ed)"	"Eig03_EA(bo)"	"CATS3D_02_LL"
[4,]	"MWC04"	"piPC04"	"Yindex"	"BIC3"	"ATS4m"	"JGI4"	"SpMax4_Bh(m)"	"SpMax4_Bh(s)"	"SM05_EA(ed)"	"Eig04_EA(bo)"	"CATS3D_03_LL"
[5,]	"MWC05"	"piPC05"	NA	"BIC4"	"ATS5m"	"JGI5"	"SpMax5_Bh(m)"	"SpMax5_Bh(s)"	"SM06_EA(ed)"	"Eig05_EA(bo)"	"CATS3D_04_LL"
[6,]	"MWC06"	"piPC06"	NA	"BIC5"	"ATS6m"	"JGI6"	"SpMax6_Bh(m)"	"SpMax6_Bh(s)"	"SM07_EA(ed)"	"Eig06_EA(bo)"	"CATS3D_05_LL"
[7,]	"MWC07"	"piPC07"	NA	NA	"ATS7m"	"JGI7"	"SpMax7_Bh(m)"	"SpMax7_Bh(s)"	"SM08_EA(ed)"	"Eig07_EA(bo)"	"CATS3D_06_LL"
[8,]	"MWC08"	"piPC08"	NA	NA	"ATS8m"	"JGI8"	"SpMax8_Bh(m)"	"SpMax8_Bh(s)"	"SM09_EA(ed)"	"Eig08_EA(bo)"	"CATS3D_07_LL"
[9,]	"MWC09"	"piPC09"	NA	NA	"ATS1v"	"JGI9"	NA	NA	"SM10_EA(ed)"	"Eig09_EA(bo)"	"CATS3D_08_LL"
[10,]	"MWC10"	"piPC10"	NA	NA	"ATS2v"	"JGI10"	NA	NA	"SM11_EA(ed)"	"Eig10_EA(bo)"	"CATS3D_09_LL"
[11,]	NA	NA	NA	NA	"ATS3v"	NA	NA	NA	"SM12_EA(ed)"	"Eig11_EA(bo)"	"CATS3D_10_LL"
[12,]	NA	NA	NA	NA	"ATS4v"	NA	NA	NA	"SM13_EA(ed)"	"Eig12_EA(bo)"	"CATS3D_11_LL"
[13,]	NA	NA	NA	NA	"ATS5v"	NA	NA	NA	"SM14_EA(ed)"	"Eig13_EA(bo)"	"CATS3D_12_LL"
[14,]	NA	NA	NA	NA	"ATS6v"	NA	NA	NA	"SM15_EA(ed)"	"Eig14_EA(bo)"	"CATS3D_13_LL"
[15,]	NA	NA	NA	NA	"ATS7v"	NA	NA	NA	NA	"Eig15_EA(bo)"	"CATS3D_14_LL"
[16,]	NA	NA	NA	NA	"ATS8v"	NA	NA	NA	NA	NA	"CATS3D_15_LL"
[17,]	NA	NA	NA	NA	"ATS1e"	NA	NA	NA	NA	NA	"CATS3D_16_LL"
[18,]	NA	NA	NA	NA	"ATS2e"	NA	NA	NA	NA	NA	"CATS3D_17_LL"
[19,]	NA	NA	NA	NA	"ATS3e"	NA	NA	NA	NA	NA	NA
[20,]	NA	NA	NA	NA	"ATS4e"	NA	NA	NA	NA	NA	NA

FIGURE 2.2 – Echantillon des familles créées

Pour vérifier la cohérence de nos familles, nous avons effectué plusieurs tests. Etant donné que nos familles sont censées regrouper des variables dont le sens chimique est proche, les variables d'une même famille doivent être proches au sens statistique et donc elle doivent hypothétiquement présenter des corrélations fortes. Les résultats d'une ACP sur les familles devraient nous donner un premier axe principal comptant pour une très forte partie de l'inertie. Nous avons donc effectué ces deux tests sur des familles choisies au hasard et il s'est avéré que dans la plupart des cas, les familles que nous avons constituées vérifient les hypothèses que nous avons émises comme on peut voir sur les Figures 2.3 et 2.4.

	MWC01	MWC02	MWC03	MWC04	MWC05	MWC06	MWC07	MWC08	MWC09	MWC10
MWC01	1.0000000	0.9885686	0.9736879	0.9557092	0.9340741	0.9178053	0.8984717	0.8833846	0.8673723	0.8532530
MWC02	0.9885686	1.0000000	0.9927502	0.9841608	0.9681067	0.9556267	0.9397045	0.9261557	0.9121353	0.8984879
MWC03	0.9736879	0.9927502	1.0000000	0.9966241	0.9899313	0.9811166	0.9712624	0.9604621	0.9505045	0.9389553
MWC04	0.9557092	0.9841608	0.9966241	1.0000000	0.9965595	0.9923892	0.9845402	0.9771105	0.9684264	0.9594517
MWC05	0.9340741	0.9681067	0.9899313	0.9965595	1.0000000	0.9979186	0.9948324	0.9889592	0.9836464	0.9757965
MWC06	0.9178053	0.9556267	0.9811166	0.9923892	0.9979186	1.0000000	0.9982426	0.9957374	0.9913706	0.9863279
MWC07	0.8984717	0.9397045	0.9712624	0.9845402	0.9948324	0.9982426	1.0000000	0.9985341	0.9967495	0.9924948
MWC08	0.8833846	0.9261557	0.9604621	0.9771105	0.9889592	0.9957374	0.9985341	1.0000000	0.9989936	0.9972478
MWC09	0.8673723	0.9121353	0.9505045	0.9684264	0.9836464	0.9913706	0.9967495	0.9989936	1.0000000	0.9988776
MWC10	0.8532530	0.8984879	0.9389553	0.9594517	0.9757965	0.9863279	0.9924948	0.9972478	0.9988776	1.0000000

FIGURE 2.3 – Matrice de corrélation dans la famille n° 61

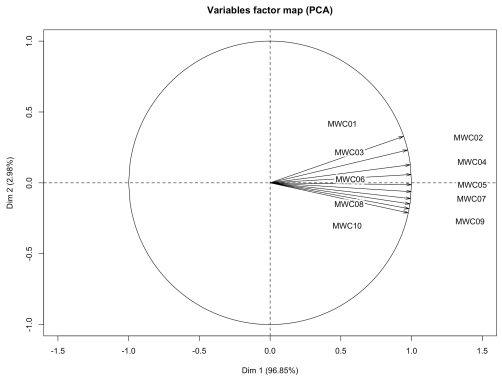

FIGURE 2.4 – ACP dans la famille n° 61

En effet, on peut voir que dans cette famille, la corrélation entre chaque membre est forte et que les coefficients de cette matrice sont rarement inférieures à 0.8. Toutes les corrélations sont positives donc les variables varient toutes dans le même sens. Les résultats de l'ACP confirment ces résultats, le premier axe principal compte pour 96.8% de l'inertie et graphiquement, les variables semblent très proches les unes des autres. Cette famille, représentative de la majorité des familles constituées, nous permet de montrer que nos familles sont bien constituées et de confirmer le choix de notre algorithme de constitution des familles. Cependant, comme nous l'avons dit plutôt, notre algorithme à ses failles et environs un trentaine de familles semblent assez mal constituées, soit à peine 7% de nos familles. Un exemple de ces familles est donné Figures 2.5 et 2.6.

	F06[C-C]	F06[C-N]	F06[C-O]	F06[C-S]	F06[C-F]	F06[C-Cl]	F06[N-N]	F06[N-O]	F06[N-S]	F06[N-F]	F06[N-Cl]
F06[C-C]	1.00000000	0.04846373	-0.1680463	-0.094548816	0.08654693	-0.38148609	0.226971973	-0.14242456	-0.16068818	0.11953796	-0.25057767
F06[C-N]	0.04846373	1.00000000	-0.4081611	-0.030401845	0.42289938	-0.05733560	0.544051924	0.10663633	0.12897377	-0.30871517	-0.06125733
F06[C-O]	-0.16804633	-0.40816114	1.0000000	-0.185099049	-0.22708441	-0.21649044	-0.302391296	0.15381590	-0.16795432	0.61327500	-0.16817262
F06[C-S]	-0.09454482	-0.03040184	-0.1850990	1.000000000	0.04316627	-0.06683385	0.007221694	-0.08034494	0.72634509	-0.04162539	-0.05191741
F06[C-F]	0.08654693	0.42289938	-0.2270844	0.043166275	1.00000000	-0.03311661	0.285939102	0.05441181	0.17914916	0.05864145	-0.10852946
F06[C-Cl]	-0.38148609	-0.05733560	-0.2164904	-0.066833851	-0.03311661	1.00000000	-0.063673154	-0.09397083	-0.03589554	-0.25851063	0.45775205
F06[N-N]	0.22697197	0.54405192	-0.3023913	0.007221694	0.28593910	-0.06367315	1.000000000	-0.07654528	0.03258085	-0.03366786	-0.02257089
F06[N-O]	-0.14242456	0.10663633	0.1538159	-0.080344940	0.05441181	-0.09397083	-0.076545279	1.00000000	-0.04315215	-0.19654721	-0.07299778
F06[N-S]	-0.16068818	0.12897377	-0.1679543	0.726345087	0.17914916	-0.03589554	0.032580846	-0.04315215	1.00000000	-0.04599035	-0.02788412
F06[N-F]	0.11953796	-0.30871517	0.6132750	-0.041625392	0.05864145	-0.25851063	-0.033667857	-0.19654721	-0.04599035	1.00000000	-0.23771909
F06[N-Cl]	-0.25057767	-0.06125733	-0.1681726	-0.051917413	-0.10852946	0.45775205	-0.022570888	-0.07299778	-0.02788412	-0.23771909	1.00000000

FIGURE 2.5 – Matrice de corrélation dans la famille n° 398

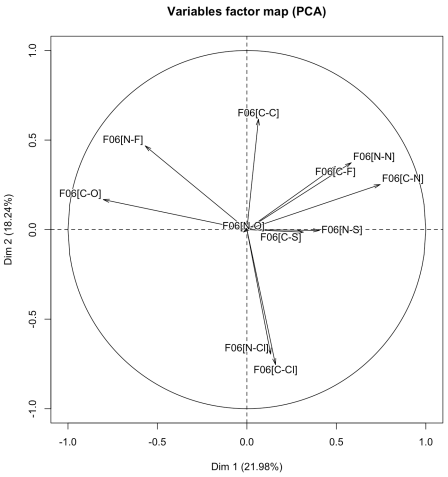

FIGURE 2.6 – ACP dans la famille n° 398

On peut voir pour cette famille, qui est un cas assez extrême, que la matrice de corrélation est plutôt hétérogène, avec des corrélations tantôt positives, tantôt négatives, et que la corrélation dépasse rarement 0.5 ce qui est très décevant et plutôt mauvais car on devrait avoir à l'intérieur d'une famille de grandes corrélations. L'ACP de cette famille confirme ces impressions, le premier axe principal ne compte que pour 21.9% de l'inertie et le second pour 18.4% de l'inertie, c'est-à-dire qu'avec deux axes, on atteint à peine 40% de l'inertie ce qui est très faible. Ce cas de familles mal constituées reste rare mais il est présent et nous en tiendrons compte dans la suite de l'étude.

Les familles que nous avons constituées semblent donc plutôt cohérentes des deux points de vue : chimique et statistique.

2.1.2 Création des familles à partir de la classification

Une seconde facon de créer des familles est d'utiliser la classification sur les nuages de variables. En effet, cette méthode statistique nous permet de créer des classes de variables, tel que à l'intérieur des classes les variables soient homogènes. Les méthodes de classification vont également créer ces classes de sorte qu'elles soient toutes bien distinctes entre elles. Nous allons d'abord appliquer à la table de données brutes, c'est-à-dire celle contenant 3193 variables explicatives, la classification hiérarchique ascendante, puis la méthode K-means.

Résultats de la méthode CHA

Tout d'abord, la Figure 2.7 est un dendogramme associé à nos données. On peut remarquer que regrouper les variables en familles semblent judicieux.

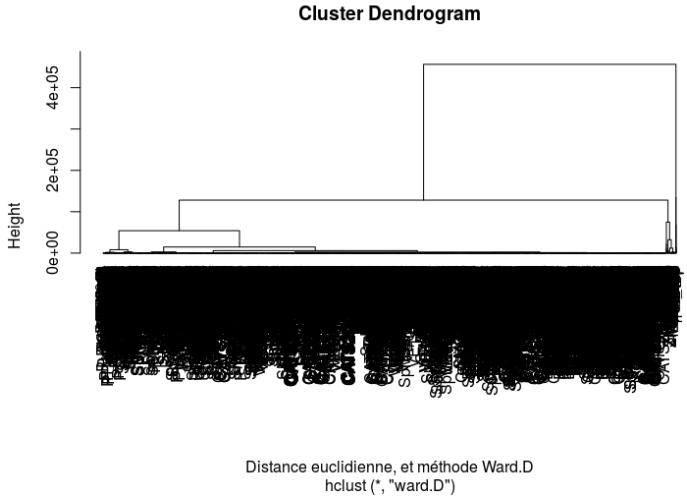

FIGURE 2.7 – Arbre de classification

Après plusieurs tests, nous avons choisi comme critère de regroupement celui qui concerne l'inertie inter-classe. En effet, les Figure 2.8, 2.9, 2.10 et 2.11 récapitulent le nombre de variables dans chaque classe en utilisant les différents critères de regroupement de la classification hiérarchique, appliqués avec la même distance, et dans le but d'obtenir le même nombre de groupe. Pour trouver ces résultats, nous avons utilisé les fonctions `VecteurVarParFam` et `Famille_CHA` .

```
> VarParFam <- VecteurVarParFam(N);VarParFam
  [1]   1  91  16  11 371 373  28 113  44  16 200 247  41  21  94 248  22  64 135  93 177   1   1   1   1   7   5   1   1
 [30]   3   1   1   3   2   1   1   1  43  27  12  12  41   2  54   9   1   1   1   1   1   1   4   4   1  27   5   2
 [59]   2   4   1   1   2   2   3   9  31   1   6   5  92   1   1   4   1   6   1   4   1   1   1   2   5   1   2   4   1
 [88]   3   6   2   1   2   3   1   1  13  14  39   1   1   1   1   1   1   1   2   1   2   1   6   1   1   1   1   3   4
[117]   1   4   3   2   3   2   2   3   1   2   7   5   2   3   2   1   1   1   2   1   1   2   1   1   1   1   1   1   1
[146]   1   1   1   4   2   1   1   1  10   1   1   3  25  18   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
[175]   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   3   3   3   3   1   1   2   2   1   2
[204]   2   2   2   2   1   2   1
```

FIGURE 2.8 – Critère Ward.D

```
> VarParFam <- VecteurVarParFam(N);VarParFam
  [1]   1 2527 296   44    1    1    1    1   20    3    1    1    1    1    1    1    1    1    1   18    1
 [24]   1    1    1    1    1    1    1    1    3    8    1    1    2    1    1    2    2    1    3    1    1    8    5
 [47]   1    1    1    4    1    1    4    1    1    1    1    2    2    4    1    1    4    2    1    1    2    1    4    1
 [70]   2    1    1    1    1    1    1    1    1    1    1    1    1    1    1    6    1    1    1    1    1    1    1
 [93]   1    4    1    1    1    3    2    1    1    1    2    1    1    1    1    1    1    1    1    2    1    2
[116]   1    1    1    1    1    1    2    1    1    1    1    1    1    1    1    1    1    1    1    1    4    2
[139]   1    1    1    1    1    1    3    3    1    1    1    1    1    1    1    1    1    1    1    1    1    1
[162]   1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
[185]   1    1    3    1    3    1    2    1    1    1    2    2    2    1    2    2    2    1    2    1    1    1    1
[208]   1    1    1
```

FIGURE 2.9 – Critère Average linkage

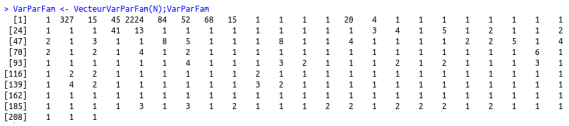

FIGURE 2.10 – Critère Complete Linkage

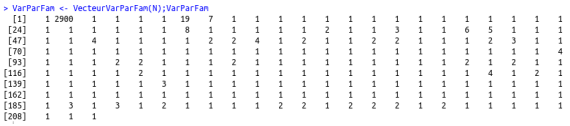

FIGURE 2.11 – Critère Single linkage

On remarque tout de suite que le critère de regroupement "Ward.D" est le meilleur. Cependant, il n'est pas parfait, vu que beaucoup de classes ne comportent qu'une seule variable. Les critères "Average Linkage", "Complete Linkage" et "Single Linkage" ne sont pas du tout adaptés à ces données, en effet, on peut voir que la plupart de nos 3193 variables sont regroupées dans une seule classe, ce qui ne nous apporte rien dans l'analyse que nous souhaitons réaliser.

Ensuite, nous avons comparé les trois distances possibles : L_1, L_2, et L_∞. En comparant les Figures 2.8, 2.12, et 2.13, où on applique la classification hiérarchique avec les mêmes paramètres, sauf la distance, on peut remarquer que les résultats sont assez similaires. Nous décidons de choisir la distance euclidienne.

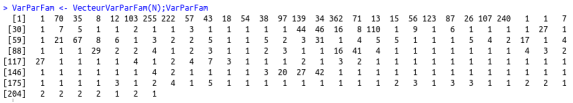

FIGURE 2.12 – Distance L_1

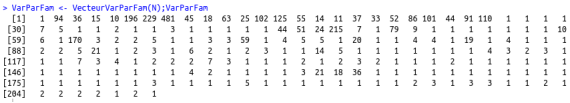

FIGURE 2.13 – Distance euclidienne

Par la suite, nous allons voir quel choix de nombre de groupes donne des résultats optimaux, c'est-à-dire les meilleurs R^2 et MSE, en fonction de la méthode utilisée. Nous allons donc appliquer les différentes méthodes que nous avons implémenter : la sélection par corrélation la plus grande, la sélection par contribution la plus grande (ACP), et la sélection par composantes principales (ACP) que nous détaillerons dans la partie 2.2.

2.1.3 Familles du spécialiste

Après avoir échangé avec le spécialiste et dans le but d'avoir des résultats corrects sur le plan chimique, le spécialiste nous a donné les familles qu'il à créé sur la base des propriétés physico-chimiques des différentes variables. Les familles du spécialiste regroupent donc des variables dont les propriétés sont proches chimiquement parlant mais qui ne le sont pas forcément statistiquement parlant. Le spécialiste nous a envoyé une table contenant les noms de 5270 variables, leur signification et leur famille. Ce nombre de 5270 variables nous a surpris étant donné que dans notre table nous ne disposons que de 5255 variables. Nous avons donc supprimé les variables inconnues puis nous avons repris cette table et avons créé la fonction `FamilleSpecialiste` pour créer une table contenant les noms des variables regroupées en familles comme nous avons pu le faire précédemment. Au final nous sommes arrivés à 30 familles distinctes dont voici les noms :

Constitutional indices	Ring descriptors	Topological indices
Walk and path counts	Connectivity indices	Information indices
2D matrix-based descriptors	2D autocorrelations	Burden eigenvalues
P_VSA-like descriptor	ETA indices	Edge adjacency indices
Geometrical descriptors	3D matrix-based descriptors	3D autocorrelations
RDF descriptors	3D-MoRSE descriptors	WHIM descriptors
GETAWAY descriptors	Randic molecular profiles	Functional group counts
Atom-centred fragments	Atom-type E-state indices	CATS 2D
2D Atom Pairs	3D Atom Pairs	Charge descriptors
Molecular properties	Drug-like indices	CATS 3D

Le chiffre de 30 familles nous à d'abord largement étonné, là où nous avons réussi à nous ramener à 433 familles en nous basant sur le nom des variables, qui pour certaines, comme on a pu le voir plutôt, ne sont pas très bien constituées d'un point de vue statistique, le spécialiste a plus de 10 fois moins de familles que nous. En regardant les familles du spécialiste, nous nous sommes aperçus que les familles sont très inégalitaires car le nombre d'individus par famille peut varier du simple au centuple. Ensuite, comme nous l'avons fait pour nettoyer la table, nous avons retiré les variables de variance nulle et nous avons regardé la composition des nouvelles familles. Nous nous sommes alors rendus compte que la famille "Charge descriptors" avait complètement disparue. La répartition des variables reste toutefois inégalitaire, les familles comptant entre 15 et 600 variables.

Nous avons dans un second temps vérifié la composition de ces familles pour voir s'il y avait des similarités avec celles que nous avons constituées. A première vue, il semble que le regroupement du spécialiste soit proche du notre en ce sens que lorsque nous avons regroupé deux variables, le spécialiste les a aussi regroupées. Cependant, les regroupements du spécialiste sont beaucoup plus larges, en effet, beaucoup des familles que nous avons crées semblent être regroupées sous de très grosses familles par le spécialiste comme on peut le voir Figure 2.14 et 2.15. En effet, on peut voir que le spécialiste regroupe dans une seule famille 1596 variables que nous séparons en 37 familles.

```
 [1] "T(N..N)"   "T(N..O)"   "T(N..S)"   "T(N..P)"   "T(N..F)"   "T(N..Cl)"  "T(N..Br)"  "T(N..I)"   "T(O..O)"   "T(O..S)"
[11] "B01[C-P]"  "B01[C-F]"  "B01[C-Cl]" "B01[C-Br]" "B01[C-I]"  "B01[C-B]"  "B01[C-Si]" "B01[C-X]"  "B01[N-N]"  "B01[N-O]"
[21] "B02[C-Cl]" "B02[C-Br]" "B02[C-I]"  "B02[C-B]"  "B02[C-Si]" "B02[C-X]"  "B02[N-N]"  "B02[N-O]"  "B02[N-S]"  "B02[N-P]"
[31] "B03[C-I]"  "B03[C-B]"  "B03[C-Si]" "B03[C-X]"  "B03[N-N]"  "B03[N-O]"  "B03[N-S]"  "B03[N-P]"  "B03[N-F]"  "B03[N-Cl]"
[41] "F01[C-P]"  "F01[C-F]"  "F01[C-Cl]" "F01[C-Br]" "F01[C-I]"  "F01[C-B]"  "F01[C-Si]" "F01[C-X]"  "F01[N-N]"  "F01[N-O]"
[51] "F02[C-Cl]" "F02[C-Br]" "F02[C-I]"  "F02[C-B]"  "F02[C-Si]" "F02[C-X]"  "F02[N-N]"  "F02[N-O]"  "F02[N-S]"  "F02[N-P]"
```

FIGURE 2.14 – Famille 2D Atom Pairs du spécialiste

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]	[,11]	[,12]
[1,]	"T(N..N)"	"T(O..O)"	"T(S..S)"	"T(F..F)"	"B01[C-O]"	"B02[C-F]"	"B02[O-F]"	"B03[C-Cl]"	"B03[O-F]"	"B03[F-Cl]"	"B04[C-F]"	"B04[O-F]"
[2,]	"T(N..O)"	"T(O..S)"	"T(S..F)"	"T(F..Cl)"	"B01[C-F]"	"B02[C-Cl]"	"B02[F-F]"	"B03[N-N]"	"B03[S-F]"	NA	"B04[C-Cl]"	"B04[S-F]"
[3,]	"T(N..S)"	"T(O..F)"	"T(S..Cl)"	NA	"B01[C-Cl]"	"B02[N-N]"	"B03[C-F]"	"B03[N-O]"	NA	NA	"B04[N-O]"	"B04[F-F]"
[4,]	"T(N..F)"	"T(O..Cl)"	NA	NA	"B01[N-N]"	"B02[N-S]"	NA	"B03[N-S]"	NA	NA	"B04[N-S]"	"B05[C-F]"
[5,]	"T(N..Cl)"	NA	NA	NA	NA	"B02[N-Cl]"	NA	"B03[N-F]"	NA	NA	"B04[N-F]"	NA
[6,]	NA	NA	NA	NA	NA	NA	NA	"B03[N-Cl]"	NA	NA	"B04[N-Cl]"	NA
[7,]	NA	NA	NA	NA	NA	NA	NA	"B03[O-O]"	NA	NA	"B04[O-O]"	NA
[8,]	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	"B04[O-S]"	NA

FIGURE 2.15 – Familles n° 366 à 377 basées sur les noms

Pour comparer la cohérence statistique des familles du thésard avec celle que nous avions constituées selon les noms, nous avons réalisé des ACP dans ces familles. Comme pour nos familles, nous avons constaté que certaines familles étaient très bien constituées, c'est le cas pour la famille "Randic molecular profiles" dans laquelle la première composante principale compte pour 92% de l'inertie, Figure 2.16. Tandis que d'autres sont très mal constituées, c'est le cas pour la famille "Atom-centred fragments" dans laquelle la première composante principale compte pour 12% de l'inertie, Figure 2.17.

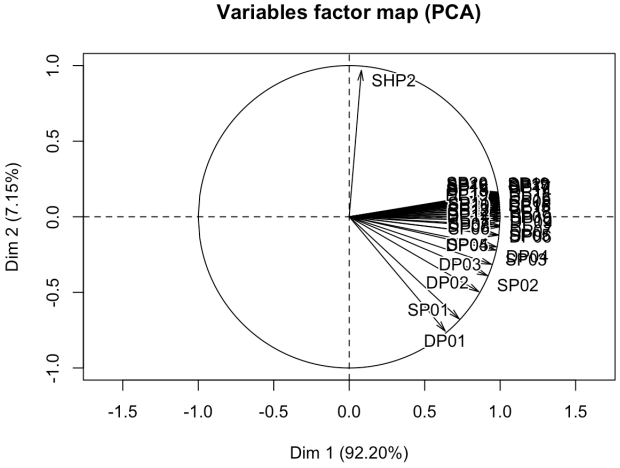

FIGURE 2.16 – ACP dans la famille Randic

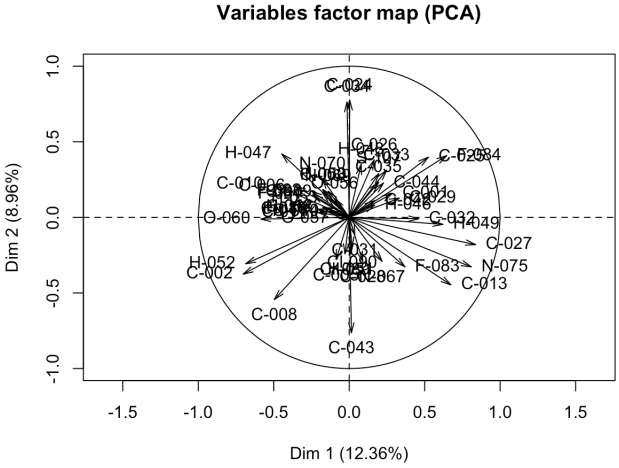

FIGURE 2.17 – ACP dans la famille Atom

Nous travaillerons dans la suite sur les données du spécialiste en considérant les résultats obtenus comme des résultats cible, qu'on aimerait, si possible pouvoir améliorer.

2.2 Autres Fonctions

VecteurVarParFam

Cette fonction, dont le principe est assez simple, a pour but de savoir combien d'éléments comporte chaque famille. Ceci nous sera très utile par la suite, en effet, elle nous permettra de savoir si nos familles sont plutôt cohérentes, et nous l'utilisons dans beaucoup d'autres fonctions.

ElectionChefParCorr

L'idée derrière la création était de regrouper des variables de sens chimique proche pour ensuite pouvoir choisir un représentant de chaque famille et pour travailler sur les représentants élus. La première idée qui nous est spontanément venue est de choisir dans chaque famille la variable la plus corrélée avec *Activity* puis de travailler sur ce sous groupe de variables. La sélection des variables les plus corrélées avec l'activité de fait grâce à la fonction `ElectionChefParCorr`.

CompPrincFamille

Pour sélectionner et discriminer les familles, nous nous sommes dit qu'un moyen assez neutre de différencier les familles serait de faire une ACP dans chaque famille et de travailler sur les premières composantes principales de chaque famille. Nos familles étant plutôt représentatives des variables qu'elles regroupent, pour beaucoup d'entre elles, la première composante principale contient une très grande partie de l'inertie et donc ne conserver que la première composante principale semble suffisant à bien représenter une famille. Cependant, pour les familles les moins bien constituées, pour lesquelles les variables sont assez hétérogènes, ne garder que la première composante principale peut être réducteur. Pour ces familles, il peut donc être intéressant de conserver les deux premières composantes principales. Nous avons donc décidé de créer la fonction `CompPrincFamille` qui, pour un seuil donné, va extraire, pour chaque famille, le premier facteur principal, et dans le cas où la proportion d'inertie expliquée par le second facteur dépasse le seuil, alors celui ci est conservé aussi.

SelectionParACP

Le problème de travailler avec les facteurs principaux est qu'ils peuvent être compliqués à interpréter. D'autant plus que dans notre cas les variables sont des données chimiques, expliquer une composante principale par famille serait donc compliqué étant donné que nous ne sommes pas en mesure de comprendre tous les composants chimiques inclus dans la table. Une autre manière d'utiliser l'ACP pour discriminer les famille est d'utiliser la contribution des variables. En effet, comme nous l'avons vu dans le Chapitre 1, dans une ACP la contribution et la qualité de sa représentation au sein du nuage peut-être quantifiée, de manière neutre. Nous sommes alors partis du même principe que précédemment, c'est-à-dire de regarder l'inertie expliquée par la première composante et par la seconde si celle-ci dépasse un seuil donné. Seulement, cette fois ci, au lieu de retenir les composantes principale, nous garderons la variable qui a la contribution la plus élevée sur cet axe. Dans le cas particulier où une même variable aurait la plus grande contribution sur les deux premières composantes principales, alors on gardera cette variable et la variable ayant la seconde plus forte contribution sur le second axe. Cette opération est réalisée par la fonction `SelectionParACP` que nous avons créé pour l'occasion.

Famille_Kmeans et Famille_CHA

Ces fonctions appliquent la classification K-means, et hiérachique ascendante à notre table, et créént une matrice contenant chaque famille. Elles utilisent les fonction `hclust` et `kmeans` du package R `stats`

Choixnbgroupes

Dans le cas des familles créés par la classification, nous avons besoin d'une fonction qui nous permettrait de comparer les résultats en fonction du nombre de groupes/familles que l'on souhaite obtenir au final. La fonction `Choixnbgroupes`, permet de calculer le MSE associé au modèles construits avec un nombre de groupe variant entre 110 et 400, et le nombre de variables retenues, dans le but de trouver l'endroit où l'arbre est optimal. Cette fonction commence par appliquer la classification hiérarchique ascendante avec le critère de regroupement de classes visant à minimiser l'inertie intra-classe, ou la méthode K-means. Ensuite, une fois les familles crées, nous sélectionnons des variables dans chaque classe créée grâce aux fonctions `ElectionParCorrelation`, `SelectionParACP`, et `CompPrincFamille` , puis applique la méthode LASSO sur les variables restantes pour obtenir un modèle final.

LASSO

Pour utiliser le LASSO, nous avons choisi d'utiliser le package `R` nommé `glmnet` [3]. Les deux fonction que nous utiliserons principalement sont `glmnet` et `cv.glmnet`.
La fonction `cv.glmnet` réalise des validations croisées pour trouver le $\lambda_{optimal}$. La k-validation croisée est une méthode d'apprentissage automatique qui consiste à diviser les données en k échantillons d'apprentissage. La fonction va alors calculer le $\lambda_{optimal}$ de chaque échantillon d'apprentissage et tester le modèle obtenu avec ce lambda sur les autres échantillons et calculer l'erreur d'apprentissage. La fonction va réaliser cette opération pour chacun des k échantillons puis elle va renvoyer le $\lambda_{optimal}$ pour lequel l'erreur de prédiction est minimale. Dans notre cas, nous prendrons $k = 10$.
La fonction `glmnet` calcule un modèle linéaire par maximum de vraisemblance pénalisé. La pénalisation est une pénalisation dite elastic-net définie comme suit : $(1-\alpha)\|\beta\|_2^2 + \alpha\|\beta\|_1$. Cette pénalisation est un mélange de la pénalisation de LASSO (pour $\alpha = 1$) et de Ridge (pour $\alpha = 0$). Dans notre cas nous prendrons $\alpha = 1$ car c'est la pénalisation de LASSO qui nous intéresse. Cette fonction donne la possibilité de donner le λ en paramètre de la fonction.

Nous avons donc dans un premier temps décidé de lancer `cv.glmnet` puis de prendre le $\lambda_{optimal}$ trouvé par cette fonction pour ensuite lancer `glmnet` avec le paramètre obtenu. Cependant au cours des différents lancés du LASSO, nous nous sommes rendus compte que les résultats étaient peu stables. L'instabilité des résultats était en réalité due à la fonction `cv.glmnet` car celle-ci, lorsqu'elle effectue la validation croisée choisit les échantillons au hasard, le choix du $\lambda_{optimal}$ et surtout les résultats obtenus sont donc variables. Pour supprimer cette variabilité, nous avons créé notre propre fonction `LASSO` (voir en Annexe) qui lance 30 fois de suite la fonction `cv.glmnet` en stockant à chaque fois le $\lambda_{optimal}$ obtenu et qui lance pour finir la fonction `glmnet` en prenant pour λ la valeur médiane des $\lambda_{optimal}$ obtenus par la fonction précedente. Nous avons ajouté au résultat de cette fonction deux mesures d'ajustement : le MSE et le MSELASSO comme on les a défini dans la Partie 1.5.

Chapitre 3

Application et Résultats

Pour rappel, nous cherchons une relation du type : $Y = X\beta + \epsilon$, où Y représente l'activité chimique du VHC, X la matrice des différentes mesures chimiques et ϵ l'erreur de prédiction. L'objectif est de trouver un β qui permettrait d'obtenir un modèle le plus explicatif possible. Notre problème est dit de grande dimension, c'est-à-dire que le nombre d'individus est largement inférieur au nombre de variables explicatives. Dans ce cas, on souhaiterait appliquer des méthodes adaptées à ce problème, c'est-à-dire des méthodes tels que LASSO ou la sélection stepwise sur des données regroupées en familles.

Dans un premier temps, nous allons effectué une ACP et une sélection LASSO sur l'ensemble des données ce qui nous permettra d'avoir une vue d'ensemble des données, de savoir combien de variables pourraient être susceptibles d'être retenues, et enfin d'avoir une base de départ sur laquelle se fixer. En effet, l'objectif principal est d'obtenir de meilleurs résultats après notre travail sur la table, que les résultats obtenus sur les données brutes.

Nous travaillons sur différentes familles de variables. Dans le but d'optimiser la table, et vu que certaines variables peuvent être considérées comme similaires du point de vue chimique, nous souhaitions créer des "familles" de variables, pour pouvoir ensuite faire une première sélection de variables à l'intérieur de ces familles. Nous avons obtenus trois différents regroupements de variables :

- les familles que nous avons créées en fonction des noms des variables.
- les familles issues des méthodes de classification (CHA et K-means).
- les familles obtenues par le spécialiste.

Nous allons appliquer différentes méthodes de sélection de variables à chacun de ces regroupements de variables. Ces méthodes sont :

- une sélection de variables par corrélation dans chaque famille, grâce à la fonction `ElectionChefParCorr`, puis parmi les variables restantes, on sélectionne un modèle via la sélection LASSO ou Stepwise.
- une sélection de variables par composantes principales dans chaque famille, grâce à la fonction `CompPrincFamille`, puis parmi les variables restantes, on sélectionne un modèle via la sélection LASSO ou Stepwise.
- une sélection de variables par meilleure contribution aux axes principaux de l'ACP de chaque famille, grâce à la fonction `SelectionParACP`, puis parmi les variables restantes, on sélectionne un modèle via la sélection LASSO ou Stepwise.

3.1 Table de données initiales

3.1.1 ACP sur la table brute

Dans un premier temps, nous allons réaliser une ACP globale, et appliquer la méthode LASSO sur l'ensemble des données.

Vous trouverez ci-dessous le diagramme des valeurs propres de la table des 3193 variables non normalisées, Figure 3.1, ainsi que la courbe représentant les valeurs propres cumulées, Figure 3.2. Ces résultats semblent très anormaux, en effet, seuls deux axes suffisent à expliquer plus de 95% de l'inertie totale alors qu'on a à disposition 3193 variables différentes. Le problème est en réalité que les variables sont exprimées selon des échelles très différentes, certaines prennent des valeurs plutôt petites comme c'est le cas pour la variable GD qui varie entre 0,06 et 0,079 alors que d'autres prennent des très grande valeurs comme c'est le cas pour $ZM2Per$ qui varie entre 748.4 et 1127.3. Dans le but de ne pas laisser ces différences influer sur le résultat et fausser l'ACP, nous choisissons de normaliser les données. C'est-à-dire que nous avons retranché à chaque variable sa moyenne et nous l'avons divisée par sa variance, pour que les variables soient mesurées sur une même échelle. La Figure 3.3 est le diagramme des valeurs propres après normalisation, et la Figure 3.4 représente le pourcentage cumulé d'inertie expliquée. Après avoir normalisé les données, le diagramme est plus logique, si on fixe un seuil de pourcentage d'inertie expliquée à 80%, on choisirait de garder les 12 premières composantes principales.

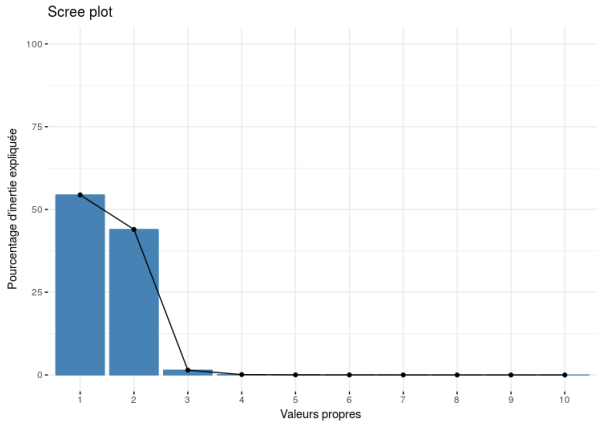

FIGURE 3.1 – Diagramme des valeurs propres

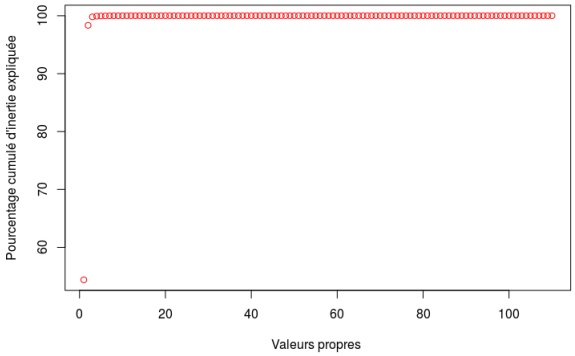

FIGURE 3.2 – Graphique des valeurs propres cumulées

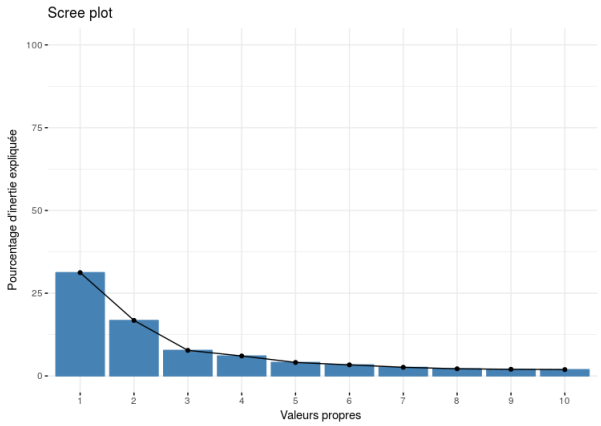

FIGURE 3.3 – Diagramme des valeurs propres avec normalisation

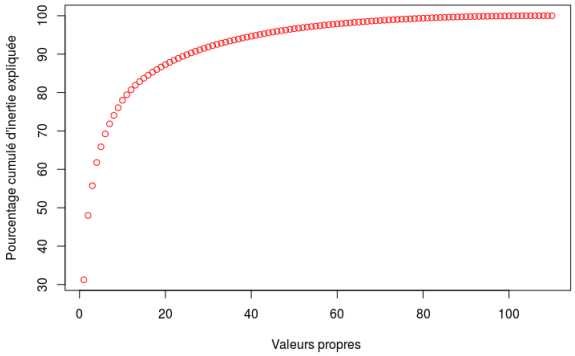

FIGURE 3.4 – Graphique des valeurs propres cumulées avec normalisation

Étant donné que les composantes principales synthétisent l'information du nuage de points, nous avons réalisé une régression linéaire multiple sur ces composantes principales. Nous avons choisit de garder 12 composantes pour expliquer au moins 80 % de l'inertie. Le modèle obtenu est le suivant :

Axe	Coefficient
Axe n° 1	2.627132e-04
Axe n° 2	1.318871e-01
Axe n° 3	-8.915024e-01
Axe n° 4	-1.075686e+01
Axe n° 5	-3.824694e-03
Axe n° 6	8.887033e-04
Axe n° 7	1.229414e-03
Axe n° 8	-1.072665e+00
Axe n° 9	-5.777214e-02
Axe n° 10	-9.995229e-02
Axe n° 11	-1.443925e-03
Axe n° 12	-1.826669e-01

Pour ce modèle, les valeurs d'ajustement sont les suivantes : $R^2 = 0,3856$, $R^2_{ajusté} = 0,3104$ et $MSE = 0,7341$

3.1.2 Résulats de la sélection par LASSO et par Stepwise sur toute la table

Notre objectif étant de construire un modèle minimisant le risque quadratique moyen (MSE) tout en gardant le moins de variables possible, nous avons commencé par appliquer brutalement le LASSO à la table de données non normalisée. Le résultats du LASSO sont les suivants, on peut voir Figure 3.5 l'histogramme des différentes valeurs de λ à chaque lancée de la fonction `cv.glmnet` ainsi que la médianne de cet échantillon que l'on à pris pour $\lambda_{optimal}$. Sur la figure 3.6 on voit l'évolution de la valeur du MSE selon les différents λ testés par la fonction `cv.glmnet`.

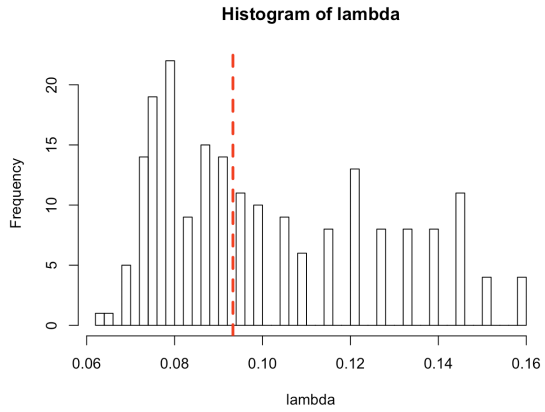

FIGURE 3.5 – Histogramme des valeurs de λ

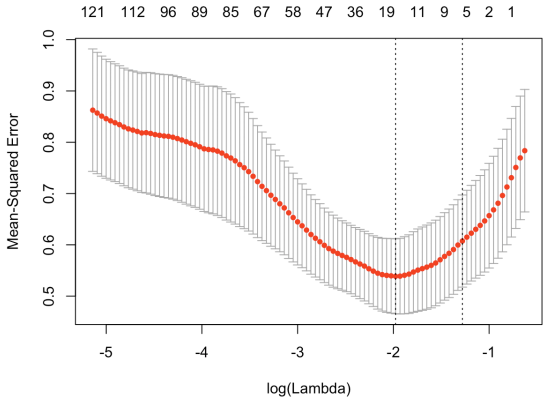

FIGURE 3.6 – Evolution du MSE selon les valeurs de λ

On obtient pour un $\lambda_{optimal} = 0,0932$ un modèle à 37 variables pour un $MSE = 0,2816$ et $MSELASSO = 2,0431$. Le tableau suivant donne le modèle obtenu avec les variables sélectionnées et les coefficients qui leurs sont associés.

Variable conservée	Coefficient	Variable conservée	Coefficient
$D/Dtr05$	2.627132e-04	$MATS5p$	3.203728e+00
$VE3sign\_Dz(i)$	1.318871e-01	$GATS4m$	9.966767e-02
$VE1\_B(m)$	-8.915024e-01	$TDB09i$	1.603143e-01
$VE2sign\_B(m)$	-1.075686e+01	$RDF125m$	1.990330e-03
$RDF150e$	-3.824694e-03	$E1u$	9.676666e-01
$RDF085s$	8.887033e-04	$H4s$	3.732696e-02
$RDF125s$	1.229414e-03	$H6s$	3.221013e-02
$G1u$	-1.072665e+00	$R4s$	1.692580e-02
$nCXr$	-5.777214e-02	$H-054$	-2.598272e-07
$nHBonds$	-9.995229e-02	$CATS2D\_02\_AL$	-6.429872e-02
$C-012$	-1.443925e-03	$CATS2D\_07\_AL$	-2.000986e-02
$H-053$	-1.826669e-01	$B03[F-Cl]$	-9.759212e-02
$B04[N-Cl]$	-8.092785e-02	$F03[F-Cl]$	-7.607448e-03
$B04[F-F]$	2.624247e-01	$F04[N-Cl]$	-2.447103e-04
$B05[S-F]$	-1.133001e-05	$F04[F-F]$	7.651131e-05
$F03[N-O]$	-1.465243e-01	$F05[S-F]$	-4.974816e-06
$F06[O-F]$	-1.615654e-16	$CATS3D\_14\_DL$	-3.103798e-01
$CATS3D\_10\_DA$	-4.100506e-02	$CATS3D\_11\_AA$	-1.572101e-01
$CATS3D\_07\_AL$	-1.808009e-02		

Nous avons également effectué une régression linéaire avec la méthode stepwise. Pour cela, nous avons choisit de prendre comme seuil d'entrée 0.05 et comme seuil de sortie 0.1. Vous trouverez ci-dessous les 21 variables sélectionnées dans le modèle final, ainsi que leur coefficient. Le R^2 associé au modèle est 0.8613, le R^2 ajusté est 0.8305 et le MSE est de 0.36398.

Variable conservée	Coefficient	Variable conservée	Coefficient
$Intercept$	0.184478	$J\_Dz\_p\_$	4.645922
$VE1\_B\_i\_$	-2.511472	$MATS5p$	8.505981
$MATS1s$	31.184218	$SpMin7\_Bh\_v\_$	4.671517
$SM14\_AEA\_ri\_$	1.928203	$Eig11_EA_bo$	-1.769249
$RDF125m$	0.093445	$HATS5e$	5.554832
$R6s$	0.531242	$nHBonds$	-1.096226
C_029	-0.252482	H_053	-1.341535
$SsCH3$	-0.161969	$CATS2D\_07\_DL$	0.655175
$B04[N\check{}C]$	-0.658700	$F06[N\check{}F]$	0.342892
$CATS3D\_10\_AA$	-0.662057	$CATS3D\_07\_AL$	-0.114784
$CATS3D\_07\_AL$	0.224991		

Après avoir normalisé les données, et en effectuant les mêmes démarches, nous obtenons des résultats similaires. Le LASSO sélectionne 37 variables avec un $MSE = 0,3402$ et $MSELASSO = 0,3061$, les résultats du stepwise sont identiques aux précédents mis à part les coefficients des variables sélectionnées.

3.2 Chefs de familles

Dans cette partie nous allons travailler avec la fonction `ElectionChefParCorr`. Nous allons choisir dans chaque famille quelle est la variable la plus corrélée avec l'activité chimique et nous allons appliquer nos méthodes de sélection de variable sur les variables sélectionnées.

3.2.1 Nos familles

Stepwise

Pour pouvoir donner un modèle a partir de ces données, nous avons effectué une sélection par stepwise en choisissant des paramètres d'entrée et de sortie assez restrictifs ($\alpha_E = 0.05$ et $\alpha_S = 0.05$). Nous avons obtenu le modèle suivant :

Variable conservée	Coefficient	Variable conservée	Coefficient
Intercept	33.360468	$HOMA$	-21.749734
$ZM2MulPer$	-0.019504	$H4s$	0.189998
$PJI2$	-2.575100	$C-025$	-0.335409
$VE1sign\_Dz(i)$	3.267887	$CATS2D\_05\_AL$	0.150573
$MATS5p$	11.158950	$T(O..F)$	-0.007414
$SM03_EA(dm)$	0.480956	$T(F..F)$	0.020315

Nous obtenons un modèle à 12 variables pour lequel le $R^2 = 0,6705$, $R^2_{ajusté} = 0,6339$ et $MSE = 0,5348$.

LASSO

Ensuite nous avons appliqué la sélection par LASSO aux variables sélectionnées et avons obtenu le modèle suivant :

Variable conservée	Coefficient	Variable conservée	Coefficient
$VE1sign\_D$	0.7936335	$R4s$	0.05680229
$MATS5p$	3.490576	$nCH2RX$	0.01471668
$SpMax2\_Bh(p)$	12.96068	$T(F..F)$	0.000034904
L/Bw	-0.02113976	$B03[F-Cl]$	-0.353958
$CMBL$	0.02890916	$F03[F-Cl]$	-1.9969×10^{-13}
$E1u$	1.075860	$F05[N-F]$	0.0815914
$G1m$	8.712239	$G(S..F)$	0.00272815
$H4s$	0.04428144	Hy	-0.424315
$HATS4s$	0.009797210	DLS_02	0.155740
$R8v+$	65.66817		

Nous obtenons un modèle à 19 variables pour lequel le $MSE = 0,3606$ et $MSELASSO = 7,2823$.

La selection par stepwise est satisfaisante car elle conserve peut de variables cependant son MSE est plutôt élevé. La selection par LASSO arrive à un MSE intéressant cependant elle garde beaucoup de variables.

3.2.2 Familles du spécialiste

Dans cette partie, nous allons appliquer la sélection par corrélation, puis un stepwise. En effet, en sélectionnant une, ou même deux variable dans chaque famille, on se retrouve dans un cas plus basique où le nombre d'individus est supérieure au nombre de variables explicatives, et donc le LASSO n'est plus une méthode adaptée.

La sélection par corrélation nous laisse 29 variables explicatives. En effet, cette méthode de sélection ne nous permet pas de sélectionner plusieurs variables par famille, car sinon l'information deviendrait redondante. Nous obtenons les résultats suivants après le stepwise : $R^2 = 0.5455$, $R^2_{ajusté} = 0.505$, et $MSE = 0.6219$, avec 10 variables. On obtient un modèle avec peu de variables, mais assez peu explicatif. Vous trouverez un récapitulatif des variables sélectionnées et de leur coefficients sur le tableau suivant :

Variable conservée	Coefficient	Variable conservée	Coefficient
Intercept	-17.7360	$R4s$	0.1973
$MATS5p$	7.6293	$DP01$	-3.9315
$SpMax7\_Bh(s)$	-0.3828	$G(F..F)$	0.0188
$HOMT$	-0.9184	Uc	13.0134
$RDF050m$	0.0809	$CATS3D\_10\_DL$	0.3055

3.2.3 Familles issues de classification

CHA

Maintenant, nous cherchons quel est le nombre de classes qui nous donnera le meilleur MSE, dans le cas de la sélection par corrélation. Sur la Figure 3.7, vous trouverez sur la première ligne les différents choix de nombre de groupes, puis sur la deuxième les MSE associés à différents choix, et sur la troisième le nombre de variables sélectionnées dans le modèle.

```
          [,1]       [,2]       [,3]       [,4]       [,5]       [,6]       [,7]       [,8]       [,9]      [,10]
[1,] 110.0000000 120.0000000 130.0000000 140.0000000 150.0000000 160.000000 170.0000000 180.0000000 190.0000000 200.0000000
[2,]   0.3940139   0.3832671   0.3889367   0.3820371   0.3823049   0.382279   0.3838462   0.3838473   0.3852956   0.3822031
[3,]   8.0000000   7.0000000   6.0000000   7.0000000   8.0000000   9.000000   9.0000000   9.0000000   9.0000000  10.0000000
          [,11]      [,12]      [,13]      [,14]      [,15]      [,16]      [,17]      [,18]      [,19]      [,20]
[1,] 210.0000000 220.0000000 230.0000000 240.000000 250.0000000 260.0000000 270.0000000 280.0000000 290.0000000 300.0000000
[2,]   0.3840744   0.3566015   0.3589164   0.358906   0.3585898   0.3585898   0.3611482   0.3520466   0.3513341   0.3556551
[3,]  10.0000000  12.0000000  12.0000000  12.000000  13.0000000  13.0000000  13.0000000  14.0000000  14.0000000  14.0000000
          [,21]      [,22]      [,23]      [,24]      [,25]      [,26]      [,27]      [,28]      [,29]      [,30]
[1,] 310.0000000 320.0000000 330.000000 340.0000000 350.0000000 360.0000000 370.0000000 380.0000000 390.0000000 400.000000
[2,]   0.3416699   0.3089327   0.294213   0.3114733   0.3009193   0.3461353   0.3461773   0.3504786   0.3479983   0.345636
[3,]  14.0000000  18.0000000  19.000000  18.0000000  21.0000000  13.0000000  13.0000000  14.0000000  15.0000000  15.000000
```

FIGURE 3.7 – Choix du nombre de groupes

Le but étant de minimiser le MSE, on pourrait décider de choisir soit un nombre de groupe variant entre 310 et 350. Nous pourrions choisir de retenir 330 groupes, car c'est le nombre de groupes qui nous permet d'avoir un MSE relativement stable, et le plus faible possible. Dans ce cas, l'erreur quadratique moyenne varie autour de 0.2888339, et elle est associé à un modèle contenant 19 variables. Cependant, nous choisissons un nombre de groupes égal à 360, car c'est le nombre de groupes qui nous permet de faire un bon compromis entre un MSE bas et un nombre de variable faible (MSE entre 0.34 et 0.35, et environ 13 variables sélectionnées).

Sur les Figures 3.8 et 3.9, vous trouverez des exemples de familles créés avec la classification hiérarchique, en coupant le dendogramme avec pour but d'avoir 360 classes de variables.

[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]
"S2K"	"PHI"	"PW3"	"MAXDN"	"DELS"	"TIE"	"Psi_i_s"	"Psi_e_A"
"MWC06"	"MWC03"	"X1A"	"SRW04"	NA	NA	"SpAbs_B(s)"	"MWC05"
"SRW08"	"piPC05"	"X2A"	"MPC08"	NA	NA	"SpPos_B(s)"	"piPC08"
"piPC10"	"WiA_D"	"X1Av"	"MPC09"	NA	NA	NA	"piPC09"
"X5"	"SpPosA_D"	"Xindex"	"MPC10"	NA	NA	NA	"TPC"
"X5sol"	"SpMaxA_D"	"SIC0"	"piPC04"	NA	NA	NA	"X1Per"
"IDM"	"SpPosLog_L"	"BIC0"	"SpMax_L"	NA	NA	NA	"SM4_L"
"Chi_H2"	"SpDiam_H2"	"VE1sign_D"	"SpDiam_L"	NA	NA	NA	"SM5_H2"
"SM6_H2"	"SM4_H2"	"SpPosA_X"	"SM2_L"	NA	NA	NA	"SpMAD_Dz(Z)"
"SpPosLog_Dt"	"SM2_D/Dt"	"ChiA_H2"	"VE1_L"	NA	NA	NA	"SpMAD_Dz(m)"

FIGURE 3.8 – Exemple de familles

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]	[,11]
1,]	"RDF135m"	"RDF040i"	"RDF045i"	"RDF090i"	"RDF020s"	"RDF025s"	"RDF030s"	"RDF035s"	"RDF040s"	"RDF045s"	"RDF050s"
2,]	"RDF115v"	"RDF055i"	NA	"Mor02v"	NA	NA	NA	NA	NA	NA	NA
3,]	"RDF120v"	"RDF065i"	NA	"Mor02p"	NA	NA	NA	NA	NA	NA	NA
4,]	"RDF125v"	"RDF070i"	NA	"L1u"	NA	NA	NA	NA	NA	NA	NA
5,]	"RDF115p"	NA	NA	"L1e"	NA	NA	NA	NA	NA	NA	NA
6,]	"RDF120p"	NA	NA	"L1i"	NA	NA	NA	NA	NA	NA	NA
7,]	"RDF125p"	NA	NA	"L1s"	NA	NA	NA	NA	NA	NA	NA
8,]	"Mor06m"	NA	NA	"Tm"	NA	NA	NA	NA	NA	NA	NA
9,]	"Mor04i"	NA	NA	"Tv"	NA	NA	NA	NA	NA	NA	NA
0,]	"nCs"	NA	NA	"Tp"	NA	NA	NA	NA	NA	NA	NA

FIGURE 3.9 – Exemple de familles

On remarque que les regroupements de la classification hiérarchique sont très différents des notre, et de ceux effectués par le spécialiste. Sur la Figure 3.9, on voit clairement que le regroupement n'est pas optimal, au vue des regroupements fait par le spécialiste.

K-means

Sur les Figures 3.10 et 3.11, vous trouverez une représentation des MSE obtenus avec différents choix de nombre de groupes, en éxécutant deux fois notre fonction `Choixnbgroupes` avec la sélection par corrélation (`ElectionChefParCorr`).

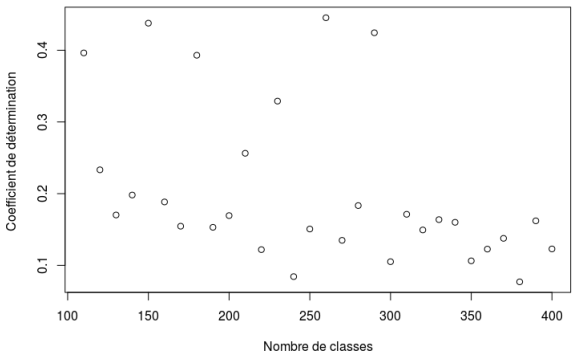

FIGURE 3.10 – MSE selon le nombre de groupes

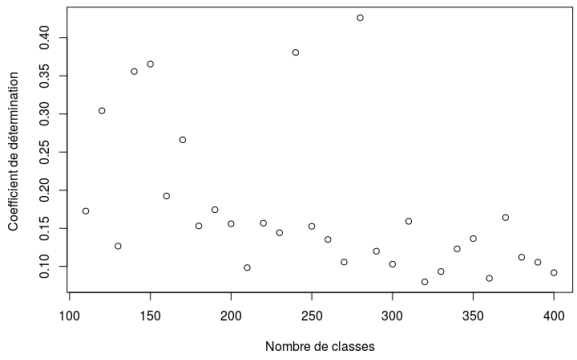

FIGURE 3.11 – MSE selon le nombre de groupes

Les deux graphiques affichent les mêmes tendances, ce qui correspond à nos attentes. En choisissant un nombre de groupes entre 300 et 400, on obtient un MSE plutôt stable inférieure à 0.2. Cependant, un appliquant la fonction `Famille_Kmeans` puis une sélection par corrélation et par LASSO, on obtient des MSE respectivement égaux à 0.1760214, 0.240137 et 0.2155977. Ces résultats sont plutôt ressemblant, et on pourrait attribuer une moyenne d'environ 0.2 comme

MSE, pour la classification K-means avec 350 groupes.

Ce MSE, même s'il varie légèrement, est meilleur que les résultats obtenus avec la classification hiérarchique ascendante. Cependant, on garde toujours au moins une trentaine de variables dans le modèle final.

Ensuite, nous avons appliqués l'algorithme K-means avec comme centres initiaux les centres de gravité des familles obtenus grâce à la méthode CHA. Sur les Figures 3.12 et 3.13, vous trouverez le nombre d'éléments par classe avant et après application de la méthode K-means.

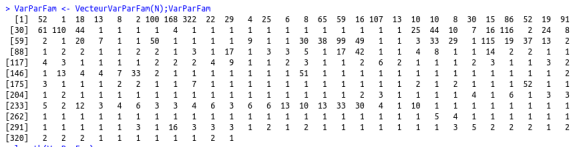

FIGURE 3.12 – Nombre d'éléments par famille obtenues par CHA

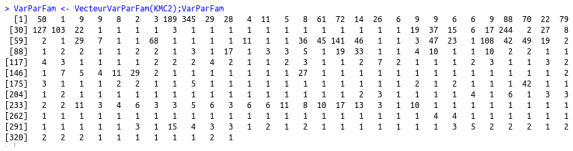

FIGURE 3.13 – Nombre d'éléments par famille obtenues par Kmeans

Les familles semblent sensiblement les mêmes, cependant, quand on applique ensuite la sélection par corrélation puis le LASSO, on obtient des résultats moins bons que ceux obtenus avec la CHA. En effet, le modèle sélectionné au final nous donne un MSE proche de 0.3511483, mais avec plus de variables (17 variables).

Le modèle final sélectionné dans cette partie est donc celui ci-dessous, issu de la classification hiérarchique ascendante, avec un MSE égal à 0.3421632, et 13 variables sélectionnées :

Variable conservée	Coefficient	Variable conservée	Coefficient
$MATS5p$	3.5511205512	$P\_VSA\_MR\_3$	-0.0012204074
$H4s$	0.0613845058	$RDF150i$	-0.0304528284
$D/Dtr05$	0.0022817005	$CATS3D\_13\_AL$	0.0214841533
$TDB05i$	0.9466104467	$RDF085s$	0.0009172892
$VE3sign\_Dz(i)$	0.2401428697	$RDF125s$	0.0037769427
$RDF130s$	0.0033913200	$T(N..O)$	-0.0047019691
$TPSA(NO)$	-0.0073981575		

3.3 ACP sur les familles : Composantes principales

3.3.1 Nos familles

En prenant un seuil à 25% d'inertie expliquée au minimum pour le second axe, il nous reste 570 composantes que nous traiterons de deux manière, d'une part, nous allons effectuer une procédure stepwise et d'autre part une procédure LASSO. Pour clarifier les résultats obtenus, nous nommerons les axes de la manière suivante : si la composante est une première composante d'une famille alors celle-ci porte le nom de la première variable de la famille et si la composante est une deuxième composante principale alors celle-ci porte le nom de la seconde variable de la famille.

Stepwise

Pour pouvoir donner un modèle à partir de ces données, nous avons effectué une sélection par stepwise en choisissant des paramètres d'entrée et de sortie assez restrictifs ($\alpha_E = 0.05$ et $\alpha_S = 0.05$). Nous avons obtenu le modèle suivant :

Axe conservé	Coefficient
Intercept	7.258780
$SpMax\_EA(bo)$	0.165010
$Cl - 089$	0.130303
$SssS$	-0.340183
$B04[S - F]$	-0.234565
$F06[O - F]$	0.181833
$G(S..S)$	0.204884

Nous obtenons un modèle à 7 variables pour lequel le $R^2 = 0.6705$, $R^2_{ajusté} = 0.6339$ et $MSE = 0.5348$.

LASSO

Nous avons ensuite appliqué la sélection de variable LASSO et avons obtenu les résultats suivants :

Axe conservé	Coefficient	Axe conservé	Coefficient
$H\%$	0.0516798	$SssS$	-0.028864
$VE1sign\_D$	0.0224129	$F04[S - F]$	0.033530
$H0v$	0.000538	$F05[C - C]$	0.079937
$nCar$	0.043498	$F10[C - C]$	0.026029
$sCRX3$	0.034949		

Nous obtenons un modèle à 9 variables pour lequel le $MSE = 0.4892$ et $MSELASSO = 0.2952$.

Les deux modèles obtenus par Stepwise et LASSO sont équivalents, ils conservent peut de variables cependant le MSE est assez élevé dans les deux cas.

3.3.2 Familles du spécialiste

Etant donné que les familles du spécialiste sont peu nombreuses, nous avons choist de conserver automatiquement les deux premières composantes principales pour chaque famille. La sélection dans chaque famille nous laisse 58 composantes principales. Nous obtenons les résultats suivants après le stepwise : $R^2 = 0.5020$, $R^2_{ajusté} = 0.4353$, et $MSE = 0.66432$, avec 27 variables. Vous trouverez un récapitulatif des variables sélectionnées et de leur coefficients dans le tableau suivant :

Axe conservé	Coefficient	Axe conservé	Coefficient
Intercept	7.2587	$ATS2m$	-0.1276
Sv	0.2531	$SpMax1\_Bh\_m\_$	0.1127
$nCIR$	0.7150	$SpMax2\_Bh\_m\_$	0.0464
$MWC01$	0.2516	Eta_alpha	-0.5915
$X0$	-0.5237	$SpMaxA\_EA$	0.0187
$X1$	0.1529	$G2$	-0.0694
J_A	0.1043	Wi_G	-0.1166

3.3.3 Familles issues de classification

Dans cette section, nous allons appliquer une sélection de variables par composantes principales, grâce à la fonction `CompPrincFamille`, dans chaque famille, puis nous aboutiront à un modèle final grâce à une sélection LASSO sur les variables restantes. Cette méthode sera effectué sur les familles issues de la classification hiérarchique ascendante, et de la méthode K-means.

CHA

Sur la Figure 3.14, vous trouverez les MSE et nombres de variables retenues associés à différents nombres de groupes voulues (obtenue grâce à la fonction `Choixnbgroupes`). Un choix de groupes entre 140 et 160 semble optimal. En regardant la Figure 3.14, on remarque qu'il serait préférable de choisir un nombre de groupes entre 310 et 350 pour satisfaire nos deux critères (MSE et nombre de variables faibles). En testant la fonction pour différents seuils d'inertie du deuxième axe, on retrouve des résultats similaires.

```
> n
          [,1]       [,2]       [,3]       [,4]       [,5]       [,6]       [,7]       [,8]       [,9]      [,10]
[1,] 110.0000000 120.0000000 130.0000000 140.0000000 150.0000000 160.0000000 170.0000000 180.0000000 190.0000000 200.0000000
[2,]   0.2997039   0.2991694   0.2720828   0.2338873   0.2309681   0.2354081   0.256087   0.3940483   0.3855076   0.3226465
[3,]  25.0000000  25.0000000  26.0000000  29.0000000  28.0000000  33.0000000  31.000000  14.0000000  15.0000000  22.0000000
          [,11]      [,12]      [,13]      [,14]      [,15]      [,16]      [,17]      [,18]      [,19]      [,20]
[1,] 210.0000000 220.0000000 230.0000000 240.0000000 250.0000000 260.0000000 270.0000000 280.0000000 290.0000000 300.0000000
[2,]   0.3854611   0.3856558   0.3831503   0.3869124   0.3843464   0.3839875   0.3890438   0.3826298   0.3933393   0.3955188
[3,]  17.0000000  15.0000000  17.0000000  15.0000000  17.0000000  17.0000000  17.0000000  17.0000000  15.0000000  16.0000000
          [,21]      [,22]      [,23]      [,24]      [,25]      [,26]      [,27]      [,28]      [,29]      [,30]
[1,] 310.0000000 320.0000000 330.0000000 340.0000000 350.0000000 360.0000000 370.0000000 380.0000000 390.0000000 400.0000000
[2,]   0.3869226   0.3869226   0.3894023   0.3869226   0.3894024   0.3893879   0.3924383   0.3809357   0.3973484   0.3997875
[3,]  15.0000000  15.0000000  15.0000000  15.0000000  15.0000000  16.0000000  16.0000000  18.0000000  17.0000000  17.0000000
> which.min(M[2,])
```

FIGURE 3.14 – Résultats de la fonction `Choixnbgroupes`

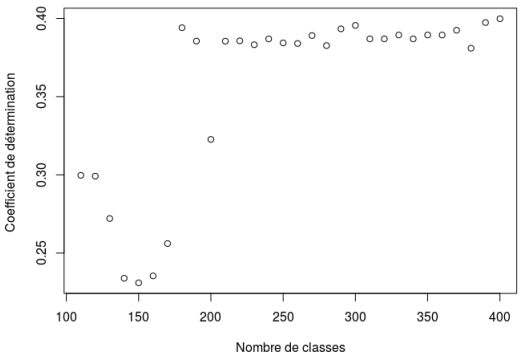

FIGURE 3.15 – MSE selon le nombre de groupes

Kmean

La Figure 3.16 et la Figure 3.17 représentent les MSE en fonction du nombre de groupes de deux exécutions du programme. On remarque bien que les schémas n'ont pas de tendance, les résultats sont très aléatoire. En choisissant un nombre de groupes égal à 300, on obtient un MSE variant ente 0.05 et 0.15, ce qui est très bien, et serait préférable pour nous, cependant, le nombre de variables contenus dans le modèle est compris entre 50 et 80, ce qui est très grand. On peut donc se poser la question suivante : préfères t-on avoir un modèle très explicatif mais coûteux, ou un modèle contenant le moins de variables possibles ?

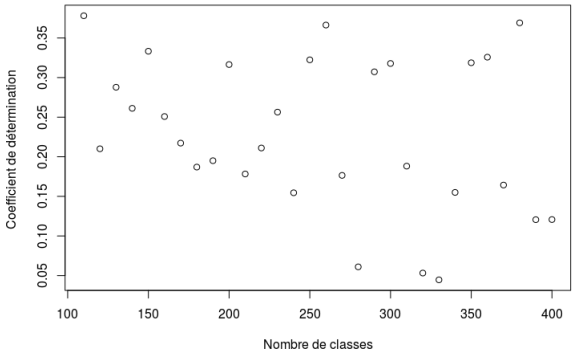

FIGURE 3.16 – MSE selon le nombre de groupes

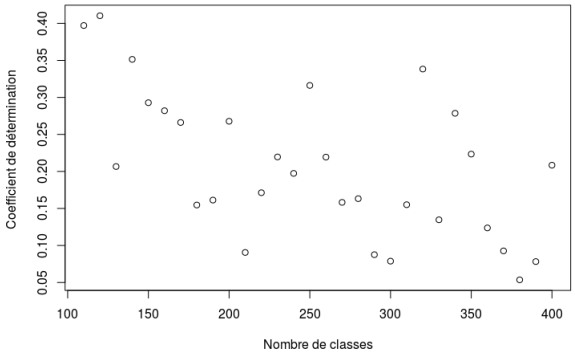

FIGURE 3.17 – MSE selon le nombre de groupes

Dans cette partie, nous sélectionnons donc le modèle issue de la classification hiérarchique, qui nous donne un MSE à 0.3969109 et 18 variables sélectionnées.

Axe conservé	Coefficient	Axe conservé	Coefficient
nO	-0.194213	$C\%$	0.08946679
$D/Dtr04$	-2.000557e-04	$D/Dtr05$	4.358862e-03
$VE3sign\_D$	13.913001	$ATSC4s$	0.802016
$MATS4m$	-3.255033e-01	$P\_VSA\_MR\_1$	10035678e-03
$P\_VSA\_e\_2$	4.959819e-03	$P\_VSA\_ppp\_ar$	6.425110e-05
$RDF150u$	-5.791137e-02	$RDF125m$	2.690267e-02
$RDF050s$	5.727953e-03	$RDF085s$	3.513686e-03
$RDF095s$	1.911732e-04	$RDF130s$	2.974802e-03
$T(O..O)$	-4.092941e-03	$T(F..F)$	9.490102e-04

3.4 ACP sur les familles : Variables à forte contribution

3.4.1 Nos familles

En prenant un seuil à 25% d'inertie expliquée au minimum pour le second axe, il nous reste, comme précédemment, 570 variables que nous traiterons de deux manières, d'une part, nous allons effectuer une procédure stepwise et d'autre part une procédure LASSO.

Stepwise

Pour pouvoir donner un modèle a partir de ces données, nous avons effectué une sélection par stepwise en choisissant des paramètres d'entrée et de sortie assez restrictifs ($\alpha_E = 0.05$ et $\alpha_S = 0.05$). Nous avons obtenu le modèle suivant :

Variable conservée	Coefficient	Variable conservée	Coefficient
Intercept	-60.370725	$SssCH2$	0.173740
$nR05$	0.585464	$SsNH2$	-0.347293
$piPC04$	13.913001	$CATS2D\_03\_DL$	0.302016
$X1sol$	-0.945784	$F04[F-F]$	1.683128
$L3p$	-0.812867		

Nous obtenons un modèle à 7 variables pour lequel le $R^2 = 0.5466$, $R^2_{ajusté} = 0.5111$ et $MSE = 0.6181$.

LASSO

Nous avons ensuite appliqué la sélection de variable LASSO et avons obtenu les résultats suivants :

Variable conservée	Coefficient	Variable conservée	Coefficient
$nR05$	0.198304	$O-056$	-0.0337653
$VE2sign\_D$	30.0619	$N-067$	0.293580
$ChiA\_X$	-167.903	$Cl-090$	-0.552191
$SpMax\_AEA(dm)$	1.688638	$SssO$	-0.00580435
$L3u$	-0.1204524	$NaaCH$	0.107294
$G2u$	2.98094	$CATS2D\_03\_DL$	0.106441
$E1u$	3.783428	$T(F..F)$	0.0368097
$G1m$	7.963685	$B03[F-Cl]$	-0.616769
$E2m$	0.0658104	$B05[S-F]$	-0.005773189
$L3p$	-0.8224145	$F03[F-Cl]$	-0.00008586
$G1p$	-6.836353	$F04[F-F]$	0.501492
$G3i$	1.197747	$F05[N-F]$	0.0861215
Gm	4.469015	$F07[C-O]$	-0.0927797
$H6i$	-0.695949	$G(S..F)$	0.00203639
$nCXr$	-0.485989	$TPSA(Tot)$	-0.00372277
$nHBonds$	-0.349777	$Neoplastic-80$	-0.2508676
$SpMAD\_X$	2.008741		

Nous obtenons un modèle à 33 variables pour lequel le $MSE = 0.2402$ et $MSELASSO = 10.9306$.

Les résultats du Stepwise sont peu satisfaisant, le procédure ne conserve que 7 variables ce qui est en accord avec ce que nous voulons, cependant le MSE est élevé et le R^2 est assez bas. Les résultats du LASSO nous donnent un excellent MSE cependant le modèle à 33 variables ce qui est beaucoup trop pour nous.

3.4.2 Familles du spécialiste

Comme expliqué précédemment, nous allons appliquer la méthode Stepwise, et non le LASSO, après avoir sélectionné des variables dans nos familles grâce à des ACP. Nous avons choisit de sélectionner des variables issus des deux premiers axes principaux de l'ACP. Ce premier épurage nous laisse encore 58 variables explicatives. Nous obtenons les résultats suivants : $R^2 = 0.6072$, $R^2_{ajusté} = 0.5636$, et $MSE = 0.58401$, avec 12 variables. Vous trouverez un récapitulatif des variables sélectionnées et de leur coefficients dans le tableau suivant :

Variable conservée	Coefficient	Variable conservée	Coefficient
Intercept	-139.3527	$SpMax5\_Bh\_p\_$	10.500615
$ON1V$	2.5551	$P\_VSA\_s\_4$	0.0462
PCR	-19.8819	$SpAD\_EA\_ed\_$	-0.1033
$X1Mad$	0.9716	$QXXm$	0.01333
$ISIZ$	0.3653	Wi_G_D	-0.0604
$ATSC4s$	0.0125	$SpMaxA\_G\_D$	101.8313

3.4.3 Familles issues de classification

CHA

Nous avons testé de prendre différents seuils d'inertie pour le deuxième axe de l'ACP, à partir duquel on garde une variable ayant une grande contribution sur le deuxième axe. Les résultats nous montrent que prendre toujours en compte le deuxième axe nous donne des meilleurs MSE, dans le cas de la classification hiérarchique. La Figure 3.18 représente les MSE associés à différents choix de nombre de classes, dans le cas de la sélection par ACP, avec un seuil d'inertie de 25%.

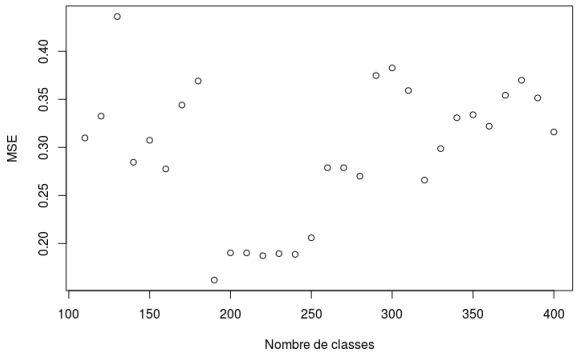

FIGURE 3.18 – MSE selon le nombre de groupes

Ici on pourrait décider de choisir soit un nombre de groupe égal à 190, soit de plutôt choisir 190 ou 250 groupes. En effet, en relançant la fonction plusieurs fois, on remarque que les résultats

changent légèrement, c'est-à-dire que le minimum est atteint parfois à 190 la plupart du temps, mais les "voisins" gauches de ce nombre de groupes sont beaucoup plus élevés. Lorsque l'on choisit de répartir les données entre 190 et 250 groupes, le MSE reste très petit, et plutôt stable. Cependant, le MSE n'est pas le seul critère à prendre en comtpte, car le nombre de variables sélectionnées est également important. Nous choisissons donc de retenir 250 groupes, ce qui nous donne une erreur quadratique moyenne qui varie autour de 0.3916128, associé à un modèle contenant 13 variables. En prenant d'autres seuils d'inertie, les résultats sont similaires, mais le nombre de variables sélectionnées augmente légèrement (environ 18).

K-means

La méthode de classification par l'algorithme K-means donne des résultats plus aléatoires que la classification hiérarchique ascendante. En effet, l'algorithme démarre en choisissant k centres de classes aléatoirement.

La Figure 3.19 et la Figure 3.20 représentent les MSE en fonction du nombre de groupes de deux exécutions du programme, en choisissant la sélection de variables par ACP.

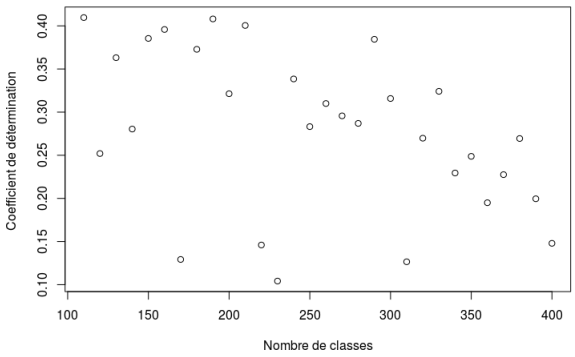

FIGURE 3.19 – MSE selon le nombre de groupes

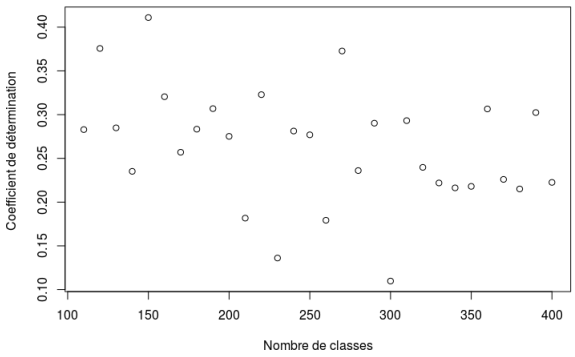

FIGURE 3.20 – MSE selon le nombre de groupes

On remarque bien que les deux graphiques différent complètement. On pourrait choisir 220 ou 230 au vus du premier schéma, ce qui serait meilleur que les résultats obtenus avec la classification hiérarchique. Cependant, le deuxième graphique nous donne des résultats complètement différents, le schéma n'a pas de tendance, ce qui nous empêche d'avoir des résultats stables. Avec un seuil d'inertie pour le deuxième axe principale très petit, le modèle obtenu est très bon avec des MSE très petits, mais le modèle contient plus de trente variables, ce que l'on juge trop important au vu de nos données. Si l'on augmente le seuil à 25%, les résultats sont meilleurs, mais restent moins bons que ceux trouvés précédemment avec la classification hiérarchique.

Ensuite, nous avons essayé d'appliquer la méthode K-means en lui donnant des centres initiaux, obtenus à partir de la méthode CHA, ce qui permettrait de retirer une partie de l'aspect aléatoire. Les classes changent légèrement. Cependant, quand on décide d'appliquer ensuite la méthode de sélection de variables par ACP, puis le LASSO, on obtient des résultats inférieures à ceux obtenus avec la CHA, c'est-à-dire que le MSE reste stable mais le nombre de variables sélectionnées augmente légèrement.

De ces résultats, on peut conclure que la classification hiérarchique donne de meilleurs résultats, et donc on choisit de retenir le modèle suivant dans ce cas.

Variable conservée	Coefficient	Variable conservée	Coefficient
$D/Dtr05$	0.00311175287	$RDF050s$	0.0004690488
$ATSC4s$	0.0004127945	$RDF085s$	0.0045056319
$P\_VSA\_MR\_1$	0.0018815075	$RDF125s$	0.0024216779
$P\_VSA\_e\_2$	0.0017707235	$RDF130s$	0.0025067558
$P\_VSA\_ppp\_ar$	0.0078578676	$T(N..O)$	-0.0078879045
$RDF150v$	-0.1657414882	$T(O..O)$	-0.004705152
$G(F..F)$	0.0023099474		

Le MSE associé à ce modèle vaut 0.3916128, avec 13 variables sélectionnées.

3.5 Récapitulatif

Méthode	Famille	Sélection	R^2	$R^2_{ajusté}$	MSE	Nb de var.
Table entière : Régression sur les 12 premières comp. princ			0.3856	0.3104	0.7341	12
Table entière : LASSO					0.2816	37
Table entière : Stepwise			0.8613	0.8305	0.36398	21
Corrélation	Nos familles	Stepwise	0.6705	0.6339	0.5348	11
Corrélation	Nos familles	LASSO			0.3606	19
Corrélation	Spécialiste	Stepwise	0.5455	0.5050	0.6219	10
Corrélation	Familles issues de classification	LASSO			0.3421632	13
Composantes principales	Nos familles	Stepwise	0.4937	0.4645	0.6468	7
Composantes principales	Nos familles	LASSO			0.4892	9
Composantes principales	Spécialiste	Stepwise	0.5020	0.4353	0.66432	27
Composantes principales	Familles issues de classification	LASSO			0.3969109	18
Variables à forte contrib.	Nos familles	Stepwise	0.5466	0.5111	0.6181	7
Variables à forte contrib.	Nos familles	LASSO			0.2402	33
Variables à forte contrib.	Spécialiste	Stepwise	0.6072	0.5636	0.58401	12
Variables à forte contrib.	Familles issues de classification	LASSO			0.3916128	13

Conclusion

Notre objectif était de constituer un modèle permettant d'expliquer au mieux l'activité chimique du VHC, pour pouvoir la prédire en fonction de peu de paramètres. Pour cela, nous disposions d'une table de données de 5255 variables explicatives quantitatives, que nous avons choisi de traiter avec diverses méthodes de sélection de variables statistiques. La première partie de notre travail a consisté à réunir la totalité des variables en familles, pour réduire la masse de données à traiter, et car certaines variables admettent des sens chimiques proches. De ce fait, nous avons utilisé différentes méthodes pour créer ces familles : tout d'abord, une méthode basée sur les noms, puis nous avons utilisé les méthodes de classification. Après ce travail, le spécialiste nous a transmis ses familles de variables, ce qui a nous a permis d'évaluer la pertinence de notre travail.

Dans le but de ne pas avoir d'information redondante dans le modèle final, nous avons choisi de sélectionner quelques représentants par famille. Pour ce faire, nous avons mis en place différentes méthodes de choix des représentants. Dans un premier temps, nous avons sélectionné dans chaque famille la variable la plus corrélée avec l'activité chimique du VHC. Ensuite, nous avons créé de nouvelles variables représentatives de leur famille, qui correspondent aux composantes principales de l'Analyse en Composantes Principales de chaque famille. Enfin, nous avons pensé à écrémer les familles de variables grâce à l'ACP, en choisissant les variables ayant la plus grande contribution sur le premier et deuxième axe principale le cas échéant.

Après ce premier tri, nous avons appliqué deux méthodes statistiques de sélection de variables : la procédure pas à pas Stepwise, et la méthode de régression pénalisée LASSO.

Une fois toutes ces opérations effectuées, nous avons obtenu un certain nombre de modèle, que nous avons récapitulé dans la Partie 3.5. Nous avons donc discuté avec le spécialiste sur la manière de choisir le modèle le plus pertinent. En effet, il faut faire un compromis entre un MSE bas et un faible nombre de variables. Selon lui, en général, l'objectif est d'obtenir un R^2 supérieur à 0.6, ce qui correspond à un MSE inférieur à 0.5. De plus, selon certaines heuristiques, il faut garder une variable explicative pour 10 individus, donc dans notre cas, comme nous avons 115 individus, nous devrions conserver entre 8 et 14 variables.

Le modèle qui nous apparaît le plus pertinent est celui issu des familles créées grâce à la classification, dans lesquelles nous avons sélectionné un représentant par corrélation et auxquels nous avons appliqué une méthode de LASSO. Ce modèle est associé à un MSE valant 0.3421632, pour 13 variables explicatives. Le voici :

Variables	Coefficient	Variables	Coefficient
MATS5p	3.5511205512	$P\_VSA\_MR\_3$	-0.0012204074
H4s	0.0613845058	RDF150i	-0.0304528284
D/Dtr05	0.0022817005	$CATS3D\_13\_AL$	0.0214841533
TDB05i	0.9466104467	RDF085s	0.0009172892
$VE3sign\_Dz(i)$	0.2401428697	RDF125s	0.0037769427
RDF130s	0.0033913200	$T(N..O)$	-0.0047019691
$TPSA(NO)$	-0.0073981575		

Nous avons voulu comparer ces résultats avec ceux que le spécialiste nous a transmis. Ce dernier obtient un modèle à 21 variables pour un MSE à 0,3487149. En comparant les variables sélectionnées dans les deux modèles, nous nous sommes rendus compte que notre modèle contenait beaucoup de variables issues des mêmes familles que celles du spécialiste, ce qui est positif.

Nous avons poussé l'analyse de notre modèle final pour évaluer sa convenabilité, en effectuant une analyse des résidus. En effet, dans l'idéal, ceux-ci devraient suivre une loi normale de moyenne 0. Dans notre cas, nous trouvons une moyenne de -6.881119e-16, ce qui est très proche de 0. Nous avons effectué deux tests de normalité des résidus : le test de Kolmogorov-Smirnoff, et le test de Shapiro-Wilk. Nous ne rejetons pas l'hypothèse de normalité des résidus dans les deux cas, car les p-valeurs sont grandes.

```
> shapiro.test(Residus)

        Shapiro-Wilk normality test

data:  Residus
W = 0.98728, p-value = 0.3805
```

```
> ks.test(Residus,dnorm(0,sd(Residus)))

        Two-sample Kolmogorov-Smirnov test

data:  Residus and dnorm(0, sd(Residus))
D = 0.73874, p-value = 0.5357
alternative hypothesis: two-sided
```

Sur les Figures 3.21 et 3.22, vous trouverez un histogramme et un diagramme quantiles-quantiles des résidus. Cela confirme l'hypothèse de normalité des résidus.

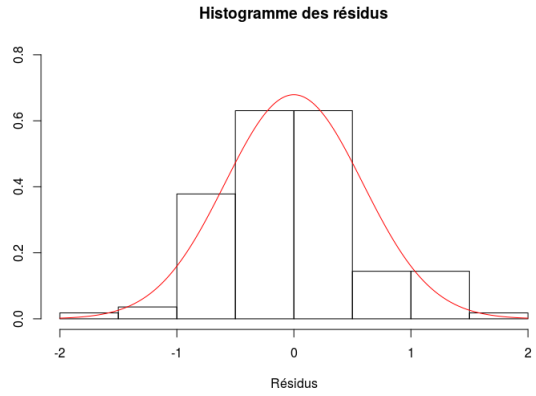

FIGURE 3.21 – Histogramme des résidus

FIGURE 3.22 – qqPlot des résidus

Cependant, sur la Figure 3.23, on voit clairement une tendance dans les résidus. En effet, pour des petites de l'activité chimique, le modèle a tendance a surestimer, et pour des grandes valeurs, le modèle a tendance a sous-estimer.

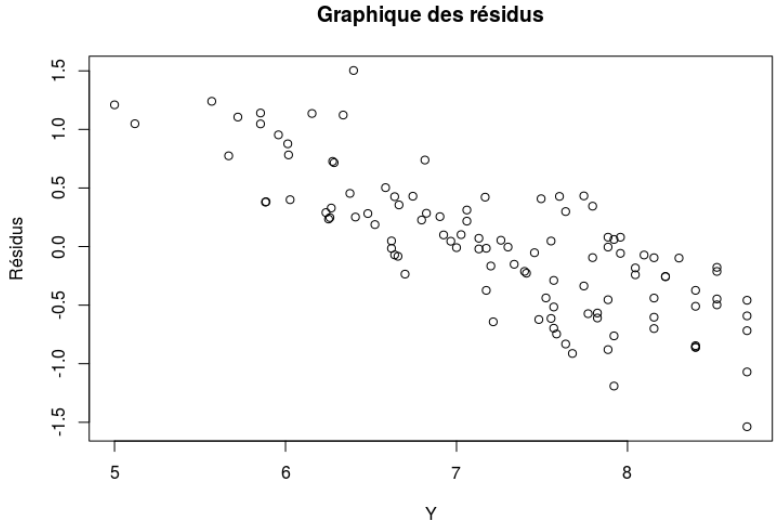

FIGURE 3.23 – Graphique de Y en fonction des résidus

On en conclut que notre modèle final est satisfaisant car il explique bien l'activité chimique, et que les résidus semblent bons. Cependant, il reste un biais au niveau des résidus. Malgré ceci, nous avons réussir à obtenir de meilleurs résultats que ceux obtenus sur les données brutes, et similaires à ceux obtenus par le spécialiste avec significativement moins de variables contenues dans le modèle.

La problématique de notre Travail Encadré de Recherche était de mettre en place différentes méthodes statistiques, et de montrer l'efficacité des statistiques dans un contexte de Big Data. Comme nous avons pu le constater, le modèle final est issu d'une combinaison de ces méthodes (Classification et LASSO), et est meilleur que les modèles obtenus par d'autres méthodes, qui font moins appel aux statistiques. C'est le cas par exemple de la création des familles par les noms qui relève plus de l'intuition que des statistiques, ou encore du travail effectué sur les données brutes.

Bibliographie

[1] Peter Bühlmann and Sara van de Geer. *Statistics for High-Dimensional Data.* Springer, 2011.

[2] N. Draper and H. Smith. *Applied Regression Analysis.* New York : John Wiley and Sons, Inc., 1981.

[3] Jerome Friedman, Trevor Hastie, Robert Tibshirani, Noah Simon, Balasubramanian Narasimhan, and Junyang Qian. *glmnet : Lasso and Elastic-Net Regularized Generalized Linear Models.* CRAN, 2018.

[4] Trevor Hastie, Robert Tibshirani, and Martin Wainwright. *Statistical Learning with Sparsity.* CRC Press, Taylor & Francis Group, 2015.

[5] Anisse Ismaili and Pierre Gaillard. Le lasso, ou comment choisir parmi un grand nombre de variables à l'aide de peu d'observations. *Exposé de maîtrise*, 2009.

[6] Philippe Mahey. *Cours Optimisation Convexe.* Université Clermont Auvergne.

[7] Daoud Ounaissi. *Méthodes quasi-Monte Carlo et Monte Carlo : Application aux calculs des estimateurs Lasso et Lasso bayésien.* PhD thesis, École doctorale Sciences pour l'Ingénieur de l'Université de Lille, 2016.

[8] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B*, 58(1) :267–288, 1996.

Annexe

```
RechercheNA <- function(Table){
  #On stocke et on retourne la position des "NA"
  Positioni=vector()
  Positionj=vector()
  k=1
  for(i in 1:dim(Table)[1]){
    for(j in 2:dim(Table)[2]){
      if(Table[i,j]==’na’){ Positioni[k] <- i; Positionj[k] <- j; k <- k+1; }
    }
  }
  Position=matrix(nrow=length(Positioni), ncol=2)
  Position[,1] <- Positioni; Position[,2] <- Positionj
  return(Position)
}
```

```
VecteurVarParFam <- function(famille){
  VarParFam=vector()
  #On parcours tous les éléments de chaque famille
  for(j in 1:dim(famille)[2]){
    for(i in 1:(dim(famille)[1]-1)){
      if( !is.na(famille[i,j]) && is.na(famille[i+1,j]) ){
        #On cherche la place du dernier membre de la famille et on la stocke
        VarParFam[j]=i
      }
    }
    if(!is.na(famille[dim(famille)[1],j])){
      #Si le dernier membre est aussi le dernier élément de la matrice, on le stocke
      VarParFam[j]=dim(famille)[1]
    }
  }
  return(VarParFam)
}
```

```
CreationFamille <- function(Table){
  L=names(Table)  #On stocke le nom des variables
  nbvar=length(L)
  ######################   Famille avec les trois dernières lettres   ############################
  famille=array(dim=c(nbvar,nbvar))   #On créé un tableau pour ranger en colonne les membres
                                      #d'une même famille
  famille[1,1]=L[1]
  l=1; f=1      #f est l'indicateur de famille et l l'indicateur de position
  for(i in 2:nbvar){
    #On compare deux à deux les noms de variables
    if( substring(L[i-1], first = (nchar(L[i-1])-2), last = nchar(L[i-1])) ==
                                substring(L[i], first = (nchar(L[i])-2), last = nchar(L[i])) ){
      #Si les trois dernières lettres son les mêmes, on stocke le nom de la variable dans
      #la colonne (ie famille) en cours de création
      l <- l+1
      famille[l,f]=L[i]
    }
    else
    {
      #Sinon, on change de colonne (ie famille) et on commence à la remplir
      f <- f+1
      l <- 1
      famille[l,f]=L[i]
    }
  }

  #On calcule le nombre de familles crées pour pouvoir créer la table de l'étape suivante
  #qui ne soit pas trop grande
  dimf1=0
  for(i in 2:nbvar){
    if( !(is.na(famille[1,i])) && is.na(famille[1,i+1]) ){
      dimf1=i
    }
  }
  dimf1

  #On retire les cases vides
  famille=famille[-c(dimf1+1:nbvar),-c(dimf1+1:nbvar)]
  dim(famille)

  ######################   Famille avec les trois premières lettres   ############################
  famille2=array(dim=c(dimf1,dimf1))  #On crée le tableau pour la seconde étape
  famille2[1,1]=famille[1,1]          #On le pré-rempli avec le tableau précédent
  l=1; f=1    #on initialise les indicateur de position et famille
  for(i in 2:dimf1){
    #On regarde si l'élément à classer et son voisin n'ont pas déjà été regroupés
    if( is.na(famille[2,i-1]) && is.na(famille[2,i])){
      #Si les deux éléments ne sont pas regroupés, on compare leurs noms
      if( substring(famille[1,i-1], first = 1, last = 3) ==
                                              substring(famille[1,i], first = 1, last = 3) ){
        #Si leurs trois premières lettres sont communes, ont les regroupes
        l <- l+1
        famille2[l,f]=famille[1,i]
      }
      else
      {
        #Sinon on change de colonne (ie famille) et on commence à la remplir
        f <- f+1
        l <- 1
        famille2[l,f]=famille[1,i]
```

```
      }
    }
    else
    {
      #Si l'élément est déjà regroupé, on recopie la colonne (ie famille) crée à l'étape précédente
      #et on change de famille
      f <- f+1
      l <- 1
      famille2[,f]=famille[,i]
    }
  }

  #On calcule le nombre de familles crées pour pouvoir créer la table de l'étape suivante
  #qui ne soit pas trop grande
  dimf2=0
  for(i in 2:dimf1){
    if( !(is.na(famille2[1,i])) && is.na(famille2[1,i+1]) ){
      dimf2=i
    }
  }
  dimf2

  #On retire les cases vides
  famille2=famille2[,-c(dimf2+1:dimf1)]


  ### On abrége les étapes suivantes consistant à regrouper les variables pour lesquelles les
  ### deux premières lettres sont communes, puis les deux dernières, puis la première puis
  ### la dernière. En effet, le code de ces étapes est similaire à celui présenté ci-dessus


  VarParFam <- VecteurVarParFam(famille6)    #On calcule le nombre de membres dans les
                                             #différentes familles finales
  m=max(VarParFam)                           #On trouve le nombre maximale de membre par famille
  famille6 <- famille6[-c(m+1:dim(famille6)[1]),]    #On retire les colonnes excédentaires
  return(famille6)
}



QuelleFamille <- function(nom, Famille){
#Pour un nom de variable donnné, cette fonction le localise dans la table Famille
#Si le nom n'est pas trouvé, la fonction vaut NA
  for(i in 1:dim(Famille)[1]){
    for(j in 1:dim(Famille)[2]){
      if(!is.na(Famille[i,j])){if(nom == Famille[i,j]){ return(c(j,i)) }}
    }
  }
}



ElectionChefParCorr <- function(famille, Table){
#On élit dans chaque famille un représentant ayant une forte corrélation avec l'activité
  ChefDeFamille = vector()   #On créé le vecteur qui va contenir le nom des variables élues
  VarParFam <- VecteurVarParFam(famille)
  for(j in 1:dim(famille)[2]){
    ChefDeFamille[j] <- famille[ 1 , j ]   #On élit provisoirement le premier membre comme chef
    if(VarParFam[j] > 1){                  #Dans chaque famille où il y a plus d'un individu,
      for(i in 1:VarParFam[j]){            #On cherche si un autre membre présente une
                                           #corrélation plus grande que le chef provisoire
        if( cor(x = Table[,1] , y = Table[ famille[ i , j ] ] )  >
```

```
                          cor(x = Table[,1] , y = Table[ ChefDeFamille[j] ] ) ){
          ChefDeFamille[j] <- famille[ i , j ]              #Si c'est le cas, il est élu chef
        }
      }
    }
  }
  return(ChefDeFamille)
}


SelectionParACP <- function(Famille, Table, Seuil2){
#On selectionne des variables selon leur corrélation avec la première composante principale
  Famille <- as.matrix(Famille)
  M=matrix(nrow = dim(Table)[1], ncol = 1)      #On crée les éléments nécessaire au stockage
  Noms=vector()                                 #des variables sélectionnées
  VarParFam <- VecteurVarParFam(Famille)
  NbFam=dim(Famille)[2]
  for(i in 1:NbFam){    #Dans chaque famille contenant au moins 2 membres on réalise une ACP
    if(VarParFam[i] > 1){
      ACP <- PCA( Table[Famille[ c(1:VarParFam[i]), i ]] , scale.unit=TRUE, graph=FALSE)
      Contrib1 = ACP$var$contrib[,1]             #On trouve la variable ayant la plus
      m1=which.max(Contrib1)                     #grande contribution pour la première
      M=cbind(M,Table[, Famille[ m1 , i ] ])     #composante et on la stocke
      Noms <- c(Noms , Famille[ m1 , i ])
      if(ACP$eig[2,2] > Seuil2){                 #Si le pourcentage d'inertie expliquée par
        Contrib2=ACP$var$contrib[,2]             #la seconde composante est plus grande que le
        m2=which.max(Contrib2)                   #seuil sélectionnéon stocke la variable ayant la
        if(m2 == m1){                            #plus grande contribution pour cette composante
          m3=which.max(Contrib2[-m2])            #si celle-ci est différente de la première sinon
          if(m3 >= m2){                          #on prend celle ayant la seconde plus grande
            m3 <- m3+1                           #contribution
          }
          M=cbind(M,Table[, Famille[ m3 , i ] ])
          Noms <- c(Noms , Famille[ m3 , i ])
        }
        else
        {
          M=cbind(M,Table[, Famille[ m2 , i ] ])
          Noms <- c(Noms , Famille[ m2 , i ])
        }

      }
    }
    else
    {
      M=cbind(M, Table[ Famille[ 1 , i ] ])      #Si la famille ne comporte qu'une variable on
      Noms <- c(Noms , Famille[ 1 , i ])         #stocke cette variable
    }
  }
  M=M[,-1]
  names(M) <- Noms
  return(M)
}


CompPrincFamille <- function(Famille, Table, Seuil2){
#On extrait les composantes principales des différentes familles
  Famille <- as.matrix(Famille)
  M=matrix(nrow = dim(Table)[1], ncol = 1)      #On crée les éléments nécessaire au stockage
  Noms=vector()                                 #des variables sélectionnées
```

```
  VarParFam <- VecteurVarParFam(Famille)
  NbFam=dim(Famille)[2]
  for(i in 1:NbFam){        #Dans chaque famille contenant au moins 2 membres on réalise une ACP
    if(VarParFam[i] > 1){
      ACP <- PCA( Table[Famille[ c(1:VarParFam[i]), i ]] , scale.unit=TRUE, graph=FALSE)
      M=cbind(M,ACP$ind$coord[,1])
      Noms <- c(Noms , Famille[ 1 , i ])            #On stocke la première composante principale
      if(ACP$eig[2,2] > Seuil2){                    #de chaque famille et si le pourcentage d'inertie
        M=cbind(M,ACP$ind$coord[,2])                #expliquée par la seconde composante est plus
        Noms <- c(Noms , Famille[ 2 , i ])          #grande que le seuil sélectionné on la stocke aussi
      }
    }
    else
    {
      M=cbind(M, Table[ Famille[ 1 , i ] ])         #Si la famille ne comporte qu'une variable on
      Noms <- c(Noms , Famille[ 1 , i ])            #stocke cette variable
    }
  }
  M=M[,-1]
  names(M) <- Noms
  return(M)
}



LASSO <- function(M){
#Les fonction cv.glmnet et glmnet proviennent du package GLMNET
  x=model.matrix(Activity~.-1,data=M)   #On formate la matrice pour le LASSO
  y=as.vector(as.matrix(M[,1]))
  lambda = vector()
  for(i in 1:30){
    LASSO_model_cv=cv.glmnet(x,y, alpha =1)     #On réalise 30 validations croisées
    plot(LASSO_model_cv)
    lambda[i]=LASSO_model_cv$lambda.min         #On stocke les lambda obtenus
  }
  hist(lambda, breaks = 50)
  LambdaOpt=median(lambda)                      #On choisit pour LambdaOpt le lambda médian obtenu
  LASSOFinal=glmnet(x, y ,alpha = 1, lambda=LambdaOpt)   #On lance le LASSO pour le Lambda optimal choisit

  Pred=predict(LASSOFinal ,newx=x)              #On calcule les indicateurs d'ajustement
  MSE = mean((Pred-y)**2)
  MSEBis = sum((Pred-y)**2)/(2*length(y)) + LambdaOpt * sum(abs(LASSOFinal$beta))
  LASSOFinal$Qualite$MSE <- MSE
  LASSOFinal$Qualite$MSELASSO <- MSEBis

  s=LASSOFinal$beta                             #On stocke spécifiquement les variables ayant été
  FacteursSelecLASSO=vector()                   #sélectionnées par LASSO
  Num=vector()
  Noms=vector()
  for (i in 1:length(s)){
    if (s[i]!=0){
      Num=c(Num,i)
      FacteursSelecLASSO=c(FacteursSelecLASSO,s[i])
      Noms = c(Noms, names(M)[i+1])
    }
  }
  Num
  FacteursSelecLASSO
  Garde <- cbind(FacteursSelecLASSO, Num)
  rownames(Garde) <- Noms
  LASSOFinal$Selected <- Garde
```

```
  return(LASSOFinal)
}



Famille_CHA <-function(HCV,nb,methode){
  HCVtr=t(HCV[,-1]) #On transpose la matrice car on souhaite appliquer la classification aux
  H=hclust(dist(HCVtr,method="euclidean"),method =methode)
  C=cutree(H,k=nb)
  C=as.numeric(C)
  N=matrix(nrow=3192,ncol=nb)
  for (i in 1:3192){
    j=1
    while (!is.na(N[j,C[i]])){
      j<-j+1
    }
    N[j,C[i]]=names(HCV[,i+1])
  }
  return (N)
}

Famille_Kmeans <-function(HCV,G){  #G est le nombre de groupes que l'on souhaite obtenir
                                   #ou une matrice contenant des centres de gravité
  K=kmeans(t(HCV[,-1]),G)          #On retire "Activity", et on transpose la table pour
  if (is.numeric(G)){p=G} else {p=dim(C)[2]}
  M=matrix(nrow=dim(HCV[,-1])[2],ncol=G)
  for (i in 1:dim(HCV[,-1])[2]){
    j=1
    while (!is.na(M[j,K$cluster[i]])){
      j<-j+1
    }
    M[j,K$cluster[i]]=names(HCV[,i+1])
  }

  return(M)
}

Choixnbgroupes <- function(HCV,selectionvar,seuil,selectionclass){
  A=matrix(nrow=2,ncol=30)
  if (selection==1){
    for (i in 1:30){
      G=100+10*i    #On applique la méthode pour un nombre de groupes entre 110 et 400.
      if (selectionclass==1) {M=Famille_Kmeans(HCV,G)} else {M=Famille_CHA(HCV,G,"ward.D")}
      VarACP <- SelectionParACP(M, HCV, seuil)
      HCV1=HCV[c("Activity",names(VarACP))]
      L=LASSO(HCV1)
      A[1,i]=G
      A[2,i]=L$Qualite$MSE
      A[3,i]=length(L$Selected[,2])
    }
  }
  else if (selection==2) {

    for (i in 1:30){
      G=100+10*i
      if (selectionclass==1) {M=Famille_Kmeans(HCV,G)} else {M=Famille_CHA(HCV,G,"ward.D")}
      VarCorr <- ElectionChefParCorr(M,HCV)
      HCV1=HCV[c("Activity",VarCorr)]
      L=LASSO(HCV1)
      A[1,i]=G
      A[2,i]=L$Qualite$MSE
```

```
      A[3,i]=length(L$Selected[,2])
    }
  }
  else {
    for (i in 1:30){
      G=100+10*i
      if (selectionclass==1) {M=Famille_Kmeans(HCV,G)} else {M=Famille_CHA(HCV,G,"ward.D")}
      VarACPbis <- ExtractionFacteursPrincipauxParFamille(M, HCV, seuil)
      HCV1=cbind(HCV["Activity"],VarACPbis)
      L=LASSO(HCV1)
      A[1,i]=G
      A[2,i]=L$Qualite$MSE
      A[3,i]=length(L$Selected[,2])
    }
  }
  return (A)
}


VarianceNulle <- function(Table){
  NoVar=vector()
  for (i in 1:dim(Table)[2]){ if (var(Table[,i]) == 0){NoVar=c(NoVar,i)}}
  #On recherche les variables sans variances
  return(NoVar)
}


FamilleSpecialiste <- function(Table){
  M <- matrix(nrow=nrow(Table), ncol=1)
  M[1,1] <- as.character(Table[1,1])
  Famille <- as.character(Table[1,3])
  f<-1          #On initialise l'indicateur de famille à 1
  j<-1          #On initialise l'indicateur de position à 1
  for(i in 2:dim(Table)[1]){
    if(as.character(Table[i,3]) == Famille){
      j <- j+1          #Si un élement est dans la même famille que le premier élément de
      M[j,f] <- as.character(Table[i,1])    #cette famille alors on l'ajoute à la famille
    }
    else{               #Si un élement n'est pas dans la même famille, alors on en créé une nouvelle
      Famille <- as.character(Table[i,3])
      M <- cbind(M,matrix(nrow=nrow(Table), ncol=1))
      f<-f+1            #L'indicateur de famille est incrémenté
      j<-1              #L'indicateur de position est réinitialisé
      M[j,f] <- as.character(Table[i,1])
    }
  }
  VarParFam <- VecteurVarParFam(M)
  m=max(VarParFam)
  M <- M[-c(m+1:dim(M)[1]),]    #On retire les lignes vides
  return(M)
}
```